UNIVERSITY OF GRANADA

Department of Software Engineering

"*IDDG-Octree*. An Hibryd representation for the visualization and manipulation of volumes"

presented by

Alejandro José León Salas

for the degree of Doctor of Philosophy

Granada. December 19, 2008

UNIVERSITY OF GRANADA

Department of Software Engineering

"*IDDG-Octree*. An Hibryd representation for the visualization and manipulation of volumes"

presented by

Alejandro José León Salas

for the degree of Doctor of Philosophy

Dr. Juan Carlos Torres Cantero
Dr. Francisco Velasco Anguita
Advisors

Granada. December, 2008

This document is intended to be a guide, in English, that contains the main elements of the PhD. thesis document to which it is attached. Chapters have been summarized and the main contributions of the PhD. thesis are maintained. All references to figures, tables, equations, definitions, etc. are related to the PhD. thesis Document.

The framework of this thesis is the context of volume models. If something can be distinguished in these type of models it is that they try to characterize real objects through the representation of certain property values belonging to certain property domains. Therefore, with these kind of models we are interested in representing both the space occupied by the object and the change in values of the property domains of interest.

The goal of developing such models is to provide a set of software tools that permit study of the properties of the objects thus represented, displaying the information contained in such a way that the observer can understand that information and simulate the real object manipulation on the computational model. Figure 1 shows some examples of images that can be obtained from such models.

In general, a volume can be defined as a subset of 3D space whose points represent values of a property domain. A volume defined in this way can be represented by a function that matchs points of a subset V of the 3D Euclidean space in a property domain Γ:

$$f : V \subset \mathbb{R}^3 \mapsto \Gamma$$

However, when working with volumes that have been obtained in a discrete manner, we do not know the value of the function f throughout the volume of interest. We only have available the set of samples with the associated values of f. What is often used in these type of volume models is an estimate F of the function f obtained from the property values of the samples.

Nowadays, due to the large number of samples that are generated by the devices of volumetric information scanning and also to the technological improvement of computers, the amount of information that it is necessary to represent in volume models is increasingly growing, so it is necessary to provide models that are capable of reducing this amount of raw information provided by such equipment.

This work proposes a representation scheme that reduces the amount of information needed to represent the volumetric data. In addition, a volumetric representation that allows the visualization and manipulation of such data is proposed. Chapter 1 shows the context of volume visualization, in which our work is included. Chapter 2 shows a representation that combines the advantages of the two main approaches used in the discrete volume representations, voxels and cells. Chapter 3 describes our proposal for volume representation and evaluates its characteristics with respect storage space saving, processing time and quality of the isosurfaces extracted from it. Chapter 4 shows an application of our representation to virtual sculpture, in which its ability to allow interactive response times during the application of modeling operations can be seen. Finally, Chapter 5 presents conclusions of our thesis and future research paths.

Part of the work outlined in this PhD. thesis has been published earlier in the following works: short communication in the Ibero-American Symposium in Computer Graphics (SIACG 2006) [LTV06], communication in the Congreso Español de Informática Gráfica (CEIG'07) [LTV07a], two chapters in the book publishing research results for the research project of the Commission for Science and Technology of the Spanish Government encoded as TIN2004-06326-C03-02 [LTV07b,LTVS07], an article in the issue number 4 (August 2008) of Computer & Graphics [LTV08] and a communication in the Congreso Español de Informática Gráfica (CEIG'08) [LVS08].

This work has been funded with two research projects of the Comission for Science and Technology of the Spanish Government encoded as TIN2004-06326-C03-02 and TIN2007-67474-C03-02 and with the project of excellence of the Andalusian Government encoded as PE-TIC-401.

Alejandro J. León Salas

CHAPTER 1

A Review of Volume Modeling

Chapter 1 shows the volume visualization framework including concepts, terminology and the main problems that arise when working with volume data. It references and discusses the most outstanding earlier works which in our opinion deal with such problems.

Without intending to cover the large number of work on the theme, our aim is to establish a nutshell that enables exposure of terminology and raises the issues that arise in this area. The section *Solid Modeling* describes the methods of representation of homogeneous objects. Models of this kind permit us to establish clearly the properties that are required for modeling real objects and, these models, in some cases with some additional extensions, permit us to represent the inhomogeneity inherent in volume models. The section on *Representation Schemes* establishes a formal framework upon which modeling of objects is performed and presents some desirable properties for such schemes. Section 1.4 describes the various strategies that are used to represent a solid which can be applied through small extensions to volume modeling.

Section 1.5, *Volume Modeling*, presents the main elements that are required to establish a volume model. There are two major strategies for volume modeling: continuous representations and discrete representations, and in particular we emphasize the latter ones, since this thesis develops a representation that falls into that second group. In the next section, visualization has been specially distinguished from other operations, as it is that which has received special attention in the bibliography. This section presents the two major strategies used to address visualization and makes special emphasis on contouring, as this has been the strategy used for visualizing our representation.

CHAPTER 2

Multiresolution Representation of Volumes

A discrete volume representation must be capable of calculating the property value associated with any point $\boldsymbol{P}$ included in the volume bounded by the positions of the samples. The incessant increase in the resolution of the sets of samples produces representations needing high storage requirements. The fixed-resolution discrete representations maintain all the sample information, making their use unfeasible for sets with many samples of high resolution. The alternative is to build discrete representations of variable resolution (multiresolution representations) from the samples, using for this purpose a mechanism to reduce the amount of information necessary for storing.

The goal of the representation proposed in this chapter is to save space with respect to a volumetric representation of fixed resolution, taking into account the spatial coherence among the values of the samples. The idea is to reduce the amount of information stored in the regions of the initial set of samples that may be considered homogeneous.

This chapter shows the main approaches used when building discrete representations, both as fixed or variable resolution. The problems posed by the latter and our proposal for a multiresolution representation. The section *discrete volumetric representations* characterizes these type of representations, then presents a possible definition general enough to cover fixed and variable resolution representations. Next, it describes the main elements of a fixed-resolution representation under the terms of the two main work approaches: voxel and cell. Following this, we study the continuity provided by both approaches and the basic operation making it possible to estimate the property value at any point $\boldsymbol{P}$ included in the volume represented. The section ends presenting variable-resolution representations. For these representations we examine the use of the two approaches previously mentioned on these type of representations and highlight the particularities of each of them on a octree, which acts as data structure for supporting

them. Finally, it compares the two approaches with respect to its ability to reduce space requirements and the continuity provided for calculating the property value at any point $\boldsymbol{P}$ included in the represented volume.

Section 2.3 shows the work that has been undertaken in reducing the space required by the discrete representations. There are two areas: volume compression and volume simplification. Section 2.4 presents our proposal for multiresolution representation based on the concept of the homogeneous region which exploits the spatial coherence of property values. Firstly, the concept is described and then is formally defined on a uniform grid of samples. Then, it is applied in constructing our representation: *HRB-Octree*. Next, we compare our representation with the traditional approaches, voxels and cells, to show its advantages and drawbacks. Finally, section 2.5 explores the error produced when calculating the property value at a point using our representation, compared to using a fixed-resolution representation based on the cells approach. It is shown that the error only depends on the error tolerance set in the domain of property values, an further, an analytical expression which determines that error is provided. Next, we summarize the main sections of the chapter.

2.1. Volume Representations

Merely a brief introduction.

2.2. Discrete Volume Representation

The two main features of the discrete volume representations are the discrete nature of the data from which they are built and the layout or spatial structure of the data. The first feature generally requires some type of interpolation. The second feature permits classification of these representations both regular and irregular. We may define a *discrete volume representation* of an object as a finite set S of pairs of values $(\boldsymbol{P_i}, v_i)$, where v_i represents the value of an object's property located at position $\boldsymbol{P_i}$ in 3D space. The pair may represent samples of any property of an object taken at different positions, or evaluations of a known function f in a given volume of space. The domain where v_i values are included may be a single set of values (single-valued) representing a unique property of the volumetric object; the Cartesian product of several sets of values representing several properties of the volumetric object, and even, v_i may be a vector. In addition, in order to allow the calculation of the property value across the space taken up by the represented object, a discrete representation must provide a way of interpolating among pairs of values $(\boldsymbol{P_i}, v_i)$. Next, we propose a more formal definition of discrete volume representation:

Definition 1 (Discrete Volume Representation) *A discrete volume representation is a pair consisting of:*

- *A finite set of samples, S, whose elements are pairs $(\boldsymbol{P_i}, v_i)$. The first element of the pair represents a position in 3D space and the second represents*

a property value that belongs to a property values domain, (V). In addition, each point $\boldsymbol{P_i}$ must have a single property value v_i.

- *An interpolation function, F, to estimate the property value at any point within the space bounded by the positions represented by the $\boldsymbol{P_i}$ of S.*

$$\begin{aligned}\mathcal{RV}_d \equiv \quad & (S \equiv \{\boldsymbol{P_i} : \mathbb{R}^3; f(\boldsymbol{P_i}) = v_i : \mathcal{V} \mid \boldsymbol{P_i} \neq \boldsymbol{P_j} \; \forall \; i \neq j \; \wedge \\ & \wedge \; i, j = 0, 1, \ldots, n \; \cdot (\boldsymbol{P_i}, v_i)\}, \; F : \mathbb{R}^3 \mapsto \mathcal{V})\end{aligned}$$

where $f(\boldsymbol{P_i}) = v_i$ represents the case in which the function from which the samples have been taken is known, although there may be cases in which this function is not known, so that only the value v_i corresponding with each sample is available.

2.2.1. Fixed-resolution discrete representations

This subsection shows the two major interpolation functions that are used to calculate the property value in discrete volume representations: zero-order interpolation and trilinear interpolation. Each of them determines the two main approaches, voxels and cells, used to construct discrete volume representations. In these kind of representations, the set of samples S is laid out in a uniform manner. This way, points $\boldsymbol{P_i}$ determine a uniform grid into the space taken up by the representation.

2.2.2. Calculating the property value at a point in fixed-resolution representations

This subsection shows how we estimate the property value of any point $\boldsymbol{P}$ included in the space associated with a discrete volume representation for both approaches voxel and cell. We assume the data structure that supports the representation is a *Volume Buffer*. It is necessary to define a function τ to obtain the samples of S that are needed to perform interpolation. These samples depend on the subvolume (voxel or cell) that includes $\boldsymbol{P}$. An efficient method of identifying the subvolume that includes $\boldsymbol{P}$ is proposed for each of the approaches.

Figure 2.4 shows the shape of the interpolation function in the voxel approach (top image) and in cell approach (bottom image) for the 1D case. It can be seen that the function for the voxel approach presents discontinuities between neighboring voxels, while the function for the cell approach is a continuous function with C^0 continuity.

2.2.3. Variable-resolution discrete representations

In these representations the set S is not uniformly distributed. Therefore, subvolumes that make up the partition of the space associated with S present a variable size, unlike fixed-resolution representations in which they only have a

single uniform size. This subsection discusses the advantages and disadvantages of both approaches, voxel and cell, with regard to:

- Memory space saving for avoiding the storage of oversampled regions.
- The shape of the interpolation function for property values.

For the disccusion it is assumed that the function $f(x)$ that distributes property values over the volumetric object is unknown. It is therefore necessary to undertake a process that determines which regions in the volume are oversampled. For each region of this kind, we associate a larger subvolume that represents the subvolumes of higher resolution which are related to this region of similar property value. This variation in the size of the subvolumes that make up such representations justifies the need to use a hierarchical spatial structure to support them. In this subsection the choice of an octree is justified, displaying its advantages and disadvantages.

We study the behavior of the octree with regard to memory saving, considering the saving of samples produced by the aggregation of neighboring voxels (or cells) that present a similar property value, along with the restrictions which the octree imposes. It should be noted that subvolumes associated with an isotropic set have cubic shape and the ones associated with an anisotropic set have a parallelepiped shape. However, although its shape varies, the number of samples of the set S associated with each subvolume simply depends on the approach used. It should be remembered that in the voxel approach the number of samples associated with a subvolume is just one, and this number is equal to eight in the cell approach, regardless of subvolume size. Therefore, the study of the number of samples that are saved by adding neighboring subvolumes of higher resolution into one of lower resolution is independent of the type of geometry imposed by the distance between samples.

With the aim of saving storage space, when a region consisting of eight neighboring subvolumes having a similar property value is detected, removal of samples can be done. The number of samples which we can avoid storing in the new subvolume is expressed by the following equations, depending on the approach used:

$$\text{Voxel approach} \quad : \quad n_S = n^{dim} - 1 \tag{2.1}$$

$$\text{Cell approach} \quad : \quad n_S = (n-1)^{dim} \tag{2.2}$$

where n is the number of samples per dimension of the new subvolume and dim the dimension of the space in which we are working. In this case $dim = 3$. n_S indicates the number of samples that are eliminated, allowing the storage space that they were taking up to be saved.

Figure 2.6 shows this fact graphically for case 2D. It can be seen that the number of samples that may be saved using the voxel approach is **three**, as the space taken up by one of them is used to store the property value associated with

the new voxel. However, in the cell approach only the space taken up by **a single** property value can be saved. It is necessary to keep the storage space taken up by the rest of samples, as these samples are needed to permit the possibility of calculating the property value of any point $\boldsymbol{P}$ included in any of the neighboring cells of higher resolution.

The explanation lies in the fact that, with the voxel approach, the subvolume contains the sample in its interior, while in the cell approach, samples are shared between adjacent subvolumes. Therefore, the number of samples which the aggregation of subvolumes allows us to save in the voxel approach is independent of the size of adjacent voxels, and is determined by equation 2.2. However, the space saving that is taken up by the samples in the cell approach, expressed by the equation 2.3, depends on the size of adjacent cells. In fact, this equation is a pessimistic estimate, since all adjacent cells have a smaller size. Whenever the size of adjacent cells is less than that of the new cell, then all the samples must be kept. The greatest saving that can be achieved in the cell approach occurs when the six face-neighboring cells, together with the twelve possible edge-neighboring cells, have a size greater than or equal to that of the new cell. The vertex-neighboring cells have no influence, since the storage space of the shared vertex can never be saved, regardless of the size of these cells. In the most favorable situation, the number of samples that are saved is given by the following equation:

$$\text{Cell approach} \quad : \quad n_S = (n+1)^{dim} - 2^{dim} \tag{2.3}$$

where n is the number of higher resolution cells added per dimension and dim the dimension of the space in which we are working. In this case $dim = 3$. n_S indicates the number of samples that are eliminated, allowing the storage space that they were taking up to be saved.

As a result, the cell approach provides only an upper and lower bound in the number of samples that we can avoid storing. In addition, as shown in Figure 2.7, the cell approach penalizes the requirement for space saving because there are property values (or references to them) that must be stored more than once in the leaf nodes of the tree, such as the four surrounded by a red circle shown in the figure. This redundancy produces a waste of storage space.

2.2.4. Calculating the property value at a point in variable-resolution representations

This subsection shows that in variable-resolution representations the interpolation function for the voxel approach remains discontinuous between neighboring voxels. However, as shown in Figure 2.8, in the case of these representations, the function for the cell approach may be discontinuous on the border between cells of different sizes.

2.3. Previous work in space saving

In this section we comment on the preliminary work related to space saving in discrete volumetric representations.

2.4. Our representation proposal

Considering the advantages and disadvantages presented by the voxel and cell approaches with regard to space saving and the continuity of the function that calculates the property value for any point in the volume, our multiresolution discrete representation establishes a compromise between the two.

The aim is to start from a uniform discrete representation and to achieve as a result an irregular representation of the same data, with a loss of information bounded by an error tolerance. The resulting irregular representation does not store redundant information at regions where it is unnecessary. The highest resolution voxels contained in these regions are replaced by lower resolution voxels. In addition, it is our intention that the resulting representation shows a property values distribution as near as possible to the initial uniform discrete representation. Our representation is based on an octree whose leaf nodes represent voxels of variable size.

2.4.1. Homogeneous regions

This subsection is a formalization of the concept of homogeneous region in a discrete volume representation with scalar property values.

Definition 2 (Homogeneous regions) *Given a finite set S composed by pairs of values $(\boldsymbol{P_i}, v_i)$, where v_i represents the value associated with any scalar property of the object located at position $\boldsymbol{P_i}$ in 3D space, a* homogeneous region of S *is defined as a subset $U \subseteq S$ whose elements satisfy the following conditions:*

1. *All the elements in U make up a connected set on the 3D grid structure of S.*
2. *The similarity criterion is the equality, i.e. the property value v_i of each element in the set U is the same.*

$$U = \{(\boldsymbol{P_i}, v_i) \in S \mid \forall\ (\boldsymbol{P_i}, v_i),\ v_i = K\} \wedge U \textit{is connected}$$

Definition 3 (Quasi-homogeneous region) *Given a finite set S composed by pairs of values $(\boldsymbol{P_i}, v_i)$, where v_i represents the value associated to any scalar property of the object located at position $\boldsymbol{P_i}$ in 3D space, a* quasi-homogeneous region of S *is defined as a subset $U \subseteq S$ whose elements satisfy the following conditions:*

1. *All the elements in U make up a connected set on the 3D grid structure of S.*

2. *The similarity criterion is a numerical error range in the property values domain, so that the difference between any two property values for the elements in the set never exceeds a threshold* ϵ.

$$U = \{(\boldsymbol{P_i}, v_i), (\boldsymbol{P_j}, v_j) \in S \mid \forall\ (\boldsymbol{P_i}, v_i), (\boldsymbol{P_j}, v_j), |v_i - v_j| < \epsilon\} \wedge U \textit{is connected}$$

Next, the concepts N_6-Neighborhood, 6-adjacency, and connectivity on grids are defined because they are used in the definitions above. These concepts are illustrated graphically in Figure 2.9. Figure 2.10 shows a 2D example that illustrates the constraints imposed by the octree on the shape of the homogeneous regions it may represent. These constraints are due to the type of spatial subdivision carried out.

2.4.2. Homogeneous regions with edge *Octree*: *HRB-Octree*

This subsection describes our representation proposal. Our idea is to combine an octree whose leaf nodes represent voxels and a way of calculating the property value based on the cell approach, but applied to those voxels, so as to achieve continuity in the distribution of values.

To achieve this goal we will establish a partition in the space occupied by each voxel, considering two well differentiated areas. All voxels have an *outer zone or edge* with a thickness $\frac{\Delta_x}{2}$, $\frac{\Delta_y}{2}$, and $\frac{\Delta_z}{2}$ for x, y, and z dimensions respectively. The highest resolution voxels are made up only for this type of zone. Voxels representing homogeneous regions generated from aggregation operations are made up by an edge zone and an *inner zone* which corresponds with the rest of its volume. Figure 2.11 illustrates this partition of the space taken up by each voxel graphically. The inner zones are colored in green, and the edge zones in pale blue. It can be seen that the highest resolution voxels which are located at the bottom right lack inner zones.

Using this partition, when the calculation of the property value of a point $\boldsymbol{P}$ in the volume is performed, firstly, we must determine whether this point is included in an inner zone or in an edge, and then, depending on the type of zone the property value is estimated by zero-order interpolation (inner zone) or trilinear interpolation (edge).

Proceeding this way, a distribution of property values which presents C^0 continuity between neighboring voxels with different size is achieved. In the central part of the figure 2.13 the two types of zones are shown, inner and edge, colored in green and pale blue respectively, along with the shape of the distribution function for property values in 1D. The picture located at the top shows the distribution of property values for a voxel of the highest resolution and a double-sized one obtained as a result of carrying out an aggregation operation of two neighboring voxels of the highest resolution. The distribution of property values for the two voxels before the aggregation operation is shown in grey. The picture located at the bottom shows the distribution of property values for a cell of the highest resolution and another one with twice its size that results from eliminating the

sample shared by the two aggregated cells of the highest resolution. In the resulting cell the distribution of property values assigned by interpolation in each of the smaller cells appears in grey.

In the central image the distribution of property values provided by our representation can be seen. Among the highest resolution voxels a linear interpolation is carried out taking into account that the associated property value is located at its center. In the larger voxel case, a linear interpolation is undertaken if the point belongs to the interval between the center of the highest resolution voxel and the border of the larger voxel, that is, if the point falls on the edge (pale blue). The other end of the interpolation is the center of the corresponding neighbor voxel of the highest resolution. Whenever the point lies in the inner zone the property value stored in the voxel of greater size is assigned directly. This zone is shown in green in the picture.

Figure 2.14 shows the distribution of property values that has a quadtree based on the cell approach (top) and a quadtree based on the voxel approach (bottom) with the same number of subvolumes partitioning the space and the same size for each of them. The lines in light red and dark red represent the distribution of property values that is provided by the trilinear interpolation (bilinear in the figure), in case of the cell approach, and zero-order interpolation, in case of the voxel approach. In Figure 2.15 you can see the distribution of property values provided by the *HRB-Octree*.

The main advantage of our representation compared to the cell approach used in an irregular representation, in relation to the calculation of property value at a point $\boldsymbol{P}$, is the absence of the lack of continuity problem in geometric elements (edges and faces) shared by neighboring cells of different sizes. On the other hand, the main drawback here is when calculating the property value for a given point in volume. To carry out this operation, the cell approach permits fast identification of the cell ($O(n\ log\ n)$) in which the point lies, and how it stores all eight property values needed to carry out the interpolation in the corresponding leaf node, the calculation of the property value is immediate. However, our method only improves on the cell approach when the point is included in the voxel's inner zone, as the stored property value is returned directly, without having to carry out any kind of interpolation. When a point lies in the voxel's edge, our method uses much more time to get the property value at the point. Next, we show a concrete example of the distribution of property values that provides the cell approach, the voxel approach, and the *HRB-Octree* using the same similarity criterion for the octree (See figures 2.17, 2.18, and 2.19 respectively).

Our representation provides a distribution of property values by a continuous function (C^0 continuity) that combines a zero-order interpolation used by the voxel approach and a trilinear interpolation used by the cell approach. Nevertheless, the trilinear interpolation applied in our representation is free of the inconsistency problem described above for irregular representations based on the cell approach and therefore it does not need to elaborate partial solutions to this problem. In figure 2.19, the picture at the bottom shows the distribution of property values provided by our representation. The partition of voxels in inner and edge zones

appears colored in green and pale blue respectively. Property values that correspond to points lying in inner zones present a plane appearance because they are assigned by zero-order interpolation. However, the property value of points lying in edge zones is calculated by trilinear interpolation (bilinear interpolation in the example shown in the figure) from the values of the neighboring voxels. In this way, we have the advantage of saving space by applying the concept of homogeneous region to a voxel approach and the advantage of obtaining a continuous function as is achieved in the cell approach. The added advantage is that in our hybrid approach is not necessary to address explicitly the inconsistency problem posed by the cell approach on irregular grids.

2.5. Evaluating our representation

This section compares the *HRB-Octree* to a representation based on the cell approach applied on a uniform grid, which is the most commonly used (See figure 2.19). The metric that has been used for the comparison between representations is set out in the space of property values through a standard measure of error. The deviation produced when calculating the property value at a point in space by *HRB-Octree* is considered with respect to the calculation of the property value in the same point by the cell approach. To establish the measure of error we take as the correct value the one provided by the cell approach on every point and, as the estimated value the one provided by *HRB-Octree*.

2.5.1. Preliminary study of interpolation error

This subsection discusses the shape of the linear interpolation with the aim of carrying out a glance at the error produced in calculating the property value in *HRB-Octree*, with respect to calculating the property value on the cell grid (See equation 2.5). The analysis is carried out taking into account how the similarity criterion used in the operation of adding voxels (See inequalty 2.6) affects the linear interpolation, as well as the allocation of the average value of the voxels that make up the new voxel.

It is noted that the higher the average difference between pairs of property values is, the greater the error produced by using the *HRB-Octree* is, this error depends on the final value assigned to the new homogeneous region. Therefore, the greater the distance between property values allowed by the similarity criterion is, the higher the error obtained. It should be borne in mind that in *HRB-Octree*, the homogeneous regions (voxels) of different size are represented by its leaf nodes (See figure 2.20). It is also shown that the *HRB-Octree* guarantees that linear interpolation remains unchanged regardless of the width of the interpolation range $[x_i, x_{i+1}]$. This is because the linear interpolation is performed only on the voxels' edges (See figure 2.21).

Therefore, we can conclude that the error in calculating property values in the *HRB-Octree*, with respect to a uniform grid of cells, depends solely on the difference in the property values of the highest resolution voxels with respect to

the property value allocated to the new lower resolution voxel that results from an aggregation operation.

2.5.2. Error metric

This subsection develops an analytical error metric based on the above observations (See equations 2.17 and 2.18). This metric provides an error function, $E(x)$, which represents the error obtained in calculating the property value of any point lying in the volume represented by the *HRB-Octree*, with respect to the value calculated for this point using the uniform grid of cells. Figures 2.22 and 2.23 illustrate the parts that define the error function $E(x)$ graphically.

2.6. Conclusions

Fixed-resolution and variable-resolution discrete representations have been presented using the two main approaches, voxel and cell, which are based on the type of interpolation function used by the representation. In addition, the advantages and disadvantages of using each approach in variable-resolution representations have been studied, emphasizing the possibility of saving space for storing samples and the continuity provided by the interpolation function in each approach.

A representation based on the voxels approach, the *HRB-Octree*, has been presented which helps to reduce storage space using the concept of homogeneous region, whose idea underlies the use of the spatial coherence among property values of the samples. This representation resolves the problem of lack of continuity that presents the classic voxels approach by means of the establishment of a partition in the voxels space, dividing it into two areas: inner zone and edges. This new partition uses an interpolation function that is different for each type of area, achieving continuity C^0 among areas and, therefore, in the voxels partition.

The error produced by our representation with respect to a representation of fixed resolution based on the cell approach has been studied. It has been shown such error in calculating the property value of a point $\boldsymbol{P}$ solely depends on the error tolerance chosen for the property values domain. Based on this result, an analytical expression of the error has been provided.

CHAPTER 3

Visualization by Contouring in Discrete Volume Representations

This chapter presents a method for displaying discrete volume representations using the strategy of isosurface extraction (contouring). Contouring requires a partition of cells that occupy the entire space represented in the volume model. Our aim is to visualize the representation based on homogeneous regions by contouring a grid of cells dual to the voxels grid represented in the octree.

The first section presents the key concepts about this type of strategy for volume visualization. Section 3.2 shows a series of concepts related to tiles on the plane and space that will enable us to define clearly the type of cell partition that we propose. The next section (3.3) briefly describes how the isosurface extraction on discrete representations may be addressed, for both fixed and variable resolution which use the two main approaches: voxel and cell (already mentioned in the previous chapter). In the subsection of previous works some of the works that have addressed the problem of extracting isosuperficies in multiresolution discrete representations are discussed. Particular emphasis is given to the problem of cracks (holes on the extracted isosurface) that may appear in the approximation of the isosurface and the various techniques used for their solution. At the end, this section shows the partition of cells that we propose in order to carry out the extraction.

Section 3.4 presents our proposed hybrid representation: the *Octree with an Implicitly Defined Dual Grid (IDDG-Octree)*. This representation adds the topology of a dual grid of cells to the *HRB-Octree* which permits to make contouring avoiding the possibility of getting holes in the extracted isosurface. Section 3.5 shows the process of extracting isosurfaces from our representation. Finally, sections 3.6 and 3.7 evaluate the representation with respect to space and processing time requirements and the quality of the extracted isosurface.

The bulk of the work developed in this chapter was published in [LTV08]. Therefore, we attach to this guide the full text of this paper. However, sections

corresponding to the evaluation of results will be commented upon here in greater detail because, for reasons of space, they did not appear as detailed in the article. Section 3.6 shows the results for *IDDG-Octree* with respect to storage space and processing time required. Section 3.7 evaluates the quality of the extracted isosurface.

3.1. Visualization by contouring

This section describes the problem of extracting isosurfaces applied to discrete volume representations, presenting a possible definition of isosurface for such representations which is based on our definition of discrete volume representation.

3.2. Preliminary concepts

This section defines some concepts that will be used in the rest of the chapter and explains their relationship with the two main approaches used for representing volumes discretely: voxel and cell.

3.3. Contouring in voxel and cell approaches

This section explores the possibilities, advantages and disadvantages associated with the extraction of isosurfaces from representations based on each approach, taking into account the previous definitions and the necessary discretization of space. Firstly we consider both approaches in the case of fixed-resolution representations, and thereafter, we consider the multiresolution representations, along with our representation (*HRB-Octree*).

Finally, we comment on the preliminary work associated with contouring multiresolution representations posing the problem of holes in the extracted isosurface and placing special emphasis on methods of solution based on the concept of duality.

3.4. Homogeneous regions *Octree* with Implicitly Defined Dual Grid: *IDDG-Octree*

Focusing on isosurface extraction, a rough method is established that allows the calculation of isosurface points without having to pay the costs associated with maintaining an additional structure of cells. To do this a *3-D mosaic of cells dual* to the 3D mosaic defined by the homogeneous regions (voxels) represented by the octree's leaf nodes is constructed, which will be stored implicitly in the representation. Consequently, the advantages in terms of space saving and independence of the neighboring homogeneous regions in the voxel approach are maintained, and at the same time, we avoid the loss of efficiency in the calculation of the property value at a point that was a disadvantage of the *HRB-Octree*. Figure 3.28 shows an example that illustrates our representation. (See [LTV08]).

3.5. Isosurface extraction from *IDDG-Octree*

Our objective is to extract an isosurface, defined by a property value (the threshold), from our variable-resolution discrete representation. In the representation we have defined implicitly the cells on which we are going to make the calculation of the triangles that approximate the isosurface. Firstly, the active cells are detected by traversing the octree and using the information provided by the responsibilities assigned to minimal voxels. Secondly, geometric meaning is attached to the implicitly defined cells based on the adjacent voxels that are required to produce valid cells. Then a modified Marching Cubes Algorithm is applied on each of the cells to get the mesh of triangles that approximates to the isosurface.(See [LTV08]).

3.6. Evaluating the representation

The most important characteristics of a volume representation are its memory usage and rendering time. The *IDDG-Octree* uses a hierarchical structure that can be used inmediately to index the volume (for instance storing the property interval in the internal nodes). This type of index has been used to accelerate marching algorithms at the cost of increasing the memory requirement [VT01]. Therefore, we do not want to test the use of the representation as a volume index (as it is obviously faster than Marching Cubes), but rather test it directly against the uniform grid with Marching Cubes. Accordingly, we do not make use of the octree as a spatial index. Our conclusion is that the behaviour of the proposed representation is similar to that of a uniform grid, both in terms of the space used and isosurface extraction.

We tested the method taking as the input grids generated from synthetic models and various volumetric data sets. For the synthetic models we used two different sampling approaches to determine the simplification ratio obtained by our octree. The first approach generates samples with equal property values within the object, while the second one produces property values which decrease with increasing distance from the position where the sample is taken to the centre of the object. Specifically, the synthetic models that have been used for tests are defined as follows, according to the strategy of discretization taken:

Let α_1 be a function to perform sampling in the binary domain $\{0, 255\}$, defined as follows:

$$\alpha_1(x, y, z) : \boldsymbol{V} \subset \mathbb{R}^3 \mapsto \{0, 255\}$$

$$\alpha_1(x, y, z) = \begin{cases} 255 & \text{si } r_i \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

Let α_2 be a function to perform sampling in the discrete domain $[0, 255]$, defined as follows:

$$\alpha_2(x, y, z) : \boldsymbol{V} \subset \mathbb{R}^3 \mapsto [0, 255]$$

$$\alpha_2(x, y, z) = \begin{cases} 255(1 - r_i) & \text{si } r_i \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

where V is a cube of length 2 centered on the origin and aligned with axes and r_i is defined as the synthetic model chosen as:

Esfera

$$r_1 = \sqrt{x^2 + y^2 + z^2} \tag{3.3}$$

Tronco

$$r_2 = \sqrt{x^2 + 2yz} \tag{3.4}$$

Silla

$$r_3 = \sin(xz) + y + \cos(xz) \tag{3.5}$$

Jarrón

$$r_4 = x\sin(x) + y\cos(y) + z\sin(z) \tag{3.6}$$

In order to compare the sensitivity of the representation against the size of the data set, several volumetric representations with different sampling frequencies were generated using both methods. For the approach that generates samples in a binary domain, we have chosen a homogeneity criterion based on an equal property value. In the case of sampling with a discrete domain of property values, a tolerance error value was used to test the equality among samples. All tests were carried out on a standard PC with an AMD 64 (2 MHz) processor and 2GB of RAM memory.

3.6.1. Results

The tables presented below show, from left to right, the first column with the different resolutions of the models. The next two columns show the number of internal and leaf nodes of the octree. The following two columns show the size (in *Kbytes*) required by the *IDDG-Octree* and the grid (Rej.). The last column expresses the percentage of space requiered by our representation with respect to the uniform grid (*IDDG-O*/Rej.). The number of internal and leaf nodes of the octree are shown separately because the space occupied by each type of node is different. The property value in the regular grid uses 2 bytes while in the octree, each internal node requires 4 bytes and each leaf node requires 2 bytes for the property value and 1 byte for storing responsibilities (one it for each possible responsibility).

Tables 3.1–3.3 show the results for the space occupied by the model (esfera). The sampling strategy used produces values in the range 0–255. Table 3.1 shows the results obtained using as similarity criterion an error tolerance in the difference between property values $\epsilon = 0$. This is an example that serves to show the

worst case where low compression is achieved in the initial set of samples as the similarity criterion chosen is the equality among property values. However, it can be seen that as the number of samples increases the rate of reduction of required space improves. Similarly, tables 3.2 and 3.3 show the results for the same sampling strategy, but using the error tolerances $\epsilon = 1$ and $\epsilon = 2$ respectively. Obviously, increasing the error tolerance increases the rate of space reduction for smaller sized sets of samples.

Similarly, the tables 3.4–3.6, 3.7–3.9 and 3.10–3.12 show the results for space occupied by the models: **tronco**, **silla** and **jarrón** for similarity criteria with error tolerances $\epsilon = 0$, $\epsilon = 1$ and $\epsilon = 2$ respectively. In the tables we can check the trend of reducing space requirements as the number of samples increases and similarity criteria between samples relaxes. The four charts at the top of the figure 3.36 summarize the percentage of space saving (*IDDG-O*/Rej.) that the *IDDG-Octree* produces with respect to the uniform grid for each of the synthetic models sampled with the domain 0–255.

Obviously, when applying the binary sampling strategy it yields a higher simplification level. In addition, it is not necessary to consider an error tolerance in applying the similarity criterion between property values, as they only have two possible values for the samples, indicating interior and exterior of the object. The results of applying this strategy to synthetic models are shown in the tables 3.13–3.16. The chart at the bottom of the figure 3.36 summarizes the percentage of space saving (*IDDG-O*/Rej.) that *IDDG-Octree* obtains with respect to the uniform grid for each of the synthetic models sampled with a binary domain.

By using the second sampling strategy, the space requirements of the different resolutions obtained are considerably lower than those obtained with the first approach. It is logical to expect this outcome because, when sampling with discrete domain, data have different property values in the volume of the grid representing the inside of the object. Obviously, because of our similarity criterion, we are going to obtain higher rates of space saving as there are more regions with similar property values and the size of these regions is larger.

The tables presented below show the time results (in seconds) spent in the major processings executed in the*IDDG-Octree* and the grid. Again, we use different sampling ratios of the synthetic models for both sampling strategies. Specifically, with respect to *IDDG-Octree* three columns are presented (from left to right) showing the time spent in the process of octree simplification (Simplif.), the process of generating the dual grid defined implicitly (Rejilla Dual) and the process of isosurface extraction (Extrac.). With respect to the grid a single column is shown with the time spent in the process of isosurface extraction (Extrac.).

Tables 3.17–3.19 show the results for processing times spent by the model (esfera). The sampling strategy used produces values in the range 0–255. Table 3.17 shows the results obtained using as similarity criterion an error tolerance $\epsilon = 0$. Similarly, tables 3.18 and 3.19 show the results for the same sampling strategy, but using error tolerances $\epsilon = 1$ and $\epsilon = 2$ respectively. As it is shown, our extraction method spent a greater time than the extraction algorithm applied on the grid for sampling ratios less than 512^3 samples. However, when the error tole-

rance is increased it can be verified that this time improves substantially. This is reasonable because increasing error tolerance yields a lower number of voxels and therefore fewer dual cells, thereby reducing the number of cells to be processed.

Similarly, the tables 3.20–3.22, 3.23–3.25 and 3.26–3.28 show the results for processing times used by the models: **tronco**, **silla** and **jarrón** for similarity criteria with error tolerances $\epsilon = 0$, $\epsilon = 1$ and $\epsilon = 2$ respectively. In the tables we can check the trend of improving the extraction time spent by the *IDDG-Octree* as the number of samples increases and similarity criteria between samples relaxes. The four charts at the top of the figure 3.37 sumarize the time, in seconds, used in the isosurface extraction (Extracción (s.)) by the *IDDG-Octree* and the uniform grid for each of the synthetic models sampled using the domain 0–255.

Obviously, a higher level of simplification is yielded in the octree when applying the binary sampling. As a result, fewer dual cells are obtained and, therefore, the extraction time spent by the *IDDG-Octree* remarkably improved with regard to the extraction time spent by the grid. The results of applying this strategy to synthetic models are shown in tables 3.29–3.32. The chart at the bottom of figure 3.37 summarizes the time, in seconds, used in the isosurface extraction (Extracción (s.)) by *IDDG-Octree* and the uniform grid for each of the synthetic models sampled using a binary domain.

As can be seen in the tables presented above, our representation spends extra time in generating on the fly the geometry of the cells needed to perform the extraction (without using the octree as a spatial index), with respect to make the extraction on a uniform grid. However, if the number of cells that it is needed to process reduces similarly to what happens to the models represented by a set of 512^3 samples (collected in the fifth row of the tables), or the error tolerance used in the similarity criterion is relaxed, such as occurs in the tables showing the values of the models **silla** and **jarrón** for $\epsilon = 2$ (tables 3.25 and 3.28), then our representation spent an extraction time lower than the grid. This fact can be tested for the sampling values 256^3 and 512^3 in the models discretized using the binary sampling strategy.

Next, the space requirements and processing times for several examples of volumetric data sets are analyzed. For this, we use four datasets obtained from the repository volren (`http://www.volren.org`). Similarly to the analysis done in the case of synthetic models presented above, the following tables show the space required for both representations, uniform grid and *IDDG-Octree*, and the time spent in the construction of the simplified octree, in the process of assigning responsibilities which implicitly defined the dual grid of cells, and in the isosurface extraction from our representation and the uniform grid.

Specifically, with respect to the storage space required, the tables 3.33–3.35 show the space, in *Kbytes*, required by the*IDDG-Octree* (*IDDG-O*), by the grid (Rej.) and the percentage of space required by our representation with respect to the uniform grid (*IDDG-O*/Rej.). Table 3.33 shows the results obtained using an error tolerance $\epsilon = 0$, and tables 3.34 and 3.35 show the results for error tolerances $\epsilon = 1$ and $\epsilon = 2$ respectively. The chart on the left side of figure 3.38

summarizes the percentage of space saving (*IDDG-O*/Rejilla) that *IDDG-Octree* yields with respect to the uniform grid for each dataset.

The tables 3.33–3.35 show that datasets *aneurism* and *bonsai* require considerably less storage space in our representation that in the uniform grid. However, the dataset *skull* requires a much larger space. The reason for this result is that the former do not contain noise on the outside of the represented objects, and therefore the similarity criterion provides very high aggregation rates in this subvolume. Despite that, in the case of *bonsai* there are many high frequency data, so the octree can not reach simplification rates as high as those obtained in the *aneurism*. In the cases of the kind that exemplifies the dataset *skull* a solution taking into account more information from the set of samples can be used with the goal of eliminating noise. If all intervals in the property value domain that are important to establish thresholds are known, then an initial filtering of the volume can be carried out.

Tables 3.36–3.38 report the results in relation to the processing time spent (in seconds). Specifically, with respect to *IDDG-Octree* three columns are presented (from left to right) showing the time spent in the process for octree simplification (Simplif.), the process of generation of the dual grid defined implicitly (R. Dual) and the process of isosurface extraction (Extrac.). With respect to the grid a single column showing the time spent in the process of isosurface extraction (Extrac.) is presented. The chart located on the right side of the figure 3.38 summarizes the time, in seconds, used in the isosurface extraction (Extraction (s.)) in the *IDDG-Octree* and the uniform grid for each of the datasets.

With regard to the extraction time from volumetric data sets, it should be noted a situation similar to that which occurs in the case of synthetic models. Our representation uses an extra time in the process of isosurface extraction if a high rate of simplification in the *IDDG-Octree* is not obtained, as it is the case with the dataset *aneurism*. First row of tables 3.33–3.35 shows the space saving achieved by *IDDG-Octree* for this dataset, and first row of tables 3.36–3.38 shows the extraction time employed by our method. In this case, our representation improves on uniform grid, in terms of space saving, and our extraction method improves on Marching Cubes algorithm in relation to the time spent in isosurface extraction.

Figures 3.39 and 3.40 show images generated by our extraction method applied to the *IDDG-Octree* and by the Marching Cubes algorithm applied to the uniform grid. It can be seen examples for each synthetic model using the two sampling strategies for the same number of samples (128^3). Specifically, from left to right, the first three columns show the images generated by our extraction method applied to the *IDDG-Octree*, using the similarity criterion with error tolerances $\epsilon = 0$, $\epsilon = 1$ and $\epsilon = 2$, respectively. The fourth column shows the result of applying the Marching Cubes algorithm in the uniform grid. The first row of the two dedicated to each model shows images solely for the first ($\epsilon = 0$) and last (grid) columns because it represents the binary sampling, and it makes no sense other than a tolerance for error $\epsilon = 0$ for the *IDDG-Octree*.

In figures 3.39 and 3.40 two rows for every synthetic model are shown. The

first row shows images resulting from binary sampling and the second images resulting from sampling in the range $[0, 255]$. All the images located in the first row of the two which corresponds with each synthetic model presents a "jagged" effect because their property values belong to the binary domain. This effect is eliminated when the synthetic object is sampled using different property values, as it can be seen at the images in the second row. We can see that the images generated by Marching Cubes (fourth column) and those generated by our method using the *IDDG-Octree* with error tolerances $\epsilon = 0$ and $\epsilon = 1$ (first two columns respectively) are difficult to distinguish visually. However, when an error tolerance $\epsilon = 2$ (third column) is applied, the differences that the isosurface extracted from *IDDG-Octree* present with respect to the one extracted from the uniform grid can be seen, except in the case of the model **silla** in which there are no appreciable differences.

With regard to the tested datasets, figure 3.41 shows a row for each dataset with a configuration of columns equivalent to the case of synthetic models. In this case, there are no appreciable differences between the images generated from *IDDG-Octree* and those generated from the uniform grid, even with an error tolerance $\epsilon = 2$ (third column).

3.7. Evaluating the quality of the extracted isosurface

In order to evaluate the goodness of the isosurface estimation we will compare the results obtained by making the extraction using *IDDG-Octree* with those obtained using Marching Cubes on the uniform grid using the same threshold. In both methods isosurface points are obtained using a cell discretization of space. The points obtained correspond to the crossing points of the isosurface with the cell edges whose property values (located at the extremes) fall one above and another below the threshold chosen.

However, the cells obtained for each type of partition are different, so we can not compare directly at each of the crossing points obtained, as these will be different. We have therefore used error measures that take into account the number of points that are located at a certain distance from their corresponding correct points located on the isosurface. In other words, the range of the error function is discretized using subintervals and an error value is calculated for each crossing point. Using the previous discretizacion the error value allows us to increase a counter of the number of points falling in the corresponding error interval. At the end a discrete error distribution is generated which may be used for comparing the goodness of the estimation produced in each method. Nevertheless, it should be note that the number of crossing points obtained in each method is different because it depends on the configuration of cells used in each representation.

The strategy to carry out the comparison is based on having a known function $f(x, y, z)$ which permits to calculate the property value throughout the volume, along with the functions $F_1(x, y, z)$ and $F_2(x, y, z)$ which are the interpolation functions used by the uniform grid and the *IDDG-Octree* respectively. To follow this strategy it is necessary to have a reference analytical object that allows us

to know the property value for each point simply by evaluating this point in the analytic expression related to that object. If the same number of samples are used in the voxelization of the reference object, then a comparison between the two representations can be established. In order to set the estimation error it is assumed as the correct value the one provided by the analytical expression at the point of interest, and as the estimated value the one provided by each of the extraction methods.

With regard to apply our comparison strategy to volumetric data sets, we have a drawback because the function $f(x, y, z)$ is not known. To solve this, we consider that the interpolation function $F_1(x, y, z)$ defined on the highest resolution grid acts like such reference function. The dataset where the extraction is applied both for the uniform grid and for the *IDDG-Octree* is obtained by subsampling the initial set of volumetric data.

3.7.1. Error study

The error produced by *IDDG-Octree* in estimating $f(x, y, z)$ with regard to the estimation achieved by the uniform grid is determined by the accumulation of two errors: an error associated with the property values assigned to the vertices of the dual cells and an error associated with the edges of these cells. In the first case, successive aggregation operations cause a loss of information in the property values domain due to the replacement of all the values of the aggregated voxels by the average of these values in the resulting voxel . In the second case, the edge segments where the interpolation really occurs may have an orientation different to the edges of the uniform grid, and consequently a different length. Figure 3.42 illustrates the idea in a 2-D example. The picture on the left shows the result of aggregating four voxels of the highest resolution into a larger one, with the consequent loss of the information related to the voxels' property values. The picture on the right shows the dual grid of cells in continuous red stroke and the cell of the uniform grid in red dashed lines. In addition, the points delimiting the edge segment where the interpolation is calculated are shown in black. As it can be seen, these edge segments have an orientation and length distinct to the edges of the uniform grid.

Both extraction methods use linear interpolation in the crossing edges of the cells to determine the point on the edge that belongs to the isosurface. Figure 3.43 shows an example that illustrates the elements that can be used to assess the error produced by the estimation. The figure shows two functions, $f(x)$ and $F(x)$, representing the real function and the interpolation function respectively that estimates it. The extraction method allows to find the point that solves the equation $F(x) = \nu$ in the crossing edge. This point is labeled as x_0 in the figure. Moreover, in disposing of the real function that we are estimating, we can calculate the value of the function $f(x)$ at that point evaluating it. The result is the value ν_0 in the figure. However, the point that truly fulfills the equation $f(x) = \nu$ in the interval is x_R.

In figure 3.43 the crossing points x_0 and x_R and the property values ν and ν_0 can be seen. These pairs of values, which are located in the domains E and

$\mathcal{V}$, respectively, allow to establish two different error metrics. If we relied on the difference in absolute value between ν and ν_0, we can have an indirect measure of the goodness of estimation by linear interpolation. A greater difference indicates that the function $F(x)$ gives a worse estimation of $f(x)$ at the point x_0 for the isovalue ν. As the difference is smaller, the function $F(x)$ provides a better estimation of $f(x)$ for the isovalue chosen.

In order to obtain a direct measure of the goodness of the estimation we use the distance in the space E between the point x_R that really fulfills the equation $f(x) = \nu$ and the point x_0 estimated by $F(x)$. In this way, the error produced at the points obtained by the estimation can be determined. However, this approach poses a problem. The function $f(x)$ is a function easy to evaluate at any point but difficult for determining the solution for $f(x) = \nu$ analytically, except for simple cases. The solution of the previous equation provides the point x_R. In fact, this is one of the reasons why the discretization of the space and the interpolation at the edges is used. Therefore, we must find an approximate method for calculating x_R. Next section shows the metric chosen to measure the error directly, that is, using the distance between x_0 and x_R in the space E, along with the way to approximate the value of x_R.

3.7.2. Error metric

Figure 3.44 shows the elements necessary to measure the error that occurs when calculating the isosurface point by linear interpolation with respect to calculating it directly using the function $f(x)$. The problem lies in the difficulty of solving the equation $f(x) = \nu$ which give us the point x_R needed to obtain the error $e_\nu = x_R - x_0$. To solve this problem we relied on the expression 3.14. This formula allows us to estimate the value of a function $f(x)$ at a point x in an interval, Δ_x, sufficiently close to a point x_0 whose value is known, wherever it is possible to calculate the derivative of the function, $f'(x)$, at that point.

$$f(x) \approx f(x_0) + f'(x_0)\ (x - x_0) \tag{3.7}$$

Clearing the term in the equation that interests us and by substituting values known yields the following equation that estimates the error produced when approximating x_R by x_0:

$$e_\nu = (x_R - x_0) \approx \frac{1}{f'(x_0)}(\nu - \nu_0) \tag{3.8}$$

In the 3D case, the gradient of the function, ∇f, at the point $\boldsymbol{X_0}$ is used. The direction of ∇f is the direction in which the directional derivative has the highest value and $|\nabla f|$ is the value of the directional derivative. Therefore, the metric for the error estimation has the following form:

$$E_\nu(\boldsymbol{X_0}) = (\boldsymbol{X_R} - \boldsymbol{X_0}) \approx \frac{1}{|\nabla f(\boldsymbol{X_0})|}(\nu - \nu_0) \tag{3.9}$$

where $\boldsymbol{X_0}$ represents the solution to the equation $(F(x, y, z) = \nu$ and $\nu_0 = f(x_0, y_0, z_0)$ being $\boldsymbol{X_0} = (x_0, y_0, z_0)$. Obviously we are assuming that there is only one solution at the crossing edges of the cells.

3.7.3. Procedure for applying the metric

With the metric above we obtain an estimation of the error produced in the calculation of each isosurface point at each crossing edge of a cell in both representations: uniform grid and *IDDG-Octree*. As discussed previously, due to the difference between the cells of both representations, it is not possible to obtain the same crossing points in order to make a point to point comparison. Therefore, the procedure for applying the error metric is based on calculating the error for all the isosurface points calculated in each of the methods, establishing relative measurements based on studying the groups of points located at a concrete distance from the real isosurface, i.e. groups of points which have the same distance error.

In analogy to the case of the evaluation of the representation, a test bench based on synthetic models and real volumetric data sets has been established. In the case of synthetic models the function $f(x, y, z)$ is known, whereas in the case of volume data sets we use as function $f(x, y, z)$ the trilinear interpolation function defined on the highest resolution grid. In the latter case, the uniform grid that is compared to the *IDDG-Octree*, which is also used for its construction, comes from a subsampling on the volume data set. The subsampling generates a uniform grid with half the samples of the original, taking double sampling interval, that is, one in two consecutive samples are used.

The procedure starts from a reference object in order to establish the comparison, either a synthetic model or a volume data set of the highest resolution, of which its function of property values distribution $f(x, y, z)$ and its gradient $\nabla(f(x, y, z))$ are known. In the case of synthetic models the object is discretized to a specific resolution, thereby obtaining a uniform grid. In the case of volumetric data these data are subsampled, as explained previously, to obtain a uniform grid with half the samples.

From the uniform grid a *IDDG-Octree* is constructed. Next, a threshold is chosen and the corresponding isosurface is extracted in each of the methods. For each crossing point the associated error is calculated using the error metric (See expression 3.16). The sequence of steps we carry out for each point with a crossing edge is as follows:

1. Obtaining the crossing point with the edge, $\boldsymbol{X_0}$, by linear interpolation.
2. Calculating the value of the function f in such a point, $f(x_0, y_0, z_0) = \nu_0$.
3. Calculating the gradient module of f in such a point, $|\nabla f(x_0, y_0, z_0)|$.
4. Obtaining the value of the signed distance $E_\nu(\boldsymbol{X_0}) = (\boldsymbol{X_R} - \boldsymbol{X_0})$ by means of equation 3.16.

In order to obtain measures for comparing the error produced by both isosurface estimation methods, the absolute distance from the point estimated to the correct point is considered. In addition, we use serveral intervals to discretize the error function domain with a fixed precision. The error in each crossing point produces the increase of a counter representing the number of points falling in a given interval. Ensuring that every error produced in all crossing points belongs to one of these intervals, then an error distribution is obtained in which each interval, in fact its related counter, contains the number of points that are at a distance from the isosurface, particularly at a distance between the extrema of such interval. This type of error distribution, obtained for each of the two representations, allows us to establish comparisons between both of them, resolving the problem in which the crossing points do not coincide and the number of crossing points obtained in each representation is different. To establish the comparisons the relative frequency distribution of error for each extraction process is used together with the derived curves of relative frequency distribution of error.

With the aim of having a visual representation of the error behavior a linear color transfer function has been constructed which provides a color related to each crossing point in terms of the error produced in its calculation. The distances provided by the equation 3.16 are the domain of the transfer function and its range is in the domain of the RGB color model. The transfer function is defined on the basis of two complementary color transfer functions that range in the channels R (red) and B (blue) of the color model. We have called these functions $R(x)$ and $B(x)$ respectively. Figure 3.45 shows the shape of these functions. The color transfer function provides a fixed value, K, for the channel G of the RGB color model. Formally, the color transfer function, $C(x)$ is defined as follows:

$$\begin{array}{rcl} C(x) : \mathbb{R} & \longrightarrow & \mathbb{R} \text{ x } \mathbb{R} \text{ x } \mathbb{R} \\ x & & (R(x), K, B(x)) \end{array}$$

where

$$R(x) = R_{MIN} + \frac{R_{MAX} - R_{MIN}}{MAX} \; x$$

$$B(x) = B_{MAX} + \frac{B_{MIN} - B_{MAX}}{MAX} \; x$$

being $[R_{MIN}, R_{MAX}]$ the color interval used in the R channel and $[B_{MIN}, B_{MAX}]$ the color interval used in the B channel.

The color gradient that provides the transfer function ranges from blue to red. The blue value matches the case in which the error produced is zero, i.e. the distance between the estimated crossing point and the real point on the isosurface is 0. The red value coincides with the largest permitted error in each model, as when the estimated isosurface has been moved towards the inner space delimited by the real isosurface, $+MAX$, as in the opposite case when the estimated isosurface has been moved towards outer spade, $-MAX$. Figure 3.46 shows the color gradient and the error values that bound the domain of error.

3.7.4. Results

In order to test the isosurface extraction error produced by *IDDG-Octree* and the uniform grid both synthetic and real volume data sets models have been used. The error range (distances to the correct isosurface) which has been considered is $[0, 3]$ and the accuracy in the discretization of this interval is 0,001. Therefore, the number of subintervals in which the domain of error considered is subdivided is 3000. All points whose error falls above the value 3 are accumulated apart. With regard to the color transfer function $C(x)$ the following parameters have been chosen: $R_{MIN} = 0{,}1$, $R_{MAX} = 0{,}9$, $B_{MIN} = 0{,}1$, $B_{MAX} = 0{,}9$ y $K = 0{,}1$.

Firstly, the comparison between representations is established using the synthetic models. Each synthetic model is discretized by a resolution of 128^3 samples using the sampling method that provides discrete values in the domain $[0, 255]$, presented in section 3.6. From the discretizations the uniform grid representation is constructed and, from it, the *IDDG-Octree* is constructed. Following, the isosurface is extracted using a threshold value of 60 and the value obtained for all crossing points is accumulated in its corresponding error subinterval of the error function domain (distances). This is carried out for both representations. In this way, the absolute frequency distribution of error F is generated.

Tables 3.39–3.42 show the error results obtained for each tested synthetic model. From left to right, the first column shows the error subinterval (Int.), which accumulates the number of points that are located at the indicated distance from the reference model. The interval is represented by a single numerical value, for example 0,001, indicating that the number of associated points is included in the interval $(0{,}000, 0{,}001]$. The following columns show the sets of error results obtained for the *IDDG-Octree*, using three similarity criteria with different error tolerances $\epsilon = 0$, $\epsilon = 1$ and $\epsilon = 2$ respectively, and for the uniform grid. Each set of results is shown in three columns. The first one shows the absolute frequency of error, F^i, the second indicates the relative frequency of error, f^i, and the third shows the accumulated relative frequency of error, $f^i(x)$. Depending on the set of associated results, the index i states:

- $i = o, j$. The measure F^i or f^i or $f^i(x)$ refers to the *IDDG-Octree* (o) which has been constructed using an error tolerance $\epsilon = j$ for $j = 0, 1, 2$.
- $i = r$. The measure F^i or f^i or $f^i(x)$ refers to the uniform grid (r).

Figures 3.47–3.50 show visually the error produced by the *IDDG-Octree* and the uniform grid in three different ways. At the top, the first three columns show the images obtained by our extraction method applied in the *IDDG-Octree* with error tolerances $\epsilon = 0$, $\epsilon = 1$ and $\epsilon = 2$ respectively. The fourth column shows the image obtained by applying Marching Cubes in the uniform grid. In the middle, beneath each image a chart representing its histogram of relative frequencies is displayed. In the histogram's X axis we indicate the numerical values corresponding to the first and last error subinterval in which there is at least a point included. Finally, at the bottom the figure shows a graph containing the four frequency distribution curves obtained in each extraction process. In this case, the numerical values listed

in the X axis indicate the subinterval in which the distribution curve(s) reach(es) the value 1. It should be noted that some graphics do not display the four curves because they have the same shape and they overlap each other. For the benefit of understanding the charts, it has become match the color of the relative frequency distribution curve of error of each image with the color of their related histogram.

For reasons of space in the paper some tables do not present the full set of results obtained. However, both the images and the statistical measures of error which are presented in figures 3.47–3.50 has been made taking into account all the data obtained for the error domain considered.

In the images at the top of the figures 3.47–3.50 it can be checked visually the difference between the distinct isosurfaces extracted from the *IDDG-Octree* and the uniform grid. The images resulting from contouring on *IDDG-Octree* using error tolerances $\epsilon = 0$ and $\epsilon = 1$ are not distinguished from the image produced from the uniform grid. This permits to infere that the process of simplification of initial samples has not reduced important information of the extracted isosurface. However, the images generated from *IDDG-Octree* using an error tolerance $\epsilon = 2$ differ from those generated from the uniform grid (except in the case of the model **silla**). This effect can be seen especially in the upper part of the model **jarrón** (See figure 3.50).

The effect in the extracted image, which is produced by the variation of the error tolerance used by the similarity criterion, can be understood more easily if we compare the charts that represent the histograms of relative frequencies of error corresponding with each image (located beneath each of them). In the case of the model **esfera** (See figure 3.47), the histogram related to the image generated for the *IDDG-Octree*, simplified with error tolerance $\epsilon = 2$, shows that there are points in the isosurface estimation falling at error subintervals distinct of 0,765. In this subinterval is where most of the points of the estimatation provided by our method are falling. It should be noted that in the case of the remaining images histograms match. This fact can be seen in the chart below which shows the relative frequency distribution curves of error. The orange curve shows the error variation of the image that corresponds to the error tolerance $\epsilon = 2$ with respect to the curves related to the remaining images. It should be noted that because the curves have the same shape, we only can distinguish the curve (in blue) corresponding with the image generated from the uniform grid. In the models **tronco** (figure 3.48) and **silla** (figure 3.49) can be checked on these comments similarly. However, in the case of the model **jarrón** (figure 3.50) the histograms seem to have all the same shape. However, it can be seen that the subinterval in which the error points for the error tolerance $\epsilon = 2$ begin to accumulate (0,639) is different from the subinterval corresponding with the other images (0,654).

In a manner similar to the synthetic models the comparison between representations is established using real volumetric data sets. In this case, each dataset has its own resolution and corresponding property values, but apart from this fact, the methodology and descriptions of tables and figures are identical to those made in the case of synthetic models.

Tables 3.43–3.46 show the error results obtained for each of the datasets tested.

From left to right, the first column shows the error subinterval (Int.), which accumulates the number of points that are located at the indicated distance from the reference model. The following columns show the error results obtained for the *IDDG-Octree*, using three similarity criteria with error tolerances $\epsilon = 0$, $\epsilon = 1$ and $\epsilon = 2$ respectively, and for the uniform grid. Each set of results is shown in three columns. The first one shows the absolute frequency of error, F^i, the second indicates the relative frequency of error, f^i, and the third shows the accumulated relative frequency of error, $f^i(x)$. The index i has the same meaning as in the case of synthetic models.

In the same way that it occurs with the synthetic models tested, figures 3.51–3.54 show visually the error produced by the *IDDG-Octree* and the grid in three different ways. At the top, the first three columns show the images obtained by our extraction method applied in the *IDDG-Octree* using error tolerances $\epsilon = 0$, $\epsilon = 1$ and $\epsilon = 2$ respectively. The fourth column shows the image obtained by applying Marching Cubes in the uniform grid. In the middle, beneath each image a chart that represents its relative frequencies histogram is displayed, and at the bottom a graph showing the four relative frequency distribution curves of error obtained in each extraction is displayed.

In the images at the top of figures 3.51–3.54 it can be checked visually the difference between the distinct isosurfaces extracted from *IDDG-Octree* and the uniform grid. In figure 3.51, which corresponds with the dataset *aneurism*, it can be noted that the areas of error (in red) may not be distinguished between images easily. The dataset *skull* (figure 3.54) produces a similar result, although the number of error areas is greater. However, in the case of the datasets *bonsai* and *lobster* it can be seen more clearly the difference in the areas of error. In the dataset *bonsai* (figure 3.52), the image generated from the uniform grid does present almost no area of error while the images generated from *IDDG-Octree* have well-remarked areas of error, although these areas are distributed nearly in an identical manner in these images. The same is true for the dataset *lobster* (figure 3.53).

The difference between the areas of error that show the images produced from the *IDDG-Octree* and the uniform grid can be validated using the charts that represent the histograms of relative frequencies of error for each image and the chart showing the error distribution curves.

3.8. Conclusions

An hybrid representation from the voxels and cells approaches, the *IDDG-Octree*, has been presented which benefits from the advantages of both. On the one hand, it saves storage space thanks to the voxels octree and, secondly, through the implicit representation of cells that are dual to these voxels it permits to make contouring with very similar results to those obtained by uniform representations, without paying the cost of maintaining an additional structure.

The concept of minimal voxel has been formalized for the *IDDGO-Octree*, and from it, a method that guarantees the automatic generation of the topology as-

sociated with the dual grid of cells has been developed. This dual grid is stored implicitly in our representation. In addition, in order to ensure that the cells partition fully covers the volume represented by the octree, a method for wrapping with dual cells the vertices of voxels that fall on the bounding box of the volume has been described.

A contouring method that follows the strategy of marching across the dual cells has been presented. This method takes into account the difference in size of the voxels that are connected by these cells. For this, the method restricts the edge segment where interpolation is performed to the edge zone of the voxels. Proceeding this way, it obtain an isosurface very close to that which is obtained from uniform representations applied on the same set of input data.

Results have been presented with respect to the storage space required and the execution time spent by the *IDDG-Octree*. In addition, an error metric and a procedure to apply it has been presented which allows us to establish comparisons between the quality of the isosurfaces extracted from *IDDG-Octree* and the uniform grid.

CHAPTER 4

Applying *IDDG-Octree* to Virtual Sculpture

Progressive improvements in the computer technologies, especially those relating to specialized graphics hardware, allow us to interactively manipulate virtual objects in a 3D virtual space. The construction of complex objects in an interactive fashion is a complex problem, both from the viewpoint of the domain of possible objects that the representation must support and from the point of view of the ease of setting up by the user. Virtual sculpture offers an approach for creating objects interactively which abstracts the user from the complexity of the representations, allowing him/her to focus on the interaction with the object through modeling tools that emulate the sculptor's real ones.

With the aim to test the response of our representation with regard to interactive manipulation, it has been developed a prototype of application that allows virtual sculpture. The interaction with the virtual object is carried out using a haptic device. The first section explores the requirements that should satisfy both representations of objects and modeling tools that allow the operation on the objects. The following section explains the process of *IDDG-Octree* updating which occurs as a result of applying a tool. To explain the method of updating proposed it is used the implementation of a tool that allows to remove material of an object represented by a *IDDG-Octree*. Finally, results and conclusions are presented.

4.1. Introduction

This section shows some studies that have addressed the virtual sculpture metaphor. Then, there are the design requirements that must be considered in a representation that allows the virtual sculpture: modification of the model with interactive response times; multiresolution representation to save space when the representation is updated through the use of sculpture tools, which may have

different shapes and sizes; and hiding the complexity of the representation to the application user. At the end of the section it is proposed a definition for a sculpture tool based on a volume that determines its shape and a field that determines how it affects to the voxels included in the volume of the tool when it is applied. Additionally, the tools can enable to change its orientation in addition to its position.

4.2. Virtual sculpture using *IDDG-Octree*

This section shows the method that has been developed to apply *IDDG-Octree* to virtual sculpture, allowing its updating in interactive time. The application of modeling operations causes the aggregation or subdivision of voxels in *IDDG-Octree*, and therefore the necessary modification of the topology of the part of the dual grid of cells affected by the changes, along with the generation of the new portion of associated isosuperficie. Our representation allows that the determination of the operation scope, modification of the corresponding portion of the dual grid , if any, and the extraction of the associated isosurface be done in interactive time. In this way, our representation remains hidden for the user ((s)he perceives the isosurface as the unique virtual model on which to work) and allows the interactive response, both fundamental requirements of the technique of virtual sculpture.

4.2.1. Modeling operations

Subsection shows our definition of a tool for virtual sculpture and the choice of an sphere defined implicitly for its implementation in the prototype. The sphere may vary in size and the field function follows a uniform distribution of material, considering the center of the sphere as the point of greatest density, decreasing the value of density as we approach to its border. The new property value that provides the field function for a voxel included in the scope of the tool takes into account the relative position of this voxel from the center of the field, as well as its current property value. This discrete allocation of property values to voxels causes aliasing effects in the object being modeled. However, our local-updating method is independent of the field function used, so it is possible to use other functions based on *kernels* to avoid this effect.

To identify on which part of the isosuperficie is a tool being applied and when the tool must not cross the isosuperficie (which is the only representation of the object from the user's point of view) it has been used an approximate method of detecting inclusion in voxels. The isosurface is calculated from the cells that are implicitly defined in the representation. However, these cells have no adjacency information associated. On the other hand, the octree allows a rapid testing of the voxels that are included in the tool. Our solution is to detect voxels included in the tool and check if they have associated cells that are crossed by the isosurface. When detected, the tool can not move towards the interior of the object until its succesive application causes the lack of isosurface in those cells. Although the solution is approximate and it is not detected directly the collision of the tool

with the geometry associated to the isosurface, the effect is hardly distinguishable by the user, yet allows dramatically increase the efficiency of the calculation (See figure 4.1).

Because we use a haptic device with force feedback to carry out the interaction on our model, it is necessary to determine when it starts to provoke the response in force. This response is carried out as soon as our approximate method detects a collision. The response in force uses functions provided by the library H3DAPI [H3d] who simply follow the Hook law.

4.2.2. Operation for removing material

Subsection describes our method for updating *IDDG-Octree* using as an example a tool for removing material with a spherical shape. The application of the tool causes, apart from the change in property values, a series of aggregations and subdivisions in the voxels that make up our representation.

4.2.3. Updating the geometry for visualization

Subsection describes the OpenGL extension named Vertex Buffer Objects (VBO) and shows how this mechanism has been used for excluding the invalid geometry, which occurs as a result of the change in property values associated to the cells' vertices and/or as a consequence of the topological changes of voxels, and for displaying the new geometry that result from these changes. Changes occur in applying the sculpture tools on *IDDG-Octree*.

During the initial process of extracting the entire isosuperficie, the VBO stores sequentially the geometry extracted for each active cell. The final state of the VBO after this extraction process is depicted in the image above of the figure 4.3. This approach follows the classic technique for allocating free space from a pool. However, whenever a tool is applied to our representation are produced changes in the geometry of the cells as a result of changes in property values of its vertices, or the cells are no longer valid because one(s) of the voxels that they connect disappear(s). Therefore, its geometry associated ceases to be valid and the space taken up in the VBO is considered free space from that moment. In the center image of figure 4.3 can be observed how, after carrying out the subdivision and/or aggregation of voxels, the hollow structure presents free space where previously it was stored geometry of the cells that have finished be valid.

4.3. Results and conclusions

To verify that the method of updating the *IDDG-Octree* allows an interactive response when the representation is updated, in addition to generating only the local geometry within the extent of the tool, a prototype to make sculpture virtual has been developed. The application allows interaction through a haptic device that provides six degrees of freedom. Due to the use of this device for interacting on the model, it is necessary to calculate both the collision of the tool with the model and a response force to be able to check on progress and feedback the

device. The calculation of the collision has already been shown and, in terms of the response in force a library (openhaptics) is used that solves the problem with a very simple model, but enough to our expectations, based on the Hooke law.

The prototype implements a spherical tool for adding material and another for removing material represented by an implicit function. No other tools have been implemented because our goal has been to test the feasibility of using *IDDG-Octree* for the interactive operation. However, like all algorithms implemented are based on obtaining the extent of the tool based on the implicit function, the application is easily extended to any tool that can be defined in this way. The prototype has been tested on a standard PC with dual core and 2 GB of RAM. Concerning to the overall structure of implementation is worth noting that one processing thread is responsible solely for calculating collisions and response in force and its execution is carried out in solely in one processor, while another thread, responsible for updating the structure and generating the new geometry, is assigned to another processor. The prototype has been tested with models of up to 256^3 samples and results of operations allowed to work in an interactive fashion.

Figure 4.4 shows the result of applying successive operations of adding material on a cube. Figure 4.5 shows the result of applying successive operations of removing material on a cube. Figure 4.6 shows in picture above an intermediate state for a virtual sculpture session whose final result is shown in the image below.

We can conclude by stating that has been presented a method for updating the *IDDG-Octree* that allows the application of this representation to virtual sculpture. Because the octree permits to determine the voxels that fall under the extent of a tool, updating the structure is local, and it is only necessary to process this information. However, as is shown in the chapter 3, *IDDG-Octree* has the disadvantage of not having explicitly stored the dual grid of cells, so the cost of updating the structure remains being the search for neighbors. Because this is the bottleneck of our representation to obtain good extraction times, appendix A shows the method for searching neighbors that has been implemented, which aims to optimize the most of such a search in the octree.

CHAPTER 5

Conclusions and future work

This last chapter summarizes the major contributions made in this PhD. thesis along with open lines of research that will provide jobs in the future.

5.1. Main contributions

A representation based on the voxel approach, the *HRB-Octree*, has been presented. This representation helps to reduce storage space using the concept of homogeneous region, whose idea underlies the use of the spatial coherence among property values of the samples. The *HRB-Octree* resolves the problem of lack of continuity that the classic voxel approach presents by means of the establishment of a partition in the voxels space, dividing it into two areas: inner zone and edges. This new partition uses an interpolation function that is different for each type of area, achieving continuity C^0 among areas and, therefore, in the voxels partition.

An hybrid representation from the voxels and cells approaches, the *IDDG-Octree*, has been presented which benefits from the advantages of both. On the one hand, it saves storage space thanks to the voxels octree and, secondly, through the implicit representation of cells that are dual to these voxels it permits to make contouring with very similar results to those obtained by uniform representations, without paying the cost of maintaining an additional structure.

The concept of minimal voxel has been formalized for the *IDDGO-Octree*, and from it, a method that guarantees the automatic generation of the topology associated with the dual grid of cells has been developed. This dual grid is stored implicitly in our representation. In addition, in order to ensure that the cells partition fully covers the volume represented by the octree, a method for wrapping with dual cells the vertices of voxels that fall on the bounding box of the volume has been described.

A contouring method that follows the strategy of marching across the dual cells has been presented. This method takes into account the difference in size of the voxels that are connected by these cells. For this, the method restricts the

edge segment where interpolation is performed to the edge zone of the voxels. Proceeding this way, it obtain an isosurface very close to that which is obtained from uniform representations applied on the same set of input data.

A method for updating the *IDDG-Octree* has been presented which allows the application of this representation to the virtual sculpture. A prototype that supports virtual sculpture by applying different modeling tools on objects represented by *IDDG-Octree* has been developed. This prototype allows carry out of a process of virtual sculpture with interactive time responses.

5.2. Future works

Throughout the development of this work issues have arisen that have posed problems which we consider interesting for a more detailed future study. Following, some of these lines of future work are listed:

- The similarity criterion used to simplify the octree is based on the spatial coherence of the property values among the samples included in a volume region. We intend to use additional information to obtain a higher simplification rate. The gradient of property is a promising measure that we are considering.
- Our method of contouring from the *IDDG-Octree*'s dual cells permits that any contouring algorithm based on the idea of marching across cubical cells can be applied. This is because we solve the table of cases using the topological cells. Our goal is to study the various geometric configurations of dual cells and to establish a specific table of cases for these cells.
- When applying modeling tools on the *IDDG-Octree*, only operations that combine property values with the volume have been implemented, i.e. boolean operations. These operations do not retain the volume associated with the representation. We intend to use the representation to permit the implemetation of deformation tools that preserve the initial volume.