Enhanced Algorithm for Reconstruction of Three-Dimensional Mesh from Medical Images using Tessellation of Recent Graphics Cards

Lamyae Miara[1], Said Benomar El Mdeghri[2], Mohammed Ouçamah Cherkaoui Malki[3]
Department of Computer Science, Faculty of Science Dhar El Mahraz
University Sidi Mohammed Ben Abdellah
Fez, Morocco

***Abstract*—The reconstruction of a 3D mesh using displacement vectors for medical images is a recent method that allows the exploitation of modern GPUs. This method demonstrated its efficiency by accelerating 3D visualization calculations and optimizing the storage process. In fact, it is divided into two main stages. The first step is the construction of a basic mesh by applying the Marching Cubes algorithm, and the second step is the extraction of the displacement vectors, which represent the details lost in the basic mesh. In fact, the Marching Cubes algorithm used to build the basic mesh suffers from some problems that we will try to overcome in this article. These problems are summarized in the ambiguity encountered during the construction of the basic mesh in some cases. Also, the resulting basic mesh must undergo modifications, in order not to have errors of form, which requires time and memory, and which gives the end a final mesh which is not optimal and even erroneous in certain situations. Our method is based on extracting the contours of the anatomy to be reconstructed from a sequence of 2D images. Each contour will be represented by a triangle. The shape of the basic mesh will then be the result of the connection of these triangles. This strategy avoids the use of the marching cubes algorithm in the reconstruction of the basic mesh in order to overcome the problems mentioned above.**

***Keywords*—3D reconstruction; medical imaging; marching cubes; displacement vectors; contour extraction; contour matching**

I. Introduction

Using medical imaging, it is now possible to visualize the patient's internal anatomy without resorting to surgery. The images obtained are 2D slice series on which it is not always easy to interpret the different problems faced by physicians. In order to overcome these problems, 3D imaging has been developed [1]. This technique will allow doctors, to visualize his patient in virtual 3D. This will bring a better representation of the internal anatomy in order to optimize the chances of a good diagnosis [2].

On the other hand, the recent technological development of devices used in the medical field has contributed to the resolution of the medical images obtained. The reconstruction of a 3D mesh of the human anatomy from these images generates a very large number of polygons. It preserves many details of the 2D image that lead to problems in the amount of information stored, and in terms of the complexity of the display of calculations for real-time visualization.

Therefore, this paper focuses on the algorithms of the 3D reconstruction of medical images using the tessellation of recent graphics cards. These algorithms, based in principle on the Marching Cubes algorithm [3] and a displacement map, have shown their efficiency by optimizing the amount of information to be stored in the 3D mesh, and by accelerating the rendering computations. These algorithms are based on a sequence of 2D medical images, which will be segmented to extract only the organ to be reconstructed. Then, the Marching Cubes algorithm is applied to these segmented images.

The size of the cubes selected in the Marching Cubes algorithm must be large to have a low-resolution basic mesh, this is the basic mesh. The algorithm developed by [4] allows us to extract the lost details from each basic mesh triangle using the information we have in the medical images. These details are stored as displacement vectors.

The Marching Cubes algorithm used to build a low-resolution basic mesh causes some ambiguity problems during the construction of the mesh in some cases, so the basic mesh has to be modified to avoid shape errors, which requires time and memory, and gives a final mesh that is not optimal and even erroneous in some situations.

The rest of this paper is organized as follows: the background and related works are reviewed in Section 2; in Section 3, our methodology is discussed; Section 4 was devoted to discussing our results, and in Section 5, we summarize our work with conclusion.

II. Related Works

Among the 3D reconstruction algorithms used in the medical field is the Marching Cubes. The latter generates a 3D polygonal object from a three-dimensional scalar field [5]. This algorithm executes this scalar field, taking eight points at a time from an imaginary cube and determines the creation polygons to represent part of the iso-surface contained in this cube. This algorithm is based on a precomputed table of 256 configurations of the number of polygons in a cube [6], treating each of the eight scalar values as a bit in an 8-bit integer. Although the MC algorithm has proven to be efficient, it suffers from several problems. Many solutions have been proposed to solve these problems.

The first problem detected in the MC algorithm is the case of ambiguities found in some configurations. In fact, the same cube can be triangulated in several ways. The author in [7] have identified the problem of face ambiguity, which occurs when two diagonally opposite vertices are labeled positive and two negatives are labeled negative. This ambiguity can lead to holes in the topology. The authors in [8] and [9] independently show another type of ambiguity that occurs inside a cube. These ambiguities can often be resolved by adding additional test points in each cube. Several methods are proposed to solve this problem, either for ambiguous cases on faces [7] or inside cubes [10] and [11].

The Marching Cubes algorithm divides the space into a regular cubic grid. The resolution of the resulting polygonal surface depends directly on the size of the grid. Increasing the number of cubes in the grid can increase the resolution of the polygonal area, but the number of resulting triangles can be large even if the original area is quite simple. To reduce the number of triangles, several methods have been developed to apply the Marching Cubes algorithm of the adaptive grid. A method that introduces the concept of multi-resolution grid generation [12], an algorithm that adapts the size of the triangles to the shape of the surface, has been proposed by [9]. The author in [13] proposed another method that keeps the same grid size and increases the resolution of the moving surface from the grid peaks to the surface of the cube, thus increasing the number of cubes containing the surface.

In all the proposed solutions to improve the Marching Cubes algorithm, we always have the problem of the huge mesh obtained at the end of this treatment, which directly influences the fluidity of the visualization of this mesh in real time. Among the algorithms proposed to solve this problem, we have the algorithm proposed by [14] which is based on the Marching Cubes and on the tessellation of recent graphics cards. This algorithm has proven its interest in optimizing the amount of information to store for 3D mesh and to speed up display calculations using recent GPUs.

This algorithm uses a sequence of medical images. These images are segmented to extract only the organ to be reconstructed. Then we apply, on these segmented images, the Marching Cubes algorithm [15].

The size of the cubes chosen in the Marching Cubes algorithm or its extensions must be important to have a low-resolution mesh, it is the basic mesh.

The algorithm developed by [14] extracts the lost details of each polygon from the basic mesh using the information that we have in medical images. These details are saved as displacement vectors.

The method used in the work of [14] causes some problems in the precision of the final high-resolution mesh.

Indeed, the use of Marching Cubes algorithm in the reconstruction of basic mesh always poses some problems:

- The first problem posed by this algorithm is the cases of ambiguities found in certain configurations. For the same cube we can have several ways of triangulating (Fig. 1).

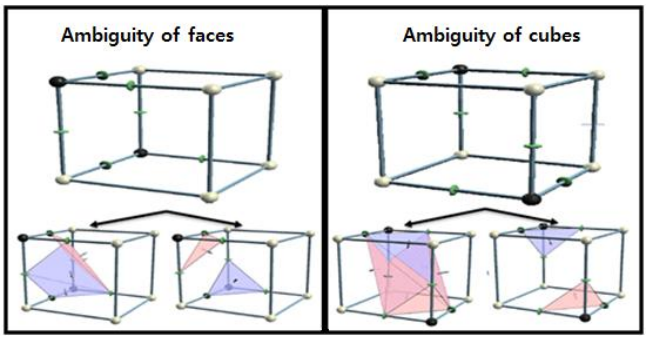


Fig. 1. Ambiguity of Faces and Cubes. Several Triangulations possible for the same case.

These ambiguities can cause inconsistencies, such as holes in the basic mesh (Fig. 2).

In Fig. 2, we represent two neighboring cells. The left cell is triangulated, the voxels having the value 1 in a separate way (case 1), the neighboring cell on the right is triangulated, the voxels have the value 1 in a connected way (case 2). Both triangulations are valid, but they generate a triangular interface incompatible with a hole. The dotted lines indicate respectively the edges of the triangle in the other cell.

- The second problem detected is the need to modify the mesh obtained by Marching Cubes in order to be used in Displacement Mapping. This treatment consists in removing some vertex from the mesh and merging triangles to obtain quadrilaterals. The application of this treatment on certain areas, indicated as complicated, leads to a deformation of the basic mesh (Fig. 3).

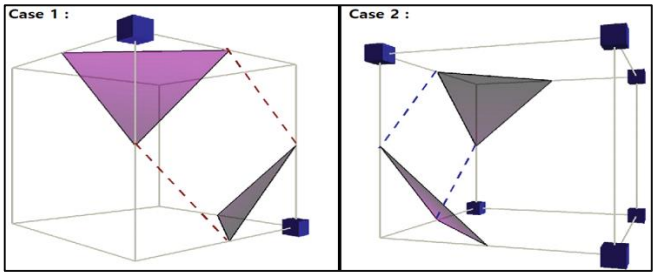


Fig. 2. The Possible Triangulations of Marching Cubes for Two Neighboring Cells.

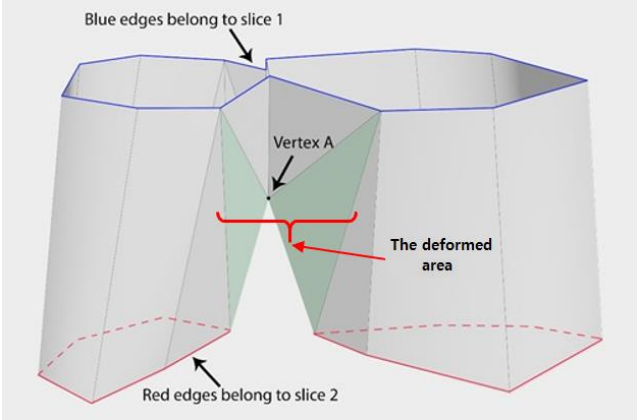


Fig. 3. Complicated Area in a basic Mesh.

In this figure, we have a mesh obtained after the application of Marching Cubes on two slices. Vertex A is located in a complicated area, the removal of this vertex leads to the deformation of this mesh. The solution adapted by the original article is not to change its position, which causes the existence of triangles (the two green triangles) in which we cannot add the details lost with displacement mapping.

In this paper, we proposed a new algorithm to reconstruct the low-resolution mesh. This technique is essentially based on an algorithm that extracts the contours and their correspondences to form this mesh. Our method overcomes the problems associated with the use of Marching Cubes.

III. Methodology

This section describes the method developed in this article to reconstruct a basic mesh without using the Marching Cubes algorithm. This method consists in extracting, from a sequence of 2D medical images, the contours of the anatomy to be reconstructed, and then to represent them by triangles, and in the final step we connect these triangles to form the low-resolution mesh.

In the first part, we determine the input of our algorithm and explain the basic idea. The other two parts describe in more detail the main steps of the algorithm: contour extraction and linking the triangles (the contour points). A last part is here to show how to build the high-resolution mesh.

A. Overview of the Algorithm

To overcome the above-mentioned problems, we have chosen a new strategy to build the basic mesh. This strategy is based on the extraction of the contours of the anatomy to be reconstructed from a sequence of 2D medical images. This contour will be divided into three equal parts in order to deduce the coordinates of the three points that will represent this contour by a triangle. The basic mesh will be formed by the correspondence of the triangles of the image N with the image N+1. The positions of the points of the sides that exist between two 2D images are estimated by applying the principle of interpolation, because we have no real information between 2D images (Fig. 4). The result of this processing represents the basic shape of the element to be reconstructed.

The next step is to extract the lost details from each triangle of the basic mesh using the information we have in these 2D images. These details are recorded as displacement vectors. And using this method we can also automatically generate the displacement map for our basic mesh (Fig. 5).

The inputs of our basic mesh construction method are the same inputs as the original algorithm. They are the 2D medical images undergoing a segmentation to extract only the part that we are concerned (Fig. 6).

B. Contour Extraction

Based on the segmented slices, we must extract the contours of the object to be reconstructed.

To detect these contours, several methods are possible, grouped into several classes, those based on non-linear filtering such as the median filter -and more recently- [16], those using high-pass filtering, such as the Prewitt, Sobel and Canny detectors [17], those of multi-scale analysis developed with wavelet theory [18] and [19], and those based on the rare approximation by redundant dictionary [20].

The previously mentioned methods of contouring consist of defining abrupt changes in pixel intensity. However, our issue recommends that we need to get a chained list of contour points, respecting a fixed order. So, the strategy we used is to extract one point from the considered contour, and then we must extract the other points in an orderly way by following the path of these contours.

In the next step we will represent each contour by a triangle. We then calculate the center of gravity of each contour. The center of gravity will be the average of the points that make up the contour (Fig. 7). The horizontal projection of this point on the contour gives at least two points. The point with the maximum abscissa value will be the first vertex of the triangle, i.e. point A. The two vertices of the triangle, the remaining point B and C, must be separated by the same number of contour points.

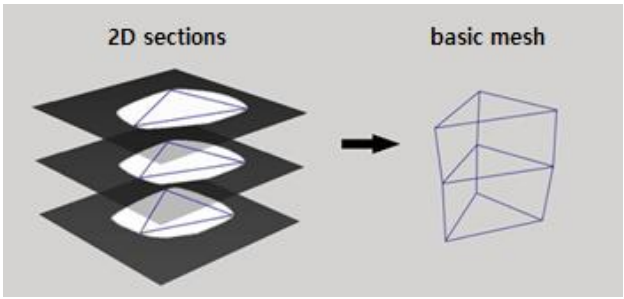


Fig. 4. The Result of the First Stage of the Reconstruction of a basic 3D Mesh from Medical Images.

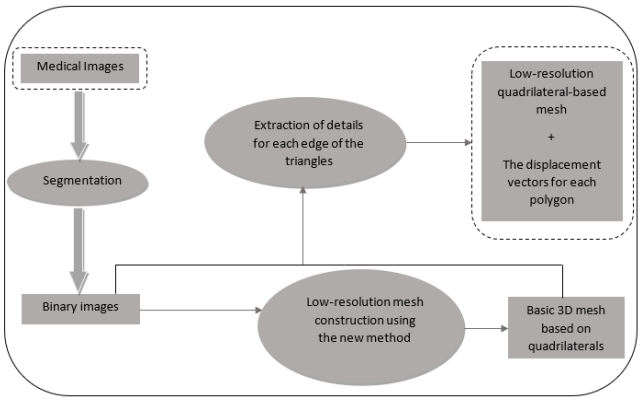


Fig. 5. Diagram Representing the different Steps of our Algorithm.

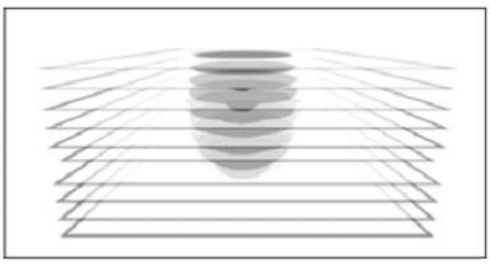


Fig. 6. The Inputs to our Algorithm, the Medical Images [14].

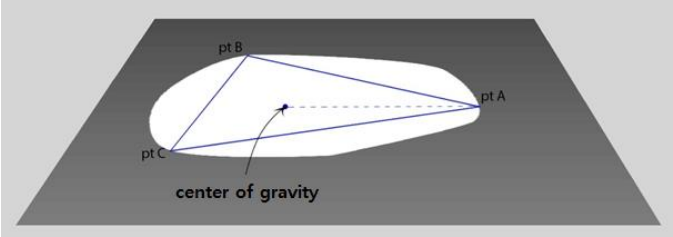


Fig. 7. Calculation of the Center of Gravity and Three Points that represent an Outline of a 2D Image.

Each contour of slice N will be represented by three points forming a triangle. These will be the basis of the low-resolution mesh of the anatomy to be reconstructed.

C. Construction of the basic Mesh

1) Contour matching: In our method, the construction of a basic mesh consists in making the correspondence between the contours of each slice n with those of slice n+1[21].

Drawing faces from the contours isn't an easy thing, since it depends on solving three problems [22]:

- Matching problem: How to match the contours in the N slice with a contour of the n+1 slice?
- Tiling problem: How to connect the points of the contour Cn in the n slice with the points of the contour Cn+1 in the n+1 slice?
- Connection problem: How to divide the contour Cn in slice n that corresponds to the contours Cn+1a and Cn+1b in slice n+1?

The representations shown below, highlights feasible possibilities to solve the matching problem in the case where the number of contours in adjacent slices is not the same (Fig. 8) (contours can be split or merged). Although the matching problem depends on the connection issue, the assumption that it can occur in isolation is not ruled out; especially if the contour curves change strongly between adjacent slices.

To determine the relationship between the contours of two consecutive slices, we used a correspondence factor. To calculate this factor, each contour is represented by a binary matrix of the same size as the slice. The pixels that are inside the contour will be represented by the value 1 and the others by the value 0 (Fig. 9).

Thus, to determine the value of the correspondence factor between 2 contours C1 and C2 of two consecutive slices (Fig. 10), we apply the logical AND operator on the matrices of the two contours.

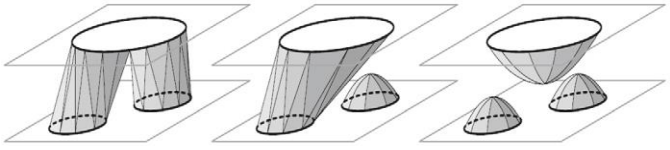


Fig. 8. Three Possible Solutions to the Problem of Matching in the Event of a Change of Topology.

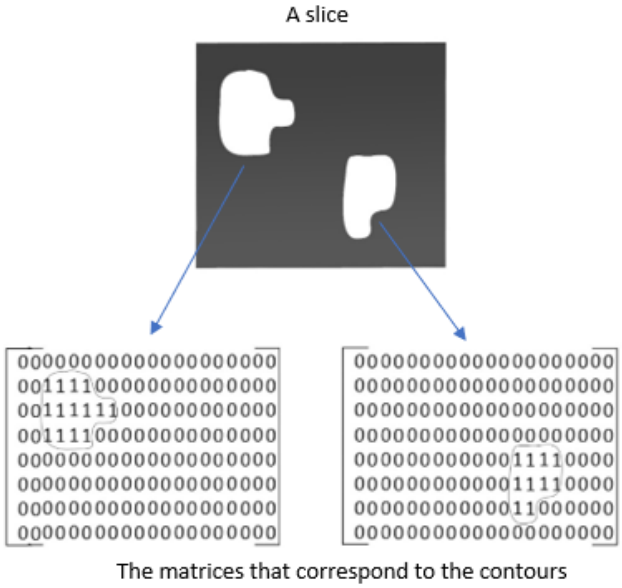


Fig. 9. Representation of Matrices for the Contours of a Slice.

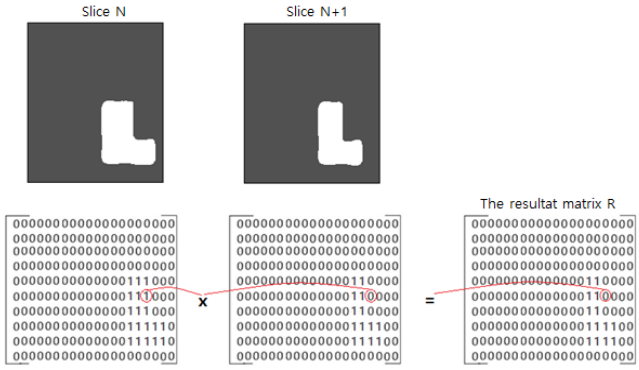


Fig. 10. Calculation of the Result of Two Binary Matrices.

The correspondence factor of a contour is then the sum of the elements of the resulting matrix, divided by the sum of the elements of the matrix of the same contour.

We deduce that the correspondence factor of the contour C1 in relation to C2 is:

$$Fc(C1/C2) = \frac{\sum elt\ R}{\sum elt\ C1} = \frac{4}{6}$$

and the correspondence factor of the contour C2 in relation to C1 is:

$$Fc(C2/C1) = \frac{\sum elt\ R}{\sum elt\ C2} = \frac{4}{4}$$

So, to determine the relationship between the contours of slice N and the contours of slice N+1, using the correspondence factor, there are two possible cases:

- Contour C1 of slice N is connected to only one contour C2 of slice N+1.
- Contour C1 of slice N is connected to several contours of slice N+1.

In order to verify the first case, there must be a correspondence between C1 and C2 in both directions; i.e. the contour C1 must correspond to C2, and C2 must correspond to C1 (Fig. 11). This correspondence exists when the two correspondence factors Fc (C1 / C2) and Fc (C2 / C1) are respectively greater than a given threshold. In our case, we are working on a series of slices from recent scanners, with a distance between slices in the order of millimeters. So, the correspondence factor between the linked contours is close to 100%.

In this article, we must determine a threshold for the correspondence factor (S_f) from which we can determine whether there is a correspondence between 2 contours or not. To make this choice, we have a compromise to make. On the one hand we must choose the lowest possible threshold in order to take into account all the particular cases of contour change from one slice to the other, and on the other hand, we must not have conflicts of correspondence if we accept very low correspondence factors. An example of this conflict is to have a correspondence of one contour of slice n with several contours of slice n+1. The value of the matching threshold we have chosen to meet this compromise is 50%.

In the rest of this article, we assume that the C1 contour is related to a C2 contour if the two correspondence factors are greater than 50%. This assumption will not be true if there is a large distance between the slices. The thing that is not valid in our situation.

And for the second case, we have one contour of slice N which corresponds to two or more contours (C2, C3… CK) of slice N+1. In this case, the correspondence factor is checked only in one direction (Fig. 12).

In this figure, the contours of slice N+1 correspond to the same contour of slice N. The correspondence factors Fc(C2/C1) and Fc(C3/C1) are greater than the determined threshold. However, the correspondence factors Fc(C1/C2) and Fc(C1/C3) may be necessarily lower than the determined threshold.

In the special case, where there are two contours in the same slice N which correspond to each other, and which correspond to the same contour in slice N+1 (Fig. 13), one contour must be removed from slice N to take off this correspondence. In fact, the stronger match is the one that will remain. Also, since Fc(C2/C3) + Fc(C1/C3) is greater than Fc(C2/C1) + Fc(C1/C2), then contour C2 will be removed from this matching assumption.

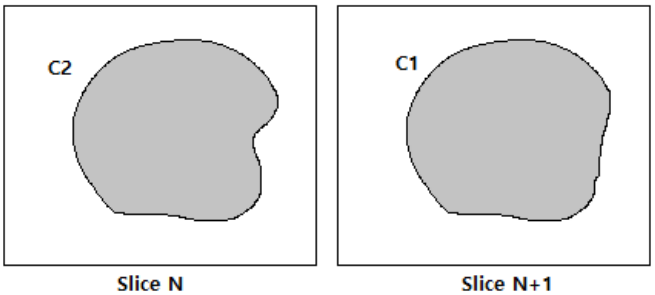


Fig. 11. Contour C1 of Slice N and Contour C2 of Slice N+1 are Linked.

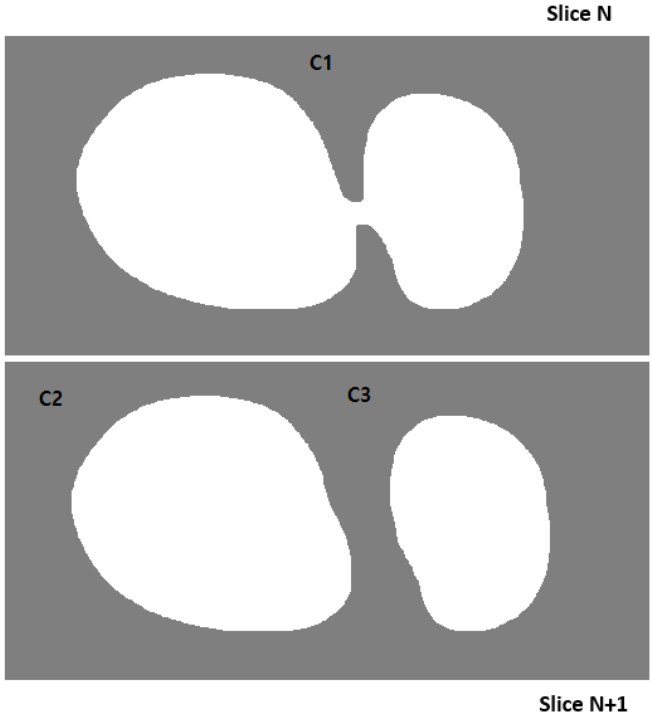


Fig. 12. The Contour C1 is Divided into Two Contours C2 and C3 in the following Slice.

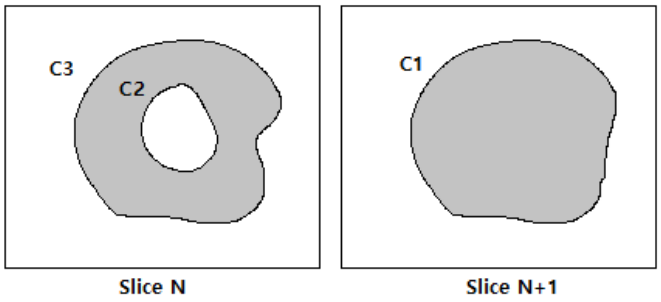


Fig. 13. Two Consecutive Slices that have Contours that Correspond to each other.

2) Construction of the low-resolution mesh: The construction of the mesh amounts to linking the corresponding contours. And to obtain a low-resolution mesh we just connect the triangles that represent the contours. In this step, there are two cases:

- A contour of slice N is connected to a single contour of slice N+1.
- One contour of slice N is connected to two or more contours of slice N+1.

In the first case, the construction of the mesh amounts to connecting the vertices (A, B, C) of the contour C1 with the vertices (A', B', C') of the contour C2 respectively (Fig. 14).

In the second case, the construction of the mesh amounts to connecting the contour C1 of the slice N with the vertices ((A', B', C'), (A", B", C"),…) of contours (C2, C3,…, Ck) by taking into account the correspondence factors Fc (C1 / C2), Fc (C1 / C3),…, Fc (C1 / Ck).

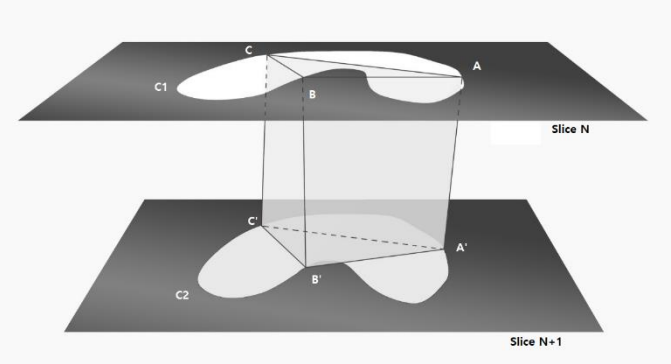


Fig. 14. Connection of Contour C1 of Slice N to Contour C2 of slice N + 1.

Contour C1 will be divided according to the number of contours that correspond to it, and according to the percentage of correspondence of contours C2, C3, ..., Ck with respect to contour C1 is calculated.

The matching factor is used to calculate the matching percentage.

We suppose that: $\sum_{n=1}^{k} Fc\left(\frac{C1}{Cn}\right) = X$

And we use the triangular relation to calculate the percentage of correspondence:

$Fc\left(\frac{C1}{Cn}\right)$ → $Pc\left(\frac{C1}{Cn}\right)$

X → 100%

Then, we have:

$Pc(C1/Cn) =$ $Fc(C1/Cn)$ / X

This percentages of correspondence indicate the percentage of correspondence of the contours C2, C3, ..., Ck with respect to the contour C1. And to locate the position of a contour Ck in the contour C1, we project the center of gravity of the contour Ck on the contour C1. Next, the correspondence area is calculated using the correspondence percentage (Fig. 15). Henceforth, we can determine the three points (A, B, C) of the sub-contour which corresponds to each contour Ck.

In the figure above, the two contours C2 and C3 correspond to the contour C1. So, to calculate the correspondence surface of each contour of slice N + 1 in contour C1, we first projected the centers of gravity of the two contours on contour C1. Then, we made the intersection of the straight line (D) passing by the two projections and the contour C1. However, we calculate the correspondence percentages Pc (C1 / C2) and Pc (C1 / C3). Subsequently, we did a sweep from the end of each intersection of the contour C1 with the line (D) according to the match percentages. In Fig. 15 the sub-outline on the right (green color) represents 33.33% of the contour C1, and the sub-contour on the left (orange color) represents 66.67% of the contour C1.

The next step is to match each contour of the N+1 slice to its corresponding contour in the N slice (Fig. 16).

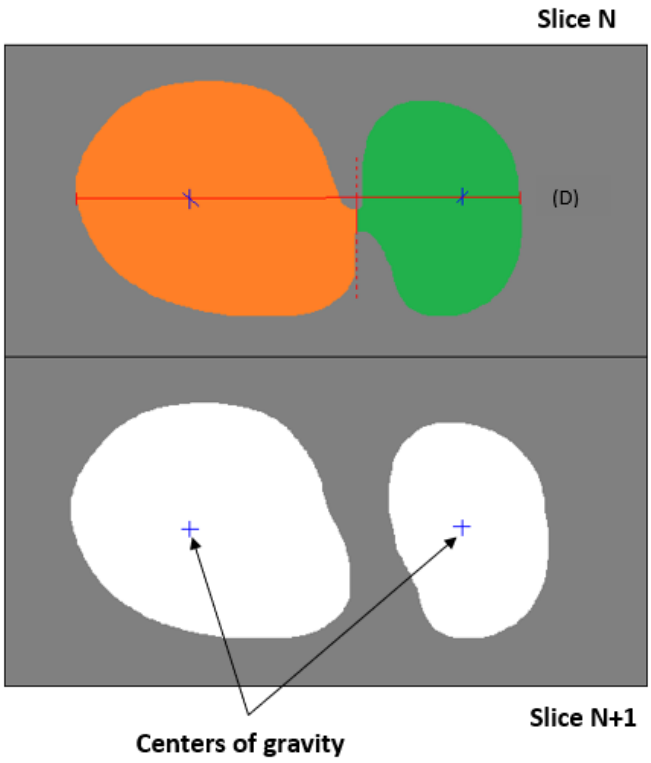


Fig. 15. The Correspondence of the Contour C1 to Two Contours C2 and C3 of the Slice N + 1.

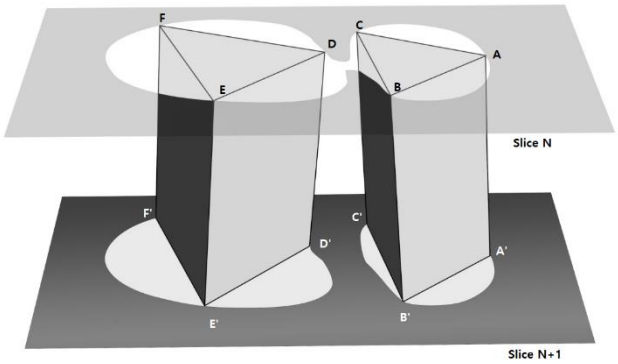


Fig. 16. Construction of the Low-Resolution Mesh from Two Contours C2 and C3 which Correspond to the Contour C1.

D. Construction of the High-Resolution Mesh

As shown in the previous diagrams, the constructed mesh is a basic mesh that does not reflect the true shape of the anatomy in question. It is necessary to add to it the details lost during the construction of the basic mesh in real-time visualization. The displacement vectors must then be extracted using the contour data and quadrilateral edges obtained after linking the triangles that represent the contours.

The detail extraction method used in the original article, is based on the discretization of the edges that are on the 2D images in N points, then to extract for each point the corresponding displacement vector.

Two problems arise when using the displacement vector extraction algorithm to obtain a high-resolution mesh; the first is that all sides of polygons are discretized to N points, with N fixed. However, the widths of the sides are not identical (Fig. 17), and therefore logically the homogeneity of the final 3D mesh will be altered with respect to the details extracted from each polygon.

It can be noted that the level of detail extracted from side A is less important than that of side B.

The second problem concerns the final shape of the anatomy to be reconstructed, which presents errors in certain situations. Drawing a perpendicular line from the points resulting from the discretization of each side sometimes gives two or more points of the intersection with the contour in 2D images. The choice of the extraction method of the original algorithm leads to erroneous results, because through this method the choice of the vector is always made between a point that belongs to the side and the closest intersection point of this side. This alters the accuracy of the result especially since the details no longer correspond to the true shape of the anatomy (Fig. 18).

We notice that the obtained shape (c), using these displacement vectors (b), does not really resemble the true shape (a).

In a more recent work by the same author [4] the extraction method has been modified to overcome the problems mentioned above.

The new strategy used is to extract the displacement vectors is based on the extraction of the contour of the object to be reconstructed from the medical images, then, using discretization of these contours we generate the displacement vectors.

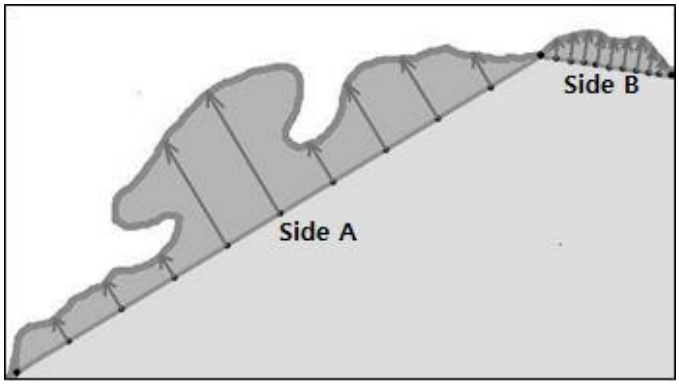


Fig. 17. Heterogeneity of Details Extracted from each Side [4].

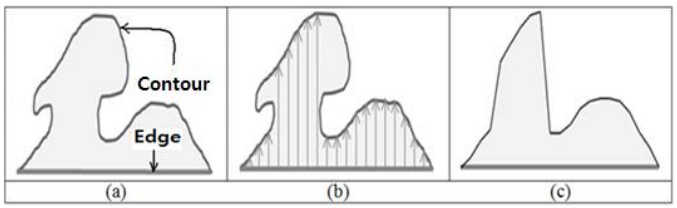


Fig. 18. Errors in Extracting the Small Reliefs. The Contour Obtained (c) by using the Displacement Vectors (b) does not Correspond Exactly to the True form (a) [4].

This new extraction method has an importance related to two main issues; the first concerns the generation of vectors considering the contour itself, this allows more precision for the construction of a real anatomical shape. The second interest is the integration of the same level of detail in all the polygons of the basic mesh, through the discretization of the contour of the anatomy with a fixed point.

Displacement vectors will be used for the automatic generation of the displacement map, which is an image allowing to determine the distance and direction used to move the points of the surface during real-time visualization.

We follow the same approach used in the original paper to generate the high-resolution mesh based on the basic mesh obtained with our new method and the displacement map (Fig. 19).

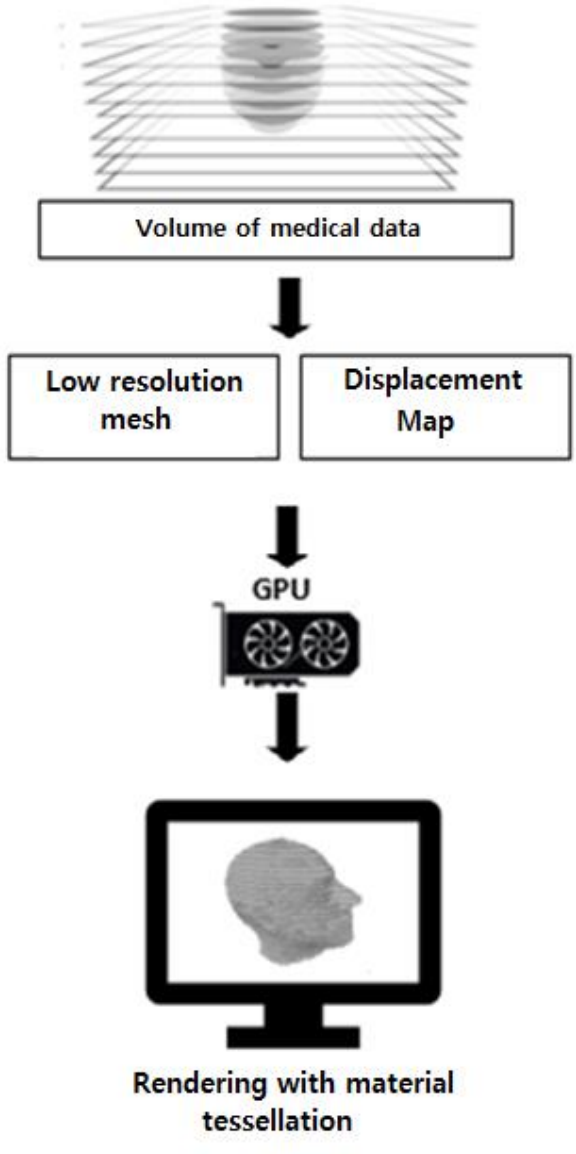


Fig. 19. Rendering with Hardware Tessellation to Generate the High-Resolution Mesh.

IV. Results and Discussion

The implementation of our new method for reconstructing a 3D mesh from medical images is based on:

- The detection and extraction of contours from a sequence of medical images;
- The construction of a low-resolution 3D mesh from the contours;
- The construction of the high-resolution mesh from the displacement vectors and the basic mesh.

The image sequence used in this paper is segmented to extract only the anatomy to be reconstructed.

Initially, we tested our new 3D mesh reconstruction method in terms of the quality of the obtained 3D mesh.

The basic mesh construction method used by the original algorithm is based on the use of the Marching Cubes algorithm, while determining the size of the cubes. This method causes cases of ambiguity in some configurations; i.e. for the same cube, there can be several ways of triangulation. Also, with this method, an additional step is mandatory to render the mesh obtained from the Marching Cubes algorithm in the form of quadrilaterals, in order to apply displacement mapping. And in some situations, complicated areas are confronted during this step, which cause deformations in the reconstruction of the high-resolution mesh.

In Fig. 20, point A is located in a complicated area. This point will lead to deformation when applying displacement vectors, since there is no information at this point.

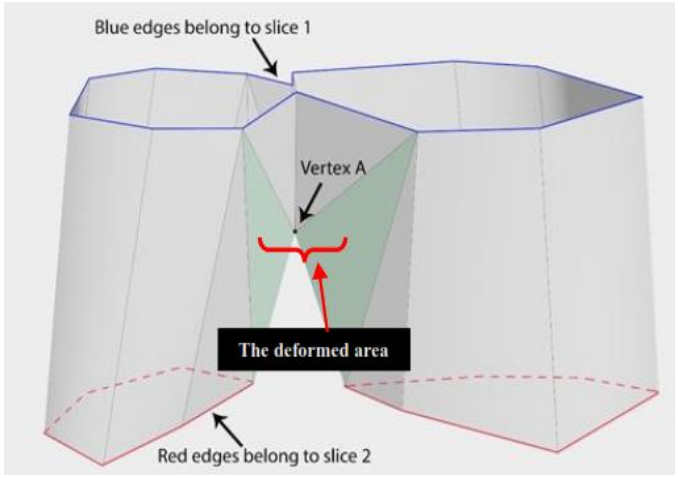
In our new method, the extraction of the contours and the use of the correspondence factor allows the construction of a 3D mesh without ambiguity, and directly in the form of quadrilaterals.

In fact, after extracting the contours in the 2D images, the correspondence factor for the contours of each image N is calculated with the image N+1. This factor makes it possible to indicate the contours that match each other unambiguously. Also, the construction of the basic 3D mesh is done by representing each contour by a triangle. Then, the triangles of the contours that correspond to each other are connected. This connection results in a mesh in the form of quadrilaterals (Fig. 21).

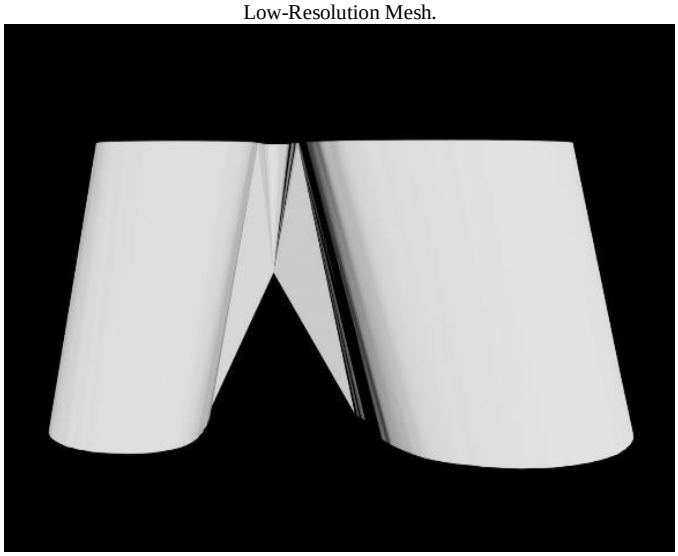
We also note that our new method incorporates an improvement made to this article in terms of displacement vectors [4] (Fig. 22).

Note that the mesh obtained with high resolution corresponds to the real anatomical shape.

Subsequently, we tested our new method at the storage level. The algorithm used in the original method allows us to generate a basic mesh with many vertices. But with our 3D mesh construction method which is based on contour extraction and the representation of each contour by a triangle, we notice that the amount of storage is less important (Fig. 23 and Fig. 24).



Low-Resolution Mesh.



High-Resolution Mesh.

Fig. 20. The Modification Step Leads to Deformations in Complicated Areas.

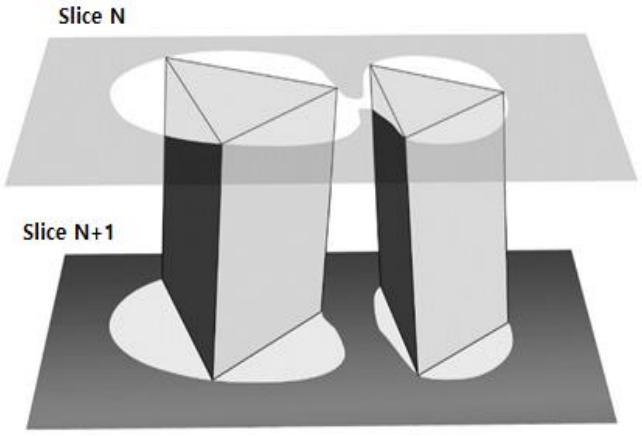


Fig. 21. A basic Mesh in the form of Quadrilaterals with our New Method.

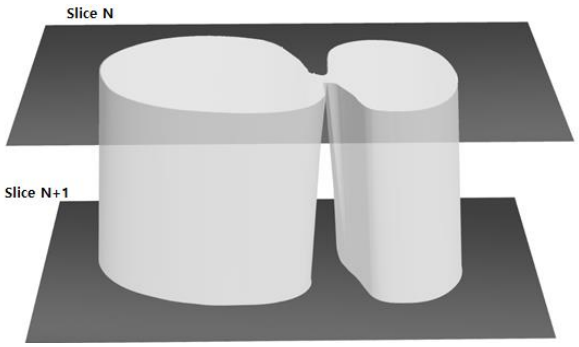


Fig. 22. The High-Resolution 3D Mesh Obtained after the Application of the Displacement Vectors by Integrating the Improvement.

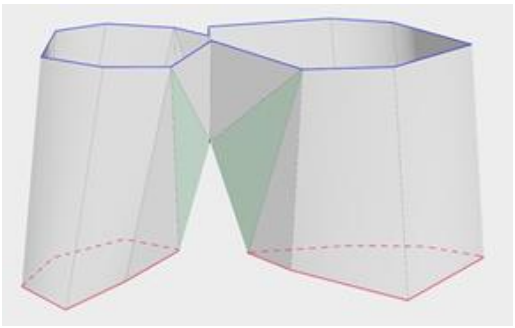


Fig. 23. Basic Mesh with the Original Method.

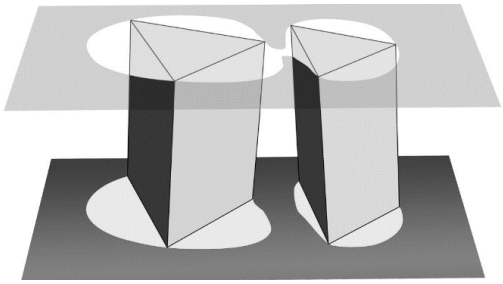


Fig. 24. Basic Mesh with our New Method.

If we compare the used memory space of the two 3D meshes between two 2D images (Table I), we find:

We worked on a medical volume consisting of 30 slices. With the original method, we obtained a basic mesh size of 40 501 bytes, and with our method, we obtained a basic mesh size of 37 675 bytes.

We can see from these calculations that our new method is optimized in terms of storage.

TABLE I. CALCULATION AND COMPARISON OF THE SIZE OF MESH IN THE ORIGINAL METHOD AND THE NEW METHOD

	Basic mesh between two 2D images	**Displacement vectors**	**Total:**
The original method	27 vertex *3 float(x,y,z) *4 octets	500 vectors *2 char(x,y) *1 octet	=1324 bytes
The new method	12 vertex *3 float(x,y,z) *4 octets	500 vectors *2 char(x,y) *1 octet	=1144 bytes

V. CONCLUSION

In this paper, we proposed an improvement of the algorithm that allows the reconstruction of the 3D mesh for a 2D medical image sequence, using low-resolution mesh and displacement vectors. The use of displacement vectors to add small reliefs on a basic mesh has proven to reduce the amount of information stored in the final 3D mesh. Also, the automatic generation of the displacement map speeds up rendering calculations using the GPU.

In this work, we modified the method used to build the basic mesh. This modification aims to eliminate the cases of ambiguity, particularly in certain types of objects which cannot be treated easily with the original method which uses the algorithm of Marching Cubes. Thus, it also makes it possible to obtain a low-resolution mesh directly usable during rendering without going through a modification step.

In our new method, we extracted the contour of the anatomy to be reconstructed for the 2D images, then we built a 3D mesh in minimal time, and optimized in terms of memory. Then we generated displacement vectors by discretizing the contour according to the level of detail we want to see.

Our method has proven itself on two main points:

- The reconstruction of the low-resolution 3D mesh using triangles that represent the contours provides a basic mesh without deformation problems that can be used in rendering.
- The edge detection method allows to overcome the ambiguity problems from which the Marching Cubes algorithm suffers.

REFERENCES

[1] J. Tan, J. Chen, Y. Wang, L. Li, and Y. Bao, “Design of 3D Visualization System Based on VTK Utilizing Marching Cubes and Ray Casting Algorithm,” 2016.

[2] Bücking, T. M., Hill, E. R., Robertson, J. L., Maneas, E., Plumb, A. A., & Nikitichev, D. I.,” From medical imaging data to 3D printed anatomical models. Plos One,” 12(5), 2017.

[3] W. E. Lorensen, "History of the Marching Cubes Algorithm," IEEE Computer Graphics and Applications, Vol. 40, No. 2, pp. 8–15, 2020.

[4] B.E. Said, M.O.C. Malki. and K. Abdelhak, “Improvement of the generation of displacement vectors in the reconstruction of a 3D mesh for medical images,” Int. J. Medical Engineering and Informatics, Vol. 9, No. 1, pp.20–21, 2017.

[5] S .Mady and M. El Seoud, “An Overview of Volume Rendering Techniques for Medical Imaging, ” International Journal of Online and Biomedical Engineering, Vol. 16, No. 6, pp 95–106, 2020.

[6] P. Visutsak, “Marching Cubes and Histogram Pyramids for 3D Medical Visualization,” Journal of Imaging, 6(9), 88, 2020.

[7] G. M. Nielson and B. Hamann, “The asymptotic decider: resolving the ambiguity in Marching Cubes,” in Proc. of IEEE Visualization, pp.83–91, 1991.

[8] B.K. Natarajan, “On generating topologically consistent isosurfaces from uniform samples,” The Visual Computer, Vol. 11, No. 1, pp.52–62, 1994.

[9] E. Chernyaev, “Marching Cubes 33: Construction of Topologically Correct Isosurfaces,” (No. CERN-CN-95-17), 1995.

[10] A. Lopes and K. Brodlie, “Improving the robustness and accuracy of the marching cubes algorithm for isosurfacing,” IEEE Transactions on Visualization & Computer Graphics, Vol. 9, No. 1, pp.16–29, 2003.

[11] G. M. Nielson, “On marching cubes,” IEEE Transactions on Visualization & Computer Graphics, Vol. 9, No. 3, pp.283–297, 2003.

[12] R. Shu, Z. Chen and M.S. Kankanhalli, "Adaptive marching cubes," The Visual Computer, Vol. 11, No. 4, pp.202–217, 1995.

[13] J. Congote, A. Moreno, I. Barandiaran, J. Barandiaran and O. Ruiz, "Extending marching cubes with adaptative methods to obtain more accurate iso-surfaces," Computer Vision, Imaging and Computer Graphics. Theory and Applications Communications in Computer and Information Science, Vol. 68, pp.35–44, Springer, Berlin, Heidelberg, 2010.

[14] B.E. Said, M.O.C. Malki, K. Abdelhak, and E. Abdelali, "Reconstruction of a 3D mesh with displacement vectors for medical images," Int. J. Medical Engineering and Informatics, Vol. 7, No. 3, pp.209–221, 2015.

[15] E. Lorensenw and H. E. Cline, "Marching cubes: a high-resolution 3D surface construction algorithm," Proc. of ACM SIGGRAPH, pp.163–169, 1987.

[16] O. Laligant, and F. Truchetet, "A nonlinear derivative scheme applied to edge detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, February, Vol. 32, No. 2, pp.242–257, 2010.

[17] J. Canny, "A computational approach to edge detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI, November, Vol. 8, No. 6, pp.679–698, 1986.

[18] S. Yi, D. Labate, D. Easley, and H. Krim, "A shearlet approach to edge analysis and detection," IEEE Transactions on Image Processing, May, Vol. 18, No. 5, pp.929–941, 2009.

[19] L. Zhang and P. Bao, "Edge detection by scale multiplication in wavelet domain," Pattern Recognition Letters, Vol. 23, No. 14, pp.1771–1784, 2002.

[20] J. Mairal, M. Leordeanu, F. Bach, M. Hebert and J. Ponce, "Discriminative sparse image models for class-specific edge detection and image interpretation," in Forsyth, D., Torr, P. and Zisserman, A. (Eds.): Computer Vision: ECCV 2008, Lecture Notes in Computer Science, Vol. 5304, pp.43–56, Springer, Berlin, Heidelberg, 2008.

[21] R. Mukundan, "Reconstruction of High-Resolution 3D Meshes of Lung Geometry from HRCT Contours," IEEE International Symposium on Multimedia, pp 247–252, 2016.

[22] D. Meyers, S. Skinner and A. K. Sloan, "Surfaces from contours," ACMTransactions on Graphics, 11(3), p. 228–258, 1992.