

Real-time external labeling layout algorithm for Direct Volume Rendering

Algoritmo de posicionamiento en tiempo real de etiquetas externas para (DVR)

Engr. Luis Guillermo Silva Rojas,^I Ph.D. Ramón Carrasco Velar^{II}

I Universidad de las Ciencias Informáticas, La Habana, Cuba. E-mail: lgsilva@uci.cu

II Universidad de las Ciencias Informáticas, La Habana, Cuba. E-mail: rcarrasco@uci.cu

SUMMARY

Illustrations used in technical and scientific texts often employ labels to correlate the graphic elements and their textual descriptions. Researchers have proposed several algorithms to determine the layout of the annotations on images rendered at interactive frame rates. Generally these layouts can be classified as internal or external. This paper proposes a new algorithm for locating external labels during the real-time direct rendering of volume data. The proposed algorithm uses only the rows of pixels corresponding to the labels anchor points, which optimizes the performance and facilitates its implementation, avoiding the computation of the convex hull for the generated image. Both, the overall visualization performance and the cost of the proposed algorithm are kept in real-time (60 fps) for medium size volumes (about 2563 voxels).

Key words: labeling, layout, real-time, volume rendering.

RESUMEN

Las ilustraciones utilizadas en documentos científicos y técnicos utilizan frecuentemente etiquetas para correlacionar los elementos gráficos y sus textos descriptivos. Los investigadores han propuesto diversos algoritmos para determinar el posicionamiento en tiempo real de las correspondientes anotaciones en las imágenes obtenidas en un marco interactivo. Generalmente estos posicionamientos se clasifican como internos o externos. Este artículo propone un nuevo algoritmo

para ubicar etiquetas externas en tiempo real durante la obtención de datos de volumen. El algoritmo propuesto usa solo las filas de píxeles correspondientes a los puntos de presentación de las etiquetas lo que optimiza el desempeño y facilita la implementación haciendo innecesarios algunos cálculos. Tanto el desempeño general de la vista como el costo del algoritmo propuesto se obtienen en tiempo real (60 fps) para volúmenes de mediana talla (alrededor de 256 voxels).

Palabras claves: etiquetado, posicionamiento, tiempo real, obtención de volumen.

INTRODUCTION

Illustrations with a careful positioning of texts on the images are used to describe the structure or for constructing complex objects, resulting in an important tool for developing learning materials such as textbooks, user manuals and technical specifications.

Although the labeling of technical and scientific illustrations have been used for centuries, its main function remains to facilitate the transmission of information, establishing correlations between the graphics elements of the illustration and their textual descriptions¹. The clarity of the relationship between these two basic components, determines the amount of information that a reader can extract from the content shown.²

With the development of three-dimensional techniques for data acquisition, such as Computed Axial Tomography (CT), Magnetic Resonance (MRI) and the posterior development of Direct Volume Rendering (DVR) algorithms at the end of the 80s,^{3,4} the need arose for labeling the obtained representations, balancing the quality of the label layout on the image and the computational cost to calculate these positions.

Medicine is one of the sciences that obtains more benefits from labeling algorithms, because medical illustrations are constantly used in anatomy, surgical planning and collaborative diagnosis. Therefore, modern radiological stations and medical training systems have functionalities for the manual labeling of anatomical structures.⁵

This paper proposes a new algorithm for the placement and visualization of external labels in real time, for DVR of medical images in the industry standard DICOM format. The algorithm uses only the rows of pixels corresponding to the labels anchor points, to avoid the computation of the convex hull for the generated image. The algorithm perform well in real-time for medium size datasets (about 256³ voxels).

1.1. Labeling algorithms

The labeling can be done interactively or automatically. The interactive labeling is done by specialists, e.g. radiologists, placing annotations on two-dimensional views of the dataset.¹ To facilitate this manual process, the images can be previously

segmented. Modern software offers lines, arrows and text labels to support this process. In the case of automatic labeling, the software automatically rearranges the labels and takes care of eliminating crossed lines, overlapping of labels and proximity between the labels and their linked structure.¹

1.2. General requirements for labeling algorithms

The requirements for the labeling algorithms used in three-dimensional illustrations have been previously defined in several approaches.^{6,7,1,8,9,10} These techniques can also be used in DVR implementations. Although the defined requirements vary slightly, they can be summarized as follows:

- Readability: Labels must not overlap. The font must remain legible.
- Unambiguity: the relationship between labels and their associated structures must be observed without ambiguity.
- Prevention of visual clutter: the lines connecting the labels should not cross.
- Interactivity: calculating the layout of the labels should not significantly affect the visualization performance.
- Compaction: minimize the area occupied by the labels.
- Temporal coherence during exploration: prevent discontinuities and breaks of labels positions.

The management of all the requirements is a complex task. This situation is aggravated by the interactivity, due to the constant computation each time a new image is rendered during the user exploration. Hartmann et al.⁷ propose some metrics to evaluate, in a certain degree, the functional requirements and aesthetic attributes of labeling algorithms. Finding the optimal solution to the labeling task without labels overlaps, is considered an NP-Hard problem.^{6,11,12,13} There are numerous implementations trying to balance the aesthetic constraints and the computational complexity.

There are different criteria for classifying labeling algorithms considering the label layout on the image. In general, they can be classified into:

- Internal: appropriate when there is enough space to place the labels.¹⁴ Have the advantage of easy visual association of the text and its structure, because they are superposed on the said structure.¹⁵ Its main disadvantages are that occlude part of the labeled structure and require a prior segmentation of the structures present in the image (Fig. 1a).
- External: used when there is enough white space outside the image¹⁴ (Fig. 1b) and are located on the outside or background of the image.¹⁵
- Hybrid: incorporates internal and external labels depending on the image magnification level. If an object is close enough to the camera, an internal label is used, otherwise an external one is used^{1,12} (Fig. 1c).

The proposed algorithm is designed for general purpose labeling of images obtained by DVR, regardless the content type of the image or its segmentation level. Having no segmentation information, the boundaries of the structures are unknown for the image, hence, it is necessary to place the labels on the outside, as shown in figure 1b.

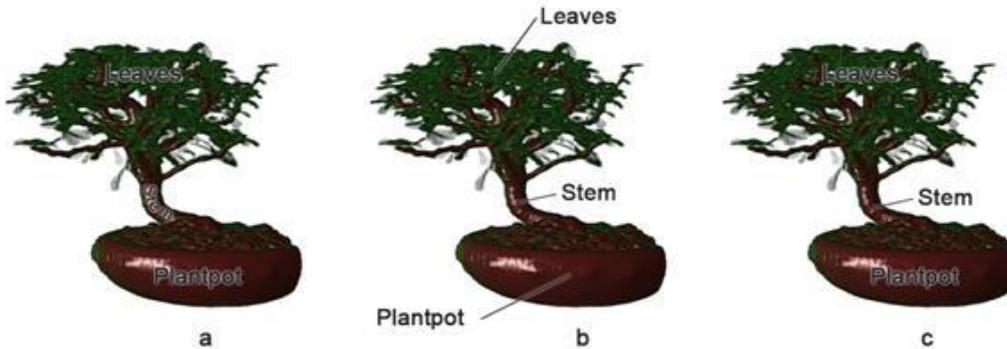


Fig. 1. Classifications according to the labels arrangement. (a): Internal labels, (b): External labels, (c) Hybrid layout

1.3. External labeling

External labeling algorithms are usually inspired by the traditional illustrative techniques used by illustrators. Ali et al.⁶ perform a manual analysis of numerous high-quality illustrations that uses external labeling and classifies its layouts according to their common properties:

- Straight-Line: Labels and anchor points are connected using straight lines, see (Fig. 2a,b,d).
- Orthogonal: The connecting lines are parallel to the coordinate axes and the bends are orthogonal angles, see (Fig. 2c).
- Flush-Layout: Labels are assigned to different spatial areas:
 - * Flush Left-Right: Labels are placed on the left and/or right of the graphical object, (Fig. 2a,b,c).
 - * Flush Top-Bottom: Labels are placed on the top and/or bottom of the graphical object, (Fig. 2d).

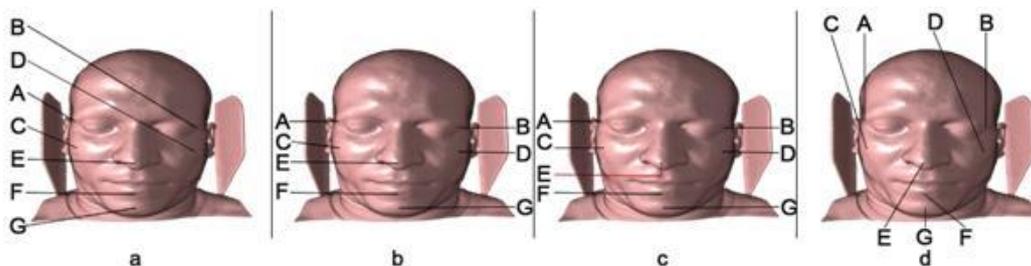


Fig. 2. Flush Layouts: a) Flush-Left layout. b) Flush Left-Right layout. c) Flush Left-Right layout with orthogonal bends. d) Flush Top-Bottom

- Circular-Layout: Labels are aligned around the silhouette of the graphical model in a circular fashion:
 - * Ring: The labels are placed at regular intervals on a ring circumscribing the graphical model, (Fig. 3a).
 - * Radial: Labels are located radially relative to a common origin, (Fig. 3b).
 - * Silhouette-based: Labels are placed near the silhouette of the graphical object, minimizing the distance with its anchor point, (Fig. 3c).

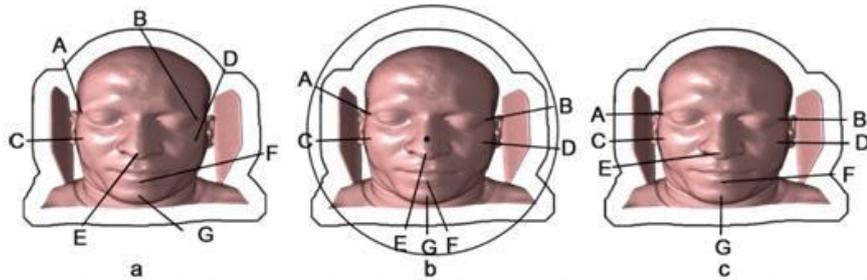


Fig. 3. Circular Layouts: a) Labels are arranged at regular intervals (Ring). b) Labels are radially arranged with a common origin (Radial). c) Labels are aligned to the image silhouette (Silhouette-based)

2. Related work

Preim et al.² propose an algorithm and its implementation in the ZOOM ILLUSTRATOR software that lets to put annotations around the image. The labeling remains consistent after rotate or scale the image. The level of detail of the labeling fits the available space, employing fisheye techniques.²

Moreover, the own Preim¹⁶ proposes an extension to the previous algorithm that modifies the labels and the structures, influenced by user interaction. Includes changes in material properties of the relevant structures to ensure its visibility and recognition. In both cases, the scene depicted consists of predefined polygonal models.

Hartmann et al.⁹ present the Floating Labels algorithm, a new method for determining an attractive arrangement of labels over complex objects. The algorithm uses Dynamic Potential Fields for the calculation of attraction and repulsion forces between the graphic elements and their textual labels. The algorithm works on 2D projections of 3D geometric objects from manually selected points of view. Calculating the labeling layout for complicated views, although it is visually appealing, it is of about 10 seconds, which precludes its use for applications that require interactivity.

Kamran and colleagues⁶ propose different layouts for placing external labels, which can be classified into two main groups: Flush Layouts, where labels are vertically aligned to the left and right or horizontally aligned up and down. The other classification is Circular Layouts (Fig. 3), where the labels positions conform to the shape of the rendered object. They use a general algorithm for all layouts, which is responsible for determining the positions of the anchor points and blank space regions. Then, the selected specific algorithm, determines the starting positions of the labels and calculates the final positions to meet the aesthetic restrictions.

Bekos et al.¹¹ introduce the Boundary Labels model. In this model, the labels are placed around a rectangle aligned with the coordinate axes that contains the anchor points. Each label connects with its anchor point by a polygonal line. They pay special attention to avoid lines crossing. The model is optimized for the labeling of maps.

Moreover, Timo et al.^{12,14} present a new architecture that combines internal and external labels on projections of complex 3D objects, balancing, in real-time, requirements that may be contradictory, such as unambiguity, readability, aesthetic considerations and temporal coherence during interaction. The architecture is divided into three modules: Analysis, Classification and Layout Manager. During analysis a color-coded projection of the scene is represented. This representation is

segmented, a skeletonization algorithm is applied and is transformed into a graph (skeleton graph). The best paths in the graph are then selected to place labels. The classification module takes as input the best graph path for each structure, which the authors consider is that the best you can fit the desired text, and classifies the labels as internal or external (internal labels are prioritized unless the space does not permit). Finally, the Layout Manager is responsible for determining the final location of the labels, build the connection lines for the case of external labels and ensure, as far as possible, the temporal coherence. Although this architecture is one of the most complete methods reported to date, its main disadvantage is the computational complexity that represents the real-time calculation of the skeletonization method.

Timo et al.¹⁷ presented a new layout, based on the report of Kamran⁶, to organize the labels in contextual groups. The position of a group of labels is calculated from the centroid of all visible members. After calculating the initial position, the group moves until fully located on the background or leave the screen. To maintain the temporal coherence, contextual groups remain in place until interfere with visualization. When this happens, the centroid and the new position for the group in conflict is recalculated. The intersections of the connection lines within groups are solved by exchanging the positions of their labels. Being based on the method of Kamran⁶, a skeletonization is also used to determine the label anchors points, which presupposes that the displayed image is segmented or color-coded.

Moreover, Cmolík and Bittner¹⁰ formulates labeling as an optimization problem with multiple criteria, which they solve using fuzzy logic with greedy optimizations. With the implementation of the method in GPU they achieve interactive time for polygonal models that the authors consider as medium.

Bruckner and Gröller¹⁸ propose the VolumeShop software, to generate interactive illustrations using DVR. The VolumeShop labeling algorithm approximates the object shape by calculating its convex hull, the obtained polygon is parameterized by its radius, therefore, the position of the annotations are defined by a number in the range [0,1]. The labels are located outside the convex hull using its parametric position. All entries are initially located at the closest point between the anchor point and the shape of the convex polygon. An iterative algorithm is repeated until all intersections and overlaps are resolved or the maximum number of iterations is reached. Due to the labels are initialized close to its anchor point, they move smoothly during the interaction. Only occurs jumps when the discontinuities and overlaps are resolved. Although the resulting positions are not optimal, the algorithm keeps interactive times for a practical number of labels (usually no more than 30 labels are used in an illustration) and obtains visually appealing layouts.

Generally, the above algorithms have two main disadvantages. The first one is the need for a prior segmentation of the dataset or at least a color-coded projection^{2,6,12,14,16,17} which is not suitable for all datasets. The second is the computational complexity of calculating the convex hull of the generated image^{6,18}, the skeletonization algorithm^{12,14,17} or directly the labels positions.⁹

3. Labeling algorithm

A group of previously well-defined constraints were taken into account during the design of the proposed algorithm, especially those outlined by Brucker and Gröller.¹⁸

1. Labels must not overlap.
2. The connecting lines between the labels and their anchor points should not cross.

3. Labels should not occlude the structures of the image.
4. Labels should be placed as close as possible to their anchor points.
5. Avoid discontinuities during the interaction.

In addition to this selection and compilation of restrictions, it was considered advisable to include the following:

6. The algorithm must do a general purpose labeling, that is, on any volume, regardless of content, nature or segmentation level.
7. The labeling must not significantly affect the visualization performance.

The specific algorithms presented in the previous section have substantial differences, but they share three basic steps:

1. Placement or calculation of anchor points.
2. Calculation of the labels initial positions.
3. Correction of intersections, overlaps and discontinuities.

3.1. Anchor points calculation

To be general purpose (restriction 6), the algorithm must be able to label any volume, therefore it is assumed that the obtained images are not segmented or segmenting makes no sense, meaning that the user should provide the anchor points and of course, the text of the labels. To facilitate this process three planes aligned with the coordinate axes (XY, YZ and XZ) were added, as shown to the left on the view of the developed software that is presented in figure. 4. The horizontal and vertical lines represent the projection of each plane on the remaining planes (XY: blue, YZ: red, XZ: green).

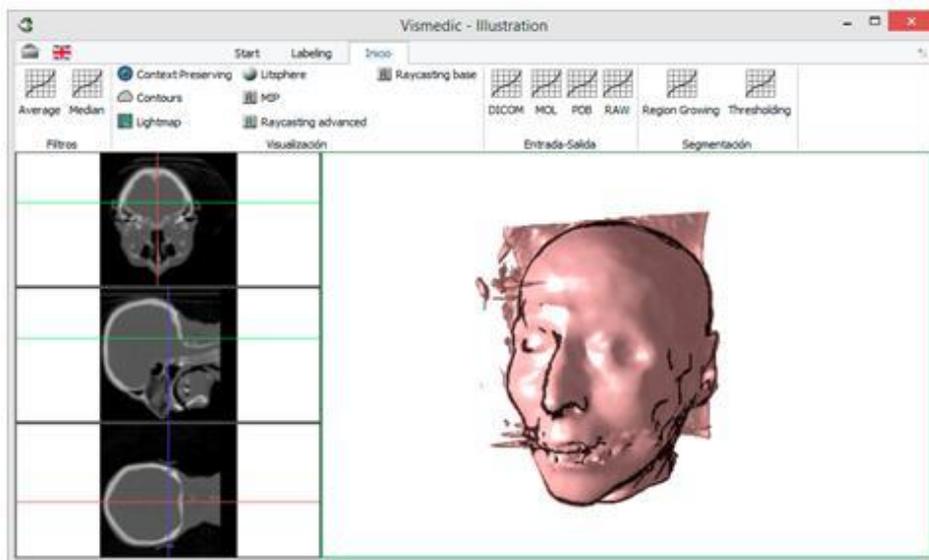


Fig. 4. Integration of the proposed algorithm with the software Vismedic-Illustration

The placement of a label can be done by pressing the right button of the mouse on any plane and write the text in the pop-up textbox. The label is automatically displayed on the 3D viewer.

3.2. Calculation of the labels initial positions

Direct Volume Rendering, as its name implies, uses no intermediate geometry (polygonal mesh) to obtain the dataset representation, therefore, the content of the final image is not known until the image is displayed. This image can vary widely for the same volume depending on the rendering technique and its parameters. Given these characteristics, the algorithm must work directly on the generated image, whose only information known are its pixels.

To determine the labels initial positions the following steps are taken:

1. Project the 3D positions of the anchor points on the 2D image, this is done by simply calling the **gluProject** function from the OpenGL library. The obtained 2D anchor points are ordered from highest to lowest by its **y** coordinate. The results are shown in Fig. 5a.
2. For each anchor point, analyze the row of pixels (coordinate **y**) to which it belongs to store the first position of the graphical object (**x₁**) and last (**x₂**). This is done by comparing the current pixel with the background. See Fig. 5b.
3. Determining the closest point (**x₁** or **x₂**), this would be **x₁** if $|x - x_1| \leq |x - x_2|$, or **x₂** otherwise. A small displacement to the left is used if the point is **x₁** and to the right if it is **x₂**, in order to slightly separate the labels from the image. If **x₁** is chosen the label is also shifted left considering the size of text on screen. See Figure. 5c.

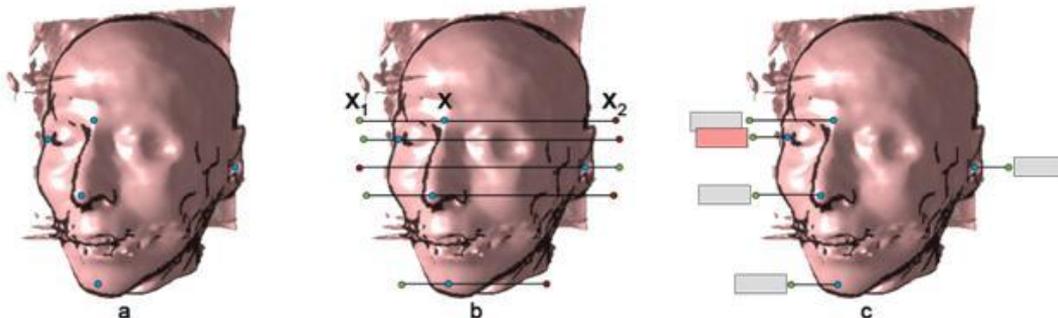


Fig. 5. Calculation of the labels initial positions. (a): Projection of the anchor points (b): Analysis of the row of pixels corresponding to the anchor point (c) Calculate the nearest position according to the silhouette

3.3. Correction of intersections, overlaps and discontinuities

At this point the labels are ordered by its **y** coordinate, we also know the height of the text boxes and if they belong to the left or right of the image. To represent the labels only need to compare the position of the current label with its predecessor, according to whether they belong to the left or right. In case of overlap (Fig. 6a) the current label is moved to the position determined by the end of the previous label plus a delta for separation. The end result is shown in Fig. 6b.

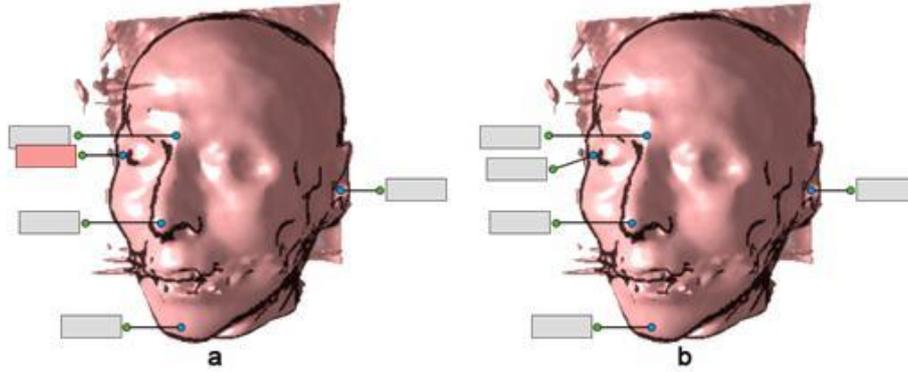


Fig. 6. Solution of overlaps

As the labels are ordered and their final positions are dynamically updated, no intersections occur during rendering. During the interaction, the algorithm behaves stably due to the proximity of the labels with their anchor points and the adaptability to the image silhouette. The discontinuities only occur when exists overlaps or the end positions change of side (left or right), in both cases only the involved labels are affected.

3.4. Implementation details

Although the proposed algorithm can be implemented using any programming language and graphics library, the authors recommend the use of C++ as programming language and the industry standard OpenGL as graphics library, the selection of these technologies favors the need for performance and portability.

The algorithm was integrated for this publication with the software Vismedic-Illustration, which also uses Qt framework for the graphical user interface and GLSL as shading language for the visualization algorithm.

RESULTS AND DISCUSSION

Among the main advantages of the proposed algorithm are its ability to label volumes of any kind (as shown in Fig. 7), not needing to compute the convex hull of the image, its simplicity of implementation and the performance to calculate the label layout.

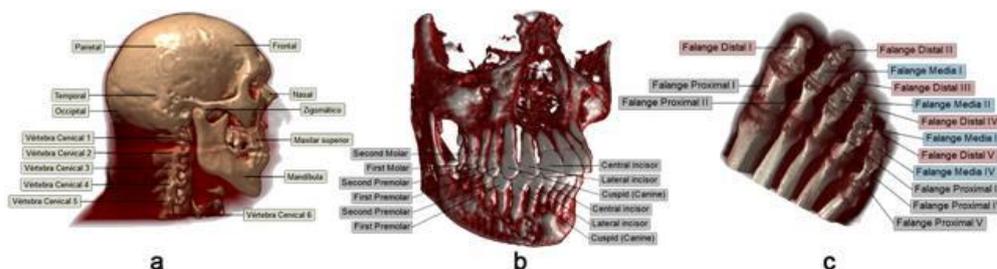


Fig. 7. Visual results for different volumes and number of labels.

Table 1 shows the results obtained by the proposed algorithm on medium size datasets. The tests environment consisted of a personal computer with an Intel Core i3 2120 processor, Nvidia GeForce GTX 285 as GPU, 4GB of RAM DDR3, at a resolution of 800x600 pixels. The render algorithm was a GPU based Raycasting

implementation for all the images. The used volumes are employed repeatedly in the literature to check the results of the volume visualization algorithms, except for the chemical structure of Menthol, which was obtained for this work using a voxelization from the 3D chemical structure of its molecule.

As shown in Table 1, the display performance for datasets of about 2563 is not significantly affected for a practical number of labels, it was kept in real time for all cases (about 60 frames per second) after the activation of the labeling algorithm.

The key to reduce the computational cost was the use of the pixel lines of the labels. For example, to compute the convex hull of a RGBA image of 8-bits and 800x600 pixels with 10 labels need to traverse 480000 pixels (1920000 bytes), however, using the proposed algorithm is only necessary to go through 8000 (32000 bytes), which represent the 1.66% of the total pixels in the image.

Table 1. Performance results for different datasets and number of labels

Dataset	Dimensions	Number of labels	FPS without labeling	FPS with labeling
Visible Male	128x256x256	14	62	60
Skull	256x256x256	12	62	60
Foot	256x256x256	14	62	60

CONCLUSIONS

This paper presents a new algorithm for the placement of external labels for DVR. The proposed algorithm works in image space, using only the rows of pixels corresponding to the labels anchor points, which optimizes the performance and facilitates the algorithm implementation, because it does not need to calculate the convex hull of the generated image. Both, the overall visualization performance and the cost of the proposed algorithm are kept in real-time (60 fps) for medium size volumes (about 2563 voxels), which ensures interaction during visualization.

REFERENCES

1. Oeltze S. and Preim, B. Survey of Labeling Techniques in Medical Visualizations. 2014. Eurographics Workshop on Visual Computing for Biology and Medicine.
2. Preim B, Ritter Alf, Strothotte T. Consistency of Rendered Images and Their Textual Labels. 1995. Proc. of CompuGraphics. Vol. 95, pp. 201-210.
3. Levoy M. Display of surfaces from volume data. IEEE, 1988, Computer Graphics and Applications, IEEE, Vol. 8, pp. 29-37.
4. Drebin RA, Carpenter L, Hanrahan P. Volume rendering. 1988. ACM Siggraph Computer Graphics. Vol. 22, pp. 65-74.

5. Preim, Bernhard and Botha, Charl P. Visual Computing for Medicine: Theory, Algorithms, and Applications. Morgan Kauffman, 2013.
6. Ali K, Hartmann K, Strothotte T. Label layout for interactive 3D illustrations. Václav Skala-UNION Agency, 2005.
7. Hartmann K, Götzelmann T, Ali K, Strothotte T. Metrics for functional and aesthetic label layouts. 2005. Smart Graphics. pp. 115-126.
8. Vollick I, Vogel D, Agrawala M, Hertzmann A. Specifying label layout style by example. 2007. Proceedings of the 20th annual ACM symposium on User interface software and technology. pp. 221-230.
9. Hartmann K, Ali K, and Strothotte T. Floating labels: Applying dynamic potential fields for label layout. 2004. Smart Graphics. pp. 101-113.
10. Cmolík L, Bittner J. Layout-aware optimization for interactive labeling of 3D models. Elsevier, 2010, Computers & Graphics, Vol. 34, pp. 378-387.
11. Bekos MA, Kaufmann M, Symvonis A, Wolff A. Boundary labeling: Models and efficient algorithms for rectangular maps. 2005. Graph Drawing. pp. 49-59.
12. Götzelmann T, Ali K, Hartmann K, Strothotte T. Form Follows Function: Aesthetic Interactive Labels. 2005, Computational aesthetics, Vol. 5.
13. Stein T, Décoret X. Dynamic label placement for improved interactive exploration. 2008. Proceedings of the 6th international symposium on Non-photorealistic animation and rendering. pp. 15-21.
14. Götzelmann T, Ali K, Hartmann K, Strothotte T. Adaptive labeling for illustrations. 2005. Proc. of 13th Pacific Conference on Computer Graphics and Applications, S. pp. 64-66.
15. Ropinski T, Praßni JS, Roters J, Hinrichs K. Internal Labels as Shape Cues for Medical Illustration. 2007. VMV. Vol. 7, pp. 203-212.
16. Preim B, Raab A, Strothotte T. Coherent zooming of illustrations with 3D-graphics and text. 1997. Graphics Interface. Vol. 97, pp. 105-113.
17. Götzelmann T, Hartmann K, Strothotte T. Contextual Grouping of Labels. 2006. SimVis. pp. 245-258.
18. Bruckner S, Gröller ME. VolumeShop: An Interactive System for Direct Volume Illustration. 2005. Proceedings of IEEE Visualization 2005. pp. 671-678. ISBN: 0780394623.

Recibido: 25 de noviembre de 2015.

Aprobado: 9 de marzo de 2016.