A morphing approach for kidney dynamic modeling From 3D reconstruction to motion simulation

Valentin Leonardi
LSIS, UMR CNRS 7296
Campus de Luminy
13288 Marseille cedex 9,
France
valentin.leonardi@univ-amu.fr

Jean-Luc Mari
LSIS, UMR CNRS 7296
Campus de Luminy
13288 Marseille cedex 9,
France
jean-luc.mari@univ-amu.fr

Vincent Vidal
L2PTV, EA 4264
CERIMED
13385 Marseille cedex 5,
France
vincent.vidal@ap-hm.fr

Marc Daniel
LSIS, UMR CNRS 7296
Campus de Luminy
13288 Marseille cedex 9,
France
marc.daniel@univ-amu.fr

ABSTRACT

Motion simulation of an organ can be useful in some cases like organ study, surgery aid or tumor destruction. When using a non-invasive way of tumor destruction through transcutaneous transmition of waves, it is primordial to keep the wave beam focused on the tumor. When the tumor is not in movement, such a task is trivial. But when the tumor is located in a moving organ like the kidney, motion simulation is necessary. We present here an original method to obtain the kidney motion simulation: this is done using a mesh morphing (we consider the kidney has already been segmented and reconstructed for three different phases of the respiratory cycle). Such an approach allows a smooth transition between the different kidney models, resulting in a motion simulation. Thus, the method is purely geometric and does not need any kind of markers or tracking device. It gives directly a full 3D simulation and models are animated in real time. Finally, our approach is automatic and fast, so that it can easily be used in a medical environment.

Keywords: Geometrical modeling, organ motion simulation, kidney modeling, mesh morphing

1 INTRODUCTION

Tumors can be treated by low-invasive approaches. The goal is to minimize interactions between the surrounding environment and the patient in order to limit the consequences of surgery (incision treatment, convalescence) and their possible complications (nosocomial infections). Kidney tumors can be treated by radiofrequency. Radiofrequency is a low-invasive, non-surgical percutaneous heat treatment. The principle is to locate the tumor through CT scan, and insert a radiofrequency electrode in its center. An electric current is then delivered, in order to destroy the tumor. However, there is a chance of cancerous cell displacement when removing the electrode.

The KiTT project (for **Ki**dney **T**umor **T**racking, of which we take part) is fully involved in the low-invasive protocol. Its goal is to create a *totally non-invasive* new approach by transmitting radiofrequency waves, in a transcutaneous way until tumor eradication. The main difficulty is to keep the wave beam continuously focused on the tumor while the kidney is deformed and moves because of the respiratory cycle. A kidney (and a tumor) *tracking* is therefore necessary. Before this organ tracking stage, we need to obtain a solid 3D model of it. This work is described in our previous paper [LMVD11].

What we propose here is a new method which has two goals: the first one is the motion and deformation visualization of an organ (the kidney in this case) under the influence of natural breathing. The second goal results directly from the first one and is the tracking of a part of this organ: its tumor. The originality of this method is that it uses a fully geometric approach: mesh morphing. Thus it is fast and only needs three models corresponding to three breathing phases: inhale phase (when volume of air in lungs is maximal), exhale phase (when volume of air in lungs is minimal) and the middle phase between the two previous ones, which we refer as the mid-cycle phase. Moreover, the results obtained are fully geometric; the output is an animated 3D model. Thereby general motion and all deformations can be studied at once where some methods only offer the possibility of a 2D visualization. Finally, as the tumor is also animated, it is possible to know its position at any time.

Section 2 of this article deals with the previous work in organ tracking and mesh morphing. Section 3 introduces the general process of our method: a brief recall of the kidney reconstruction is done, then the algorithm used for mesh morphing is described in detail.

In section 4, we present the obtained results, their performances and we discuss them. Finally in section 5, we present the limits of our method, the possibilities to overcome them and the perspectives of future work.

2 RELATED WORK

2.1 Organ tracking

Organ tracking methods are either based on mathematical models which represent the respiratory cycle as a periodical function, or on empirical algorithms which predict future movements by observation and analysis of previous ones.

The most intuitive way to track an organ is to put a **marker** which is highly detectable by a classic medical imaging acquisition near this organ [NUG+08, NPSA07, OTW+05, SGB+00, SSK+00]. This formalism is also used in all-in-one robotic radiosurgery systems such as the Cyberknife [MCG+03]. This kind of method requires a surgical intervention which is not suitable for our problematic.

The following approaches assume the kidney has been segmented and reconstructed previously for two or more phases of the respiratory cycle. Most of the time, only two models are needed, but three [SBMG06] or even more [RMOZ01] are sometimes necessary. These extra acquisitions can be used to improve the precision of the organ deformation. In other cases, it is not an extra acquisition of the organ that is needed, but other kind of data essential to the method. Hostettler *et al.* [HNS+08] use the diaphragm movement in order to reflect it on the abdomen organs. In [SBMG06], air, tissues and lungs have to be segmented for three acquisitions in order to get an organ tracking.

Deformation fields are used to understand the motion of an organ. This field computes the deformations necessary to apply on a given source model M_s to deform it into a given target model M_t. The deformation field can be computed using several methods like Maximum Likelihood / Exceptation-Maximisation [RMK+05], least squares [SBMG06] or approaches based on Normalized Mutual Information [RSH+99]. Deformations can also be applied on a mesh through a deformable superquadratic in order to get the movement of an organ [BCA96].

Registration methods are also a good way to have an organ tracking. Nicolau *et al.* [NPSA07] use two acquisitions: on the first one, markers are used in order to get the position of the organ of interest. Then a second acquisition is done without these markers. By analyzing the difference of position of the spine for both acquisitions, the registration is performed using the minimization of the Extended Projective Point Criterion. In [RSH+99] two operations are done to compute the registration: affine transformation is used for global movements while Free Form Deformation is used for local movements. Two registration algorithms based on optical flow are implemented and accelerated using GPU programming in [NdSE+08] in order to perform an image-guided radiotherapy.

2.2 Principle of morphing

Mesh morphing is a method used to transform progressively a source model M_s into a target model M_t. The most usual method to perform a mesh morphing is to find a common vertex/edge/face network for both models in order to compute a metamesh M_m which contains the topology of M_s and M_t. The common network is obtained by mapping the mesh into arbitrary shapes, using different kind of mappings based on the resolution of a linear system. This approach was first used by Kent *et al.* in [KCP92], where both models are mapped onto the unit disk. In [ACOL00] Alexa *et al.* perform a mesh morphing where the interior is also considered. A 3D mesh is decomposed into a set of tetrahedrons and a 2D shape into a set of triangles. The interpolation of a tetrahedron is done using a rotation and a scale-shear with positive scaling matrices. In [GSL+98] both M_s and M_t are divided into an equal number of patches P. Each patch is then mapped so that the patch P_k of M_s is morphed into the patch P_k of M_t. This approach is close to the one used by [KSK00] since models are also divided into n arbitrary shapes. These shapes are mapped onto a polygon, where vertices of the border of a shape lie on the border of the polygon. The position of the remaining vertices is then computed by considering the shape as a spring-mass system at rest, the border vertices being the the fixed masses. In [LDSS99] M_s and M_t are simplified into M'_s and M'_t using the MAPS method [LSS+98]. Vertices for which vertex-vertex correspondences is already known are kept. M_s and M_t are finally mapped in order to compute the correspondences. Unfortunately, the previous approaches always need either user interaction (which can be very time consuming for some methods) or vertices correspondence between M_s and M_t prior to the mesh morphing. In both case, our constraints do not allow to spend a lot of time on cutting up a mesh manually or making vertex-vertex correspondences.

It is possible to fully automate a mesh morphing algorithm by using an automatic mesh cutting up. Indeed, such a process allows to separate a model into at least two different parts which can then be mapped. Several works can be found, although it is a difficult problem: some methods are based on the use of a single patch [KSK97], where others are related to graph

theory problem and aim at balancing the size of patches [EDD+95, KK99]. However, our models are close to each other, which does not justify such an advanced approach. Another way to have a complete automatic mesh morphing is to map models onto the unit sphere [KCP92]. Indeed, there is no need to divide the models anymore since they are homeomorphic to a sphere. On the other hand, the model has to be star-shaped, which is not the case of a kidney. Alexa [ACOL00] introduces a variant for sphere mapping: as for barycentric mapping, each vertex is placed at the centroid of its neighbors. Finally, a new approach for mesh morphing is done by Lee *et al.* in [YHM07]. The principle is to compute a constraints field C in order to deform M_s into M_t. C is then interpolated in order to determine a new constraints field C' for an intermediate model M_i between M_s and M_t.

3 GENERAL PROCESS

3.1 Overview

A general view of our entire workflow is presented on Figure 1. It shows how to get the kidney motion visualization from 3 sets of images. These sets result from three CT-scan or MRI acquisitions. First, the kidney and tumor are segmented for the three sets of images. Then, the organ is reconstructed in order to have three 3D models (called M_1, M_2 and M_3), each one corresponding to a precise breathing phase. Mesh morphing between M_1 and M_2 and between M_2 and M_3 is computed. Our mesh morphing approach is based on an automatic mesh cutting up, unit disk mapping and metamesh creation. Since models are relatively close to each other, they can be divided into only two patches. Moreover, the frontier between the two patches always has the same orientation on the kidney, which allows this step to be entirely automatic. Once the two patches are defined for M_s and M_t, a unit disk mapping is performed in order to overlay the two mappings of a corresponding patch of both models. We cannot use a sphere mapping here as kidney models are not star-shaped. Detections of mapped edge intersections and mapped vertices positions allow to create a metamesh which comprises the topology of both models. As an initial and a final position are known for all metamesh vertices, the metamesh is animated by linear interpolation. The alternation between metameshes coming from M_1 and M_2 and from M_2 and M_3 gives a full kidney animation from inhale to exhale phase, *i.e.* the whole respiratory cycle. The only part of our method done in real time and during the whole tumor destruction process is the metamesh vertices interpolation. Everything before this step is done once and for all and takes less than 2 minutes (from kidney segmentation to metamesh creation).

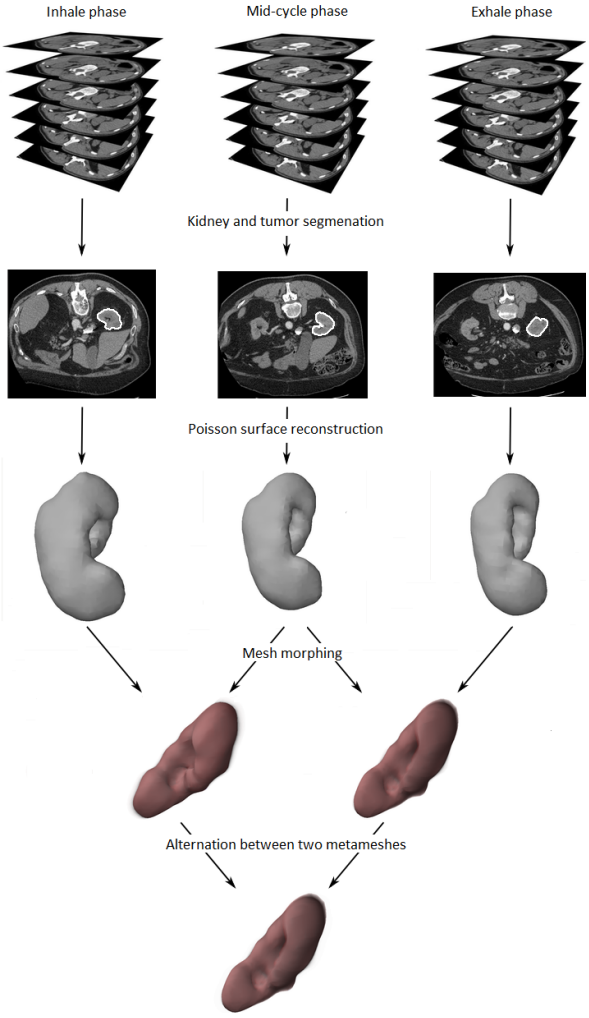

Figure 1: *Overview of our entire workflow: three sets of images resulting from a medical imaging acquisition for the inhale, mid-cycle and exhale phase is done (first line). The kidney and the tumor are segmented for every images of these three acquisitions (second line). The Poisson surface reconstruction is then applied to the point cloud extracted from the segmentation of each three different phases. We call the resulting models M_1, M_2 and M_3 (third line). Mesh morphing is computed between M_1 and M_2 and between M_2 and M_3. The results are two metameshes which allow a smooth transition between M_1 to M_2 and M_2 to M_3 (fourth line). By alternating the two metameshes, a full and smooth transition from M_1 to M_3 is possible, resulting in the kidney motion visualization (fifth line).*

3.2 Kidney reconstruction

A full description of the kidney reconstruction is detailed in [LMVD11]. The method used to get the kidney model is divided into two stages: the first one is the kidney segmentation from which a point cloud is extracted. The second stage consists in reconstructing

this point cloud in order to obtain a model.

A region growing approach is used in order to segment the kidney. Despite some methods exist to define automatically the seed needed for initialization, our approach uses a minor user-interaction and needs a single mouse click to define it. However, this is done only for one image (since the whole kidney is present in at least 60 slices). The region segmented in an image I_{k-1} is used to get the seed for the next image I_k: the weighted barycenter of the points defining the contour in I_{k-1} defines the seed for I_k. Results of this method are shown on Figure 2

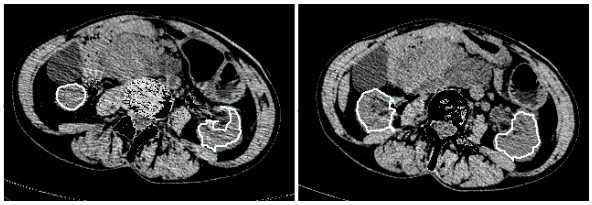

Figure 2: *Final results using our kidney segmentation approach for left and right kidney on two different slices.*

The point cloud extracted from the segmentation stage is reconstructed using the Poisson Surface Reconstruction [KBH06]. The principle of this algorithm is to define an indicator function χ peculiar to a model M, which is 0 for every point outside the model and 1 inside. Deducing χ directly from the oriented point cloud is the major problem in this case. The solution is to use the gradient of χ since the point cloud can be considered as samples of $\vec{\nabla}\chi$ (see Figure 3). Indeed, $\vec{\nabla}\chi$ is a vector field that is 0 almost everywhere except near the surface. A vector field $\vec{V}$ which is an approximation of $\vec{\nabla}\chi$ is found using the original normals. χ must now be deduced from $\vec{V}$, *i.e.* $\vec{\nabla}\chi = \vec{V}$. This is done by applying the divergence operator to express it as a Poisson equation: $\Delta\chi \equiv \nabla \cdot \vec{\nabla}\chi = \nabla \cdot \vec{V}$. The resolution of this equation is a well known problem (especially in physics) but will not be discussed here. The final reconstruction is then obtained from the extraction of an appropriated isosurface (see Figure 4).

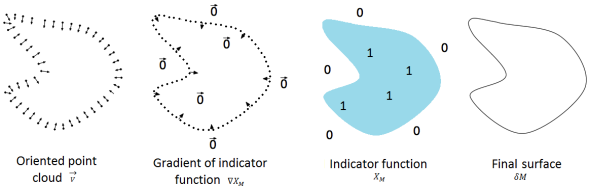

Figure 3: *Overview of the Poisson surface reconstruction.*

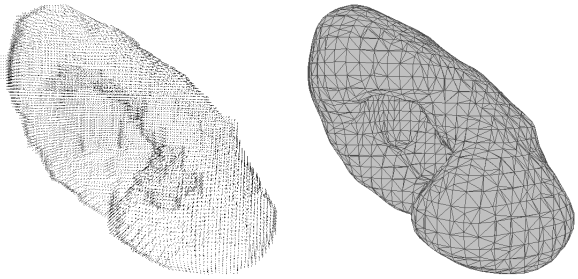

Figure 4: *Final result of a kidney point cloud (left) and its reconstruction using the Poisson surface reconstruction (right).*

3.3 Morphing

The morphing stage must have very basic user- interactions. The two models to morph are close to each other since they both come from the same kidney. Thus, the mesh morphing method uses an automatic mesh cutting up in two patches, unit disk mapping and metamesh creation. We cannot map onto a sphere as kidney models are not star-shaped. All these steps are described in detail hereunder. For the rest of this paper, we will use the following symbols: M represents a given model, M_s is the source model and M_t the target model, as used in the previous sections. C is the connectivity between vertices, edges and faces of M. $V = \{v_1, v_2, v_3, ..., v_n\}$ is the position in $\mathbb{R}^3$ of vertices. Edges are represented as a pair of vertices $\{i, j\}$ and faces as a triplet of vertices $\{i, j, k\}$. Finally, $N(i)$ is the set of adjacent vertices to vertex $\{i\}$, *i.e.* $N(i) = \{\{j\} | \{i, j\} \in C\}$.

Mesh cutting up: obtaining the *tearing path*

The first stage of the mesh dissection consists in computing its principal axis. This can be done by considering only the vertices and using Principal Component Analysis (PCA). Moreover, the PCA gives the 3 principal vectors of the mesh; the firsts two and the barycenter of the mesh define the *principal plane*. Thus, the next stage consists in computing the intersections between edges of M and the principal plane, defining what we call the *intersected edges*. In the same way, the vertices $\{i, j\}$ of an intersected edge are called *intersected vertices*. This set of the intersected edges is the first stage of the final tearing path (see Figure 5).

The tearing path must be a unique loop of edges in C, *i.e.* $\{\{i_1, i_2\}, \{i_2, i_3\}, ..., \{i_{n-1}, i_n\}, \{i_n, i_1\} \,|\, \{i_k, i_m\} \in C \; \forall (k, m) \in [1; n]$; this set of edges is a subset of C and is called c. Thus, two successive intersected edges must share a same vertex. The purpose of the first post-process of the intersected edges is to remove dead-end edges from c. Such edge has one of its vertices which is not shared with any other intersected edge, *i.e.* $\{\{i, j\} | \forall l \in N(j) \{j, l\} \notin c\}$. To detect such edges, we

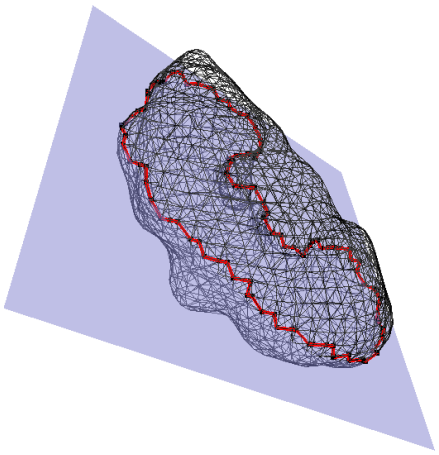

Figure 5: *Intersection between the kidney model and its principal plane (in blue). The resulting tearing path is displayed in red.*

first compute the partial adjacency list of each vertex in c. This list is the set of adjacent vertices $\{j\}$ in c to a vertex $\{i\}$, *i.e.* $\{\{j\}|\{i,j\} \in c\}$. A dead-end edge is then simply detected when at least one of its vertices has only one neighbor, *i.e.* its partial adjacency list length is 1 (see Figure 6 - b). The second post-process consists in removing *local loops*: the tearing path must be a unique succession of edges and each vertex must be shared by two and only two edges. Thanks to the partial adjacency list, vertices from which the tearing path separates are easily detected: such vertices have, at least, 3 neighbors. Thus, local loops are removed as follow. Starting from a 2-adjacency vertex we choose arbitrarily one of its neighbors and so on, until a 3-adjacency vertex is reached. During this step, each vertex is skimmed only once so that it appears at most once in the final tearing path. An arbitrary neighbor of the current 3-adjacency vertex is still chosen, but every other edges containing the current vertex is suppressed from c. As such a process creates new dead-end edges, every edge of each 2-adjacency neighbor is recursively suppressed until the neighbor is a 3-adjacency vertex (see Figure 6 - c,d,e). As the current 3-adjacency vertex becomes a 2-adjacency vertex, the whole process is repeated until we fall back on the first vertex.

Mapping mesh onto the unit disk

Once the tearing path has been computed, vertices are tagged in three different way. We call them *tag* 0, 1 and 2. Tagging the mesh allows to define the two parts of it which will be mapped later. Vertices defining the tearing path are tagged as 0. A unique arbitrary neighbor of a vertex tagged as 0 is tagged as 1. We recursively tag all its neighbors, so that a whole part of the mesh is tagged as 1. The other part is tagged as 2. Both

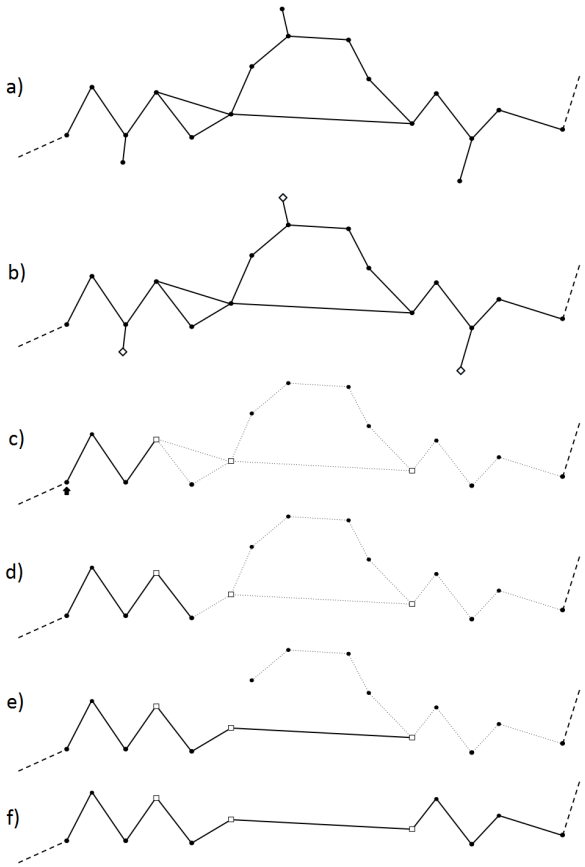

Figure 6: *Whole example of the post-process of a tearing path. Although this example cannot exist in a real situation, it presents all the cases needed to understand how the full post-process works. From top to bottom: (a) Original tearing path - (b) 1-adjacency vertex detection (diamond) and dead-end edges suppression - (c) 3 (or more)-adjacency vertex detection (square). Starting from the pointed vertex, an arbitrary neighbor is chosen. - (d) For a 3-adjacency vertex, we still choose an arbitrary neighbor, but every other edge is suppressed. - (e) To avoid apparition of new dead-end edges when edges are suppressed, recursive suppression of every edges from 2-adjacency neighbor is done. - (f) The final tearing path obtained after the post-process.*

meshes are then rotated so that their principal planes are aligned with xz-plane. This way, it is possible to check if parts tagged the same in the two models have the same y orientation. If not, tags 1 and 2 of one model are swapped. This step is essential since the part of M_s tagged as 1 (resp. 2) will be morphed into the part of M_t tagged as 1 (resp. 2) (see Figure 7).

Now that every vertex is tagged, they can be mapped onto the unit disk. Although any kind of mapping is applicable, we choose the discrete harmonic mapping [Pol00] since it preserves as much as possible the topo-

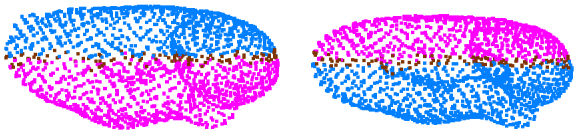

Figure 7: *Example of two models for which a same tag has a different y orientation. Vertices in red are tagged as 0, vertices in cyan tagged as 1 and vertices in magenta tagged as 2.*

logy of faces of both models. The most straightforward step of this mapping is for the intersected vertices. They are fixed on the unit circle in a way that arc length between each pair of successive vertices is proportional to the original length of edge in mesh. For vertices tagged as 1 or 2, discrete harmonic mapping (as well as other mapping) amounts to solving a linear system described as follow. Two distinct mappings are done, one for each tag. Let V_i be the vertices to map with $0 \leq i < n$ index of vertices tagged as 1 (resp. 2) and $n \leq i < N$ index of vertices tagged as 0. Then, the linear system to solve is:

$$(I-\Lambda)\begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_{n-1} \end{pmatrix} = \begin{pmatrix} \sum_{i=n}^{N-1} \lambda_{0,i} v_i \\ \sum_{i=n}^{N-1} \lambda_{1,i} v_i \\ \sum_{i=n}^{N-1} \lambda_{2,i} v_i \\ \vdots \\ \sum_{i=n}^{N-1} \lambda_{n-1,i} v_i \end{pmatrix}$$

where $\Lambda = \{\lambda_{i,j}\}$ and $\lambda_{i,j}$ is a coefficient depending on the mapping used. Here, for discrete harmonic mappings, we have:

$$\lambda_{i,j} = \begin{cases} \frac{cot\alpha_{i,j}+cot\beta i,j}{\sum_{j\in N(i)}(cot\alpha_{i,j}+cot\beta_{i,j})} & \text{if} \quad \{i,j\} \in C \\ 0 & \text{if} \quad \{i,j\} \notin C \end{cases}$$

with $\alpha_{i,j} = \angle(i,k_0,j)$ and $\beta_{i,j} = \angle(i,k_1,j)$. Edge $\{i,j\}$ is adjacent to two and only two faces since M is a triangular mesh. k_0 and k_1 are the two vertices that define these faces. We call M'_{sN} M'_{tN} the mapping of M_s and M_t for tag N. Similarly we call $\{i'\}$ a mapped vertex. Although such notation should not exist since only the position of vertices (v_i) changed during the mapping, this notation will make further expressions more straightforward.

Metamesh creation and animation: computing intersections and barycentric coordinates

The next step is to overlay M'_{sN} M'_{tN} for both tags in order to compute the metamesh. The first stage is to detect intersections between mapped edges. When two edges $\{i',j'\} \in C$ for M'_s and $\{k',l'\} \in C$ for M'_t cross, a new vertex is created. Two valid definitions of this intersection point are $v'_i + \alpha\overrightarrow{v'_i v'_j}$ and $v'_k + \beta\overrightarrow{v'_k v'_l}$. Coefficient α and β are saved along with the new vertex. These coefficients will be necessary for intermediate models as they are sufficient to compute the coordinates of the vertex, even when v_i, v_j, v_k and v_l are interpolated. These kind of vertex is called an *intersection vertex*. Once an intersection vertex is created, appropriate edges and faces are created along with it in order to build the topology of the metamesh M_m. These new edges and faces will allow M_m to combine topology of both M_s and M_t and to have a continuous interpolation between the two models (see Figure 8).

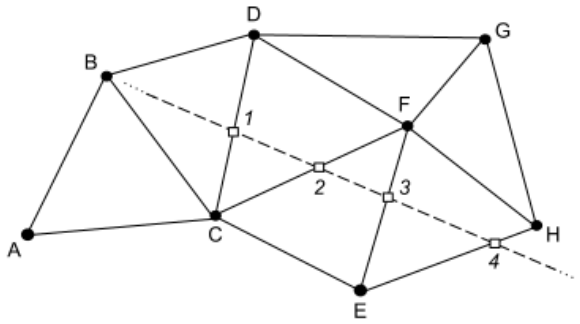

Figure 8: *Example of intersections between mapped edges of M_s (solid line) and M_t (dotted line). Intersection points 1, 2, 3 and 4 are created, as well as appropriate edges (C1, 1D, C2, 2F, ...) and faces (C12, F23, ...).*

The second stage of the metamesh creation is the computation of barycentric coordinates (BC) for every vertices of M_s and M_t. To do that, we first want to know on which mapped face $\{i',j',k'\}$ of M'_t (resp. of M'_s) a mapped vertex v'_m of M'_s (resp. M'_t) lies on. The BC are a unique triplet u,v,w such that $v'_m = uv'_i + vv'_j + wv'_k$. The face where v'_m lies on and its BC are saved. This kind of vertex is called a *mesh vertex* (see Figure 9).

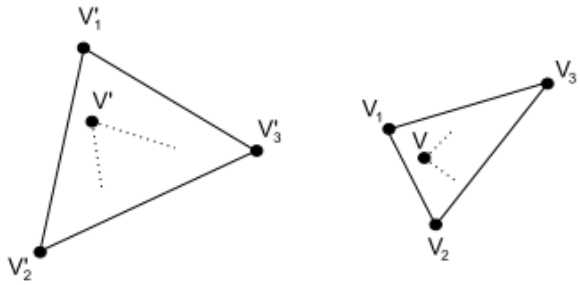

Figure 9: *Mapped vertex of M'_t, v' lies on face $\{v'_1,v'_2,v'_3\}$ of M'_s. Its BC are computed so that $v' = 0.8v'_1 + 0.7v'_2 + 0.2v'_3$. Position of v on face $\{v_1,v_2,v_3\}$ of M_s is then known thanks to these coordinates: $v = 0.8v_1 + 0.7v_2 + 0.2v_3$.*

Thus, the metamesh is completely built and composed of a set of intersection vertices and mesh vertices.

Intermediate models can now be easily obtained by interpolating positions of vertices. The interpolation is possible since we know, for each one of them, an initial and a final position as following: for a mesh vertex coming from M_s, the initial position is its position in M_s. The final position is known by the combination of its BC and the face of M_t it lies on. Inversely, for a mesh vertex coming from M_t, the initial position is known using its BC and the face of M_s it lies on. The final position is its natural position in M_t. For an intersection vertex, the initial position is known thanks to its α coefficient and the edge of M_s it lies on. The final position is computed using its β coefficient and the edge of M_t it lies on.

3.4 Tracking the tumor

The tumor tracking is the second goal of our method. It is important to know where it is located to adjust the wave beam accordingly. From this point of view, there are two main differences between the tumor and the kidney. The first one is the tumor is not deformed by the respiratory cycle, it only moves along with the kidney. The second one is the tumor is similar to an ellipsoid. In the segmentation step, the tumor is segmented separately from the kidney and in a such way that the center of the tumor is known. An other mesh morphing to obtain the tumor movements (*i.e.* its tracking) would be inappropriate since its shape remains the same from one breathing phase to another. Moreover, it would cost useless computational time. A more convenient way to do that is to interpolate the position of the tumor since we have the coordinates of its center for the inhale, exhale and mid-cycle phases. We can use a quadratic Bézier curve interpolation, which gives the tumor position for intermediate phases. In real conditions, the patient will be anesthetized and on respirator, allowing a full control on his breathing, *i.e.* the phase of his respiratory cycle is known at any time. Therefore, it is really easy to synchronize the metamesh and the tumor interpolation along with the patient's breathing. Thus, the 3D coordinates of the tumor are known at any time and correspond to its real position, resulting in the tumor tracking.

4 RESULTS

Our method has been tested on a set of three kidney models M_1, M_2 and M_3 obtained as described in section 3.2. Theses models correspond respectively to the inhale phase, mid-cycle phase and exhale phase. Two mesh morphing were performed: the first between M_1 and M_2 and the second between M_2 and M_3. Figures 11, 12 present several intermediate models obtained while performing the morphing from M_1 to M_2 to M_3. As results are not very explicit with frozen models, an animated version can be seen at the following URL: http://www.youtube.com/watch?v=dhPqLp2X8NQ.

General movement and deformation of the kidney are respected. The natural rotation of the principal axis of the organ is present here, as well as its enlargement. On the other hand, local deformations are not totally satisfying, especially for the tumor. The one on the morphed kidney is absorbed into a part of the kidney and reappear from a different part, right next to its original location. The natural deformation would have been a smooth displacement between theses two locations, almost like a translation. This is due to the morphing method itself. Although these false deformations are not really noticeable, they become obvious when tumor is displayed: it sticks out of the kidney model (Figures 10). Another drawback of our method is the cutting of the mesh by a plane. In order to have a correct morphing, the tearing path obtained from the intersection of the mesh and this plane must be composed of one connected component. Such a criterion is not always guaranteed. From our experiment and analysis, computing intersection between a kidney model and its principal plane will result in a one connect component tearing path (this is due to the shape of the organ itself), but that would not be necessarily the case for another organ or some kind of arbitrary models.

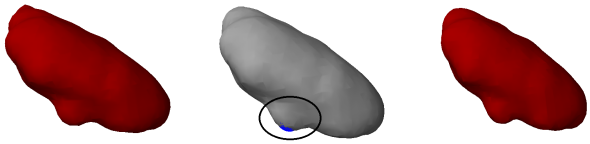

Figure 10: *Highlighting local deformation problem. Intermediate model with tumor (blue ellipsoid) presents local inaccuracy, especially for the tumor region (encircled).*

As the three models have more or less the same number of vertices, edges and faces, computation times for one morphing are equivalent for the other.s The models we have are composed by up to 2,300 vertices, 6,900 edges and 4,600 faces. A morphing is computed in *40s*, each step being repeated twice, one for each tag (data was processed on a laptop with an Intel Core i7 processor and 4 Go of RAM). Although that does not allow to compute mesh morphing in real time, this execution time is acceptable for our medical environment, where interventions used for our non-invasive tumor destruction (**Hi**gh **F**ocused **U**ltrasound) are very long (up to 3 hours). Moreover, the whole computation time is almost needed only for the metamesh creation, which is done only once. Its animation can be done in real time as it is simply an interpolation between an initial and final position of its vertices as seen in section 3.3.

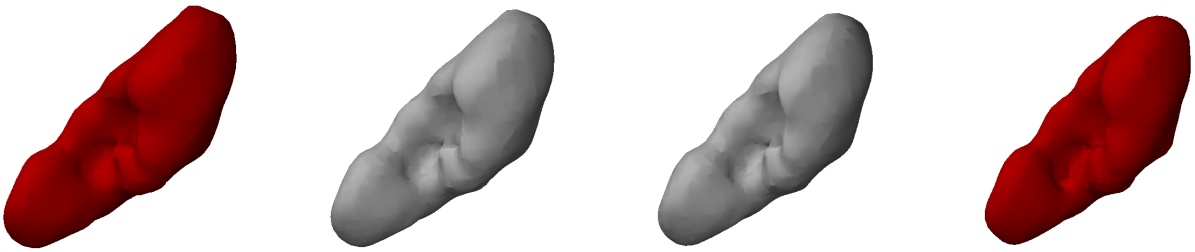

Figure 11: *Final results showing natural movements of the right kidney due to respiration. Source and target models obtained from reconstruction are displayed in red. Intermediate models are displayed in grey. Morphing from M_1 to M_2 is showed here (from left to right).*

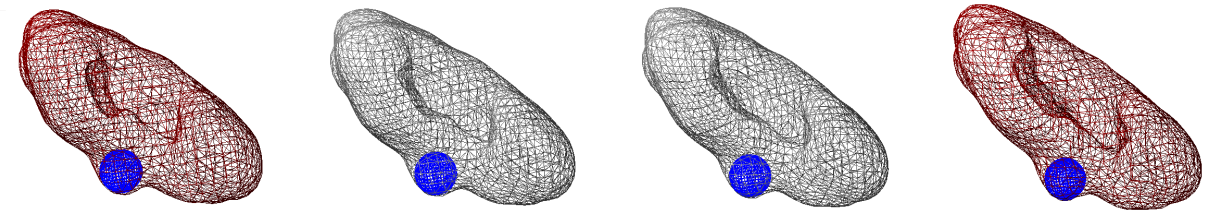

Figure 12: *Morphing between M_2 to M_3 from a different point of view (rotation of 180 degrees around vertical axis). Models are displayed in wireframe and the tumor is visible (blue ellipsoid).*

5 CONCLUSION

We have presented an original and geometric approach to obtain the natural motion simulation of the kidney under the respiratory cycle. Starting from three medical imaging acquisitions of the organ, each one for a different phase of the cycle, kidney is first segmented then reconstructed in order to create one model for each phase. Kidney is finally animated in 3D and respiratory movements are simulated through mesh morphing among the three models we previously had (from first to second model and from second to third). To do that, it is first necessary to cut the mesh, which is done automatically here. Then the two different parts of a mesh are mapped onto the unit disk. This mapping is used to compute a metamesh which comprises the topology of two successive models. Soft transition between two models, and thus the kidney animation, is finally obtained by interpolating each vertex of the metamesh between an initial and a final position. Although general deformation and movement of the kidney is well simulated, local deformations are not precise enough, especially for tumors near the surface. A way to overcome this problem would be to force regions with similar curvature to morph into each other.

ACKNOWLEDGMENT

This work is granted by the Foundation "Santé, Sport et Développement Durable", presided by Pr. Yvon Berland. The authors would like to thank everyone involved in the KiTT project: Christian Coulange for his precious help, Marc André, Frédéric Cohen and Philippe Souteyrand for their wise advices and for providing CT scan data, and Pierre-Henri Rolland for his support.

REFERENCES

[ACOL00] Marc Alexa, Daniel Cohen-Or, and David Levin. As-rigid-as-possible shape interpolation. *Proceedings of Computer Graphics and Interactive Techniques*, 2000.

[BCA96] Eric Bardinet, Laurent Cohen, and Nicholas Ayache. Tracking and motion analysis of the left ventricle with deformable superquadrics. *Medical Image Analysis*, 1(2):129 – 149, 1996.

[EDD+95] Matthias Eck, Tony DeRose, Tom Duchamp, Hugues Hoppe, Micheal Lounsbery, and Werner Stuetzle. Multiresolution analysis of arbitrary meshes. *Proceedings of SIGGRAPH*, pages 173 – 182, 1995.

[GSL+98] A. Gregory, A. State, M.C Lin, D. Manocha, and M.A. Livingston. Feature-based surface decomposition for correspondence and morphing between polyhedra. *Proceedings of Computer Animation*, pages 64 – 71, 1998.

[HNS+08] Alexandre Hostettler, Stéphane Nicolau, Luc Soler, Yves Rémond, and Jacques Marescaux. A real-time predictive simulation of abdominal organ positions induced by free breathing. *International Symposium on Biomedical Simulation*, pages 89 – 97, 2008.

[KBH06] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. *Eurographics Symposium on Geometry Processing*, 2006.

[KCP92] James Kent, Wayne Carlson, and Richard Parent. Shape transformation for polyhedral objects. *Computer Graphics*, 26(2), July 1992.

[KK99] George Karypis and Vipin Kumar. Multilevel k-way hypergraph partitioning. *Proceedings of the 36th annual ACM/IEEE Design Automation Conference*, 1999.

[KSK97] T. Kanai, H. Suzuki, and F. Kimura. 3d geometric metamorphosis based on harmonic map. *Proceedings of The Fifth Pacific Conference on Computer Graphics and Applications*, 1997.

[KSK00] Takashi Kanai, Hiromasa Suzuki, and Fumihiko Kimura. Metamorphosis of arbitrary triangular meshes. *Proceedings of Computer Graphics and Application*, 20(2), March 2000.

[LDSS99] Aaron Lee, David Dobkin, Win Sweldens, and Peter Schroder. Multiresolution mesh morphing. *Proceedings of Computer Graphics and Interactive Techniques*, 1999.

[LMVD11] Valentin Leonardi, Jean-Luc Mari, Vincent Vidal, and Marc Daniel. Reconstruction 3d du volume rénal à partir d'acquisitions scanner volumiques. *Journée du Groupe de Travail en Modélisation Géométrique, GTMG*, pages 83 – 92, March 2011.

[LSS+98] Aaron Lee, Win Sweldens, Peter Schroder, Lawrence Cowsar, and David Dobkin. Maps: Multiresolution adaptive parameterization of surfaces. *Proceedings of SIGGRAPH*, pages 95 – 104, July 1998.

[MCG+03] Martin Murphy, Steven Chang, Iris Gibbs, Quynh-Thu Le, Jenny Hai, Daniel Kim, David Martin, and John Adler. Patterns of patient movement during frameless image-guided radiosurgery. *International Journal of Radiation Oncology Biology Physics*, 55(5):1400 – 1408, 2003.

[NdSE+08] Karsten Ostergaard Noe, Baudouin Denis de Senneville, Ulrik Vindelev Elstrom, Kari Tanderup, and Thomas Sangild Sorensen. Acceleration and validation of optical flow based deformable registration for image-guided radiotherapy. *Acta Oncology*, 47(7):1286 – 1293, 2008.

[NPSA07] Stéphane Nicolau, Xavier Pennec, Luc Soler, and Nicholas Ayache. Clinical evaluation of a respiratory gated guidance system for liver punctures. *Medical Image Computing and Computer-Assisted Intervention*, pages 77 – 85, 2007.

[NUG+08] Masahiko Nakamoto, Osamu Ukimura, Inderbir Gill, Arul Mahadevan, Tsuneharu Miki, Makoto Hashizume, and Yoshinobu Sato. Realtime organ tracking for endoscopic augmented reality visualization using miniature wireless magnetic tracker. *Medical Imaging and Augmented Reality*, pages 359 – 366, 2008.

[OTW+05] B. Olbricha, J. Trau, S. Wiesner, A. Wicherta, H. Feussner, and N. Navab. Respiratory motion analysis: Towards gated augmentation of the liver. *Computer Assisted Radiology and Surgery*, 1281:248 – 253, 2005.

[Pol00] K. Polthier. Conjugate harmonic maps and minimal surfaces. Technical report, Technische University of Berlin, 2000.

[RMK+05] Mauricio Reyes, Grégoire Malandain, Pierre Malick Koulibaly, Miguel Gonzalez Ballester, and Jacques Darcourt. Respiratory motion correction in emission tomography image reconstruction. *Medical Image Computing and Computer-Assisted Intervention*, pages 396 – 376, 2005.

[RMOZ01] Torsten Rohlfing, Calvin Maurer, Walter O'Dell, and Jianhui Zhong. Modeling liver motion and deformation during the respiratory cycle using intensity-based free-form registration of gated MR images. *Medical Imaging 2001: Visualization, Image-Guided Procedures, and Display*, pages 337 – 348, February 2001.

[RSH+99] Daniel Rueckert, L. I. Sonoda, C. Hayes, D. L. G. Hill, M. O. Leach, and D. J. Hawkes. Nonrigid registration using free-form deformations: Application to breast MR images. *IEEE Transactions on Medical Imaging*, 18(8), August 1999.

[SBMG06] David Sarrut, Vlad Boldea, Serge Miguet, and Chantal Ginestet. Simulation of four-dimensional CT images from deformable registration between inhale and exhale breath-hold CT scans. *Medical Physics*, 33(3), March 2006.

[SGB+00] Achim Schweikard, Greg Glosser, Mohan Bodduluri, Martin Murphy, and John Adler. Robotic motion compensation for respiratory movement during radiosurgery. *Journal of Computer-Aided Surgery*, 2000.

[SSK+00] H. Shirato, S. Shimizu, K. Kitamura, T. Nishioka, K. Kagei, S. Hashimoto, H. Aoyama, T. Kunieda, N. Shinohara, H. Dosaka-Akita, and K. Miyasaka. Four dimensional treatment planning anf fluoroscopic real-time tumor tracking radiotherapy for moving tumor. *International Journal of Radiation Oncology Biology Physics*, 48:435 – 442, September 2000.

[YHM07] Han-Bing Yan, Shi-Min Hu, and Ralph Martin. 3d morphing using strain field interpolation. *Computer Science and Technology*, 1, 2007.