UNIVERSIDAD DE GRANADA

E.T.S. de Ingenierías Informática y de Telecomunicación

Departamento de Lenguajes y Sistemas Informáticos

"*IDDG-Octree*. Una Representación Híbrida para la Visualización y Manipulación de Volúmenes"

Tesis presentada por

Alejandro José León Salas

para obtener el título de Doctor

en Informática

Granada. 19 de Diciembre de 2008

Editor: Editorial de la Universidad de Granada
Autor: Alejandro José León Salas
D.L.: GR. 189-2009
ISBN: 978-84-691-8585-8

UNIVERSIDAD DE GRANADA

E.T.S. de Ingenierías Informática y de Telecomunicación

Departamento de Lenguajes y Sistemas Informáticos

"*IDDG-Octree*. Una Representación Híbrida para la Visualización y Manipulación de Volúmenes"

Memoria presentada por

Alejandro José León Salas

para la obtención del grado de Doctor
en Informática

Dr. Juan Carlos Torres Cantero
Dr. Francisco Velasco Anguita
Directores de tesis

Granada. Diciembre de 2008

A mis padres. Sin su esfuerzo no estaría donde estoy.

A Lourdes, mi mujer. Sin tí no lo hubiera conseguido.

Agradecimientos

En primer lugar, quiero agradecer su paciencia y comprensión a la persona que ha tenido que sufrir una merma sustancial en el ya poco tiempo que podemos dedicar a estar juntos. Lourdes, ¡va por los dos y la peque!

Debido a mi, quizás, demasiado larga trayectoria como estudiante de doctorado, quiero agradecer la paciencia de Juan Carlos, que aparte de su ayuda como tutor, me ha animado en los momentos bajos, a lo largo de todo el tiempo que ha transcurrido desde que nos conocemos. Por supuesto quiero incluir a Paco, que se embarcó a mitad de este viaje y sin cuyos agudos comentarios no habría llegado a buen puerto. Y no olvido a Paco Soler, que casi llegando a puerto se incorporó en este trajín.

No puedo olvidar a mis compañeros "despacharios", Jesús, Dani, Carlos, Javi y Juan, por su amistad y ánimo continuados. En especial a Rosana, que ha participado plenamente de mis vaivenes de humor, ya que me ha tenido que sufrir, y sigue haciéndolo, como compañero de despacho.

Quiero mostrar mi agradecimiento a todo el Departamento de Lenguajes y Sistemas Informáticos. No por ser un grupo numeroso me gustaría dar la falsa idea de agradecimiento general del tipo "por quedar bien". Tengo bien grabados en mi memoria los momentos pasados con cada uno de sus componentes. Creo que todos ellos me han visto crecer y con todos ellos he compartido algo en algún momento. Especialmente quiero recordar a mis compañeros de asignaturas Domingo, José María, Patricia, Marisa, Emilia, Marian, José Antonio, José Luís, José Miguel y Pedro, por su comprensión y apoyo durante todo este tiempo; y, como no, a todos los compañeros del Grupo de Investigación en Informática Gráfica, Pedro, Javi, Jorge,... y todos los demás (somos muchos). Cada uno de ellos, algunos más que otros, conoce de cerca las visicitudes que han sucedido en el transcurso de la realización de este trabajo.

No debo, ni quiero, olvidar a Joaquín Fernández Valdivia y a Javier Pérez González, que aún estándo ubicados en otros departamentos me han regalado generosamente parte de su tiempo con dudas que me han surgido durante el desarrollo de este trabajo. Además me gustaría agradecer al Dr. Dirk Bartz su interés y consejos aportados durante la realización de mi estancia en la Universidad de Tübingen durante el año 2005.

Este trabajo ha sido subvencionado con los proyectos de investigación de la Comisión

Interministerial de Ciencia y Tecnología codificados como TIN2004- 06326-C03-02 y TIN2007-67474-C03-02 y con el proyecto de excelencia de la Junta de Andalucía codificado como (PE-TIC-401).

Resumen

La presente memoria se enmarca en el contexto del desarrollo de modelos de volumen. Si por algo se pueden distinguir este tipo de modelos es por tratar de caracterizar a los objetos reales mediante la representación de unos determinados valores pertenecientes a unos determinados dominios de propiedad. Por tanto, en este tipo de modelos interesa representar tanto la extensión espacial que ocupa el objeto como la variación de valores de los dominios de propiedad de interés.

El objetivo del desarrollo de tales modelos es proporcionar un conjunto de herramientas informáticas que permitan estudiar las propiedades de los objetos representados, visualizar la información contenida en estos de forma que se le facilite al observador el comprender dicha información y simular la manipulación del objeto real sobre el modelo computacional. La figura 1 muestra algunos ejemplos de imagenes que pueden obtenerse a partir de este tipo de modelos.

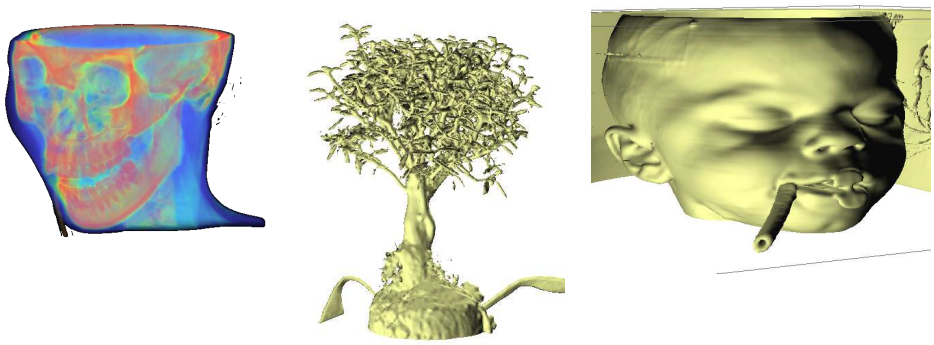

Figura 1: Algunas imágenes obtenidas a partir de modelos de volúmen.

De forma general, se puede definir un volumen como un subconjunto del espacio tridimensional cuyos puntos representan valores de un dominio de propiedad. Un volumen así definido puede ser representado mediante una función que haga corresponder los puntos de un subconjunto V del espacio euclídeo tridimensional en un dominio de propiedad Γ:

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

Sin embargo, cuando se trabaja con volumenes que han sido obtenidos de forma discreta, no podemos conocer el valor de dicha función f en todo el volumen de interés. Tan sólo se dispone del conjunto de muestras espaciales con los valores de f asociados. Lo que se suele utilizar en este tipo de modelos de volumen es una estimación F de la función f obtenida a partir de los valores de propiedad de las muestras.

Hoy en día, debido al elevado número de muestras que son capaces de obtener los aparatos de medida de información volumétrica y al incremento de las prestaciones de los ordenadores, la cantidad de información que es necesario representar en los modelos es cada vez mayor, con lo que es necesario contemplar modelos que permitan reducir la cantidad de información "en bruto" proporcionada por dichos aparatos.

En esta memoria se propone un esquema de representación que permite reducir la cantidad de información necesaria para representar los datos volumétricos. Así mismo, se propone una representación volumétrica que permite la visualización y manipulación de este tipo de datos. El capítulo 1 muestra el contexto de la visualización de volúmenes, en el cual se enmarca nuestro trabajo. El capítulo 2 presenta una representación que aúna las ventajas de los dos principales enfoques utilizados en las representaciones de volúmenes discretas. En el capítulo 3 se describe nuestra propuesta de representación de volumen y se evaluan sus características con respecto al ahorro de espacio de almacenamiento, tiempos de procesamiento y calidad de las isosuperficies que se extraen a partir de ella. En el capítulo 4 se muestra una aplicación de nuestra reprentación a la escultura virtual en la que se puede constatar su capacidad para permitir tiempos de respuesta interactivos durante la aplicación de operaciones de modelado. Por último, el capítulo 5 presenta las conclusiones de nuestra tesis y las líneas futuras de investigación que ha permitido abrir.

Parte del trabajo expuesto en esta memoria de tesis doctoral ha sido publicado previamente en el Ibero-American Symposium in Computer Graphics (SIACG 2006) [LTV06], en el Congreso Español de Informática Gráfica (CEIG'07) [LTV07a], en el libro que publica los resultados de investigación correspondientes al proyecto de investigación de la Comisión Interministerial de Ciencia y Tecnología codificado como TIN2004-06326-C03-02 [LTV07b, LTVS07], en el número 4 (Agosto del 2008) de la revista Computer & Graphics [LTV08] y en el Congreso Español de Informática Gráfica (CEIG'08) [LVS08].

Alejandro J. León Salas

Índice

Notación matemática

$\boldsymbol{P}, \boldsymbol{Q}, \boldsymbol{R}, \ldots$	Puntos del espacio euclídeo.
$\vec{a}, \vec{b}, \overrightarrow{PQ}, \overrightarrow{QR}, \ldots$	Vectores.
$A, B, \mathcal{A}, \mathcal{B}, \ldots$	Conjuntos.
$\{x, y, \ldots : A;\ z : B; \ldots \mid$ Predicado· ·Forma_De_Los_Elementos$\}$	Definición de conjuntos.

Índice de figuras

Índice de tablas

CAPÍTULO 1

Introducción al modelado de volúmenes

El modelado de volúmenes es una línea de investigación dentro de la informática gráfica en la que se ha estado trabajando con profusión en los últimos años. Los modelos de volumen pretenden representar la falta de homogeneidad de los objetos del mundo real, formados por multitud de tipos de materiales, presentando estos una gran diversidad de propiedades. Para conseguir este objetivo se han utilizado diferentes representaciones basadas en distintas herramientas de modelado matemático junto con las estructuras de datos que permiten su representación computacional. Junto a éstas, se han desarrollado multitud de algoritmos que permiten la realización de operaciones sobre tales representaciones.

En el presente capítulo se presenta el contexto en el que se asienta el modelado de volumen. Sin tener la pretensión de abarcar la gran cantidad de trabajos realizados sobre el tema, se pretende establecer una visión general que permita exponer la terminología y plantear los problemas que se presentan en este ámbito. En la sección de *Modelado de Sólidos* se describen los métodos de representación de objetos homogéneos. Los modelos de este tipo permiten establecer claramente las propiedades que se requieren para el modelado de objetos reales y que permiten, en algunos casos y con alguna extensión adicional, representar la falta de homogeneidad que persiguen los modelos de volumen. La sección que trata sobre los *Esquemas de representación* establece el marco formal sobre el que se trabaja para realizar el modelado de objetos y presenta algunas propiedades deseables para este tipo de esquemas. En la sección 1.4 se describen las distintas estrategias que se utilizan para representar sólidos y que pueden aplicarse mediante pequeñas extensiones al modelado de volumen.

En la sección 1.5, *Modelado de Volúmenes* se presentan los principales elementos que se requieren a la hora de establecer un determinado modelo de volumen. Se presentan dos grandes estrategias de modelado: las representaciones continuas y las representaciones discretas, haciendo especial énfasis en las últimas, ya que en la presente memoria de tesis doctoral se ha desarrollado una representación que pertenece a esta segunda clase. En la siguiente sección, se ha distinguido especialmente la operación de visualización sobre modelos de volumen, ya que es la que ha recibido especial atención en

la bibliografía. En esta sección se presentan principalmente las dos grandes estrategias utilizadas para abordar la visualización y, de forma similar a la anterior sección, se hace especial énfasis en la visualización mediante extracción de isosuperficies debido a que ésta ha sido la estrategia utilizada a la hora de visualizar nuestra representación.

1.1. Modelos computacionales de objetos reales

De forma general, podemos pensar que un **sistema** es cualquier *conjunto de elementos* con identidad propia, los cuales presentan una serie de *relaciones* entre ellos mismos y con el medio que les rodea. El conocimiento que tenemos o adquirimos acerca de un sistema (sus componentes y relaciones) varía en función de nuestra experiencia con él, por lo que las fronteras que se establecen entre sistemas deberán ser dinámicas.

En una primera aproximación podríamos definir un modelo como la *representación de un sistema en un lenguaje determinado*. Lo más importante a la hora de construir el modelo de un sistema es establecer el *significado* que tendrán para nosotros los componentes del lenguaje[1] que usemos para describir el sistema, así como los *aspectos relevantes* que nos interesa destacar de dicho sistema. Por consiguiente, a la hora de construir un modelo será primordial la *observación* que realicemos del sistema. La observación determina las características del sistema que el constructor del modelo estima oportuno que se vean reflejadas en éste, teniendo en cuenta el uso que se va a hacer del modelo.

Un *modelo computacional* debe permitir utilizar los ordenadores para representar un objeto externo a estos. Obviamente la elaboración de modelos computacionales viene determinada por las características inherentes a los ordenadores, ya que son el sistema en el que "viven" las representaciones. Quizá la más destacable de estas características es el componente discreto que presentan. En otras palabras, cualquier representación computacional estará compuesta por elementos discretos.

Para representar fenómenos y objetos del mundo real (o uno imaginario) en un ordenador es necesario establecer un modelo computacional del sistema a representar. Estas representaciones mantienen información de los aspectos relevantes del sistema en función de las características que el creador del modelo considera oportuno que sean descritas en éste y, junto a esta información descriptiva, mantienen información procedimiental, es decir, las operaciones que se desea puedan ser llevadas a cabo sobre tal modelo. En palabras más "informáticas", las representaciones computacionales están formadas por *estructuras de datos* y *algoritmos*.

En el ámbito de la informática gráfica, se habla de *modelado de sólidos* cuando al observador no le preocupa representar la falta de homogeneidad del objeto real, sino más bien la forma de dicho objeto. De alguna forma, las representaciones usadas para este tipo de modelos describen el volumen interior de forma implícita, considerándolo como un todo homogéneo. En estas representaciones se busca como fin último describir la apariencia final de la forma exterior del objeto. Como contrapunto, se habla de *modelado de volumen* cuando las representaciones muestran la falta de homogeneidad en el interior del objeto, es decir, permiten representar variaciones de alguna(s) propiedad(es) a través del interior de éste. La figura 1.1 muestra imagenes generadas a partir de cada clase de representación. Las imagenes situadas en la parte superior se obtienen a partir de

[1] Entendemos *lenguaje* en un sentido amplio. Cualquier conjunto de símbolos y reglas que permita la descripción y a los que podamos aplicar una interpretación del significado del sistema que pretendemos reflejar con el modelo.

un modelo de sólidos. La imagen de la izquierda utiliza una malla poligonal y la de la derecha superpone texturas a dicha malla. En contraposición, las imagenes de la parte inferior muestran variaciones en la concentración de ozono (izquierda) y variaciones de presión sobre el Océano Pacífico (derecha).

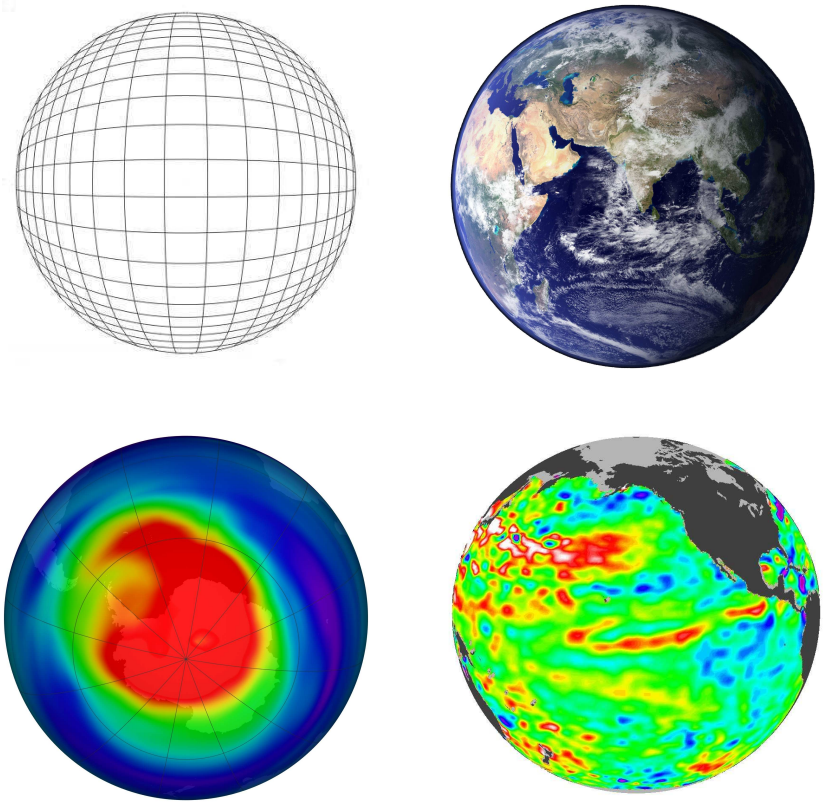

Figura 1.1: Imágenes para ilustrar la diferencia entre un modelo de sólidos (imágenes superiores) y un modelo de volumen (imagenes inferiores). Las imágenes en color son cortesía de la NASA (`http://visibleearth.nasa.gov`).

1.2. Modelado de sólidos

A lo largo de la historia se ha descrito la forma de los objetos reales usando una rama de las matemáticas: *la geometría*. Los modelos geométricos han permitido representar la forma de los sólidos y las operaciones que pemiten describirlos y obtener información sobre determinadas propiedades de estos. Podemos considerar el *Espacio Euclideo tridimensional*[2] , E^3, como una idealización adecuada del espacio real en el que se encuentran los objetos que se pretenden modelar. Las entidades geométricas abstractas que permiten modelar objetos son *subconjuntos de* E^3. Requicha [Req77] describió las

[2] En el texto utilizaremos 3-D para referirnos a tridimensional, y los términos análogos 1-D y 2-D para los espacios euclideos unidimensional y bidimensional respectivamente.

características de los objetos a modelar[3] que tiene que reflejar un modelo geométrico para recoger la noción de "sólido abstracto":

1. **Rigidez**. Un sólido abstracto debe tener una forma o configuración invariantes que sean independientes de la posición y orientación del sólido.

2. **Tridimensionalidad homogénea**. Un sólido debe tener un interior y la frontera de un sólido no puede presentar partes aisladas, en el sentido de elementos aislados con una dimensionalidad que no tiene su contraparte en el mundo real.

3. **Finitud**. Un sólido debe ocupar una porción finita del espacio.

4. **Clausura frente a transformaciones rígidas y ciertas operaciones Booleanas**. Las transformaciones rígidas (traslaciones y/o rotaciones) y las operaciones que añaden o sustraen material deben producir sólidos cuando se aplican a otros sólidos.

5. **Describilidad finita**. Los modelos de sólidos en E^3 deben presentar una componente finita que asegure que van a poder ser representados en un ordenador. No podemos representar de forma directa conjuntos de puntos continuos en un ordenador, por lo que necesitamos métodos indirectos que permitan tal representación.

6. **Determinismo de frontera**. La frontera de un sólido debe determinar sin ambigüedad qué está "dentro", y por tanto configura el sólido.

En este trabajo Requicha describió las implicaciones matemáticas de las propiedades enumeradas, argumentando que los modelos adecuados para el concepto de sólido abstracto son las clases congruentes de subconjuntos de E^3 cerrados, acotados, regulares y semianalíticos. Definió una clase congruente como una colección de conjuntos cada uno de los cuales puede ser obtenido a partir de otro mediante secuencias de traslaciones y rotaciones. Denominó a tales conjuntos *r-sets* (del inglés *regular sets*).

La definición de *r-set* no establece restricciones en cuanto a que el conjunto sea conexo y no pueda presentar túneles (agujeros que lo atraviesen completamente). Los *r-sets* no son algebráicamente cerrados bajo el conjunto de operaciones Booleanas tradicional. Para cumplir la propiedad (4), Requicha [RV77, RT77] estableció el conjunto de operaciones Booleanas regularizadas, el cual proporciona un resultado correcto realizando la clausura del interior de conjunto resultado de la operación booleana clásica.

Los *r-sets* Tienen que ser expresados por otros objetos matemáticos de cara a poder ser descritos en una representación computacional. Se pueden tomar dos estrategias: expresar el *r-set* incluyendo su interior y frontera o, utilizar solamente una expresión de su frontera. Tanto en el primer caso, como en el segundo, podemos utilizar una representación explícita o implícita del *r-set*. Veamos un ejemplo para aclarar la distintas combinaciones de posibilidades.

Se puede utilizar una función implícita para determinar un *r-set* mediante una inecuación, o la solución a un conjunto de inecuaciones, de la forma:

$$f(x, y, z) \leq c, c \in \mathbb{R}, f : \mathbb{R}^3 \mapsto \mathbb{R}$$

[3] A partir de ahora utilizaremos el término objeto para referirnos a los objetos a modelar, ya sean objetos del mundo real u objetos de un mundo imaginario

El conjunto de puntos $P \in \mathbb{R}^3 \mid f(P) \leq c$ determina un *r-set* siempre que las expresiones matemáticas contenidas en la función f sean algebráicas. De esta forma se determina implícitamente el *r-set* y su frontera. De forma análoga, podemos expresar el *r-set* como una combinación de funciones implícitas siempre que tal combinación garantice la generación de un *r-set* válido. De esta forma, utilizamos un conjunto explícito de funciones. Este es el enfoque utilizado por ejemplo en la representación CSG [RV77, Mä88].

Si por el contrario queremos definir el *r-set* únicamente mediante su frontera, se puede utilizar una función implícita de la forma:

$$f(x, y, z) = c, c \in \mathbb{R}, f : \mathbb{R}^3 \mapsto \mathbb{R}$$

El conjunto de puntos $P \in \mathbb{R}^3 \mid f(P) \leq c$ determina un *r-set* siempre que las expresiones matemáticas contenidas en la función f sean algebráicas. De esta forma el *r-set* queda definido implícitamente por su frontera. Por otra parte, si definimos la frontera como una combinación de funciones implícitas, estamos utilizando un método que explícitamente enumera cada uno de los componentes de dicha frontera. Por ejemplo, en las comúnmente llamadas representaciones de frontera, el *r-set* se representa dividiendo la frontera en un número finito de subconjuntos acotados denominados normalmente caras o parches.

Por otra parte, es posible caracterizar las propiedades asociadas al concepto de "sólido abstracto" basándonos en propiedades de superficies. La caracterización basada en superficies ve al sólido únicamente como su frontera. La frontera se considera formada por una colección de "caras" que están unidas para formar una envoltura cerrada alrededor del objeto.

Intuitivamente, una superficie puede considerarse como un subconjunto de E^3 de dimensión dos. La dimensionalidad de una superficie significa que se pueden estudiar sus propiedades por medio de un modelo bidimensional. Se puede definir de forma abstracta una superficie en base al concepto de *variedad*. Una 2-variedad M es un espacio topológico donde cada punto tiene una vecindad topológicamente equivalente a un disco abierto de E^2. Si nos centramos en un *r-set* A, si la frontera de A, $b(A)$, y una 2-variedad M son topológicamente equivalentes podemos afirmar que A es una *realización* de M en E^3. De esta forma podríamos determinar *r-sets* utilizando 2-variedades. Desafortunadamente, no todos los *r-sets* son realizaciones de alguna 2-variedad. Para establecer condiciones suficientes para la realización de una 2-variedad, es necesario un modelo que permita expresar todas las 2-variedades de forma que nos permita razonar sobre sus propiedades. Para este propósito se usa el *modelo plano* [Mä88]. El modelo plano nos permite definir perfectamente la noción de sólido abstracto, estableciendo las condiciones necesarias que garantizan la realización de un modelo plano en E^3.

La relación entre la topología de las superficies y la topología de los conjuntos de puntos se establece debido a que se puede establecer una correspondencia entre los puntos y los conjuntos abiertos del modelo plano, y los puntos y conjuntos abiertos de los *r-sets*, mediante una transformación continua. En la práctica, se utilizan correspondencias "más operacionales" que puedan ser representadas mediante la asignación de información geométrica: coordenadas, ecuaciones de curvas y superficies, etc. a los vértices, aristas y polígonos del modelo plano.

Una vez definido el conjunto de objetos matemáticos que permiten la definición de la abstracción de sólido, vamos a presentar formalmente como se pueden establecer relaciones entre estos y el espacio de representaciones computacionales.

1.3. Esquemas de representación

Marti Mäntylä [Mä88] presenta un esquema que divide el proceso de modelado de sólidos en tres niveles bien diferenciados:

1. *Objetos físicos*. Este nivel se refiere bien a cualesquiera elementos visibles situados en nuestro mundo real tridimensional, o bien a los objetos imaginados por un diseñador.

2. *Objetos matemáticos*. Es necesario adoptar una idealización adecuada de los objetos físicos con el objetivo de establecer modelos de estos en un ordenador. Estos objetos idealizados deberían mantener características de los objetos físicos, y a la vez ser lo suficientemente simples para que se les puedan asignar representaciones computacionales. Mäntylä caracteriza estos objetos utilizando conceptos de las teorías de topología de conjuntos de puntos y topología algebraica.

3. *Representaciones*. El paso final es asignar a los objetos matemáticos una representación adecuada para poder ser utilizada en el ordenador.

Una vez definidos los *r-sets* como los elementos matemáticos adecuados para capturar las características deseables de solidez, vamos a definir qué es una representación y un esquema de representación, el cual permite establecer correspondencias entre dichos objetos matemáticos y las representaciones computacionales.

Una representación sintácticamente correcta está formada por estructuras finitas de símbolos construidas a partir de un conjunto de símbolos inicial (alfabeto) siguiendo unas reglas sintácticas. La colección de todas las representaciones sintácticamente correctas se denomina un *espacio de representación* R. La semántica de las representaciones se define en base a los objetos matemáticos. Para ello, se define un *espacio de modelado matemático* M cuyos elementos son los *r-sets* definidos previamente, y se establece una correspondencia entre los elementos de M y los elementos de R mediante un esquema de representación. La figura 1.2 ilustra el esquema de representación.

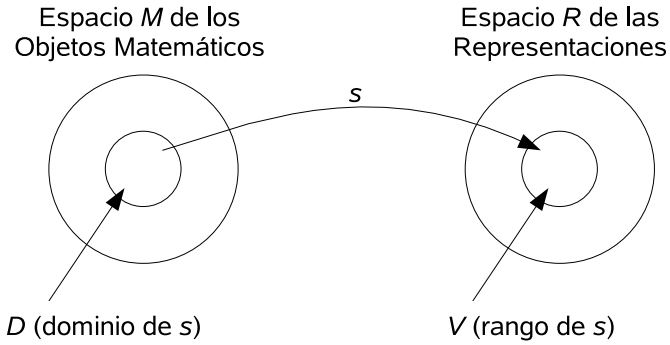

Figura 1.2: Definición de esquema de representación utilizando diagramas de Venn.

Se define formalmente un *esquema de representación* como una relación $s : M \mapsto R$. Notamos el dominio de s como D, y la imagen o rango de D mediante s como V. Cualquier representación incluida en V se dice que es *válida* ya que es sintáctica y semánticamente correcta. La definición de esquema de representación no asume que

todos los objetos matemáticos sean representables, $D \neq M$, ni tampoco que todas las representaciones sintácticamente correctas sean válidas, $V \neq R$.

Requicha [Req80] presenta una serie de propiedades formales que cumplen los esquemas de representación, junto con algunas de índole práctico que es deseable que cumplan. Hemos de destacar las siguientes. Una representación $r \in V$ es *no ambigua (precisa) o completa* si se corresponde con un único objeto matemático. Es decir, si el conjunto $s^{-1}(r)$ es un subconjunto $m \subset D$ con un único elemento. Una representación $r \in V$ es *única* si sus correspondientes objetos en D no admiten otra representación distinta de r en el esquema. Formalmente, si $s(s^{-1}(r)) = r$. Extendiendo estas definiciones se puede decir que un esquema de representación s es no ambiguo o completo si toda representación $r \in V$ es no ambigua, es decir la relación inversa s^{-1} es una función. De forma similar, un esquema de representación es único si toda representación $r \in V$ es única o, lo que es decir que s es una función. Los esquemas de representación que son a la vez completos y únicos son muy deseables ya que establecen correspondencias uno-a-uno entre el dominio de objetos matemáticos y el rango de las representaciones. Esto implica que representaciones distintas se corresponden con objetos distintos.

Otro aspecto interesante de la definición del esquema de representación es cómo determinar el conjunto de representaciones que son válidas, o desde un punto de vista práctico, cómo determinar de forma automática la validez de una determinada representación. Esto puede consiguirse utilizando dos estrategias. Por una parte, proporcionando algoritmos que comprueben la validez de las representaciones una vez que hayan sido creadas, y por otra, incluyendo restricciones de validez en las operaciones que permiten la construcción de representaciones.

Por último es de destacar una propiedad que sería razonable que cumpliesen los esquemas de representación debido al necesario alojamiento de las representaciones en un ordenador: ser *conciso*. La concisión de una representación se refiere al tamaño de esta. Las representaciones concisas son convenientes a la hora de almacenarlas en la memoria y transmitirlas a través de enlaces de comunicación. Además, son interesantes de cara a la interacción del usuario.

Para ilustrar el concepto de esquema de representación vamos a utilizar un ejemplo simple. Consideremos un esquema de representación que permita trabajar con poliedros convexos. Un poliedro convexo puede definirse formalmente como el conjunto de soluciones a un sistema lineal de inecuaciones de la forma

$$m\ x \leq \boldsymbol{b}$$

donde m es una matriz real de dimensión $sx3$ y $\boldsymbol{b}$ es un vector de reales de dimensión s. Adicionalmente la solución debe estar acotada [Wei08].

Vamos a definir los poliedros convexos mediante la determinación de su frontera. En este caso, los objetos matemáticos (poliedros) están definidos mediante polígonos planos, cada uno de ellos definido mediante el conjunto de puntos de E^3 que lo conforman (obviamente con la restricción de que sean coplanares), siguiendo éstos una ordenación adecuada que permita representar la frontera de forma consistente. Podemos definir la sintáxis de la representación utilizando la gramática que genera tuplas de longitud variable que contienen valores reales. La notación para cada representación podría ser un conjunto de n tuplas de tamaño variable conteniendo valores "reales"[4] :

[4] La representación utilizaría la representación computacional asociada a los números reales matemáticos, es decir, representación mediante mantisa y exponente con una precisión dada

$$
\begin{aligned}
t_0 &\equiv ({P^0}_0, {P^0}_1, \ldots, {P^0}_{u_0}) \\
t_1 &\equiv ({P^1}_0, {P^1}_1, \ldots, {P^1}_{u_1}) \\
&\ldots \\
t_n &\equiv ({P^n}_0, {P^n}_1, \ldots, {P^n}_{u_n})
\end{aligned}
$$

Cada elemento ${P^i}_j$ perteneciente a una tupla es a su vez una terna de valores reales $({x^i}_j, {y^i}_j, {z^i}_j)$, teniendo cada tupla $u_i + 1$ elementos. Una vez definida la notación de la representación es necesario definir su semántica (significado geométrico), especificando para ello una regla que asocie geometría con representación, es decir, que establezca el esquema de representación. En nuestro caso, cada elemento ${P^i}_j$ para $i = 0, 1, \ldots, n$ y $j = 0, 1, \ldots, u_n$ representa las coordenadas de un punto del espacio E^3, representando cada tupla t_i las coordenadas de los vértices que determinan un polígono. El conjunto total de tuplas es la representación de la frontera poligonal de una poliedro. La figura 1.3 ilustra visualmente este ejemplo de esquema de representación mediante un poliedro formado por 72 caras poligonales y su representación computacional correspondiente.

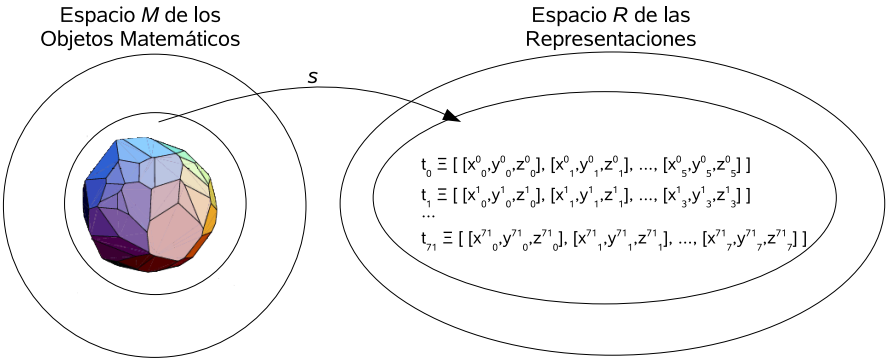

Figura 1.3: Ejemplo de esquema de representación que permite representar poliedros convexos definidos mediante caras poligonales usando para ello tuplas de valores reales que representan las coordenadas de los vértices de cada cara.

1.4. Representaciones de sólidos abstractos adecuadas para modelar el interior

En las secciones anteriores se ha mostrado que la noción de sólido abstracto puede ser capturada mediante conjuntos de puntos de E^3, expresados bien utilizando *r-sets*, o bien mediante modelos planos realizables. Los esquemas de representación establecen formalmente el marco para permitir asignar representaciones computacionales finitas a los modelos matemáticos. Mäntylä [Mä88] establece una clasificación de las representaciones en tres grandes clases: *Modelos por Descomposición*, *Modelos Constructivos* y *Modelos de Frontera*. Solamente vamos a describir de forma general algunas representaciones incluidas en las dos primeras clases. Esto se debe a que dichas estrategias de representación permiten, debido a su estructura, reflejar la falta de homogeneidad en el interior de los conjuntos de puntos, que es el elemento fundamental del modelado de volúmenes. Al final se presentará brevemente el enfoque del modelado implícito, el cual permite establecer representaciones tanto de sólidos como de volúmenes.

1.4.1. Representaciones por descomposición espacial

La noción de sólido abstracto recogida en los modelos matemáticos descritos anteriormente hace que estos se vean como conjuntos de puntos continuos. Al ser imposible almacenar en el ordenador todo el conjunto de puntos que constituyen el sólido, se opta por discretizar el espacio ocupado por éste. De esta forma, el sólido se representa *descomponiendo* el espacio que ocupa en un conjunto de celdas y manteniendo información relativa a cada una de ellas. La clase de celdas utilizadas y la forma en la que se combinan determina las distintas estrategias llevadas a cabo sobre esta idea base. No obstante, las distintas realizaciones de la idea tienen un requisito común: Las celdas deben ser disjuntas o tener solamente caras y/o aristas y/o vértices comunes. Es decir, deben formar un recubrimiento espacial del sólido sin solapamientos entre celdas, o lo que es lo mismo, una *partición espacial*. A continuación se definen formalmente ambos conceptos.

Definición 1 (Recubrimiento) *Sea I un conjunto de índices de la forma $I = \{0, 1, \ldots, n\}$. Se dice que una familia de subconjuntos $\{A_i \subseteq A \mid i \in I\}$ de un conjunto A es un recubrimiento de A si se cumplen las siguientes condiciones:*

1. $A_i \neq \varnothing,\ \forall i \in I$
2. $\bigcup_{i \in I} A_i = A$

Definición 2 (Partición) *Sea I un conjunto de índices de la forma $I = \{0, 1, \ldots, n\}$. Se dice que una familia de subconjuntos $\{A_i \subseteq A \mid i \in I\}$ de un conjunto A es una partición sobre A si la familia es un recubrimiento y además se cumple que:*

1. $A_i \cap A_j = \varnothing,\ \forall i, j \in I, i \neq j$

En conclusión, las distintas estrategias tienen en común que el espacio que representa al sólido está dividido en celdas que no se solapan y, por tanto, cualquier punto perteteciente a éste puede ser identificado como incluido en una determinada celda (o celdas caso de caer en la frontera entre celdas vecinas).

1.4.1.1. Instanciación de primitivas

Se basa en utilizar *familias de objetos*. Cada uno de los miembros de una familia de objetos se distingue del resto mediante una serie de parámetros. Cada familia de objetos se denomina *primitiva genérica* y cada uno de los objetos individuales que constituyen la familia se conoce con el nombre de *instancia de primitiva*. La característica principal que distingue los esquemas de instanciación de primitivas puros de otras representaciones es la falta de operaciones para combinar instancias de primitivas y obtener como resultado nuevos objetos con una estructura más compleja.

Los esquemas basados en instanciación de primitivas puros son completos, no necesariamente únicos, fáciles de validar y concisos. Mäntylä [Mä88] muestra un ejemplo que ilustra las propiedades de completitud y posible falta de unicidad de estos esquemas de representación. Este tipo de representaciones es un caso muy particular de descomposición espacial ya que el número de celdas que constituyen una determinada representación es siempre igual a uno.

1.4.1.2. Descomposición en celdas

Esta representación parte de un conjunto de celdas que permiten establecer una partición del espacio ocupado por el *r-set*. Las celdas pueden ser cualquier objeto topológicamente equivalente a una esfera, pudiendo incluir caras curvas.

La representación es completa ya que una descomposición de celdas válida determina completamente un *r-set*, pero no es una representación única. Además, presenta el inconveniente de que es difícil asegurar la validez, ya que la representación está constituida por un conjunto desordenado de celdas. Por consiguiente, para garantizar la validez es necesario llevar a cabo un test de intersección para cada par de celdas. Con respecto a la concisión de la representación, aún dependiendo de las características de las celdas y el tipo de *r-set* a representar, es un esquema bastante conciso dentro de la clase de representaciones por descomposición espacial. Sin embargo, es necesario almacenar información de la posición y estructura de cada celda. Por ejemplo, establecemos un esquema de representación en el que las celdas son poliedros curvos de seis caras. Podemos definir un poliedro curvo de seis caras utilizando 20 puntos, 8 de ellos formando los vértices y 12 sobre las líneas entre las esquinas. Cada línea pasa por tres puntos que definen una curva cuadrática, y cada cara del poliedro está formada por un parche de superficie bicuadrática determinado por 8 puntos. Por tanto, para cada celda habría que almacenar referencias a los 20 puntos que la forman junto con la información topológica para formar la estructura de la celda.

1.4.1.3. Enumeración de la ocupación espacial o enumeración exhaustiva

En esta estrategia la celda base es normalmente un cubo de tamaño y orientación uniformes que forman una *subdivisión regular* del espacio. Debido a la propiedad de regularidad, para representar cada celda es suficiente con almacenar las coordenadas de una de sus esquinas. En el caso de que a priori limitemos el alcance de nuestra representación a un subconjunto de E^3, la colección de celdas puede estructurarse en forma de rejilla regular, permitiendo almacenar las coordenadas que identifican cada celda de forma implícita, manteniendo solamente una coordenada para la rejilla en su conjunto. Cada cubo de la rejilla puede tener asociados dos valores indicando pertenencia al *r-set* (negro) o espacio vacio (blanco). En el caso de que interese representar la falta de homogeneidad del *r-set*, como es el caso de las representaciones de volumen, se puede usar un rango de valores más amplio.

Estos esquemas son completos, únicos (excepto por la falta de unicidad posicional) y fáciles de validar. Sin embargo, no son concisos en general, ya que esta propiedad depende de la clase de objetos que se vaya a representar. No es lo mismo que el dominio de objetos matemáticos esté formado por objetos poliédricos con caras ortogonales que objetos delimitados por superficies curvas. Sin embargo, estos métodos son adecuados para construir una representación no ambigua a partir de un conjunto de puntos obtenidos mediante medidas de objetos reales o simulaciones computacionales de fenómenos. Estos puntos pueden caer en la frontera o en el interior del objeto y si la coordenada de cada punto es tomada como la información de la representación de los *r-sets* asociados, se obtiene por lo general una representación ambigua. El enfoque de enumeración espacial permite transformar esta representación de coordenadas de puntos ambigua en una representación mediante celdas no ambigua. Además, si se asocia la medida de una determinada propiedad en el caso de tratar con puntos interiores al objeto, la representación por enumeración permite reflejar la falta de homogeneidad.

La figura 1.4 muestra algunas imagenes que ejemplifican este tipo de representación. Las imagenes situadas en la parte superior muestran las representaciones de un toroide y una esfera. Las imagenes inferiores muestran un objeto inicial y dos representaciones por enumeración espacial a distintas resoluciones.

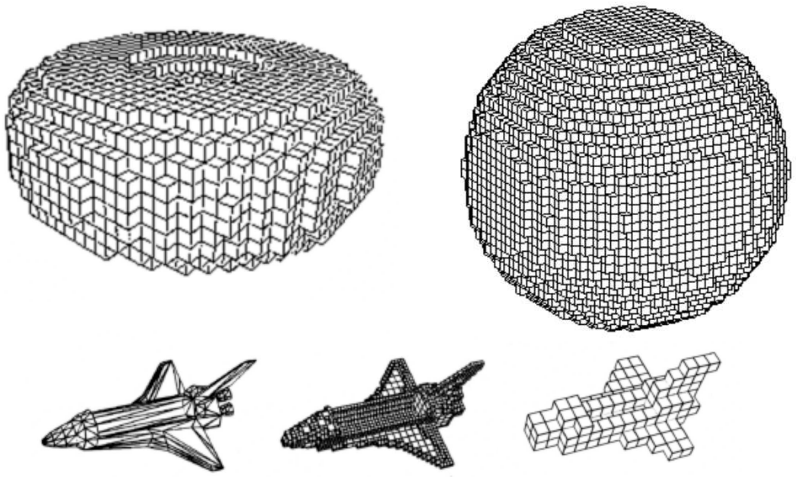

Figura 1.4: Varias imagenes que ilustran la representación basada en la enumeración de la ocupación espacial.

1.4.1.4. Esquemas jerárquicos de particionamiento espacial

El punto débil de los esquemas mediante enumeración espacial es la gran cantidad de memoria que utilizan para la representación. Esto se debe a que conforme se quiere obtener una mejor aproximación del *r-set* a representar es necesario ir reduciendo el tamaño del cubo base y, por tanto, aumentar el número de cubos. Los esquemas de subdivisión espacial reemplazan la partición regular del espacio por una partición que se adapta a la forma del objeto a representar, permitiendo un tamaño variable en las distintas regiones que forman dicha partición.

La idea de los esquemas jerárquicos de particionamiento espacial se basa en subdividir una región del espacio en varias regiones, y esta misma subdivisión ser aplicada recursivamente a cada una de las nuevas regiones así creadas, hasta que se cumple un criterio de parada para la subdivisión. Considerando que la región inicial, normalmente de forma cúbica, engloba al *r-set*, el criterio de parada utilizado es poder clasificar la región como conteniendo únicamente una porción del sólido o conteniendo únicamente espacio exterior a éste. Las regiones generadas utilizando el procedimiento recursivo se mantienen en un árbol denominado *árbol de particionamiento espacial*. La mayoría de estos esquemas utilizan planos para establecer la subdivisión de las regiones. La elección de estos planos determina las distintas variantes de este tipo de esquemas. A continuación mostraremos algunas de las variantes más utilizadas:

Bintrees En este caso, la región se subdivide en dos subregiones de igual tamaño utilizando para ello la alternancia estricta y cíclica, en cada paso de recursión, de tres planos ortogonales con los tres planos del sistema de coordenadas cartesiano [ST85, Die88]. Cada región puede representar tres situaciones excluyentes: contener completamente un porción del *r-set*, con lo que se almacena valor negro

para la región; contener completamente espacio vacío, con lo que se almacena valor blanco; o englobar parte del *r-set* con lo que se procede al siguiente paso de recursión.

La representación presenta la forma de árbol binario (de aquí el término *bintree*). Los nodos internos representan regiones parcialmente ocupadas por el sólido que han sido divididas por un determinado plano y los nodos terminales representan regiones totalmente ocupadas por el espacio que ocupa el *r-set* o regiones que representan espacio libre.

BSP-Trees Al igual que en los *bintrees*, se utiliza un solo plano para dividir la región en cada paso de recursión. Sin embargo, en este caso se utiliza un plano con una orientación y posición arbitrarias. Los *BSP-Trees* fueron utilizados inicialmente para establecer una partición de un entorno virtual compuesto por polígonos en la que se obtenía como resultado una base de datos jerárquica de polígonos [FKN80]. Posteriormente, Fuchs [FAG83] extendió su uso para acelerar la visualización de *r-sets* definidos mediante caras poligonales. Más adelante, esta estructura jerárquica se ha utilizado propiamente como representación de *r-sets* definidos mediante sólidos poliédricos [TN87, Nay90a, NAT90, PY90].

La representación es un árbol binario en el que los nodos internos almacenan la información del plano de subdivisión y los nodos terminales representan regiones del sólido o regiones del espacio vacío. La elección de la posición y orientación de cada plano en cada paso de recursión, junto con el criterio de parada de la recursión, depende de la utilidad que se le vaya a dar al árbol. Por ejemplo, si se va a usar para detección de colisiones, el criterio de subdivisión tratará de obtener una partición con el objetivo de llegar a nodos terminales en los que la colisión sea fácil de detectar. Hay que tener en cuenta que si no se aplican restricciones, el plano de subdivisión puede atravesar caras del objeto poliédrico por lo que hay que dividirlas en dos, ya que los nodos terminales deben ser objetos independientes. En su aplicación al modelado de sólidos poliédricos, los planos de subdivisión se eligen de forma que coincidan con los planos de las caras que definen el *r-set*, por lo que los nodos terminales pueden ser etiquetados como negro si contienen un porción del sólido o como blanco si continen espacio vacío. Aunque esto evita el cálculo de la intersección plano de subdivisión–cara, como resultado se obtienen árboles peor balanceados que con otras posibles elecciones de los planos de subdivisión.

kd-trees Los *kd-trees* [Ben75] pueden considerarse una generalización de los *bintrees* y a su vez un caso particular de los *BSP-trees*. Al igual que los primeros, cada paso de recursión determina un plano que subdivide a la región cíclicamente, sin embargo, al igual que los segundos, el plano no divide a la región en dos subregiones de igual tamaño. La diferencia con los *BSP-trees* estriba en que el plano en este caso es ortogonal a los planos cartesianos. A pesar de que se han utilizado principalmente como índices espaciales también existen trabajos que los utilizan como una representación propiamente dicha [GK04].

Octrees A diferencia de los dos esquemas anteriores, el *octree* utiliza tres planos en lugar de uno para subdividir la región en cada paso de recursión. Los tres planos son siempre los mismos y ortogonales a los planos del sistema de coordenadas cartesiano. Estos planos dividen la región en ocho subregiones de igual tamaño

Nombre	Planos Ortogonales	p	s
Bintree	Si	1	2
BSP-tree	No	1	2
kd-tree	Si	1	2
Octree	Si	3	8

Tabla 1.1: La tabla muestra las propiedades de los métodos de particionamiento espacial relacionadas con el número de planos de subdivisión de la región, p, y el número de subregiones que se forman, s.

denominadas *octantes*. El *octree* es una generalización al espacio 3-D de los *quadtrees* [Mea82, Gar82a, Sam84].

La representación es un árbol en el que cada nodo interno tiene ocho nodos hijo asociados, uno por cada octante en el que se ha subdividido la región que representa el nodo padre. Gracias a la invarianza de los planos de subdivisión, los nodos internos no necesitan almacenar información adicional, representando regiones parcialmente ocupadas por el sólido que han sido subdivididas. Los nodos terminales representan regiones que ocupan totalmente parte del espacio del *r-set* o bien regiones ocupando espacio vacío. Dichos tipos de nodo terminal se etiquetan respectivamente como negro o blanco.

Los *octrees* se han aplicado, al igual que los *BSP-Trees* como estructura de indexación espacial [Sam90, FK85] y como herramienta para acelerar la detección de colisiones [Sam90]. En el ámbito del modelado de sólidos se han utilizado con profusión [Mea80, Mea82, FYK84, Gar82b, SW88a, SW88b].

En la tabla 1.1 se muestran las propiedades de los métodos de particionamiento espacial que se acaban de exponer donde p representa el número de planos utilizados para la subdivisión y s el número de subregiones formadas.

1.4.2. Representaciones constructivas

Como se ha visto en la sección anterior, los esquemas de descomposición espacial presentan la característica común de que las celdas forman una partición del sólido abstracto definido como conjuntos de puntos continuos. En las representaciones constructivas, se utilizan unas "celdas simples" (primitivas) que permiten representar los conjuntos de puntos mediante operadores booleanos regularizados y transformaciones rígidas aplicadas sobre estas. Tradicionalmente se comparan ambos esquemas indicando que en los de descomposición, la única operación que se aplica a las celdas es el "pegado", mientras que las operaciones aplicadas a las primitivas de las representaciones constructivas son mucho más potentes [Mä88].

Para definir las primitivas se utilizan conjuntos de semiespacios, que a su vez suelen venir definidos mediante superficies implícitas, $F(x, y, z) = 0$. De esta forma, la superficie divide el espacio en dos porciones: $F(x, y, z) >= 0$ y $F(x, y, z) < 0$. Las primitivas vienen definidas normalmente por el resultado de la intersección del conjunto de semiespacios asociado. Las representaciones están formadas por árboles binarios. Los nodos no terminales representan operaciones que pueden ser transformaciones rígidas u operaciones booleanas regularizadas: unión, intersección o diferencia. Los nodos terminales

pueden representar las primitivas que permita el esquema de representación o bien los argumentos de las transformaciones rígidas.

Debido a la estructura de la representación, estos esquemas se diferencian de los anteriores en que no se mantienen de forma explícita las características geométricas y topológicas del objeto representado. Para realizar operaciones sobre el objeto es necesario evaluar el árbol de la representación ya que sus distintos nodos son los que especifican la forma del objeto final. En cuanto a las propiedades, el esquema es no ambiguo y conciso pero no es único. La ventaja que presenta es que se puede garantizar la validez de las representaciones de forma automática utilizando operaciones de combinación que aseguren la clausura del objeto resultado de la aplicación de dichas operaciones [Req80, Mä88]. En la figura 1.5 se muestran dos ejemplos de árboles CSG.

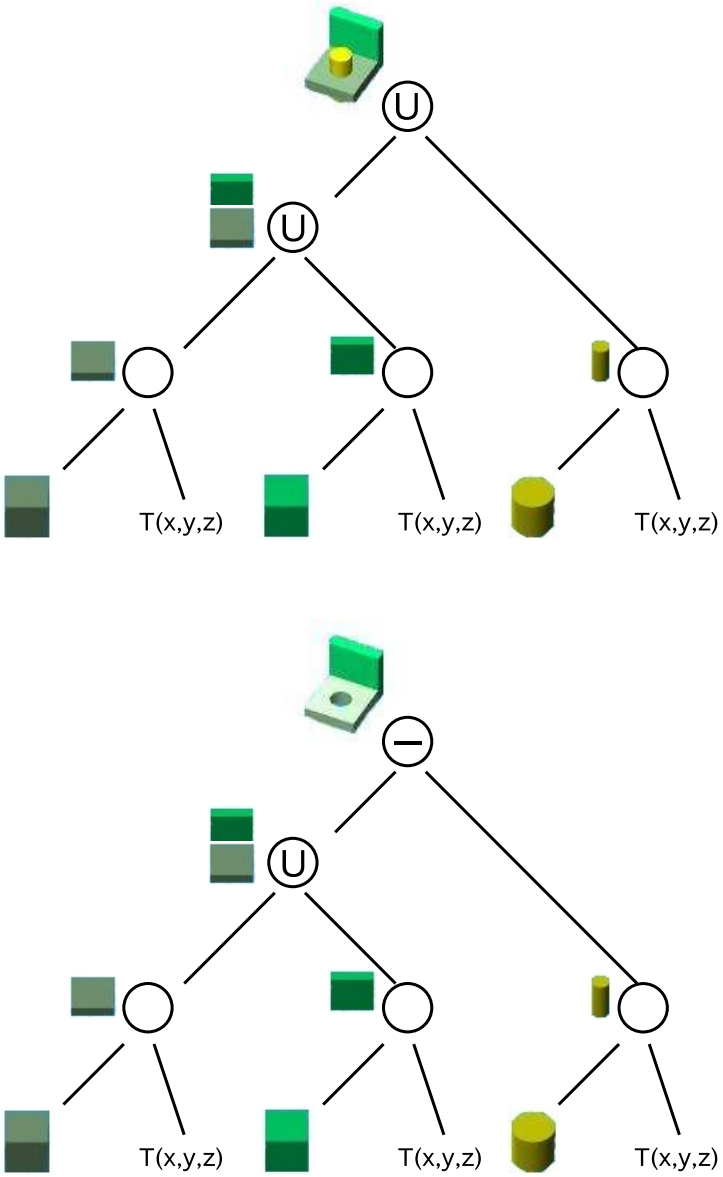

Figura 1.5: Dos ilustraciones de dos árboles que describen el proceso de construcción de dos sólidos diferentes mediante la utilización del mismo conjunto de primitivas básico y la aplicación de transformaciones geométricas y operaciones booleanas.

1.4.3. Modelado implícito

Este enfoque del modelado es apto tanto para el modelado de sólidos como para el modelado de volúmenes. Se basa en la aplicación de funciones implícitas. Esta forma de expresar funciones permite asignar una valor a cualquier punto del espacio en el que estemos trabajando (3-D en nuestro caso, $f(x, y, z)$). A partir de este hecho, se puede definir una superficie implícitamente estableciendo una condición $f(x, y, z) = 0$ (o un volumen mediante la condición $f(x, y, z) \geq 0$). La función f no describe explícitamente la superficie (o el volumen) pero implica su existencia. En la literatura se conoce también a esta función como campo escalar, función de campo o función potencial por su similitud con los campos de energía que se estudian en física.

Por consiguiente, la función f permite caracterizar un volumen. Muchos tipos de funciones proporcionan un valor distinto de cero para un punto p, y este valor es proporcional a la distancia entre p y la superficie $f(x, y, z) = 0$. El modelado implícito usa esta propiedad y utiliza funciones de distancia. Las funciones de distancia se calculan una distancia euclídea a un conjunto de primitivas de generación del campo, como por ejemplo puntos, líneas o polígonos. Para profundizar en este campo se pueden consultar por ejemplo [MW96, Blo97].

1.5. Modelado de volúmenes

La principal característica que diferencia al modelado de volumen frente al modelado de sólidos estriba en la necesidad de que las representaciones muestren la falta de homogeneidad en todo el volumen ocupado por el objeto, de forma que aparezca representada la variación de alguna(s) propiedad(es) en todo su volumen. Por tanto, los esquemas de representación de volumen deben establecer una correspondencia entre los conjuntos de puntos mediante una relación trivaluada cuya variable independiente sea una posición en el espacio 3-D y cuya variable dependiente sea un escalar, una tupla de escalares o un vector que represente dicha variación de propiedad(es).

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

1.5.1. Representaciones de volúmenes

El desarrollo de los aparatos de medida que permiten el muestreo de una determinada propiedad de objetos del mundo real ha generado la necesidad de establecer representaciones computacionales que permitan trabajar con tales datos. Junto a estos dispositivos, la simulación de fenómenos físicos mediante métodos numéricos proporciona de la misma forma otro amplio conjunto de datos volumétricos (*datasets*).

Con el objetivo de obtener una representación volumétrica, se obtienen las coordenadas de un número, normalmente elevado, de puntos y se mide algún valor de propiedad en cada una de éstas, muestreando todo el volumen ocupado por el objeto de interés. Este conjunto de muestras, posición y valor, podría considerarse como una representación computacional. Sin embargo, para evitar la ambigüedad de esta representación es necesario dotar de significado topológico a la colección de muestras. El modelo comúnmente utilizado se basa en utilizar *celdas* para especificar la topología de la representación, y las coordenadas de las muestras para especificar la geometría. De esta forma se consiguen representaciones completas a partir de los conjuntos de puntos muestreados. Las coordenadas mantienen información de los datos conocidos

y las celdas permiten interpolar entre los puntos, permitiendo establecer una valor de propiedad para todo el dominio de interés (volumen) de la relación trivaluada. Las celdas se definen especificando una secuencia (ordenada) de puntos, que especifica las relaciones de conectividad, la cual determina la topología de la celda. Las coordenadas de dichos puntos especifican la geometría de cada celda.

La estructura que presentan los puntos especifica las relaciones que presentan las celdas y los puntos, y permite clasificar el tipo de representaciones utilizadas. Las representaciones se pueden caracterizar atendiendo a si su estructura es regular o irregular. Una representación es regular si existe una relación matemática única entre los puntos y las celdas. Si los puntos presentan una estructura regular, entonces la geometría de la representación es regular. Si las relación topológica entre las celdas es regular, entonces la topología de la representación es regular. Los datos que presentan una estructura regular puden ser representados implícitamente, con el consiguiente ahorro de memoria y tiempo de procesamiento. Por el contrario, los datos que presentan una estructura irregular deben ser representados explícitamente. A continuación se presentan algunas de las representaciones basadas en celdas más comúnmente usadas [SML96] (ver figura 1.6):

Rejilla uniforme La representación se basa en una colección de puntos y celdas dispuestas formando una rejilla rectangular y regular. La rejilla uniforme es regular tanto en su geometría como en su topología por lo que pueden ser representadas en su totalidad de forma implícita. La representación solamente necesita mantener el número de puntos por dimensión, las coordenadas de un punto origen que actua como referencia para las coordenadas del resto de puntos y la distancia entre puntos en cada una de las dimensiones. Si la distancia entre puntos en cada dimensión coincide se denomina isotrópica, y si varía se conoce como anisotrópica.

Rejilla rectilínea Formada por una colección de puntos y celdas dispuestas en una rejilla regular. Mientras que la topología es regular, la geometría solo es parcialmente regular. Esto significa que los puntos están alineados con los ejes de coordenadas pero el espacio entre puntos puede variar.

Rejilla estructurada La representación presenta una topología regular y una geometría irregular. La rejilla puede deformarse en cualquier configuración en la que las celdas ni se solapen ni se autointersecten.

Rejilla no estructurada La topología y geometría de la representación no están estructuradas.

Las representaciones basadas en celdas presentan el problema de que la relación trivaluada que permite calcular el valor de propiedad en cualquier punto del volumen solo aparece definida para los puntos representados. Por consiguiente, es necesario realizar una estimación de los valores de propiedad situados entre los puntos, o lo que es lo mismo, contenidos en las celdas. Con este fin se utilizan normalmente funciones de interpolación.

1.5.2. Funciones de interpolación

Las representaciones basadas en celdas mantienen solamente valores de propiedad en los puntos, independientemente de que estos hayan sido suministrados por aparatos de medida o hayan sido muestreados a partir de representaciones continuas. A menudo

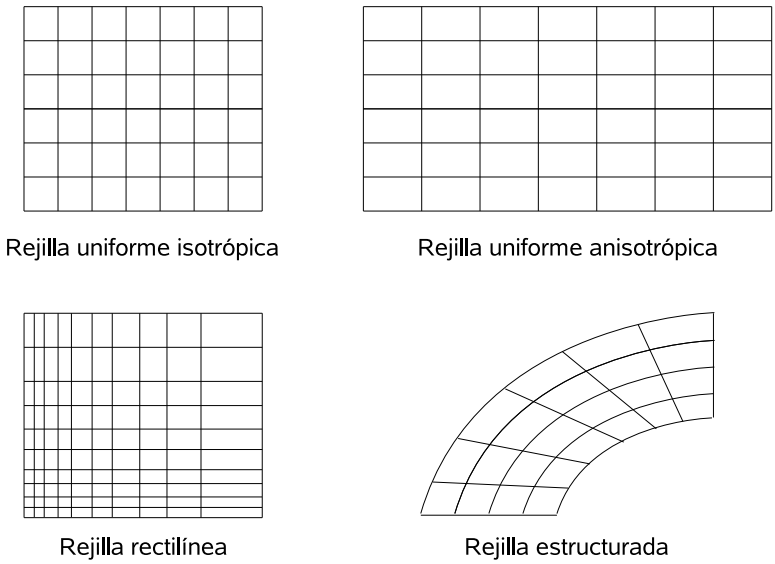

Figura 1.6: Ejemplos 2-D de tipos de estructuras con forma de rejilla.

es necesario determinar el valor de propiedad en posiciones distintas de los puntos, por lo que se hace necesario interpolar los valores de propiedad situados en los puntos mediante *funciones de interpolación*. Las funciones de interpolación permiten calcular el valor de propiedad de cualquier punto situado en el interior de una celda en base al valor de propiedad de los puntos de la celda. Marschner y Lobb [ML94] han presentado una gran variedad de posibles funciones de interpolación.

Para calcular el valor de propiedad de un punto p contenido en una celda constituida por los puntos p_i es necesario disponer de:

- Los valores de propiedad asociados a cada punto p_i.
- Las coordenadas paramétricas del punto p en la celda.
- La función de interpolación escogida para el tipo de celda.

De esta forma, la función de interpolación es una combinación lineal de los valores de propiedad en los puntos p_i de la celda que presenta la siguiente forma general:

$$d = \sum_{0}^{n-1} W_i \cdot d_i \tag{1.1}$$

donde d es el valor de propiedad en la posición (r, s, t) de la celda, d_i es el valor de propiedad asociado al punto p_i y W_i es el peso asociado al punto p_i. Los pesos (funciones de interpolación) son funciones de las coordenadas paramétricas $W_i = W(r, s, t)$ del punto p.

Debido a que es primordial mantener el valor de propiedad asociado a los puntos de la celda, cuando el punto p coincide con cualquier punto p_i es necesario que el valor de propiedad calculado coincida con el del punto. Para ello, se imponen la siguiente restricción adicional a los pesos:

$$W_i = 1, W_j = 0 \quad \text{cuando} \quad p = p_i \wedge i \neq j$$

Para dotar de coherencia al valor calculado en el interior de las celdas, adicionalmente se necesita que d no sea menor que el mínimo valor de propiedad asociado a los p_i ni mayor que el máximo. Por consiguiente, los pesos también deben satisfacer la siguiente restricción:

$$\sum W_i = 1, \quad 0 \leq W_i \leq 1$$

1.5.3. Fuentes de datos volumétricos

En esta sección se van a presentar algunas de las fuentes más comunes que proporcionan los valores de propiedad que posteriormente serán utilizados para construir las representaciones de volumen. Normalmente estos datos son denominados "*datasets*".

1.5.3.1. Imagen médica 3-D

La imágen médica comprende un conjunto de técnicas para visualizar estructuras internas del cuerpo humano. Para ello se utilizan los escáner, los cuales permiten examinar el cuerpo de un paciente tomando muestras de la distribución de alguna propiedad física en todo su cuerpo. Cada técnica de escaneo proporciona información de una o más propiedades físicas. Las principales técnicas utilizadas son Tomografía Axial Computerizada (TAC), Imagen por Resonancia Magnética (IRM), Medicina Nuclear (en sus modalidades PET y SPECT) y Imagen por Ultrasonidos (US). Se puede consultar una información más detallada en [Com85, ea89, SFF91].

Los escáneres TAC miden el grado en el que los tejidos son trasparentes a los rayos X. Distintos tejidos presentan diferentes coeficientes de absorción. Los rayos son detectados y en función de su intensidad se determina la densidad del tejido. Los valores de propiedad del *dataset* representan la densidad de los tejidos. Normalmente se obtienen *datasets* de 512^3 muestras con una precisión de 12 bits. Permiten un alto contraste entre tejidos densos (huesos) frente a tejidos blandos pero, como contrapartida, no permiten un alto contraste entre tejidos blandos.

Los escáneres MRI miden la oscilación de los núcleos de los átomos de hidrógeno sometidos a un campo magnético. La elección de este átomo se debe a que está presente en el cuerpo humano en densidades muy elevadas. La señal que miden es un pulso en radiofrecuencia. Los *datasets* obtenidos son normalmente de 512^3 muestras con una precisión de 12 bits. Las principal ventaja de la técnica MRI frente a la TAC es el mejor contraste entre los tejidos blandos, en detrimento del tejido óseo.

Los escáneres de medicina nuclear utilizan la emisión de radioactividad. Se introduce una sustancia radioactiva en el cuerpo del paciente y la radiación que producen ciertos núcleos atómicos es medida mediante detectores de rayos gamma. La información que se obtiene es la concentración del fármaco radiactivo en un determinado punto. La ventaja de esta modalidad es que proporciona imágenes fisiológicas en lugar de las imágenes anatómicas que proporcionan las técnicas MRI y TAC. Como contrapartida, los datos presentan mayor ruido que los obtenidos en las modalidades anteriores.

Los escáneres de ultrasonido (ecógrafos) están basados en la reflexión de las ondas de sonido en las fronteras entre tejidos, las cuales producen un fenómeno de eco. Los *datasets* que se obtienen representan la fuerza del eco que se obtiene en cada punto.

Debido a las características estructurales de los *datasets* obtenidos en las distintas modalidades de imagen médica 3-D, el esquema de representación más ampliamente utilizado es la rejilla uniforme.

1.5.3.2. Microscopía confocal

A diferencia de la microscopía óptica tradicional, la microscopía confocal permite eliminar la luz procedente de las zonas fueras del foco, con lo que se obtiene un mayor contraste y, además, permite tomar secciones ópticas a diferentes profundidades obteniendose información tridimensional de la muestra. Para más información se puede consultar [Mar03, Paw90]. Por el mismo motivo que la imagen médica 3-D, la representación utilizada es la rejilla uniforme.

1.5.3.3. Simulación de fenómenos físicos

El *análisis de diferencias finitas* es una técnica de análisis numérico que permite aproximar la solución a sistemas de ecuaciones en derivadas parciales. Estos sistemas de ecuaciones permiten simular el comportamiento de sistemas tales como dinámica de fluidos y transferencia de calor. Los *datasets* obtenidos mediante esta técnica se suelen representar por medio de una rejilla estructurada.

El *análisis de elementos finitos* es otra técnica numérica utilizada para solucionar los sistemas de ecuaciones en derivadas parciales. Estas técnicas se aplican por ejemplo al análisis estructural, análisis vibracional y de fatiga. Los *datasets* obtenidos mediante esta técnica se suelen representar por medio de una rejilla no estructurada.

1.5.3.4. Muestreo de objetos definidos de forma continua: voxelización

La idea subyacente en la voxelización [KS86, A.87b, WK93] es muestrear un objeto definido de forma continua mediante expresiones analíticas para obtener un *dataset*. La forma en la que se realiza el muestreo puede ser utilizando un intervalo fijo entre muestras o un intervalo que se adapte a determinadas características del objeto. Los primeros algoritmos de voxelización fueron binarios y asignaban 1 a los puntos interiores al objeto y 0 a los exteriores [A.87a]. Este enfoque ignora la teoría de muestreo y por tanto sufre del efecto de *aliasing*. Las técnicas de voxelización que permiten una representación suave de los objetos se clasifican en dos grandes enfoques: las técnicas de filtrado 3-D y las técnicas de campos de distancias [PT92, SK99, Sra01].

1.5.4. Operaciones

En esta sección se presenta una clasificación general del tipo de operaciones que se llevan a cabo sobre representaciones discretas de volumen. Para una descripción más detallada se puede consultar [SML06].

Operaciones geométricas. Estas operaciones modifican la geometría del *dataset* manteniendo inalterada su topología. Normalmente se trata de modificar las coordenadas de las muestras mediante transformaciones rígidas de rotación, traslación o escalado. Aparte, las operaciones de modelado que alteran la forma del *datasets* forman parte de esta categoría.

Operaciones topológicas. Las principales operaciones de este tipo son: determinar la dimensión topológica de una celda y acceder a las celdas vecinas que comparten caras, aristas o vértices.

- Determinar la dimensión topológica de una celda. Dada una celda C_i de dimensión topológica d, la celda está compuesta implícitamente por celdas

frontera de dimensión topológica $d-1, d-2, \ldots, 0$. En nuestro caso, normalmente trabajamos con $d = 3$ y las dimensiones menores se corresponden con caras, aristas y vértices. Este tipo de operaciones devuelven información sobre el número de celdas de dimensión menor que forman la frontera de la celda, así como una lista ordenada de los vértices que definen cada una de las celdas de menor dimensión.

- Acceder a celdas vecinas. Esta operación permite recabar información topológica sobre la adyacencia de celdas. Las operaciones típicas son determinar las celdas vecinas a una dada u obtener una lista de todas las celdas que contienen un punto determinado.

De forma análoga, las operaciones de modelado forman parte de esta categoría.

Operaciones sobre propiedades. Convierten los valores de propiedad y crean nuevos valores de propiedad a partir de los datos de entrada. La estructura del *dataset* no se ve alterada por este tipo de operaciones. Las técnicas de correspondencias de color (del inglés *color mapping*) mediante tablas de color o funciones de transferencia son un ejemplo típico de este tipo de operaciones. Las técnicas de extracción de isosuperfices también pueden ser consideradas en este tipo de operaciones, aunque puedan hacer uso de operaciones topológicas.

1.6. Visualización de volúmenes

La visualización es la operación que permite procesar las representaciones de volumen con el objetivo de obtener imágenes. A continuación se describen las principales estrategias que se han llevado a cabo haciendo hincapié en las técnicas que requieren una representación intermedia basada en primitivas gráficas poligonales.

Los primeros trabajos sobre visualización de volúmenes trabajaban *slice* a *slice*. La idea clave era extraer los contornos que delimitan el objeto de interés sobre cada uno de los *slices* y posteriormente unir esos contornos mediante polígonos para obtener una aproximación de la superficie frontera [Kep75, FKU77, CS78, LLC88, CLLC90, OPC96, BMM$^+$07]. El problema se conoce como reconstrucción de objetos 3-D a partir de series de secciones transversales. Posteriormente, Herman presentó el algoritmo *cuberille*, que permite evitar el paso de detección de contornos en cada *slice*. En este trabajo se considera que las muestras (*voxels*) forman un conjunto de cubos que no se solapan. El algoritmo de detección de superficie produce una aproximación a la superficie a visualizar utilizando las caras de los cubos. A diferencia de los primeros métodos, este garantiza la continuidad de la superficie extraida [HL79, AFH81, HU83].

A finales de los 80 y principios de los 90 varios trabajos abordan la visualización de volúmenes usando para ello una aproximación de superficies definidas implícitamente llevada a cabo mediante triángulos. Estas isosuperficies se pueden "extraer" bien a partir de los *datasets* [LC87, CLL88] o bien a partir de expresiones analíticas [WMW86b, WMW86a, Blo88, Blo94]. Junto a esta estrategia surge otra forma de visualizar volúmenes que sigue una estrategia más directa, permitiendo evitar el paso intermedio de generación de primitivas poligonales[Bli82, FGR85, DCH88, DNFM90, Lev88, Lev90]. A continuación se describen brevemente algunos trabajos realizados en el marco de estas dos estrategias, centrándonos en la extracción de isosuperficies que es el enfoque que se ha utilizado en nuestro trabajo. En el artículo de Brodlie [BW01] puede encontrarse un estudio más extenso de los tipos de técnicas para visualización de volúmenes.

1.6.1. Visualización directa de volúmenes

La visualización directa de volúmenes (del inglés *direct volume rendering*) genera imágenes directamente a partir del *dataset* sin la necesidad de extraer previamente una representación poligonal. Estas técnicas utilizan un *modelo óptico* para establecer una correspondencia entre los valores de las muestras y determinadas propiedades ópticas como el color y la opacidad. Durante el proceso de generación de imágenes, las propiedades ópticas se acumulan a lo largo de cada rayo de visión para formar una imagen de los datos. Los valores asociados a posiciones distintas a las de las muestras se obtienen mediante la interpolación de elementos de volumen vecinos. Los parámetros ópticos pueden venir especificados por los valores de las muestras, o pueden calcularse aplicando *funciones de transferencia* sobre los datos. Las imágenes se crean muestreando el volumen a lo largo de los rayos de visión y acumulando las propiedades ópticas obtenidas para cada rayo.

Existen dos grandes estrategias para abordar el problema basadas en el orden en el cual los *voxels* son procesados para generar la imagen: técnicas de *ordenación por imagen* y técnicas de *ordenación por objeto*. Las técnicas de ordenación por imagen recorren el espacio de imagen y determinan los *voxels* que contribuyen al valor del *pixel* que se está considerando [Lev88, SK94]. Por el contrario, las técnicas de ordenación por objeto recorren el *dataset* y determinan los *pixels* que pueden verse afectados por cada *voxel* [Wes90]. Ken Brodlie y Jason Wood [BW01] presentan una revisión de las distintas técnicas que se han desarrollado. Para una introducción a la visualización directa de volúmenes se puede consultar el libro de Lichtenbelt [LCN98].

Muchos trabajos han mejorado el rendimiento de los algoritmos de visualización directa mediante la utilización de técnicas de aceleración hardware. Este campo de trabajo se conoce como *visualización de volumen basada en texturas*. La idea se basa en almacenar en la memoria de la tarjeta gráfica el *dataset* (o parte de él) como una textura 3-D y utilizar un conjunto de polígonos que atraviesen el volumen perpendicularmente con respecto a la dirección de vista. Para cada vértice, se calcula una coordenada de textura 3-D y la función de transferencia y se genera la imagen final. Para una descripción detallada de esta técnica se puede consultar [Fer04].

1.6.2. Aproximación de superficies definidas implícitamente

La idea de esta estrategia se basa en aproximar mediante una representación poligonal (normalmente una malla de triángulos) una isosuperficie definida a partir de una representación de volumen. Una superficie definida implícitamente esta formada por todos los puntos $x \in \mathbb{R}^3$ tales que $\rho(x) = c$ siendo $\rho : \mathbb{R}^3 \mapsto \mathbb{R}$ un campo escalar y $c \in \mathbb{R}$ una constante. La superficie resultante se denomina c-superficie. El campo puede estar representado mediante una función continua o como un conjunto de valores muestreados y una función de interpolación por partes, correspondiendo cada parte con una celda de la representación. En el caso de representaciones continuas, por ejemplo una función implícita de la forma $f(P_i) = 0$, el espacio se particiona en un conjunto de celdas y la representación implícita se evalua en los vértices de dichas celdas. El objetivo es encontrar las intersecciones de la superficie implícita con las aristas de las celdas y conectar los puntos de corte para formar polígonos o triángulos. En el caso de representaciones discretas mediante celdas, se utiliza un isovalor que determina una isosuperficie usando la función de interpolación de la representación $F(P_i) = c$. El procedimiento para determinar la aproximación poligonal a la isosuperficie es igual al

anterior caso. Existen multitud de factores a tener en cuenta en esta estrategia, prueba de ello es la multitud de trabajos llevados a cabo sobre el tema. A continuación revisaremos brevemente algunos de estos factores.

Los dos primeros trabajos que abordaron el problema de la extracción de isosuperficies fueron [WMW86b, LC87]. Ambos métodos utilizan una rejilla uniforme de celdas cúbicas del mismo tamaño. Sin embargo, a pesar de ser posterior, el algoritmo *Marching Cubes* de Lorensen es el que más amplia repercusión ha tenido (ver por ejemplo [Nie03, NY06]). A continuación presentaremos distintos problemas que ha planteado el uso de esta técnica y los trabajos que los han abordado. Consideramos dos aspectos de la extracción que permiten establecer una clasificación general del tipo de problemas que se plantean: *qué tipo de aproximación* se consigue para la isosuperficie y el *rendimiento* del método de extracción. Algunos temas relacionados con la bondad de la aproximación tienen que ver con garantizar la falta de ambigüedad topológica y con la precisión con que se aproxima la isosuperficie. El rendimiento tiene que ver con mejorar la localización de la celdas que son realmente atravesadas por la isosuperficie y con reducir el número de triángulos que van a ser visualizados. A caballo entre ambos aspectos podemos considerar las representaciones multiresolución que permiten establecer un compromiso entre el rendimiento que permite una representación simplificada del volumen y el error que se comete en el proceso de simplificación a la hora de obtener una isosuperficie determinada con respecto a obtenerla a partir de la representación original.

1.6.2.1. Tipo de aproximación a la isosuperficie

Podemos estudiar la aproximación poligonal a la isosuperficie considerando su correctitud topológica y el grado de precisión alcanzado con relación a la isosuperficie definida sobre el volumen. La forma cúbica, escogida mayoritariamente para las celdas, puede presentar problemas de ambigüedad topológica, lo cual puede provocar la aparición de agujeros en las superficies extraidas. Los métodos que abordan la solución a este problema se basan en asegurar la consistencia topológica a la hora de elegir las poligonalizaciones [MSS94] que aproximan a la isosuperficie, e incluso, asegurar la correctitud topológica (fidelidad a la geometría de la superficie, bien sea a $f(P_i) = 0$ en representaciones continuas, o a $F(P_i) = c$ en representaciones discretas) [WvG90, NH91, NB93, Nat94, vGW94, Che95, HTF97, CGMS00, The02, LB03, VTLS08]. Se puede optar incluso por realizar una tabla de triangulaciones que refleje explícitamente los 256 casos posibles para una celda cúbica [BM00].

Obviamente se puede optar por recurrir a una forma tetraédrica para las celdas, la cual no presenta ambigüedad topológica si se elige cuidadosamente una tetraedrización para las celdas cúbicas [PT90, GH95], se opta por una tetraedrización independiente de la partición cúbica [ZCT95], o por un particionamiento espacial en tetraedros incluidos en cubos orientados de distinta forma[CP98, TPG99] u otros tipos de poliedros[CTM03]. El principal inconveniente de los métodos de partición espacial en celdas tetraédricas es la generación de un mayor número de triángulos para aproximar la superficie.

El otro aspecto abordado es el incremento de la precisión de la representación poligonal que aproxima la porción de isosuperficie interna a cada celda. Los trabajos que abordan este problema se justifican en entornos en los que existe una resolución muy baja en el *dataset* o se realiza una exploración cercana del volumen [HTF97, Lop99, CGMS00, Bai00]. Al igual que cuando se utilizan celdas tetraédri-

cas en la representación de volumen, este tipo de trabajos presenta el problema del aumento del número de triángulos que aproximan la isosuperficie.

1.6.2.2. Rendimiento

El volumen de información que compone los datos volumétricos ha ido creciendo por encima de las capacidades de procesamiento y almacenamiento de la tecnología, como suele ser habitual en informática. Existen varias estrategias para mejorar el rendimiento de la extracción de isosuperficies a partir de *datasets* estructurados y no estructurados. Una estrategia pasa por realizar un preprocesamiento con el objetivo de permitir la búsqueda eficiente de las celdas que contienen una porción de la isosuperficie (celdas activas). Otra estrategia es generar una representación multiresolución a partir del *dataset* original.

Un inconveniente importante que plantea el uso de representaciones basadas en rejillas estructuradas es el enorme volumen de celdas que es necesario procesar para extraer una isosuperficie. Se ha mostrado en [vGW94] que alrededor del 70 % pueden ser celdas no atravesadas por la isosuperficie (vacias). Con el objetivo de minimizar el procesamiento de celdas vacias se han planteado varias soluciones. Una clase de estrategias utilizan información del campo escalar para evitar atravesar celdas vacias [WMW86b, Blo94, IK94, HB94]. Se conocen como algoritmos de seguimiento de contornos (del inglés *contour tracing*). Otro tipo de estrategias se basan en examinar las celdas en un paso de preprocesamiento, almacenando información para la clasificación de éstas en una estructura adicional [WvG92, IK94, SJ95, CMSS96, VT01]. Esta estructura adicional provoca un gasto extra de memoria en estos métodos. Livnat y otros [LSJ96] han realizado un resumen muy completo de los algoritmos de preprocesamiento.

Los esquemas de ordenación de celdas persiguen reducir el tiempo empleado en la extracción de la isosuperficie mediante la reducción del número de celdas que es necesario comprobar. A diferencia de estos, las estrategias multiresolución [CFM$^+$94, WG94, ZCK97] persiguen mejorar este tiempo reduciendo el número de celdas utilizadas en la representación, mediante la combinación de celdas de menor tamaño que cumplan una serie de condiciones. Para ello se utiliza un método de combinación basado en alguna medida de error [SZK95, SFYC96, WKE99]. Estas representaciones no solo permiten simplificar el número de celdas a comprobar, sino que además producen un menor número de triángulos en la aproximación.

Sin tener en cuenta las formas distintas a la cúbica que pueden ser elegidas para las celdas (ver por ejemplo[PT90, ZCT95, ZCK97, GR00], se han desarrollado métodos que utilizando representaciones adaptativas permiten adecuar la partición del espacio a determinadas características de la superficie [HW90, MS93, SFYC96, OR97, GP00, BMTB02] obteniéndose celdas de distinto tamaño. El problema fundamental estriba en los *agujeros* que se producen en la isosuperficie entre celdas adyacentes de distinta resolución. En el capítulo3 se mostrarán los trabajos que han propuesto soluciones a este problema.

Por último reflejar brevemente que se han desarrollado trabajos en los que se convierte el *dataset* inicial sobre el que se desea extraer la isosuperficie a un campo escalar de distancias a dicha isosuperficie. La isosuperficie extraida a partir de este nuevo campo a menudo proporciona mejores resultados que el campo original[HZB92, PT92, JC94, Gib98, JS01]. Obviamente el inconveniente que presenta es que el campo de distancias

depende de la isosuperficie escogida y si se varía el valor umbral es necesario construir un nuevo campo. Jones[JBS06] presenta un estado del arte de los campos de distancias.

1.6.3. Gradiente

La aplicación de diferentes tonalidades (*shading*) a las imágenes generadas por ordenador les proporcionan un grado de realismo extra. En la generación de imágenes a partir de representaciones volumétricas se utiliza a menudo el valor del gradiente para aproximar la normal a la superficie de cara a utilizar un modelo de iluminación. El modelo de Phong [Pho75] se usa ampliamente tanto en la extracción de isosuperficies como en la visualización directa. En el caso de la extracción de isosuperficies, nos interesa calcular el vector gradiente en los puntos que definen la geometría de la malla resultante.

El gradiente de una superficie es la derivada parcial de la superficie con respecto a las tres dimensiones, es decir, dada una función $f(x, y, z)$ el gradiente se define como:

$$\nabla f(x, y, z) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z} \right)$$

Cuando se trabaja con representaciones discretas la función $f(x, y, z)$ no es conocida, y por tanto el gradiente tiene que ser calculado en base a los valores de propiedad muestreados. Para calcular el valor del gradiente en la posición de una muestra se suele emplear el método de diferencias centrales. Según este método, el graciente en la posición de muestra (x, y, z) se calcula de la siguiente forma:

$$\begin{aligned} g(x, y, z) = \nabla f(x_i, y_j, z_k) \quad &= \quad (\frac{1}{2}(f(x_{i+1}, y_j, z_k) - f(x_{i-1}, y_j, z_k)), \\ &, \quad \frac{1}{2}(f(x_i, y_{j+1}, z_k) - f(x_i, y_{j-1}, z_k)), \\ &, \quad \frac{1}{2}(f(x_i, y_j, z_{k+1}) - f(x_i, y_j, z_{k-1}))) \end{aligned}$$

Para calcular el gradiente en cualquier punto distinto a las posiciones de las muestras es necesario utilizar la función de interpolación por partes $F(x, y, z)$ asociada a la representanción. En función de la dimensión topológica de la celda en la que está incluido el punto sobre el que se quiere calcular el gradiente se utiliza la función de interpolación. Por ejemplo, si es necesario calcular el gradiente en un punto de corte de la isosuperficie con una arista (dimensión topológica $d = 2$, y se está usando la función de interpolación trilineal como interpolante en la celda, basta con realizar una interpolación lineal de los gradientes situados en los vértices de la arista. Se pueden utilizar métodos de estimación del gradiente que tengan en cuenta el valor de propiedad de más muestras (se pueden consultar [PTWH90, Gos94]).

1.7. Conclusiones

Se ha mostrado el marco en el que se realiza la investigación sobre modelado de objetos reales, centrándonos en el establecimiento del marco formal y propiedades adecuadas para el modelado de volúmenes, junto con las principales estrategias de representación. Se han presentado los trabajos más relevantes, a nuestro parecer, especialmente en el campo de las representaciones discretas y la visualización mediante extracción de isosuperficies.

CAPÍTULO 2

Representación Multiresolución de Volúmenes

Las representaciones discretas de volumen tienen gran utilidad a la hora de modelar conjuntos de muestras obtenidas, bien mediante dispositivos de captura de valores de una determinada propiedad (o propiedades) de un objeto del mundo real, o bien muestreando los resultados numéricos de modelos de simulación computacionales. Estas representaciones proporcionan una forma de utilizar la distribución espacial de los valores muestrales para conseguir, mediante interpolación, obtener una distribución continua de dichos valores en el espacio. En otras palabras, una representación volumétrica discreta debe ser capaz de calcular el valor de propiedad asociado a cualquier punto $\boldsymbol{P}$ incluido en el volumen delimitado por las posiciones de las muestras del conjunto.

El constante incremento en la resolución de los conjuntos de muestras produce unos elevados requisitos de almacenamiento de las representaciones en el ordenador. Las representaciones discretas de resolución fija mantienen toda la información muestral, haciendo inviable su utilización para conjuntos de muestras de gran resolución. La alternativa pasa por construir representaciones discretas de resolución variable (multiresolución) a partir de las muestras, utilizando para ello algún mecanismo que permita reducir la cantidad de información que es necesario almacenar para representar los objetos.

El objetivo de la representación propuesta en este capítulo es ahorrar espacio de almacenamiento con respecto a una representación volumétrica de resolución fija teniendo en cuenta el concepto de *región homogénea*. Este concepto permite reducir la cantidad de información almacenada en regiones del volumen en donde no es necesario representar el número de muestras que está disponible, para tales regiones, en el conjunto de muestras inicial.

En este capítulo se muestran los principales enfoques que se utilizan a la hora de construir representaciones discretas, tanto de resolución fija como variable, los problemas que plantean estas últimas a la hora de permitir la multiresolución y nuestra propuesta de representación multiresolución. La sección *Representaciones volumétricas discretas* caracteriza este tipo de representaciones, presentando a continuación una posible definición lo suficientemente general para abarcar a las de resolución fija y variable. A

continuación, se describen los elementos principales de una representación de resolución fija bajo el punto de vista de los dos principales enfoques de trabajo: *voxels* y celdas. Seguidamente se estudia la continuidad que proporcionan ambos enfoques en este tipo de representaciones y se muestra la operación básica que permite calcular el valor de propiedad de cualquier punto $\boldsymbol{P}$ incluido en el volumen de la representación. La sección termina presentando las representaciones de resolución variable. Se estudia la utilización de los dos enfoques previamente comentados sobre este tipo de representaciones y se ponen de manifiesto las particularidades de cada uno de ellos sobre un *octree* que actua como estructura de datos soporte para las representaciones. Finalmente, se comparan ambos enfoques con respecto a su capacidad para reducir los requisitos de espacio y la continuidad proporcionada para el cálculo del valor de propiedad en cualquier punto $\boldsymbol{P}$ incluido en el volumen de la representación.

La sección 2.3 muestra los trabajos previos que se han realizado en el ámbito de la reducción del espacio requerido por las representaciones discretas. Se distinguen dos ámbitos: la compresión de volúmenes y la simplificación de volúmenes. La sección 2.4 presenta nuestra propuesta de representación multiresolución basada en el concepto de región homogénea, el cual explota la coherencia espacial de los valores de propiedad. Primeramente se plantea el concepto y se define formalmente sobre una rejilla regular y uniforme de muestras. A continuación, se muestra su aplicación para construir nuestra representación: *HRB-Octree*. Seguidamente se compara nuestra representación con los enfoques tradicionales de *voxels* y celdas para mostrar sus ventajas e inconvenientes. Por último, en la sección 2.5 se estudia el error en el cálculo del valor de propiedad en un punto que se obtiene al usar nuestra representación frente a usar una representación de resolución fija basada en el enfoque de celdas. Se muestra que dicho error solo depende de la tolerancia, fijada en el dominio de valores de propiedad, para la reducción de valores muestrales y se proporciona una expresión analítica que determina dicho error.

2.1. Representaciones volumétricas

Como se ha mostrado en el capítulo 1, el ordenador nos permite representar un objeto del mundo real mediante un modelo o representación computacional de éste. Se pueden distinguir varios tipos de representaciones en función de los distintos aspectos del objeto en los que deseamos que incida el modelo. En informática gráfica, se suelen seguir dos enfoques distintos a la hora de clasificar dichos modelos. Un primer tipo de modelos están interesados exclusivamente en la información del exterior del objeto, es decir, de su superficie, con el objetivo de poder determinar su forma. Un segundo tipo de modelos tratan de representar la falta de homogeneidad que presentan los objetos en todo su volumen, tanto en su superficie como en su interiors [Mä88, FvDFH95]. A este último tipo de modelos se les conoce como modelos de volumen o volumétricos.

Los modelos volumétricos pueden ser clasificados, en general, en dos grandes familias. Por una parte se encuentran los modelos que representan de forma simbólica, por medio de expresiones u objetos matemáticos, los objetos del mundo real. Los modelos de este tipo más ampliamente utilizados son los basados en ecuaciones e inecuaciones implícitas [Blo97]. Por otra parte disponemos de los modelos que representan objetos a partir de un conjunto de valores (o muestras) de una determinada propiedad del objeto distribuidas (o tomadas) en un determinado volumen del espacio. Denominaremos a estos últimos *representaciones volumétricas discretas*. La representación desarrollada en nuestro trabajo se enmarca en esta segunda categoría. Para fijar nuestro ámbito de

interés presentamos una definición general de representación volumétrica basada en el objetivo que se plantea el observador.

2.2. Representaciones volumétricas discretas

Las dos principales características de las representaciones de volumen discretas son: la naturaleza discreta de los datos a partir de los cuales se construyen las representaciones y la disposición o estructura espacial de dichos datos. El hecho de disponer del valor de propiedad solamente en determinadas posiciones del espacio implica que no tenemos conocimiento del valor de propiedad en el resto de posiciones del espacio asociado con la representación del objeto. Esto presenta un problema, ya que existen multitud de operaciones que deseamos realizar sobre nuestras representaciones, las cuales requieren determinar el valor de los datos en todo el volumen ocupado por el objeto representado. Este problema se soluciona en la mayoría de los casos utilizando algún tipo de *interpolación*.

Por otro lado, la estructura espacial de los datos, caso de que exista, permite clasificar las representaciones en *regulares* e *irregulares*. Las representaciones regulares presentan inherentemente una relación entre las posiciones en donde se encuentran situados los datos. Por ejemplo, si realizamos un muestreo en un conjunto de posiciones igualmente espaciadas, no necesitamos almacenar las coordenadas de todos los puntos, solo la posición del primer punto, el espacio entre puntos (intervalo o resolución de muestreo), y el número total de puntos. Las posiciones del resto de puntos del conjunto se conocen de forma implícita. Las representaciones irregulares no permiten este ahorro pero, como contrapartida, tienen la ventaja de poder representar la información densamente (mayor resolución) en regiones del espacio en donde cambia más rapidamente y almacenar menor información en zonas en donde no es necesaria tanta resolución. Por tanto, las representaciones irregulares permiten "adaptarnos" a la cantidad de información presente en las distintas regiones del objeto volumétrico representado. En ocasiones, la naturaleza del muestreo utilizado para la obtención de las muestras proporciona directamente la información en forma irregular. Por ejemplo, en geología se suelen hacer catas a distintas profundidades para poder estimar la composición del suelo. En este caso, tanto la distribución bidimensional de las catas sobre el terreno, como la profundidad de cada una de ellas, no suele proporcionar información distribuida regularmente.

Podemos definir una *representación volumétrica discreta* de un objeto como un conjunto finito S de pares de valores $(\boldsymbol{P}_i, v_i)$, donde v_i representa el valor de alguna propiedad del objeto situada en la posición $\boldsymbol{P}_i$ del espacio 3-D. Los pares pueden representar muestras de alguna propiedad de un objeto tomadas en distintas posiciones espaciales o evaluaciones de una función conocida f en un determinado volumen del espacio. El dominio al que pertenecen los valores v_i puede ser un único conjunto de valores (univaluado) representando una única propiedad del objeto volumétrico; el producto cartesiano de varios conjuntos de valores (multivaluado) representando un grupo de varias propiedades del objeto volumétrico; e incluso, v_i puede ser vectorial. Además, y de cara a permitir el cálculo del valor de propiedad en todo el espacio ocupado por el objeto representado, una representación discreta debe proporcionar una forma de interpolación entre los pares de valores $(\boldsymbol{P}_i, v_i)$. A continuación proponemos una definición más formal de representación volumétrica discreta:

Definición 3 (Representación Volumétrica Discreta) *Un representación volumétrica discreta es un par formado por:*

- *Un conjunto finito de muestras, S, cuyos elementos son pares $(\boldsymbol{P}_i, v_i)$. El primer elemento del par representa una posición en el espacio 3-D y el segundo representa un valor de propiedad perteneciente al dominio de valores de propiedad, $\mathcal{V}$. Además, cada punto $\boldsymbol{P}_i$ solamente puede tener asociado un único valor de propiedad v_i.*
- *Una función de interpolación, F, que permite estimar el valor de propiedad en cualquier punto del espacio interior delimitado por las posiciones representadas por los $\boldsymbol{P}_i$ de S.*

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

en donde $f(\boldsymbol{P}_i) = v_i$ representa el caso en el que la función a partir de la cual se han tomado las muestras es conocida, aunque puede darse el caso en el que no se conozca dicha función, por lo que sólo se dispondría del valor v_i asociado a cada muestra.

Como se ha mostrado en la sección 1.5.3 del capítulo 1, los datos de volumen se obtienen principalmente de *conjuntos de datos volumétricos* (*datasets*), siendo las principales fuentes de estos datos los dispositivos de imagen médica; o bien, mediante discretizaciones (*voxelizaciones*) de objetos matemáticos o modelos de simulación. Las representaciones volumétricas discretas se construyen a partir de estos datos de volumen.

Se ha mostrado que dependiendo de la estructura espacial de los elementos de S podemos distinguir entre reprentaciones volumétricas regulares e irregulares. Normalmente, todos los dispositivos de captura de datos volumétricos proporcionan el conjunto de muestras estructurado en forma regular. Por el contrario, los procedimientos que muestrean objetos matemáticos o modelos de simulación proporcionan un conjunto de muestras irregular. El disponer de la representación matemática del volumen posibilita la construcción de representaciones volumétricas irregulares en las que el intervalo de muestreo se adapta a determinadas características del objeto matemático. Sin embargo, en el caso de los conjuntos de datos proporcionados por dispositivos de captura, esto no es posible. La precisión (intervalo de muestreo) viene determinada por los aparatos y, por consiguiente, seleccionar distintos tipos de intervalos requiere volver a utilizar el aparato. Por este motivo, lo más común es obtener un conjunto de datos volumétricos con estructura regular.

Cuando construimos una representación volumétrica regular a partir de estos conjuntos de datos volumétricos, las posiciones de los elementos del conjunto S se sitúan a intervalos regularmente espaciados a lo largo de los tres ejes ortogonales que define el espacio euclideo cartesiano tridimensional. Cuando el espacio entre las posiciones a lo largo de cada eje es una constante, se dice que el conjunto S es *isotrópico*. Si el espacio entre posiciones es distinto para cada uno de los ejes se dice que el conjunto S es *anisotrópico*.

En el caso de que los elementos de S estén estructurados en forma de *rejilla regular*, independientemente de que S sea isotrópico o anisotrópico, podemos optar por representar de forma directa el objeto volumétrico mediante un *array 3-D*. Los índices del

array indican la posición de la muestra en la rejilla regular. El espacio en memoria ocupado por el *array* dependerá del conjunto de muestras que se tome en cada dimensión y del tamaño requerido para almacenar la información de cada muestra. A esta información, recogida para cada muestra, la denominamos *valor de propiedad*. Dependiendo de que el conjunto S sea isotrópico o anisotrópico será necesario almacenar en una o tres variables, respectivamente, la distancia entre muestras para cada uno de los ejes ortogonales. En la literatura podemos encontrar diferentes términos para denominar el *array* 3-D: búfer de volumen [Kau90, KCY93], "*cubic frame buffer*" o "*3D raster*".

Para completar una representación volumétrica discreta se ha de tener en cuenta que el dominio de valores de propiedad, $\mathcal{V}$ asociado al objeto volumétrico solamente se encuentra definido en posiciones discretas $(\boldsymbol{P}_i, v_i)$ del espacio 3-D. Para conseguir estimar los valores de propiedad en cualquier posición continua del espacio ocupado por el objeto volumétrico es necesario definir la función de interpolación $F(x, y, z)$. Normalmente se utiliza una función de interpolación en vez de una función de aproximación para garantizar que el valor proporcionado por dicha función en las posiciones de las muestras (elementos de S) coincida con el valor de propiedad v_i, asociado a cada posición. Por cuestiones de eficiencia de procesamiento es necesario acotar el número de elementos del conjunto S que intervienen en la función de interpolación, cuando ésta se aplica a calcular el valor de propiedad en una determinada región del espacio de la representación. Por tanto, el espacio ocupado por la representación se divide en *subvolúmenes*, los cuales vienen definidos por el número de elementos de S que van a ser utilizados para interpolar el valor de cualquier punto perteneciente a cada uno de dichos subvolúmenes. El número de elementos de S asociados con cada subvolumen junto con la función de interpolación escogida determinan la forma de cada subvolumen, tanto topológica como geométricamente. El número de muestras asociado a cada subvolumen viene normalmente determinado por el orden escogido para la función de interpolación y su forma vendrá dada por las posiciones $\boldsymbol{P}_i$ de las muestras que incluye.

En el caso de que los elementos de S estén estructurados en forma de *rejilla irregular* no se puede utilizar el *array* 3-D como estructura de datos soporte. Para poder representar los subvolúmenes de diferente tamaño se opta por utilizar una estructura de datos de particionamiento espacial como el *octree* o el *kd-tree*.

2.2.1. Representaciones discretas de resolución fija

En esta clase de representaciones, el conjunto S está distribuido de forma uniforme y regular. De esta forma los puntos $\boldsymbol{P}_i$ determinan una rejilla regular en el espacio ocupado por la representación. Las dos funciones de interpolación que se van a describir a continuación, junto con los elementos de S, definen dos rejillas 3-D que producen las dos interpretaciones comúnmente utilizadas a la hora de representar objetos mediante una representación volumétrica discreta regular. Supondremos, sin pérdida alguna de generalidad, que el conjunto S sobre el que se definen las dos funciones de interpolación es isotrópico. Por tanto, la estructura de los elementos del conjunto tiene forma de *rejilla regular cúbica*.

La función de interpolación más simple que se puede utilizar es una *interpolación de orden cero*, que es en realidad una función de cálculo del vecino más próximo. El valor de cualquier posición $\boldsymbol{P} = (x, y, z)$ del espacio 3-D es simplemente el valor de propiedad del elemento de S más cercano. Usando este método de interpolación existe una región de valor constante alrededor de cada una de las posiciones $\boldsymbol{P}_i$ de las muestras del conjunto S. Para calcular la posición de la muestra más cercana a $\boldsymbol{P}$ se necesita

una métrica que permita medir distancias en el espacio cartesiano 3-D (por ejemplo la Distancia Euclidea).

Ya que los elementos de S se encuentran estructurados formando una rejilla regular cúbica, cada región que rodea la posición de un valor de propiedad presenta una forma y tamaño uniformes. El subvolumen de valor constante que envuelve cada valor de propiedad se conoce como *voxel*, siendo cada voxel un cubo de seis caras, doce aristas y ocho vértices[1] . En este caso, solo es necesario un elemento del conjunto S para realizar la interpolación en cada uno de los subvolúmenenes (*voxels*) en los que se particiona el espacio ocupado por la representación.

En la figura 2.1 se puede observar la rejilla resultante para el caso 2-D. Esta rejilla permite determinar el valor de propiedad en cualquier posición continua del espacio ocupado por el objeto volumétrico. Las posiciones $\boldsymbol{P}_i = (x_i, y_i)$ de los elementos de S se encuentran situadas en los centros de los cubos.

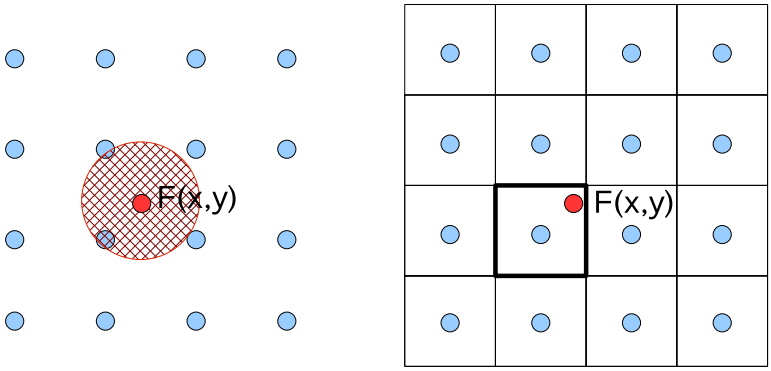

Figura 2.1: Ejemplo 2-D en el que $F(x, y)$ está definida por partes mediante una interpolación de orden cero. La imagen de la izquierda muestra que el valor de F en la posición (x, y) es el valor de propiedad v_i de la muestra $(x_i, y_i, v_i) \in S$ cuya posición asociada (x_i, y_i) está más cerca. La imagen de la derecha muestra que la región de valor constante centrada en la posición (x_i, y_i) de cada elemento de S tiene forma cuadrada, en el caso de que el conjunto S sea isotrópico.

A pesar de que se pueden usar funciones de interpolación de orden superior para definir la función $F(x, y, z)$, la función de interpolación más comúnmente utilizada es una función por partes conocida como interpolación de primer orden o *interpolación trilineal*. Su amplia utilización se debe a que proporciona un buen compromiso entre eficiencia de cálculo y exactitud en la aproximación. Al usar interpolación trilineal se asume que los valores de propiedad varían linealmente a través de las direcciones paralelas a uno de los tres ejes ortogonales. Sea $\boldsymbol{P} = (x, y, z)$ un punto situado en el interior de un subvolumen cúbico, conocido como *celda*, definido por las posiciones de ocho elementos vecinos del conjunto S. La función de interpolación trilineal es de la siguiente forma:

$$F(x, y, z) = a + bx + cy + dz + eyz + gxz + hxy + ixyz \tag{2.1}$$

donde

[1] En el caso de que el conjunto S sea anisotrópico, el voxel es un paralelepípedo regular.

$$
\begin{aligned}
a &= v_0 \\
b &= v_1 - v_0 \\
c &= v_2 - v_0 \\
d &= v_4 - v_0 \\
e &= v_6 - v_4 - v_2 + v_0 \\
g &= v_5 - v_4 - v_1 + v_0 \\
h &= v_3 - v_2 - v_1 + v_0 \\
e &= v_7 - v_6 - v_5 - v_3 + v_4 + v_2 + v_1 - v_0
\end{aligned}
$$

siendo $v_i,\ i \in \{0, 1, \ldots, 7\}$ los valores de propiedad de los elementos de S cuyas posiciones coinciden con las esquinas del cubo que se puede observar en la figura 2.2. x, y, z son variables que indican la posición del punto, dentro de la celda que se está considerando, en el cual se quiere calcular el valor de propiedad. Estas variables toman valores dentro del intervalo $[0, 1]$. De esta forma, la terna de valores $x = 0, y = 0, z = 0$ alude al vértice $\boldsymbol{V}_0$, la terna $x = 1, y = 0, z = 0$ al vértice $\boldsymbol{V}_1$ y así sucesivamente hasta $x = 1, y = 1, z = 1$ que alude al vértice $\boldsymbol{V}_7$.

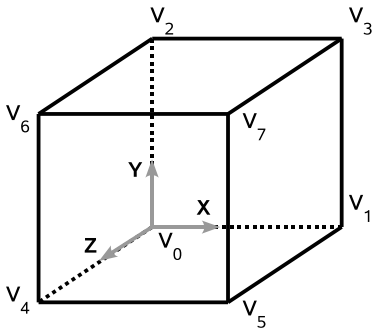

Figura 2.2: Elemento de interpolación trilineal (celda) con vértices etiquetados.

Utilizando esta función de interpolación, las posiciones $\boldsymbol{P}_i = (x_i, y_i)$ de los elementos de S coinciden con los puntos de intersección de los cubos que forman la rejilla 3-D. En la figura 2.3 se puede observar la rejilla resultante que nos permite determinar el valor de propiedad de cualquier posición continua del espacio ocupado por el objeto volumétrico. En este caso, son necesarios ocho elementos del conjunto S para realizar la interpolación en cada uno de los subvolúmenenes (celdas) en los que se particiona el espacio ocupado por la representación.

2.2.2. Cálculo del valor de propiedad en un punto sobre representaciones de resolución fija

La mayoría de las operaciones que se definen sobre representaciones de volumen requieren el cálculo del valor de propiedad en cualquier punto $\boldsymbol{P}$ del espacio determinado por las muestras del conjunto S. En ambos enfoques, *voxels* o celdas, disponemos de una rejilla regular 3-D que produce un mosaico de teselas cúbicas que cubre todo el espacio ocupado por el volumen. Dependiendo de la tesela en la que esté incluido el punto $\boldsymbol{P}$ y del enfoque utilizado para la representación, se calcula un valor de propiedad para dicho punto.

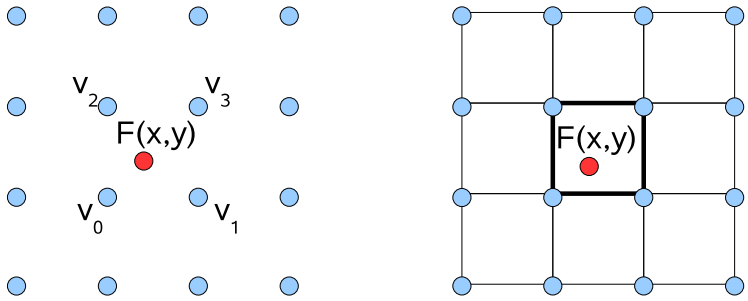

Figura 2.3: Ejemplo 2-D en el que $F(x, y)$ está definida por partes mediante interpolación trilineal. La imagen de la izquierda muestra que el valor de F en la posición (x, y) se calcula mediante la interpolación trilineal de los valores v_i situados en las esquinas de la celda. La imagen de la derecha muestra que las celdas coinciden con los cubos de la rejilla 3-D que une las posiciones (x_i, y_i) asociadas a las muestras $(x_i, y_i, v_i) \in S$.

En cualquiera de los enfoques utilizados es necesario definir una función que devuelva la tesela que contiene al punto. Basádonos en que ambos tipos de representación tienen un *array* 3-D como estructura de datos soporte, la función debe devolver el índice del *array* que representa el centro del *voxel* en el que está incluido el punto o, el índice asociado a la esquina que identifica la celda en la que se encuentra incluido el punto, dependiendo de que se use uno u otro enfoque. En otras palabras, la función debe obtener las muestras del conjunto S que son necesarias para realizar la interpolación en el subvolumen (tesela) que incluye a $\boldsymbol{P}$. Con este objetivo se define una función τ de la siguiente forma:

$$\begin{array}{rcl} \boldsymbol{\tau} & : & (\mathbb{R}_0^+)^3 \longrightarrow (\mathbb{Z}_0^+)^3 \\ & & (x, y, z) \longrightarrow [i, j, k] \end{array}$$

siendo, (x, y, z) el punto del volumen donde deseamos conocer el valor de propiedad y, $[i, j, k]$ los índices del *array* 3-D que representan la tesela buscada.

Un método que permita calcular el valor de propiedad v_P de un punto cualquiera $\boldsymbol{P} = (x, y, z)$ en el enfoque de *voxels* debe determinar cual es el *voxel* en el que está incluido dicho punto. A continuación describimos una forma eficiente de conseguir este objetivo:

SI Parte_Decimal$(\frac{x}{\Delta_x}) \geq 0,5$
ENTONCES
$i \leftarrow \lceil \frac{x}{\Delta_x} \rceil$
SI_NO
$i \leftarrow \lfloor \frac{x}{\Delta_x} \rfloor$

SI Parte_Decimal$(\frac{y}{\Delta_y}) \geq 0,5$
ENTONCES
$j \leftarrow \lceil \frac{y}{\Delta_y} \rceil$
SI_NO
$j \leftarrow \lfloor \frac{y}{\Delta_y} \rfloor$

SI Parte_Decimal($\frac{z}{\Delta_z}$) $\geq 0,5$
ENTONCES
$k \leftarrow \lceil \frac{z}{\Delta_z} \rceil$
SI_NO
$k \leftarrow \lfloor \frac{z}{\Delta_z} \rfloor$

siendo $\lceil . \rceil$ el operador que devuelve el primer entero mayor o igual que el operando y, $\lfloor . \rfloor$ el operador que devuelve el primer entero menor o igual que el operando. $\Delta_x, \Delta_y, \Delta_z$ representan la distancia entre las posiciones de los elementos de S en cada una de las dimensiones. Es necesario realizar la transformación de $\mathbb{R}$ a $\mathbb{Z}_0^+$ debido a que este último es el dominio de los índices de un *array*. Además, es necesario considerar las distancias entre elementos, ya que $\Delta_x, \Delta_y, \Delta_z$ no tienen porqué coincidir con 1, que es la distancia entre los valores índice en cada dimensión del *array*.

Una vez determinados los valores índice $[i, j, k]$, simplemente resta acceder al *array* 3-D para asignar el valor de propiedad al punto (x, y, z), ya que el método determina el *voxel* en el que éste se encuentra incluido. En la parte superior de la figura 2.4 se muestra gráficamente la idea para el caso 1-D. Se pueden observar las posiciones de las muestras, x, los índices del array, i, y el valor asociado a cada punto en función de la interpolación $F(x)$. Δ_x representa la distancia entre muestras. Como puede comprobarse, la función $F(x)$ asociada a al representación es una función por partes que presenta discontinuidades de salto entre *voxels* vecinos.

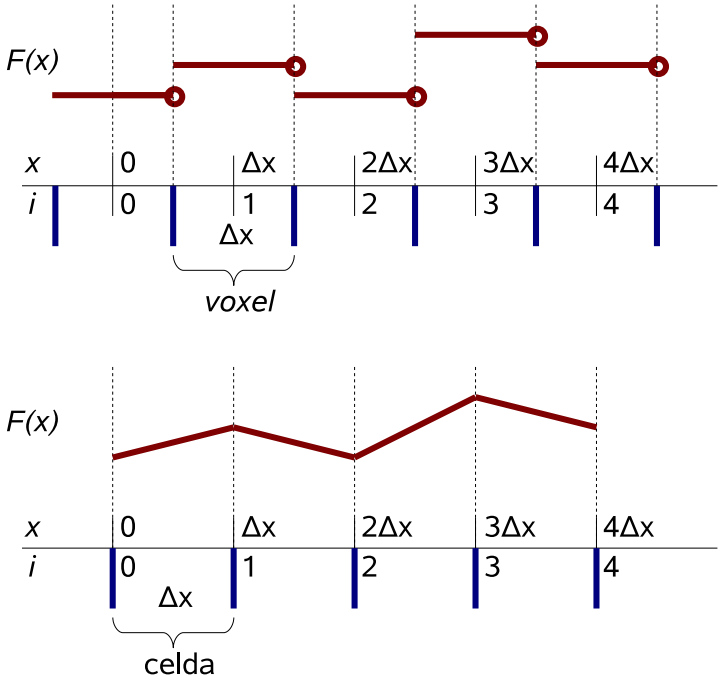

Figura 2.4: Ejemplo unidimensional del cálculo del valor de propiedad asociado a un punto cualquiera de una representación discreta regular mediante: el enfoque de *voxels* (imagen superior) y el enfoque de celdas (imagen inferior).

El cálculo del valor de propiedad de un punto $\boldsymbol{P} = (x, y, z)$ en el enfoque de celdas se

realiza de forma similar, aunque ahora lo que devuelve la función es la esquina inferior, izquierda y posterior de la celda que contiene al punto[2] .

$$i \leftarrow \lfloor \frac{x}{\Delta_x} \rfloor$$
$$j \leftarrow \lfloor \frac{y}{\Delta_y} \rfloor$$
$$k \leftarrow \lfloor \frac{z}{\Delta_z} \rfloor$$

De esta forma, podemos acceder al *array* 3-D para obtener todos los valores de propiedad necesarios para calcular el valor $v_{\boldsymbol{P}}$ del punto $P = (x, y, z)$ mediante interpolación trilineal. Los valores están determinados por las siguientes posiciones del *array* 3-D: $[i, j, k], [i+1, j, k], [i, j+1, k], [i+1, j+1, k], [i, j, k+1], [i+1, j, k+1], [i, j+1, k+1], [i+1, j+1, k+1]$. En la parte inferior de la figura 2.4 se muestra gráficamente la idea en el caso 1-D. Utilizando este enfoque, la función $F(x)$ asociada a la representación es una función por partes con continuidad de orden 0, C^0, entre celdas vecinas.

Independientemente de la función de interpolación escogida, el problema que subyace en este tipo de representaciones es el enorme tamaño que normalmente presentan los conjuntos de datos volumétricos. El tamaño del conjunto de datos volumétrico depende del tamaño del objeto del que provienen y de la resolución utilizada para obtener las muestras (valores de propiedad), así como del espacio necesario para almacenar cada uno de los valores de propiedad que han sido muestreados. Así mismo, suele ser necesario almacenar información extra como puede ser el vector gradiente de valor de propiedad para algunos tipos de propiedad muestreados. De hecho, este problema se ha ido incrementando conforme la precisión de los aparatos de captura ha ido aumentando. Las representaciones discretas de resolución variable tratan de paliar este problema.

2.2.3. Representaciones discretas de resolución variable

En este tipo de representaciones, el conjunto S está distribuido de forma irregular. Por consiguiente, los subvolúmenes que particionan el espacio asociado al conjunto S presentan un tamaño variable, a diferencia de las representaciones de resolución fija en el que solamente presentaban un único tamaño uniforme. A continuación se estudian las ventajas e inconvenientes de los dos enfoques presentados anteriormente, *voxels* y celdas, con respecto al ahorro de espacio en memoria que conlleva el evitar almacenar regiones sobremuestreadas, y con respecto al cálculo del valor de propiedad en cualquier punto P incluido en la representación. Al igual que en el caso de las representaciones discretas de resolución fija, supondremos que el conjunto S sobre el que se definen las dos funciones de interpolación es isotrópico. Por tanto, los subvolúmenes de resolución variable tendrán forma cúbica[3] .

Si se dispone de la función $f(x)$ que define el objeto matemático, existe la posibilidad de realizar un muestreo adaptativo, basándonos en algún criterio que permita determinar donde es necesario un menor intervalo de muestreo y donde es suficiente con muestrear a un intervalo mayor. En el caso de disponer únicamente del conjunto de datos muestreado a una determinada resolución fija, se pueden aplicar otro tipo de técnicas de reducción de la cantidad de información en zonas en donde no sea necesaria tal cantidad de resolución.

Para nuestro estudio supondremos que no se conoce la función $f(x)$ de distribución de valor de propiedad del objeto volumétrico. Por tanto, es necesario realizar un proceso

[2] La asociación de esta esquina a la identificación de la celda es arbitraria.

[3] Los subvolúmenes tendrán forma de paralelepípedo en el caso de que el conjunto S sea anisotrópico

que determine que regiones del volumen están sobremuestreadas y, para cada región de este tipo, asociar un subvolumen de mayor tamaño que represente los subvolúmenes de menor resolución asociados a esta región con valor de propiedad similar.

Debido a la diferencia de tamaño entre los subvolúmenes, es necesario utilizar algún tipo de estructura que permita almacenar la información de posición y dimensión de cada uno de estos. Normalmente se opta por algún tipo de estructura jerárquica de particionamiento espacial. Se puede utilizar un *octree* como estructura soporte para la representación y, al mismo tiempo, como índice espacial de acceso a los subvolúmenes. Su utilización presenta ventajas fundamentales en dos aspectos esenciales de las representaciones de resolución variable:

- Permite reducir el espacio de almacenamiento, ya que la estructura de datos que permita reprentar un conjunto de subvolúmenes de tamaño variable requiere mantener la localización espacial y dimensiones de cada uno. El *octree* permite almacenar ambos datos, representándolos de forma *implícita* mediante la posición en el *octree* de cada nodo hoja, que representa la localización geométrica del subvolumen y, por el nivel en el que se encuentra dicho nodo hoja (dimensión).

- Permite acelerar el cálculo del valor de propiedad en un punto del volumen, ya que la estructura de datos es un índice espacial a cada uno de los elementos de interpolación de tamaño variable y, por tanto, permite localizar eficientemente el elemento de interpolación requerido para el cálculo de dicho valor.

Sin embargo, la utilización del *octree* presenta restricciones a la hora de poder representar los subvolúmenes de tamaño variable. Estas restricciones vienen impuestas por la forma de particionamiento espacial que define la estructura de datos:

- El *octree* solo permite representar subvolúmenes cuyas dimensiones respondan a la siguiente ecuación:

$$dim(x, y, z) = (2^n * \Delta_x, 2^n * \Delta_y, 2^n * \Delta_z), \quad n = 0, 1, \ldots$$

  donde $\Delta_x, \Delta_y, \Delta_z$ representan la distancia, en cada dimensión, entre las posiciones de los elementos de S de máxima resolución, y n representa el número de subvolúmenes de máxima resolución por dimensión de cada subvolúmen representado.

- Por consiguiente, pueden detectarse regiones en el conjunto de datos que, aún presentando un valor de propiedad similar, no puedan ser representadas como un único subvolúmen de menor resolución, teniendo que ser representadas como varios subvolúmenes distintos en el *octree*. Este hecho se debe a la restricción de subdivisión binaria del espacio subyacente en el diseño de la estructura. En la figura 2.5 puede observarse un ejemplo 2-D en el que aparecen dos únicos valores de propiedad distintos para los elementos del conjunto S (en azul y rojo). En la imagen de la derecha se muestra la configuración de subvolúmenes que viene forzada por la utilización de una representación mediante un *quadtree* en este caso bidimensional.

Figura 2.5: Restricciones que impone el *octree* en relación a la forma de las regiones homogéneas.

A continuación se van a describir las ventajas e inconvenientes que presenta cada enfoque (*voxels* y celdas) con respecto al ahorro de espacio en memoria. El estudio se lleva a cabo sobre la posibilidad de ahorro de muestras con respecto a la agregación de *voxels* (o celdas) vecinos que presenten un valor de propiedad similar, junto con las restricciones que vienen impuestas por el *octree*. Como se ha definido anteriormente, los subvolúmenes asociados a un conjunto isotrópico tienen forma cúbica y los asociados a un conjunto anisotrópico forma de paralelepípedo. Sin embargo, aunque su forma geométrica varía, el número de muestras del conjunto S asociadas a cada subvolúmen solo depende del enfoque utilizado. Recordemos que en el enfoque de *voxels* el número de muestras asociadas a un subvolumen es una, siendo este número igual a ocho en el enfoque de celdas, independientemente del tamaño del subvolumen. Por tanto, el estudio sobre el número de muestras que se ahorran al agregar subvolúmenes vecinos de mayor resolución en uno de menor resolución es independiente del tipo de geometría impuesta por la distancia entre muestras.

Con el objetivo de ahorrar espacio de almacenamiento, cuando se detecta que una región compuesta por ocho subvolúmenes vecinos presenta un valor de propiedad similar, se puede proceder a eliminar muestras. El número de muestras que se puede evitar almacenar en el nuevo subvolumen viene expresado por las siguientes ecuaciones, dependiendo del enfoque utilizado:

$$\text{Enfoque de } \textit{voxels} \quad : \quad n_S = n^{dim} - 1 \tag{2.2}$$

$$\text{Enfoque de celdas} \quad : \quad n_S = (n-1)^{dim} \tag{2.3}$$

siendo n el número de muestras por dimensión del nuevo subvolumen y dim la dimensión del espacio en el que estamos trabajando. En nuestro caso $dim = 3$. n_S indica el número de muestras que son eliminadas, permitiendo ahorrar el espacio de almacenamiento que ocupan.

La figura 2.6 muestra gráficamente este hecho para el caso 2-D. Puede observarse que el número de muestras que pueden ahorrarse siguiendo el enfoque de *voxels* es **tres**, ya que el espacio que ocupa una de las muestras es utilizado para almacenar el valor de propiedad asociado al nuevo subvolumen. Sin embargo, en el enfoque de celdas solamente se puede ahorrar el espacio de **un** único valor de propiedad. Es necesario mantener el espacio de almacenamiento ocupado por el resto de muestras, ya que son necesarias para permitir la posibilidad de calcular el valor de propiedad de un punto $\boldsymbol{P}$ incluido en alguna de las celdas vecinas de mayor resolución.

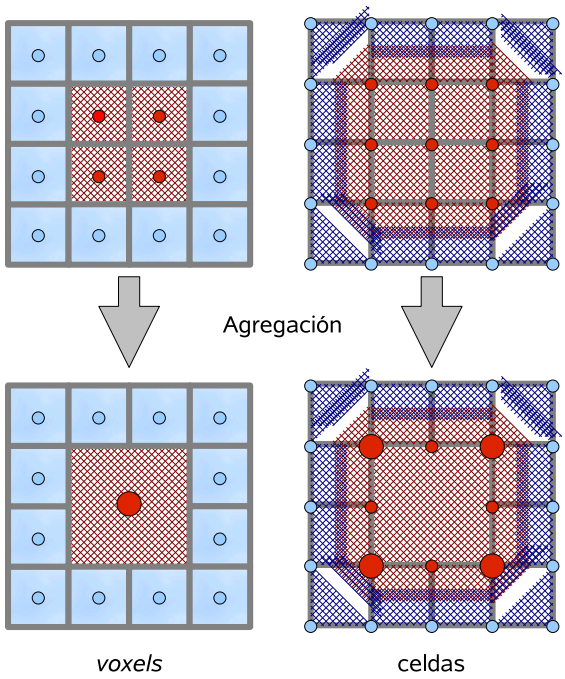

Figura 2.6: Ahorro de muestras en cada uno de los dos enfoques estudiados mediante la agregación de subvolúmenes de mayor resolución en uno de resolución menor.

La explicación recae en el hecho de que, en el enfoque de *voxels*, el subvolúmen contiene la muestra en su interior, mientras que en el de celdas, las muestras son compartidas entre subvolúmenes adyacentes. Por tanto, el número de muestras que permite ahorrar la agregación de subvolúmenes en el enfoque de *voxels* es independiente del tamaño de los *voxels* adyacentes, viniendo determinado por la ecuación 2.2. Sin embargo, el ahorro del espacio ocupado por las muestras en el enfoque de celdas, expresado mediante la ecuación 2.3, depende del tamaño de las celdas adyacentes. De hecho, esta ecuación es una estimación pesimista, ya que, todas las celdas adyacentes tienen un tamaño menor. Siempre que el tamaño de las celdas adyacentes sea menor que el del nuevo subvolumen se deberán mantener todas las muestras. El mayor ahorro que se puede conseguir en el enfoque de celdas se produce cuando las seis posibles celdas vecinas de cara a la nueva celda resultado de la agregación, junto con las doce posibles celdas vecinas de arista, presenten un tamaño mayor o igual que el de la nueva celda. Las vecinas de vértice no influyen, ya que nunca se puede ahorrar el almacenamiento del vértice compartido, independientemente del tamaño de éstas. En el caso más favorable, el número de muestras que se ahorran viene dado por la siguiente ecuación:

$$\text{Enfoque de celdas} \quad : \quad n_S = (n+1)^{dim} - 2^{dim} \tag{2.4}$$

donde n representa el número de celdas de máxima resolución agregadas por dimensión, y dim la dimensión del espacio en el que estamos trabajando. En nuestro caso $dim = 3$. n_S indica el número de muestras que son eliminadas, permitiendo ahorrar el espacio de almacenamiento que ocupan.

Por consiguiente, en el enfoque de celdas, solamente disponemos de una cota superior e inferior del número de muestras que se puede evitar almacenar. Una vez estudiada la

posibilidad de ahorro de muestras que se puede llevar a cabo en cada enfoque, se va a estudiar qué ocurre cuando se utiliza el *octree* para almacenar las muestras en función del enfoque utilizado. En ambos enfoques, cada nodo hoja se va a corresponder con un subvolumen de una resolución determinada.

En el enfoque de *voxels*, cada nodo hoja del *octree* representa un *voxel*, por lo que sólo almacenará un valor de propiedad[4] . De esta forma, se sigue manteniendo la posibilidad de calcular el valor de propiedad de cualquier punto $\boldsymbol{P}$ incluido en el volumen con sólo acceder al nodo hoja que representa la tesela en la que se encuentra incluido dicho punto.

Sin embargo, en el enfoque de celdas es necesario almacenar en cada nodo hoja los ocho valores de propiedad incluidos en cada celda, para, de esta forma, seguir manteniendo la posibilidad de calcular, mediante interpolación trilineal, el valor de propiedad de cualquier punto $\boldsymbol{P}$ incluido en el volumen. Así, al acceder a la tesela, se dispone de los valores necesarios para calcular dicho valor. Esta necesidad de redundancia en el almacenamiento penaliza adicionalmente, de cara al ahorro de espacio, esta modalidad de representación discreta.

En la figura 2.7 se puede observar gráficamente la diferencia conceptual entre el enfoque de celdas y el de *voxels*, junto con las necesidades de almacenamiento de ambos enfoques, utilizando como estructura de datos de particionamiento espacial un *octree*. Se puede observar que en el enfoque de celdas existen valores de propiedad (o referencias a éstos) que deben almacenarse más de una vez en los nodos hoja del árbol, como por ejemplo los que aparecen rodeados por un círculo rojo en la figura. Esta redundancia de almacenamiento provoca un desperdicio de espacio en memoria.

A continuación se estudiarán las ventajas e inconvenientes a la hora de utilizar cada enfoque para calcular el valor de propiedad de un punto $\boldsymbol{P}$ incluido en el volumen. No se va a tener en cuenta el tiempo empleado en el acceso a la tesela que incluye al punto, ya que es similar en ambos y depende de la profundidad a la que se encuentre (en el árbol) el nodo hoja que la representa.

2.2.4. Cálculo del valor de propiedad en un punto sobre representaciones discretas de resolución variable

Si se considera el enfoque de *voxels*, la representación construida sobre un *octree* permite localizar de forma eficiente la tesela que contiene al punto $\boldsymbol{P}$. El valor de propiedad del punto será el valor de la muestra almacenada en el nodo que se corresponde con dicha tesela. De forma similar, si se considera el enfoque de celdas, el *octree* sigue permitiendo localizar eficientemente la tesela que contiene al punto $\boldsymbol{P}$. En este caso, el valor de propiedad del punto se calcula mediante interpolación trilineal sobre las ocho muestras que almacena el nodo correspondiente.

El inconveniente que presenta el enfoque de *voxels* es el mismo que se produce en el caso regular. La función $F(x)$ asociada a al representación es una función por partes que presenta discontinuidades de salto entre *voxels* vecinos. Sin embargo, la función $F(x)$ asociada al enfoque de celdas es una función por partes con continuidad de orden 0, C^0, entre celdas vecinas. Cuando se produce la agregación de *voxels* vecinos de la misma resolución, la función continúa presentando discontinuidades, mientras que la agregación de celdas vecinas mantiene la continuidad en el interior de la celda. Sin embargo, en la frontera entre celdas de distinto tamaño surge un problema de

[4] La información de posición y tamaño del elemento se encuentra definida implícitamente en el *octree* por ser una estructura espacial jerárquica.

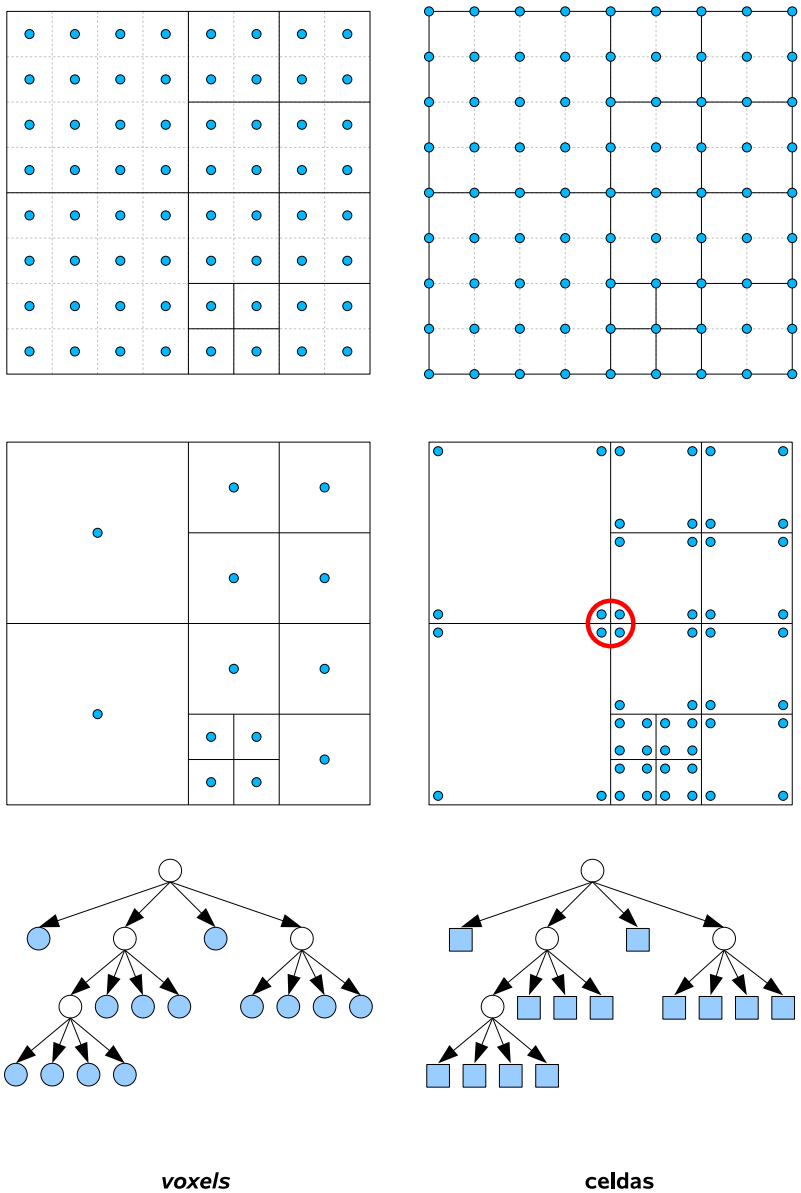

Figura 2.7: Representación mediante un *octree* de subvolúmenes de tamaño variable en el enfoque de *voxels* y en el de celdas.

ambigüedad, ya que se puede realizar la interpolación utilizando las muestras de la celda de mayor tamaño o las de las celdas de menor tamaño para cualquier punto $\boldsymbol{P}$ que esté situado sobre las caras o las aristas. La figura 2.8 muestra un ejemplo 2-D de esta situación. La celda de mayor tamaño presenta el mismo valor de propiedad en los extremos de la arista compartida con la celda de menor tamaño. Sin embargo, la segunda presenta dos valores de propiedad diferentes en sus extremos. Dependiendo de la celda escogida para realizar la interpolación lineal sobre la arista, se obtendrá un valor de propiedad distinto para el punto $\boldsymbol{P}$.

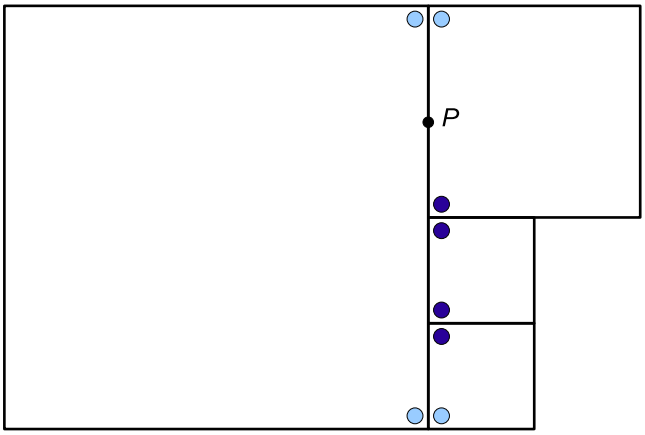

Figura 2.8: Ejemplo 2-D en el que se muestra que dependiendo de los valores de muestra escogidos entre celdas de distinto tamaño se obtiene un valor de propiedad u otro para el punto $\boldsymbol{P}$ que cae en la arista frontera entre ambas.

2.3. Trabajos previos relacionados con el ahorro de espacio

Se pueden considerar dos estrategias diferentes a la hora de reducir el espacio ocupado por los *datasets*: la compresión de volumen y la simplificación de volumen. Aunque las técnicas de simplificación pueden considerarse como una estrategia de compresión con pérdida, acotada por un error, se van a tratar por serparado.

Los algoritmos de compresión para datos volumétricos pueden clasificarse atendiendo a varios aspectos. La compresión puede llevarse a cabo con algoritmos con pérdida o sin pérdida de información; puede basarse en técnicas jerárquicas o adecuarse para transmisión o generación progresiva de imágenes aunque haya solamente una parte disponible de los datos. Así mismo, algunas técnicas permiten la generación de imágenes directamente a partir de los datos comprimidos o requieren un paso previo de descompresión. Fowler y Yagel [FY94] utilizan una técnica de compresión sin pérdida que usa predicción basada en los valores de propiedad de los vecinos de un *voxel*. Lipper y otros [LGK97] presentaron una técnica de compresión de volumen basada en descomposición mediante Wavelets, que permite la generación y transmisión progresiva de imágenes. Se puede consultar el trabajo de Komma y otros [KFDB07] para ver una comparativa entre algunos de los esquemas de compresión de volúmenes sin pérdida. Ning y Hesselink [NH93] presentaron un método de compresión con pérdida basado en cuantización vectorial. Su técnica agrupa *voxels* vecinos en un "*brick*" y los atributos de los *voxels* incluidos forman un vector que se cuantiza haciendolo corresponder con el elemento representativo más cercano de un conjunto fijo. Chiueh y otros [CYH+97] presentaron un enfoque similar utilizando la transformadada discreta de Fourier para comprimir los "*bricks*". Este método permite generar imágenes sin una transformación previa al dominio espacial.

De forma similar a los métodos de compresión con pérdida, los trabajos basados en la simplificación de volumen tratan de aproximar el *dataset* original usando un nuevo *dataset* de volumen con menos *voxels*. Normalmente la reducción se consigue utilizando algún tipo de modelo que caracterice el error cometido en cada paso de simplificación [ZCK97, CMPS97, BS98, CGMS00, HK03, CFM+04].

Nuestra estrategia se basa en reconocer, en el conjunto de datos regular, regiones con valores de propiedad similar y construir una representación discreta adaptativa que

ahorre espacio de almacenamiento. Al mismo tiempo, deseamos que la representación mantenga una distribución de valores de propiedad similar a la que se obtendría en una representación regular construida a partir del mismo conjunto de datos. Es decir, queremos que para cualquier punto perteneciente al volumen, la diferencia entre el valor de propiedad obtenido a partir de la representación regular y el obtenido a partir de la representación adaptativa se controle mediante un modelo simple de error proporcionado por el usuario. En la siguiente sección describimos nuestra propuesta de solución.

2.4. Nuestra propuesta de representación

En secciones anteriores se ha mostrado que utilizar un *octree* para representar subvolúmenes de tamaño variable utilizando el enfoque de *voxels* permite la posibilidad de ahorrar más espacio de almacenamiento en memoria que el enfoque de celdas. Sin embargo, la principal desventaja que encontramos en este enfoque viene derivada de su forma de calcular los valores de propiedad asociados a los puntos incluidos en el volumen. Debido a la estrategia de la interpolación de orden cero, los valores de propiedad en el espacio del objeto presentan un aspecto discontinuo, estableciéndose saltos bruscos entre los diferentes subvolúmenes que lo forman. Este problema se reduce en el caso del enfoque de celdas debido a que el valor de propiedad de los puntos del objeto se calcula mediante interpolación trilineal. Considerando las ventajas y desventajas que presentan ambos enfoques, con respecto al ahorro de espacio de almacenamiento y a la continuidad de la función que calcula el valor de propiedad para cualquier punto del volumen, nuestra representación discreta multiresolución consigue establecer un compromiso entre ambos.

El objetivo que se persigue es partir de una representación discreta regular y conseguir como resultado una representación irregular de los mismos datos, con una pérdida de información acotada mediante una tolerancia de error. La representación irregular resultante no almacenará información redundante en regiones del objeto en donde no sea necesario. Los *voxels* de máxima resolución incluidos en tales regiones son sustituidos por *voxels* de menor resolución. Además, se requiere que la representación resultante presente una distribución de los valores de propiedad lo más parecida posible a la representación discreta regular original.

Se parte del supuesto de que el conjunto de datos volumétricos, a partir del cual se construye nuestra representación, sigue una distribución uniforme y regular. Hay que tener en cuenta que solamente se dispone de este conjunto inicial de datos, por lo que no se pueden obtener nuevos conjuntos de datos a distintas resoluciones. Es decir, no se dispone de la función $f(x)$ que permitiría generar muestras en cualquier posición del espacio. La idea parte del hecho de que existen regiones en el espacio de muestras que incluyen subconjuntos de muestras cuyo valor de propiedad asociado es similar. Si se construye una representación regular a partir del conjunto de datos se almacenarán múltiples instancias de un valor de propiedad similar en determinadas regiones del volumen. Este fenómeno se conoce con el nombre de *sobremuestreo*.

Nuestra representación se basa en un *octree* de *voxels* de tamaño variable. La idea clave es utilizar una medida que nos permita identificar regiones sobremuestreadas del volumen y permitir agregar los *voxels* que las componen. Las regiones estarán representadas mediante un conjunto de *voxels* vecinos. Una vez determinado que todos los *voxels* incluidos en una determinada región cumplen un *criterio de similaridad*, se utiliza una operación que agrega los *voxels* con valores de propiedad similar en un nuevo

voxel de menor resolución. El nuevo *voxel* tiene asociado un nuevo valor de propiedad y se ahorrará el espacio ocupado por siete de los ocho *voxels* de mayor resolución que han sido agregados. El valor de propiedad del nuevo *voxel* es función de los valores de los *voxels* de mayor resolución. Si se hubiese utilizado una representación regular, habría sido necesario almacenar para la región representada por el nuevo *voxel* un número de muestras innecesario, provocando unos requerimientos extra de almacenamiento. Utilizando una representación irregular que tenga en cuenta este hecho se consigue que la representación se adapte a estas "regiones homogéneas" con valor de propiedad similar.

A continuación, se va a formalizar el concepto de región homogénea y se va a determinar la forma que es necesario que tenga para poder construir nuestra representación discreta basada en un *octree* de *voxels* de tamaño variable. Seguidamente se presenta nuestra representación, comparándola con las representaciones irregulares basadas en *octrees* de *voxels* y de celdas.

2.4.1. Regiones homogéneas

Como se ha visto con anterioridad, un requisito fundamental de las representaciones volumétricas discretas es poder calcular el valor de propiedad en cualquier punto $\boldsymbol{P}$ del volumen. La elección de un *criterio de similaridad*, que permita reemplazar muestras de mayor resolución situadas en una región del espacio por una muestra de menor resolución, determina la variación en el resultado de calcular el valor de propiedad en un punto del volumen representado de forma irregular, con respecto al cálculo del valor de propiedad del mismo punto a partir de las muestras de mayor resolución. Un criterio de similaridad que solamente permita la sustitución de muestras con igual valor de propiedad, proporciona como resultado una representación irregular; en la cual, la variación del valor de propiedad en un punto con respecto a la representación regular es cero. Conforme el criterio permite aumentar la diferencia permitida entre los valores de propiedad de las muestras que son reemplazadas (error cometido en la simplificación), la variación del valor de propiedad en un punto, con respecto a la representación regular original, aumenta.

Como consecuencia, a la hora de aplicar la idea de región que engloba muestras con valores de propiedad similares se han de considerar dos aspectos fundamentales: la *forma de la región* que determina el número de subvolúmenes de menor tamaño de la representación que serán sustituidos por un subvolumen de mayor tamaño, una vez comprobada su homogeneidad y, el *criterio de similaridad* que determina cuando será posible agregar subvolúmenes de menor tamaño en uno de tamaño mayor.

A continuación se va a definir el *concepto de región homogénea* en un conjunto de datos muestreados en el espacio tridimensional. Para definir el concepto vamos a restringir el dominio de los valores de propiedad al caso de conjuntos de elementos escalares. Consideraremos para su definición dos tipos principales de dominios: Un dominio *discreto* con un rango de valores determinado y, un dominio de valores *binario* cuyos dos valores asociados indican la presencia del objeto y la ausencia de éste, respectivamente. En otras palabras, los dos valores del dominio binario representan el interior y exterior del objeto. Este dominio permite definir de forma discreta la *función característica* del objeto en el espacio tridimensional.

De hecho, el dominio binario proporciona una forma de representar sólidos, en el sentido comentado en el capítulo 1, mediante un modelo de volumen. Hemos utilizado este dominio para poder disponer de una cota superior a la hora de comprobar la

máxima cantidad de espacio que puede llegar a ahorrarse con nuestro modelo. De hecho, nos permite definir el criterio de similaridad óptimo (la igualdad) con respecto a la pérdida de información, independientemente de la forma del objeto volumétrico. Obviamente, la forma del objeto determinará también la tasa de reducción de espacio obtenido finalmente.

Si el conjunto de datos regular está definido en el dominio binario sólo tiene sentido establecer un criterio de similaridad basado en la igualdad. Sin embargo, si el conjunto de datos está definido en un dominio discreto, tiene sentido establecer criterios de similaridad que pueden ir desde la igualdad estricta de valores hasta una tolerancia en la diferencia entre los valores de propiedad de las muestras que van a ser sustituidas. Hablaremos de *región homogénea* cuando el criterio de similaridad utilizado para permitir la sustitución de muestras sea la igualdad, independientemente de que el dominio de valores de propiedad sea binario o discreto. Cuando el criterio de similaridad aplicado incluya una tolerancia de error en la diferencia entre los valores de las muestras a sustituir, hablaremos de *región cuasi-homogénea*. En resumen, vamos a considerar las siguiente terminología:

- Si se tiene el mismo valor de propiedad para todas las muestras incluidas dentro de una región del volumen podemos considerar que dicha región es *homogénea* con respecto a dicho valor de propiedad.

- Si los distintos valores de propiedad de todas las muestras incluidas dentro de una región del volumen pueden representarse mediante un único valor de propiedad, salvo un error determinado o algún valor que determine la pérdida de información que se produce al realizar la representación mediante dicho valor único, podemos considerar que dicha región es *cuasi-homogénea*.

Un ejemplo 2-D de región cuasi-homogénea sería la región topológicamente equivalente a un disco sobre el plano que engloba un conjunto de muestras cuya diferencia de valor entre todas ellas es menor que un error, ϵ. Si nos basamos en métricas de error en el espacio de valores de muestras podemos hablar de región ϵ-homogénea.

Dependiendo del número de regiones homogéneas y/o cuasi-homogéneas[5] presentes en el conjunto de datos volumétrico se puede reducir el espacio que ocupa una representación regular, *sustituyéndola* por una representación irregular.

A continuación, se van a definir formalmente los conceptos de región homogénea y cuasi-homogénea, los cuales vendrán determinados por el criterio de similaridad aplicado. Una vez establecida la definición, concretaremos la forma que debe presentar la región homogénea (o cuasi-homogénea) para que sea útil de cara a poder ser representada mediante un *octree*. En ese momento nos centraremos en el enfoque usado por nuestra representación, ya que los *voxels* son las estructuras geométricas que dotarán de forma a la nueva región (*voxel* de mayor tamaño). Para definir los conceptos nos basaremos en una estructura de muestras en forma de rejilla regular 3-D.

Definición 4 (Región homogénea) *Dado un conjunto finito S de pares de valores $(\boldsymbol{P_i}, v_i)$, donde v_i representa el valor de alguna propiedad de tipo escalar del objeto, situada en la posición $\boldsymbol{P_i}$ del espacio 3-D, se define una* región homogénea de S *como un subconjunto $U \subseteq S$ cuyos elementos cumplen las siguientes condiciones:*

[5] A partir de este momento utilizaremos el término región homogénea para referirnos a ambos tipos de regiones. Solamente haremos referencia explícita a uno de los términos cuando el contexto lo requiera.

1. *Los elementos de U forman un conjunto conexo sobre la estructura en rejilla 3-D de* S^6 .

2. *El criterio de similaridad es la igualdad. Es decir, el valor de propiedad* v_i *de cada elemento del conjunto U, coincide.*

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

Definición 5 (Región cuasi-homogénea) *Dado un conjunto finito S de pares de valores* $(\boldsymbol{P_i}, v_i)$*, donde* v_i *representa el valor de alguna propiedad de tipo escalar del objeto, situada en la posición* $\boldsymbol{P_i}$ *del espacio 3-D, se define una* región cuasi-homogénea *de S como un subconjunto* $U \subseteq S$ *cuyos elementos cumplen las siguientes condiciones:*

1. *Los elementos de U forman un conjunto conexo sobre la estructura en rejilla 3-D de S.*

2. *El criterio de similaridad es un entorno de error en el dominio de valores de propiedad, por lo que la diferencia entre cualesquiera dos valores de propiedad de los elementos del conjunto nunca supera un valor umbral* ϵ*.*

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

Se va a utilizar una relación de adyacencia suficiente para nuestro propósitos, la cual permite definir la conectividad, y como consecuencia un conjunto conexo de elementos de S, en la estructura de rejilla regular que presentan los datos volumétricos. Antes se presenta una definición del concepto de vecindad en rejillas sobre la que se basa la definición de adyacencia. Se va a suponer, por claridad en la definición, que la vecindad esta definida sobre una rejilla regular isotrópica cuya distancia entre muestras es 1.

Definición 6 (N_6-Vecindad) *La vecindad de un punto* $\boldsymbol{P_i} = (x_i, y_i, z_i)$ *correspondiente a la posición de un elemento* $(\boldsymbol{P_i}, v_i) \in S$ *es el conjunto de puntos* $N_6(\boldsymbol{P_i})$ *tal que:*

$$\begin{aligned} N_6(\boldsymbol{P_i}) \quad = \quad & \{(x_i, y_i, z_i), (x_i + 1, y_i, z_i), (x_i - 1, y_i, z_i), (x_i, y_i + 1, z_i), \\ & , (x_i, y_i - 1, z_i), (x_i, y_i, z_i + 1), (x_i, y_i, z_i - 1)\} \end{aligned}$$

Definición 7 (6-adyacencia) *Dos puntos,* $\boldsymbol{P_i}$ *y* $\boldsymbol{P_j}$*, de la rejilla son* 6*-adyacentes si y solo si* $\boldsymbol{P_i} \neq \boldsymbol{P_j}$ *y* $\boldsymbol{P_i} \in N_6(\boldsymbol{P_j})$*.*

Definición 8 (Conectividad) *Un subconjunto* $U \subset S$ *es conexo si, para todo par de elementos* $u_a = (\boldsymbol{P_a}, v_a), u_b = (\boldsymbol{P_b}, v_b) \in U$ *cuyas posiciones vienen definidas por* $\boldsymbol{P_a} = (x_a, y_a, z_a)$ *y* $\boldsymbol{P_b} = (x_b, y_b, z_b)$ *respectivamente, existe una secuencia* $\boldsymbol{P_0}, \cdots, \boldsymbol{P_n}$ *de posiciones de elementos de U tal que* $\boldsymbol{P_0} = \boldsymbol{P_a}$*,* $\boldsymbol{P_n} = \boldsymbol{P_b}$*, y* $\boldsymbol{P_i}$ *es adyacente a* $\boldsymbol{P_{i-1}} (1 \leq i \leq n)$*. Tal secuencia se denomina* camino *y se dice que* conecta *a* $\boldsymbol{P_a}$ *y* $\boldsymbol{P_b}$ *en U.*

[6] Suponiendo definida una relación de adyacencia en la rejilla. Las definición de conjunto conexo (componente conexa) y conceptos relacionados pueden consultarse en [KR04].

Como se puede comprobar en la definición de conectividad, la relación de adyacencia puede ser cualquiera que definamos sobre los puntos de la rejilla. Podemos aplicar la 6-*adyacencia*, que puede verse como los vecinos de cara de un cubo centrado en cualquier punto de la rejilla, o incluir vecindades con mayor número de elementos $N_{18}(p)$, o $N_{26}(p)$ que proporcionan a su vez la definición de 18-*adyacencia* y 26-*adyacencia*. Estas últimas pueden verse, respectivamente, como los vecinos de cara más los vecinos de arista; y como los vecinos de cara, más los vecinos de arista, más los vecinos de vértice. La definición de conjunto conexo basada en la 6-adyacencia es suficiente para poder determinar la forma de una región homogénea, ya que, como veremos a continuación, el requisito de ahorro de espacio que perseguimos, junto con la coherencia necesaria a la hora de realizar el cálculo del valor de propiedad en cualquier punto del volumen, van a restringir las posibles formas de dicha región; y las formas que se pueden conseguir mediante dicha relación de adyacencia cumplen de sobra con ambos requisitos. Lo único que debemos tener en cuenta es que la necesaria homogeneidad de la región en el dominio de valores de propiedad debe ser añadida como requisito al de ser un conjunto de muestras 6-conexo, con el objetivo de que este conjunto represente una región homogénea en la rejilla.

En la figura 2.9 se ilustran estos conceptos en un ejemplo 2-D. En la imagen de la izquierda se muestra la relación de N_4-vecindad (que equivaldría a la N_6-vecindad en el caso 3-D) para el punto $\boldsymbol{P}$ situado en una rejilla regular. Cualquiera de los cuatro puntos vecinos, distintos del punto $\boldsymbol{P}$, son 4-adyacentes a éste. La imagen de la derecha muestra un ejemplo de conjunto conexo formado por los vértices de la rejilla coloreados en azul oscuro. Si los valores de propiedad asociados a las muestras situadas en estos vértices cumplen el criterio de similaridad, entonces este conjunto conexo representa una región homogénea en la rejilla.

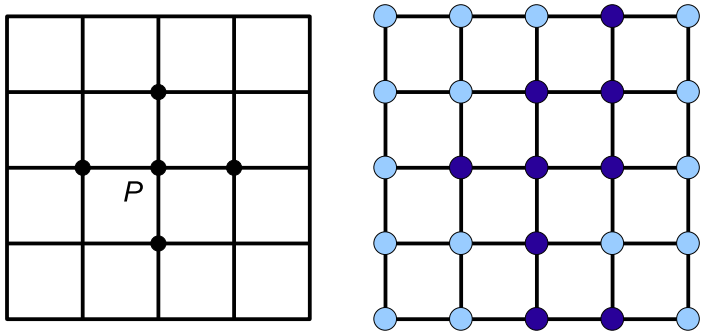

Figura 2.9: La imagen de la izquierda muestra la N_4-vecindad de un punto $\boldsymbol{P}$ situado en una rejilla regular y los cuatro puntos 4-adyacentes a éste. La imagen de la derecha muestra un conjunto conexo (azul oscuro) que utiliza la relación de 4-adyacencia.

La definición de región homogénea permite que ésta adopte una forma genérica. Sin embargo, a la hora de usar este concepto en nuestra representación es necesario concretar cuál es la forma adecuada, descartando las regiones homogéneas que no van a poder ser representadas. Un *octree* que almacene en sus nodos hoja *voxels* de tamaño variable sólo permite representar regiones homogéneas que tengan forma cúbica, caso de que el conjunto de muestras inicial sea isotrópico, o forma de paralelepípedo, en el caso de que dicho conjunto sea anisotrópico. Además de esta requisito, el *octree* impone restricciones adicionales en base a su estrategia de particionamiento espacial, como se ha mostrado en secciones anteriores.

Para conseguir representar de forma correcta regiones homogéneas de tamaño variable en un *octree*, la operación de agregación de *voxels* debe definirse de forma que se garantize su clausura, en el sentido de que constituya un elemento representable. Por tanto, esta operación permite agregar ocho *voxels* vecinos del mismo tamaño en uno del doble de tamaño cuando estos constituyan una región homogénea y cumplan las restricciones del *octree*. Inicialmente solo se pueden agregar *voxels* de máxima resolución, pero conforme aparecen nuevos *voxels* de menor resolución, pueden producirse agregaciones a distintos niveles. La figura 2.10 muestra, en la imagen de la izquierda, un ejemplo 2-D de región homogénea sobre una rejilla regular, en la central, el conjunto de *voxels* que contienen las muestras y, en la imagen de la derecha, la representación mediante un *quadtree*. Como puede comprobarse, las restricciones de la estructura provocan que la región homogénea tenga que ser representada por varios *voxels* de diferente tamaño, a pesar de que el valor de propiedad es el mismo en todo el conjunto conexo de *voxels* de máxima resolución coloreados en azul oscuro.

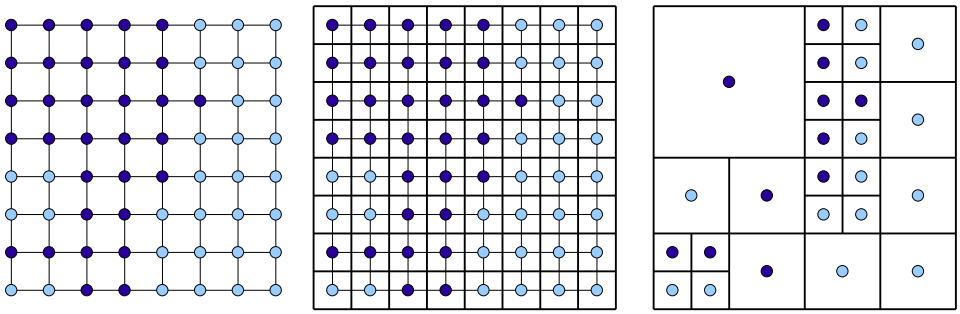

Figura 2.10: La imagen de la izquierda representa una región homogénea de muestras en azul oscuro sobre una rejilla regular. La imagen central representa los *voxels* de máxima resolución asociados a las muestras de la rejilla regular. La imagen de la derecha muestra un *quadtree* cuyos nodos hoja representan "*voxels*" homogéneos de distinto tamaño.

Una vez formalizado el concepto de región homogénea, junto con la operación de agregación de *voxels*, la cual permite representar dichas regiones en un *octree* de *voxels* de tamaño variable, procederemos con la descripción de nuestra representación.

2.4.2. *Octree* de Regiones Homogéneas con Borde: *HRB-Octree*

Como hemos mostrado en la sección 2.2, almacenar subvolúmenes cúbicos en un *octree* siguiendo el enfoque de *voxels* permite ahorrar más espacio que almacenarlos utilizando el enfoque de celdas. Así mismo, la distribución de valores de propiedad que permite el enfoque de *voxels* es discontinua, mientras que la distribución de valores de propiedad en el enfoque de celdas es continua. Nuestra idea es conjugar un *octree* basado en *voxels* y una forma de cálcular el valor de propiedad basada en el enfoque de celdas, pero aplicada a dichos *voxels*, de manera que se consiga continuidad en la distribución de valores.

Para cumplir este objetivo vamos a establecer una partición en el espacio ocupado por cada *voxel*, considerando para ello dos zonas bien diferenciadas. Todos los *voxels* presentan una *zona exterior o borde* con grosor $\frac{\Delta_x}{2}$, $\frac{\Delta_y}{2}$, $\frac{\Delta_z}{2}$, para las dimensiones x, y y z respectivamente. Los *voxels* de máxima resolución están formados solamente por este tipo de zona. Los *voxels* que representan regiones homogéneas generadas a partir de operaciones de agregación están formados por la zona borde y por una *zona interior* que

se corresponde con el resto de su volumen. En la figura 2.11 se ilustra gráficamente esta partición del espacio ocupado por cada uno de los *voxels* que constituyen el volúmen. Las zonas interiores aparecen coloreadas en verde y los bordes en azul celeste. Puede observarse que los *voxels* de máxima resolución situados en la parte inferior derecha carecen de zonas interiores.

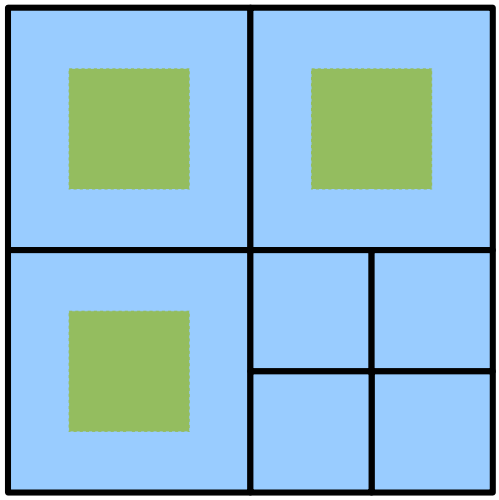

Figura 2.11: Zonas interiores (en verde) y bordes (en azul celeste) en las que se divide el espacio ocupado por cada *voxel* de nuestra representación.

Utilizando esta partición, cuando se realiza el cálculo del valor de propiedad de un punto $\boldsymbol{P}$ del volumen, se determina si está incluido en una zona interior o exterior. Dependiendo del tipo de zona, se actua de la siguiente forma para asignar el valor de propiedad:

- Si el punto $\boldsymbol{P}$ está incluido en una zona interior, su valor de propiedad es el que se encuentra almacenado en el *voxel* correspondiente.
- Si el punto $\boldsymbol{P}$ está incluido en un borde, es necesario determinar que *voxel* de máxima resolución, de entre todos los que han sido agregados para formar el *voxel* de mayor tamaño, contiene a $\boldsymbol{P}$. Obviamente, si el *voxel* de partida ya es de máxima resolución no es necesario realizar este paso previo. A continuación, es necesario encontrar todos los *voxels* de máxima resolución, incluidos en los *voxels* vecinos al *voxel* determinado. Esto es debido a que tales *voxels* van a permitir configurar la topología de las celdas de interpolación que cubren todo el espacio asociado a dicho *voxel*. La figura 2.12 muestra un ejemplo 2-D de este proceso. La imagen de la izquierda muestra el *voxel* en cuyo borde se encuentra incluido el punto $\boldsymbol{P}$ y el *voxel* de máxima resolución (en línea roja discontinua), dentro de éste, que lo contiene. En línea negra discontinua se muestran los *voxels* de máxima resolución vecinos incluidos en *voxels* de mayor tamaño, junto con los *voxels* vecinos de máxima resolución que no han sufrido ninguna operación de agregación (los situados a la derecha). La imagen de la derecha muestra la topología de las celdas de interpolación necesarias para cubrir todo el espacio del *voxel* de máxima resolución (menos el área de éste que se corresponde con zona interior). Las celdas tienen sus vértices situados en los centros de los *voxels* de máxima resolución y sus valores de propiedad son los correspondientes a cada

uno de los *voxels*. Si el *voxel* de máxima resolución está incluido en uno de mayor tamaño, se le asigna el valor de propiedad de éste. En otro caso, se asigna su correspondiente valor de propiedad almacenado.

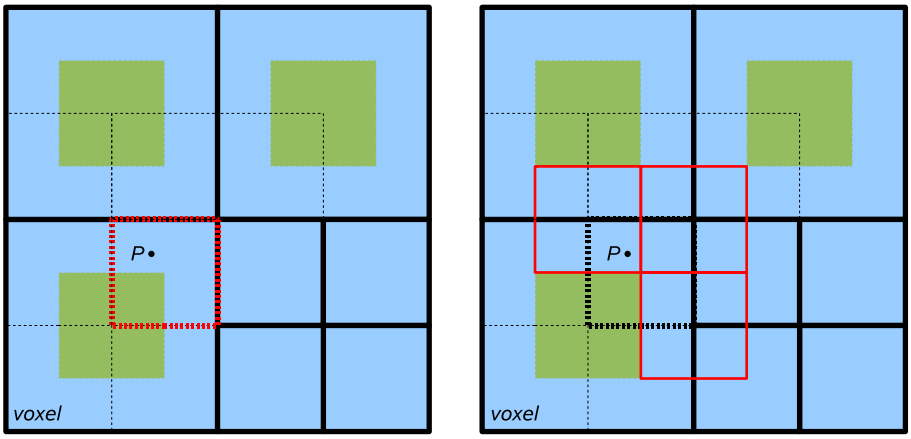

Figura 2.12: Ejemplo 2-D que ilustra el cálculo del valor de propiedad de un punto $\boldsymbol{P}$ usando el *HRB-Octree*. La imagen izquierda muestra el *voxel* en el que se encuentra incluido el punto $\boldsymbol{P}$, junto con el *voxel* de máxima resolución que lo contiene (rojo) y los *voxels* vecinos de máxima resolución (negro). La imagen de la derecha muestra las celdas de interpolación (rojo) que cubren todo el espacio de interés asociado al *voxel* de máxima resolución.

- Por último, cada celda de interpolación permite calcular un valor de propiedad para todos puntos incluidos en ella. Ya solo resta determinar la celda en la que está incluido el punto $\boldsymbol{P}$ y calcular su valor de propiedad mediante interpolación trilineal.

Procediendo de esta forma se consigue una distribución de valores de propiedad que presenta una continuidad de orden cero entre los *voxels* vecinos de diferente tamaño. En la parte central de la figura 2.13 se muestran los dos tipos de zonas, interior y borde, en verde y azul celeste respectivamente, junto con la forma de la función de distribución de valores de propiedad para el caso 1-D. En la imagen situada en la parte superior se muestra la distribución de valores de propiedad para un *voxel* de máxima resolución y otro del doble de tamaño obtenido como resultado de efectuar una operación de agregación de dos *voxels* vecinos de máxima resolución. La distribución de valores de propiedad de los dos *voxels* antes de la operación de agregación se muestra en color gris. En la imagen situada en la parte inferior de la figura se muestra la distribución de valores de propiedad para una celda de máxima resolución y una celda del doble de tamaño, resultado de eliminar la muestra compartida por las dos celdas de máxima resolución agregadas. En la celda resultante aparece en color gris la distribución de valores de propiedad asignados mediante interpolación en cada una de las celdas de menor tamaño.

En la imagen central se puede apreciar la distribución de valores de propiedad que proporciona nuestra representación. Entre *voxels* de máxima resolución se realiza un interpolación lineal considerando que el valor de propiedad asociado se encuentra situado en su centro. En el caso del *voxel* de mayor tamaño, se realiza una interpolación lineal si el punto pertenece al intervalo comprendido entre el centro del *voxel* de máxima resolución y la frontera del *voxel* de mayor tamaño, es decir, si el punto cae en el borde (en

color azul celeste). El otro extremo de la interpolación es el centro del correspondiente *voxel* vecino de máxima resolución. En el caso de que el punto pertenezca a la zona interior se asigna directamente el valor de propiedad almacenado en el *voxel* de mayor tamaño. Esta zona aparece en color verde en la imagen.

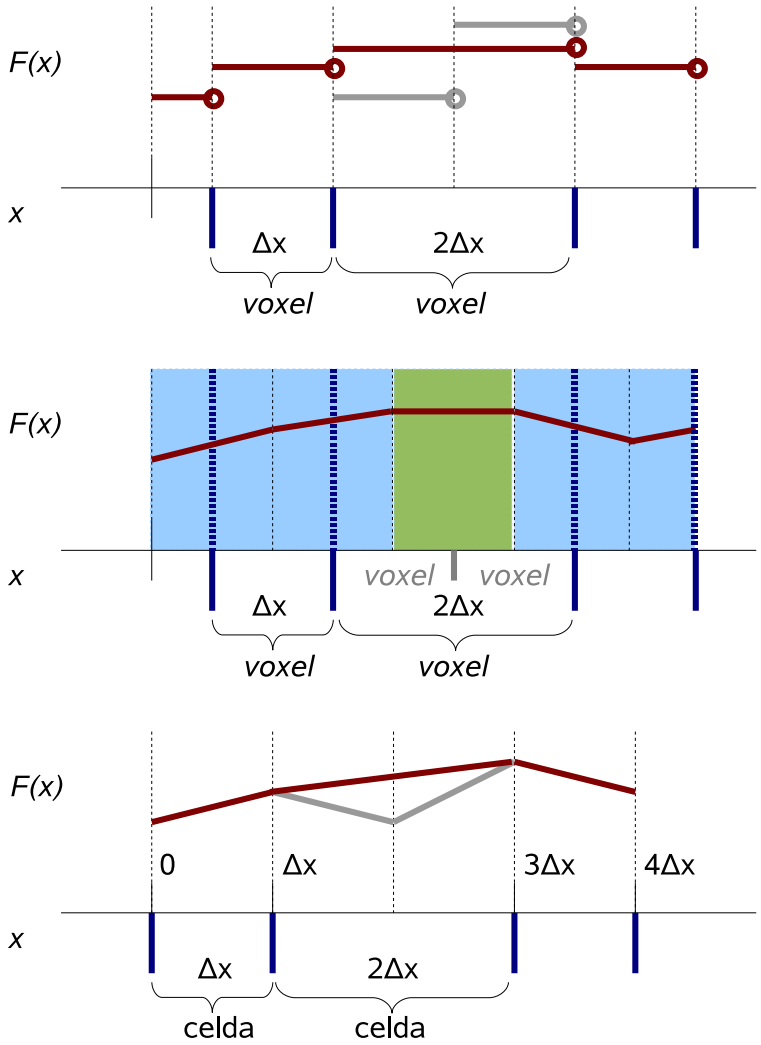

Figura 2.13: Ejemplo unidimensional que ilustra la forma de la función de cálculo del valor de propiedad en un punto cualquiera que usa el *octree* de regiones homogéneas junto con la partición en zonas interior y borde (parte central) frente a las que usan los enfoques de *voxels* (parte superior) y celdas (parte inferior), también definidos sobre regiones homogéneas.

Como se ha mostrado, la partición que divide a los *voxels* en zonas interiores y exteriores utiliza dos tipos de interpolación, vecino más próximo y trilineal, para calcular el valor de propiedad de cualquier punto $\boldsymbol{P}$ incluido en el volumen representado. En la figura 2.14 se puede observar la distribución de valores de propiedad que presenta un *quadtree* basado en el enfoque de celdas (parte superior) y un *quadtree* basado en el enfoque de *voxels* (parte inferior), con el mismo número de subvolúmenes particionando el espacio y el mismo tamaño para cada uno de ellos. Las líneas en rojo claro y rojo oscuro representan la distribución de valores de propiedad que proporciona la interpo-

lación trilineal (bilineal en la figura), en el caso del enfoque de celdas, y la interpolación de orden cero, en el caso del enfoque de *voxels*.

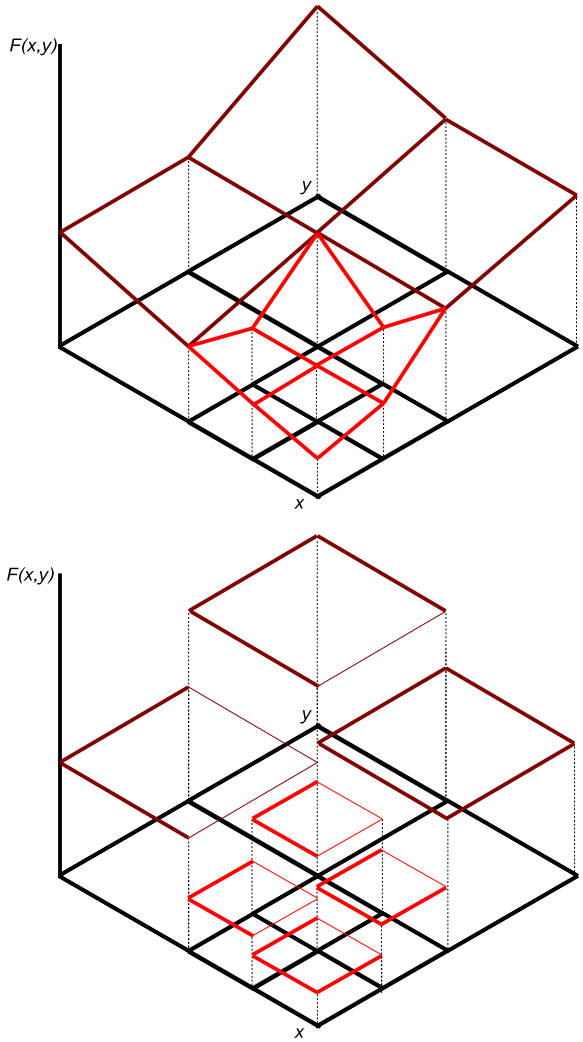

Figura 2.14: Ejemplo 2-D de distribución de valores de propiedad para el enfoque de celdas (parte superior) y el enfoque de *voxels* (parte inferior) considerando el mismo número de subáreas en el *quadtree*.

Se ha distinguido el color (rojo claro y oscuro) en función del tamaño del subvolumen asociado para resaltar el problema de ambigüedad en el cálculo del valor de propiedad en las fronteras entre celdas de diferente tamaño. Este problema se ha comentado en la sección de representaciones discretas irregulares.

Inconsistencia de valores en aristas y caras comunes El cálculo del valor de propiedad de los puntos situados en los elementos geométricos que forman la frontera entre celdas de distinto tamaño puede producir valores distintos dependiendo de la celda elegida para realizar la interpolación. En el caso 2-D, la frontera problemática está formada por las aristas, en el caso 3-D por las aristas y por las caras.

En el *quadtree* de celdas situado en la parte superior de la figura 2.14 se puede observar

el problema en las aristas que forman la frontera entre las celdas de mayor y menor tamaño. El valor de propiedad para cualquier punto de dichas aristas, exceptuando a los vértices, depende de si el cálculo se realiza desde la celda de mayor tamaño o desde la de menor tamaño[7] . Esta situación se muestra mediante líneas de color rojo oscuro en las celdas de mayor tamaño y líneas en rojo claro en las de menor tamaño. Estas líneas delimitan el área de valores de las correspondientes funciones de interpolación obtenidas en cada celda. Por tanto, la función de distribución de valores de propiedad, que era una función continua por partes, con continuidad de orden C^0, en el caso de una rejilla regular, puede presentar discontinuidades en el caso de rejillas de resolución variable.

Como se puede observar en la parte inferior de la figura 2.14, el enfoque de *voxels* se comporta de manera análoga al caso regular, ya que la agregación de *voxels* no influye a la hora de la continuidad. La función de distribución de valores sigue siendo discontinua a saltos.

Sin embargo, con nuestra representación se obtiene una distribución de valores de propiedad que sigue una función continua por partes con continuidad de orden C^0. En la figura 2.15 se puede observar la distribución de valores en color rojo oscuro y claro. La partición de los *voxels* en zonas interiores y bordes aparece en color verde y azul celeste respectivamente. Los valores de propiedad que se corresponden con las zonas interiores aparecen como una superficie plana en color rojo oscuro. Los valores de propiedad correspondientes a las zonas exteriores (bordes) aparecen delimitados por líneas en color rojo oscuro y claro.

La principal ventaja de nuestra representación frente al enfoque de celdas utilizando una representación irregular, en relación al cálculo del valor de propiedad en un punto $\boldsymbol{P}$ es la ausencia del problema de falta de continuidad en elementos geométricos (aristas y caras) compartidos por celdas vecinas de distinto tamaño.

Por otro lado, el principal inconveniente se presenta a la hora de realizar el cálculo del valor de propiedad para un determinado punto del volumen. Para realizar esta operación, el enfoque de celdas permite determinar rápidamente ($O(nlogn)$) la celda en la que se encuentra dicho punto y, al almacenar los ocho valores de propiedad necesarios para realizar la interpolación en el nodo hoja correspondiente, el cálculo del valor de propiedad es inmediato. Sin embargo, nuestro método solo obtiene ventaja sobre el de celdas en el caso de que el punto esté incluido en la zona interior del *voxel*, ya que se asigna directamente el valor de propiedad almacenado, sin necesidad de realizar ningún tipo de interpolación. En el caso de que el punto esté incluido en la zona borde, se emplea mucho más tiempo para obtener su valor de propiedad. Para tener una idea del tiempo empleado se van a enumerar brevemente los pasos requeridos para realizar el cálculo del valor de propiedad de un punto $\boldsymbol{P}$ utilizando nuestra representación de volúmenes.

1. Localizar en el *octree* el *voxel* en el que está incluido el punto $\boldsymbol{P}$.

2. Determinar si el punto está incluido en la zona interior o en el borde del *voxel*. El borde siempre tiene un tamaño igual a la mitad de la distancia entre muestras en cada dimensión. Hay que tener en cuenta que los *voxels* de máxima resolución están constituidos solamente por zona borde.

[7] Obviamente, se está considerando que el punto situado en el vértice de las celdas de menor tamaño, que cae en la arista compartida con la celda de mayor tamaño, no tiene un valor igual al proporcionado por la interpolación lineal de la arista de la celda de mayor tamaño

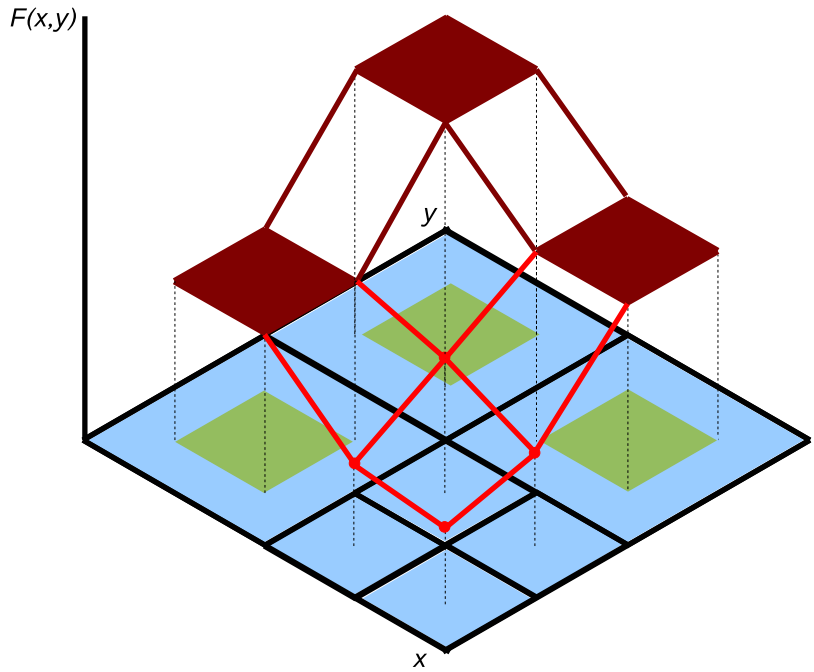

Figura 2.15: Ejemplo 2-D de la distribución de valores de propiedad que proporciona el *HRB-Octree* considerando el mismo número y tamaño de subvolúmenes que los utilizados en el *quadtree* de la figura 2.14.

3. Si el punto está incluido en la zona interior, devolver el valor de propiedad almacenado en el nodo hoja.

4. Si el punto está incluido en el borde:

   a) Determinar el *voxel* de máxima resolución que contiene al punto $\boldsymbol{P}$.

   b) Buscar en el *octree* todos los *voxels* de máxima resolución necesarios para establecer la topología de las celdas de interpolación necesarias.

   c) Determinar la celda de interpolación en la que está incluido el punto.

   d) Calcular el valor de propiedad del punto $\boldsymbol{P}$ mediante interpolación trilineal.

Para obtener una visión más clara de la distribución de valores de propiedad que proporcionan los dos enfoques clásicos (*voxels* y celdas), tanto en una distribución regular como irregular de muestras, junto con la proporcionada por nuestra representación, se presentan a continuación varias figuras que ilustran dicha distribución sobre un ejemplo concreto de valores de propiedad. A partir de un conjunto regular de valores de propiedad se aplica el concepto de región homogénea con un criterio de similaridad $\epsilon = 1,5$, con el objetivo de obtener representaciones irregulares en cada uno de los enfoques enumerados.

En la parte superior de todas las figuras se presenta un ejemplo de rejilla regular y uniforme bidimensional con valores de propiedad escalares, junto con la distribución de valores de propiedad que se obtiene usando el enfoque de celdas e interpolación bilineal. En la parte inferior se mostrarán, dependiendo de la figura, las diferentes distribuciones de valores de propiedad que producen las representaciones sobre las que se

desea establecer una comparación. La representación mediante un *quadtree* que aparece en la parte inferior de las figuras sólo muestra los valores de propiedad de la rejilla regular, obviando el resto de valores impuestos por la partición binaria espacial de la estructura de datos.

En la figura 2.16 se presenta la distribución de valores de propiedad que proporciona el enfoque de celdas (parte superior) y el enfoque de *voxels* (parte inferior) sobre la rejilla regular y uniforme de ejemplo. La imagen inferior permite observar las discontinuidades de salto entre *voxels* vecinos que provoca la interpolación de orden cero utilizada en el enfoque de *voxels*.

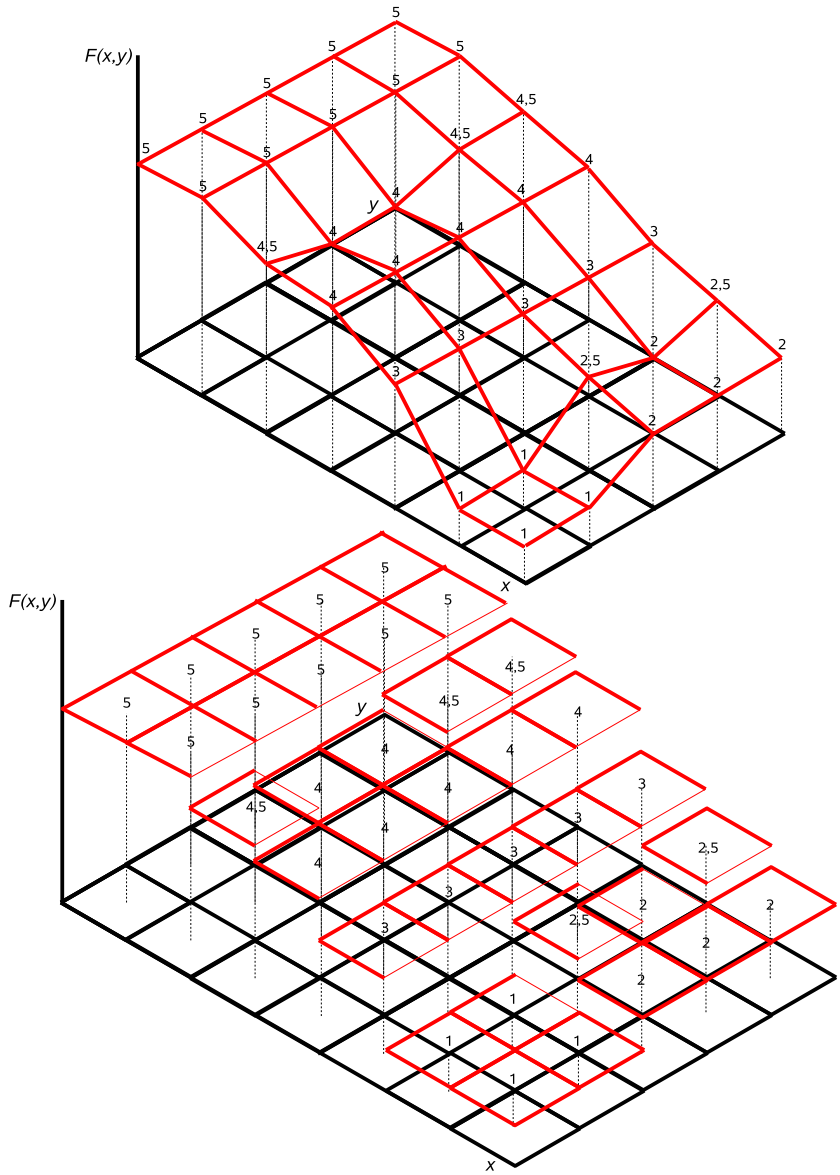

Figura 2.16: Ejemplo 2-D en el que se muestra la distribución de valores de propiedad asociada a una rejilla regular de muestras usando: el enfoque de celdas (imagen superior) y el de *voxels* (imagen inferior).

En la parte inferior de la figura 2.17 se puede observar la distribución de valores

de propiedad que resulta de aplicar la agregación de celdas de menor tamaño en una de tamaño mayor. Las muestras asociadas a las celdas de menor tamaño cumplen un criterio de similaridad $\epsilon = 1,5$. Como puede comprobarse, la agregación de celdas suaviza la distribución de valores de propiedad en las áreas de la rejilla regular en donde se ha detectado una región homogénea. La función de distribución de valores de propiedad resultante es continua aunque puede presentar los problemas de ambigüedad entre celdas vecinas de distinto tamaño, como se ha comentado con anterioridad.

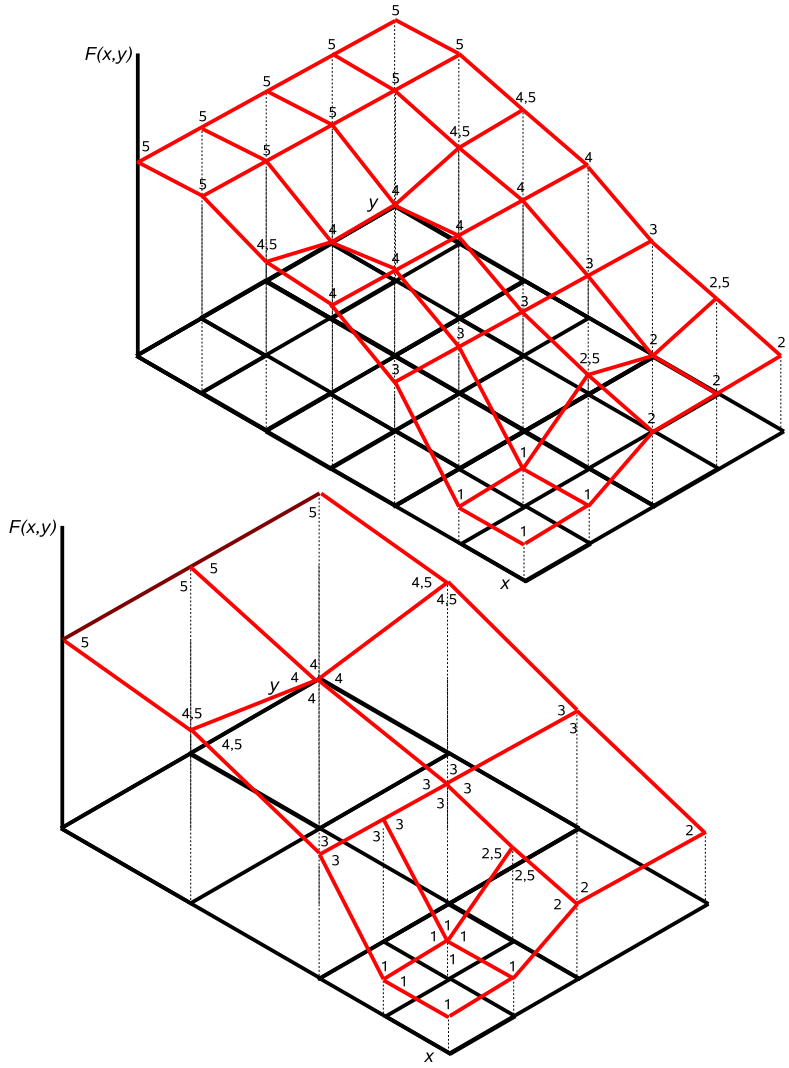

Figura 2.17: Ejemplo 2-D en el que se muestra la distribución de valores de propiedad asociada a una rejilla regular de muestras usando el enfoque de celdas (imagen superior) y la distribución asociada al *quadtree* obtenido al aplicar un criterio de similaridad entre muestras igual a $1,5$ (imagen inferior).

En la parte inferior de la figura 2.18 se puede observar la distribución de valores de propiedad resultado de aplicar el concepto de región homogéna a un enfoque de *voxels* con un criterio de similaridad $\epsilon = 1,5$. La función de distribución de valores de propiedad continua presentando discontinuidad de salto.

Nuestra representación proporciona una distribución de valores de propiedad me-

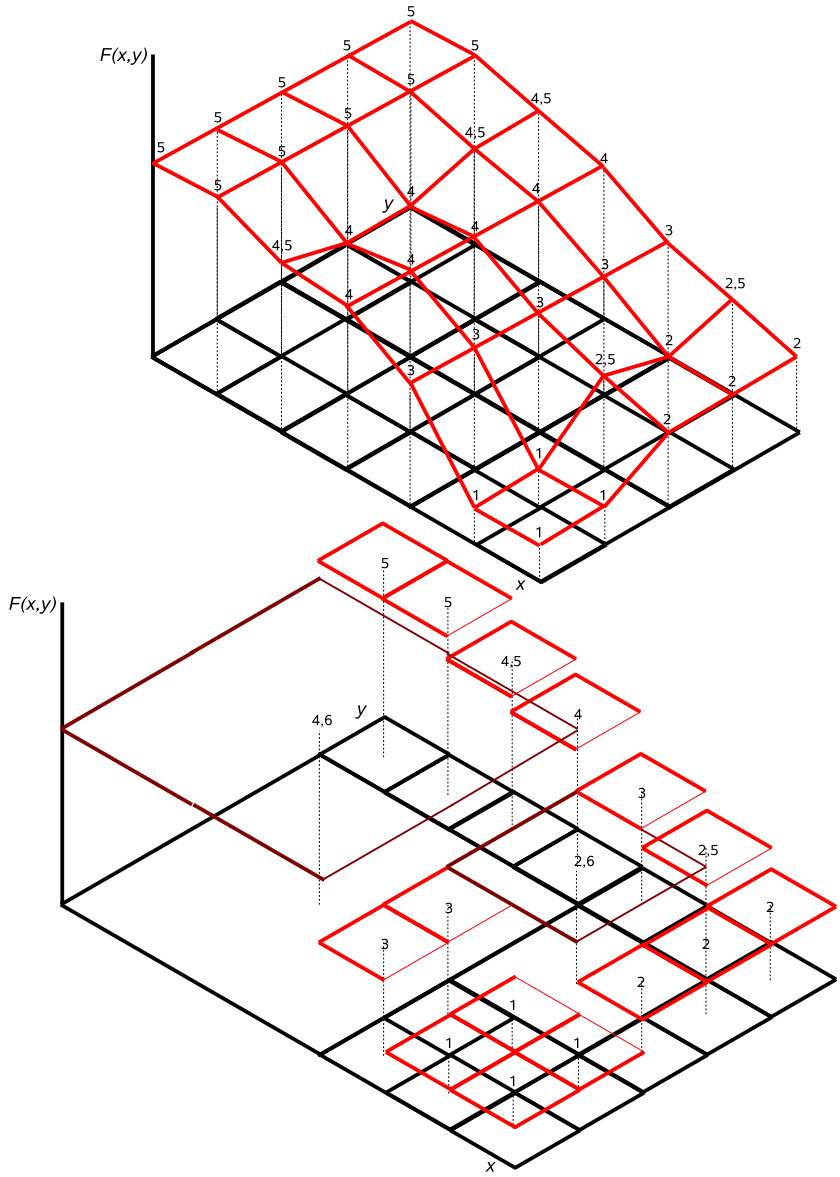

Figura 2.18: Ejemplo 2-D en el que se usa un *quadtree* de "*voxels*" de tamaño variable (imagen inferior) para representar las regiones homogéneas que aparecen en la rejilla regular y uniforme que se muestra en la imagen superior.

diante una función continua (continuidad C^0) por partes que combina la interpolación de orden cero del enfoque de *voxels* y la interpolación trilineal del enfoque de celdas. Sin embargo, la interpolación trilineal aplicada en nuestra representación está libre de los problemas de inconsistencia descritos anteriormente para representaciones irregulares basadas en el enfoque de celdas y, por tanto, no es necesario realizar soluciones parciales a dichos problemas. En la parte inferior de la figura 2.19 se puede observar la distribución de valores de propiedad que proporciona nuestra representación. La partición de los *voxels* en zonas interiores y bordes aparece coloreada en verde y azul celeste respectivamente. Los valores de propiedad que se corresponden con los puntos incluidos en las zonas interiores presentan un aspecto plano ya que son asignados mediante inter-

polación de orden cero. Sin embargo, el valor de propiedad de los puntos incluidos en las zonas de borde se calcula mediante interpolación trilineal (bilineal en el caso de la figura) de los valores correspondientes a los *voxels* vecinos. De esta forma, obtenemos la ventaja del ahorro de espacio al aplicar el concepto de región homogénea a un enfoque de *voxel* y la ventaja de la continuidad de la función por partes que se puede conseguir en el enfoque de celdas, una vez salvados los problemas de inconsistencia anteriormente descritos. La ventaja adicional es que en nuestro enfoque híbrido no es necesario resolver explícitamente los problemas de inconsistencia del enfoque de celdas sobre rejillas irregulares.

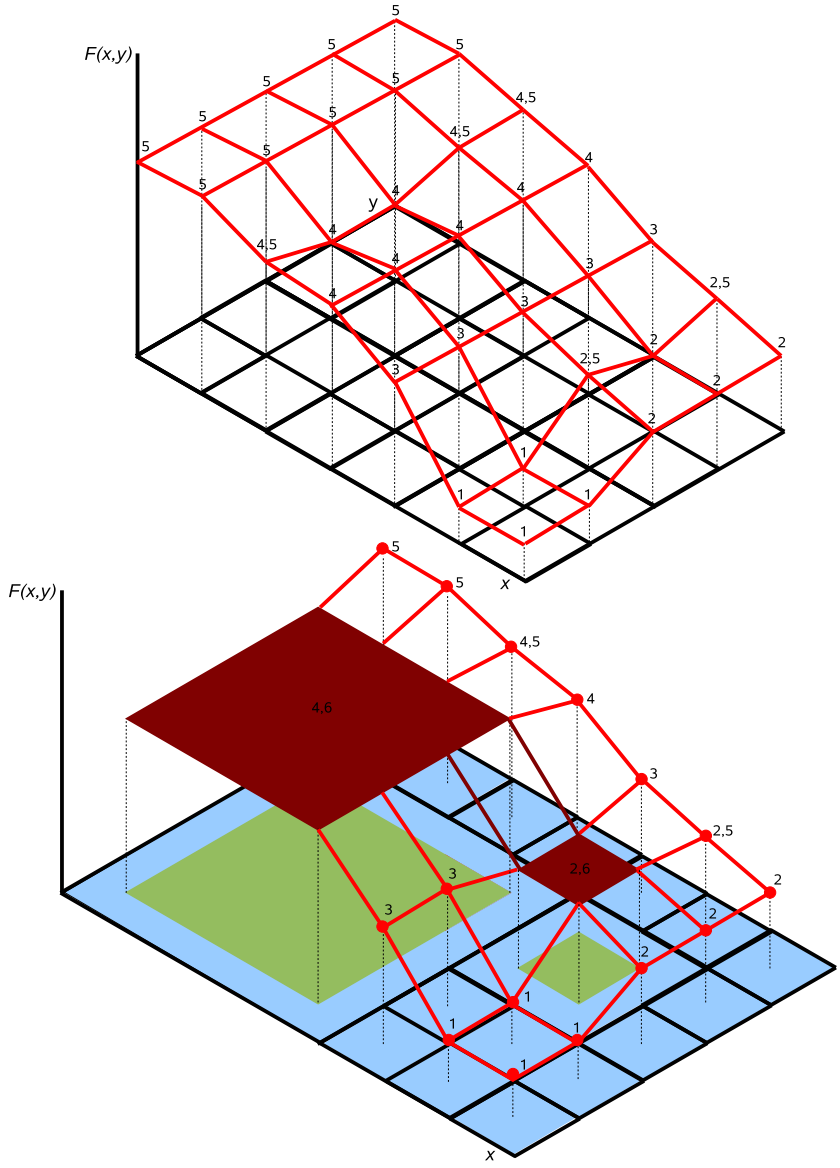

Figura 2.19: Ejemplo 2-D en el que se usa el *HRB-Octree* y la distribución de valores de propiedad que proporciona. Se ha utilizado un criterio de similaridad $\epsilon = 1,5$ para detectar regiones homogéneas sobre la rejilla regular y uniforme.

2.5. Evaluación de la representación

Para evaluar nuestra representación vamos a realizar una comparación con respecto a una representación basada en el enfoque de celdas aplicado sobre una rejilla regular, que es el más comúnmente utilizado (Ver figura 2.19).

La métrica que utilizaremos para la comparación entre representaciones se establece en el espacio de valores de propiedad mediante una medida estándar del error. Vamos a considerar la desviación cometida a la hora de calcular el valor de propiedad de un punto del espacio mediante la representación de regiones homogéneas con borde, con respecto al cálculo del valor de propiedad del mismo punto mediante el enfoque de celdas. Para establecer la medida del error cometido consideramos como valor correcto el que proporciona el enfoque de celdas sobre cualquier punto y, como valor estimado, el proporcionado por el enfoque de regiones homogéneas con borde.

2.5.1. Estudio preliminar del error de interpolación

La representación de volúmenes mediante rejilla regular de celdas utiliza normalmente una función de interpolación trilineal para calcular el valor de propiedad de cada punto incluido en las celdas. La función se define por partes, coincidiendo el dominio de cada parte con el espacio ocupado por una celda de la rejilla. En cada celda se utiliza una función de interpolación que depende de los valores de propiedad situados en sus vértices.

La representación de volúmenes mediante *HRB-Octree* utiliza una partición del espacio ocupado por las regiones homogéneas (*voxels*), particionándolo en zonas interiores y bordes. La representación utiliza una función de interpolación de orden cero para calcular el valor de propiedad de los puntos incluidos en zonas interiores y una interpolación trilineal para el cálculo del valor de propiedad de los puntos incluidos en los bordes.

Teniendo en cuenta lo anterior, procedemos a analizar la forma de la función de interpolación lineal con el objetivo de realizar un primer acercamiento al error que se comete en el cálculo del valor de propiedad en el *HRB-Octree*, con respecto al cálculo del valor de propiedad en la rejilla de celdas. Se va a estudiar, por cuestiones de claridad, la interpolación lineal, es decir, el caso 1-D, ya que los resultados obtenidos son fácilmente generalizables al caso 3-D.

La función de interpolación lineal en el caso de un intervalo (celda 1-D) $[x_i, x_{i+1}]$ puede expresarse de la siguiente manera:

$$\frac{x - x_i}{x_{i+1} - x_i} = \frac{f(x) - f_{xi}}{f_{xi+1} - f_{xi}}$$

$$f(x) = \frac{x - x_i}{x_{i+1} - x_i}(f_{xi+1} - f_{xi}) + f_{xi} \tag{2.5}$$

donde f_{xi}, f_{xi+1}, son los valores de propiedad correspondientes a los extremos del intervalo $[x_i, x_{i+1}]$, x representa cualquier punto incluido en el intervalo y $f(x)$ la expresión de la función de interpolación.

En la expresión 2.5 podemos comprobar que la función de interpolación depende de los valores de propiedad asociados a los extremos del intervalo, así como de las posiciones de dichos extremos, en particular de la distancia entre estos (amplitud del intervalo).

Recordemos el concepto de región homogéna, junto con su criterio de similaridad asociado, el cuál permite construir regiones homogéneas en base a regiones homogéneas vecinas de tamaño menor, utilizando para ello una operación de agregación. Como se ha definido anteriormente, un conjunto de regiones homogéneas vecinas del mismo tamaño pueden formar una región homogenea de mayor tamaño, siempre que, los valores de propiedad de todas las regiones que van a ser agregadas cumplan, dos a dos, la siguiente inecuación:

$$|v_i - v_j| < \epsilon, \forall\ (\boldsymbol{P_i}, v_i), (\boldsymbol{P_j}, v_j) \in U \tag{2.6}$$

siendo v_i, v_j los valores de propiedad de todos los elementos que permiten la formación de la nueva región.

Si todas las parejas de valores cumplen el criterio se asigna un nuevo valor de propiedad a la región homogénea resultante. Dicho valor es una combinación de los valores de las regiones componentes. En nuestro caso hemos utilizado un criterio simple basado en asignar el valor medio de propiedad.

Teniendo en cuenta el criterio de similaridad 2.6 y la asignación a la nueva región del valor medio de propiedad de las regiones componentes podemos inferir, analizando la expresión 2.5 que, conforme mayor es la diferencia media entre pares de valores de propiedad, mayor error se comete utilizando el *HRB-Octree*, dependiendo dicho error del valor finalmente asignado a la nueva región homogénea. Por tanto, cuanto mayor sea la distancia entre valores de propiedad permitida por el criterio de similaridad, mayor error se cometerá en la nueva región homogénea. Hay que tener en cuenta que en el *HRB-Octree*, las regiones homogéneas están representadas mediante *voxels* de diferente tamaño asociados a sus nodos hoja.

En la figura 2.20 se puede observar una serie de funciones de interpolación lineal sobre los bordes vecinos de dos *voxels* vecinos de distinto tamaño. En la parte superior se muestra la interpolación lineal en la celda de la rejilla uniforme y regular que se corresponde con los bordes de interés en el *HRB-Octree*. El valor Δ_x indica la distancia entre las muestras de máxima resolución. La primera función de la serie (siguiente gráfica hacia abajo) muestra que el *voxel* de mayor tamaño procede de dos *voxels* de menor tamaño con igual valor de propiedad. En este caso, no existe diferencia alguna entre los valores de propiedad (segmento de línea rojo) calculados en los bordes de interés, antes y después de la operación de agregación. Por este motivo no se ve el segmento de línea en gris que representa la distribución de valores de propiedad previa a la agregación. Además, no existe diferencia entre dichos valores y los calculados, para la misma zona, usando la rejilla de celdas. Los siguientes elementos de la serie de funciones muestran, en orden creciente, un aumento de la diferencia entre los valores de propiedad de los *voxels* que se agregan para formar el *voxel* de mayor tamaño. Como puede comprobarse visualmente, el error aumenta progresivamente en función del valor medio asignado. El valor de propiedad para los puntos incluidos en los *voxels* de máxima resolución que forman el *voxel* de tamaño $2\Delta_x$, aparece representado por segmentos de línea coloreados en gris. Dicho valor coincide con el obtenido a partir de la rejilla de celdas.

Por otra parte, la expresión 2.5 también depende de la amplitud del intervalo $[x_i, x_{i+1}]$. Sin embargo, en este caso, la representación *HRB-Octree* permite que la interpolación sea invariante con respecto a la variación de dicha amplitud. El borde de un *voxel*, independientemente del tamaño de éste, mantiene siempre la misma dimen-

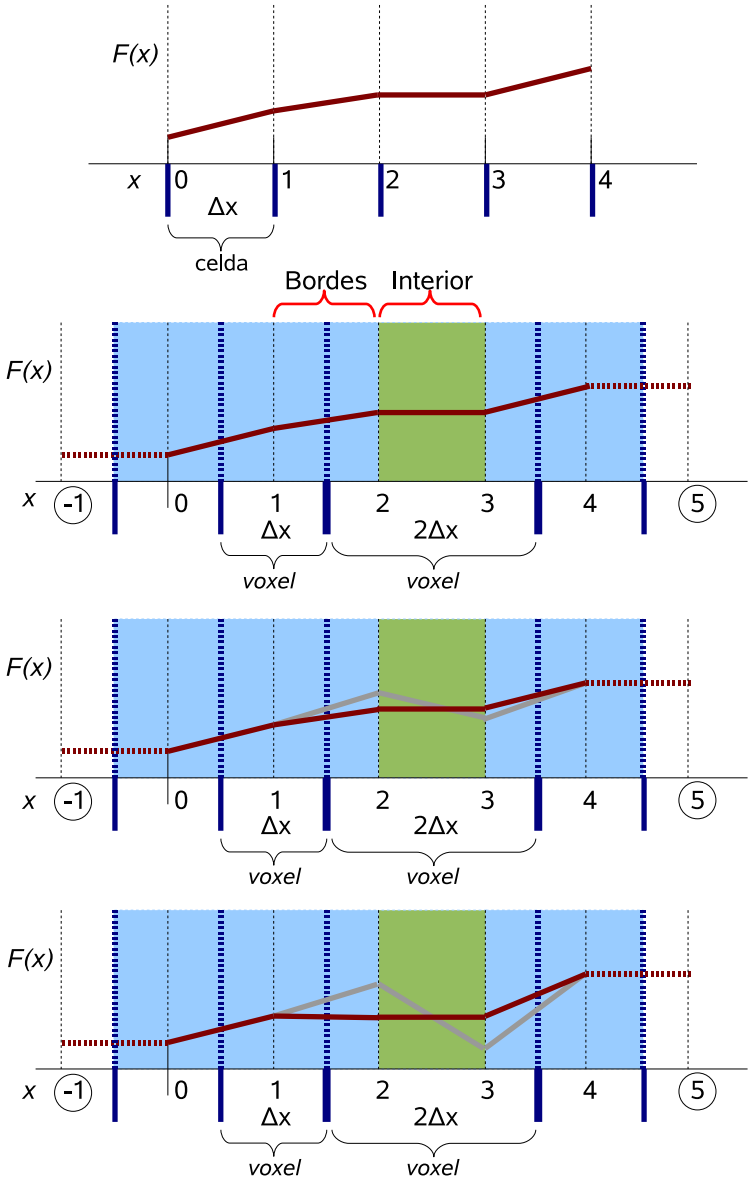

Figura 2.20: Comparación entre los valores de propiedad obtenidos mediante interpolación lineal en los bordes de dos *voxels* vecinos de diferente tamaño conforme aumenta la diferencia entre sus valores de propiedad (gris) y, los valores de propiedad obtenidos usando el *HRB-Octree*.

sión a la hora de la interpolación. Los bordes solo crecen longitudinalmente, en cada dimensión correspondiente, conforme aumenta el tamaño del *voxel* al que pertenecen. El tamaño de *voxel* depende exclusivamente del número de operaciones de agregación aplicadas.

En la figura 2.21 se muestra una serie de funciones de interpolación sobre los bordes vecinos de dos *voxels* vecinos, izquierdo y derecho. La primera función de la serie muestra la interpolación en los bordes de interés para dos *voxels* de máxima resolución. Conforme aumenta el tamaño del *voxel* de la derecha se puede observar que la función de interpolación resultante mantiene los mismos valores en los bordes de interés. Obviamente, estamos suponiendo que las sucesivas operaciones de agregación no modifican

el valor inicial que presentaba el *voxel* de máxima resolución situado a la derecha. Es decir, los valores de propiedad de los subsiguientes *voxels* de mayor tamaño son iguales al inicial. Sin embargo, es razonable pensar que conforme aumente el tamaño de un *voxel* y, en consecuencia, el número de *voxels* de menor tamaño agregados, aumentará la probabilidad de que no se mantenga dicho valor.

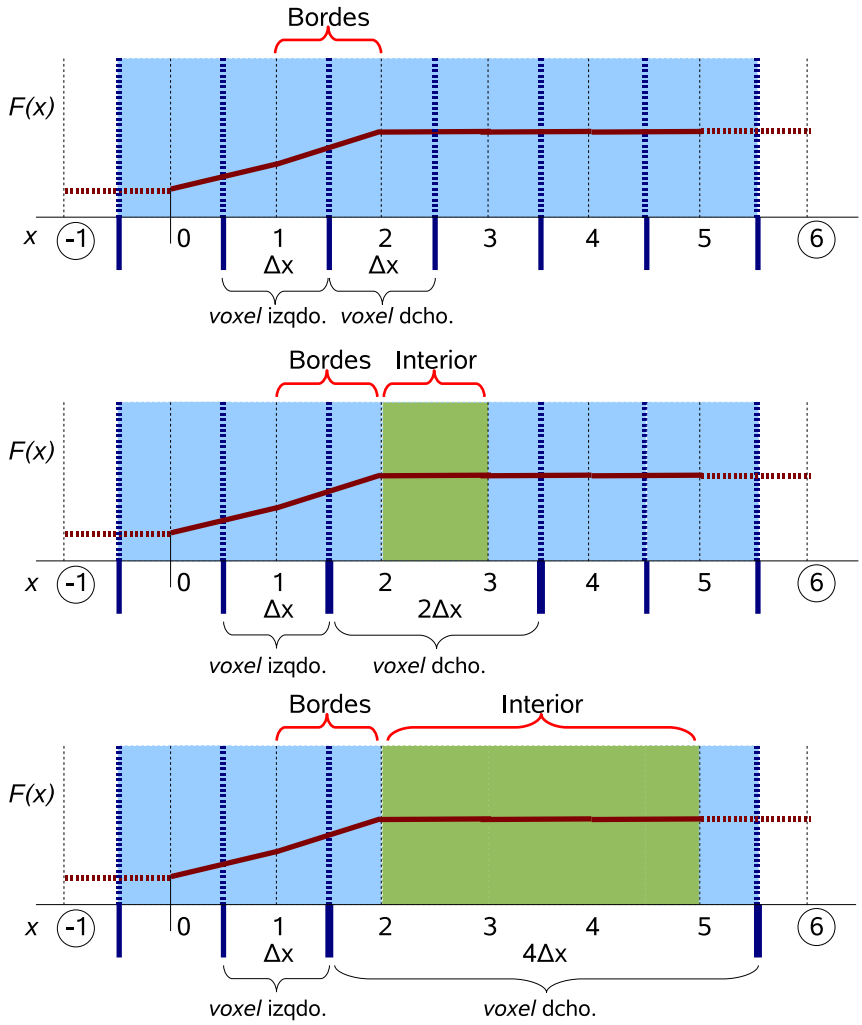

Figura 2.21: Comparación entre los valores de propiedad obtenidos mediante interpolación lineal en los bordes de dos *voxels* conforme el tamaño de una de ellos aumenta, suponiendo que el valor de propiedad asignado a cada nuevo *voxel* permanece constante.

Podemos concluir, una vez realizadas las anteriores observaciones que, el error cometido al calcular valores de propiedad en el *HRB-Octree*, con respecto a una rejilla regular y uniforme de celdas, depende exclusivamente de la diferencia de los valores de propiedad que presenten los *voxels* de máxima resolución que componen un *voxel* de mayor resolución con respecto al valor de propiedad asignado a este *voxel*. Nuestra representación evita que la interpolación dependa de la amplitud del intervalo, usando para ello la partición en zonas interiores y bordes y, por tanto, reducimos la posibilidad de cometer error al dominio de valores de propiedad, no teniendo que considerar el dominio de las posiciones espaciales de las muestras. Este hecho nos permite definir la métrica que se

describirá en el siguiente apartado. Esta métrica tomará únicamente en consideración una distancia en el dominio de valores de propiedad con el objetivo de medir el error.

2.5.2. Métrica de error

Para establecer una medida del error consideramos que la función, $F_C(x, y, z)$, la cual proporciona los valores de propiedad en todo el volumen ocupado por una rejilla regular de celdas, viene definida como una función de interpolación trilineal por partes. Cada parte está representada por cada una de las celdas que componen la rejilla. Por tanto, la definición de dicha función puede expresarse formalmente de la siguiente manera:

$$F_C(x,y,z) = \begin{cases} F_{[x_0,y_0,z_0]}(x,y,z) & \text{si } (x,y,z) \in Cell_{x_0,y_0,z_0} \\ F_{[x_0,y_0,z_1]}(x,y,z) & \text{si } (x,y,z) \in Cell_{x_0,y_0,z_1} \\ \cdots & \\ F_{[x_i,y_j,z_k]}(x,y,z) & \text{si } (x,y,z) \in Cell_{x_i,y_j,z_k} \\ \cdots & \\ F_{[x_{L-1},y_{M-1},z_{N-1}]}(x,y,z) & \text{si } (x,y,z) \in Cell_{x_{L-1},y_{M-1},z_{N-1}} \end{cases} \tag{2.7}$$

donde $F_{[x_i,y_j,z_k]}(x, y, z)$ es la función de interpolación trilineal en la celda $Cell_{x_i,y_j,z_k}$ con $i \in \{0, 1, \ldots, L\}$, $j \in \{0, 1, \ldots, M\}$, $k \in \{0, 1, \ldots, N\}$, siendo $L-1$ x $M-1$ x $N-1$ el número total de celdas de la rejilla. La forma de la función permite una continuidad de clase C^0 en todo el espacio ocupado por la rejilla.

La función que proporciona los valores de propiedad en todo el volumen ocupado por un *HRB-Octree*, $F(x, y, z)$, está definida por partes sobre la partición que divide a los *voxels* en zonas interiores y exteriores (bordes). El cálculo del valor de propiedad en las zonas interiores se realiza mediante una función constante igual al valor de propiedad asociado a cada *voxel*. Sin embargo, en los bordes este cálculo se realiza aplicando funciones de interpolación trilineal, que son seleccionadas dependiendo de la posición del punto $\boldsymbol{P} = (x, y, z)$. Notaremos las funciones de interpolación trilineal de la siguiente forma:

$$F_{[i]_j}(x, y, z)$$

donde $i = 0, 1, \ldots, L$ representa un determinado *voxel* del volumen, y $j = 0, 1, \ldots, M$ identifica cada una de las M funciones de interpolación que engloban el borde de dicho *voxel* i. Dependiendo de la ubicación del punto $\boldsymbol{P} = (x, y, z)$ en el borde, varía la función de interpolación $F_{[i]_j}(x, y, z)$ aplicada. El número de funciones de interpolación, M, depende del tamaño del *voxel* y de los tamaños de sus *voxels* vecinos. Formalmente podemos expresar la función de la siguiente manera:

$$F(x,y,z) = \begin{cases} F_0 & \text{si } (x,y,z) \in Inside_0 \\ F_{[0]_j}(x,y,z), \text{para algún } j = 0,1,\ldots,n_0 & \text{si } (x,y,z) \in Border_0 \\ \\ F_1 & \text{si } (x,y,z) \in Inside_1 \\ F_{[1]_j}(x,y,z), \text{para algún } j = 0,1,\ldots,n_1 & \text{si } (x,y,z) \in Border_1 \\ \cdots & \\ F_i & \text{si } (x,y,z) \in Inside_i \\ F_{[i]_j}(x,y,z), \text{para algún } j = 0,1,\ldots,n_i & \text{si } (x,y,z) \in Border_i \\ \cdots & \\ F_L & \text{si } (x,y,z) \in Inside_L \\ F_{[L]_j}(x,y,z), \text{para algún } j = 0,1,\ldots,n_L & \text{si } (x,y,z) \in Border_L \end{cases} \tag{2.8}$$

donde F_i representa el valor de propiedad del *voxel* i, $Inside_i$ representa el volumen del espacio ocupado por la zona interior de este *voxel* y, $Border_i$ representa el volumen ocupado por el borde.

Consideramos $F_C(x,y,z)$ como la función correcta de distribución de valores de propiedad en todo el volumen y por tanto ocupa el lugar de la función conocida, $f(x,y,z)$, que es interpolada en una representación volumétrica. $F(x,y,z)$ es la función que estima la distribución "correcta" de valores de propiedad. Se define una función, $E(x,y,z) = f(x,y,z) - F(x,y,z)$, que permite medir el error cometido al calcular el valor de propiedad en un punto $\boldsymbol{P} = (x,y,z)$ usando *HRB-Octree* con respecto a calcular su valor utilizando la rejilla regular y uniforme de celdas. Si se observa la definición de las funciones f y F se puede comprobar que ambas toman valores en el dominio $\mathbb{R}^3$ y tienen como rango el espacio de valores de propiedad $\mathcal{V} \subset \mathbb{R}$.

$$f : \mathbb{R}^3 \longrightarrow \mathcal{V} \subset \mathbb{R}$$
$$F : \mathbb{R}^3 \longrightarrow \mathcal{V} \subset \mathbb{R}$$

Por tanto, para establecer la función de error $E(x,y,z)$ se puede utilizar una métrica en el conjunto $\mathbb{R}$, ya que como se ha mostrado, el error cometido solo depende de la diferencia entre valores de propiedad. La función valor absoluto, $|\cdot|$ de la diferencia entre los valores de propiedad de ambas funciones permite establecer una métrica correcta en $\mathbb{R}$.

$$E(x,y,z) = |f(x,y,z) - F(x,y,z)| \tag{2.9}$$

Una vez que se dispone de la métrica para medir el error se procede a estudiar la forma que tiene esta función. Por cuestiones de simplicidad en la exposición del cálculo, a la hora de estudiar el error se va a considerar el caso 1-D, aunque los resultados obtenidos pueden generalizarse al caso 3-D. El error cometido al utilizar *HRB-Octree* proviene de las operaciones de agregación que generan *voxels* de mayor tamaño que los *voxels* originales de máxima resolución. Como se ha descrito en secciones anteriores, si no se realiza ninguna operación de agregación, nuestra representación proporciona los mismos valores de propiedad que la rejilla regular y uniforme de celdas. Basándonos en este hecho, comenzamos estudiando el error que se comete al realizar la agregación de dos *voxels* de máxima resolución. En el gráfico de la parte inferior de la figura 2.22 puede observarse la distribución de valores de propiedad para una serie de *voxels* de máxima resolución (en gris), la cual coincide con la proporcionada por la rejilla de

celdas (parte superior de la figura). Además, se puede ver la nueva distribución de valores de propiedad (en rojo) resultado de aplicar una operación de agregación sobre los *voxels* centrados en los puntos 2 y 3. El nuevo *voxel* está centrado en el punto x_C, tiene asociado un valor de propiedad F_C que es la media de los valores de propiedad de sus *voxels* componentes y, se encuentra delimitado por el intervalo $[a, b]$.

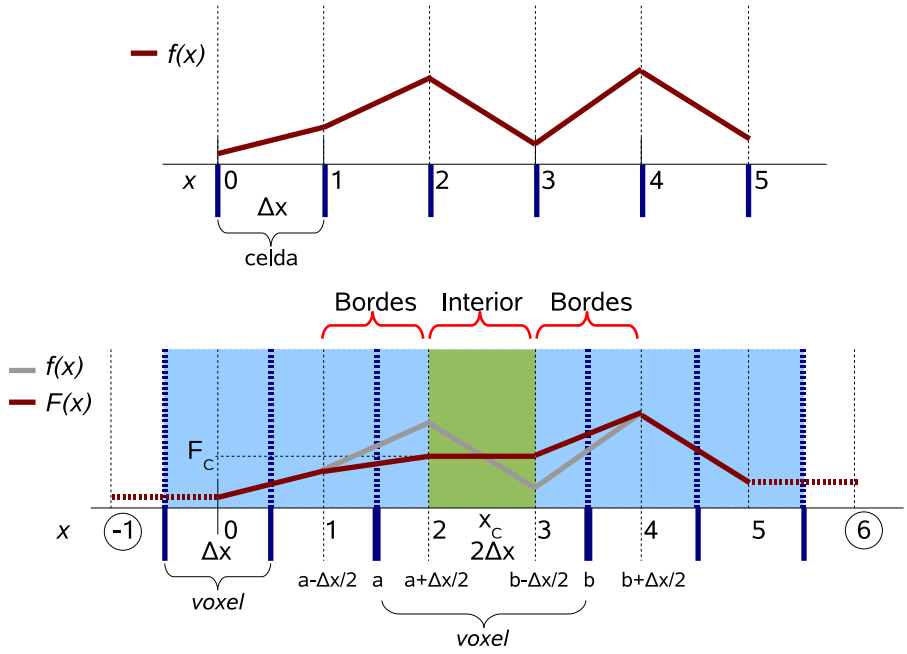

Figura 2.22: Error cometido en el cálculo del valor de propiedad cuando se realiza una operación de agregación de dos *voxels* vecinos de máxima resolución.

Como se puede comprobar visualmente en la figura 2.22, el error resultado de la agregación se comete en el intervalo $[a - \frac{\Delta_x}{2}, b + \frac{\Delta_x}{2}]$. Debido a que las funciones $f(x)$ y $F(x)$ están definidas por partes, la función de error 2.9 para cualquier punto del intervalo en el caso 1-D viene definida de la siguiente forma:

$$E(x) : [a - \frac{\Delta_x}{2}, b + \frac{\Delta_x}{2}] \longrightarrow \mathcal{V} \subset \mathbb{R}$$

$$E(x) = \begin{cases} |f_{[a-\frac{\Delta_x}{2}, a+\frac{\Delta_x}{2}]}(x) - F_{[a-\frac{\Delta_x}{2}, a+\frac{\Delta_x}{2}]}(x)| & \text{si } x \in [a - \frac{\Delta_x}{2}, a + \frac{\Delta_x}{2}] \\ |f_{[a+\frac{\Delta_x}{2}, b-\frac{\Delta_x}{2}]}(x) - F_{[a+\frac{\Delta_x}{2}, b-\frac{\Delta_x}{2}]}(x)| & \text{si } x \in [a + \frac{\Delta_x}{2}, b - \frac{\Delta_x}{2}] \\ |f_{[b-\frac{\Delta_x}{2}, b+\frac{\Delta_x}{2}]}(x) - F_{[b-\frac{\Delta_x}{2}, b+\frac{\Delta_x}{2}]}(x)| & \text{si } x \in [b - \frac{\Delta_x}{2}, b + \frac{\Delta_x}{2}] \end{cases} \quad (2.10)$$

Hemos utilizado la notación $f_{[a,b]}(x)$ para representar la función de interpolación lineal para cualquier punto del intervalo $[a, b]$ dados los valores de propiedad f_a y f_b en los extremos de dicho intervalo. Desarrollando algebráicamente la función 2.10 obtenemos la siguiente expresión para el error, en función del valor de propiedad, F_C, asignado al nuevo *voxel*:

$$E(x) = \begin{cases} |(\frac{1}{2} + \frac{x-a}{\Delta_x}) \cdot (f_{a+\frac{\Delta_x}{2}} - F_C)| & \text{si } x \in [a - \frac{\Delta_x}{2}, a + \frac{\Delta_x}{2}] \\ |(\frac{1}{2} + \frac{x-a}{(b-a)-2\Delta_x}) \cdot (f_{b-\frac{\Delta_x}{2}} - f_{a+\frac{\Delta_x}{2}}) + \\ +(f_{a+\frac{\Delta_x}{2}} - F_C)| & \text{si } x \in [a + \frac{\Delta_x}{2}, b - \frac{\Delta_x}{2}] \\ |(\frac{1}{2} + \frac{x-b}{\Delta_x}) \cdot (F_C - f_{b-\frac{\Delta_x}{2}}) + \\ +(f_{b-\frac{\Delta_x}{2}} + F_C)| & \text{si } x \in [b - \frac{\Delta_x}{2}, b + \frac{\Delta_x}{2}] \end{cases} \quad (2.11)$$

Hemos utilizado la notación f_i para representar los valores de propiedad $f(i) = v_i$ que son proporcionados inicialmente para cada uno de los puntos i. Si tomamos como valor de F_C la media de todos los valores de propiedad de los *voxels* agregados para generar el nuevo *voxel*, entonces se obtiene como resultado la siguiente expresión:

Sea,

$$F_C = \frac{f_{a+\frac{\Delta_x}{2}} + f_{b-\frac{\Delta_x}{2}}}{2}$$

Entonces,

$$E(x) = \begin{cases} |(\frac{1}{4} + \frac{x-a}{2\Delta_x}) \cdot (f_{a+\frac{\Delta_x}{2}} - f_{b-\frac{\Delta_x}{2}})| & \text{si } x \in [a - \frac{\Delta_x}{2}, a + \frac{\Delta_x}{2}] \\ |\frac{x-a}{(b-a)-2\Delta_x} \cdot (f_{b-\frac{\Delta_x}{2}} - f_{a+\frac{\Delta_x}{2}})| & \text{si } x \in [a + \frac{\Delta_x}{2}, b - \frac{\Delta_x}{2}] \\ |(\frac{b-x}{2\Delta_x} - \frac{1}{4}) \cdot (f_{b-\frac{\Delta_x}{2}} - f_{a+\frac{\Delta_x}{2}}) + f_{b-\frac{\Delta_x}{2}}| & \text{si } x \in [b - \frac{\Delta_x}{2}, b + \frac{\Delta_x}{2}] \end{cases} \quad (2.12)$$

Las expresiones (2.11) y (2.12) permiten determinar el error cometido por el *HRB-Octree* con respecto a la rejilla regular de muestras en cualquier punto del espacio ocupado por las representaciones. Sin embargo, estas expresiones del error sólo son válidas en el caso de agregar dos regiones de máxima resolución. Como el *HRB-Octree* permite la agregación de regiones de cualquier resolución, siempre que cumplan la restricción de partición espacial binaria y el criterio de similaridad, es necesario generalizar la expresión del error.

Vamos a poner de manifiesto una serie de consideraciones que nos ayudan a entender la expresión general de $E(x)$. En primer lugar, hemos de tener en cuenta que la representación jerárquica mediante partición espacial binaria impone una restricción a la hora de generar regiones homogéneas por agregación: el tamaño, t_i, de cualquier región homogénea siempre es igual a:

$$t_i = 2^i \Delta_x, \quad i = 0, 1, \ldots, n \quad (2.13)$$

siendo 2^i el número de *voxels* de máxima resolución contenidos en el *voxel* i y, Δ_x el tamaño de los *voxels* de máxima resolución.

Además, hemos de tener en cuenta que debido a la misma restricción comentada anteriormente, un nuevo *voxel* siempre va a estar constituido por dos *voxels* vecinos del mismo tamaño. Por tanto, el tamaño, t_{xC} del nuevo *voxel* es igual a:

$$t_{xC} = t_i + t_{i+1} = 2^i \Delta_x + 2^i \Delta_x = 2^{i+1}, \quad i = 0, 1, \ldots, n \quad (2.14)$$

siendo $t_i = t_{i+1} = 2^i \Delta_x$ el tamaño de las dos *voxels* vecinos agregados i e $i+1$.

En segundo lugar, hay que tener en cuenta que el intervalo en el que debemos estudiar el error siempre está formado por los bordes del *voxel* resultado de agregaciones (y los bordes vecinos de sus *voxels* vecinos correspondientes) y por su zona interior; independientemente de su tamaño. Este hecho puede comprobarse en las figuras 2.22 y 2.23. El intervalo $[a - \frac{\Delta_x}{2}, a + \frac{\Delta_x}{2}]$ se corresponde con los bordes situados en la parte izquierda del nuevo *voxel*, el intervalo $[a + \frac{\Delta_x}{2}, b - \frac{\Delta_x}{2}]$ se corresponde con la zona interior, y el intervalo $[b - \frac{\Delta_x}{2}, b + \frac{\Delta_x}{2}]$ se corresponde con los bordes situados en la parte derecha del nuevo *voxel*. La amplitud del primer y último intervalos no depende del número de los *voxels* de máxima resolución que componen el nuevo *voxel*, ya que es un invariante de la representación. El número de *voxels* de máxima resolución agregados hasta el momento sólo influye en el tamaño de la zona interior, que es igual a $2^i \Delta_x - \Delta_x$.

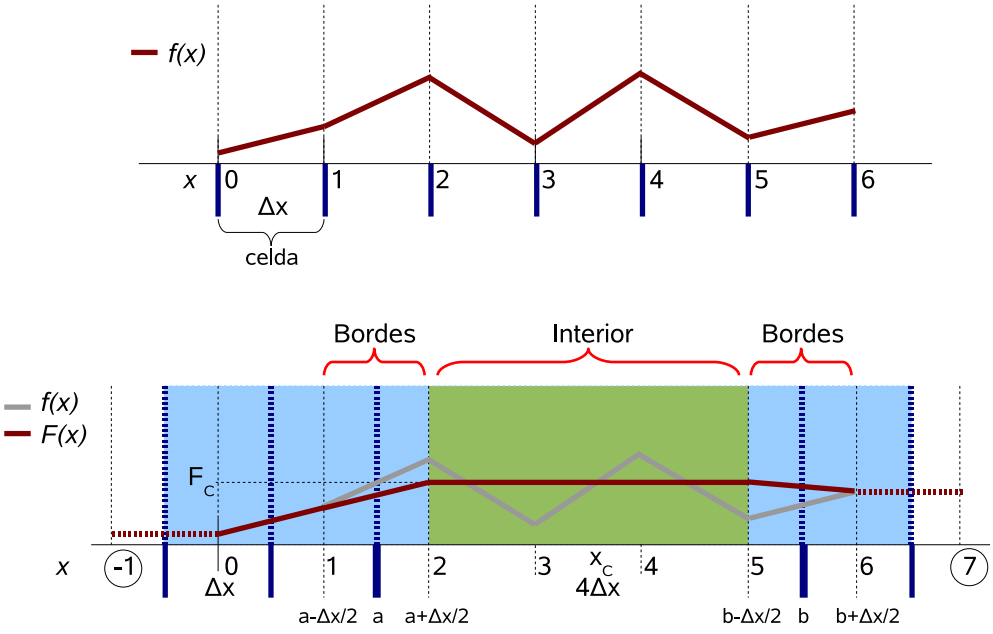

Figura 2.23: Error cometido en el cálculo del valor de propiedad cuando se realiza una operación de agregación de dos *voxels* vecinos de tamaño $2\Delta_x$.

Por último lugar, como uno de los intervalos (la zona interior) de definición del dominio de la función de error tiene una amplitud que depende del tamaño del *voxel* agregado, tenemos que expresar la función de error en dicho intervalo como un número variable de funciones, correspondiéndose cada una de ellas con el "tramo de interpolación" que existiría entre *voxels* de máxima resolución, caso de no haberse producido la serie de operaciones de agregación que han dado como resultado el nuevo *voxel*. El número de "tramos de interpolación" entre *voxels* de máxima resolución para cualquier *voxel* de mayor tamaño coincide con el número de operaciones de agregación que han sido aplicadas para generarlo; y es igual a $2^i - 1$ siendo 2^i el número de *voxels* de máxima resolución incluidos.

Teniendo en cuenta las anteriores consideraciones, podemos definir la función de error, $E(x)$, en el intervalo $[a - \frac{\Delta_x}{2}, b + \frac{\Delta_x}{2}]$ de la siguiente forma:

$$E(x) : [a - \frac{\Delta_x}{2}, b + \frac{\Delta_x}{2}] \longrightarrow \mathcal{V} \subset \mathbb{R}$$

$$E(x) = \begin{cases} |f_{[a-\frac{\Delta_x}{2},a+\frac{\Delta_x}{2}]}(x) - F_{[a-\frac{\Delta_x}{2},a+\frac{\Delta_x}{2}]}(x)| & \text{si } x \in [a - \frac{\Delta_x}{2}, a + \frac{\Delta_x}{2}] \\ |f_{[a+\frac{\Delta_x}{2},b-\frac{\Delta_x}{2}]}(x) - F_{[a+\frac{\Delta_x}{2},b-\frac{\Delta_x}{2}]}(x)| & \text{si } x \in [a + \frac{\Delta_x}{2}, b - \frac{\Delta_x}{2}] \\ |f_{[b-\frac{\Delta_x}{2},b+\frac{\Delta_x}{2}]}(x) - F_{[b-\frac{\Delta_x}{2},b+\frac{\Delta_x}{2}]}(x)| & \text{si } x \in [b - \frac{\Delta_x}{2}, b + \frac{\Delta_x}{2}] \end{cases} \tag{2.15}$$

siendo,

$$f_{[a+\frac{\Delta_x}{2},b-\frac{\Delta_x}{2}]}(x) = \begin{cases} f_{[a+\frac{\Delta_x}{2},(a+\frac{\Delta_x}{2})+ \\ +\Delta_x]}(x) & \text{si } x \in [a + \frac{\Delta_x}{2}, \\ & ,(a + \frac{\Delta_x}{2}) + \Delta_x] \\ f_{[(a+\frac{\Delta_x}{2})+\Delta_x,(a+\frac{\Delta_x}{2})+ \\ +2\Delta_x]}(x) & \text{si } x \in [(a + \frac{\Delta_x}{2}) + \Delta_x, \\ & ,(a + \frac{\Delta_x}{2}) + 2\Delta_x] \\ \dots & \\ f_{[(a+\frac{\Delta_x}{2})+(2^i-2)\Delta_x,(a+\frac{\Delta_x}{2})+ \\ +(2^i-1)\Delta_x]}(x) & \text{si } x \in [(a + \frac{\Delta_x}{2}) + (2^i - 2)\Delta_x, \\ & ,(a + \frac{\Delta_x}{2}) + (2^i - 1)\Delta_x] \end{cases} \tag{2.16}$$

cumpliéndose la igualdad $b - \frac{\Delta_x}{2} = (a + \frac{\Delta_x}{2}) + (2^i - 1)\Delta_x$.

La notación utilizada es la misma que para el caso base descrito con anterioridad. Desarrollando algebráicamente la función (2.15) obtenemos la siguiente expresión para el error, en función del valor de propiedad, F_C, asignado a la nueva región homogénea:

$$E(x) = \begin{cases} |(\frac{1}{2} + \frac{x-a}{\Delta_x}) \cdot (f_{a+\frac{\Delta_x}{2}} - F_C)| & \text{si } x \in [a - \frac{\Delta_x}{2}, a + \frac{\Delta_x}{2}] \\ |f_{[a+\frac{\Delta_x}{2},b-\frac{\Delta_x}{2}]}(x) - F_C| & \text{si } x \in [a + \frac{\Delta_x}{2}, b - \frac{\Delta_x}{2}] \\ |(\frac{1}{2} + \frac{x-b}{\Delta_x}) \cdot (F_C - f_{b-\frac{\Delta_x}{2}})+ & \\ +(f_{b-\frac{\Delta_x}{2}} + F_C)| & \text{si } x \in [b - \frac{\Delta_x}{2}, b + \frac{\Delta_x}{2}] \end{cases} \tag{2.17}$$

donde, siendo $d(x) = |f_{[a+\frac{\Delta_x}{2},b-\frac{\Delta_x}{2}]}(x) - F_C|$ tenemos que,

$$d(x) = \begin{cases} |\frac{x-a}{\Delta_x} - \frac{1}{2} \cdot (f_{(a+\frac{\Delta_x}{2})+\Delta_x} - f_{a+\frac{\Delta_x}{2}}) + \\ +(f_{a+\frac{\Delta_x}{2}} - F_C)| & \text{si } x \in [a + \frac{\Delta_x}{2}, \\ & ,(a + \frac{\Delta_x}{2}) + \Delta_x] \\ |\frac{x-a}{\Delta_x} - \frac{3}{2} \cdot (f_{(a+\frac{\Delta_x}{2})+2\Delta_x} - f_{(a+\frac{\Delta_x}{2})+\Delta_x}) + \\ +(f_{(a+\frac{\Delta_x}{2})+\Delta_x} - F_C)| & \text{si } x \in [(a + \frac{\Delta_x}{2}) + \Delta_x, \\ & ,(a + \frac{\Delta_x}{2}) + 2\Delta_x] \\ \dots & \\ |\frac{x-a}{\Delta_x} - \frac{1}{2} - \frac{2^{i+1}-2^2}{2} \cdot (f_{(a+\frac{\Delta_x}{2})+(2^i-1)\Delta_x} - \\ -f_{(a+\frac{\Delta_x}{2})+(2^i-2)\Delta_x}) + (f_{(a+\frac{\Delta_x}{2})+(2^i-2)\Delta_x} - \\ -F_C)| & \text{si } x \in [(a + \frac{\Delta_x}{2}) + (2^i - 2)\Delta_x, \\ & ,(a + \frac{\Delta_x}{2}) + (2^i - 1)\Delta_x] \end{cases} \tag{2.18}$$

De esta forma, disponemos de una función de error, $E(x)$, que representa el error cometido a la hora de calcular el valor de propiedad de cualquier punto incluido en el volumen representado por el *HRB-Octree*, con respecto al valor calculado para ese mismo punto mediante la representación de rejilla regular y uniforme de celdas.

2.6. Conclusiones

Se han presentado las representaciones discretas de resolución fija y variable utilizando para ello los dos principales enfoques de trabajo que se basan en la forma de la función de interpolación que utiliza la representación. Así mismo, se han estudiado las ventajas e inconvenientes de utilizar cada uno de los enfoques en representaciones de resolución variable, haciendo hincapié en la posibilidad de ahorro de espacio de almacenamiento de muestras y en la continuidad proporcionada por la función de interpolación por partes en cada caso.

Se ha presentado una propuesta de representación basada en el enfoque de *voxels*, *HRB-Octree*, la cual permite reducir el espacio de almacenamiento utilizando el concepto de región homogénea, cuya idea subyace en el aprovechamiento de la coherencia espacial de los valores de propiedad de las muestras. Esta representación permite solucionar el problema de falta de continuidad que presenta el enfoque clásico de *voxels* mediante el establecimiento de una partición del espacio de *voxels* en dos zonas: interior y bordes. Esta nueva partición utiliza una función de interpolación distinta para cada tipo de región, consiguiendo continuidad C^0 entre las distintas regiones y, por consiguiente, en la partición de *voxels*.

Se ha estudiado el error cometido por nuestra representación con respecto a una representación de resolución fija basada en el enfoque de celdas. Para ello se ha mostrado que el error que comete nuestra representación, a la hora de calcular el valor de propiedad de un punto $\boldsymbol{P}$, sólo depende de la tolerancia de error escogida en el dominio de valores de propiedad. En base a este resultado, se ha proporcionado una expresión analítica del error.

CAPÍTULO 3

Visualización de representaciones volumétricas discretas mediante extracción de isosuperficies

El acto de *visualizar* un volumen nos permite presentar una imagen (bidimensional) o varias (una aproximación tridimensional) cuyo contenido informativo permite al usuario comprender, de manera más abordable, la gran cantidad de información que contiene un modelo volumétrico. De esta forma el usuario dispone de una herramienta que le permite comprender "por partes" las propiedades del volumen representadas en el modelo.

En este capítulo se presenta un método para visualizar representaciones discretas de volumen usando la estrategia de visualización de isosuperficies. Para llevar a cabo esta estrategia es necesario disponer de una partición de celdas que ocupe todo el espacio representado en el modelo de volumen. Nuestro objetivo es realizar una visualización de la representación basada en el concepto de región homogénea mediante una estrategia de extracción de isosuperficies.

En la primera sección, se presentan los conceptos fundamentales sobre este tipo de estrategia de visualización de volúmenes. La sección 3.2 muestra una serie de conceptos relacionados con los mosaicos en el plano y en el espacio que nos van a permitir definir con claridad el tipo de partición de celdas que proponemos. La siguiente sección describe brevemente como se puede abordar la extracción de isosuperficies en las representaciones discretas, tanto regulares como irregulares, que utilizan los dos principales enfoques: *voxels* y celdas (ya comentados en el capítulo anterior). Al final de esta sección se presenta la partición de celdas que proponemos para realizar la extracción.

En la sección de trabajos previos se comentan algunos de los trabajos que han abordado el problema de la extracción de isosuperficies en representaciones discretas multiresolución. Se hace especial hincapié en el problema de los agujeros que pueden aparecer en la aproximación de la isosuperficie y en las distintas técnicas empleadas para su solución.

La sección 3.4 presenta nuestra propuesta de representación híbrida: el *Octree* con rejilla dual definida implícitamente (*IDDG-Octree*). Esta representación añade al *HRB-*

Octree la topología de una rejilla dual de celdas que va a permitir realizar la extracción evitando la posibilidad de que se produzcan agujeros en la aproximación a la isosuperficie. La siguiente sección muestra el proceso de extracción de isosuperficies a partir de nuestra representación. Por último, las secciones 3.6 y 3.7 evaluan la reprentación en relación al espacio ocupado y tiempo de procesamiento empleado y a la calidad de la isosuperficie extraida, terminando con las conclusiones del capítulo.

3.1. Visualización por extracción de isosuperficies

Una vez que se dispone de una representación volumétrica, la operación básica que se desea realizar es la visualización de la información que está contenida en dicho volumen. Los dos principales problemas a resolver son: por una parte, determinar las zonas del volumen que se van a visualizar y, por otra, y quizá la más importante, es resolver el problema de la *oclusión*. Como se ha descrito en la sección 1.6 del capítulo 1, existen, en general, dos formas de abordar la visualización de volúmenes: extracción de isosuperficies y generación directa de imágenes a partir del volumen (*volume rendering*).

La extracción de isosuperficies parte de la idea de determinar el conjunto de puntos incluidos en el volumen cuyo valor de propiedad es igual a un valor umbral fijado *a priori*: *isovalor*. De aquí el término *isosuperficie*. De forma general, el volumen puede estar definido mediante expresiones matemáticas o mediante representaciones discretas de datos volumétricos. Nuestro trabajo se centra en las representaciones del segundo tipo. A continuación se describe el problema de la extracción enfocado en este tipo de representaciones.

3.1.1. Extracción de isosuperficies en representaciones volumétricas discretas

La técnica de extracción de isosuperficies aplicada a representaciones volumétricas discretas permite aproximar una isosuperficie definida a partir de la ecuación $F(x, y, z) = \nu$, en donde $F(x, y, z)$ es la función de interpolación de la representación y ν es el isovalor que determina dicha isosuperficie. La técnica proporciona puntos de la isosuperficie y, normalmente, dichos puntos se unen formando una malla de triángulos que será la aproximación final resultante. Tomando como punto de partida nuestra definición de representación volumétrica discreta se puede definir formalmente una isosuperficie de la siguiente forma:

Sea,

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

una representación volumétrica discreta.

Sea,

$$SP = \{(\boldsymbol{P}_i, v_i) : \mathcal{RV}_d \mid \forall\ (\boldsymbol{P}_i, v_i) \cdot \boldsymbol{P}_i = (x_i, y_i, z_i)\}$$

el conjunto de puntos (*Sample Positions*) en los que se sitúan las muestras de la representación volumétrica $\mathcal{RV}_d$, y sea,

$$P = \{x, y, z : \mathbb{R} \mid (x, y, z) \in Int(Conv(SP)) \cdot (x, y, z)\} \tag{3.1}$$

el conjunto de puntos interiores, $Int(\cdot)$, al *convex hull*, $Conv(\cdot)$, del conjunto finito de puntos SP. Nótese que P es un conjunto continuo y no un subconjunto del conjunto SP de posiciones de muestras.

Definición 9 (Isosuperficie) *Se define una isosuperficie con valor umbral (isovalor) ν, I_ν, sobre una representación volumétrica discreta $\mathcal{RV}_d$, como el subconjunto de puntos incluidos en P, que cumplen la condición de que su valor de propiedad es igual a ν.*

$$I_\nu = \{(x, y, z) : P \mid F(x, y, z) = \nu\} \tag{3.2}$$

siendo $F(x, y, z)$ la función de interpolación asociada a la representación volumétrica $\mathcal{RV}_d$ a partir de la cual se define el conjunto P.

En la figura 3.1 se muestra el equivalente 2-D (isocurva) a una isosuperficie 3-D. La isocurva está formada por todos los puntos que satisfacen la ecuación $f(x, y) = \nu = 15$ para la función $f(x, y) = x^2 + y^2$. Se puede observar la representación gráfica de la superficie embebida en el espacio E^3 cuyos puntos presentan la forma $(x, y, z = f(x, y))$, la cual se encuentra representada solamente en los intervalos $x, y \in [-5, 5]$. En la figura se muestra la proyección en el plano XY de la isocurva resultado de imponer la condición $f(x, y) = 15$. En la figura 3.2 se pueden observar las proyecciones de distintas isocurvas correspondientes a distintos valores de propiedad asignados al umbral ν.

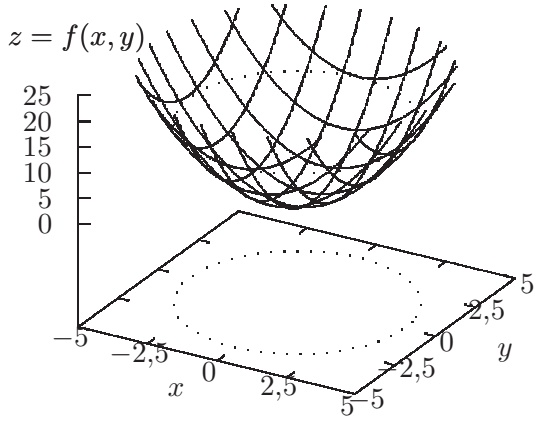

Figura 3.1: Isocurva definida por la ecuación $x^2 + y^2 = \nu = 15$ a partir de la función implícita $f(x, y) = x^2 + y^2$. La isocurva aparece proyectada en el plano XY.

Una vez definida formalmente la idea de isosuperficie, se procede a justificar brevemente el porqué de la estrategia que se adopta a la hora de calcularla. Como se puede comprobar en la definición de isosuperficie (3.2)), para obtener el conjunto de puntos que la constituyen es necesario determinar qué elementos del conjunto P cumplen la ecuación $f(x, y, z) = \nu$. Sin embargo, el conjunto P es infinito, como puede comprobarse en su definición (ver (3.1)). Por tanto, existe una alta probabilidad de que si

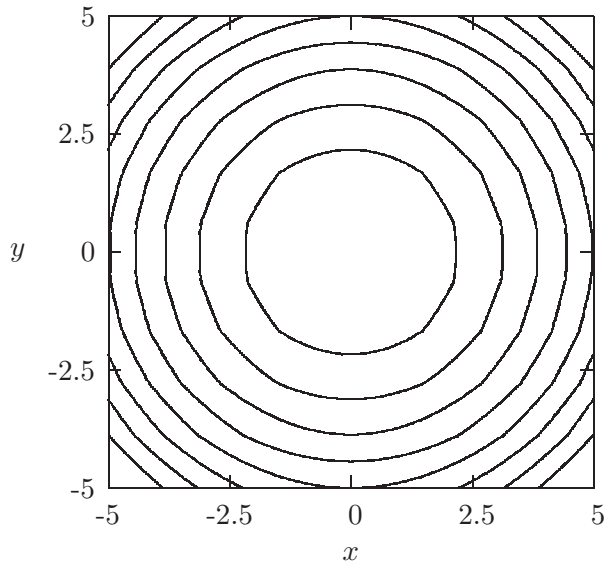

Figura 3.2: Representación gráfica de varias isocurvas obtenidas aplicando distintos valores de ν, desde 0 hasta 40 en incrementos de 5, sobre la ecuación $x^2 + y^2 = \nu$.

existe la isosuperficie definida por ν en P, ésta esté formada por un conjunto infinito de puntos. Si éste es el caso, es imposible calcular todos los puntos que pertenecen a la isosuperficie I_ν. En la figura 3.3 se muestran, para el caso 1-D, las posibilidades a la hora de determinar el conjunto de puntos que cumplen la ecuación $f(x) = \nu$. Puede verse que el conjunto puede ser vacío, finito o infinito. Esta situación se agudiza en los espacios 2-D y 3-D, caso de que exista isosuperficie (imágenes central y derecha de la figura 3.3). En el espacio 2-D, $f(x, y) = \nu$ permite determinar isocontornos en la superficie definida implícitamente mediante $(x, y, f(x, y))$ (ver figura 3.1) y, en el espacio 3-D, $f(x, y, z) = \nu$ permite determinar isosuperficies en el volumen definido implícitamente mediante $(x, y, z, f(x, y, z))$.

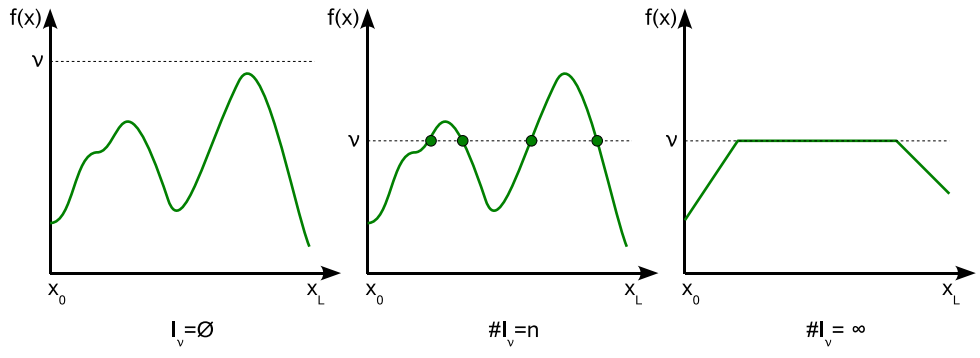

Figura 3.3: Las tres posibilidades que pueden darse a la hora de calcular las raices de una función 1-D en el intervalo (x_0, x_L).

La solución comunmente utilizada para abordar el problema pasa por *discretizar* el espacio definido por el conjunto P. Una vez fijado un conjunto de puntos que discretizan dicho espacio, se establece una partición o división que permita delimitar intervalos (caso 1-D), celdas poligonales (caso 2-D) o celdas poliédricas (caso 3-D). La elección de los

puntos de discretización puede establecerse *a priori* o de forma dinámica en función del comportamiento de la función en determinados intervalos del dominio. La discretización del espacio ocupado por una representación volumétrica discreta se basa en establecer un *mosaico o embaldosado de poliedros* de dicho espacio.

Para construir el mosaico de poliedros se puede aprovechar la distribución regular y uniforme de los datos volumétricos. De esta forma se establece una partición de celdas poliédricas cuyos vértices coinciden con las posiciones de las muestras. Una vez establecida la partición, basta con utilizar el valor de propiedad de las muestras situadas en los vértices de cada celda poliédrica y, en función de la configuración de dichos valores, determinar si la isosuperficie atraviesa la celda. Si es así, se calculan los puntos de corte de la isosuperficie con los elementos geométricos (aristas y vértices) que conforman la celda. Obviamente, a nivel de celda se presenta el mismo problema que teníamos a nivel de dominio del conjunto P. En este caso, se opta por utilizar algún tipo de aproximación para decidir cuantos puntos de la isosuperficie obtenemos en el interior de la celda.

Dependiendo de la forma poliédrica de las celdas y de la función que se utilice para estimar la isosuperficie en el interior de éstas, aparecen las cadenas poligonales que determinarán la topología de la aproximación a la isosuperficie en el interior de cada celda. La unión de todas las porciones de aproximación de isosuperficie de las distintas celdas formará como resultado la estimación final de ésta. Normalmente, cada una de las cadenas poligonales resultantes se traduce en un conjunto de triángulos que conformará la aproximación resultante. En muchos trabajos que utilizan esta estrategia se usa simplemente la información proporcionada por los puntos de corte de las aristas para establecer las cadenas poligonales y, posteriormente, la aproximación mediante triángulos.

3.2. Conceptos preliminares

En la sección anterior se ha mostrado la necesidad de discretizar el espacio ocupado por la representación volumétrica, con el objetivo de obtener una aproximación de la isosuperficie de interés. Así mismo, se ha esbozado la estrategia general para extraer una malla de triángulos que aproxime a dicha isosuperficie. A continuación se definen algunos conceptos que serán utilizados en el resto del capítulo y se explicará su relación con los dos principales enfoques utilizados para representar volúmenes de forma discreta: celdas y *voxels*, los cuales han sido presentados en el capítulo anterior.

Definición 10 (Mosaico 2-D) *Un mosaico del plano es una descomposición del mismo en regiones (teselas) poligonales que no se solapan y que cubren la totalidad del plano a descomponer.*

Definición 11 (Mosaico 2-D "arista a arista") *Un mosaico del plano arista a arista es una mosaico 2-D que cumple la restricción de que cualesquiera de los polígonos que forman el mosaico comparten, o bien un vértice o bien una arista completa o bien no comparten ni vértices ni aristas.*

En la figura 3.4 se muestra un ejemplo de mosaico 2-D (izquierda) y un ejemplo de mosaico 2-D "arista a arista". Los polígonos que forman parte de un mosaico pueden ser "congruentes", es decir, ser el mismo polígono salvo transformaciones geométricas de translación y/o rotación. En este caso, a estos polígonos que sirven de "modelo" para

las distintas teselas se les denomina *prototeselas*. Un mosaico se denomina *monoedral*, *diedral*, *triedral*, etc. según se pueda obtener una colección de 1, 2, 3, etc. prototeselas. Si T es una prototesela de algún mosaico monoedral, diremos que T embaldosa el plano.

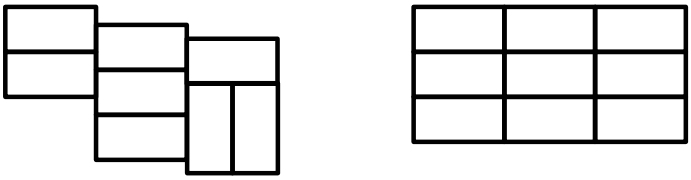

Figura 3.4: Ejemplo de mosaico 2-D monoedral (izquierda) y mosaico 2-D "arista a arista" monoedral (derecha). Ambos mosaicos están formados a partir de la misma prototesela: un rectángulo.

Definición 12 (Mosaico 2-D monoedral regular) *Un mosaico monoedral presenta estructura regular si su prototesela "modelo" tiene forma de polígono regular, en el sentido clásico de polígono regular.*

Los mosaicos 2-D regulares son muy fáciles de clasificar ya que sólo existen tres polígonos regulares que embaldosan (particionan) el plano: cuadrados, triángulos equiláteros y hexágonos regulares. En la figura 3.5 se muestran los tres tipos de mosaicos regulares 2-D.

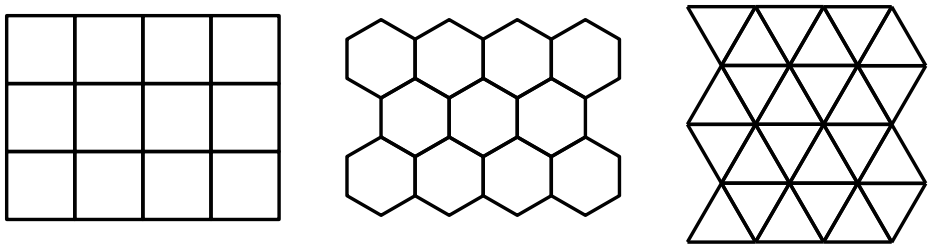

Figura 3.5: Los tres únicos tipos de mosaicos monoedrales regulares que recubren el plano.

Definición 13 (Mosaico 2-D dual) *El mosaico 2-D dual de un mosaico 2-D monoedral regular se forma tomando el centro de cada polígono como vértice del mosaico dual y uniendo los centros de los polígonos adyacentes a cada uno de los vértices de cada tesela del mosaico original.*

En la figura 3.6 se muestran los mosaicos duales 2-D correspondientes a los tres mosaicos regulares de la figura 3.5. Como puede observarse, los mosaicos basados en triángulos equiláteros y hexágonos regulares son duales entre sí, mientras que el mosaico basado en cuadrados es su propio dual.

Definición 14 (Mosaico 3-D) *Un mosaico del espacio es una descomposición del mismo en regiones (teselas) poliédricas que no se solapan y que recubren la totalidad del espacio a descomponer.*

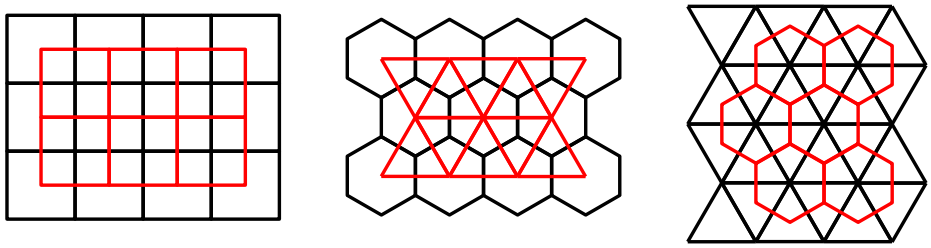

Figura 3.6: Mosaicos 2-D duales y regulares.

Definición 15 (Mosaico 3-D "arista a arista" y "cara a cara") *Un mosaico del espacio arista a arista y cara a cara es un mosaico 3-D que cumple la restricción de que cualesquiera de los poliedros que forman el mosaico comparten, o bien un vértice o bien una arista completa o bien una cara completa o bien no comparten ni vértices ni aristas ni caras.*

Definición 16 (Poliedro para recubrimiento espacial) *(del inglés* space-filling polyhedron*) Un poliedro para recubrimiento espacial es aquel que puede ser usado como prototesela para generar un mosaico 3-D "arista a arista" y "cara a cara". Nótese que el mosaico 3-D puede ser monoedral, diedral, etc.*

El único poliedro del cojunto de los poliedros regulares (o sólidos Platónicos) que incluye al hexaedro (cubo), dodecaedro, icosaedro, octaedro y tetraedro; que permite recubrir el espacio mediante un mosaico 3-D "arista a arista" y "cara a cara" es el cubo. Solamente existen cinco poliedros convexos con caras regulares que pueden recubrir el espacio mediante un mosaico 3-D monoedral "arista arista" y "cara a cara". Sin embargo, existen mosaicos 3-D de este tipo con más de una prototesela que producen un recubrimiento del espacio, como por ejemplo el mosaico 3-D diedral cuyas prototeselas son el tetraedro y el octaedro. En la parte superior de la figura 3.7 se muestran los cinco poliedros regulares o Sólidos Platónicos y en la parte inferior los cinco poliedros convexos con caras regulares que pueden recubrir el espacio formando mosaicos 3-D monoedrales.

Definición 17 (Mosaico 3-D monoedral dual) *El mosaico 3-D dual de un mosaico 3-D monoedral "arista a arista" y "cara a cara" se forma tomando el centro de cada poliedro como vértice del mosaico dual y uniendo los centros de los poliedros adyacentes a cada uno de los vértices de cada tesela del mosaico original.*

Se va a generalizar la definición de mosaico 3-D monoedral dual para el caso en el que el mosaico 3-D monoedral de partida no sea "arista a arista" y "cara a cara". Esta extensión permite definir los mosaicos 3-D duales sobre los mosaicos 3-D asociados a representaciones volumétricas irregulares. De esta forma, se pueden caracterizar completamente los mosaicos 3-D y sus correspondientes duales asociados a nuestra definición de representación volumétrica discreta. De hecho, la única restricción que añadimos para esta extensión es considerar incluida la transformación de escalado en el concepto de congruencia de las teselas.

Definición 18 (Mosaico 3-D dual) *El mosaico 3-D dual de un mosaico 3-D se forma tomando el centro de cada poliedro como vértice del mosaico dual y uniendo los*

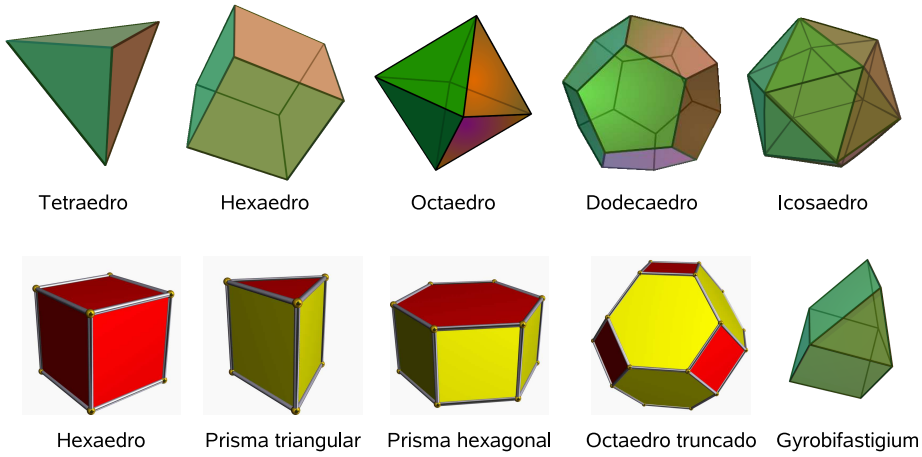

Figura 3.7: La figura muestra los Sólidos Platónicos (parte superior) y los únicos poliedros convexos con caras regulares que permiten formar cinco mosaicos 3-D "arista a arista" y "cara a cara" monoedrales distintos (parte inferior).

centros de los poliedros adyacentes a cada uno de los vértices de cada tesela del mosaico original.

Utilizando las definiciones previas se van a caracterizar los tipos de mosaicos que vienen determinados por los distintos enfoques que se utilizan comunmente para representar conjuntos de datos volumétricos. En primer lugar, podemos observar que los dos enfoques de representación discreta, uniforme y regular de datos volumétricos: *voxels* y celdas, generan un mosaico 3-D arista a arista y cara a cara que recubre todo el espacio de la representación. El poliedro para recubrimiento espacial (prototesela) usado es el cubo, caso de que el conjunto de datos volumétrico sea isotrópico, o un paralelepípedo, caso de que dicho conjunto sea anisotrópico. Además, dichos enfoques forman mosaicos 3-D arista a arista y cara a cara que son duales entre sí. En la imagen situada en la parte izquierda de la figura 3.6 puede observarse el análogo para el caso 2-D, en el cual, el mosaico 2-D determinado por el enfoque de *voxels* aparece en color negro y el mosaico 2-D determinado por el enfoque de celdas aparece en color rojo. La prototesela de ambos mosaicos de ejemplo es el mismo cuadrado. Se puede observar que el *voxel* es el elemento dual de la celda y viceversa.

Por otra parte, las representaciones discretas irregulares, las cuales pueden utilizar un *octree* para representar los poliedros de diferente tamaño como se vio en el capítulo anterior, proporcionan un mosaico 3-D sin restricciones "arista a arista" y "cara a cara". Las teselas poliédricas de este mosaico son "congruentes", en el sentido de que son el mismo poliedro salvo transformaciones geométricas de traslación y escalado. La restricción de subdivisión espacial binaria del *octree* provoca que la transformación de escalado siempre sea potencia de dos. La irregularidad soportada por esta representación provoca que las teselas del mosaico dual asociado a las representaciones de los dos principales enfoques, *voxels* y celdas, no respondan a una única prototesela. La variedad de teselas del mosaico dual depende de las distintas configuraciones de adyacencia de las teselas representadas en el *octree*. En la figura 3.8 se presenta un ejemplo en el que puede verse que en el caso de las representaciones regulares, el mosaico dual (en rojo) presenta una única prototelela, mientras que en el caso de las representaciones irregulares el mosaico dual presenta una mayor variedad de prototeselas.

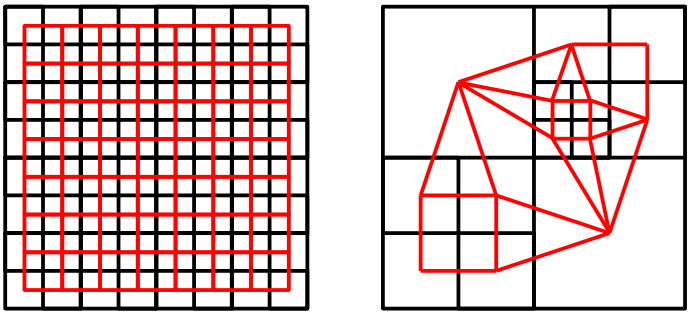

Figura 3.8: Mosaico 2-D regular y su dual (en rojo) correspondiente a una representación regular (izquierda) y, mosaico 2-D y su dual determinados por la subdivisión espacial proporcionada por un *quadtree* (derecha).

3.3. Extracción de isosuperficies en el enfoque de celdas y en el de *voxels*

Como se ha comentado en el capítulo anterior, los principales enfoques para representar de forma discreta objetos que incluyan características de volumen son el de *voxels* y el de celdas. A continuación, se estudian las posibilidades, ventajas e inconvenientes derivados de la extracción de isosuperficies a partir de representaciones basadas en cada enfoque, teniendo en cuenta las definiciones previas y la necesaria discretización espacial. En primer lugar se consideran ambos enfoques en el caso de representaciones regulares, y posteriormente, se consideran las representaciones irregurales, junto con nuestra representación (*HRB-Octree*). En aras de simplificar el estudio se va a suponer, sin perdida de generalidad, que el conjunto de datos de partida es isotrópico.

3.3.1. Extracción de isosuperficies sobre representaciones regulares

El enfoque de celdas permite aprovechar de forma directa la estructura regular de las muestras y la función de interpolación trilineal. Ambas características determinan un mosaico 3-D monoedral regular cuya prototesela es un cubo. Cada tesela configura una celda para la extracción de isosuperficies, con un valor de propiedad situado en cada vértice. De esta forma, se obtiene una partición espacial de celdas de todo el espacio ocupado por la representación volumétrica y, por tanto, se puede llevar a cabo directamente la extracción, siguiendo un método de recorrido de celdas y calculando las intersecciones de las aristas de las teselas con la isosuperficie. Las intersecciones con las aristas de corte se calculan mediante interpolación lineal.

El enfoque de *voxels* determina un mosaico 3-D monoedral regular cuya prototesela es un cubo. Sin embargo, a la hora de extraer la isosuperficie las "celdas" cúbicas presentan el mismo valor de propiedad en todo su volumen, por lo que se presentan los problemas analizados en el capítulo anterior y no se puede extraer una isosuperficie continua. La aproximación que se puede llevar a cabo se basa en visualizar las caras de los *voxels* cuyo valor de propiedad caiga por debajo (o por encima) del isovalor establecido para la extracción. Una estrategia puede ser visualizar aquellas caras compartidas por dos *voxels*, uno con valor de propiedad mayor que el isovalor y otro con valor de propiedad menor que éste.

3.3.2. Extracción de isosuperficies sobre representaciones irregulares

El enfoque de celdas sobre representaciones irregulares determina un mosaico 3-D monoedral en el que las teselas son "congruentes" con una prototesela cúbica salvo transformaciones de traslación y escalado. En este caso, cada tesela proporciona una celda de tamaño variable sobre la que se puede realizar la extracción de la isosuperficie. El problema se produce cuando se extraen parches de isosuperficie en celdas vecinas de distinto tamaño. En este caso se pueden producir agujeros en la isosuperfice final.

El enfoque de *voxels* aplicado a representaciones irregulares determina un mosaico 3-D monoedral análogo al proporcionado por el enfoque de celdas. Sin embargo, este enfoque adolece del mismo problema que se plantea sobre las representaciones regulares.

3.3.3. Extracción de isosuperficies sobre el *HRB-Octree*

El *HRB-Octree* establece una partición de las regiones homogéneas (*voxels*) representadas en un *octree* dividiendo cada una de ellas en dos zonas: interior y borde. Como vimos en el capítulo anterior, determinar el valor de propiedad en un punto del volumen usando esta representación es un proceso costoso. A la hora de la extracción de una isosuperficie la situación se agrava debido a que no se dispone de una estructura de celdas sobre la que realizar extracción, con lo que habría que generar dicha estructura adicionalmente, con el consiguiente incremento en los requisitos de espacio. En la figura 3.9 se muestra gráficamente la estructura de celdas, en color rojo, necesaria para llevar a cabo la extracción de isosuperficies sobre un ejemplo de *HRB-Octree*.

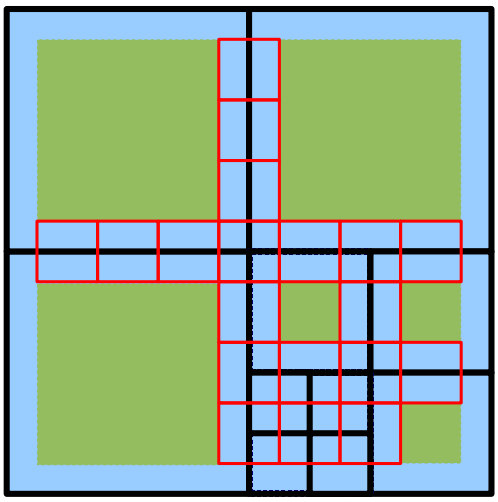

Figura 3.9: *HRB-Octree* mostrando las zonas interior (en verde) y borde (en azul), junto con las celdas requeridas para la extracción de isosuperficies (en rojo).

Nuestra propuesta de solución hace uso de la idea de mosaico 3-D dual al mosaico 3-D determinado por el *HRB-Octree*. La nueva representación extiende el *HRB-Octree* para incluir un mosaico 3-D dual al que determinan las regiones homogéneas, pero sin pagar el coste de mantener los dos mosaicos, e.d. una representación del dominio de propiedades más un mosaico dual de celdas para realizar la extracción. Para ello, la representación define implícitamente el mosaico 3-D dual de celdas para realizar extracción. En la figura 3.10 se muestra el mosaico 3-D dual de celdas para el mismo ejemplo de *HRB-Octree* de la figura 3.9.

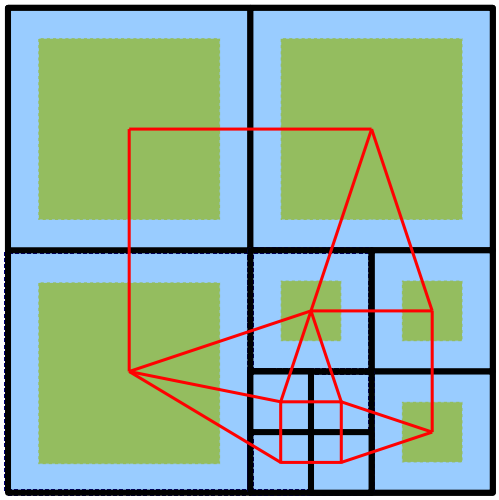

Figura 3.10: *HRB-Octree* mostrando las zonas interior (en verde) y borde (en azul), junto con las celdas duales requeridas para la extracción de isosuperficies (en rojo).

Como puede comprobarse visualmente, la forma y tamaño de las celdas utilizadas para la extración en ambas figuras varía. Por tanto, el cálculo de los puntos de la isosuperficie realizado a partir del mosaico 3-D dual será una aproximación de los puntos de isosuperficie calculados sobre las celdas para extracción mostradas en la figura 3.9. A continuación se muestran algunos trabajos que han realizado extracción de isosuperficies sobre representaciones irregulares, han abordado el problema de los agujeros y han utilizado de alguna forma el concepto de dualidad.

3.3.4. Trabajos previos

Como se ha visto en el capítulo previo, las representaciones discretas irregulares permiten ahorrar espacio de almacenamiento de muestras en zonas del volumen sobre-muestreadas. Se planteó que el principal problema de las representaciones irregulares basadas en el enfoque de celdas era la inconsistencia a la hora de calcular los valores de propiedad en aristas y caras comunes entre celdas de distinto tamaño. A esta situación se hace alusión en la bibliografía con el nombre de problema de fisuras (*crack problem*.

Para clarificar la exposición del problema se va a hacer uso de la figura 3.11. La figura muestra dos celdas adyacentes del mismo tamaño, C_1 y C_2 cuyas caras F_1 y F_2 son adyacentes. Las celdas se han separado para poder distinguir la caras vecinas ya que, en realidad F_1 y F_2 representan la cara compartida F. Además, se puede observar que la celda C_1 se encuentra subdividida en celdas de la mitad de tamaño que la celda C_2. Los valores de los vértices marcados con un círculo negro se encuentran por debajo del valor umbral y los vértices marcados con un círculo blanco tienen un valor de propiedad por encima del umbral. Las líneas AB y BC representan las aristas de intersección de la aproximación triangular a la isosuperficie obtenida para la celda C_1 que caen sobre la cara F_1, mientras que la línea PQ representa la arista de intersección sobre la cara F_2 de la aproximación triangular obtenida para la celda C_2.

Si se unen mentalmente las celdas C_1 y C_2, entonces puede comprobarse que la línea roja que une los puntos $\boldsymbol{A}$ y $\boldsymbol{B}$ es el lugar donde la línea PQ cae sobre la cara F_1 y viceversa, la polilínea roja que une los puntos $\boldsymbol{P}$ y $\boldsymbol{Q}$ es el lugar donde las líneas AB y

BC caen sobre la cara F_2. La fisura queda formada por la región triangular delimitada por las líneas rojas que caen sobre la cara F.

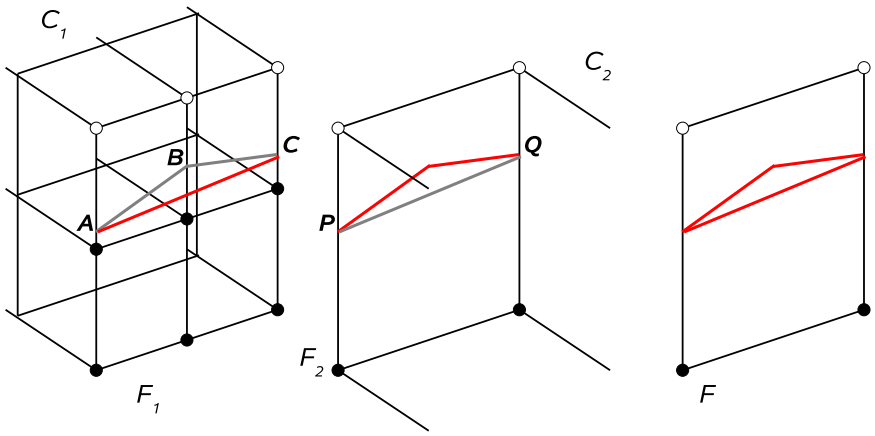

Figura 3.11: Ilustración del problema de las fisuras en la isosuperficie. Este fenómeno se produce a la hora de realizar la extracción entre celdas adyacentes de distinto tamaño.

Las distintas soluciones al problema de los agujeros en la isosuperficie extraida a partir de representaciones multiresolución (irregulares) puden clasificarse según dos estrategias generales: las que evitan la aparición de agujeros [FPRJ00, JLSW02, VTC02, VKKM03] y las que permiten que aparezcan agujeros y posteriormente los rellenan con triángulos [MS93, SZK95, SFYC96]. Además, se ha utilizado una estrategia diferente que genera isosuperficies duales [Gib98, Nie04, ZHK04] a las que son generadas utilizando los algoritmos clásicos seguidores de *Marching Cubes*. Los vértices de cada triángulo de una superficie dual se encuentran situados en el interior de las celdas, en lugar de estar localizados en las aristas de las celdas. En el caso de las superficies duales cada triángulo es atravesado por una arista de celda y cada celda tiene un único vértice asociado, por lo que no se produce ningún agujero en la isosuperficie extraida [KBSS01]. En la figura 3.12 se muestra un ejemplo 2-D de este tipo de superficies. Los círculos negros y blancos representan valores de propiedad por encima y por debajo del umbral respectivamente. Los círculos rojos representan los puntos de corte en el caso de la extracción de un isocontorno usando el método clásico y los puntos interiores a las celdas en el caso de usar un método de extracción de isosuperficie dual.

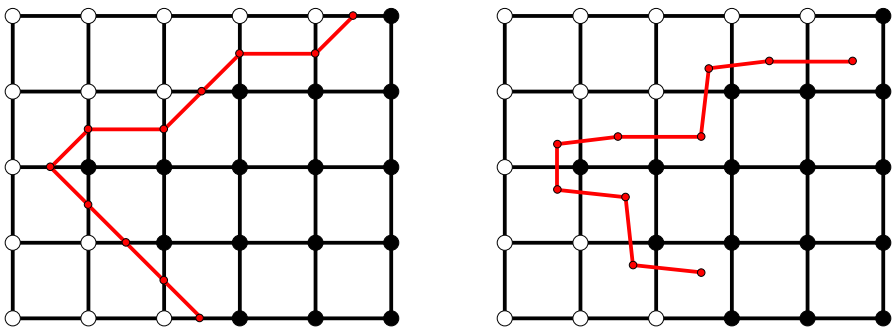

Figura 3.12: Ejemplo 2-D que muestra las diferentes aproximaciones al contorno definido por un mismo isovalor mediante un enfoque tradicional (izquierda) y mediante un enfoque de extracción de contornos duales (derecha).

Se han realizado varias propuestas para generar representaciones multiresolución. Frisken y Perry [FPRJ00, PF01] usan campos de distancias que muestrean de forma adaptativa un objeto, obteniendo de esta forma un mayor ratio de muestreo en zonas en donde existen más detalles y uno menor en zonas más homogéneas. Esta información de muestreo se almacena en un *octree*, lo cual permite procesar las muestras de forma eficiente de cara a la extracción. Ju y otros [JLSW02] propusieron una representación basada en un *octree* cuyos nodos hoja representan celdas. Cada nodo mantiene información relativa a los signos de los vértices de la celda, clasificados con respecto al umbral en negativos y positivos, junto con los puntos de corte y las normales asociadas a cada arista que muestre un cambio de signo. Para extraer la isosuperficie utilizan el método de Kobbelt [KBSS01] sobre todas las celdas, en lugar de discriminar entre celdas teniendo que calcular si existen detalles en el interior. Es decir, Ju calcula el minimizador de la función de error cuadrático para todas las celdas que exhiben un cambio de signo, obteniendo como resultado la posición del vértice interior a la celda. Antes de realizar la extracción y de cara a simplificar el *octree*, construyen una función de error cuadrático para cada nodo hoja heterogéneo y agregan nodos hoja vecinos siempre que la suma de sus funciones de error cuadrático sea menor que una tolerancia de error predefinida. Zhang y otros [ZBS03] mejoran el método de Ju añadiendo una etapa de preprocesamiento que calcula valores de propiedad máximos y mínimos para cada celda del *octree* con el objetivo de visitar solamente las celdas activas para un umbral determinado, calculando solamente los valores de la función de error cuadrático en estas celdas.

Schaefer [SW05] propuso usar un *octree* que muestrea adaptativamente la función implícita que representa la isosuperficie que deseamos extraer, almacenando una celda por cada nodo hoja. Para extraer la isosuperficie usa una rejilla dual al *octree* cuyos vértices se obtienen minimizando una función de error cuadrático aplicada sobre las celdas representadas en los nodos hoja. El uso de la rejilla dual garantiza que la isosuperficie obtenida no presentará el problema de los agujeros. Este enfoque es independiente del isovalor que se escoja. Es decir, no es necesario generar un nuevo *octree* cuando se cambia el isovalor. Esta es una ventaja con respecto a otros métodos basados en muestreo adaptativo ya que, por ejemplo, los métodos de Ju [JLSW02] y Zhang [ZBS03] almacenan información asociada a la isosuperficie a extraer y, el método de Frisken [FPRJ00] utiliza un campo de distancias adaptativo que muestrea la isosuperficie a extraer. El principal inconveniente del enfoque de Schaefer estriba en que la rejilla dual se almacena explícitamente, incrementando de esta forma el espacio requerido para la representación. De esta forma, es difícil que pueda utilizarse para realizar manipulaciones de la representación, como por ejemplo es el caso en aplicaciones de escultura virtual, ya que hay que recalcular completamente la rejilla dual cada vez que se modifique el *octree*.

Los trabajos previos han propuesto representaciones que o bien reducen el espacio de almacenamiento requerido (consultar la sección 2.3 del capítulo 2), incrementando de esta forma el tiempo de ejecución, o reducen el tiempo de procesamiento incrementando el espacio de almacenamiento usado (estrategias basadas en una estructura adicional a la rejilla de muestras que permite identificar las celdas activas). Nuestro objetivo es proponer una nueva representación de volumen que reduce los requisitos de memoria para la representación de volumenes de alta resolución, a la vez que permite la generación razonablemente rápida de isosuperficies con una calidad similar a las que se extraen utilizando el enfoque clásico de marchar sobre una rejilla regular y uniforme.

3.4. *Octree* de regiones homogéneas con celdas duales definidas implícitamente: *IDDG-Octree*

De cara a la extracción de isosuperficies, se va a establecer un método aproximado que permita realizar el cálculo de puntos de la isosuperficie sin tener que pagar el coste asociado a mantener una estructura de celdas adicional. Para ello se va a utilizar un *mosaico 3-D de celdas dual* al mosaico 3-D definido por las regiones homogéneas (voxels) representadas por los nodos hoja del *octree*, el cual va a estar definido implícitamente en la representación. De esta forma, se mantienen las ventajas en cuanto a ahorro de almacenamiento e independencia de las regiones homogéneas vecinas del enfoque de *voxels*, y al mismo tiempo, evitamos la pérdida de eficiencia en el cálculo del valor de propiedad de un punto del volumen.

El mosaico de celdas dual de un mosaico de *voxels* de diferente tamaño tiene situados sus vértices en los centros de los *voxels*. Cada celda dual engloba un vértice de *voxel* y su topología viene determinada en función de los *voxels* adyacentes que comparten el vértice englogado. Las teselas del mosaico dual constituyen las celdas que cubren el espacio ocupado por la representación y tienen asociado, en cada uno de sus vértices, el valor de propiedad almacenado en el *voxel* correspondiente del *octree*.

Las celdas del mosaico 3-D dual construido de esta forma presentan una gran variedad de formas y tamaños. Esto se debe a las posibles combinaciones de relaciones de adyacencia junto con la variabilidad de tamaños de los *voxels*) adyacentes a cada vértice del mosaico original. De hecho, no podemos considerar el mosaico 3-D dual resultante como un mosaico de poliedros con caras planas propiamente dicho. Aunque las celdas generadas pueden tener forma de poliedros planares, también existen casos en los que la topología de la celda dual, entendida como relaciones de conectividad de los vértices de dicha celda, produce celdas delimitadas por cuadriláteros no coplanares. En el caso de que los ocho *voxels* adyacentes a un vértice del mosaico original tengan el mismo tamaño se obtiene una celda cúbica dual que engloba al vértice compartido. Sin embargo, cuando los tamaños varían se produce una deformación de dicha celda cúbica dual. Estas deformaciones pueden venir dadas por:

- Traslación de uno o más vértices en el espacio.
- Degeneración de una o varias aristas debido a que los *voxels* adyacentes que formarían la conectividad de la arista han sido agregados en un *voxel* de mayor tamaño. Por consiguiente, esta arista desaparece y los dos vértices que habrían formado sus extremos coinciden en un único vértice geométrico.
- Degeneración de una cara debido a que los cuatro vértices que la habrían formado coinciden en un mismo vértice geométrico. De nuevo, esto se debe a que existe un *voxel* de mayor tamaño resultado de la agregación de los cuatro *voxels* cuyos centros hubiesen formado la cara caso de no haber sido agregados.
- Combinaciones de los tres escenarios anteriores.

En la figura 3.13 podemos ver ejemplos de algunas de las deformaciones que pueden producirse sobre una celda cúbica inicial construida en base a ocho *voxels* adyacentes del mismo tamaño que comparten un vértice. En la imagen situada más a la izquierda podemos ver la deformación producida por la traslación del vértice v_7. Esto se debe a que el *voxel*, en cuyo centro está situado dicho vértice, tiene mayor tamaño que el resto

de *voxels* cuyos centros constituyen el resto de vértices de la celda dual. En las imagenes centrales podemos ver dos ejemplos de celdas duales resultado de la degeneración de aristas. Esta degeneración viene provocada por una situación en la que dos *voxels* del mismo tamaño, cuyos centros son por ejemplo los vértices v_5 y v_7, han sido agregados a uno de mayor tamaño. Por tanto, el vértice de la celda dual se corresponde ahora con el centro del *voxel* de mayor tamaño. En la imagen situada más a la derecha se puede observar la celda dual que resulta de la agregación de los cuatro *voxels* del mismo tamaño cuyos centros correspondían a los vértices v_1, v_3, v_5 y v_7. El nuevo vértice es el centro del *voxel* de mayor tamaño resultado de la agregación.

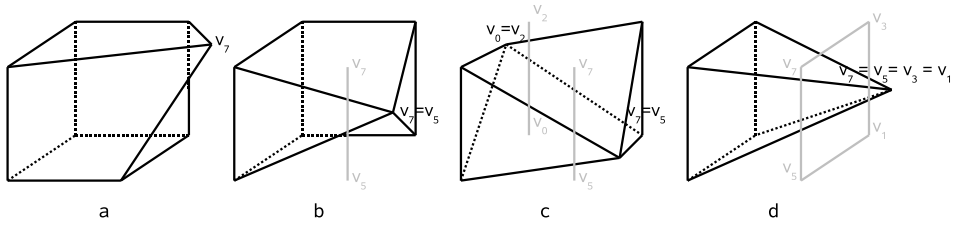

Figura 3.13: Ejemplos de deformación de una celda dual cúbica: por desplazamiento de un vértice (a), por degeneración de una arista (b), por degeneración de dos aristas (c) y por degeneración de una cara (d).

A pesar de que se puedan producir caras no planares en las celdas duales, la isosuperficie extraida a partir de éstas no presenta el problema de los agujeros. Esto se debe a que el cálculo de los puntos de corte en cada arista con cambio de signo es consistente en cada una de las celdas que la comparten. La figura 3.14 muestra un ejemplo de esta situación. Pueden verse dos celdas, separadas por cuestiones de claridad, en las que aparecen las caras no planares F_1 y F_2 asociadas a cada una de ellas, aunque dichas caras constituyen en realidad una única cara compartida. En la figura se muestran con círculos en color negro y blanco los vértices etiquetados que caen por encima y por debajo del valor umbral respectivamente. Los círculos rojos muestran los puntos de corte con las aristas que presentan un cambio de signo, y la arista resaltada en rojo es la arista de la aproximación triangular resultado de la unión de los dos puntos de corte de la cara compartida. Nótese la consistencia de dicha arista en ambas celdas. Las aristas coloreadas en gris constituyen el resto de aristas de la aproximación triangular que produce cada celda en el resto de sus caras.

Por tanto, el mosaico 3-D dual que se obtiene está constituido por celdas con formas irregulares. Este mosaico de celdas debe ocupar todo el espacio ocupado por la representación, con el objetivo de poder utilizar la estrategia de visualización por extracción de isosuperficies. Sin embargo, debido a la finitud subyacente a cualquier representación computacional, los vértices de los *voxels* que caen sobre los vértices, aristas o caras de la caja englobante del *octree* carecen del suficiente número de *voxels* adyacentes para completar su correspondiente celda dual. Denominamos a este tipo de vértices *vértices frontera* en contraposición a los vértices situados en el interior de la caja englobante. Si no se completa la construcción del mosaico 3-D dual con celdas que engloben dichos vértices frontera, entonces no se consigue cubrir completamente todo el espacio representado por los nodos hoja del *octree*. Esta situación se muestra en la figura 3.15.

En la imagen izquierda se muestra un mosaico 2-D poligonal y dual (rojo) a las teselas cúbicas de tamaño variable representadas por un *quadtree* de ejemplo. En la imagen derecha se muestran el resto de celdas duales (en línea discontinua) que cubren

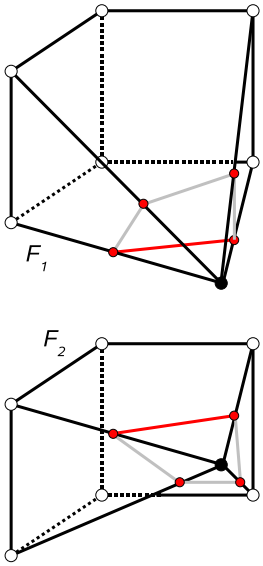

Figura 3.14: Ejemplo que muestra la consistencia en el cálculo de aristas de la aproximación triangular en caras no planares compartidas. F_1 y F_2 son la misma cara no planar compartida por dos celdas.

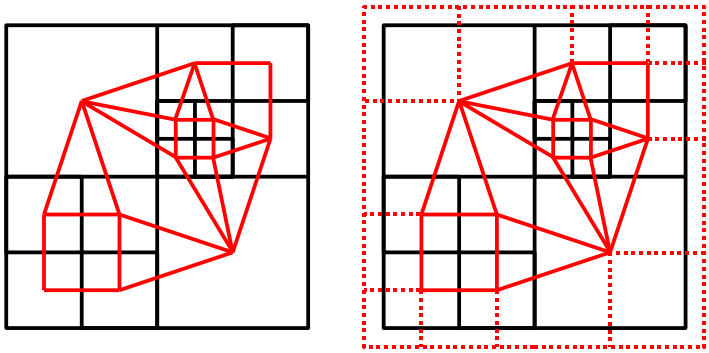

Figura 3.15: Mosaico 2-D dual del mosaico de teselas cuadradas representadas por un *quadtree* englobando solamente a los vértices interiores (izquierda). En la imagen de la derecha se muestra el mosaico 2-D dual completo para el mismo *quadtree*.

completamente el espacio ocupado por el *quadtree*. Estas celdas deben construirse de forma especial, ya que es necesario proporcionar de alguna forma el resto de *voxels* que no aporta el *quadtree*.

Cualquier conjunto de *voxels* que englobe completamente la frontera del *octree* proporciona un mosaico 3-D dual de celdas que ocupa todo el volumen. La elección del tamaño de dichos *voxels* solo influye en la forma y tamaño de las celdas extra generadas. En la figura 3.16 se muestran dos ejemplos de mosaicos de celdas duales que cubren completamente el espacio de la representación. Los mosaicos se diferencian únicamente en las celdas extra generadas debido a la diferente elección del conjunto de *voxels* que ocupan el espacio exterior.

Sin embargo, esta estrategia de añadir un conjunto de *voxels* que envuelvan la caja englobante provoca que el número de niveles del *octree* aumente. Esto no interesa de cara a ahorrar tanto en espacio de representación como en tiempo de procesamiento

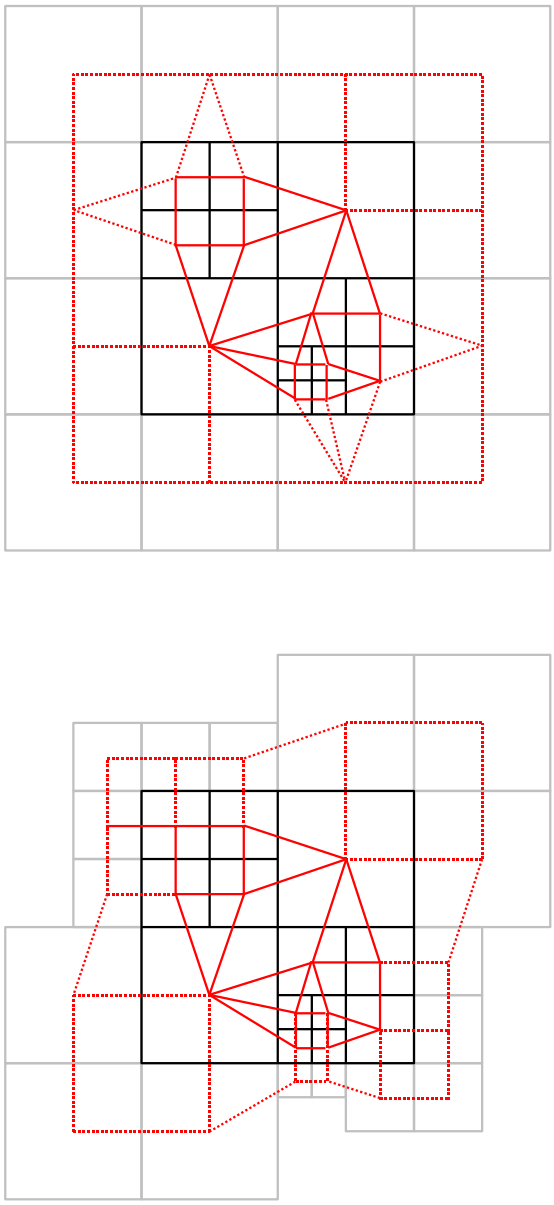

Figura 3.16: Dos ejemplos de mosaicos duales de celdas que ocupan completamente el espacio representado en un *quadtree* de ejemplo.

de celdas. Nuestra propuesta de solución evita tener que almacenar *voxels* extra, construyendo estas "celdas exteriores" de forma especial. Además, nuestro procedimiento para la construcción de celdas duales "exteriores" garantiza que siempre presenten una forma poliédrica, a diferencia de lo que puede ocurrir, como ya se ha comentado, con las celdas asociadas a los vértices interiores de los *voxels*. Como puede observarse en la imagen de la derecha de la figura 3.15, todos los vértices de celdas duales situados en el exterior de la caja englobante del *quadtree* se encuentran localizados sobre una "caja englobante exterior". Esto es debido a que en la construcción de las celdas, cada uno de los "vértices exteriores" de éstas se situa a una distancia de la caja englobante igual a la mitad de la distancia entre muestras de máxima resolución en cada una de las dimensiones. Hemos elegido este tipo de construcción para las celdas exteriores ya que, como

se verá más adelante, es suficiente para poder aplicar el tipo especial de interpolación que utilizamos para extraer isosuperficies sobre el mosaico de celdas duales.

Cualquier celda exterior está compuesta por vértices situados en los centros de *voxels* que tienen algún vértice y/o arista y/o cara situado sobre la caja englobante, y por vértices que es necesario construir y que están situados en el exterior de dicha caja englobante. Para construir una celda exterior es necesario calcular los vértices que no vienen dados como los centros de los *voxels* situados en la frontera del *octree*. El número de vértices exteriores asociados a una determinada celda dual viene determinado por el tipo de vértice frontera que engloba dicha celda. Como puede comprobarse en la figura 3.17, sólo existen cuatro tipos de vértices frontera en un *octree*. Estos cuatro tipos se clasifican en función del número de *voxels* adyacentes. Usaremos la notación v es un i-vértice, $i = 1, \ldots, 4$ para denotar que el vértice exterior v tiene i *voxels* adyacentes. En la figura 3.17 se muestran ejemplos de los cuatro tipos de vértices frontera rodeados por un círculo azul.

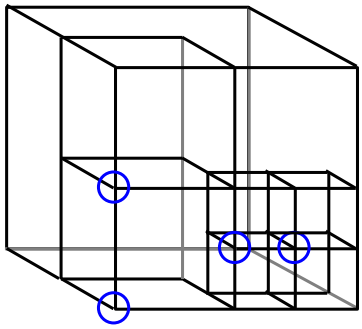

Figura 3.17: La figura muestra los cuatro tipos de vértices frontera que pueden existir en un *octree*.

En función del tipo de vértice exterior y de su situación con respecto a los vértices, aristas y caras de la caja englobante del *octree* se calculan los vértices exteriores de la celda dual de la siguiente forma:

1**-vértice** Solo hay ocho 1-vértices, los cuales coinciden con los ocho vértices de la caja englobante. El único vértice de la celda dual asociada a este tipo de vértice exterior que viene fijado por el *octree* es el que se situa en el centro del único *voxel* que incide en dicho vértice. El resto de vértices se calculan teniendo en cuenta la localización de este vértice en la caja englobante para elegir de forma adecuada los signos en la siguiente expresión:

$$\begin{aligned} v_i = (&x_C \pm 2^{n-2}\Delta_x \pm \tfrac{\Delta_x}{2}, \\ &y_C \pm 2^{n-2}\Delta_y \pm \tfrac{\Delta_y}{2}, \\ &z_C \pm 2^{n-2}\Delta_z \pm \tfrac{\Delta_z}{2}), \quad i = 0, \ldots, 6 \end{aligned} \tag{3.3}$$

siendo $v_i,\ i = 0, \ldots, 6$ cada uno de los siete vértices restantes de la celda dual, (x_C, y_C, z_C) las coordenadas del centro del *voxel*, y $\Delta_x, \Delta_y, \Delta_z$ las distancias entre muestras de máxima resolución en cada una de las dimensiones.

2**-vértice** Los vértices de este tipo se encuentran situados en las aristas de la caja englobante. Los vértices de la celda dual resultante fijados por el *octree* coinciden

con los centros de los dos *voxels* que inciden en el 2-vértice. El resto de vértices de la celda se calculan teniendo en cuenta la localización de ambos vértices en una de las doce aristas de la caja englobante y utilizando la expresión (3.3). En este caso, $i = 0, \ldots, 5$.

3-vértice Los 3-vértices se situan en las caras de la caja englobante. Los vértices de la celda dual fijados por el *octree* coinciden con los centros de los tres *voxels* que inciden en el 3-vértice. En este caso, la celda dual presenta una degeneración de arista. El resto de vértices se calculan teniendo en cuenta en cual de las seis caras de la caja englobante se encuentra situado dicho vértice, y utilizando las expresión (3.3) para $i = 0, 1, 2$.

4-vértice Igualmente que en el caso anterior, estos vértices se encuentran situados en las caras de la caja englobante. Los vértices de la celda fijados por el *octree* coinciden con los centros de los cuatro *voxels* que inciden en el 4-vértice. Los cuatro vértices restantes se calculan teniendo en cuenta la cara que contiene al 4-vértice y utilizando las expresión (3.3) para $i = 0, 1, 2, 3$.

La topología de la celda dual a cualquier i-vértice es independiente del tamaño de los *voxels* que inciden en el vértice exterior. Sin embargo, la geometría de la celda sí viene determinada por el tamaño de éstos, ya que los vértices vienen definidos por la posición de sus centros. En la figura 3.18 se muestran ejemplos de los cuatro tipos de i-vértices que se pueden encontrar. De izquierda a derecha y de arriba abajo se muestran ejemplos de 1-vértice, 2-vértice, 3-vértice y 4-vértice, junto con sus respectivas celdas duales asociadas.

A la hora de realizar la extracción de la isosuperficie, estas celdas exteriores que permiten englobar el resto de vértices de los *voxels* son tratadas de forma especial, no siendo necesario mantenerlas almacenadas de forma implícita en la representación.

3.4.1. Construcción del mosaico dual de celdas

Una vez que se dispone del mosaico 3-D dual de celdas se puede calcular, mediante interpolación lineal en las aristas de tales celdas, la malla de triángulos que aproxima a la isosuperficie que se desea extraer. En nuestra representación las celdas varían tanto en tamaño como en forma. Por tanto, necesitamos definir perfectamente la geometría de cada celda. Esto requiere consumir espacio extra. Para evitarlo, se ha desarrollado un método que permite almacenar implícitamente cada una de las celdas que componen el mosaico 3-D dual, sin tener que recurrir a almacenar una estructura geométrica adicional.

La idea fundamental que se persigue es asociar la topología de las celdas duales a los *voxels* representados en el *octree*. De esta forma se consiguen dos objetivos principales, aparte del consiguiente ahorro de espacio de representación:

- Recorrer todo el mosaico dual de celdas recorriendo el *octree*, para de esta forma poder procesar todas las celdas de cara a extraer cualquier isosuperficie.

- Mantener solamente la topología de las celdas duales y, poder definir la geometría de la celda cuando sea necesario, en base a las relaciones de adyacencia entre *voxels*.

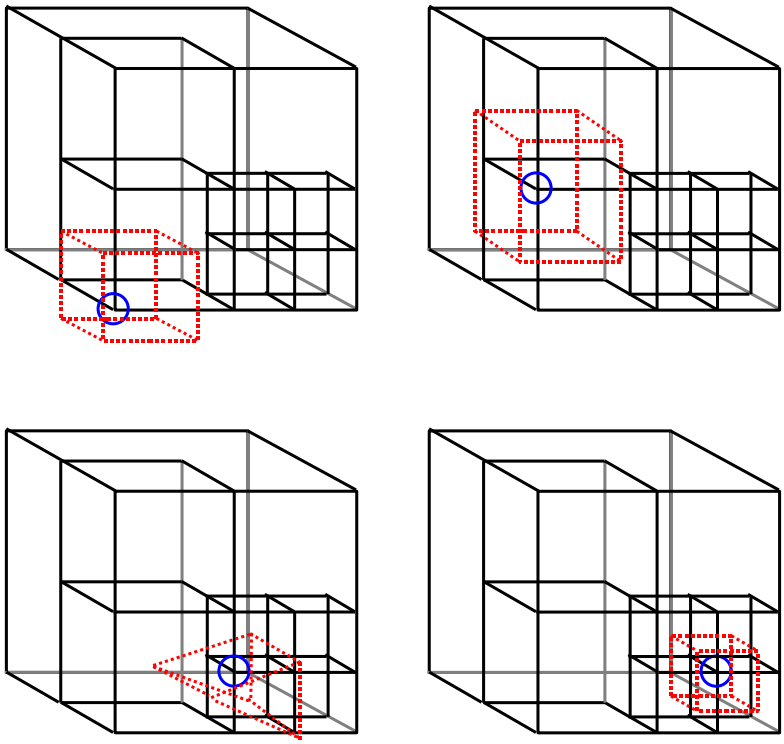

Figura 3.18: Cuatro ejemplos de celdas duales exteriores que permiten completar la ocupación del volumen representado por el *octree*. Cada celda se corresponde con uno de los cuatro tipos de vértices exteriores que existen en un *octree*.

Como se ha comentado anteriormente, el centro de cada *voxel* constituye un vértice del mosaico dual de celdas, y cada celda dual engloba uno de los vértices de la partición de *voxels*. Si consideramos el mosaico original como el formado por la rejilla regular y uniforme de *voxels*, cada una de las celdas duales tiene sus vértices situados en los ocho *voxels* adyacentes que comparten el vértice englobado por la celda dual. Este hecho se ilustra gráficamente en la imagen izquierda de la figura 3.19 para el caso 2-D. Los centros de los cuatro *voxels* adyacentes sombreados forman la celda dual que engloba el vértice compartido rodeado por un círculo azul. Por tanto, podemos asociar la topología de la celda dual a cualquiera de los *voxels* adyacentes. Desde cualquiera de ellos, es posible recuperar la topología de la celda dual teniendo en cuenta la posición relativa del vértice compartido con respecto al *voxel* elegido. Esta posición relativa viene indicada en la ilustración por las flechas rojas.

En el caso de la rejilla regular, es bastante fácil asociar la topología de cada celda dual de forma que todas las celdas puedan ser procesadas recorriendo los *voxels* de la rejilla. En la imagen de la derecha de la figura 3.19 se muestra una asignación de la topología del mosaico dual que permite procesar todas las celdas, visitando todos los *voxels* y construyendo la celda en el caso de que el *voxel* que se visita tenga asociada la topología de una celda del dual.

Cada *voxel* puede tener asociada la topología de más de una celda dual, ya que su centro es compartido por ocho celdas del mosaico dual. Por ejemplo, en la figura 3.20 se muestra un ejemplo 2-D con una asociación de celdas duales en el que se pueden observar *voxels* con más de una topología de celda dual asociada. Para denotar las distintas topologías de celdas duales asociadas a un determinado *voxel* usaremos la

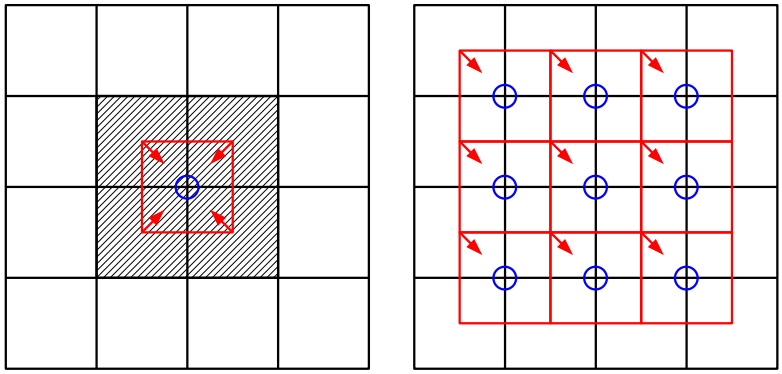

Figura 3.19: La imagen de la izquierda muestra cuatro *pixels* adyacentes y la celda dual que engloba el vértice que comparten. Las fechas indican la posición relativa del vértice con respecto al *pixel* correspondiente. La imagen de la derecha muestra una posible asignación de la topología de las celdas duales a algunos *voxels* de la rejilla regular.

clásica notación geográfica. De esta forma, podemos decir, por ejemplo, que el *voxel* de color gris claro tiene asociada la topología de la celda dual situada en la dirección Sureste relativa a dicho *voxel*.

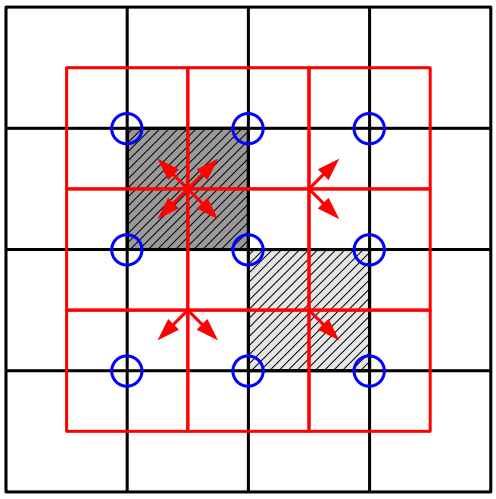

Figura 3.20: Asignación de la topología de las celdas del mosaico dual a cuatro *pixels* de la rejilla regular. El *pixel* de color gris oscuro tiene asociada la topología de las cuatro celdas duales que comparten su centro, mientras que el *pixel* de color gris claro solo tiene asociada la topología de la celda dual situada en la posición Sureste relativa a dicho *pixel*.

Esta asociación de la topología de las distintas celdas duales a los *voxels* cobra sentido cuando tratamos con nuestra representación mediante un *octree* de *voxels* de tamaño variable. En este caso, el número de topologías de celdas duales que pueden asociarse a un determinado *voxel* depende del número y tamaño de los *voxels* adyacentes a éste. En la figura 3.21 se muestra la idea para el caso 2-D. Se puede observar que el *pixel* **a** puede tener asociada la topología de hasta siete celdas duales, mientras que el *pixel* **b** puede tener asociada, como máximo, la topología de cuatro celdas duales.

De hecho, en el caso 2-D, cada vez que un *pixel* adyacente de arista de igual tamaño (o menor) es subdividido, se produce una nueva celda dual, debido a que aparece un nuevo

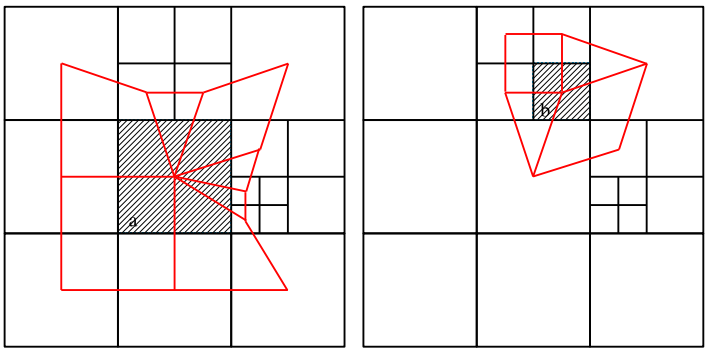

Figura 3.21: La imagen izquierda muestra las celdas duales a un *pixel* **a** que tiene varios *pixels* adyacentes de menor tamaño. La imagen a la derecha muestra las celdas duales a un *pixel* **b** cuyos *pixels* adyacentes presentan todos un tamaño igual o mayor.

vértice sobre la arista compartida (o la porción compartida de ésta). Sin embargo, si los *pixels* adyacentes de arista a uno dado tienen todos un tamaño igual o mayor que éste, no aparecen vértices en ninguna de sus aristas y, por consiguiente, el máximo número posible de topologías de celdas duales asociadas siempre es igual a cuatro, coincidiendo este número con el de vértices del *pixel*. En la imagen de la izquierda de la figura 3.21 se puede observar este fenómeno. La arista derecha del *pixel* **a** tiene un *pixel* vecino que se encuentra subdividido, el cual, a su vez tiene subdividido su componente inferior izquierdo. Cada subdivisión ha provocado la aparición de un vértice en la arista compartida por los *pixels* y, por tanto, la aparición de una nueva celda dual. El caso en el que los *pixels* adyacentes de vértice se subdividan no influye en la aparición de nuevas celdas duales, ni en el caso 2-D ni en el 3-D.

En el caso 3-D, los *voxels* adyacentes de arista y de cara de igual o menor tamaño que son subdivididos provocan la aparición de celdas duales. En el caso de los *voxels* adyacentes de arista, cada subdivisión provoca la aparición de un nuevo vértice sobre la arista compartida. En el caso de *voxels* adyacentes de cara, cada subdivisión provoca la aparición de cuatro vértices que caen sobre la cara compartida, concretamente sobre las aristas del *voxel* subdividido situadas en la cara compartida, más un vértice situado en el centro de la cara del *voxel* subdividido que se encuentra situada en la cara compartida. La figura 3.22 muestra dos ejemplos de estas situaciones. La imagen izquierda muestra dos subdivisiones de *voxels* adyacentes de arista, mientras que la imagen derecha muestra dos subdivisiones de *voxels* adyacentes de cara. Los círculos de color azul oscuro y azul celeste rodean los vértices resultado de la primera y segunda subdivisión respectivamente.

Entonces, se puede construir una representación que describa implícitamente la topología del mosaico dual de celdas irregulares, si se considera que un *voxel* puede tener la *responsabilidad de generar* todas las celdas que comparten el vértice que coincide con su punto central. Si se observa el número de celdas que puede generar un *voxel*, se puede identificar perfectamente los *voxels* que constituyen cada una de las celdas. Si se utiliza el etiquetado clásico basado en las direcciones geográficas, se pueden observar en la imagen izquierda de la figura 3.23 las etiquetas de cada uno de los *voxels* adyacentes relativas al *voxel* central (sombreado), incluida la etiqueta propia a éste. En la imagen derecha se muestra un ejemplo que determina los *voxels* necesarios para construir la topología de la celda situada en la dirección Noroeste relativa al *voxel* sombreado. Una ordenación fijada a priori de los *voxels* que constituyen cada una de las celdas duales

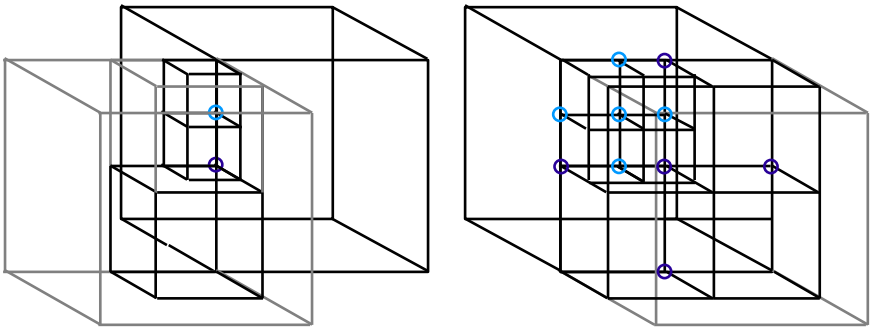

Figura 3.22: La imagen izquierda muestra un ejemplo de subdivisión de un *voxel* adyacente de arista, mientras que la imagen de la derecha muestra un ejemplo de subdivisión de *voxel* adyacente de cara. En ambos ejemplos se muestran los vértices resultantes de la primera subdivisión rodeados por un círculo en azul oscuro y los de la segunda subdivisión por círculos en azul celeste.

va a determinar perfectamente la *topología de la celda*. Por ejemplo, en este caso, la celda situada en la dirección relativa Noroeste siempre está formada por los *voxels* situados en las posiciones relativas: Oeste(W), Central(C), Noroeste(NW) y Norte(N), siempre que la ordenación fijada a priori sigua esta enumeración de *voxels* . Hablamos de topología de la celda en el sentido de que quedan perfectamente determinadas las relaciones de conectividad entre las distintas esquinas de ésta.

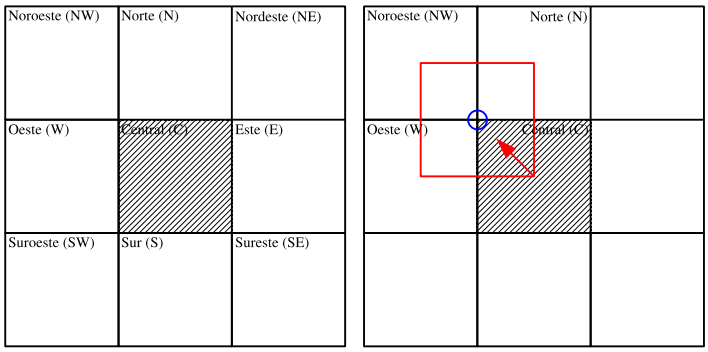

Figura 3.23: Etiquetado de *voxels* vecinos a uno dado (imagen izquierda). Topología de la celda dual que engloba al vértice del *voxel* central que se encuentra situado en la dirección relativa Noroeste (imagen derecha).

En el caso de la representación mediante *voxels* de tamaño variable, se presenta un problema a la hora de definir la topología de las celdas en el caso de que el *voxel* de interés tenga *voxels* adyacentes de menor tamaño (ver *voxel* **a** en la figura 3.21). El número de topologías de celdas que puede tener asociado un *voxel* determinado depende del número de *voxels* adyacentes. Por tanto, no se puede determinar a priori el número de celdas duales asociadas a los vértices que caen en la frontera del *voxel* de interés, y además, no se pueden establecer a priori las direcciones relativas de estos vértices. Sin embargo, en el caso del *voxel* **b** de la misma figura, se puede determinar perfectamente tanto el número de topologías de celdas duales asociadas, como las direcciones relativas al *voxel* de cada uno de los vértices que serán englobados por tales celdas duales. Este se debe al hecho de que los *voxels* adyacentes tienen todos un tamaño igual o mayor

al de dicho *voxel*, y por tanto, ningún *voxel* adyacente se encuentra subdividido con relación al tamaño del *voxel* **b**.

Tras la exposición de las anteriores observaciones, se van a definir formalmente las ideas mediante una serie de conceptos que van a permitir describir correctamente la asociación de topologías de celdas a los *voxels* que cumplen las características que presenta el *voxel* **b** de la figura 3.21.

Definición 19 (*Voxel* minimal) *Un* voxel *es minimal si tiene el menor tamaño de entre el conjunto de* voxels *adyacentes que comparten un vértice común.*

Definición 20 (Responsabilidad de generación de celdas) *La responsabilidad de generación de celdas de un* voxel *determina el conjunto de celdas del mosaico 3-D dual cuya topología está asociada a dicho* voxel*. Hablamos de responsabilidad de generación de celdas de un* voxel *en el sentido de que, cuando se recorre el* octree*, dicho* voxel *tiene que generar las celdas que están bajo su responsabilidad.*

Lema 1 (Asignación de responsabilidades a *voxels* minimales) *Si se asocia la responsabilidad de generación de las celdas duales solamente a un elemento del conjunto los* voxels *minimales asociado a cada uno de los vértices del mosaico de* voxels*, entonces está garantizado que cualquier* voxel *tendrá asociada la responsabilidad de procesar como mucho ocho celdas duales.*

Demostración. Por definición, los *voxels* minimales son los *voxels* de menor tamaño de entre el conjunto de *voxels* que comparten un vértice.

Sea vox_a un *voxel* cualquiera del mosaico 3-D de *voxels*.

Supongamos que vox_a no cumple el lema y tiene la responsabilidad de generar 9 celdas duales. Entonces, el *voxel* tiene que tener 9 vértices situados en su frontera. Supongamos que 8 de estos vértices son los que definen su forma cúbica y además existe uno cualquiera adicional. Entonces, este vértice adicional solamente puede ser consecuencia de que algún *voxel* adyacente de arista o de cara haya sido subdividido.

Sin embargo, una subdivisión provoca que los *voxels* adyacentes resultantes tengan un tamaño menor que el *voxel voxa*, por lo que, debido a la definición 19, vox_a **no es un *voxel* minimal para el vértice adicional**. Como todo vértice frontera distinto de los que definen la forma cúbica del *voxel* proviene de una subdivisión de arista o de cara, *voxa* nunca podrá ser minimal para alguno de dichos vértices y, como mucho, podrá ser el minimal de los ocho vértices que definen su forma cúbica, como se quería demostrar. La figura 3.24 ilustra visualmente nuestra demostración para el caso 2-D.

Corolario 1 *La subdivisión de un* voxel *adyacente de arista del mismo tamaño que un* voxel*,* vox_a*, provoca que el* voxel vox_a *ya no pueda ser minimal para los vértices extremos de dicha arista.*

La subdivisión de un voxel *adyacente de cara del mismo tamaño que un* voxel*,* vox_a*, provoca que el* voxel vox_a *ya no pueda ser minimal para los vértices pertenecientes a dicha cara.*

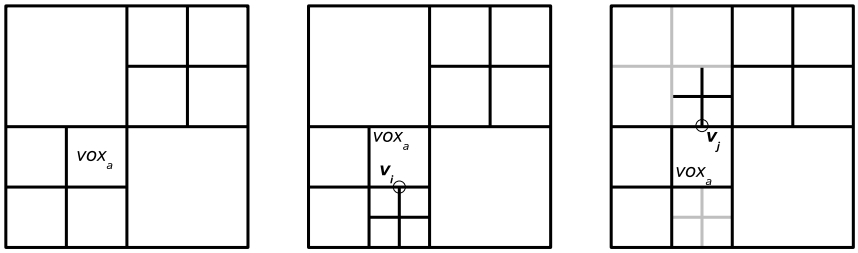

Figura 3.24: La imagen izquierda muestra un *voxel* cualquiera vox_a del mosaico 3-D. La imagen central muestra la aparición de un vértice adicional v_i como resultado de la subdivisión de un *voxel* adyacente del mismo tamaño que vox_a. Como puede comprobarse, vox_a no es un *voxel* minimal para este nuevo vértice v_i. La imagen de la derecha muestra la aparición de un nuevo vértice v_j resultado de dos subdivisiones sucesivas de un *voxel* adyacente de mayor tamaño. El resultado es el mismo ya que el elemento determinante para la aparición del vértice v_j es la segunda subdivisión que afecta al *voxel* adyacente superior de igual tamaño que vox_a.

La demostración del corolario es obvia debido a la forma de subdivisión del espacio que tiene un *octree* y puede comprobarse visualmente en la figura 3.22.

Utilizando el lema 1 se puede definir a priori, para los *voxels* minimales, el número máximo de vértices que pueden ser englobados por celdas duales, así como la topología de éstas en base a un criterio de ordenación de *voxels* adyacentes.

Definición 21 (Dirección de responsabilidad) *La dirección de responsabilidad de generación de celdas de un* voxel *minimal determina una de entre las ocho únicas posibles celdas duales, correspondientes a los ocho únicos vértices de los cuales puede ser minimal el* voxel*, que pueden ser generadas a partir de éste. Denotaremos las ocho posibles direcciones de responsabilidad de la siguiente forma: Noroeste Inferior (NOI), Nordeste Inferior (NEI), Noroeste Superior (NOS), Noreste Superior (NES), Suroeste Inferior (SOI), Sureste Inferior (SEI), Suroeste Superior (SOS), Sureste Superior (SES).*

Definición 22 (*Voxel* virtual) *Un* voxel *virtual es cada uno de los* voxels *de menor tamaño, hasta llegar al tamaño de los* voxels *de máxima resolución, en los que puede descomponerse una determinado* voxel*.*

Obviamente, para que un *voxel* pueda contener *voxels* virtuales debe haber sido obtenido como resultado de al menos una operación de agregación. En caso contrario, el *voxel* sería uno de máxima resolución y solo contendría un *voxel* virtual que coincidiría con el mismo.

Los valores de propiedad asociados a los vértices de cada celda dual generada coinciden con los de los *voxels* que son identificados mediante la topología de la celda definida por la dirección de responsabilidad correspondiente. En el caso de *voxels* vecinos de mayor tamaño, debido al concepto de región homogénea, sabemos que cualquier *voxel* virtual contenido en el *voxel* tiene su mismo valor de propiedad. Para determinar correctamente la topología de las celdas generadas en el caso de vecinos de mayor tamaño vamos a usar el concepto de *voxel* minimal junto con el de *voxel* virtual.

Consideremos un *voxel* que es minimal para el vértice situado en una dirección de responsabilidad determinada, la cual tiene asignada. Entonces, dicho *voxel* será responsable de generar la celda dual que engloba al vértice indicado por la dirección de dicha responsabilidad. Puede darse la situación de que uno de los *voxels* adyacentes

que forman la topología, fijada a priori para esa determinada dirección, sea de mayor tamaño que el *voxel* minimal. Para describir la topología de la celda dual se considera el *voxel* virtual adyacente con el mismo tamaño que el *voxel* minimal, el cual se encuentra incluido en el *voxel* adyacente de mayor tamaño. Sin embargo, a la hora de asignar significado geométrico a la celda se utiliza la posición del centro del *voxel* real. Por tanto, los *voxels* virtuales que definen la topología de una celda y que se encuentren incluidos en el mismo *voxel* real van a coincidir en el centro geométrico de éste. Este es el motivo por el cual se producen las deformaciones de celda cúbica descritas con anterioridad (Ver la figura 3.13) que provocan la aparición de celdas con topología distinta a la cúbica.

Utilizando la idea de asignar responsabilidades de generación de celdas duales exclusivamente a los *voxels* minimales hemos desarrollado un método para realizar automáticamente dicha asignación. Nuestro método debe cumplir los siguientes requisitos para representar de forma correcta el mosaico 3-D dual de celdas irregulares:

1. Únicamente deben tener asignadas responsabilidades de generación de celdas los *voxels* minimales de entre el conjunto de *voxels* del *octree*. (*Asignación minimal de responsabilidades*)

2. Para evitar la duplicidad en la generación de la misma celda dual desde varios de sus *voxels* minimales, cada celda sólo puede tener un único *voxel* minimal que la genere. Es decir, solamente puede haber un *voxel* del conjunto de *voxels* minimales de un vértice, cuya responsabilidad de generar la celda dual esté asignada. (*Unicidad*)

3. Todas las celdas que forman el mosaico 3-D dual tienen que poder ser generadas a partir de la asignación de responsabilidades de generación asociada a los *voxels* minimales. Es decir, cada vértice de *voxel* debe tener una dirección de responsabilidad asociada. (*Completitud*)

3.4.2. Método de asignación de responsabilidades de generación de celdas

Para cumplir los requisitos establecidos para el método se va a recorrer cada uno de los *voxels* representados en el *octree*. Denominaremos *voxel actual* al *voxel* que se está considerando en el momento actual para la asignación de responsabilidades de generación de celdas.

Dado un *voxel* actual, se asigna inicialmente una responsabilidad de generación en una dirección fijada a priori. Asignar solamente una dirección de responsabilidad para cada *voxel* visitado, en lugar de comprobar las ocho posibles, es suficiente para cumplir el requisito de completitud. Esto se debe a que el mosaico de celdas es dual al mosaico formado por los *voxels*, y por tanto, la celda correspondiente a dicha dirección podría ser generada a partir de cualquiera de los *voxels* que tienen en su centro un vértice de la celda. En la figura 3.25 se muestra la asignación de responsabilidad inicial para el *voxel* actual. El vértice rodeado por un círculo rojo se corresponde con la responsabilidad que se va a comprobar y las aristas y caras coloreadas en rojo constituyen la parte de la frontera del *voxel* actual que podrá presentar vértices adicionales debido a la posible subdivisión de los correspondientes *voxels* adyacentes. Denominamos al conjunto de aristas y caras, junto con el vértice determinado por la dirección de responsabilidad, *frente de traslado de responsabilidad*.

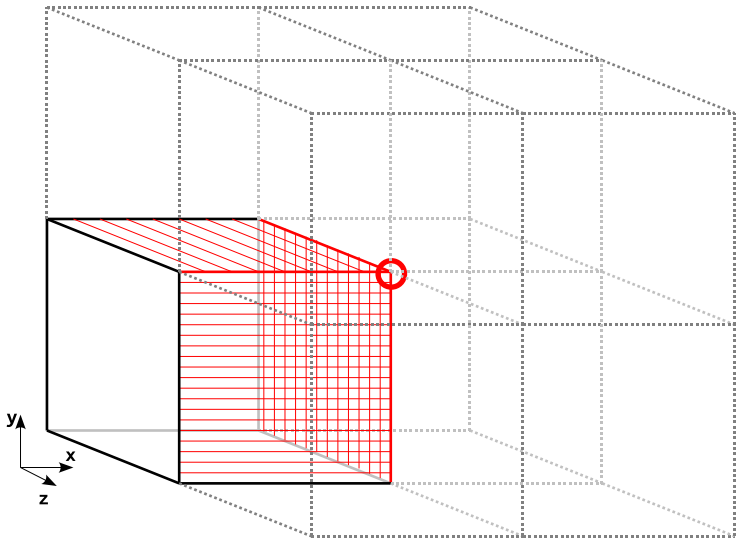

Figura 3.25: Para cada *voxel* actual (en rojo) se comprueba el vértice situado en la dirección Sureste Superior para determinar si la responsabilidad asociada será fijada o no. El tamaño de los *voxels* adyacentes necesarios para la construcción de la celda dual no se conoce a priori por lo que se debe realizar una búsqueda en el *octree*.

A priori se desconoce el tamaño que van a tener los *voxels* adyacentes al frente de traslado de responsabilidad. Por tanto, es necesario realizar una búsqueda en el *octree*. Si todos los *voxels* adyacentes encontrados tienen un tamaño mayor o igual que el *voxel* actual, entonces se puede concluir que éste es un *voxel* minimal para el vértice determinado por la dirección de responsabilidad prefijada y, por consiguiente, la responsabilidad de generación de la celda dual que engloba a dicho vértice queda establecida en el *voxel* actual. Por el contrario, si se encuentran *voxels* adyacentes cuyo tamaño es menor que el del *voxel* actual, entonces significa que se está visitando un *voxel* que no es minimal para la dirección de responsabilidad prefijada. De hecho, lo que ocurre es que algunos de los *voxels* adyacentes que podrían haber tenido el mismo tamaño que el actual se encuentran subdivididos.

En el segundo caso, la responsabilidad prefijada ya no puede ser de ninguna manera asignada al *voxel* actual, ya que seguro que habrá al menos uno de tamaño la mitad que éste que será adyacente al vértice. Adicionalmente, las posibles subdivisiones de los *voxels* adyacentes provocan la aparición de vértices situados en las aristas y caras señaladas en la figura 3.25. Con el fin de determinar la asignación de responsabilidades correcta para cada uno de estos vértices de arista y cara, junto con la del vértice determinado a priori, se realiza un proceso de *traslado de responsabilidad* sobre el frente de traslado de responsabilidad.

Las distintas direcciones de responsabilidad de generación de celdas se asignan a los *voxels* adyacentes al frente de responsabilidad en base a las relaciones de adyacencia de éstos con el *voxel* actual y a la dirección de responsabilidad que se quiere trasladar. Por supuesto, esta dirección ya no quedará fijada en el *voxel* actual. Es necesario asignar una dirección de responsabilidad para cada uno de los vértices que caen en el frente de responsabilidad del *voxel* actual. Esta asignación tiene que cumplir los requisitos establecidos para la representación.

En base a lo expuesto, el método asigna las direcciones de responsabilidad asociadas

a los vértices del frente de traslado responsabilidad, a los *voxels* adyacentes de menor tamaño, escogiendo para éstas una orientación opuesta a la dirección fijada a priori para el *voxel* actual. La figura 3.26 muestra un ejemplo 2-D del proceso de traslado de responsabilidad para un *voxel* situado en la parte superior izquierda.

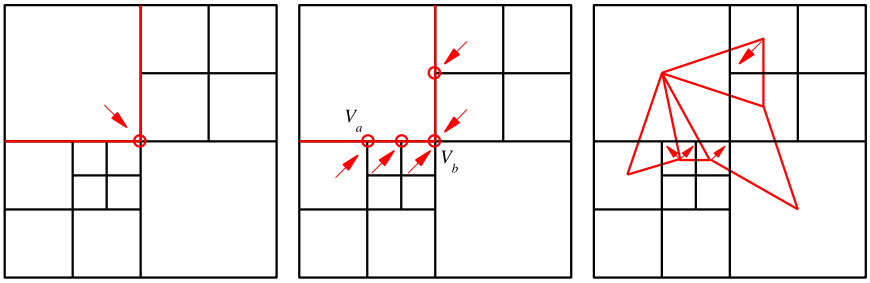

Figura 3.26: Traslado de la responsabilidad fijada a priori para el *pixel* actual (situado en la parte superior izquierda del *quadtree*) a los *voxels* de menor tamaño adyacentes al frente de traslado de responsabilidad determinado por la dirección de dicha responsabilidad.

En la figura 3.26 se observa que la asignación de responsabilidades a los *voxels* adyacentes de menor tamaño provoca que se defina la generación de celdas topológicas duales que incluyen a todos los vértices que caen en el frente de traslado como consecuencia de las distintas subdivisiones. La imagen de la izquierda muestra el *voxel* actual, situado en la parte superior izquierda del *quadtree*, junto con la dirección de responsabilidad que se va a comprobar. La imagen del centro muestra el traslado de responsabilidad a los *voxels* adyacentes de menor tamaño. Este traslado puede provocar dos situaciones (puestas de manifiesto en los vértices etiquetados como $\boldsymbol{V}_a$ y $\boldsymbol{V}_b$) que no cumplen los requisitos de la representación y que se muestran de forma más detallada en la figura 3.27. La imagen de la derecha muestra la asignación de responsabilidades final junto con las celdas duales que representa dicha asignación.

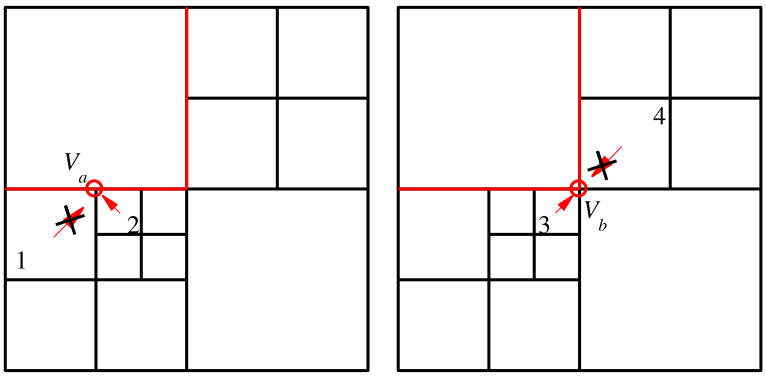

Figura 3.27: La figura muestra las dos posibles situaciones de traslado de responsabilidad a *pixels* adyacentes que no cumplen los requisitos de la representación. El vértice $\boldsymbol{V}_a$ del frente de traslado es compartido por dos *pixels* adyacentes, 1 y 2, pero la asignación de responsabilidad recae en uno de ellos que no es un *voxel* minimal del vértice $\boldsymbol{V}_a$ (asignación de responsabilidades no minimal). Dos de los *voxels* adyacentes, 3 y 4, que comparten el vértice $\boldsymbol{V}_b$, tienen asignada la responsabilidad de la celda dual que engloba a este vértice (No unicidad).

En la imagen de la izquierda de la figura 3.27 se muestra el vértice $\boldsymbol{V}_a$ y la situación

no válida provocada por el método de traslado de responsabilidad. Ésta se produce cuando un *voxel* adyacente de menor tamaño tiene asignada la responsabilidad de generación de la celda dual que engloba al vértice, pero dicho *voxel* no pertenece al conjunto de *voxels* minimales de $\boldsymbol{V}_a$. En este caso se viola el requisito de la asignación minimal de responsabilidades. Para solucionarlo simplemente se actua trasladando la responsabilidad de generación a un *voxel* minimal del conjunto de *voxels* adyacentes que comparten el vértice, en este caso sólo puede elegirse el *voxel* 2, eliminando la responsabilidad establecida con anterioridad. Así se garantiza el requisito de asignación minimal.

La imagen de la derecha muestra el vértice$\boldsymbol{V}_b$. En este caso, el método de traslado de responsabilidad provoca que se asigne la responsabilidad de generación de la misma celda topológica a varios *voxels* adyacentes (*voxels* 2 y 4), violando de esta forma el requisito de unicidad. Para cumplir este requisito, hay que comprobar el conjunto de *voxels* adyacentes que tienen la responsabilidad de generación asignada sobre la misma celda y determinar cual de ellos es un *voxel* minimal para el vértice. Para ello se determina el tamaño de todos y se escoge uno de los de menor tamaño (o, como es el caso del ejemplo, el único de menor tamaño, 3) y se elimina la responsabilidad de generar la celda correspondiente al vértice para el resto de *voxels* adyacentes. De esta forma, se garantizan conjuntamente los requisitos de unicidad y asignación minimal de responsabilidades.

Como resultado de aplicar el método sobre el *voxel* actual se obtiene una asignación de responsabilidades que cumple los requisitos de la representación. Concretamente, el requisito de completitud se consigue para todos los vértices del frente de traslado. Esta asignación permite mantener almacenado implícitamente en los *voxels* correspondientes todo el conjunto de celdas duales a dichos vértices.

Tras aplicar el método de asignación de responsabilidades a todos las *voxels* representados por nodos hoja del *octree*, se dispone de una representación que tiene implícitamente definida la topología de todas las celdas duales que engloban los vértices de los *voxels* (exceptuando los vértices exteriores que son tratados especialmente como se ha explicado con anterioridad). El mosaico 3-D dual a los *voxels* se obtiene recorriendo el *octree* y para cada nodo hoja (*voxel*) con responsabilidades de generación de celdas asignadas, se buscan los vecinos determinados por la dirección de cada responsabilidad y se asigna a cada vértice topológico su correspondiente posición geométrica que se corresponde con el centro del *voxel* determinado por las relaciones de conectividad fijadas a priori.

Por tanto, nuestra representación se sustenta en un *octree* cuyos nodos hoja representan los *voxels* y su tamaño junto con las direcciones de responsabilidad para generar celdas duales. Estos nodos almacenan el valor de propiedad de cada región homogénea, y además, almacenan un byte que representa las responsabilidades de generación de celdas para los *voxels* minimales. En la figura 3.28 podemos ver el equivalente 2-D de lo que sería un ejemplo de un objeto volumétrico representado mediante el *octree* de volumen con celdas duales definidas implícitamente (*IDDG-Octree*).

3.5. Extracción de isosuperficies a partir del *IDDG-Octree*

Nuestro objetivo es extraer una isosuperficie, definida mediante un valor de propiedad, a partir de nuestra representación discreta de resolución variable. En la representación tenemos definidas implícitamente las celdas sobre las que que vamos a realizar el cálculo de los triángulos que aproximan la isosuperficie. En primer lugar se detectan

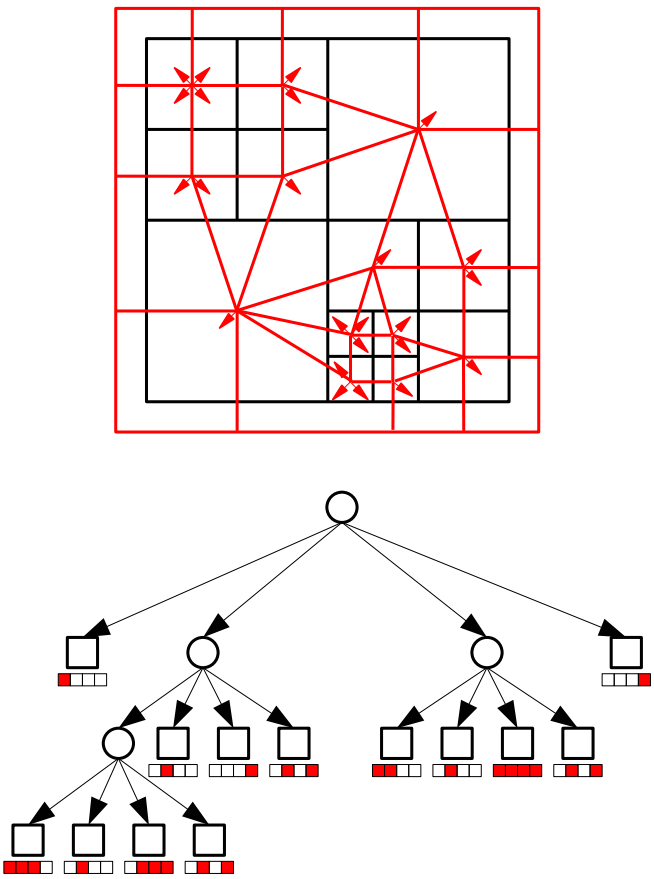

Figura 3.28: La figura muestra el equivalente 2-D (*quadtree*) de lo que sería la representación de un objeto volumétrico mediante el *octree* de volumen con celdas duales definidas implícitamente (*IDDG-Octree*).

celdas activas recorriendo el *octree* y utilizando la información que proporcionan las responsabilidades asignadas a *voxels* minimales. En segundo lugar se asigna significado geométrico a las celdas definidas implícitamente en función de los *voxels* adyacentes necesarios para obtener celdas válidas. A continuación se aplica una modificación del algoritmo *Marching Cubes* sobre cada una de las celdas para así obtener la malla de triángulos que aproxima la isosuperficie.

3.5.1. Obtención de celdas geométricas

Para visualizar nuestro modelo volumétrico utilizamos un algoritmo basado en la idea de "*marchar*" sobre las celdas que discretizan el espacio ocupado por la representación volumétrica. En nuestro caso, las celdas sobre las que se "marcha" se encuentran definidas implícitamente en los *voxels* minimales del *IDDG-Octree*. Para aplicar el algoritmo de extracción sobre una celda que es atravesada por la isosuperficie es necesario dotarle previamente de significado geométrico, ya que las responsabilidades de nuestra representación solo indican la forma de hallar la conectividad entre los *voxels* que determinan cada celda dual.

Para realizar una visualización del volumen mediante la técnica de extracción de isosoperficies, en primer lugar debemos escoger un isovalor, a partir del cual se extraerá la

aproximación a la isosuperficie que viene definida por dicho valor. Una vez fijado el umbral de isosuperficie el algoritmo recorre el *octree* y, para cada nodo hoja, examina las responsabilidades de generación de celdas almacenadas. Para cada responsabilidad asignada se considera como celda activa la celda dual determinada por la dirección de dicha responsabilidad. A continuación, se buscan en el *octree* los nodos hoja que representan al conjunto de *voxels* adyacentes que determinan la topología de la celda dual. Utilizando el isovalor fijado a priori y los valores de propiedad contenidos en los nodos hoja se determina si la celda dual presenta aristas de corte con la isosuperficie. Si no se detectan aristas de corte se pasa a procesar la siguiente responsabilidad, o el siguiente nodo hoja si ya no hay más responsabilidades asignadas en el actual. Por el contrario, si se detectan aristas de corte, se genera la geometría de la celda dual para poder ser procesada.

Como se ha comentado anteriormente, las celdas duales no tienen necesariamente una forma cúbica. A pesar de esto, debido al hecho de que las degeneraciones de aristas se producen por la agregación de *voxels*, es posible procesar las celdas como si lo fuesen. Las aristas degeneradas se producen porque dos *voxels* (o cuatro en el caso de caras degeneradas) del mismo tamaño que el *voxel* actual se encuentran agregados en un *voxel* de mayor tamaño, es decir, son *voxels* virtuales. Nótese que los *voxels* adyacentes, virtuales o no, del mismo tamaño que el *voxel* actual constituyen la topología de las posibles celdas duales. Aunque una arista representada por dos *voxels* virtuales sucesivos se degenere en el vértice situado en el centro del *voxel* real que los continene, se puede seguir considerando como una arista para los casos de cualquier algoritmo que trabaje sobre celdas cúbicas. Esto se debe a que el valor de propiedad asociado a los extremos de la arista será el mismo e igual al almacenado para el *voxel* real y, por lo tanto, no puede existir un punto de corte en tal arista. Obviamente, existe una excepción cuando el valor de propiedad asociado al vértice geométrico en el que se degenera la arista coincide con el isovalor. No obstante, esta situación también puede producirse en el caso de celdas cúbicas. La figura 3.29 muestra dos ejemplos de posibles correspondencias de celdas topológicas a sus correspondientes celdas geométricas.

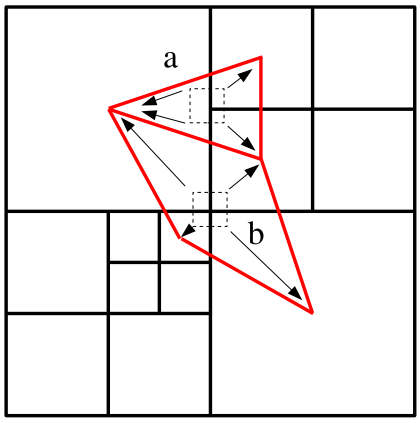

Figura 3.29: Correspondencia entre la definición topológica de una celda y la definición geométrica de la misma celda. Dos vértices de la celda topológica **a** coinciden en un único vértice geométrico, mientras que cada uno de los vértices de la celda topológica **b** se corresponde con un vértice geométrico diferente.

En concreto, para determinar la geometría de una celda representada implícitamente se consideran todos los *voxels* adyacentes necesarios para constituir la celda dual y se

procede a establecer las relaciones de conectividad de la siguiente forma. Si el *voxel* adyacente tiene el mismo tamaño que el *voxel* actual, el vértice geométrico asociado coincide con el del centro de dicho *voxel*. Para cada *voxel* adyacente que tenga un tamaño mayor que el *voxel* actual existirán coincidencias entre los vértices topológicos asociados a cada "*voxel* virtual" del mismo tamaño que el actual. Es decir, las relaciones de conectividad se comprueban como si los *voxels* necesarios fuesen del mismo tamaño que el *voxel* actual y, a la hora de asignar los vértices geométricos, todos los vértices topológicos de los "*voxels* virtuales" englobados en el mismo *voxel* coincidiran con el centro de éste. De esta forma, se obtiene una celda dual con vértices geométricos coincidentes, con la garantía de que los vértices que coinciden tienen el mismo valor de propiedad.

Se puede aplicar el algoritmo *Marching Cubes* (MC) o cualquiera de sus sucesores que realice extracción de isosuperficies sobre celdas cúbicas a las celdas topológicas y, una vez determinado el caso, se obtiene la celda geométrica y se calculan los puntos de corte. Para ello simplemente se consideran las relaciones de conectividad proporcionadas por los *voxels* adyacentes, virtuales o no, y la ordenación establecida a priori para las celdas duales. Estas relaciones de conectividad forman una celda cúbica topológicamente hablando, que puede ser utilizada para clasificar sus vértices en función del valor de propiedad almacenado en los *voxels* adyacentes y el valor umbral definido, pudiéndose comprobar seguidamente el caso de triangulación que presenta la celda. Lo único que ocurre es que los vértices asociados a los *voxels* virtuales de un mismo *voxel* real tendrán idéntico valor de propiedad y, por lo tanto, idéntica clasificación positiva o negativa. De esta forma ningún caso proporcionará un punto de corte en la arista topológica representada por ambos y, por consiguiente, no aparecerá nunca un punto de corte en la arista (o en la cara) que se degenera en el vértice central de un *voxel* real. La figura 3.30 ilustra la idea para una celda con una arista degenerada y un caso de MC. La arista degenerada proviene de dos *voxels* virtuales que establecen la topología de dicha arista y que están incluidos en un mismo *voxel* real. Por tanto, su valor de propiedad asociado es el mismo. Si consideramos que dichos *voxels* forman la arista topológica e etiquetada en la figura, se puede clasificar la celda topológica como perteneciente al tipo de caso que representa la celda de la izquierda, aunque a la hora de calcular los puntos de corte de las aristas se utilice el valor geométrico del vértice al que se degenera la arista, indicado por las flechas de la celda situada a la derecha. Lo que está claro es que nunca podrá clasificarse la celda topológica con un caso de MC que presente un punto de corte en la arista e.

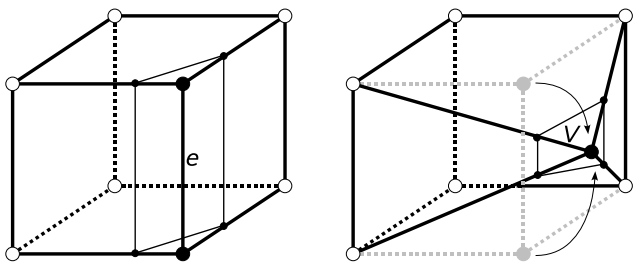

Figura 3.30: A la izquierda se muestra uno de los casos del algoritmo *Marching Cubes*, y a la derecha una celda con topología no cúbica resultado de la degeneración de la arista e. Las aristas que presentan puntos de corte son las mismas que en el caso de una celda cúbica aunque las posiciones de estos puntos varían en función de la posición del vértice $\boldsymbol{V}$.

3.5.2. Obtención de puntos de corte

El método que hemos utilizado para obtener los triángulos que aproximan la isosuperficie es una variante del algoritmo *Marching Cubes* [LC87], cuya forma de interpolación hemos adaptado a nuestra representación de resolución variable. En función de la configuración dentro/fuera del volumen englobado por la isosuperficie, definida por el umbral de valor de propiedad para cada una de las celdas, se busca en una tabla de configuraciones preestablecidas el caso correspondiente. Cada caso tiene asignada una triangulación de la celda que aproxima la isosuperficie correspondiente al caso particular. Como se ha visto en el capítulo de introducción, existen multitud de trabajos que han mejorado la tabla de casos del algoritmo MC para:

- Evitar ambigüedades en la generación de la malla, las cuales pueden provocar agujeros en la malla final.
- Incrementar la eficiencia a la hora de determinar los puntos de corte de la aristas de las celdas.
- Mejorar la aproximación de la malla generada en el interior de la celda utilizando información adicional a la proporcionada por los puntos de corte de las aristas.

Todas estas mejoras pueden ser aplicadas a nuestro método de extracción de isosuperficies con el objetivo de mejorar cada uno de los aspectos arriba indicados.

En el algoritmo de Lorensen [LC87], los vértices de los triángulos que constituyen la malla se calculan mediante interpolación lineal en las aristas de corte de las celdas de una rejilla 3-D uniforme y regular. Las aristas de corte son las aristas cuyos valores de propiedad en sus vértices extremos caen uno por encima y otro por debajo del valor umbral de propiedad que se corresponde con la isosuperficie que se desea extraer.

Las celdas definidas implícitamente en nuestra representación no forman un mosaico 3-D regular. Éstas se construyen a partir de las relaciones de adyacencia entre los *voxels* representados en el *octree*, y sus vértices están situados en los centros de estos. En consecuencia, la longitud de las aristas de las celdas duales resultantes depende del nivel en el que están situados los distintos *voxels* que constituyen la celda, es decir, *de los distintos tamaños de los* voxels *adyacentes necesarios para determinar la celda dual*.

Si se procede a interpolar directamente entre los vértices de las aristas de corte, entonces para aristas que posean idéntico valor de propiedad en cada extremo, se obtienen puntos de corte que van a depender de la longitud de cada arista. La isosuperficie que se obtiene si procedemos de esta forma se diferencia enormemente de la que se obtendría en el caso de aplicar el algoritmo *Marching Cubes* sobre la rejilla uniforme y regular a partir de la cual se obtiene nuestra representación. Este hecho viene provocado por el proceso de agregación de *voxels*, ya que éste incrementa la longitud de las aristas de las celdas del mosaico 3-D dual. En la figura 3.31 se muestra este problema en un ejemplo 2-D. Los círculos azules representan *voxels* con el mismo valor de propiedad, al igual que los rojos. La imagen superior muestra que los puntos de intersección entre la isocurva y las aristas de las celdas no caen sobre la misma línea recta, como reflejan los puntos de corte unidos por segmentos (en color gris). A pesar de que los valores de propiedad en los extremos de cada arista de corte son los mismos, la longitud de cada una de ellas es diferente, y por tanto la posición de los puntos de corte varía en cada una.

Por el contrario, en el caso de la rejilla regular y uniforme la longitud de las aristas de corte sí es la misma. Si se observa la isocurva que se habría podido extraer a partir de la

rejilla regular y uniforme inicial (imagen central de la figura 3.31) se puede comprobar que todos sus puntos de corte caen sobre una línea recta. Obviamente, si los valores de propiedad de los extremos coinciden, y la longitud de las aristas de corte coincide, la interpolación lineal proporciona este resultado. Si se observa el resultado de aplicar la interpolación lineal sobre las aristas de las celdas de nuestra representación (ver el análogo 2-D en la imagen superior de la figura 3.31) se pueden inferir dos efectos no deseables provocados por la interpolación, dado que el objetivo perseguido es que la isosuperficie extraida se parezca lo más posible a la isosuperficie extraida a partir de la representación regular:

1. Las aristas que tienen un vértice extremo en común situado en el centro del mismo *voxel* y el otro vértice situado en *voxels* de distinto tamaño con el mismo valor de propiedad, presentan puntos de corte a distintas distancias del vértice común, independientemente de tener el mismo valor de propiedad en el otro extremo.

2. Además, dependiendo de los valores de propiedad de los vértices extremos, pueden darse casos en los que la isosuperficie extraida esté situada en el interior de un *voxel*, el cual representa una región homogénea. Esta situación es totalmente contraria a la filosofía de nuestra representación. La isocurva de la imagen superior de la figura 3.31 muestra esta situación.

Para conseguir el objetivo de que la isosuperficie extraida a partir de nuestra representación sea lo más parecida posible a la extraida a partir de la rejilla regular y uniforme inicial utilizamos la siguiente idea. Nuestra estrategia se basa en "forzar" a que los puntos de corte caigan en su arista correspondiente, pero solamente en el segmento de la arista que solapa las zonas de borde de los *voxels* cuyos centros son los extremos de ésta. Estas zonas de borde se encuentran delimitadas por celdas duales de máxima resolución. En la imagen inferior de la figura 3.31 se muestran las zonas de borde entre los *voxels* adyacentes, cuyos centros son los extremos de las aristas, mediante una banda celeste.

Considerando el concepto de región homogénea, debemos rechazar la idea de que la isosuperficie pueda quedar en el interior de un *voxel*, ya que el valor de propiedad en toda su zona interior es el mismo. En nuestra representación solo es posible que exista una variación del valor de propiedad en las zonas de borde entre *voxels* adyacentes. Por tanto, si la interpolación sobre las aristas de corte se realiza solamente sobre el segmento que solapa dichas zonas, entonces se obtendrá una isosuperficie correcta desde el punto de vista del concepto de región homogénea. Más aún, considerando que nos interesa obtener una isosuperficie lo más parecida posible a la que se extraería a partir de la rejilla inicial, obtenemos una isosuperficie que, "moviéndose" en la zona correcta en función del isovalor escogido, se diferencia por la distinta orientación de las aristas de corte con respecto a la ortogonalidad presente en las aristas de corte de la rejilla inicial. Este efecto puede observarse en las imágenes central e inferior de la figura 3.31.

En la figura 3.32 se muestra un ejemplo 3-D en el que se ponen de manifiesto los problemas derivados de realizar una interpolación lineal directamente sobre las aristas de corte de las celdas duales cuyos extremos están situados en los centros de *voxels* que han sufrido un proceso de agregación. En la imagen de la izquierda se puede comprobar que los puntos de corte no caen sobre el mismo plano, como ocurre cuando se calculan a partir de la rejilla regular y uniforme, ya que cada *voxel* de menor tamaño tiene igual valor de propiedad. Este efecto se muestra más claramente en la vista que ofrece la imagen central. Si variamos el valor umbral, pueden producirse situaciones como

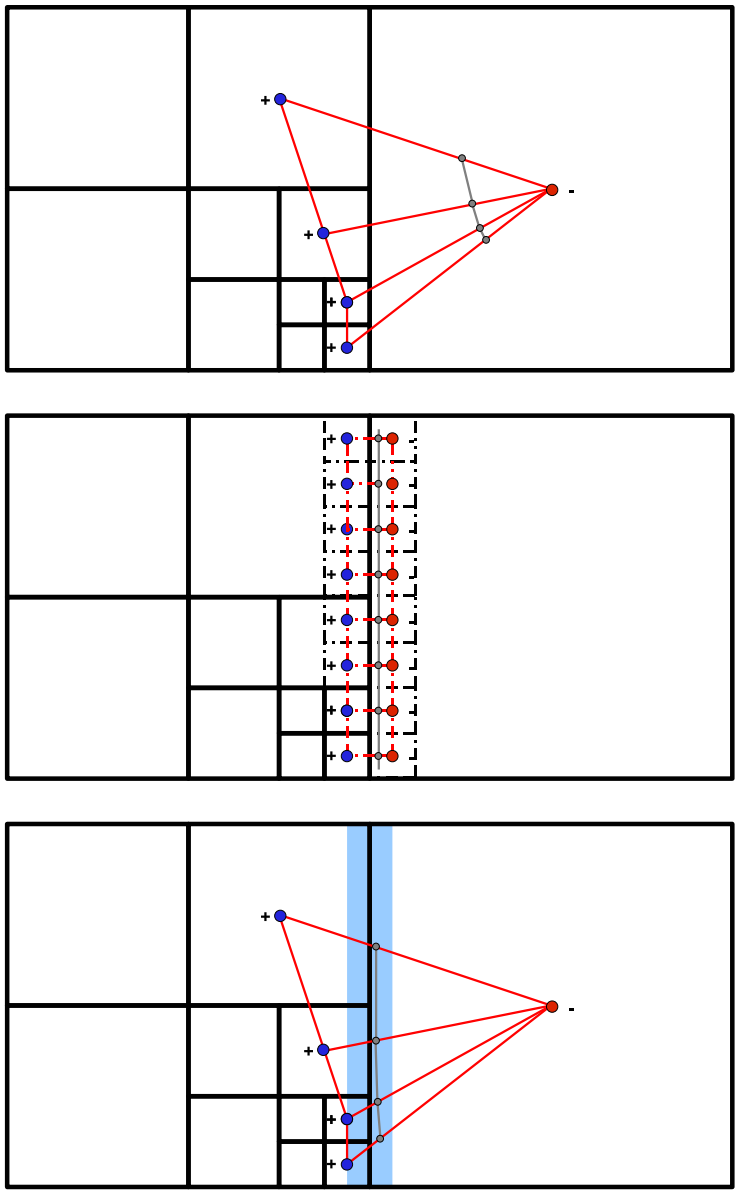

Figura 3.31: La imagen superior muestra la isocurva resultado de aplicar interpolación lineal en las aristas de las celdas duales. La imagen central muestra la isocurva generada a partir de la rejilla regular y uniforme original. La imagen inferior muestra la isocurva generada usando para la interpolación solamente el segmento de las aristas que solapa la zona borde de los *voxels* adyacentes.

la que se muestra en la imagen más a la derecha, en donde el parche de isosuperficie está situado completamente en el interior del *voxel* de mayor tamaño.

Con el objetivo de resolver el problema, nuestra función de interpolación tiene en cuenta para cada vértice de una celda dual el tamaño del correspondiente *voxel* en cuyo centro se encuentra situado dicho vértice. Básicamente la idea es realizar la interpolación en las aristas de corte de la celda dual pero, en lugar de utilizar como extremos de interpolación los vértices situados en los centros de los *voxels*, se realiza una intersección de dichas aristas con los límites de la banda de bordes y se interpola en los segmentos de las aristas de corte situados en el interior del poliedro resultante.

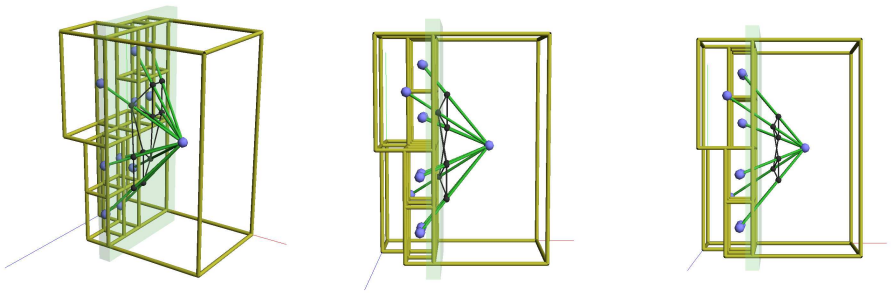

Figura 3.32: Las imágenes central e izquierda muestran dos vistas diferentes del conjunto de puntos de corte, los cuales no caen sobre el mismo plano. La imagen situada a la derecha muestra una situación en la que todos los puntos de corte caen dentro del *voxel* de mayor tamaño. La región transparente celeste representa la banda de bordes en donde caería la superficie extraida a partir de la rejilla regular y uniforme. Los segmentos de línea en negro delimitan el parche de isosuperficie que se extrae mediante interpolación lineal.

El método propuesto considera la interpolación en las aristas de corte para obtener los nuevos puntos extremos sobre la arista, los cuales delimitan el segmento real sobre el que se realizará la interpolación. Estos dos puntos extremos situados sobre la arista de corte son los más cercanos a los centros de los "*voxels* virtuales" de mayor resolución, los cuales coinciden con los *voxels* de la rejilla regular y uniforme inicial. En otras palabras, estos puntos coinciden con la intersección de la arista con los límites de la banda de bordes. Estos puntos aparecen etiquetados en la figura 3.33 como P_1' y P_2'. Para obtener la ecuación de interpolación basada en el tamaño de los *voxels* se plantea el siguiente sistema de ecuaciones que se deduce a partir de la geometría Euclidea como puede comprobarse en la figura 3.33. En esta figura se ilustra la construcción geométrica que conduce a la solución para el caso 2-D, aunque es fácilmente generalizable al caso 3-D.

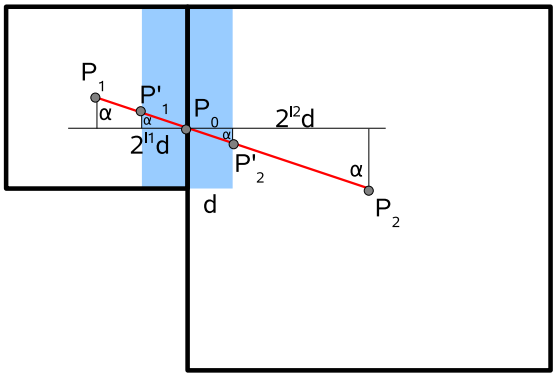

Figura 3.33: Construcción geométrica que permite calcular el valor de los puntos P_1' y P_2'.

Como puede observarse en la figura 3.33, trazando una línea paralela al eje x, que pase por el punto P_0, punto de intersección de la arista con la frontera compartida por los dos *voxels*, se obtienen dos triángulos rectángulos semejantes. Si se considera el *voxel* de menor tamaño, podemos obtener las siguientes igualdades utilizando el teorema del seno:

$$\frac{2^{l_1} \cdot d}{\sin \alpha_1} = \frac{P_1 - P_0}{\sin(\pi/2)}$$

$$\frac{2^{l_2} \cdot d}{\sin \alpha_1} = \frac{P_1' - P_0}{\sin(\pi/2)}$$

Despejando $\sin \alpha_1$ en ambas igualdades obtenemos la ecuación:

$$\frac{2^{l_1} \cdot d}{P_1 - P_0} = \frac{d}{P_1' - P_0} \tag{3.4}$$

Procediendo de igual forma en el *voxel* situado a la derecha obtenemos la siguiente ecuación:

$$\frac{2^{l_2} \cdot d}{P_2 - P_0} = \frac{d}{P_2' - P_0} \tag{3.5}$$

En este momento disponemos de dos ecuaciones para obtener los puntos P_1' y P_2'. Necesitamos una tercera ecuación que es fruto de la relación entre dichos puntos y el punto P_0:

$$P_0 = \frac{P_1' + P_2'}{2} \tag{3.6}$$

donde P_1 y P_2 están situados en el centro de cada *voxel*, P_0 es el punto situado en el plano frontera compartido por ambos, y P_1' y P_2' son los puntos que delimitan el segmento de arista en donde realmente hay que calcular el punto de corte de la isosuperficie.

Para obtener el punto de intersección buscado solamente resta interpolar linealmente entre estos dos puntos usando el valor de propiedad escalar almacenado en los correspondientes *voxels*. En la figura 3.34 se muestra el resultado de "forzar" a que la zona de extracción de isosuperficie coincida con la banda de bordes situada entre *voxels* adyacentes, siguiendo la estrategia de tomar en consideración el tamaño de dichos *voxels* de cara a realizar interpolación. En la imagen izquierda se muestra el conjunto de puntos (en negro) P_1' y P_2' asociados a cada una de las aristas de corte del ejemplo. Los puntos están situados sobre las aristas de corte y delimitan la banda de bordes. Además, entre estos conjuntos de puntos se pueden ver los puntos de intersección. Los conjuntos de puntos P_1' y P_2' están situados sobre los planos que coinciden con los planos que delimitan la banda de bordes. La imagen central muestra una vista más clara de los conjuntos de puntos P_1' y P_2' que delimitan el volumen (banda de bordes) en donde deben calcularse los puntos de corte. La imagen del medio y la imagen de la derecha muestran que la variación en el isovalor solamente provoca que los puntos de corte se acerquen más al conjunto de puntos P_1' o al conjunto P_2', pero nunca van a poder sobrepasar la zona de bordes. De esta forma, el método de interpolación garantiza que los puntos de corte siempre estén situados entre ambos conjuntos de puntos, independientemente de los valores de propiedad asociados a los extremos de las aristas de interpolación, del valor umbral escogido y del tamaño de los *voxels* en los que se situan dichos extremos.

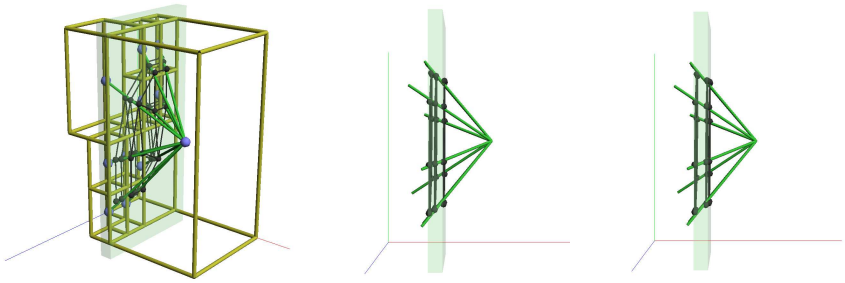

Figura 3.34: Las imagenes muestran como el conjunto de puntos de corte resultante de nuestro método de interpolación está situado en la banda de bordes y se encuentra delimitado por los conjuntos de puntos P_1' y P_2'. Las imágenes situadas a la izquierda y en el centro muestran dos vistas del conjunto de puntos de corte extraido a partir de un isovalor, y la imagen de la derecha muestra el conjunto de puntos extraidos a partir de otro isovalor. Los puntos de corte se acercan a cada uno de los límites de la banda de bordes pero nunca los sobrepasan.

3.5.3. Cálculo del gradiente en el punto de corte de la arista

Para poder completar la información requerida para realizar una estrategia de visualización mediante extracción de isosuperficies es necesario disponer de la información de las normales en los puntos de corte de las aristas. Estos puntos de corte son los vértices que constituirán los triángulos que utilizaremos como primitivas geométricas de aproximación de la isosuperficie, y estos, junto con sus normales asociadas permitirán aplicar el modelo de iluminación correspondiente.

Se suele utilizar el vector gradiente de valor de propiedad en un punto para aproximar la normal en cada punto de corte. El cálculo del vector gradiente en un punto se suele llevar a cabo en representaciones discretas mediante el método de diferencias finitas. Nuestra representación no almacena explícitamente el vector gradiente para cada elemento de volumen, lo que nos permite ahorrar espacio de almacenamiento. Por tanto, necesitamos un método que permita estimar el valor del vector gradiente en un punto a partir de los valores de propiedad representados en nuestro modelo.

En la imagen superior de la figura 3.35 podemos observar dos ejemplos de segmentos de arista que solapan la zona de bordes y que están delimitados por los puntos P' Q' y R' Q'' respectivamente. Cada uno de los puntos que delimitan el segmento de arista sobre el que queremos realizar la interpolación (que hemos denominado genéricamente P_1' y P_2') se encuentra en un *voxel* virtual de máxima resolución. Cada uno de estos *voxels* se encuentra agregado en su correspondiente *voxel*, en cuyo centro cae uno de los extremos de la arista, salvo en el caso de que el *voxel* que lo contenga sea en realidad uno de máxima resolución.

Utilizando esta observación vamos a realizar una interpolación lineal sobre la arista que une los puntos P_1' y P_2' con el objetivo de calcular el valor del gradiente en el punto de corte. El valor del gradiente en cada uno de los puntos se calcula utilizando el método de diferencias finitas sobre los *voxels* vecinos de cara de los *voxels* virtuales que incluyen a dichos puntos. Estos vecinos pueden coindidir con *voxels* virtuales o reales. La imagen inferior de la figura 3.35 representa gráficamente la idea para el cálculo del gradiente en el punto de corte.

Una vez determinado el vector gradiente en los puntos P_1' y P_2', solamente queda interpolar linealmente entre ambos para obtener el gradiente en el punto de corte y,

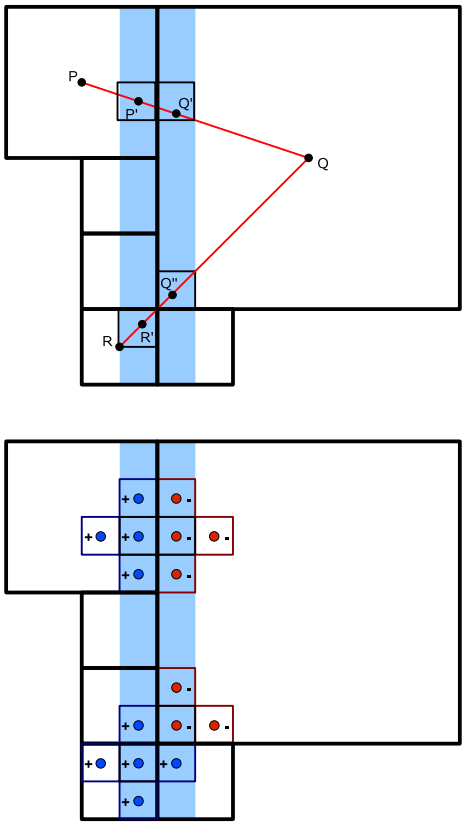

Figura 3.35: La imagen superior muestra los *voxels* virtuales de máxima resolución que contienen a los puntos P',Q' y R',Q'' correspondientes a dos aristas de corte. La imagen inferior presenta los *voxels* virtuales adyacentes a las caras (en este ejemplo 2-D serían los adyacentes a las aristas) de los *voxels* virtuales que incluyen a los puntos P', Q', R', Q'' y que se utilizan para cálcular el gradiente en tales puntos.

posteriormente, normalizando el vector resultante se obtiene la normal en dicho punto. La interpolación de los gradientes de propiedad se lleva a cabo utilizando la siguiente ecuación:

$$\vec{g}_c = \frac{|\overrightarrow{P'_1 P_c}|}{|\overrightarrow{P'_1 P'_2}|}(\vec{g}'_2 - \vec{g}'_1) + \vec{g}'_1 \tag{3.7}$$

donde g_c es el vector gradiente en el punto de corte de la arista y g'_1,g'_2 son los gradientes en los puntos P'_1 y P'_2 respectivamente.

3.6. Evaluación de la representación

Dos de las características más importantes de una representación de volúmenes son sus requerimientos de memoria y el tiempo empleado en generar imágenes. El *IDDG-Octree* se sustenta en una estructura jerárquica espacial que podría ser modificada para indexar el volumen, por ejemplo almacenando los extremos del intervalo de propiedad de cada rama en los nodos interiores. Este tipo de índices se han utilizado para acelerar algoritmos de tipo MC, con el coste de aumentar los requisitos de memoria [VT01]. Por

tanto, nuestro objetivo no es comprobar el uso de nuestra representación como un índice de volumen (si así fuese, obviamente realizaría la extracción de forma más rápida que MC), sino que se pretende comparar su comportamiento en cuanto a espacio y tiempo frente a la utilización de MC sobre la rejilla regular y uniforme, tal y como ha sido definida en las secciones anteriores, sin ninguna modificación adicional. Es decir, en ningún momento se utiliza el *octree* como un índice espacial, sin considerar obviamente que permite la identificación de los *voxels*.

Se han realizado pruebas de nuestro método tomando como entrada tanto rejillas generadas a partir de modelos sintéticos como conjuntos de datos volumétricos. En el caso de los modelos sintéticos se han utilizado dos formas distintas de muestreo para medir el ratio de simplificación que se consigue en el *octree*. La primera genera muestras con igual valor de propiedad en el interior del objeto, y la segunda proporciona muestras con un valor de propiedad decreciente conforme la distancia al centro del objeto aumenta. En concreto, los modelos sintéticos que se han utilizado para las pruebas están definidos de la siguiente forma, según la estrategia de discretización llevada a cabo:

Sea α_1 la función para realizar muestreo en el dominio binario $\{0, 255\}$, definida de la siguiente forma:

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

$$\alpha_1(x, y, z) = \begin{cases} 255 & \text{si } r_i \leq 1 \\ 0 & \text{en otro caso} \end{cases} \tag{3.8}$$

Sea α_2 la función para realizar muestreo en el dominio discreto acotado por el intervalo $[0, 255]$, definida de la siguiente forma:

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

$$\alpha_2(x, y, z) = \begin{cases} 255(1 - r_i) & \text{si } r_i \leq 1 \\ 0 & \text{en otro caso} \end{cases} \tag{3.9}$$

donde V es un cubo de lado 2 centrado en el origen y alineado con los ejes y r_i se define según el modelo sintético escogido como:

Esfera

$$r_1 = \sqrt{x^2 + y^2 + z^2} \tag{3.10}$$

Tronco

$$r_2 = \sqrt{x^2 + 2yz} \tag{3.11}$$

Silla

$$r_3 = \sin(xz) + y + \cos(xz) \tag{3.12}$$

Jarrón

$$r_4 = x\sin(x) + y\cos(y) + z\sin(z) \tag{3.13}$$

Con el objetivo de comparar la sensibilidad de la representación con respecto al tamaño del conjunto de datos se han generado varias representaciones volumétricas con diferentes frecuencias de muestreo para cada una de las estrategias. Para la forma de muestreo que genera muestras en un dominio binario se ha elegido un criterio de similaridad basado en la igualdad de valores. Para el caso del muestreo que genera muestras en un dominio discreto se ha utilizado un valor de tolerancia de error para la comprobación de la igualdad de las muestras que cumplen el criterio de similaridad. Todas las pruebas se han llevado a cabo en un PC estándar con un procesador AMD 64 (2 MHz) y 2 GB de memoria RAM.

3.6.1. Resultados

Las tablas que se presentan a continuación muestran, de izquierda a derecha, una primera columna con las diferentes resoluciones de muestreo de los modelos. A continuación, dos columnas con el número de nodos internos y nodos hoja del *octree* generado a partir de cada tipo de muestreo. Las siguientes columnas muestran el tamaño (en *Kbytes*) ocupado por el *IDDG-Octree* y por la rejilla (Rej.). La última columna expresa el porcentaje de espacio que ocupa nuestra representación con respecto a la rejilla regular y uniforme (*IDDG-O*/Rej.). Se muestran por separado el número de nodos internos y nodos hoja del *octree* porque el espacio que ocupa cada tipo de nodo es diferente. El valor de propiedad en la rejilla regular ocupa 2 *bytes*, mientras que en el *octree* cada nodo interno ocupa 4 *bytes* y cada nodo hoja ocupa: 2 *bytes* para almacenar el valor de propiedad y 1 *byte* para almacenar las responsabilidades (1 *bit* para cada una de las ocho posibles).

Las tablas 3.1–3.3 muestran los resultados relativos a espacio ocupado por el modelo **esfera**. La estrategia de muestreo utilizada produce valores en el intervalo 0 a 255. La tabla 3.1 muestra los resultados obtenidos usando como criterio de similaridad una tolerancia de error en la diferencia entre valores de propiedad $\epsilon = 0$. Este es un ejemplo que sirve para mostrar el peor caso en el que se llega a conseguir una simplificación muy baja del conjunto de muestras inicial debido a que el criterio de similaridad escogido es la igualdad entre valores de propiedad. No obstante, como puede comprobarse, conforme se incrementa el número de muestras la tasa de reducción del espacio requerido mejora. De igual manera, en las tablas 3.2 y 3.3 se muestran los resultados obtenidos para la misma estrategia de muestreo, pero utilizando tolerancias de error $\epsilon = 1$ y $\epsilon = 2$ respectivamente. Como es obvio suponer, al incrementar la tolerancia de error se aumenta la tasa de reducción de espacio para conjuntos de muestras de menor tamaño.

De manera similar, las tablas 3.4–3.6, 3.7–3.9 y 3.10–3.12 muestran los resultados relativos a espacio ocupado por los modelos: **tronco**, **silla** y **jarrón**; para criterios de similaridad con tolerancia de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$. En ellas se puede comprobar la tendencia a la reducción de los requisitos de espacio conforme el número de muestras tomadas aumenta y se relaja el criterio de similaridad entre muestras. Las cuatro gráficas situadas en la parte superior de la figura 3.36 resumen el porcentaje de ahorro de espacio (*IDDG-O*/Rejilla) que consigue el *IDDG-Octree* con respecto a la rejilla regular y uniforme para cada uno de los modelos sintéticos muestreados con el dominio $0 - 255$.

Obviamente, cuando se aplica la estrategia de muestreo binario se obtiene un mayor nivel de simplificación. Además, no es necesario considerar una tolerancia de error a la hora de aplicar el criterio de similaridad entre valores de propiedad, ya que solo disponemos de dos valores posibles para la muestras, indicando uno el interior y otro

Res.	Nodos		Espacio		
	Internos	Hoja	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
32^3	2705	18936	66	64	103
64^3	20329	142304	496	512	96
128^3	158447	1109130	3868	4096	94
256^3	1252837	8769860	30587	32768	93
512^3	1260005	8820036	30762	262144	12

Tabla 3.1: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para diferentes ratios de muestreo del modelo **esfera**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 0$.

Res.	Nodos		Espacio		
	Internos	Hoja	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
32^3	2657	18600	65	64	101
64^3	20147	141030	492	512	96
128^3	156337	1094360	3817	4096	93
256^3	158044	1106309	3858	32768	12
512^3	158960	1112721	3881	262144	1

Tabla 3.2: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para diferentes ratios de muestreo del modelo **esfera**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 1$.

Res.	Nodos		Espacio		
	Internos	Hoja	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
32^3	2609	18264	63	64	99
64^3	19885	139196	485	512	95
128^3	123076	861533	3005	4096	73
256^3	123316	863213	3010	32768	9
512^3	123449	864144	3014	262144	1

Tabla 3.3: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para diferentes ratios de muestreo del modelo **esfera**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 2$.

Res.	Nodos		Espacio		
	Internos	Hoja	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
32^3	2777	19440	68	64	106
64^3	20917	146420	511	512	99
128^3	162077	1134540	3957	4096	97
256^3	1275409	8927864	31138	32768	95
512^3	1282706	8978944	31316	262144	12

Tabla 3.4: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para diferentes ratios de muestreo del modelo **tronco**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 0$.

Res.	Nodos		Espacio		
	Internos	Hoja	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
32^3	2777	19440	68	64	106
64^3	20793	145552	508	512	99
128^3	161357	1129500	3939	4096	96
256^3	819536	5736753	20008	32768	61
512^3	824285	5770002	20124	262144	8

Tabla 3.5: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para diferentes ratios de muestreo del modelo **tronco**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 1$.

Res.	Nodos		Espacio		
	Internos	Hoja	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
32^3	2769	19384	68	64	106
64^3	20743	145202	506	512	99
128^3	149915	1049406	3660	4096	89
256^3	395020	2765141	9644	32768	29
512^3	395446	2768123	9654	262144	4

Tabla 3.6: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para diferentes ratios de muestreo del modelo **tronco**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 2$.

Res.	Nodos		Espacio		
	Internos	Hoja	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
32^3	2635	18446	64	64	100
64^3	20411	142878	498	512	97
128^3	160133	1120932	3909	4096	95
256^3	1266168	8863178	30912	32768	94
512^3	1273412	8913888	31089	262144	12

Tabla 3.7: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para diferentes ratios de muestreo del modelo **silla**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 0$.

Res.	Nodos		Espacio		
	Internos	Hoja	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
32^3	2631	18418	64	64	100
64^3	20323	142262	496	512	97
128^3	159477	1116340	3893	4096	95
256^3	1253862	8777035	30611	32768	93
512^3	437914	3065399	10691	262144	4

Tabla 3.8: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para diferentes ratios de muestreo del modelo **silla**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 1$.

Res.	Nodos		Espacio		
	Internos	Hoja	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
32^3	2629	18404	64	64	100
64^3	20253	141772	494	512	97
128^3	158823	1111762	3877	4096	95
256^3	159269	1114884	3888	32768	12
512^3	159493	1116452	3893	262144	1

Tabla 3.9: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para diferentes ratios de muestreo del modelo **silla**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 2$.

Res.	Nodos		Espacio		
	Internos	Hoja	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
32^3	3649	25544	89	64	139
64^3	28517	199620	696	512	136
128^3	224747	1573230	5487	4096	134
256^3	1742297	12196080	42536	32768	130
512^3	9150254	64051779	223395	262144	85

Tabla 3.10: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para diferentes ratios de muestreo del modelo **jarrón**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 0$.

Res.	Nodos		Espacio		
	Internos	Hoja	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
32^3	3641	25488	89	64	139
64^3	28361	198528	692	512	135
128^3	218276	1527933	5329	4096	130
256^3	1160184	8121289	28325	32768	86
512^3	1148008	8036057	28027	262144	11

Tabla 3.11: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para diferentes ratios de muestreo del modelo **jarrón**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 1$.

Res.	Nodos		Espacio		
	Internos	Hoja	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
32^3	3607	25250	88	64	138
64^3	27971	195798	683	512	133
128^3	202095	1414666	4934	4096	120
256^3	201432	1410025	4918	32768	15
512^3	202003	1414022	4932	262144	2

Tabla 3.12: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para diferentes ratios de muestreo del modelo **jarrón**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 2$.

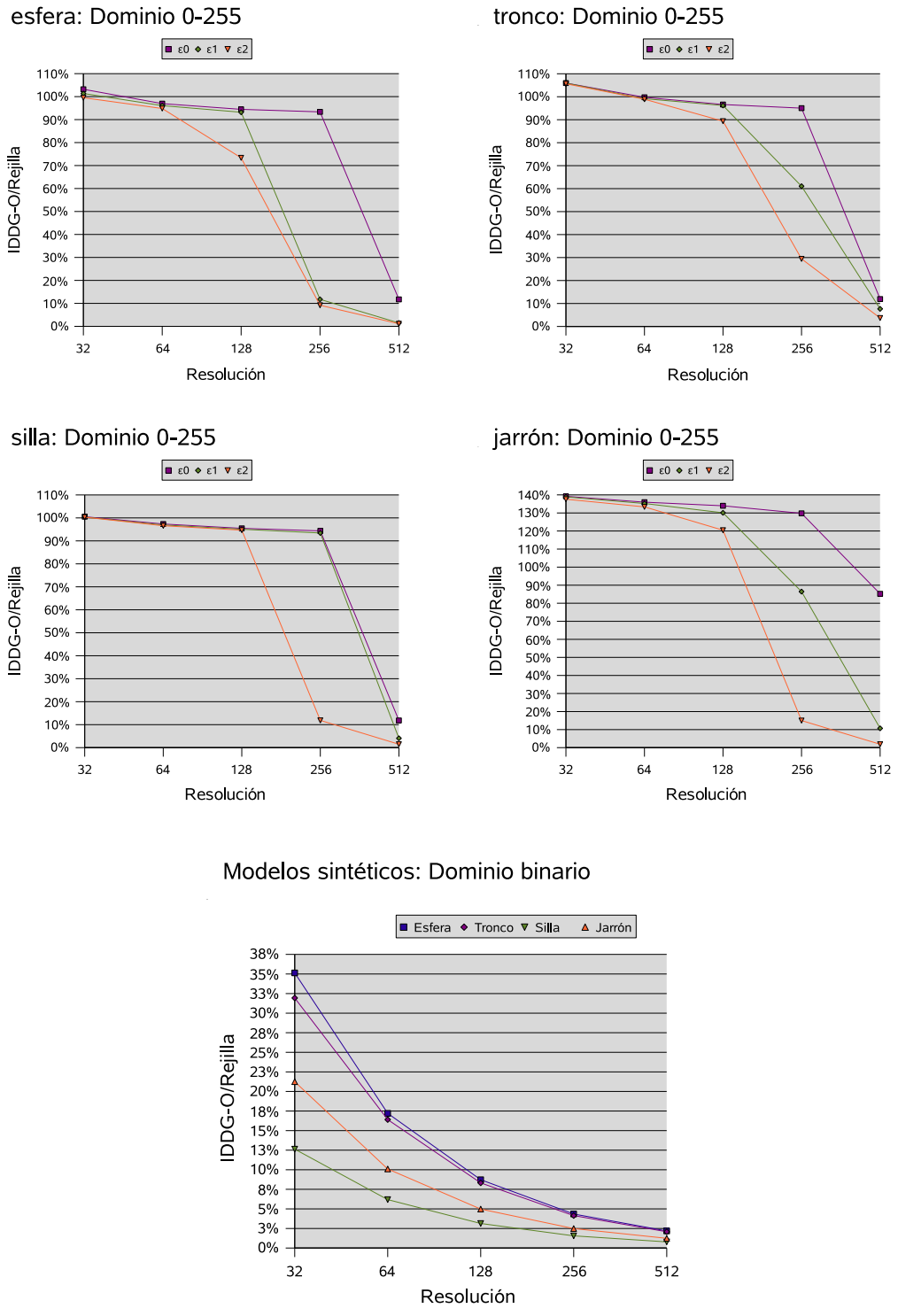

Figura 3.36: Las cuatro gráficas situadas en la parte superior resumen el porcentaje de ahorro de espacio (*IDDG-O*/Rejilla) que consigue el *IDDG-Octree* con respecto a la rejilla regular y uniforme para cada uno de los modelos sintéticos muestreados con un dominio $0 - 255$. La gráfica situada en la parte inferior muestra la misma información para el caso del muestreo con un dominio binario.

Res.	Nodos		Espacio		
	Internos	Hoja	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
32^3	921	6448	22	64	35
64^3	3609	25264	88	512	17
128^3	14641	102488	357	4096	9
256^3	58345	408416	1424	32768	4
512^3	234799	1643594	5732	262144	2

Tabla 3.13: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para diferentes ratios de muestreo del modelo **esfera**. El dominio de valores de propiedad es binario.

Res.	Nodos		Espacio		
	Internos	Hoja	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
32^3	837	5860	20	64	32
64^3	3441	24088	84	512	16
128^3	13959	97717	341	4096	8
256^3	55627	389403	1358	32768	4
512^3	223860	1567079	5466	262144	2

Tabla 3.14: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para diferentes ratios de muestreo del modelo **tronco**. El dominio de valores de propiedad es binario.

el exterior del objeto. Los resultados de aplicar esta estrategia a los modelos sintéticos se muestran en las tablas 3.13–3.16. La gráfica situada en la parte inferior de la figura 3.36 resume el porcentaje de ahorro de espacio (*IDDG-O*/Rejilla) que consigue el *IDDG-Octree* con respecto a la rejilla regular y uniforme para cada uno de los modelos sintéticos muestreados con un dominio binario.

Al utilizar la segunda estrategia de muestreo, los requerimientos de espacio en memoria de las diferentes resoluciones obtenidas son considerablemente más bajos que los obtenidos con el primer enfoque. Es lógico esperar este resultado ya que, en el enfoque de muestreo con dominio discreto los datos tienen valores de propiedad diferentes en la parte de la rejilla de muestras que representa el interior del objeto. Obviamente, debido a nuestro criterio de similaridad, se van a obtener tasas más elevadas de ahorro de espacio conforme existan más regiones con valores de propiedad similar y mayor sea el tamaño de dichas regiones.

Las tablas que se presentan a continuación muestran los resultados en relación al tiempo empleado (en segundos) en los diferentes procesamientos realizados sobre el *IDDG-Octree* y la rejilla. De nuevo se utilizan diferentes ratios de muestreo de los modelos sintéticos para cada una las dos estrategias de muestreo. Concretamente, con respecto al *IDDG-Octree* se presentan columnas (de izquierda a derecha) que muestran el tiempo empleado en: el proceso de simplificación del *octree* (Simplif.), el proceso de generación de la rejilla dual definida implícitamente (Rejilla Dual) y el proceso de ex-

Res.	Nodos		Espacio		
	Internos	Hoja	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
32^3	331	2318	8	64	13
64^3	1297	9082	32	512	6
128^3	5261	36842	128	4096	3
256^3	20965	146816	512	32768	2
512^3	84369	590833	2060	262144	1

Tabla 3.15: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para diferentes ratios de muestreo del modelo **silla**. El dominio de valores de propiedad es binario.

Res.	Nodos		Espacio		
	Internos	Hoja	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
32^3	557	3900	14	64	21
64^3	2117	14820	52	512	10
128^3	8353	58472	204	4096	5
256^3	33409	233864	815	32768	2
512^3	133848	936937	3268	262144	1

Tabla 3.16: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para diferentes ratios de muestreo del modelo **jarrón**. El dominio de valores de propiedad es binario.

Res.	*IDDG-Octree*			Rejilla
	Simplif. (s.)	Rejilla dual (s.)	Extracción (s.)	Extracción (s.)
32^3	0,02	0,25	0,54	0,15
64^3	0,17	1,43	1,68	1,04
128^3	1,14	7,89	12,97	3,88
256^3	4,79	68,11	95,56	26,68
512^3	44,36	70,15	100,58	216,14

Tabla 3.17: Tiempos de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para diferentes ratios de muestreo del modelo **esfera**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 0$.

Res.	*IDDG-Octree*			Rejilla
	Simplif. (s.)	Rejilla dual (s.)	Extracción (s.)	Extracción (s.)
32^3	0,02	0,25	0,54	0,15
64^3	0,16	1,34	1,67	1,04
128^3	1,06	8,45	12,13	3,88
256^3	5,39	8,16	13,47	26,68
512^3	48,63	8,46	14,67	216,14

Tabla 3.18: Tiempos de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para diferentes ratios de muestreo del modelo **esfera**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 1$.

tracción de la isosuperficie (Extracción). Con respecto a la rejilla se muestra una única columna con el tiempo empleado en el proceso de extracción de la isosuperficie (Extracción).

Las tablas 3.17–3.19 muestran los resultados relativos a tiempo de procesamiento empleado para el modelo **esfera**. La estrategia de muestreo utilizada produce valores en el intervalo 0 a 255. La tabla 3.17 muestra los resultados obtenidos usando como criterio de similaridad una tolerancia de error en la diferencia entre valores de propiedad $\epsilon = 0$. De la misma forma, las tablas 3.18 y 3.19 muestran los resultados obtenidos para la misma estrategia de muestreo, pero utilizando tolerancias de error $\epsilon = 1$ y $\epsilon = 2$ respectivamente. Como puede observarse, nuestro método de extracción emplea un tiempo mayor que el algoritmo de extractión aplicado sobre la rejilla para ratios de muestreo inferiores a 512^3 muestras. Sin embargo, conforme aumenta la tolerancia de error puede comprobarse que este tiempo mejora sustancialmente. Esto es razonable ya que al incrementar la tolerancia de error se obtiene un menor número de *voxels* y por consiguiente un menor número de celdas duales, con lo que se reduce el número de celdas a procesar.

De manera similar, las tablas 3.20–3.22, 3.23–3.25 y 3.26–3.28 muestran los resultados relativos a tiempos de procesamiento empleado por los modelos: **tronco**, **silla** y **jarrón**; para criterios de similaridad con tolerancia de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$. En ellas se puede comprobar la tendencia a la mejora en el tiempo de extracción del

Res.	*IDDG-Octree*			Rejilla
	Simplif. (s.)	Rejilla dual (s.)	Extracción (s.)	Extracción (s.)
32^3	0,02	0,25	0,54	0,15
64^3	0,16	1,39	1,67	1,04
128^3	0,88	7,28	11,99	3,88
256^3	6,13	6,86	10,47	26,68
512^3	45,43	7,03	12,56	216,14

Tabla 3.19: Tiempos de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para diferentes ratios de muestreo del modelo **esfera**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 2$.

Res.	*IDDG-Octree*			Rejilla
	Simplif. (s.)	Rejilla dual (s.)	Extracción (s.)	Extracción (s.)
32^3	0,02	0,26	0,77	0,10
64^3	0,16	1,67	2,18	1,04
128^3	0,88	7,98	14,68	4,06
256^3	4,77	69,02	108,95	28,11
512^3	44,17	71,09	114,67	220,07

Tabla 3.20: Tiempos de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para diferentes ratios de muestreo del modelo **tronco**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 0$.

IDDG-Octree conforme el número de muestras tomadas aumenta y se relaja el criterio de similaridad entre muestras. Las cuatro gráficas situadas en la parte superior de la figura 3.37 resumen el tiempo, en segundos, empleado en la extracción de la isosuperficie(Extracción (s.)) por el *IDDG-Octree* y la rejilla regular y uniforme para cada uno de los modelos sintéticos muestreados con el dominio $0 - 255$.

Obviamente, cuando se aplica la estrategia de muestreo binario se obtiene un mayor nivel de simplificación en el *octree*. Como consecuencia, se obtiene un menor número de celdas duales y, por consiguiente, el tiempo de extracción del *IDDG-Octree* mejora notablemente con respecto al tiempo de extracción asociado a la rejilla. Los resultados de aplicar esta estrategia a los modelos sintéticos se muestran en las tablas 3.29–3.32. La gráfica situada en la parte inferior de la figura 3.37 resume el tiempo, en segundos, empleado en la extracción de la isosuperficie (Extracción (s.)) por el *IDDG-Octree* y la rejilla regular y uniforme para cada uno de los modelos sintéticos muestreados con un dominio binario.

Como se puede comprobar en las tablas presentadas, nuestra representación gasta un tiempo extra en generar sobre la marcha la geometría de las celdas necesarias para realizar la extracción (sin utilizar el *octree* como un índice espacial), con respecto a realizar la extracción sobre una rejilla regular. Sin embargo, si el número de celdas que es necesario procesar se reduce de forma parecida a lo que le ocurre a los modelos representados mediante un conjunto de 512^3 muestras (recogidos en la quinta fila de las

Res.	*IDDG-Octree*			Rejilla
	Simplif. (s.)	Rejilla dual (s.)	Extracción (s.)	Extracción (s.)
32^3	0,01	0,26	0,77	0,10
64^3	0,17	1,14	2,81	1,04
128^3	1,15	7,98	16,07	4,06
256^3	5,06	45,64	74,03	28,11
512^3	45,65	47,32	80,62	220,07

Tabla 3.21: Tiempos de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para diferentes ratios de muestreo del modelo **tronco**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 1$.

Res.	*IDDG-Octree*			Rejilla
	Simplif. (s.)	Rejilla dual (s.)	Extracción (s.)	Extracción (s.)
32^3	0,02	0,27	0,76	0,10
64^3	0,16	1,83	2,20	1,04
128^3	1,29	7,57	13,94	4,06
256^3	5,26	23,43	40,10	28,11
512^3	38,98	24,01	48,10	220,07

Tabla 3.22: Tiempos de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para diferentes ratios de muestreo del modelo **tronco**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 2$.

Res.	*IDDG-Octree*			Rejilla
	Simplif. (s.)	Rejilla dual (s.)	Extracción (s.)	Extracción (s.)
32^3	0,02	0,21	0,45	0,07
64^3	0,16	1,11	1,50	0,48
128^3	0,94	8,11	13,67	3,79
256^3	3,95	70,00	100,72	26,57
512^3	41,58	77,10	111,01	209,34

Tabla 3.23: Tiempos de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para diferentes ratios de muestreo del modelo **silla**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 0$.

Res.	*IDDG-Octree*			Rejilla
	Simplif. (s.)	Rejilla dual (s.)	Extracción (s.)	Extracción (s.)
32^3	0,02	0,21	0,43	0,07
64^3	0,17	1,23	1,49	0,48
128^3	0,79	8,10	11,58	3,79
256^3	5,47	72,10	100,81	26,57
512^3	44,49	24,55	36,62	209,34

Tabla 3.24: Tiempo de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para diferentes ratios de muestreo del modelo **silla**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 1$.

Res.	*IDDG-Octree*			Rejilla
	Simplif. (s.)	Rejilla dual (s.)	Extracción (s.)	Extracción (s.)
32^3	0,02	0,21	0,44	0,07
64^3	0,16	1,22	1,49	0,48
128^3	0,84	7,36	12,26	3,79
256^3	5,34	7,66	12,00	26,57
512^3	44,76	7,91	14,00	209,34

Tabla 3.25: Tiempos de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para diferentes ratios de muestreo del modelo **silla**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 2$.

Res.	*IDDG-Octree*			Rejilla
	Simplif. (s.)	Rejilla dual (s.)	Extracción (s.)	Extracción (s.)
32^3	0,02	0,29	0,66	0,15
64^3	0,16	1,60	2,16	1,05
128^3	1,21	10,78	16,75	3,70
256^3	4,54	91,92	133,74	27,69
512^3	40,9	579,35	771,99	209,93

Tabla 3.26: Tiempos de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para diferentes ratios de muestreo del modelo **jarrón**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 0$.

Res.	*IDDG-Octree*			Rejilla
	Simplif. (s.)	Rejilla dual (s.)	Extracción (s.)	Extracción (s.)
32^3	0,02	0,30	0,65	0,15
64^3	0,15	2,23	2,14	1,05
128^3	1,23	10,41	16,20	3,70
256^3	4,78	62,57	91,23	27,69
512^3	44,85	63,82	95,35	209,93

Tabla 3.27: Tiempos de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para diferentes ratios de muestreo del modelo **jarrón**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 1$.

Res.	*IDDG-Octree*			Rejilla
	Simplif. (s.)	Rejilla dual (s.)	Extracción (s.)	Extracción (s.)
32^3	0,01	0,30	0,66	0,15
64^3	0,15	1,72	2,15	1,05
128^3	1,16	9,68	15,20	3,70
256^3	5,38	10,03	15,76	27,69
512^3	44,26	10,30	18,36	209,93

Tabla 3.28: Tiempos de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para diferentes ratios de muestreo del modelo **jarrón**. El dominio de valores de propiedad es $0 - 255$ y la tolerancia de error es $\epsilon = 2$.

Res.	*IDDG-Octree*			Rejilla
	Simplif. (s.)	Rejilla dual (s.)	Extracción (s.)	Extracción (s.)
32^3	0,03	0,15	0,54	0,19
64^3	0,18	0,63	1,14	0,95
128^3	1,09	1,54	5,26	4,12
256^3	5,39	6,96	23,43	27,90
512^3	46,85	31,06	104,04	215,19

Tabla 3.29: Tiempo de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para diferentes ratios de muestreo del modelo **esfera**. El dominio de valores de propiedad es binario.

Res.	*IDDG-Octree*			Rejilla
	Simplif. (s.)	Rejilla dual (s.)	Extracción (s.)	Extracción (s.)
32^3	0,02	0,15	0,52	0,16
64^3	0,19	0,69	1,54	1,13
128^3	1,12	1,62	5,78	3,96
256^3	5,54	7,32	25,75	28,62
512^3	48,15	32,67	114,34	215,10

Tabla 3.30: Tiempos de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para diferentes ratios de muestreo del modelo **tronco**. El dominio de valores de propiedad es binario.

Res.	*IDDG-Octree*			Rejilla
	Simplif. (s.)	Rejilla dual (s.)	Extracción (s.)	Extracción (s.)
32^3	0,02	0,06	0,21	0,13
64^3	0,19	0,35	0,75	0,94
128^3	1,12	1,13	4,25	3,50
256^3	5,45	4,48	16,33	26,43
512^3	47,02	28,79	101,02	208,41

Tabla 3.31: Tiempos de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para diferentes ratios de muestreo del modelo **silla**. El dominio de valores de propiedad es binario.

Res.	*IDDG-Octree*			Rejilla
	Simplif. (s.)	Rejilla dual (s.)	Extracción (s.)	Extracción (s.)
32^3	0,02	0,09	0,32	0,14
64^3	0,18	0,43	0,86	0,98
128^3	0,78	0,88	3,01	4,11
256^3	5,45	4,00	13,59	26,81
512^3	50,65	17,85	60,31	212,68

Tabla 3.32: Tiempos de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para diferentes ratios de muestreo del modelo **jarrón**. El dominio de valores de propiedad es binario.

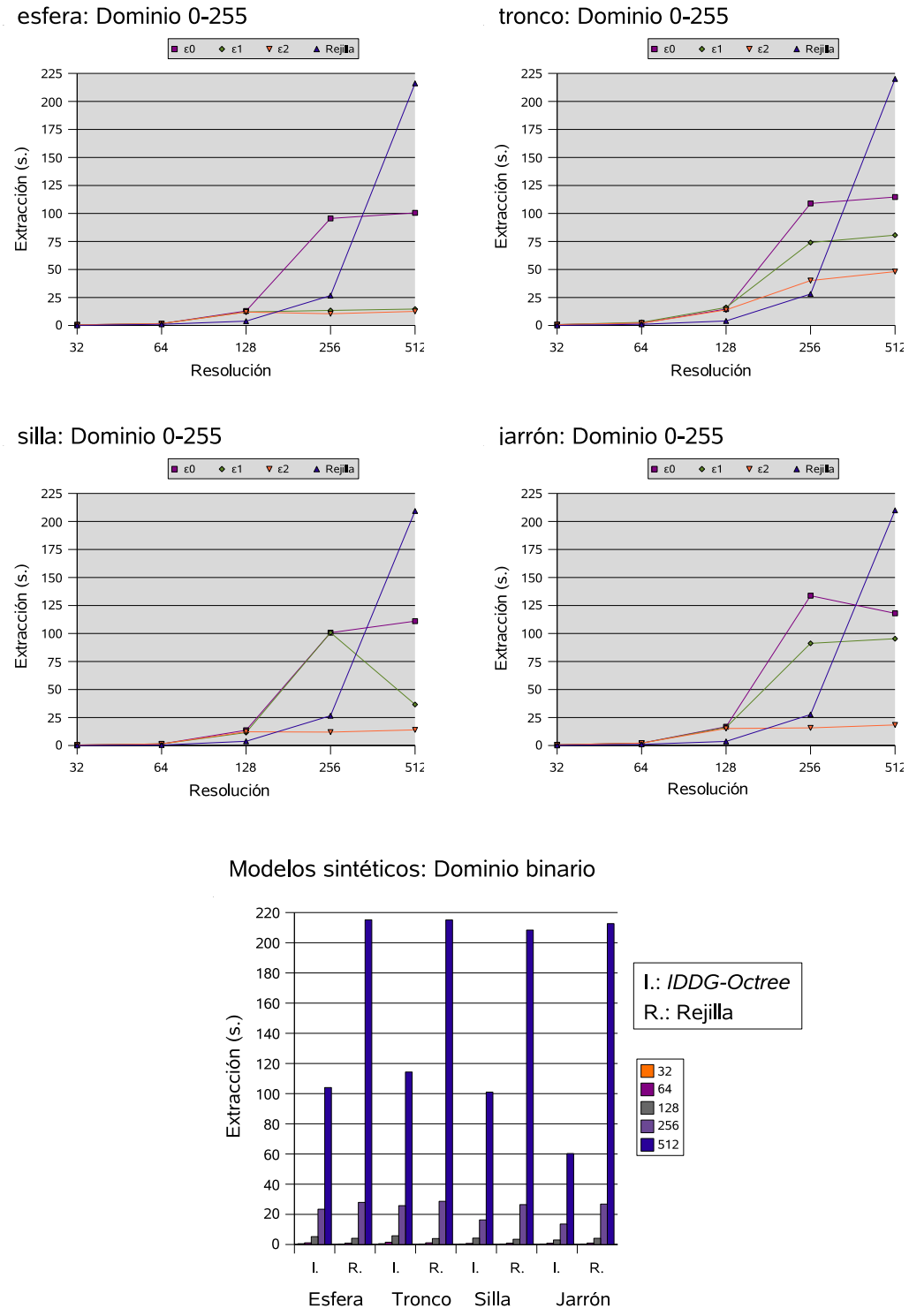

Figura 3.37: Las cuatro gráficas situadas en la parte superior resumen el tiempo, en segundos, empleado en la extracción de la isosuperficie(Extracción (s.)) por el *IDDG-Octree* y la rejilla regular y uniforme para cada uno de los modelos sintéticos muestreados con el dominio $0-255$. La gráfica situada en la parte inferior muestra la misma información para el caso del muestreo con un dominio binario.

Data set	Espacio		
Nombre: Resolución	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
aneurism: 256x256x256	2044	32768	6
bonsai: 256x256x256	16788	32768	51
lobster: 301x324x56	13585	10666	127
skull: 256x256x256	55366	32768	169

Tabla 3.33: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para los cuatro *datasets* de prueba. La tolerancia de error es $\epsilon = 0$.

Data set	Espacio		
Nombre: Resolución	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
aneurism: 256x256x256	1972	32768	6
bonsai: 256x256x256	15587	32768	48
lobster: 301x324x56	10296	10666	97
skull: 256x256x256	54720	32768	167

Tabla 3.34: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para los cuatro *datasets* de prueba. La tolerancia de error es $\epsilon = 1$.

tablas), o se relaja la tolerancia de error para el criterio de similaridad como por ejemplo ocurre en las tablas que muestran los valores de los modelos **silla** y **jarrón** para $\epsilon = 2$ (3.25 y 3.28), entonces nuestra representación requiere un tiempo de extracción menor que la rejilla. Este hecho puede comprobarse también en los modelos discretizados utilizando la estrategia de muestreo binario, para los valores de muestras 256^3 y 512^3.

A continuación, se analizan los requisitos de espacio y los tiempos de procesamiento para varios ejemplos de conjuntos de datos volumétricos. Para ello se van a usar cuatro *datasets* obtenidos del repositorio volren (`http://www.volren.org`). De manera análoga al análisis realizado para el caso de los modelos sintéticos presentado anteriormente, las tablas siguientes muestran el espacio requerido por ambas representaciones, rejilla regular y uniforme e *IDDG-Octree*, y el tiempo empleado en la construcción del *octree* simplificado, en el proceso de asignación de responsabilidades, las cuales definen implícitamente la rejilla dual de celdas, y en la extracción de la isosuperficie a partir de nuestra representación y de la rejilla regular.

Concretamente, con respecto al espacio de almacenamiento requerido, las tablas 3.33–3.35 muestran el espacio, en *Kbytes*, ocupado por el *IDDG-Octree* (*IDDG-O*), por la rejilla (Rej.) y el porcentaje de espacio que ocupa nuestra representación con respecto a la rejilla regular (*IDDG-O*/Rej.). La tabla 3.33 muestra los resultados obtenidos utilizando una tolerancia de error $\epsilon = 0$, y las tablas 3.34 y 3.35 muestran los resultados para las tolerancias de error $\epsilon = 1$ y $\epsilon = 2$ respectivamente. La gráfica situada en la parte izquierda de la figura 3.38 resume el porcentaje de ahorro de espacio (*IDDG-O*/Rejilla) que consigue el *IDDG-Octree* con respecto a la rejilla regular y uniforme para cada uno de los *datasets*.

Las tablas 3.33–3.35 muestran que los *datasets aneurism* y *bonsai* requieren un es-

Data set	Espacio		
Nombre: Resolución	*IDDG-O* (KB)	Rej. (KB)	*IDDG-O*/Rej. (%)
aneurism: 256x256x256	1904	32768	6
bonsai: 256x256x256	14735	32768	45
lobster: 301x324x56	6538	10666	61
skull: 256x256x256	52473	32768	160

Tabla 3.35: Espacio de almacenamiento requerido por la rejilla de muestras y por el *IDDG-Octree* para los cuatro *datasets* de prueba. La tolerancia de error es $\epsilon = 2$.

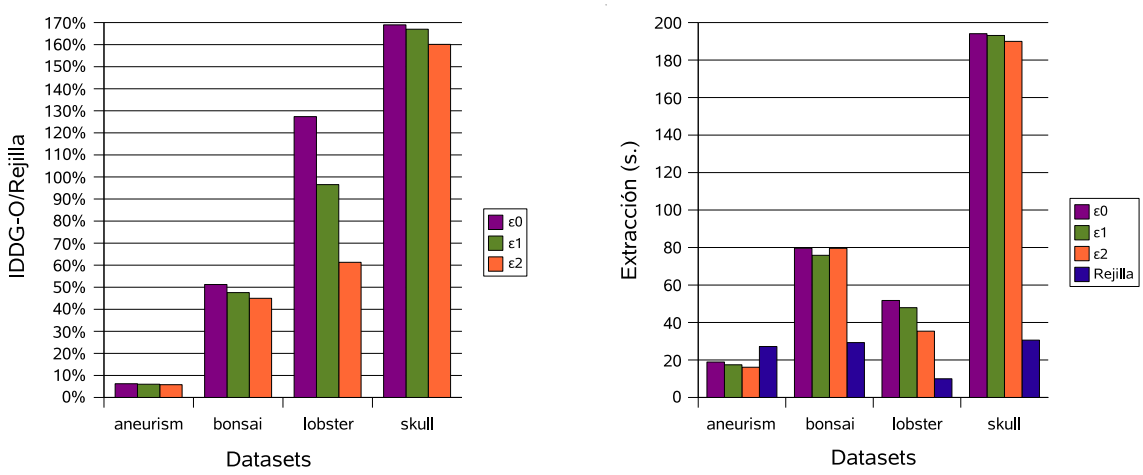

Figura 3.38: La gráfica situada en la parte izquierda resume el porcentaje de ahorro de espacio (*IDDG-O*/Rejilla) que consigue el *IDDG-Octree* con respecto a la rejilla regular y uniforme para cada uno de los *datasets*. La gráfica situada en la parte derecha resume el tiempo, en segundos, empleado en la extracción de la isosuperficie(Extracción (s.)) por el *IDDG-Octree* y la rejilla regular y uniforme para cada *dataset*.

pacio de almacenamiento considerablemente menor en nuestra representación que la rejilla uniforme y regular. Sin embargo, el *dataset skull* requiere un espacio mucho mayor. La razón de este resultado estriba en que los primeros no contienen ruido en la parte exterior de los objetos representados y, por consiguiente, el criterio de similaridad proporciona unas tasas de agregación muy altas en esta zona. A pesar de eso, en el caso del *bonsai* existen muchos datos de alta frecuencia, por lo que el *octree* no puede llegar a tasas de simplificación tan altas como en el *aneurism*. En los casos del tipo que ejemplifica el *dataset skull* se puede optar por una solución que tenga en cuenta más información del conjunto de muestras con el objetivo de eliminar ruido. En el caso de conocer el conjunto de intervalos en el dominio de valores de propiedad que son relevantes para establecer los umbrales se puede llevar a cabo un filtrado previo del volumen.

Las tablas 3.36–3.38 presentan los resultados en relación al tiempo de procesamiento empleado (en segundos). Concretamente, con respecto al *IDDG-Octree* se presentan columnas (de izquierda a derecha) que muestran el tiempo empleado en el proceso de simplificación del *octree* (Simplif.), el proceso de generación de la rejilla dual definida

Nombre: Resolución	*IDDG-Octree*			Rejilla
	Simplif. (s.)	R. dual (s.)	Extrac. (s.)	Extrac. (s.)
aneurism: 256x256x256	5,42	9,24	18,92	27,19
bonsai: 256x256x256	5,12	51,00	79,61	29,30
lobster: 301x324x56	45,36	35,19	51,74	9,98
skull: 256x256x256	4,25	125,33	194,09	30,61

Tabla 3.36: Tiempos de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para los cuatro *datasets* de prueba. La tolerancia de error es $\epsilon = 0$.

Nombre: Resolución	*IDDG-Octree*			Rejilla
	Simplif. (s.)	R. dual (s.)	Extrac. (s.)	Extrac. (s.)
aneurism: 256x256x256	6,77	10,26	17,43	27,19
bonsai: 256x256x256	5,08	49,17	75,85	29,30
lobster: 301x324x56	45,56	37,82	47,90	9,98
skull: 256x256x256	4,30	126,57	193,14	30,61

Tabla 3.37: Tiempos de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para los cuatro *datasets* de prueba. La tolerancia de error es $\epsilon = 1$.

implícitamente (R. Dual) y el proceso de extracción de la isosuperficie (Extrac.). Con respecto a la rejilla se muestra una única columna con el tiempo empleado en el proceso de extracción de la isosuperficie (Extrac.). La gráfica situada en la parte derecha de la figura 3.38 resume el tiempo, en segundos, empleado en la extracción de la isosuperficie (Extracción (s.)) por el *IDDG-Octree* y la rejilla regular y uniforme para cada uno de los *datasets*.

Con respecto al tiempo de extracción a partir de conjuntos de datos volumétricos, se puede observar una situación similar a la que se produce en el caso de los modelos sintéticos. Nuestra representación emplea un tiempo extra en el proceso de extracción de isosuperficie si no se obtiene una tasa de simplificación elevada en el *IDDG-Octree*, como es el caso del *dataset aneurism*. En la primera fila de las tablas 3.33–3.35 se muestra el ahorro de espacio conseguido por *IDDG-Octree* para este *dataset* y en la

Nombre: Resolución	*IDDG-Octree*			Rejilla
	Simplif. (s.)	R. dual (s.)	Extrac. (s.)	Extrac. (s.)
aneurism: 256x256x256	5,48	8,57	16,15	27,19
bonsai: 256x256x256	5,15	49,50	79,60	29,30
lobster: 301x324x56	44,29	27,14	35,39	9,98
skull: 256x256x256	4,57	131,07	189,99	30,61

Tabla 3.38: Tiempos de procesamiento (en segundos) empleados en la rejilla de muestras y en el *IDDG-Octree* para los cuatro *datasets* de prueba. La tolerancia de error es $\epsilon = 2$.

primera fila de las tablas 3.36–3.38 se muestra el tiempo de extracción empleado por nuestro método. En este caso, nuestra representación aventaja a la rejilla regular y uniforme, en cuanto al ahorro de espacio, y al algoritmo Marching Cubes aplicado sobre ésta, en relación al tiempo de extracción de isosuperficie.

En las siguientes figuras (3.39 y 3.40) se muestran las imágenes que genera nuestro método de extracción aplicado al *IDDG-Octree* y las generadas por el algoritmo Marching Cubes aplicado a la rejilla regular y uniforme. Se muestran ejemplos para cada modelo sintético utilizando las dos estrategias de muestreo para una misma resolución de 128^3 muestras. En concreto, de izquierda a derecha, en las tres primeras columnas, se muestran las imágenes generadas por nuestro método de extracción aplicado al *IDDG-Octree*, utilizando para el criterio de similaridad tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$, respectivamente. La cuarta columna muestra el resultado de aplicar el algoritmo Marching Cubes sobre la rejilla regular y uniforme de resolución 128^3. La primera fila dedicada a cada uno de los modelos solamente presenta imágenes para la columna $\epsilon = 0$ y la columna de la rejilla debido a que representa el muestreo binario y, en éste, no tiene sentido una tolerancia de error distinta a $\epsilon = 0$.

En las figuras 3.39 y 3.40 se muestran dos filas por cada modelo sintético. La primera fila muestra el caso de muestreo binario y la segunda el caso de muestreo en el intervalo $[0, 255]$. Todas las imágenes correspondientes a la primera fila de cada modelo sintético presentan un "efecto escalera" debido a que sus valores de propiedad pertenecen al dominio binario. Este efecto se elimina cuando el objeto sintético se muestrea usando diferentes valores de propiedad, como puede comprobarse en las imágenes de la segunda fila. Se puede observar que las imágenes generadas por *Marching Cubes* (cuarta columna) y las generadas por nuestro método usando un *IDDG-Octree* con tolerancias de error $\epsilon = 0$ y $\epsilon = 1$ (dos primeras columnas) son difíciles de distinguir visualmente. Sin embargo, cuando se aplica una tolerancia de error $\epsilon = 2$ (tercera columna) se pueden observar las diferencias que presenta la isosuperficie extraida a partir del *IDDG-Octree* con respecto a la extraida a partir de la rejilla regular y uniforme, salvo en el caso del modelo **silla** en el que no se distinguen diferencias apreciables.

Con respecto a los *datasets* de prueba, en la figura 3.41 se muestra una fila para cada *dataset* con la misma configuración de columnas que en el caso de los modelos sintéticos. En este caso, no se pueden distinguir diferencias apreciables entre las imágenes generadas a partir del *IDDG-Octree* y las generadas a partir de la rejilla regular, ni siquiera en el caso de la tolerancia de error $\epsilon = 2$ (tercera columna).

3.7. Evaluación de la calidad de la isosuperficie extraida

Con el objetivo de evaluar la bondad de la aproximación a la isosuperficie vamos a comparar los resultados que se obtienen realizando la extracción sobre *IDDG-Octree* con los que se obtienen utilizando el algoritmo *Marching Cubes* sobre la rejilla regular y uniforme utilizando el mismo valor umbral. En ambos métodos se obtienen puntos de la isosuperficie utilizando una discretización del espacio mediante celdas. Los puntos de la isosuperficie obtenidos se corresponden con los puntos de corte de ésta con las aristas de las celdas cuyos valores de propiedad en los extremos caen uno por encima y otro por debajo del umbral escogido.

Sin embargo, las celdas obtenidas para cada tipo de partición son diferentes, por lo que no podemos establecer comparaciones en cada uno de los puntos de corte con las aristas obtenidos, ya que estos serán diferentes. Por tanto, hemos optado por utilizar medidas del error que tengan en cuenta el número de puntos que están a una determi-

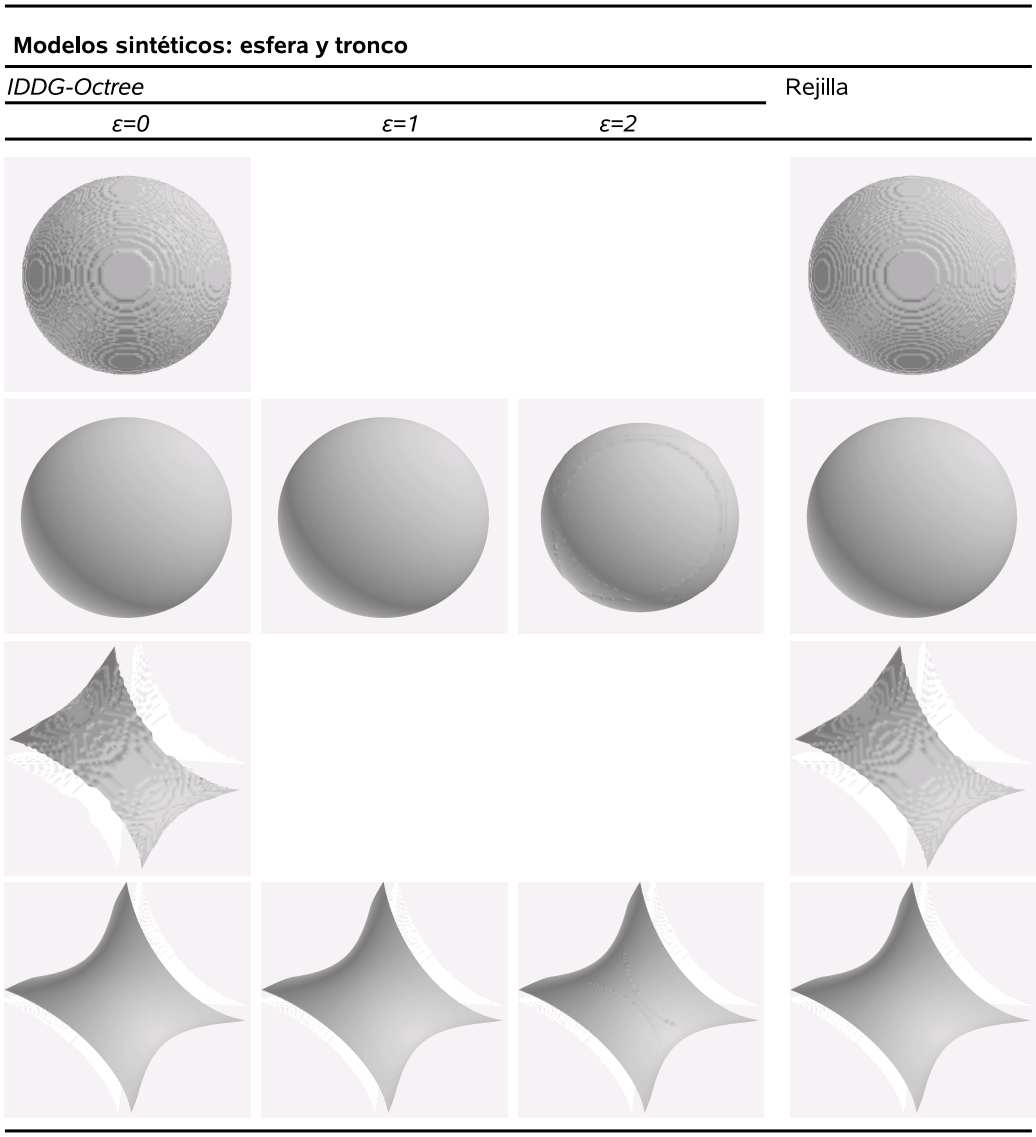

Figura 3.39: Imágenes de las isosuperficies extraidas usando las dos estrategias de muestreo para los modelos **esfera** y **tronco**. De izquierda a derecha, las imágenes situadas en las tres primeras columnas han sido generadas a partir de *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$, respectivamente. La cuarta columna muestra las imágenes generadas utilizando Marching Cubes sobre la rejilla regular y uniforme. La resolución de muestreo ha sido 128^3 para todas las imágenes.

nada distancia del punto correcto situado sobre la isosuperficie. En otras palabras, se discretiza el rango de la función de error y para cada punto de corte obtenido se calcula su valor de error, incrementando el contador del intervalo correspondiente. Al final se obtiene una distribución discreta del error que puede servir de elemento de comparación de la bondad de la aproximación a la isosuperficie en cada método. Nótese no obstante que el número de puntos de corte obtenidos en cada método es distinto, ya que depende de la densidad de celdas de cada representación.

La estrategia para realizar la comparación se basa en disponer de una función conocida $f(x, y, z)$ que permite calcular el valor de propiedad en todo el volumen, junto con

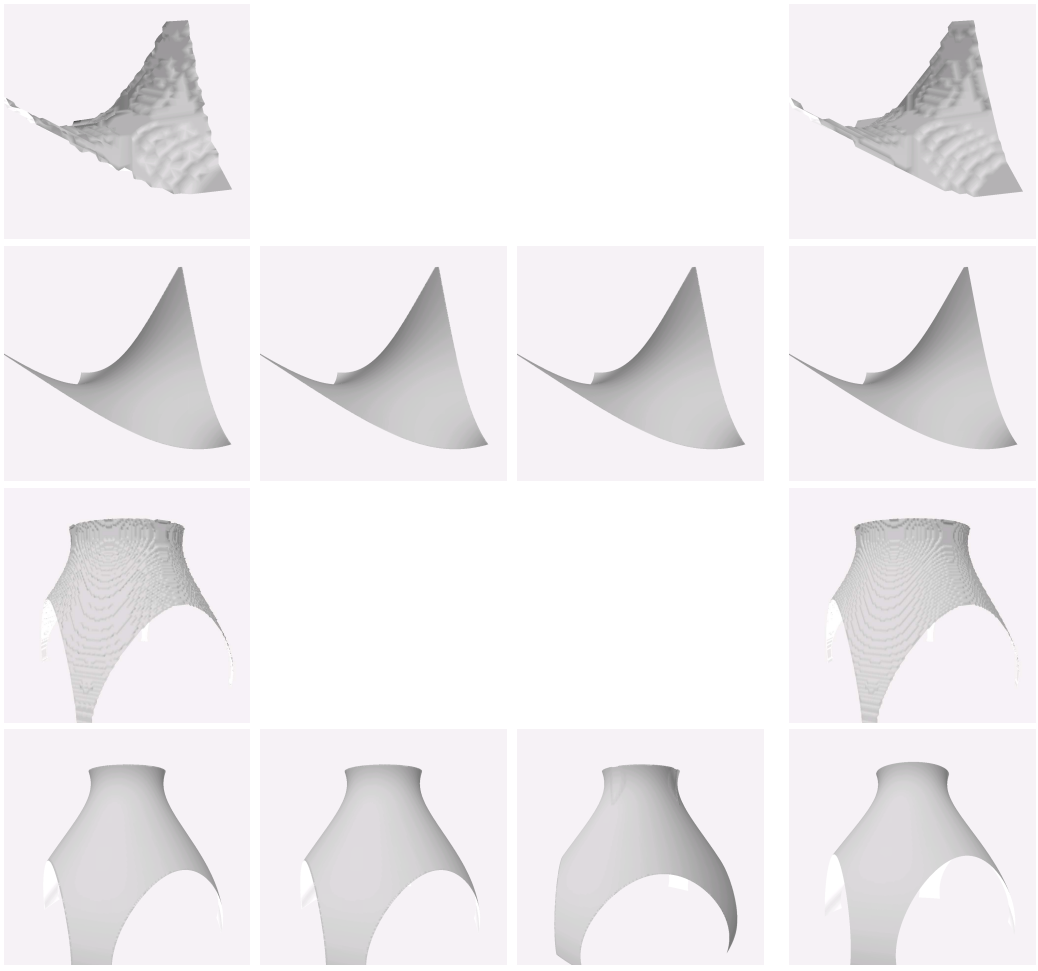

Figura 3.40: Imágenes de las isosuperficies extraidas usando las dos estrategias de muestreo para los modelos **silla** y **jarrón**. De izquierda a derecha, las imágenes situadas en las tres primeras columnas han sido generadas a partir de *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$, respectivamente. La cuarta columna muestra las imágenes generadas utilizando Marching Cubes sobre la rejilla regular y uniforme. La resolución de muestreo ha sido 128^3 para todas las imágenes.

las funciones $F_1(x, y, z)$ y $F_2(x, y, z)$ que son las funciones de interpolación que aplican la representación de rejilla regular y el *IDDG-Octree* respectivamente.

Para seguir esta estrategia de comparación es necesario disponer de un objeto analítico base que nos permita conocer perfectamente el valor de propiedad para cada punto simplemente evaluando dicho punto en la expresión analítica. De esta forma, se utilizan los dos métodos de representación/extracción sobre la misma discretización (voxelización) inicial, pudiéndose estudiar el error cometido sobre un punto en ambos métodos. Para establecer el error cometido en la aproximación se considera como valor correcto

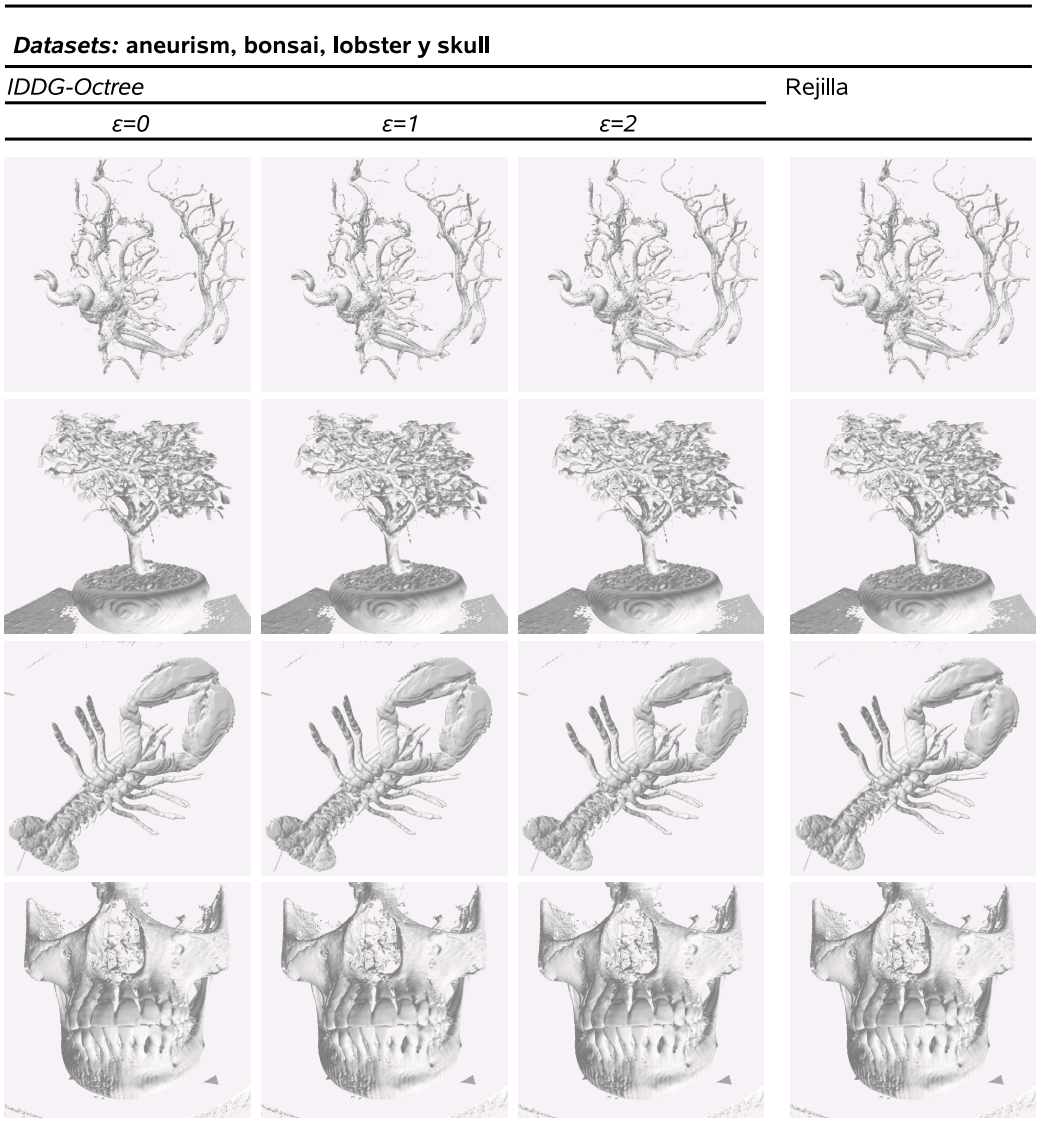

Figura 3.41: Imágenes de las isosuperficies extraidas a partir del *IDDG-Octree* y la rejilla para los *datasets* de prueba. De izquierda a derecha, las imágenes situadas en las tres primeras columnas han sido generadas a partir de *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$, respectivamente. La cuarta columna muestra las imágenes generadas utilizando Marching Cubes sobre la rejilla regular y uniforme.

el proporcionado por la expresión analítica del objeto en el punto de interés y, como valor estimado, el proporcionado por cada uno de los métodos de extracción.

Con respecto a la aplicación de nuestra estrategia de comparación a conjuntos de datos volumétricos, se presenta el inconveniente de que no se conoce la función $f(x, y, z)$. Para solventarlo, se va a considerar que dicha función la constituye la función de interpolación definida por partes $F_1(x, y, z)$ sobre la rejilla a máxima resolución.

3.7.1. Estudio del error

El error cometido por *IDDG-Octree* en la aproximación de $f(x, y, z)$ con respecto a la aproximación conseguida mediante la rejilla regular viene determinado por la acumulación de dos errores: un error asociado a los valores de propiedad asignados a los vértices de las celdas duales y un error asociado a las aristas de dichas celdas. En el primer caso, las sucesivas operaciones de agregación provocan una pérdida de información en el dominio de valores de propiedad debido a la sustitución del valor de los *voxels* agregados por la media de dichos valores en el *voxel* resultante. En el segundo caso, los segmentos de arista en donde se realiza realmente la interpolación pueden tener una orientación distinta, y debido a esto un tamaño distinto, a las aristas de las celdas de la rejilla regular y uniforme. La figura 3.42 ilustra la idea mediante un ejemplo 2-D. La imagen de la izquierda muestra el resultado de agregar cuatro *voxels* de máxima resolución en uno de mayor tamaño, con la consiguiente pérdida de información asociada a los cuatro valores de propiedad. En la imagen de la derecha se muestra la rejilla dual de celdas en trazo rojo continuo y la celda de la rejilla uniforme en trazo rojo discontinuo. Además, se muestran en color negro los puntos que delimitan el segmento de arista en donde se realiza la interpolación. Como puede comprobarse, estos segmentos de arista tienen una orientación y longitud distintos a las aristas de la rejilla regular.

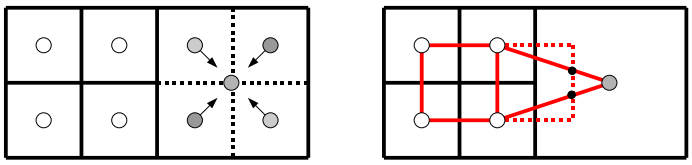

Figura 3.42: La imagen de la izquierda muestra el error cometido en el dominio de valores de propiedad como resultado de la agregación de *voxels* de menor tamaño. La imagen de la derecha muestra el error cometido debido a la distinta orientación y longitud de las aristas de las celdas duales.

Ambos métodos de extracción utilizan una interpolación lineal en las aristas de corte de las celdas para determinar el punto de la arista que pertenece a la isosuperficie. En la figura 3.43 se muestra un ejemplo que ilustra los elementos que se pueden usar para valorar el error cometido por la aproximación. La figura muestra dos funciones, $f(x)$ y $F(x)$, que representan respectivamente la función real y la función de interpolación que la estima. El método de extracción de isosuperficies permite encontrar el punto que resuelve la ecuación $F(x) = \nu$ en la arista de corte. Este punto es el x_0 en la figura. Además, al disponer de la función real que se está aproximando, se puede calcular el valor de la función $f(x)$ en dicho punto evaluándola, siendo el resultado el valor ν_0 de la figura. No obstante, el punto que verdaderamente se corresponde con la ecuación $f(x) = \nu$ en el intervalo en cuestión es x_R.

En la figura 3.43 se pueden observar tanto los puntos de corte x_0 y x_R como los valores de propiedad ν y ν_0. Estas parejas de valores, situadas en los dominios E y $\mathcal{V}$ respectivamente, permiten establecer dos métricas de error diferentes. Si nos basamos en la diferencia en valor absoluto entre los valores ν y ν_0, se puede obtener una medida indirecta de la bondad de la aproximación mediante interpolación lineal. Una diferencia mayor indica que la función $F(x)$ proporciona una peor aproximación a $f(x)$ en el punto x_0 correspondiente al isovalor ν. Conforme la diferencia es menor, la función $F(x)$ proporciona una mejor aproximación de $f(x)$ para el isovalor escogido.

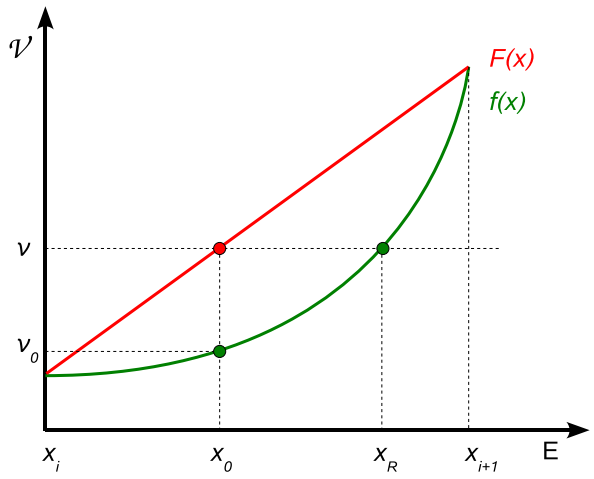

Figura 3.43: El punto x_0 estima mediante interpolación lineal al punto x_R de la isosuperficie en el intervalo x_i–x_{i+1}.

Para obtener una medida directa de la bondad de la aproximación, se va a utilizar la distancia en el espacio E entre el punto x_R que realmente está situado sobre $f(x) = \nu$ y el punto x_0 estimado mediante $F(x)$. De esta forma se puede determinar el error cometido en los puntos de la aproximación. Sin embargo, este enfoque presenta un problema. La función $f(x)$ es una función fácil de evaluar en un punto cualquiera pero es difícil determinar de forma analítica la solución para $f(x) = \nu$ salvo para casos simples. La resolución de la anterior ecuación proporciona como resultado el punto x_R. De hecho, ese es uno de los motivos por los cuales se utiliza la discretización del espacio y la interpolación entre aristas. Por tanto, hay que buscar un método aproximado para el cálculo de x_R.

En la siguiente sección se muestra la métrica escogida para medir el error de forma directa, es decir, utilizando la distancia entre x_0 y x_R en el espacio E, junto con la manera de aproximar el valor de x_R.

3.7.2. Métrica de error

En la figura 3.44 se muestran los elementos necesarios para medir el error que se produce al calcular el punto de la isosuperficie mediante interpolación lineal con respecto a calcularlo utilizando directamente la función $f(x)$. El problema estriba en la dificultad para resolver la ecuación $f(x) = \nu$ que nos proporcionaría el punto x_R para poder obtener el error $e_\nu = x_R - x_0$. Para solucionarlo, nos basamos en la expresión 3.14. Esta fórmula permite estimar el valor de una función $f(x)$ en un punto situado en un intervalo, Δ_x, suficientemente próximo a un punto x_0, cuyo valor es conocido, siempre que sea posible calcular la derivada de la función, $f'(x)$, en dicho punto.

$$f(x) \approx f(x_0) + f'(x_0)\ (x - x_0) \tag{3.14}$$

Despejando en la ecuación el término que nos interesa y sustituyendo por los valores conocidos obtenemos la siguiente ecuación que estima el error cometido al aproximar x_R por x_0:

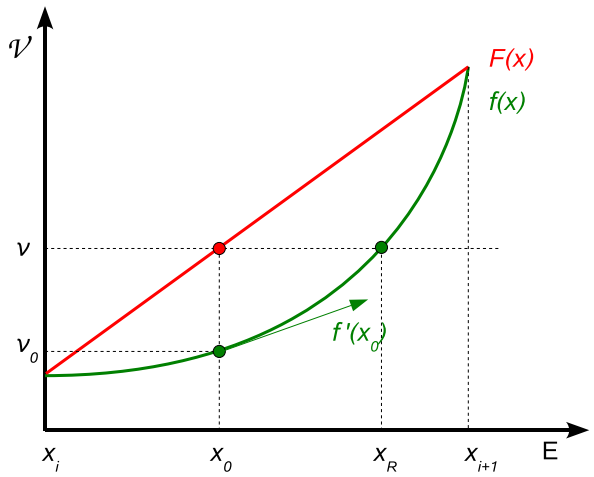

Figura 3.44: La derivada de la función $f(x)$ en el punto x_0, $f'(x_0)$, permite estimar el valor del punto x_R.

$$e_\nu = (x_R - x_0) \approx \frac{1}{f'(x_0)}(\nu - \nu_0) \tag{3.15}$$

En el caso tridimensional se va a utilizar el gradiente de la función, ∇f, en el punto X_0. La dirección de ∇f es la orientación en la cuál la derivada direccional tiene el mayor valor y $|\nabla f|$ es el valor de la derivada direccional. Por tanto, la métrica para estimar el error cometido tendrá la siguiente forma:

$$E_\nu(\boldsymbol{X}_0) = (\boldsymbol{X}_R - \boldsymbol{X}_0) \approx \frac{1}{|\nabla f(\boldsymbol{X}_0)|}(\nu - \nu_0) \tag{3.16}$$

donde $\boldsymbol{X}_0$ representa el punto solución de la ecuación $F(x, y, z) = \nu$ y $\nu_0 = f(x_0, y_0, z_0)$ siendo $\boldsymbol{X}_0 = (x_0, y_0, z_0)$. Obviamente estamos suponiendo que existe una única solución en las aristas de corte de las celdas.

3.7.3. Procedimiento para aplicar la métrica

Con la métrica establecida podemos obtener una aproximación al error cometido en el cálculo de un punto de la isosuperficie mediante interpolación lineal en cada arista de corte de una celda en ambas representaciones: rejilla regular y uniforme y *IDDG-Octree*. Como se ha comentado con anterioridad, debido a la diferencia entre las celdas de ambas representaciones no se van a obtener los mismos puntos de corte para poder realizar una comparación punto a punto. Por tanto, el procedimiento para aplicar la métrica de error se basa en calcular el error para todos los puntos de isosuperficie calculados en cada uno de los métodos.

De forma análoga al caso de la evaluación de la representación, se ha establecido un banco de pruebas basado en modelos sintéticos y conjuntos de datos volumétricos reales. En el caso de los modelos sintéticos la función $f(x, y, z)$ es conocida, mientras que en el caso de los conjuntos de datos de volumen usaremos como función $f(x, y, z)$ la función de interpolación trilineal definida por partes sobre la rejilla de máxima resolución . En este último caso, la rejilla regular que se compara con el *IDDG-Octree*, la cuál es además usada para su construcción, proviene de realizar un submuestreo sobre el conjunto de

datos de volumen. El submuestreo genera una rejilla regular con la mitad de muestras que la original, tomando el doble de intervalo de muestreo, es decir, se utiliza una de cada dos muestras consecutivas.

El procedimiento parte de un objeto base de referencia para establecer la comparación, ya sea un modelo sintético o un conjunto de datos volumétrico a máxima resolución, del cual se conoce su función de distribución de valores de propiedad $f(x, y, z)$ y su gradiente $\nabla(f(x, y, z))$. En el caso de los modelos sintéticos, el objeto se discretiza a una resolución determinada, obteniéndose así una rejilla regular y uniforme. En el caso de los datos volumétricos, se submuestrean estos datos como se ha explicado previamente para obtener una rejilla regular y uniforme con la mitad de muestras.

A partir de la rejilla regular y uniforme se construye un *IDDG-Octree*. A continuación, se elije un umbral y se extrae la isosuperficie correspondiente en cada uno de los métodos. Para cada punto de corte se calcula el error asociado utilizando la métrica de error (3.16). La secuencia de pasos a realizar para cada punto de corte con una arista sería la siguiente:

1. Obtener el punto de corte, $\boldsymbol{X}_0$, con la arista mediante interpolación lineal.
2. Obtener el valor de la función f en dicho punto, $f(x_0, y_0, z_0) = \nu_0$.
3. Obtener el módulo del gradiente de f en dicho punto, $|\nabla f(x_0, y_0, z_0)|$.
4. Obtener el valor de la distancia con signo $E_\nu(\boldsymbol{X}_0) = (\boldsymbol{X}_R - \boldsymbol{X}_0)$ usando la ecuación (3.16).

Para obtener medidas que permitan comparar el error cometido por los dos métodos de aproximación de isosuperficie se va a considerar de forma absoluta la distancia de la aproximación al punto correcto. Además, se procede a discretizar mediante intervalos el dominio de la función de error con una precisión determinada. El error cometido en cada punto de corte permite incrementar el número de puntos asociado a un determinado intervalo. Garantizando que cada error asociado a un punto de corte pertenece a uno de dichos intervalos se obtiene una distribución de error que representa el número de puntos que se encuentran a una cierta distancia de la isosuperficie, concretamente a una distancia situada entre los extremos del intervalo asociado. Este tipo de distribución de error, obtenida para cada una de las dos representaciones, permite establecer comparaciones entre ambas, solventando el problema de que los puntos de corte no coincidan y que el número de puntos de corte obtenidos en cada una de ellas sea distinto. Para establecer las comparaciones se utilizan la distribuciones de frecuencias relativas de error para cada proceso de extracción llevado a cabo, así como las correspondientes curvas de distribución de frecuencias de error.

Con el objetivo de tener una representación visual del comportamiento del error se ha construido una función de transferencia de color lineal que establece un color asociado a cada punto de corte en función del error cometido en su aproximación. El dominio de la función de transferencia son las distancias proporcionadas por la ecuación (3.16) y su rango se encuentra en el dominio de valores del modelo de color RGB. La función de transferencia está definida en base a dos funciones de transferencia de color complementarias que actuan sobre los canales R (rojo) y B (azul) del modelo de color. Hemos denominado a estas funciones $R(x)$ y $B(x)$ respectivamamente. La figura 3.45 muestra la forma de estas funciones. La función de transferencia establece un valor fijo, K, para el canal G del modelo de color RGB. Formalmente, la función de transferencia de color, $C(x)$, se define de la siguiente forma:

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

donde

$$\begin{aligned} R(x) &= R_{MIN} + \frac{R_{MAX} - R_{MIN}}{MAX} \, x \\ B(x) &= B_{MAX} + \frac{B_{MIN} - B_{MAX}}{MAX} \, x \end{aligned}$$

siendo $[R_{MIN}, R_{MAX}]$ el intervalo de color usado en el canal R y $[B_{MIN}, B_{MAX}]$ el intervalo de color usado en el canal B.

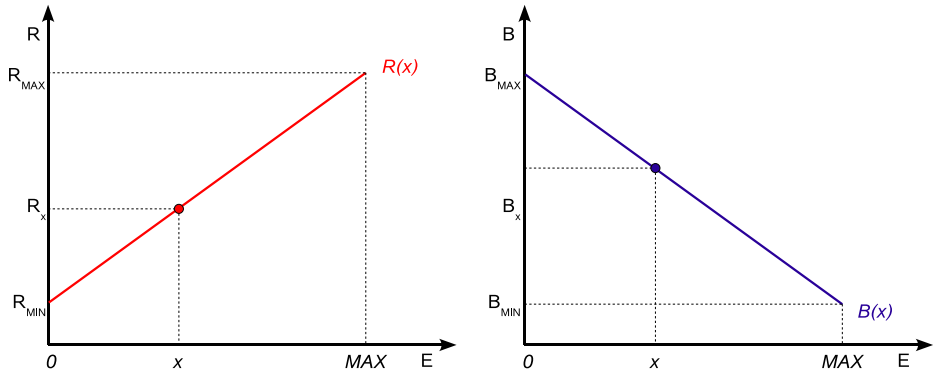

Figura 3.45: Funciones de transferencia de color $R(x)$ y $B(x)$, asociadas al canal rojo y azul respectivamente, definidas sobre el modelo de color RGB.

El gradiente de color que proporciona la función de transferencia va desde el color azul hasta el rojo. El valor azul coincide con el caso en el que el error cometido es nulo, es decir la distancia entre el punto de corte estimado y el punto real de la isosuperficie es 0. El valor rojo coincide con el mayor error cometido en cada modelo, tanto en el caso de que la isosuperficie estimada se haya desplazado hacia el interior de la isosuperficie real, $+MAX$, como en el caso opuesto, $-MAX$. La figura 3.46 muestra el gradiente de color y los valores de error que acotan el dominio de error.

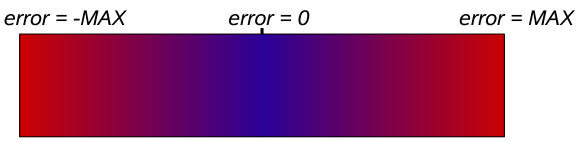

Figura 3.46: Gradiente de color que se corresponde con el error cometido en la aproximación de la isosuperficie proporcionada por el proceso de extracción. El color azul se corresponde con un error igual a 0 y el color rojo se corresponde con los errores máximos $-MAX$ y $+MAX$ cometidos en la aproximación.

3.7.4. Resultados

Para las pruebas del error cometido en la extracción de la isosuperficie por las representaciones *IDDG-Octree* y rejilla regular y uniforme se han utilizado tanto modelos sintéticos como conjuntos de datos volumétricos reales. El intervalo de error (distancias a la isosuperficie correcta) que se ha considerado es $[0, 3]$ y la precisión en la discretización de este intervalo ha sido de 1 milésima $(0, 001)$. Por tanto, los subintervalos en los que se divide el dominio de error considerado son 3000. Todos los puntos cuyo error cae por encima del valor 3 se acumulan aparte. Con respecto a la función de transferencia de color $C(x)$ se han elegido los siguientes valores para sus parámetros: $R_{MIN} = 0, 1$, $R_{MAX} = 0, 9$, $B_{MIN} = 0, 1$, $B_{MAX} = 0, 9$ y $K = 0, 1$.

En primer lugar se establece la comparativa entre representaciones haciendo uso de los modelos sintéticos. Cada modelo sintético se discretiza a una resolución de 128^3 utilizando el método de muestreo que proporciona valores en el dominio discreto $[0, 255]$, presentado en la sección 3.6. A partir de las discretizaciones se construye la representación de rejilla regular y uniforme y, a partir de ésta, se construye el *IDDG-Octree*. Seguidamente se extrae la isosuperficie utilizando un valor umbral de 60 y se acumula el valor obtenido en todos los puntos de corte, en su correspondiente intervalo del dominio de la función de error (distancias), para cada una de las representaciones. De esta forma se obtiene la distribución de frecuencias absolutas de error F.

Las tablas 3.39–3.42 muestran los resultados de error obtenidos para cada uno de los modelos sintéticos de prueba. De izquierda a derecha, la primera columna muestra el intervalo de error (Int.) en el cual se acumula el número de puntos que se situa a esa distancia con respecto al modelo de referencia. El intervalo se representa con un único valor numérico, por ejemplo $0, 001$, indicando que el número de puntos asociados se encuentra incluido en el intervalo $(0{,}000, 0{,}001]$. A continuación, se muestran los resultados de error obtenidos para el *IDDG-Octree*, utilizando tres criterios de similaridad distintos con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$ respectivamente, y para la rejilla regular y uniforme. Cada conjunto de resultados muestra tres columnas. La primera de ellas indica la frecuencia absoluta de error, F^i, la segunda indica la frecuencia relativa de error, f^i, y la tercera indica la frecuencia relativa acumulada de error, $f^i(x)$. Dependiendo del conjunto de resultados asociado, el índice i expresa:

- $i = o, j$. La medida F^i o f^i o $f^i(x)$ se refiere al *IDDG-Octree* (o) que ha sido construido utilizando como tolerancia de error $\epsilon = j$ para $j = 0, 1, 2$.
- $i = r$. La medida F^i o f^i o $f^i(x)$ se refiere a la rejilla regular y uniforme (r).

Las figuras 3.47–3.50 muestran visualmente el error cometido por el *IDDG-Octree* y por la rejilla de tres formas diferentes. En la parte superior, las tres primeras columnas muestran las imágenes obtenidas por nuestro método de extracción aplicado sobre el *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$ respectivamente. La cuarta columna muestra la imagen obtenida al aplicar Marching Cubes sobre la rejilla regular y uniforme. En la parte central, debajo de cada imagen, se muestra una gráfica que representa su histograma de frecuencias relativas. En éste se indican en el eje de abcisas los valores numéricos correspondientes al primer y último intervalo de error en el que hay algún punto incluido. Por último, en la parte inferior se muestra una gráfica que muestra las cuatro curvas de distribución de frecuencias relativas obtenidas para cada extracción. En este caso, los valores numéricos indicados en el eje de abcisas indican el intervalo en el que la curva de distribución alcanza el valor 1. Nótese que en algunas

gráficas no se visualizan las cuatro curvas debido a que son iguales y se solapan unas a otras. En beneficio de la comprensión de las gráficas, se ha hecho coincidir el color de la curva de distribución de frecuencias relativas de error de cada imagen con el color de su histograma asociado.

Por cuestiones de espacio en el documento algunas tablas no presentan todo el conjunto de resultados obtenido. Sin embargo, tanto las imágenes como las medidas estadísticas de error que se presentan en las figuras 3.47–3.50 se han realizado teniendo en cuenta todos los datos del dominio de error considerado.

En las imágenes situadas en la parte superior de las figuras 3.47–3.50 se puede comprobar visualmente la diferencia entre las distintas insosuperficies extraidas a partir del *IDDG-Octree* y la rejilla uniforme y regular. Las imagenes que son resultado de la extracción a partir del *IDDG-Octree* simplificado con criterios de similaridad cuya tolerancia de error es $\epsilon = 0$ y $\epsilon = 1$ no se distinguen de la imagen generada a partir de la rejilla dual. Esto permite inferir que la simplificación no ha llegado a reducir información importante para la forma final. Sin embargo, las imágenes generadas a partir del *IDDG-Octree* simplificado usando una tolerancia de error $\epsilon = 2$ si difieren de las generadas a partir de la rejilla (salvo en el caso del modelo **silla**). Este efecto se puede observar sobre todo en el cuello del modelo **jarrón** (Ver figura 3.50).

El efecto en la imagen extraida, el cual está producido por la variación de la tolerancia de error en el criterio de similaridad usado para la simplificación, puede comprenderse más fácilmente si se comparan las gráficas que representan los histogramas de frecuencias relativas de error de cada imagen (situadas debajo de cada una de ellas). En el caso de la **esfera** (Ver figura 3.47), el histograma asociado a la imagen generada para un *IDDG-Octree* simplificado con tolerancia de error $\epsilon = 2$ muestra que hay puntos de la estimación de la isosuperficie que caen en intervalos de error distintos al $0,765$. En este intervalo es en el que caen la mayoría de los puntos de la estimación que proporciona nuestro método. Nótese que en el caso del resto de imágenes, los histogramas coinciden. Este hecho puede comprobarse en la gráfica inferior en la que se muestran las curvas de distribución de error para cada histograma de frecuencias relativas. La curva en color naranja muestra la variación de errores de la imagen que se corresponde con la tolerancia de error $\epsilon = 2$ con respecto a las curvas del resto de imágenes. Nótese que al ser iguales solo se distingue la curva (en color azul) asociada a la imagen generada a partir de la rejilla uniforme y regular. En los modelos **tronco** (figura 3.48) y **silla** (figura 3.49) pueden comprobarse de manera análoga estas observaciones. Sin embargo, en el caso del modelo **jarrón** (figura 3.50) los histogramas parecen tener todos la misma forma. No obstante, al fijarse en el intervalo $(0, 639)$, en el cual comienzan a acumularse puntos de error para el caso de la tolerancia de error $\epsilon = 2$, se puede ver que es distinto al de las otras imágenes $(1, 498)$.

De forma análoga a los modelos sintéticos se establece la comparativa entre representaciones utilizando conjuntos de datos volumétricos reales. En este caso, cada *dataset* tiene su propia resolución y valores de propiedad asociados pero, aparte de este hecho, la metodología y descripciones de tablas y figuras es idéntica a la realizada para el caso de los modelos sintéticos.

Las tablas 3.43–3.46 muestran los resultados de error obtenidos para cada uno de los *datasets* de prueba. De izquierda a derecha, la primera columna muestra el intervalo de error (Int.) en el cual se acumula el número de puntos que se situa a esa distancia con respecto al modelo de referencia. A continuación, se muestran los resultados de error obtenidos para el *IDDG-Octree*, utilizando tres criterios de similaridad distintos con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$ respectivamente, y para la rejilla regular

y uniforme. Cada conjunto de resultados muestra tres columnas. La primera de ellas indica la frecuencia absoluta de error, F^i, la segunda indica la frecuencia relativa de error, f^i, y la tercera indica la frecuencia relativa acumulada de error, $f^i(x)$. El índice i representa lo mismo que en el caso de los modelos sintéticos.

De la misma forma que con los modelos sintéticos de prueba, las figuras 3.51–3.54 muestran visualmente el error cometido por el *IDDG-Octree* y por la rejilla de tres formas diferentes. En la parte superior, las tres primeras columnas muestran las imágenes obtenidas por nuestro método de extracción aplicado sobre el *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$ respectivamente. La cuarta columna muestra la imagen obtenida al aplicar Marching Cubes sobre la rejilla regular y uniforme. En la parte central, debajo de cada imagen, se muestra una gráfica que representa su histograma de frecuencias relativas y en la parte inferior se muestra una gráfica que muestra las cuatro curvas de distribución de frecuencias relativas obtenidas para cada extracción.

En las imágenes situadas en la parte superior de las figuras 3.51–3.54 se puede comprobar visualmente la diferencia entre las distintas insosuperficies extraidas a partir del *IDDG-Octree* y la rejilla uniforme y regular. En la figura 3.51, que se corresponde con el *dataset* aneurism, se puede observar que las zonas de error (en color rojo) no se distinguen demasiado entre imágenes. El *dataset* skull 3.54 produce un resultado similar aunque el número de zonas de error es mayor. Sin embargo, en el caso de los *datasets* bonsai y lobster se puede apreciar con más claridad la diferencia en las zonas de error. En el *dataset* bonsai 3.52, la imagen generada a partir de la rejilla uniforme no presenta casi ningúna zona de error mientras que las imágenes generadas a partir del *IDDG-Octree* presentan zonas marcadas de error, aunque estas zonas de error se encuentran distribuidas de forma casi idéntica entre las imágenes. Esto mismo ocurre en el caso del *dataset* lobster 3.53.

La diferencia entre las zonas de error que presentan las imágenes producidas a partir del *IDDG-Octree* y la rejilla regular y uniforme puede validarse mediante las gráficas que representan los histogramas de frecuencias relativas de error de cada imagen y en la gráfica en la que se muestran las curvas de distribución de error para cada histograma de frecuencias relativas.

Int.	*IDDG-Octree*									Rejilla		
	$\epsilon = 0$			$\epsilon = 1$			$\epsilon = 2$					
	$F^{o,0}$	$f^{o,0}$	$f^{o,0}(x)$	$F^{o,1}$	$f^{o,1}$	$f^{o,1}(x)$	$F^{o,2}$	$f^{o,2}$	$f^{o,2}(x)$	F^r	f^r	$f^r(x)$
0,001	0	0	0,00	0	0	0,00	0	0	0,00	0	0	0,00
0,002	0	0	0,00	0	0	0,00	0	0	0,00	0	0	0,00
...												
0,753	0	0	0,00	0	0	0,00	348	0	0,01	0	0	0,00
0,754	0	0	0,00	0	0	0,00	356	0	0,02	0	0	0,00
0,755	0	0	0,00	0	0	0,00	396	0	0,02	0	0	0,00
0,756	0	0	0,00	0	0	0,00	258	0	0,02	0	0	0,00
0,757	0	0	0,00	0	0	0,00	392	0	0,02	0	0	0,00
0,758	0	0	0,00	0	0	0,00	434	0	0,03	0	0	0,00
0,759	0	0	0,00	0	0	0,00	510	0	0,03	0	0	0,00
0,760	0	0	0,00	0	0	0,00	440	0	0,03	0	0	0,00
0,761	0	0	0,00	0	0	0,00	528	0	0,03	0	0	0,00
0,762	0	0	0,00	0	0	0,00	564	0	0,04	0	0	0,00
0,763	0	0	0,00	0	0	0,00	760	0,01	0,04	0	0	0,00
0,764	0	0	0,00	0	0	0,00	926	0,01	0,05	0	0	0,00
0,765	177600	1	1,00	177600	1	1,00	132076	0,88	0,93	177600	1	1,00
0,766	0	0	1,00	0	0	1,00	912	0,01	0,94	0	0	1,00
0,767	0	0	1,00	0	0	1,00	1466	0,01	0,95	0	0	1,00
0,768	0	0	1,00	0	0	1,00	1346	0,01	0,96	0	0	1,00
0,769	0	0	1,00	0	0	1,00	880	0,01	0,96	0	0	1,00
0,770	0	0	1,00	0	0	1,00	948	0,01	0,97	0	0	1,00
0,771	0	0	1,00	0	0	1,00	902	0,01	0,98	0	0	1,00
0,772	0	0	1,00	0	0	1,00	584	0	0,98	0	0	1,00
0,773	0	0	1,00	0	0	1,00	396	0	0,98	0	0	1,00
0,774	0	0	1,00	0	0	1,00	570	0	0,99	0	0	1,00
0,775	0	0	1,00	0	0	1,00	420	0	0,99	0	0	1,00
0,776	0	0	1,00	0	0	1,00	342	0	0,99	0	0	1,00
...												
2,999	0	0	1,00	0	0	1,00	0	0	1,00	0	0	1,00
3,000	0	0	1,00	0	0	1,00	0	0	1,00	0	0	1,00
>3	0	0	1,00	0	0	1,00	0	0	1,00	0	0	1,00

Tabla 3.39: Medidas de error para el modelo **esfera** con una resolución de 128^3 muestras usando el dominio $0 - 255$. De izquierda a derecha, la primera columna muestra los intervalos de error y, a continuación, en conjuntos de tres, se muestra la frecuencia absoluta de error, F^i, la frecuencia relativa de error, f^i, y la frecuencia relativa acumulada de error, $f^i(x)$ para el caso del *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$, y para el caso de la rejilla regular y uniforme.

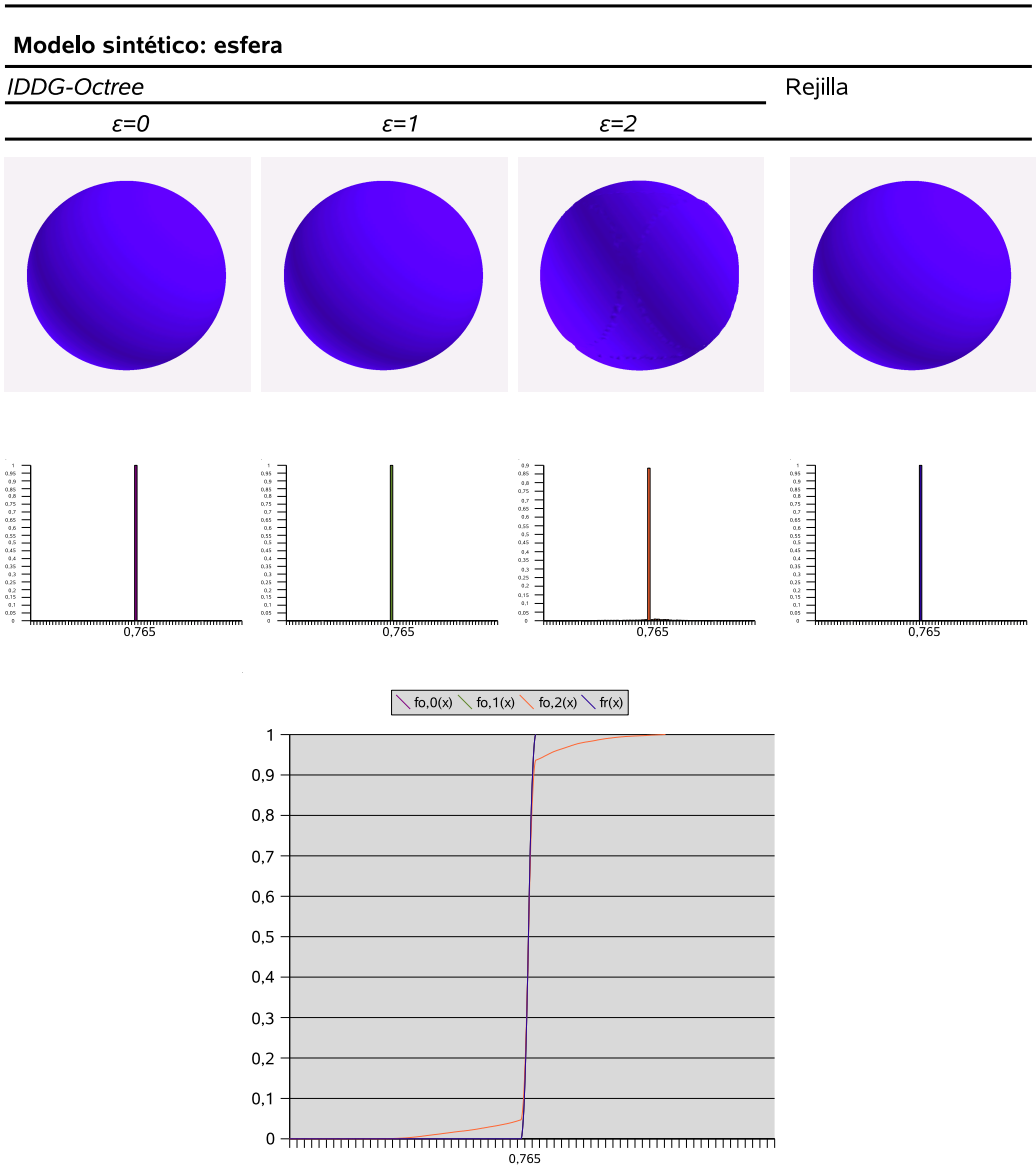

Figura 3.47: Imagenes y gráficas de medidas de error para el modelo sintético **esfera**. En la parte superior, las tres primeras columnas muestran las imágenes que han sido obtenidas usando nuestro método de extracción sobre el *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$ respectivamente. La cuarta columna muestra la imagen obtenida al aplicar Marching Cubes sobre la rejilla regular y uniforme. Todas las imágenes están coloreadas según el gradiente de error escogido. En la parte central, debajo de cada imagen, se muestra una gráfica que representa su histograma de frecuencias relativas. En éste se indican en el eje de abcisas los valores numéricos correspondientes al primer y último intervalo de error en el que hay algún punto incluido. La gráfica de la parte inferior muestra las curvas de distribución de frecuencias relativas obtenidas para cada extracción. Los valores numéricos indicados en el eje de abcisas indican el intervalo en el que la curva de distribución alcanza el valor 1.

Int.	*IDDG-Octree*									Rejilla		
	$\epsilon = 0$			$\epsilon = 1$			$\epsilon = 2$					
	$F^{o,0}$	$f^{o,0}$	$f^{o,0}(x)$	$F^{o,1}$	$f^{o,1}$	$f^{o,1}(x)$	$F^{o,2}$	$f^{o,2}$	$f^{o,2}(x)$	F^r	f^r	$f^r(x)$
0,001	0	0	0,00	0	0	0,00	0	0	0,00	0	0	0,00
0,002	0	0	0,00	0	0	0,00	0	0	0,00	0	0	0,00
...												
0,755	160	0,0007	0,06	160	0,0007	0,06	304	0,0014	0,07	160	0,0007	0,07
0,756	32	0,0001	0,06	32	0,0001	0,06	100	0,0005	0,07	32	0,0001	0,07
0,757	192	0,0009	0,06	192	0,0009	0,06	288	0,0013	0,07	192	0,0008	0,08
0,758	0	0,0000	0,06	0	0,0000	0,06	104	0,0005	0,07	0	0,0000	0,08
0,759	64	0,0003	0,07	64	0,0003	0,07	192	0,0009	0,07	64	0,0003	0,08
0,760	0	0,0000	0,07	0	0,0000	0,07	108	0,0005	0,07	0	0,0000	0,08
0,761	0	0,0000	0,07	0	0,0000	0,07	96	0,0004	0,07	0	0,0000	0,08
0,762	0	0,0000	0,07	0	0,0000	0,07	60	0,0003	0,07	0	0,0000	0,08
0,763	0	0,0000	0,07	0	0,0000	0,07	176	0,0008	0,08	0	0,0000	0,08
0,764	0	0,0000	0,07	0	0,0000	0,07	100	0,0005	0,08	0	0,0000	0,08
0,765	3728	0,0166	0,08	3728	0,0166	0,08	1816	0,0085	0,08	3728	0,0161	0,09
0,766	3392	0,0151	0,10	3392	0,0151	0,10	1264	0,0059	0,09	3392	0,0146	0,11
0,767	2704	0,0121	0,11	2704	0,0121	0,11	1424	0,0066	0,10	2704	0,0117	0,12
0,768	2176	0,0097	0,12	2176	0,0097	0,12	1200	0,0056	0,10	2176	0,0094	0,13
0,769	2256	0,0101	0,13	2256	0,0101	0,13	1552	0,0072	0,11	2384	0,0103	0,14
0,770	1120	0,0050	0,13	1120	0,0050	0,13	788	0,0037	0,11	1184	0,0051	0,14
0,771	1984	0,0089	0,14	1984	0,0089	0,14	1354	0,0063	0,12	1984	0,0086	0,15
0,772	769	0,0034	0,15	769	0,0034	0,15	625	0,0029	0,12	768	0,0033	0,15
0,773	1952	0,0087	0,15	1952	0,0087	0,15	1366	0,0064	0,13	1888	0,0081	0,16
0,774	576	0,0026	0,16	576	0,0026	0,16	614	0,0029	0,13	672	0,0029	0,17
0,775	856	0,0038	0,16	856	0,0038	0,16	682	0,0032	0,13	856	0,0037	0,17
0,776	1640	0,0073	0,17	1640	0,0073	0,17	1366	0,0064	0,14	1640	0,0071	0,18
0,777	544	0,0024	0,17	544	0,0024	0,17	540	0,0025	0,14	544	0,0023	0,18
0,778	1520	0,0068	0,18	1520	0,0068	0,18	1260	0,0059	0,15	1520	0,0066	0,19
0,779	576	0,0026	0,18	576	0,0026	0,18	436	0,0020	0,15	576	0,0025	0,19
0,780	544	0,0024	0,18	544	0,0024	0,18	624	0,0029	0,15	512	0,0022	0,19
0,781	1728	0,0077	0,19	1728	0,0077	0,19	1504	0,0070	0,16	1696	0,0073	0,20
0,782	544	0,0024	0,19	544	0,0024	0,19	572	0,0027	0,16	544	0,0023	0,20
0,783	544	0,0024	0,20	544	0,0024	0,20	528	0,0025	0,17	544	0,0023	0,20
0,784	352	0,0016	0,20	352	0,0016	0,20	358	0,0017	0,17	352	0,0015	0,20
0,785	1552	0,0069	0,20	1552	0,0069	0,20	1380	0,0064	0,17	1616	0,0070	0,21
...												
2,999	0	0	1,00	0	0	1,00	0	0	1,00	0	0	1,00
3,000	0	0	1,00	0	0	1,00	0	0	1,00	0	0	1,00
>3	0	0	1,00	0	0	1,00	0	0	1,00	0	0	1,00

Tabla 3.40: Medidas de error para el modelo **tronco** con una resolución de 128^3 muestras usando el dominio $0 - 255$. De izquierda a derecha, la primera columna muestra los intervalos de error y, a continuación, en conjuntos de tres, se muestra la frecuencia absoluta de error, F^i, la frecuencia relativa de error, f^i, y la frecuencia relativa acumulada de error, $f^i(x)$ para el caso del *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$, y para el caso de la rejilla regular y uniforme.

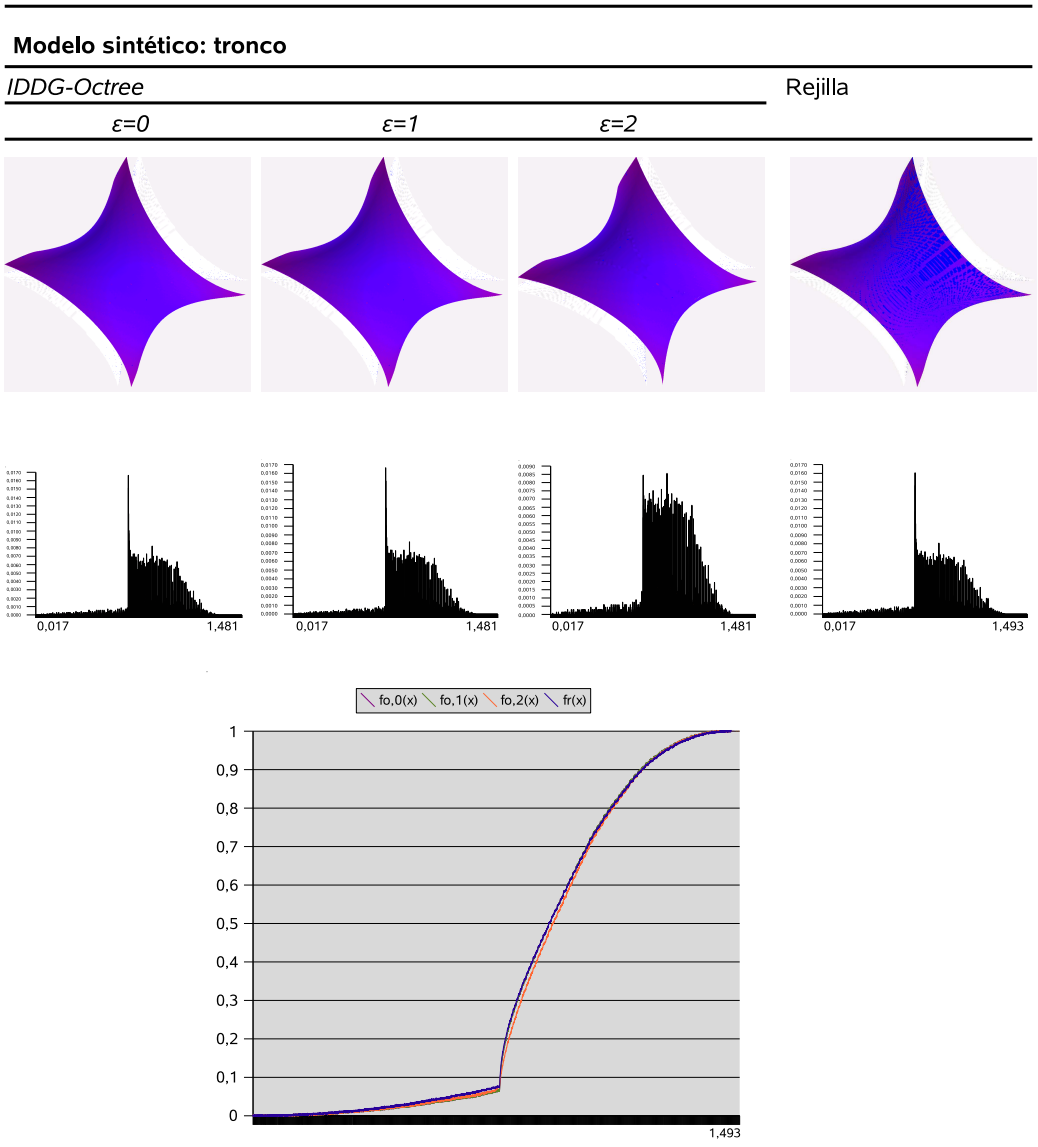

Figura 3.48: Imagenes y gráficas de medidas de error para el modelo sintético **tronco**. En la parte superior, las tres primeras columnas muestran las imágenes que han sido obtenidas usando nuestro método de extracción sobre el *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$ respectivamente. La cuarta columna muestra la imagen obtenida al aplicar Marching Cubes sobre la rejilla regular y uniforme. Todas las imágenes están coloreadas según el gradiente de error escogido. En la parte central, debajo de cada imagen, se muestra una gráfica que representa su histograma de frecuencias relativas. En éste se indican en el eje de abcisas los valores numéricos correspondientes al primer y último intervalo de error en el que hay algún punto incluido. La gráfica de la parte inferior muestra las curvas de distribución de frecuencias relativas obtenidas para cada extracción. Los valores numéricos indicados en el eje de abcisas indican el intervalo en el que la curva de distribución alcanza el valor 1.

Int.	*IDDG-Octree*									Rejilla		
	$\epsilon = 0$			$\epsilon = 1$			$\epsilon = 2$					
	$F^{o,0}$	$f^{o,0}$	$f^{o,0}(x)$	$F^{o,1}$	$f^{o,1}$	$f^{o,1}(x)$	$F^{o,2}$	$f^{o,2}$	$f^{o,2}(x)$	F^r	f^r	$f^r(x)$
0,001	0	0	0,00	0	0	0,00	0	0	0,00	0	0	0,00
0,002	0	0	0,00	0	0	0,00	0	0	0,00	0	0	0,00
...												
0,816	336	0,0027	0,12	336	0,0027	0,12	336	0,0027	0,12	336	0,0027	0,12
0,817	264	0,0021	0,12	264	0,0021	0,12	264	0,0021	0,12	264	0,0021	0,12
0,818	256	0,0021	0,12	256	0,0021	0,12	256	0,0021	0,12	256	0,0021	0,12
0,819	416	0,0034	0,13	416	0,0034	0,13	416	0,0034	0,13	416	0,0034	0,13
0,820	272	0,0022	0,13	272	0,0022	0,13	272	0,0022	0,13	272	0,0022	0,13
0,821	360	0,0029	0,13	360	0,0029	0,13	360	0,0029	0,13	360	0,0029	0,13
0,822	352	0,0028	0,14	352	0,0028	0,14	352	0,0028	0,14	352	0,0028	0,14
0,823	384	0,0031	0,14	384	0,0031	0,14	384	0,0031	0,14	384	0,0031	0,14
0,824	304	0,0025	0,14	304	0,0025	0,14	304	0,0025	0,14	304	0,0025	0,14
0,825	368	0,0030	0,14	368	0,0030	0,14	368	0,0030	0,14	368	0,0030	0,14
0,826	392	0,0032	0,15	392	0,0032	0,15	392	0,0032	0,15	392	0,0032	0,15
0,827	320	0,0026	0,15	320	0,0026	0,15	320	0,0026	0,15	320	0,0026	0,15
0,828	416	0,0034	0,15	416	0,0034	0,15	416	0,0034	0,15	416	0,0034	0,15
0,829	328	0,0026	0,16	328	0,0026	0,16	328	0,0026	0,16	328	0,0026	0,16
0,830	360	0,0029	0,16	360	0,0029	0,16	360	0,0029	0,16	360	0,0029	0,16
0,831	320	0,0026	0,16	320	0,0026	0,16	320	0,0026	0,16	320	0,0026	0,16
0,832	362	0,0029	0,16	362	0,0029	0,16	362	0,0029	0,16	362	0,0029	0,16
0,833	408	0,0033	0,17	408	0,0033	0,17	408	0,0033	0,17	408	0,0033	0,17
0,834	264	0,0021	0,17	264	0,0021	0,17	264	0,0021	0,17	264	0,0021	0,17
0,835	464	0,0037	0,17	464	0,0037	0,17	464	0,0037	0,17	464	0,0037	0,17
0,836	512	0,0041	0,18	512	0,0041	0,18	512	0,0041	0,18	512	0,0041	0,18
0,837	488	0,0039	0,18	488	0,0039	0,18	488	0,0039	0,18	488	0,0039	0,18
0,838	368	0,0030	0,18	368	0,0030	0,18	368	0,0030	0,18	368	0,0030	0,18
0,839	288	0,0023	0,19	288	0,0023	0,19	288	0,0023	0,19	288	0,0023	0,19
0,840	480	0,0039	0,19	480	0,0039	0,19	480	0,0039	0,19	480	0,0039	0,19
0,841	384	0,0031	0,19	384	0,0031	0,19	384	0,0031	0,19	384	0,0031	0,19
0,842	528	0,0043	0,20	528	0,0043	0,20	528	0,0043	0,20	528	0,0043	0,20
0,843	368	0,0030	0,20	368	0,0030	0,20	368	0,0030	0,20	368	0,0030	0,20
0,844	512	0,0041	0,20	512	0,0041	0,20	512	0,0041	0,20	512	0,0041	0,20
0,845	376	0,0030	0,21	376	0,0030	0,21	376	0,0030	0,21	376	0,0030	0,21
0,846	432	0,0035	0,21	432	0,0035	0,21	432	0,0035	0,21	432	0,0035	0,21
...												
2,999	0	0	1,00	0	0	1,00	0	0	1,00	0	0	1,00
3,000	0	0	1,00	0	0	1,00	0	0	1,00	0	0	1,00
>3	0	0	1,00	0	0	1,00	0	0	1,00	0	0	1,00

Tabla 3.41: Medidas de error para el modelo **silla** con una resolución de 128^3 muestras usando el dominio $0 - 255$. De izquierda a derecha, la primera columna muestra los intervalos de error y, a continuación, en conjuntos de tres, se muestra la frecuencia absoluta de error, F^i, la frecuencia relativa de error, f^i, y la frecuencia relativa acumulada de error, $f^i(x)$ para el caso del *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$, y para el caso de la rejilla regular y uniforme.

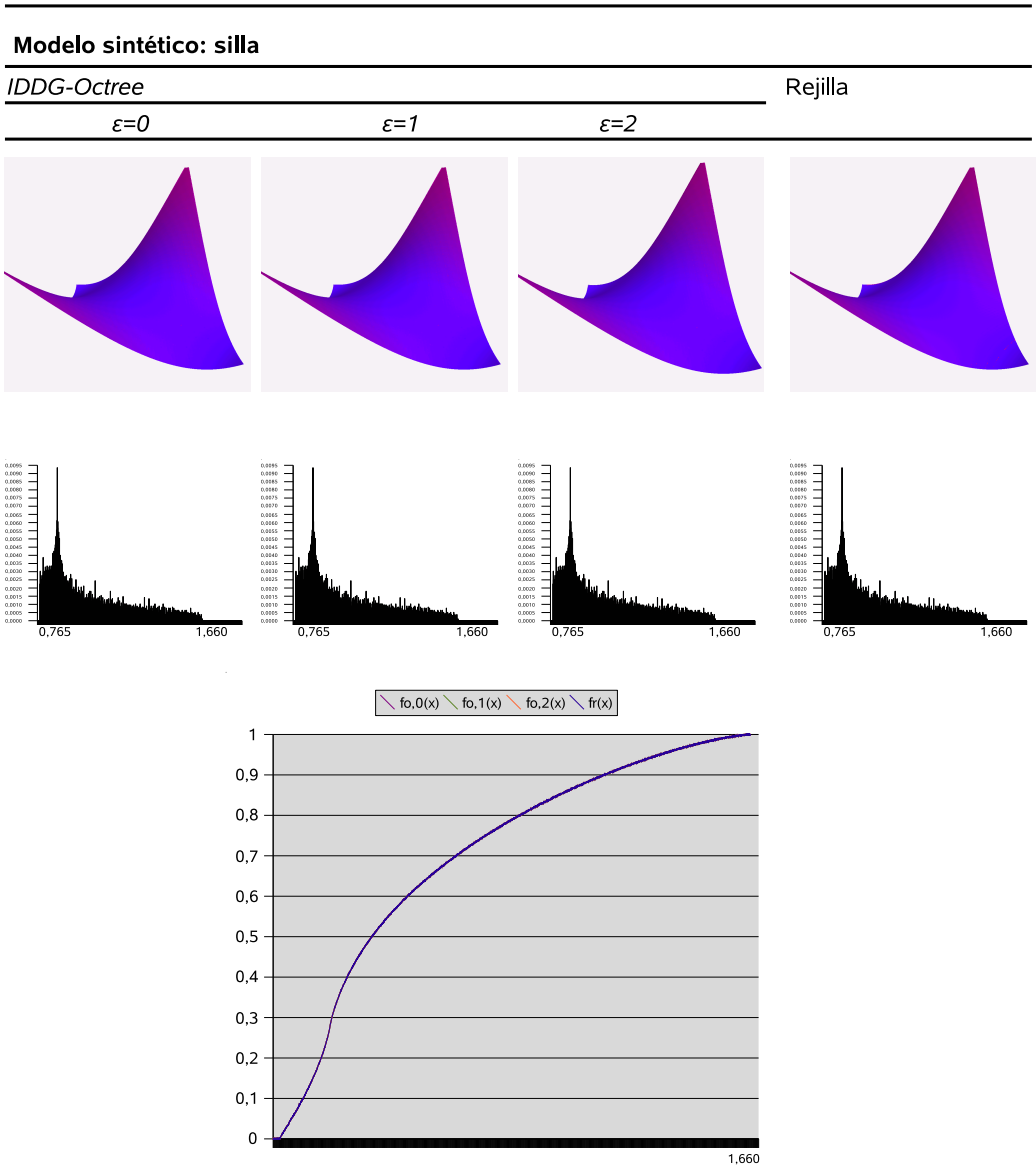

Figura 3.49: Imagenes y gráficas de medidas de error para el modelo sintético **silla**. En la parte superior, las tres primeras columnas muestran las imágenes que han sido obtenidas usando nuestro método de extracción sobre el *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$ respectivamente. La cuarta columna muestra la imagen obtenida al aplicar Marching Cubes sobre la rejilla regular y uniforme. Todas las imágenes están coloreadas según el gradiente de error escogido. En la parte central, debajo de cada imagen, se muestra una gráfica que representa su histograma de frecuencias relativas. En éste se indican en el eje de abcisas los valores numéricos correspondientes al primer y último intervalo de error en el que hay algún punto incluido. La gráfica de la parte inferior muestra las curvas de distribución de frecuencias relativas obtenidas para cada extracción. Los valores numéricos indicados en el eje de abcisas indican el intervalo en el que la curva de distribución alcanza el valor 1.

Int.	*IDDG-Octree*									Rejilla		
	$\epsilon = 0$			$\epsilon = 1$			$\epsilon = 2$					
	$F^{o,0}$	$f^{o,0}$	$f^{o,0}(x)$	$F^{o,1}$	$f^{o,1}$	$f^{o,1}(x)$	$F^{o,2}$	$f^{o,2}$	$f^{o,2}(x)$	F^r	f^r	$f^r(x)$
0,001	0	0	0,00	0	0	0,00	0	0	0,00	0	0	0,00
0,002	0	0	0,00	0	0	0,00	0	0	0,00	0	0	0,00
...												
0,735	176	0,0009	0,10	176	0,0009	0,10	144	0,0008	0,09	176	0,0009	0,10
0,736	192	0,0010	0,10	192	0,0010	0,10	168	0,0009	0,09	192	0,0010	0,10
0,737	128	0,0007	0,10	128	0,0007	0,10	36	0,0002	0,09	128	0,0007	0,10
0,738	80	0,0004	0,10	80	0,0004	0,10	80	0,0004	0,09	80	0,0004	0,10
0,739	32	0,0002	0,10	32	0,0002	0,10	44	0,0002	0,09	32	0,0002	0,10
0,740	176	0,0009	0,11	176	0,0009	0,11	112	0,0006	0,09	176	0,0009	0,11
0,741	80	0,0004	0,11	80	0,0004	0,11	56	0,0003	0,09	80	0,0004	0,11
0,742	64	0,0003	0,11	64	0,0003	0,11	88	0,0005	0,09	64	0,0003	0,11
0,743	192	0,0010	0,11	192	0,0010	0,11	184	0,0010	0,09	192	0,0010	0,11
0,744	64	0,0003	0,11	64	0,0003	0,11	92	0,0005	0,09	64	0,0003	0,11
0,745	96	0,0005	0,11	96	0,0005	0,11	116	0,0006	0,10	96	0,0005	0,11
0,746	32	0,0002	0,11	32	0,0002	0,11	40	0,0002	0,10	32	0,0002	0,11
0,747	128	0,0007	0,11	128	0,0007	0,11	140	0,0007	0,10	128	0,0007	0,11
0,748	128	0,0007	0,11	128	0,0007	0,11	128	0,0007	0,10	128	0,0007	0,11
0,749	128	0,0007	0,11	128	0,0007	0,11	116	0,0006	0,10	128	0,0007	0,11
0,750	192	0,0010	0,11	192	0,0010	0,11	80	0,0004	0,10	192	0,0010	0,11
0,751	32	0,0002	0,11	32	0,0002	0,11	40	0,0002	0,10	32	0,0002	0,11
0,752	128	0,0007	0,11	128	0,0007	0,11	128	0,0007	0,10	128	0,0007	0,11
0,753	32	0,0002	0,11	32	0,0002	0,11	32	0,0002	0,10	32	0,0002	0,11
0,754	0	0,0000	0,11	0	0,0000	0,11	4	0,0000	0,10	0	0,0000	0,11
0,755	96	0,0005	0,11	96	0,0005	0,11	104	0,0005	0,10	96	0,0005	0,11
0,756	96	0,0005	0,11	96	0,0005	0,11	112	0,0006	0,10	96	0,0005	0,11
0,757	64	0,0003	0,11	64	0,0003	0,11	64	0,0003	0,10	64	0,0003	0,11
0,758	160	0,0008	0,11	160	0,0008	0,11	160	0,0008	0,10	160	0,0008	0,11
0,759	96	0,0005	0,11	96	0,0005	0,11	96	0,0005	0,10	96	0,0005	0,11
0,760	96	0,0005	0,12	96	0,0005	0,12	104	0,0005	0,10	96	0,0005	0,12
0,761	32	0,0002	0,12	32	0,0002	0,12	32	0,0002	0,10	32	0,0002	0,12
0,762	96	0,0005	0,12	96	0,0005	0,12	96	0,0005	0,10	96	0,0005	0,12
0,763	128	0,0007	0,12	128	0,0007	0,12	108	0,0006	0,10	128	0,0007	0,12
0,764	224	0,0012	0,12	224	0,0012	0,12	96	0,0005	0,10	224	0,0012	0,12
0,765	64	0,0003	0,12	64	0,0003	0,12	72	0,0004	0,10	64	0,0003	0,12
...												
2,999	0	0	1,00	0	0	1,00	0	0	1,00	0	0	1,00
3,000	0	0	1,00	0	0	1,00	0	0	1,00	0	0	1,00
>3	0	0	1,00	0	0	1,00	0	0	1,00	0	0	1,00

Tabla 3.42: Medidas de error para el modelo **jarrón** con una resolución de 128^3 muestras usando el dominio $0 - 255$. De izquierda a derecha, la primera columna muestra los intervalos de error y, a continuación, en conjuntos de tres, se muestra la frecuencia absoluta de error, F^i, la frecuencia relativa de error, f^i, y la frecuencia relativa acumulada de error, $f^i(x)$ para el caso del *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$, y para el caso de la rejilla regular y uniforme.

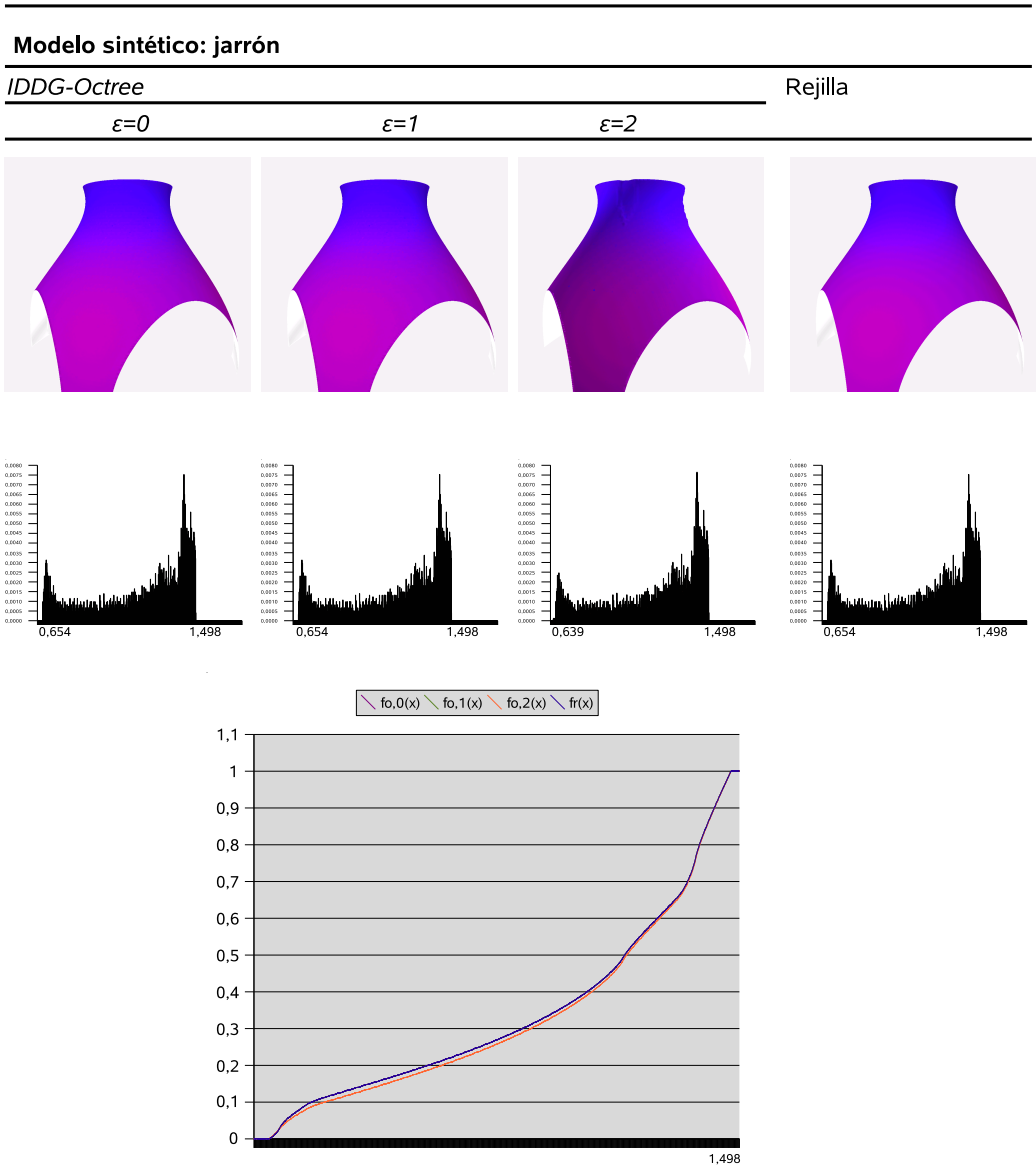

Figura 3.50: Imagenes y gráficas de medidas de error para el modelo sintético **jarrón**. En la parte superior, las tres primeras columnas muestran las imágenes que han sido obtenidas usando nuestro método de extracción sobre el *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$ respectivamente. La cuarta columna muestra la imagen obtenida al aplicar Marching Cubes sobre la rejilla regular y uniforme. Todas las imágenes están coloreadas según el gradiente de error escogido. En la parte central, debajo de cada imagen, se muestra una gráfica que representa su histograma de frecuencias relativas. En éste se indican en el eje de abcisas los valores numéricos correspondientes al primer y último intervalo de error en el que hay algún punto incluido. La gráfica de la parte inferior muestra las curvas de distribución de frecuencias relativas obtenidas para cada extracción. Los valores numéricos indicados en el eje de abcisas indican el intervalo en el que la curva de distribución alcanza el valor 1.

Int.	*IDDG-Octree*									Rejilla		
	$\epsilon = 0$			$\epsilon = 1$			$\epsilon = 2$					
	$F^{o,0}$	$f^{o,0}$	$f^{o,0}(x)$	$F^{o,1}$	$f^{o,1}$	$f^{o,1}(x)$	$F^{o,2}$	$f^{o,2}$	$f^{o,2}(x)$	F^{r}	f^{r}	$f^{r}(x)$
0,001	106659	0,96	0,96	106696	0,96	0,96	106751	0,96	0,96	103748	0,94	0,94
0,002	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
0,003	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
0,004	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
0,005	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
0,006	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
0,007	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
0,008	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
0,009	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
0,010	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
...												
2,990	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
2,991	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
2,992	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
2,993	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
2,994	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
2,995	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
2,996	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
2,997	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
2,998	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
2,999	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
3,000	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,00	0,94
>3	4415	0,04	1,00	4424	0,04	1,00	4427	0,04	1,00	6632	0,06	1,00

Tabla 3.43: Medidas de error para el *dataset* aneurism con una resolución de 256x256x256 muestras. De izquierda a derecha, la primera columna muestra los intervalos de error y, a continuación, en conjuntos de tres, se muestra la frecuencia absoluta de error, F^i, la frecuencia relativa de error, f^i, y la frecuencia relativa acumulada de error, $f^i(x)$ para el caso del *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$, y para el caso de la rejilla regular y uniforme.

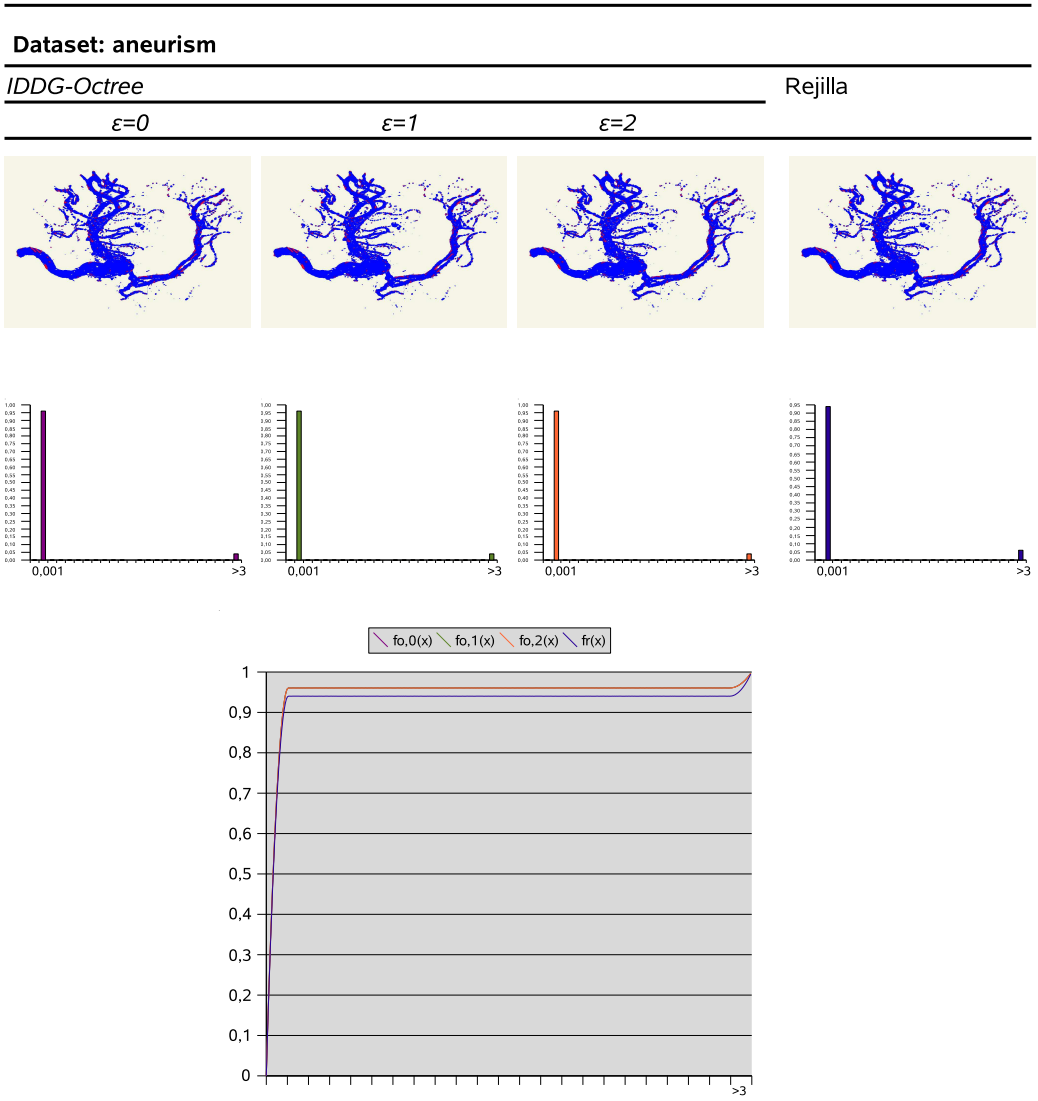

Figura 3.51: Imagenes y gráficas de medidas de error para el *dataset* aneurism. En la parte superior, las tres primeras columnas muestran las imágenes que han sido obtenidas usando nuestro método de extracción sobre el *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$ respectivamente. La cuarta columna muestra la imagen obtenida al aplicar Marching Cubes sobre la rejilla regular y uniforme. Todas las imágenes están coloreadas según el gradiente de error escogido. En la parte central, debajo de cada imagen, se muestra una gráfica que representa su histograma de frecuencias relativas. En éste se indican en el eje de abcisas los valores numéricos correspondientes al primer y último intervalo de error en el que hay algún punto incluido. La gráfica de la parte inferior muestra las curvas de distribución de frecuencias relativas obtenidas para cada extracción. Los valores numéricos indicados en el eje de abcisas indican el intervalo en el que la curva de distribución alcanza el valor 1.

Int.	*IDDG-Octree*									Rejilla		
	$\epsilon = 0$			$\epsilon = 1$			$\epsilon = 2$					
	$F^{o,0}$	$f^{o,0}$	$f^{o,0}(x)$	$F^{o,1}$	$f^{o,1}$	$f^{o,1}(x)$	$F^{o,2}$	$f^{o,2}$	$f^{o,2}(x)$	F^r	f^r	$f^r(x)$
0,001	392140	0,96	0,96	392186	0,96	0,96	392136	0,96	0,96	407310	0,9997	0,9997
0,002	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
0,003	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
0,004	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
0,005	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
0,006	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
0,007	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
0,008	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
0,009	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
0,010	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
...												
2,990	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
2,991	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
2,992	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
2,993	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
2,994	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
2,995	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
2,996	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
2,997	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
2,998	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
2,999	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
3,000	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9997
>3	15446	0,04	1,00	15420	0,04	1,00	15349	0,04	1,00	132	0,0003	1,0000

Tabla 3.44: Medidas de error para el *dataset* bonsai con una resolución de 256x256x256 muestras. De izquierda a derecha, la primera columna muestra los intervalos de error y, a continuación, en conjuntos de tres, se muestra la frecuencia absoluta de error, F^i, la frecuencia relativa de error, f^i, y la frecuencia relativa acumulada de error, $f^i(x)$ para el caso del *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$, y para el caso de la rejilla regular y uniforme.

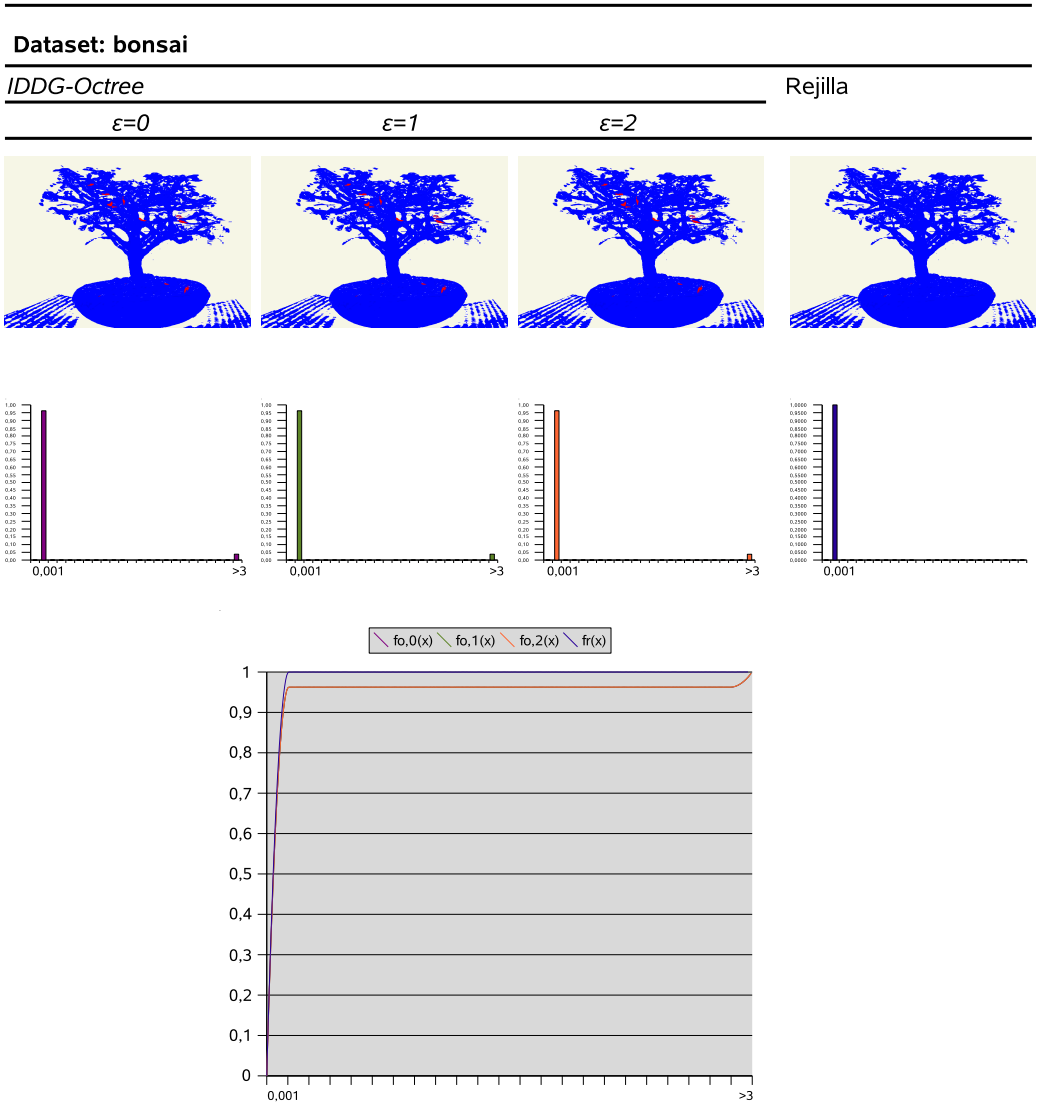

Figura 3.52: Imagenes y gráficas de medidas de error para el *dataset* bonsai. En la parte superior, las tres primeras columnas muestran las imágenes que han sido obtenidas usando nuestro método de extracción sobre el *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$ respectivamente. La cuarta columna muestra la imagen obtenida al aplicar Marching Cubes sobre la rejilla regular y uniforme. Todas las imágenes están coloreadas según el gradiente de error escogido. En la parte central, debajo de cada imagen, se muestra una gráfica que representa su histograma de frecuencias relativas. En éste se indican en el eje de abcisas los valores numéricos correspondientes al primer y último intervalo de error en el que hay algún punto incluido. La gráfica de la parte inferior muestra las curvas de distribución de frecuencias relativas obtenidas para cada extracción. Los valores numéricos indicados en el eje de abcisas indican el intervalo en el que la curva de distribución alcanza el valor 1.

Int.	*IDDG-Octree*									Rejilla		
	$\epsilon = 0$			$\epsilon = 1$			$\epsilon = 2$					
	$F^{o,0}$	$f^{o,0}$	$f^{o,0}(x)$	$F^{o,1}$	$f^{o,1}$	$f^{o,1}(x)$	$F^{o,2}$	$f^{o,2}$	$f^{o,2}(x)$	F^r	f^r	$f^r(x)$
0,001	143158	0,96	0,96	143162	0,96	0,96	143197	0,96	0,96	149824	0,9993	0,9993
0,002	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
0,003	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
0,004	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
0,005	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
0,006	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
0,007	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
0,008	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
0,009	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
0,010	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
...												
2,990	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
2,991	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
2,992	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
2,993	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
2,994	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
2,995	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
2,996	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
2,997	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
2,998	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
2,999	0	0,00	0,96	0	0,00	0,96	0	0,00	0,96	0	0,0000	0,9993
3,000	4	0,00	0,96	4	0,00	0,96	4	0,00	0,96	0	0,0000	0,9993
>3	6731	0,04	1,00	6727	0,04	1,00	6721	0,04	1,00	108	0,0007	1,0000

Tabla 3.45: Medidas de error para el *dataset* lobster con una resolución de 301x324x56 muestras. De izquierda a derecha, la primera columna muestra los intervalos de error y, a continuación, en conjuntos de tres, se muestra la frecuencia absoluta de error, F^i, la frecuencia relativa de error, f^i, y la frecuencia relativa acumulada de error, $f^i(x)$ para el caso del *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$, y para el caso de la rejilla regular y uniforme.

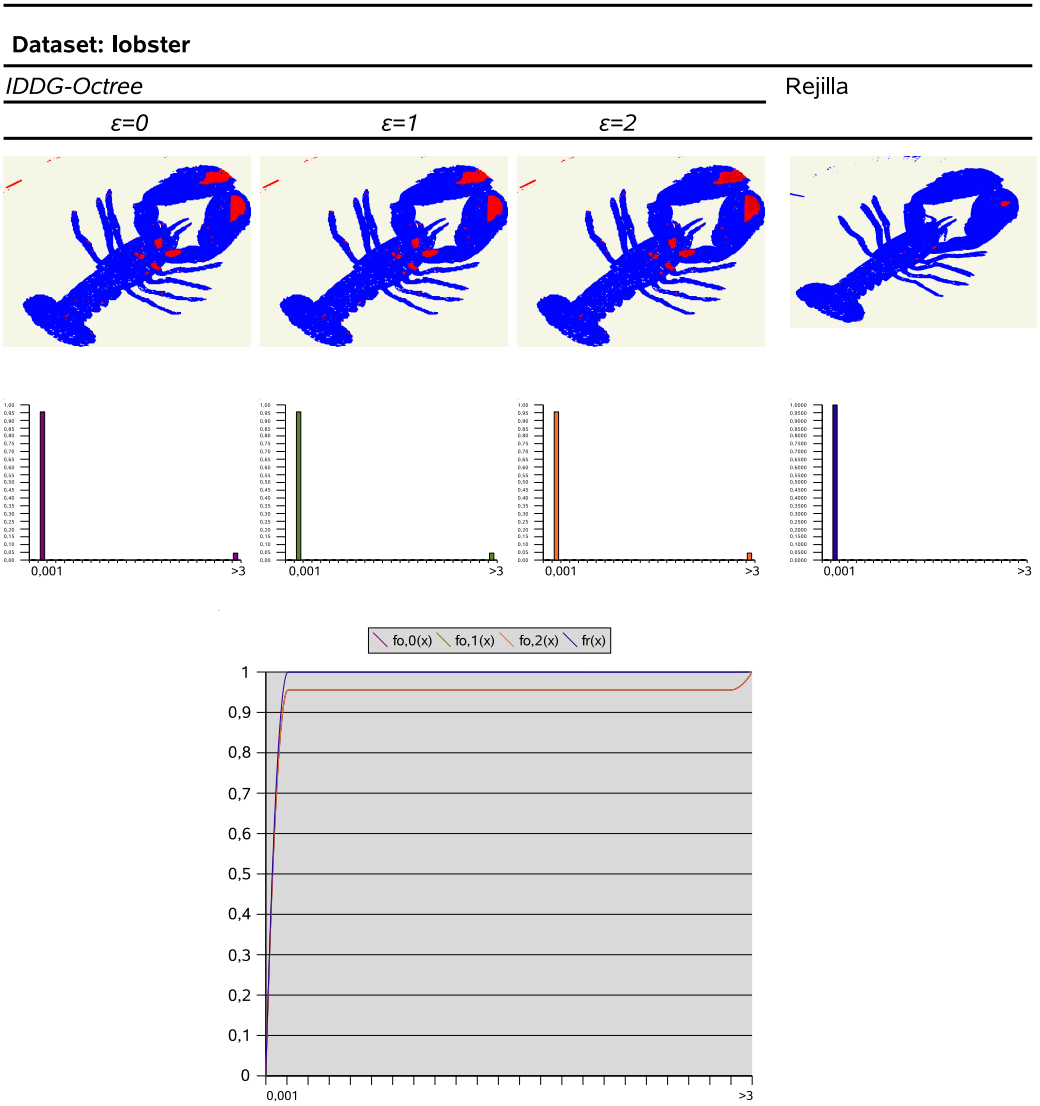

Figura 3.53: Imagenes y gráficas de medidas de error para el *dataset* lobster. En la parte superior, las tres primeras columnas muestran las imágenes que han sido obtenidas usando nuestro método de extracción sobre el *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$ respectivamente. La cuarta columna muestra la imagen obtenida al aplicar Marching Cubes sobre la rejilla regular y uniforme. Todas las imágenes están coloreadas según el gradiente de error escogido. En la parte central, debajo de cada imagen, se muestra una gráfica que representa su histograma de frecuencias relativas. En éste se indican en el eje de abcisas los valores numéricos correspondientes al primer y último intervalo de error en el que hay algún punto incluido. La gráfica de la parte inferior muestra las curvas de distribución de frecuencias relativas obtenidas para cada extracción. Los valores numéricos indicados en el eje de abcisas indican el intervalo en el que la curva de distribución alcanza el valor 1.

Int.	*IDDG-Octree*									Rejilla		
	$\epsilon = 0$			$\epsilon = 1$			$\epsilon = 2$					
	$F^{o,0}$	$f^{o,0}$	$f^{o,0}(x)$	$F^{o,1}$	$f^{o,1}$	$f^{o,1}(x)$	$F^{o,2}$	$f^{o,2}$	$f^{o,2}(x)$	F^{r}	f^{r}	$f^{r}(x)$
0,001	597530	0,92	0,92	597546	0,92	0,92	597538	0,92	0,92	597530	0,92	0,92
0,002	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
0,003	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
0,004	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
0,005	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
0,006	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
0,007	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
0,008	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
0,009	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
0,010	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
...												
2,990	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
2,991	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
2,992	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
2,993	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
2,994	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
2,995	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
2,996	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
2,997	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
2,998	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
2,999	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
3,000	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92	0	0,00	0,92
>3	54426	0,08	1,00	54422	0,08	1,00	54430	0,08	1,00	54426	0,08	1,00

Tabla 3.46: Medidas de error para el *dataset* skull con una resolución de 256x256x256 muestras. De izquierda a derecha, la primera columna muestra los intervalos de error y, a continuación, en conjuntos de tres, se muestra la frecuencia absoluta de error, F^i, la frecuencia relativa de error, f^i, y la frecuencia relativa acumulada de error, $f^i(x)$ para el caso del *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$, y para el caso de la rejilla regular y uniforme.

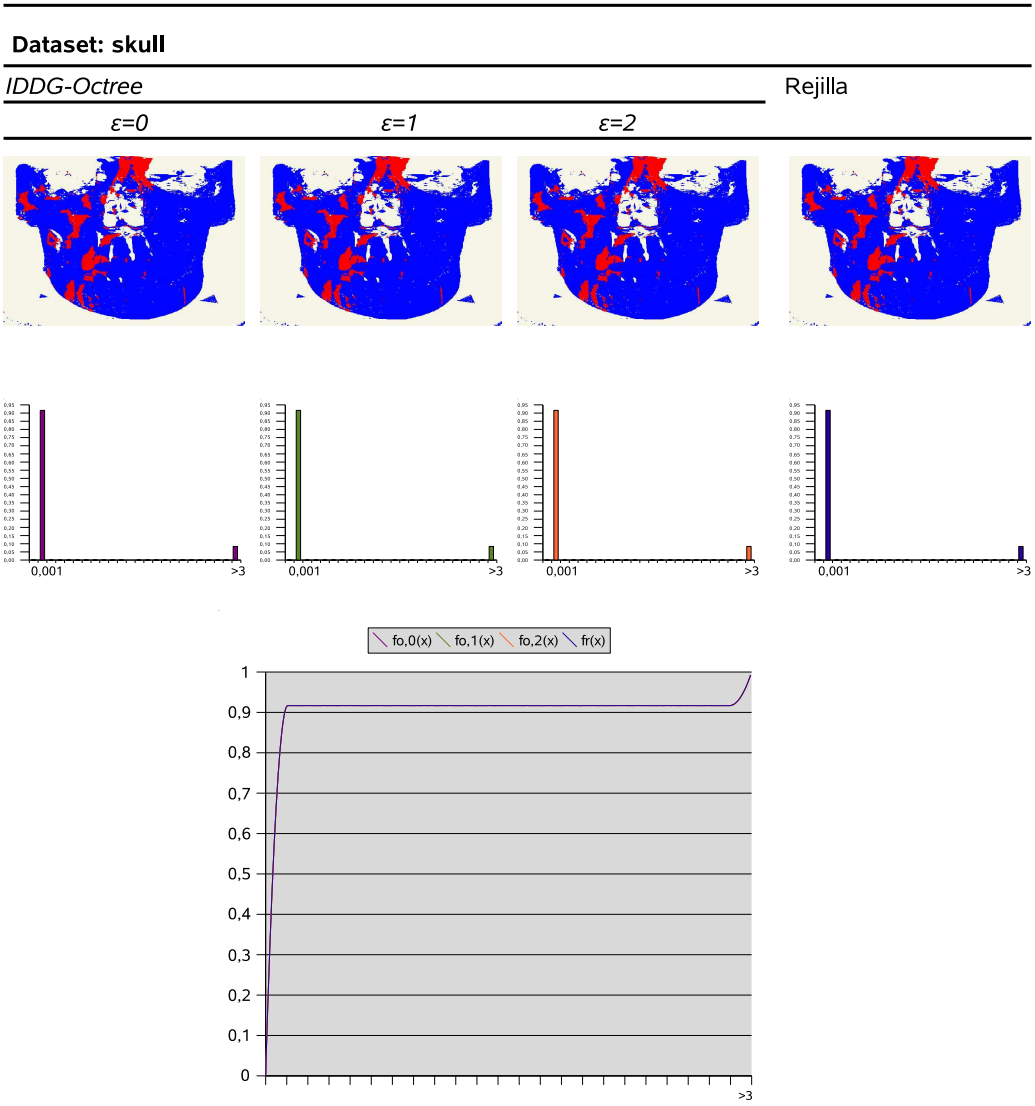

Figura 3.54: Imagenes y gráficas de medidas de error para el *dataset* skull. En la parte superior, las tres primeras columnas muestran las imágenes que han sido obtenidas usando nuestro método de extracción sobre el *IDDG-Octree* con tolerancias de error $\epsilon = 0$, $\epsilon = 1$ y $\epsilon = 2$ respectivamente. La cuarta columna muestra la imagen obtenida al aplicar Marching Cubes sobre la rejilla regular y uniforme. Todas las imágenes están coloreadas según el gradiente de error escogido. En la parte central, debajo de cada imagen, se muestra una gráfica que representa su histograma de frecuencias relativas. En éste se indican en el eje de abcisas los valores numéricos correspondientes al primer y último intervalo de error en el que hay algún punto incluido. La gráfica de la parte inferior muestra las curvas de distribución de frecuencias relativas obtenidas para cada extracción. Los valores numéricos indicados en el eje de abcisas indican el intervalo en el que la curva de distribución alcanza el valor 1.

3.8. Conclusiones

Se ha presentado una representación híbrida del enfoque de *voxels* y el de celdas que permite beneficiarse de la ventajas de ambos. Por una parte permite ahorrar espacio de almacenamiento gracias al *octree* de *voxels* y, por otra, gracias a la representación implícita de las celdas duales a dichos *voxels* permite realizar extracción de isosuperficies con resultados muy similares a los de las representaciones regulares y sin pagar el coste de mantener una estructura adicional.

Se ha formalizado el concepto de *voxel* minimal para el *IDDG-Octree*, y a partir de él se ha desarrollado un método que garantiza la generación automática de la topología asociada a la rejilla dual de celdas, la cual se almacena implícitamente en nuestra representación. Además, para garantizar que la partición de celdas cubre completamente el volumen representado en el *octree*, se ha descrito un método para envolver con celdas duales los vértices de los *voxels* que caen sobre la caja englobante del volumen.

Se ha presentado un método de extracción de isosuperficies que sigue la estrategia de marchar sobre las celdas duales y que tiene en cuenta la diferencia de tamaño de los *voxels* que son conectados por dichas celdas. Para ello se restringe el segmento de arista en donde se realiza la interpolación en las aristas con cambio de signo a la zona borde de los *voxels*. De esta forma se consigue una isosuperficie muy próxima a la que se puede obtener a partir de representaciones regulares aplicadas sobre el mismo conjunto de datos de entrada.

CAPÍTULO 4

Aplicación de *IDDG-Octree* a la escultura virtual

Las progresivas mejoras en las tecnologías de los ordenadores, especialmente las relativas al hardware especializado en gráficos, nos permiten manipular interactivamente objetos virtuales en un espacio 3D virtual. La construcción de objetos complejos de forma interactiva es un problema complejo, tanto desde el punto de vista del dominio de posibles objetos que debe soportar la representación como desde el punto de vista de la facilidad de creación por parte del usuario. La escultura virtual propone un enfoque de creación interactiva de objetos que abstrae al usuario de la complejidad de las representaciones, permitiéndole centrarse en la interacción sobre el objeto mediante herramientas de modelado que emulan las del escultor real.

Con el objetivo de probar la respuesta de nuestra representación frente a la manipulación interactiva se ha desarrollado una aplicación prototipo que permite realizar escultura virtual. La interacción con el objeto virtual se lleva a cabo mediante la utilización de un dispositivo háptico. En la primera sección se estudian los requisitos que es conveniente que cumplan tanto las representaciones de los objetos a modelar como las herramientas que permiten la operación sobre los objetos. En la siguiente sección se explica el proceso de actualización del *IDDG-Octree* que se produce como resultado de la aplicación de una herramienta. Para explicar el método de actualización propuesto se hace uso de la implementación de una herramienta que permite sustraer material de un objeto representado por un *IDDG-Octree*. Finalmente se presentan las conclusiones.

4.1. Introducción

La escultura virtual es una técnica de modelado de objetos por ordenador presentada por Galyean [GH91]. La técnica persigue el objetivo clave de que la representación computacional de los objetos permita al usuario abstraerse los detalles de dicha representación, permitiéndole concentrarse en el modelado de los objetos de la misma forma que un escultor puede hacerlo sobre los objetos del mundo real. Para conseguir este proceso de escultura simulado es necesario dotar al usuario de herramientas que permitan realizar operaciones de edición de forma interactiva.

Podemos clasificar los trabajos que permiten el uso de esta técnica basándonos en la representación utilizada para describir el objeto modelado, dividiendolos en: modelos de superficie y modelos de volumen.

El primer tipo utiliza una representación directa de la superficie sobre la que se aplican las herramientas de modelado [Nay90b, BL95, JX96, SPS01]. Este tipo de representación permite la aplicación de deformaciones interactivas pero requiere mecanismos complejos a la hora de permitir cambios en la topología del objeto modelado.

Los modelos volumétricos permiten la aplicación de esta técnica de una manera mucho más parecida a la metáfora de la escultura. La mayoría de las representaciones utilizan un campo escalar que puede estar bien almacenado en una rejilla [GH91, WK95, AS96] o bien controlado mediante primitivas [MOT98, MTY00, MQW01, HQ02]. La superficie del objeto es una isosuperficie del campo, cuyos valores representan la densidad del material a modelar. Esta representación es adecuada para realizar operaciones de volumen, tales como substraer o añadir material, modificando localmente los valores del campo. La ventaja que proporcionan estriba en que las representaciones volumétricas permiten manejar implícitamente cualquier cambio topológico sin la necesidad de establecer ningún mecanismo adicional.

Con respecto a las representaciones basadas en primitivas de generación del campo escalar, Matsumiya [MTY00] realizó escultura virtual utilizando como representación superficies implícitas a partir de primitivas geométricas (*skeletal implicit surfaces*), realizando deformaciones operando directamente sobre las funciones implícitas y extrayendo una isosuperficie para mostrar el objeto que está siendo modelado. Mizuno [MOT98] presentó una representación basada en CSG para realizar escultura virtual sobre objetos con apariencia de tallas en madera. McDonnell [MQW01] presentó una representación del campo basada en sólidos de subdivisión que eran modificados por las herramientas mediante técnicas basadas en modelado físico. Hua [HQ02] utilizó funciones implícitas que son generadas a partir de primitivas *B-spline* para definir el campo escalar.

Centrándonos en los modelos volumétricos que representan el campo escalar mediante una rejilla de valores de propiedad, estos a su vez pueden dividirse en función de si los valores de densidad de material son binarios [RBB90, BBB91, SM98] o pertenecen a un dominio de valores más amplio (representaciones escalares de volumen) [GH91, WK95, AS96, Bær98, RE00, FCG99, FCG00, PWFO01, BC02]. La ventaja de los segundos es que proporcionan superficies suaves, por lo que son los más comúnmente usados.

Las representaciones que utilizan una rejilla regular presentan un problema de sobremuestreo en regiones en donde no es necesaria mucha resolución. Las representaciones irregulares permiten solucionar este problema [FPRJ00, PF01, FCG02, JLSW02, BFCG04], adaptándose a las necesidades de resolución de cada zona del objeto. En todas ellas, la solución pasa por utilizar algún tipo de estructura jerárquica.

Frisken y otros [FPRJ00, PF01] proponen el uso de campos de distancias muestreados de forma adaptativa (ADFs del inglés *Adaptive Sampled Distance Fields*) con el objetivo de almacenar un ratio de muestreo alto en las regiones de los objetos donde exista gran nivel de detalle, almacenando un ratio bajo en las áreas donde el campo varía suavemente. Las celdas resultantes son almacenadas en un *octree* con el objetivo de procesarlas de forma eficiente. Ju y otros [JLSW02] proponen un *octree* cuyos nodos hoja contienen celdas etiquetadas con signos en sus vértices en función del umbral de isosuperficie escogido. Las aristas que presentan un cambio de signo almacenan los puntos de intersección y las normales en dichos puntos. En [FCG02, BFCG04] se utiliza una estructura jerárquica de rejillas que permite solucionar, tanto el problema de la

adaptación de las muestras requeridas, como el problema de la modificación local de la estructura cuando se aplica una operación de modelado al objeto.

4.1.1. Propiedades deseables y cuestiones de diseño

Uno de los requisitos fundamentales de la técnica es que la representación del objeto a modelar permita ocultar la complejidad de ésta al usuario. La interfase de trabajo con el usuario debe ser suficientemente intuitiva como para que éste trabaje únicamente con las herramientas virtuales, aplicándolas sobre la forma que se está modelando, evitando la presencia de puntos de control y otros artefactos adicionales tan comunes en las técnicas de modelado presentes en CAD.

Debido al proceso progresivo de aplicación de herramientas que es común en el modelado sobre materiales blandos, es necesario que las modificaciones efectuadas sobre el modelo proporcionen al usuario una respuesta en tiempo interactivo, más aún, sería deseable que la respuesta fuese en tiempo real. Este requisito proporciona una sensación más cercaca al proceso de modelado real, permitiendo ver inmediatamente los efectos que produce una determinada herramienta en el objeto que se está modelando.

Debido al que el proceso de escultura se puede desarrollar utilizando distintos tipos de herramientas, con distintas formas y tamaños, es necesario que el modelo permita la representación de la forma a distintos niveles de detalle. Por tanto, es deseable que la representación soporte resolución variable para que no sea necesario mantener regiones sobremuestreadas.

4.1.2. Herramientas

En esta sección se presentan las características generales y los tipos de herramientas más comúnmente implementadas en los sistemas de escultura virtual. La exposición se centra en las herramientas que operan sobre representaciones discretas de campos escalares.

Los sistemas para escultura virtual presentan una gran similitud a la hora de la aplicación de las herramientas. El usuario situa la herramienta en algún lugar del volumen de trabajo y la herramienta afecta una *región de influencia*. Para cada *voxel* con valor de propiedad v y posición $\boldsymbol{P}$ incluido en la región de influencia se lleva a cabo una operación que, en terminos generales, puede ser una de las dos siguientes:

- El valor v se reemplaza por una media ponderada de los valores de los *voxels* adyacentes.
- El valor v se reemplaza por una combinación de dicho valor y el valor que proporciona la evaluación de la herramienta en el punto $\boldsymbol{P}$.

Las herramientas pueden aplicarse directamente sobre la representación usando para ello operaciones booleanas. Una herramienta se define en base a un volumen que determina su forma y un campo que determina la forma en la que se verán afectados los *voxels* que constituyen la representación del objeto a modelar y que se encuentren incluidos en el volumen de la herramienta cuando ésta se aplica. Adicionalmente, las herramientas pueden permitir variar su orientación aparte de su posición.

Por tanto, a la hora de aplicar una herramienta a nuestra representación se han de tener en cuenta dos aspectos fundamentales. En primer lugar, es necesario asegurar que la herramienta se muestrea correctamente, es decir, la resolución de la representación

en el ámbito de influencia de la herramienta debe ser suficiente para permitir reflejar la forma y tamaño de ésta. En segundo lugar, para cada elemento de la representación que caiga bajo el alcance de la herramienta, es necesario combinar su valor de propiedad con el que aporta la herramienta, dependiendo de la operación asociada a ésta.

4.2. Escultura virtual mediante *IDDG-Octree*

En nuestro caso, el *IDDG-Octree* mantiene el campo escalar de valores de propiedad que representa la densidad de material del objeto a modelar. Las operaciones de modelado se realizan sobre la isosuperficie extraida a partir de la rejilla dual de celdas implícitamente representada en éste. Cuando se aplica una operación sobre el objeto, se localizan en el *octree* los *voxels* que van a ver modificados sus valores de propiedad, y en el caso de que sea necesario, se actualiza la topología de la rejilla dual afectada por la modificación. La topología de la rejilla dual cambia como consecuencia de que los nodos hoja del *octree*, que representan regiones del espacio, son agregados o subdivididos.

Tras la actualización de los valores de propiedad de los *voxels* implicados en la operación, y el posible cambio en la topología de las celdas duales asociadas a sus vértices, se extrae la isosuperficie que atraviesa las celdas para que el usuario pueda percibir el cambio que ha provocado la aplicación de la herramienta sobre el objeto. Nuestra representación permite que la determinación del alcance de la operación, la modificación de la porción de la rejilla dual correspondiente, en su caso, y la extracción de la isosuperficie asociada se realice en tiempo interactivo. De esta forma, nuestra representación permanece oculta al usuario (éste solo percibe la isosuperficie como modelo virtual sobre el que trabajar) y permite la respuesta interactiva, ambos requisitos fundamentales de la técnica de escultura virtual.

A continuación se mostrará el método que hemos desarrollado para conseguir la modificación local de nuestra estructura de datos y las herramientas clásicas de escultura virtual que hemos implementado para probarlo. Seguidamente mostraremos el uso que hemos hecho del mecanismo *Vertex Buffer Object (VBO)* (incluidos en las extensiones de OpenGL) para descartar solamente los triángulos que se ven implicados en la aplicación de las operaciones.

4.2.1. Operaciones de modelado

Definimos una herramienta para realizar escultura virtual sobre nuestra representación mediante dos parámetros: el alcance (o volumen) de la herramienta y la función que asigna valores de propiedad dentro de dicho volumen (el *campo escalar*). El *alcance* determina el conjunto de *voxels* de la representación que van a ser modificados y la *función de asignación* determina el nuevo valor que será asignado a cada uno de los *voxels* involucrados en la operación. Hay que tener en cuenta que las modificaciones no van a ser exclusivamente cambios en los valores de propiedad de los *voxels*, sino que también se producirán agregaciones/subdivisiones de éstos.

Para comprobar que el método de modificación de nuestra estructura funciona correctamente se ha elegido como herramienta una esfera de radio R definida en forma implícita. El radio determina su alcance. La definición implícita permite una evaluación rápida de si un punto determinado está incluido en el interior de la esfera y, por tanto, nos permite realizar un test de inclusión rápido con nuestro *octree*. La función de asignación sigue una distribución uniforme de material, considerando el centro de la esfera como el punto de mayor densidad, decreciendo el valor de densidad conforme nos acercamos a

su frontera. El nuevo valor de propiedad que proporciona la función de asignación para un *voxel* incluido en el alcance de la herramienta tiene en cuenta la posición relativa de dicho *voxel* con respecto al centro de la esfera, así como el valor de propiedad actual de éste.

Esta asignación discreta de valores de propiedad a los *voxels* provoca efectos de *aliasing* en el objeto que se está modelando. Sin embargo, nuestro método de modificación local es independiente de la función de asignación utilizada, por lo que es posible utilizar otras funciones basadas en *kernels* que eviten este efecto.

Para detectar sobre qué parte de la isosuperficie se está aplicando la herramienta y cuando la herramienta no puede atravesar la isosuperficie (que es la única representación del objeto desde el punto de vista del usuario) se ha utilizado un método de detección de inclusión en *voxels* aproximado. La isosuperficie se calcula en base a las celdas que están definidas implícitamente en la representación. Sin embargo, estas celdas no tienen información de adyacencia asociada. Por otra parte, el *octree* permite una evaluación rápida de los *voxels* que se encuentran incluidos en la herramienta. Nuestra solución pasa por detectar los *voxels* incluidos en la herramienta y comprobar cuando éstos tienen celdas asociadas que son atravesadas por la isosuperficie. Cuando son detectados, la herramienta no puede avanzar hacia el interior del objeto hasta que su aplicación sucesiva provoque la falta de isosuperficie en dichas celdas. Aunque la solución es aproximada y no se detecta directamente la colisión de la herramienta con la geometría asociada a la isosuperficie, el efecto es difícilmente distinguible por el usuario y, sin embargo, permite incrementar enormemente la eficiencia del cálculo. Esto se debe a que los *voxels* que tienen asociada responsabilidad son pequeños debido a la asignación minimal de nuestra representación. La figura 4.1 muestra un ejemplo 2-D en el que se puede ver la detección de colisión aproximada. La imagen de la izquierda muestra el caso en el que la herramienta no ha llegado a colisionar con la geometría de la isosuperficie y sin embargo nuestro método detecta colisión porque se ha colisionado con un *voxel* con una celda dual asociada que contiene isosuperficie. En la imagen de la derecha se muestra el caso opuesto, en el cual, la herramienta ha atravesado la isosuperficie pero la colisión no se ha detectado porque todavía no se ha colisionado con el *voxel* que incluye a la celda correspondiente.

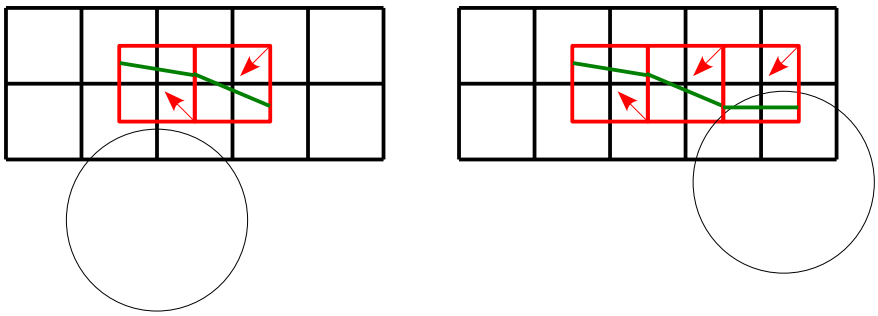

Figura 4.1: Ejemplo 2-D del método de detección de colisión aproximado isosuperficie-herramienta.

Debido a que hemos utilizado un dispositivo háptico de respuesta en fuerza para realizar la interacción sobre nuestro modelo, es necesario determinar cuando se empieza a provocar la respuesta en fuerza por parte de éste. Esta respuesta se lleva a cabo en cuanto se detecta la colisión con nuestro método aproximado. La respuesta en fuerza utiliza las funciones proporcionadas por la biblioteca H3DAPI [H3D] que simplemente

siguen la Ley de Hook. Para ilustrar el método de modificación local con respuesta interactiva se va a utilizar la herramienta para sustraer material. La herramienta para añadir material se implementa de forma similar con pequeñas variaciones.

4.2.2. Operación para sustraer material

La operación para sustraer material asigna un valor de propiedad, dentro de un rango permitido, a cada uno de los *voxels* que se encuentren incluidos en su alcance, en función de su distancia al centro de la esfera. El rango proporciona un valor mínimo por debajo del cual no se asigna valor y, de esta forma, se evitan asignaciones "en falso" que podrían prevenir el añadido de material en zonas vacías en las que, previamente, se hubiese realizado tal operación.

La aplicación de la herramienta provoca, aparte de la modificación de los valores de propiedad, una serie de agregaciones y subdivisiones en los *voxels* que constituyen nuestra representación. Estas situaciones llevan a distinguir varios casos. A continuación, se describen a nivel general los pasos de nuestro método de actualización y, posteriormente, detallaremos los distintos casos a tener en cuenta. Los pasos que constituyen el proceso de aplicación de la herramienta para sustraer material son los siguientes:

- Determinar el conjunto de *voxels*, V, que se encuentran bajo la influencia del alcance de la herramienta. Esto implica realizar un recorrido por el *octree* para determinar que nodos hoja estan al menos parcialmente incluidos en el volumen de la herramienta.

- A partir del conjunto V, efectuar los cambios necesarios en el *octree* de acuerdo con la modificación del valor de propiedad que efectuará la función de asignación de la herramienta. Esta modificación es la que puede implicar la necesidad de realizar operaciones de subdivisión y agregación de *voxels*.

- Realizar una reasignación de responsabilidades en los *voxels* del conjunto V cuya topología haya cambiado. Para optimizar este paso, en cada aplicación de la herramienta, se mantiene un conjunto de *voxels* subdivididos V_s y otro conjunto de *voxels* agregados V_a. La reasignación de responsabilidades se lleva a cabo sobre cada elemento de ambos conjuntos. Dividimos los *voxels* afectados en dos conjuntos debido a que cada tipo de *voxel* conlleva una clase de optimización distinta a la hora del traslado de responsabilidad. Los *voxels* que no son modificados no cambian sus responsabilidades, salvo en el caso en el cual éstas son modificadas por otros *voxels* adyacentes.

- Encontrar el conjunto de celdas, C, modificadas por la aplicación de la herramienta. Una celda pertenece a dicho conjunto si al menos uno de los *voxels* que configuran su topología pertenece al conjunto V. Como las celdas forman la rejilla dual al *octree* que se usa para la extracción, es necesario determinar el nuevo conjunto de celdas que podría surgir como resultado de las subdivisiones/agregaciones de los elementos de V.

- Extraer la geometría de todas las celdas activas contenidas en el conjunto C. Las celdas activas son las únicas que realmente es necesario procesar para extraer el trozo de isosuperficie que cambia tras la aplicación de la operación.

- Eliminar la geometría de las celdas que han dejado de ser válidas. Las celdas que han dejado de existir, debido a las subdivisiones/agregaciones de *voxels* generaron una geometría previamente, y esta geometría debe descartarse de cara a la actualización de la visualización del objeto modificado.

El primero de los pasos es determinar el conjunto de *voxels* V que van a ser modificados, bien cambiando solamente su valor de propiedad, bien sufriendo una agregación o una subdivisión. Para determinar los elementos del conjunto se realiza un recorrido del *octree*, encontrando los nodos hoja (*voxels*) que se encuentran incluidos en el alcance de la herramienta. Como resultado podemos obtener cuatro situaciones:

- El *voxel* ha sido descartado y no se encuentra en el alcance de la herramienta, por lo que directamente no se considera.

- El *voxel* es intersectado por la herramienta y está situado en el *octree* a un nivel menor que el nivel máximo permitido. En este caso, el *voxel* tiene que ser subdividido, y se procede recursivamente con cada uno de sus hijos para determinar su inclusión o su intersección con la herramienta. La subdivisión de un nodo provoca la eliminación de la geometría (triángulos) de todas las celdas de las que forma parte, independientemente de que tenga o no la responsabilidad de generarlas. La eliminación de la geometría extraida a partir de una celda se explica detalladamente en la sección 4.2.3. Cuando un *voxel* se subdivide recursivamente, sólo éste se almacena en el conjunto V_s, es decir, solamente se almacena el *voxel* que representa la raiz del subárbol de subdivisión. Esta optimización permite evitar la búsqueda de responsabilidades redundantes.

- El nodo que representa al *voxel* está situado en el nivel más bajo permitido en el *octree*, habiendo sido generado por subdivisión de un nodo superior y estando incluido en el alcance de la herramienta. En este caso se realiza la modificación del valor de propiedad en base a la función de asignación de la herramienta.

- El nodo que representa al *voxel* está situado en el nivel más bajo permitido en el *octree* y no ha sido generado al aplicar la herramienta. En este caso se realiza igualmente una modificación del valor de propiedad. Al cambiar el valor de propiedad del *voxel*, la geometría de las celdas de las que forma parte deja de ser válida, por lo que es necesario volver a generarla utilizando el mecanismo para la actualización de geometría que proporcionan los VBO. Adicionalmente, puede ocurrir, como muestra el ejemplo 2-D de la imagen izquierda de la figura 4.2, que la responsabilidad de generar la celda pertenezca a otro *voxel* (etiquetado como s en la figura) situado fuera del alcance de la herramienta. No obstante, el *voxel* que nos ocupa (etiquetado como r en la figura) está situado en el nivel más bajo del *octree*, por lo que, independientemente de lo que ocurra en los *voxels* adyacentes, siempre va a poder obtener la responsabilidad de generación de las celdas de las que forma parte. Las responsabilidades que adquiera este *voxel* r tras el proceso se incluyen en el conjunto de celdas C. Este es el caso de la responsabilidad adquirida por el *voxel* r que cae bajo la influencia de la herramienta y que se muestra en la imagen de la derecha en la figura 4.2.

La asignación de valores de propiedad de la herramienta provoca la agregación en los desdencientes de los *voxels* que han sido subdivididos, siempre que se cumpla el

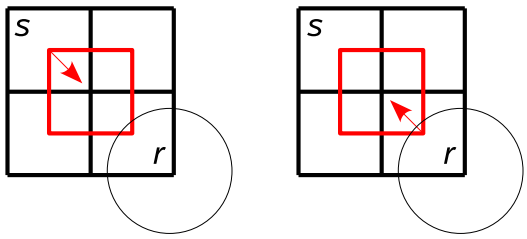

Figura 4.2: Un *voxel* r de máxima resolución bajo la influencia de una herramienta adquiere la responsabilidad de generación de una celda (rojo) cuya responsabilidad estaba asignada a un *voxel* s que no esta bajo la influencia de dicha herramienta.

criterio de similaridad establecido por el usuario. Al igual que la subdivisión, la agregación implica la modificación de la topología de las celdas que existía previamente a la reducción del número de *voxels*. Por tanto, es necesario igualmente eliminar la geometría de cualquier celda que conecte alguno de los *voxels* que van a ser agregados. Cuando se realiza la agregación de un *voxel* hay que tener en cuenta que cualquier otro elemento por debajo del nodo del *octree* que lo representa, el cual haya sido añadido a los conjuntos de *voxels* V_s o V_a, o al de celdas C, puede eliminarse para no tener que realizar operaciones innecesarias.

Una vez finalizadas todas las posibles subdivisiones y/o agregaciones que induce el alcance de la herramienta se lleva a cabo el proceso de reasignación de responsabilidades únicamente sobre los *voxels* incluidos en los conjuntos V_s y V_a. Este proceso consta de dos fases: búsqueda de *voxels* vecinos en la dirección de generación correspondiente y ejecución del algoritmo de asignación de responsabilidades entre los vecinos y el *voxel* considerado. Esta asignación genera celdas que deben ser añadidas al conjunto C para su posterior procesado.

Siguiendo nuestro método se consigue una modificación local de los valores de propiedad de los *voxels* influidos por la herramienta, así como una adaptación de la representación a la forma de la herramienta solamente en la parte del volumen en la cual ha sido aplicada.

Sin embargo, nuestra representación no almacena explícitamente la geometría que aproxima la isosuperficie extraida. Estos triángulos se extraen y se mandan directamente al hardware gráfico para permitir la visualización. Cuando se aplica una operación se hace necesario descartar los triángulos que no deben aparecer tras la modificación de la representación. Para solucionar este problema, sin tener que recurrir a una estructura geométrica de soporte adicional, lo cual penalizaría la respuesta interactiva de nuestro método, hemos utilizado un mecanismo que pertenece a las extensiones de OpenGL: los *Vertex Buffer Objects*. Su utilización para resolver nuestro problema se describe en la siguiente sección.

4.2.3. Actualización de la geometría a visualizar

Nuestra representación de volumen no mantiene de forma explícita la geometría que aproxima la isosuperficie extraida. Para modificar, eliminar o añadir nuevos fragmentos de isosuperficie es necesario habilitar un mecanismo que permita realizar estas operaciones sobre la memoria de la tarjeta gráfica, en la cual reside la geometría que se está visualizando. La solución que proponemos a este problema hace uso de una es-

tructura de datos geométricos proporcionada por una extensión de OpenGL: los *Vertex Buffer Objects* (VBO). En concreto, los VBO nos permiten:

- Almacenar la geometría del modelo en la memoria de la tarjeta gráfica y permitirnos el acceso identificado a dicha geometría. Con esto se aumenta la velocidad de la visualización de la isosuperficie.
- Almacenar información geométrica y atributos: vértices, normales, color,...
- Dibujar de forma independiente partes separadas de la geometría del modelo mediante la realización de llamadas a la función `DrawArrays`.
- Modificar la información geométrica de partes concretas del modelo mediante la realización de llamadas a la función `glBufferSubData`.

Mediante la utilización de un VBO se puede asociar a cada celda dual una parte del *buffer*, que es la que se encarga de mantener los vértices de los triángulos extraidos para esa celda. A la hora de descartar geometría durante las operaciones de subdivisión y agregación, simplemente se dejan huecos libres en el *buffer* cuyo tamaño depende del número de triángulos generados en la celda descartada.

En el VBO se va almacenando, de forma consecutiva, la geometría generada para cada celda dual que es atravesada por la isosuperficie. El espacio que ocupa la geometría asociada a una celda en el VBO viene representado por un par de la forma $<elemento\_inicial, longitud>$. La asociación de celdas a *voxels*, la cual es mantenida por nuestra representación mediante las responsabilidades de generación de celdas, determina que cada *voxel* tiene asociado un par de este tipo por cada responsabilidad de celda asignada que realmente contenga una porción de isosuperficie. La figura 4.3 representa la estructura VBO. En la primera mitad del *buffer* se almacenan los vértices de los triángulos y en la segunda la normal asociada a cada vértice.

Durante el proceso inicial de extracción de la isosuperficie completa, el VBO va almacenando va almacenando de forma consecutiva la geometría que se va extrayendo para cada celda activa. El estado final del VBO tras este proceso de extracción se representa en la imagen superior de la figura 4.3. Esta forma de proceder sigue el clásico enfoque de asignación de espacio libre a partir de un *pool*. No obstante, como ya ha sido descrito con anterioridad, al aplicar la herramienta a nuestra representación se producen cambios en la geometría de las celdas como consecuencia de la variación de los valores de propiedad de sus vértices, o las celdas dejan de ser válidas como consecuencia de la desaparición de alguno(s) de los *voxels* que conectan. Por consiguiente, su geometría asociada deja de ser válida y el espacio ocupado en el VBO es considerado como *huecos* de espacio libre a partir de ese momento. En la imagen central de la figura 4.3 se puede observa cómo, después de realizar las operaciones de subdivisión y/o agregación de *voxels*, la estructura presenta huecos de espacio libre en donde previamente se encontraba almacenada la geometría de las celdas que han dejado de ser válidas.

Posteriormente, cuando se procesan las celdas del conjunto C, se utiliza un algoritmo del primer ajuste [Sta05] para localizar el primer hueco, a partir del comienzo del VBO, en el que se pueda ubicar la geometría extraida a partir de las nuevas celdas. Siguiendo esta estrategia del primer ajuste se produce poca o nula fragmentación de espacio en el *pool* (VBO) debido a que el número de triángulos asociados a las celdas es muy similar y a que el tipo de caso de las celdas permite que usualmente haya huecos que han alojado celdas del mismo tipo con anterioridad. No obstante, como

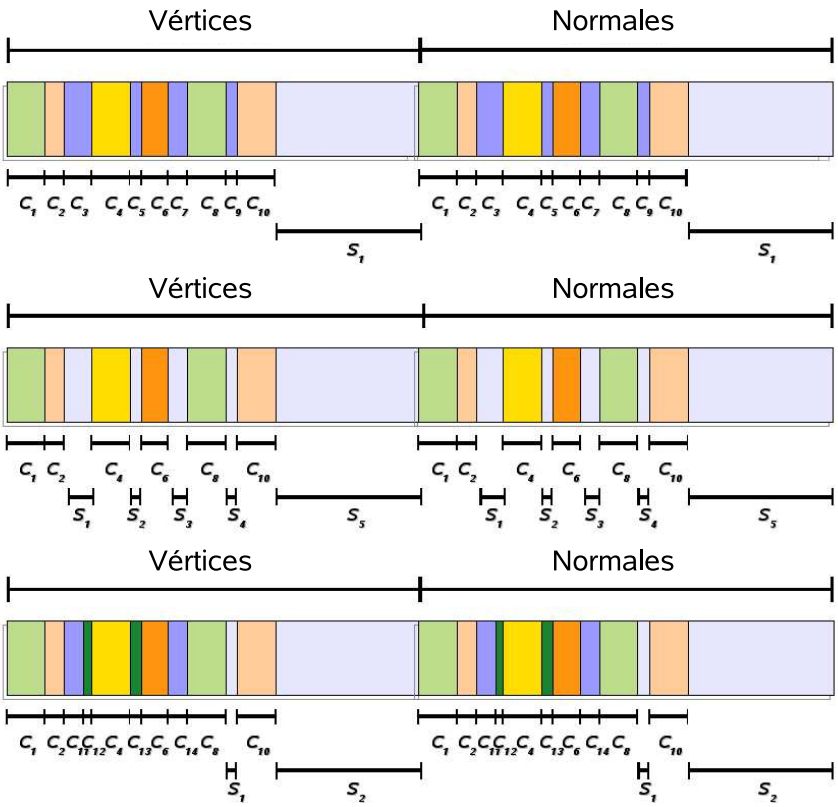

Figura 4.3: La imagen superior muestra el VBO tras la extracción inicial de la isosuperficie. Cada segmento C_n indica la zona ocupada por la geometría asociada a una celda. La imagen central muestra los huecos, marcados como S_n, que se han producido tras eliminar las celdas que han dejado de ser válidas. La imagen inferior muestra el estado del VBO tras añadir la geometría de las nuevas celdas que han sido procesadas.

muestra la imagen inferior de la figura 4.3, es probable que después de procesar las celdas del conjunto C queden huecos libres. Esto hecho no presenta ningún problema ya que la biblioteca proporciona funciones para visualizar las distintas secciones del VBO. Simplemente se hace necesario mantener una estructura adicional que almacene la información de espacio libre (huecos). La solución óptima pasa por utilizar un mapa de espacio libre [Sta05].

4.3. Resultados y conclusiones

Para comprobar que el método de actualización del *IDDG-Octree* permite una respuesta interactiva a la hora de modificar la representación, además de generar solamente la geometría local al alcance de la herramienta, se ha desarrollado un prototipo de aplicación para realizar escultura virtual. La aplicación permite la interacción a través de un dispositivo háptico que proporciona seis grados de libertad. Debido al uso de este dispositivo para la interacción sobre el modelo, es necesario calcular tanto la colisión de la herramienta con el modelo como la respuesta en fuerza para poder controlar el avance y retroalimentar el dispositivo. El cálculo de la colisión ya ha sido mostrado y, en cuanto a la respuesta en fuerza se utiliza una biblioteca (openhaptics) que resuelve

el problema con un modelo de respuesta muy sencillo, aunque suficiente para nuestras espectativas, basado en la ley de Hooke.

El prototipo implementa una herramienta esférica de adición de material y otra de sustracción representadas mediante una función implícita. No se han implementado otras formas para las herramientas porque nuestro objetivo ha sido probar la factibilidad de usar *IDDG-Octree* para la operación interactiva. No obstante, como todos los algoritmos implementados se basan en obtener el ámbito de alcance de la herramienta en base a la función implícita, la aplicación es fácilmente extensible a cualquier herramienta que pueda ser definida de esta forma. El prototipo se ha probado en un PC estándar con núcleo dual y 2 GB de memoria RAM. En cuanto a la estructura general de la aplicación es de resaltar que una hebra de procesamiento se encarga exclusivamente del cálculo de colisiones y respuesta en fuerza y su procesamiento se encarga exclusivamente a uno de los procesadores, mientras que existe otra hebra encargada de la modificación de la estructura y generación de la nueva geometría asignada al otro procesador. Se ha probado el prototipo con modelos de hasta 256^3 muestras y el resultado de las operaciones permitía el trabajo de forma interactiva.

La figura 4.4 muestra el resultado de la aplicación de sucesivas operaciones de añadir material sobre un cubo. La figura 4.5 muestra el resultado de la aplicación de sucesivas operaciones de sustraer material sobre un cubo. La figura 4.6 muestra en la imagen superior un estado intermedio de una sesión de escultura virtual cuyo resultado final se muestra en la imagen inferior.

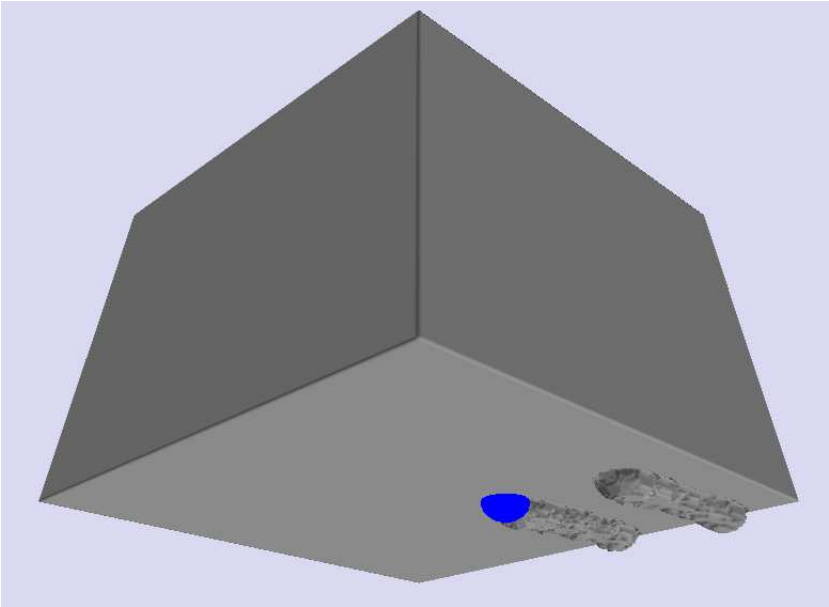

Figura 4.4: Resultado de la aplicación sucesiva de una herramienta esférica que permite añadir material sobre un objeto de forma cúbica.

Podemos concluir afirmando que se ha presentado un método de actualización del *IDDG-Octree* que permite la aplicación de esta representación a la escultura virtual. Debido a que el *octree* permite determinar los *voxels* que caen bajo el alcance de una herramienta, la modificación de la estructura es local, y solo es necesario procesar esta información. No obstante, al igual que se mostró en el capítulo de visualización, el *IDDG-Octree* presenta el inconveniente de no disponer explícitamente de la rejilla

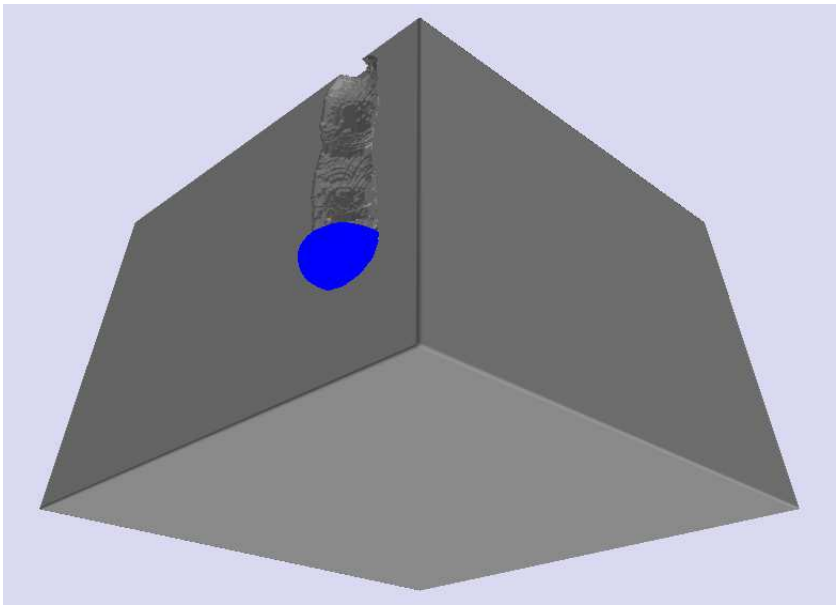

Figura 4.5: Resultado de la aplicación sucesiva de una herramienta esférica que permite sustraer material de un objeto de forma cúbica.

dual de celdas, por lo que el coste de la actualización de la estructura continua siendo la búsqueda de vecinos. Debido a que este es el gran cuello de botella de nuestra representación de cara a obtener buenos tiempos de extracción, en el apéndice A se muestra el método de búsqueda de vecinos que se ha implementado, el cual persigue optimizar al máximo dicha búsqueda en el *octree*.

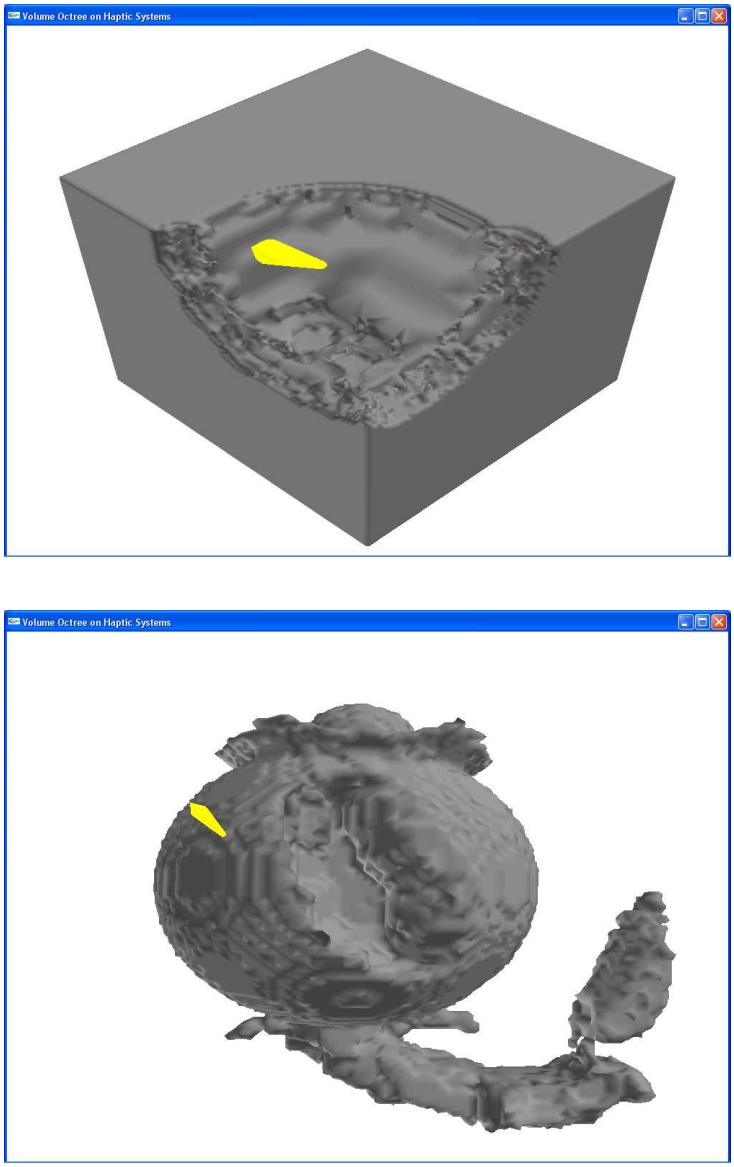

Figura 4.6: La imagen superior muestra un estado intermedio en una sesión de escultura virtual en la que se obtiene como resultado el objeto que se muestra en la imagen inferior.

CAPÍTULO 5

Conclusiones y trabajos futuros

En este último capítulo se resumen las principales aportaciones realizadas en esta memoria de tesis doctoral junto con las líneas abiertas de investigación que proporcionarán trabajos en el futuro.

5.1. Principales aportaciones

Se ha presentado una propuesta de representación basada en el enfoque de *voxels*, *HRB-Octree*, la cual permite reducir el espacio de almacenamiento utilizando el concepto de región homogénea, cuya idea subyace en el aprovechamiento de la coherencia espacial de los valores de propiedad de las muestras. Esta representación permite solucionar el problema de falta de continuidad que presenta el enfoque clásico de *voxels* mediante el establecimiento de una partición del espacio de *voxels* en dos zonas: interior y bordes. Esta nueva partición utiliza una función de interpolación distinta para cada tipo de región, consiguiendo continuidad C^0 entre las distintas regiones y, por consiguiente, en la partición de *voxels*.

Se ha presentado una representación híbrida del enfoque de *voxels* y el de celdas, *IDDG-Octree*, la cual se beneficia de la ventajas de ambos. Por una parte permite ahorrar espacio de almacenamiento gracias al *octree* de *voxels* y, por otra, gracias a la representación implícita de las celdas duales a dichos *voxels* permite realizar extracción de isosuperficies con resultados muy similares a los de las representaciones regulares y sin pagar el coste de mantener una estructura adicional.

Se ha formalizado el concepto de *voxel* minimal para el *IDDGO-Octree*, y a partir de él se ha desarrollado un método que garantiza la generación automática de la topología asociada a la rejilla dual de celdas, la cual se almacena implícitamente en nuestra representación. Además, para garantizar que la partición de celdas cubre completamente el volumen representado en el *octree*, se ha descrito un método para envolver con celdas duales los vértices de los *voxels* que caen sobre la caja englobante del volumen.

Se ha presentado un método de extracción de isosuperficies que sigue la estrategia de marchar sobre las celdas duales y que tiene en cuenta la diferencia de tamaño de los *voxels* que son conectados por dichas celdas. Para ello se restringe el segmento de arista en donde se realiza la interpolación en las aristas con cambio de signo a la zona

borde de los *voxels*. De esta forma se consigue una isosuperficie muy próxima a la que se puede obtener a partir de representaciones regulares aplicadas sobre el mismo conjunto de datos de entrada.

Se ha presentado un método de actualización del *IDDG-Octree* que permite la aplicación de esta representación a la escultura virtual. Se ha desarrollado un prototipo de aplicación de escultura virtual que soporta la aplicación de distintas herramientas de modelado sobre objetos representados mediante *IDDG-Octree*, la cual permite realizar un proceso de escultura virtual con respuestas en tiempo interactivo.

5.2. Líneas de investigación para trabajos futuros

A lo largo del desarrollo de este trabajo han surgido cuestiones que nos han planteado problemas que consideramos interesantes de cara a un estudio más detallado. A continuación se enumeran algunas de estas líneas de trabajo futuro:

- El criterio de similaridad utilizado para simplificar el *octree* se basa en la coherencia espacial del valor de propiedad de las muestras incluidas en una región. Pretendemos utililizar información adicional para obtener una tasa de simplificación mayor. El gradiente de propiedad es una medida prometedora que estamos barajando.

- Nuestro método de extracción de isosuperficies a partir de las celdas duales del *IDDG-Octree* permite que se pueda aplicar cualquier algoritmo de extracción basado en la idea de marchar sobre celdas cúbicas. Esto se debe a que resolvemos la tabla de casos sobre las celdas topológicas. Nuestro objetivo es estudiar las distintas configuraciones geométricas de las celdas duales y establecer una tabla de casos específica para estas celdas.

- A la hora de aplicar herramientas de modelado sobre el *IDDG-Octree*, solamente se han implementado operaciones de combinación de valores de propiedad sobre el volumen. Estas operaciones no conservan el volumen asociado a la representación. Pretendemos utilizar la representación para permitir la implementación de herramientas de deformación que conserven el volumen.

APÉNDICE A

Búsqueda de *voxels* vecinos de un *voxel* minimal en *IDDG-Octree*

La representación *IDDG-Octree* está diseñada para permitir ahorrar espacio de almacenamiento con respecto a las representaciones de resolución fija. Debido a este requisito, no mantiene explícitamente la rejilla dual de celdas que utiliza para realizar la extracción de isosuperficies. Por tanto, cuando es necesario generar total o parcialmente la isosuperficie se hace necesario generar la geometría de las celdas duales a partir de las responsabilidades de generación de celdas mantenidas por la representación. Esto conlleva realizar la búsqueda en el *octree* de los *voxels* adyacentes determinados por la dirección de cada responsabilidad. Esta búsqueda es el principal cuello de botella de nuestra representación en relación al tiempo de extracción de isosuperficie. Para paliar el efecto que provoca este requisito del diseño se ha desarrollado un método de búsqueda que permite optimizar el tiempo empleado. En este apéndice se describe dicho método.

A.1. Estructura de datos

La estructura de datos que soporta al *IDDG-Octree* (Implicitly-Defined Dual Grid *Octree*) está compuesta por dos *arrays* unidimensionales sincronizados. El primer *array*, `nodes`, representa las relaciones de jerarquía padre-hijo de los nodos del *octree*. Cada elemento puede contener un valor mayor que 0 (nodo interno), indicando el índice en el *array* del primero de sus ocho nodos hijo, o un valor igual a 0, indicando que dicho nodo es un nodo hoja. Los elementos que no han sido usados todavía tienen almacenado el valor -1. El primer elemento del *array* representa el nodo raíz del árbol.

El segundo *array*, `data`, representa los datos almacenados en los nodos hoja del árbol. Si el elemento del *array* se corresponde con un nodo hoja, entonces contiene la siguiente información:

- Valor de propiedad.
- Un *Byte* para almacenar las direcciones relativas de las celdas duales cuya responsabilidad de procesamiento recae en *voxel* que representa el nodo.

Si el elemento del *array* se corresponde con un nodo interno no se almacena ninguna información. La relación de correspondencia entre el *array* `nodes` y el *array* `data` determina la necesidad de sincronización de ambos, ya que establece que a cada elemento del primero le corresponde el elemento del segundo con el mismo índice. En la figura A.1 se representa gráficamente la estructura de datos equivalente para un *IDDG-Quadtree* de ejemplo situado en la parte derecha de la ilustración.

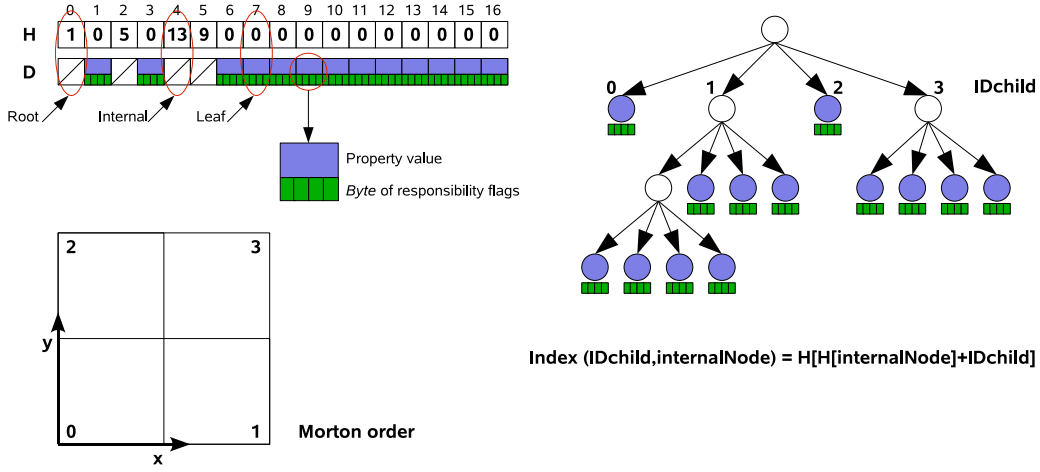

Figura A.1: Estructura de datos que implementa al *IDDG-Octree*.

A.2. Identificación unívoca de nodos

Para poder identificar unívocamente cada una de las ocho subregiones en las que se puede llegar a subdividir cada región del *octree* se utiliza el recorrido de Morton (*Morton order*). De esta forma, cada nodo hijo de un nodo interno tiene asociado un número desde el 0 al 7. Expresando en base binaria estas etiquetas identificativas se obtiene:

	zyx
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

En la tabla se puede comprobar que, siguiendo la ordenación de Morton, las etiquetas identificativas en base binaria coinciden completamente con las coordenadas z, y, x asociadas a cada una de las regiones en las que se subdivide una región dada.

Utilizando esta idea se puede expresar de forma unívoca la posición de cualquier nodo del *octree* siempre que conozcamos el nivel máximo del árbol. Supongamos, sin perdida alguna de generalidad, que el *octree* está situado en el primer octante, y que su máxima profundidad es l. Comenzando por el nivel superior del árbol, se puede ir almacenando en las variables x, y, z, para cada nivel, el valor del *bit* que se corresponde

con la subregión que representa el nodo, y así sucesivamente, cuando lleguemos a la región de interés, dispondremos de su posición exacta. En la figura A.2 se muestra la idea mediante un ejemplo sencillo usando un *quadtree*. La flecha roja indica el nodo (*voxel*) a localizar.

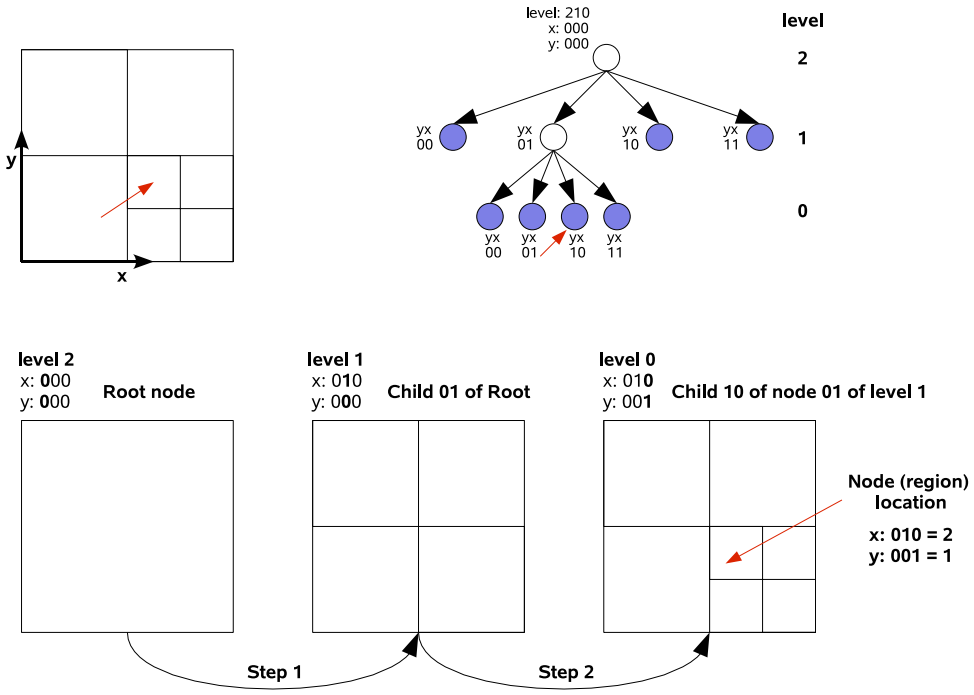

Figura A.2: Ejemplo 2-D que ilustra el procedimiento para asignar un identificador único a cada nodo de un *octree*.

A.3. Localización de *voxels* vecinos a uno dado

Para procesar la celda asociada a una dirección de responsabilidad de un *voxel* minimal es necesario buscar en el *octree* todos los *voxels* vecinos necesarios para generar la celda.

Partimos del hecho de que se está recorriendo el árbol para generar la isosuperficie (o una parte de ésta) y, por tanto, se dispone de la localización exacta del *voxel* cuya responsabilidad estamos procesando: *voxel actual*. Es decir, conocemos su posición (x, y, z) y su nivel en el árbol, l. El procedimiento se divide en dos tareas principales:

- Determinar la *clave de identificación* de cada uno de los *voxels* vecinos al *voxel* actual en la dirección determinada por la responsabilidad de procesamiento de celda.
- Procesar cada clave de identificación para obtener la localización de su *voxel* correspondiente.

La clave de identificación se determina en función de la dirección asociada a la responsabilidad. Esta dirección determina que *voxels* vecinos forman la celda. Tomando como punto de referencia la localización del *voxel* actual y su nivel se calculan las claves de los hipotéticos *voxels* vecinos de la misma resolución que el *voxel* actual. De

hecho, estos *voxels* vecinos calculados no tienen porqué existir realmente, ya que pueden encontrarse agregados en niveles superiores. Lo único que garantiza la propiedad de minimalidad en la asignación de responsabilidades es que nunca se podrán encontrar subdivididos. La figura A.3 muestra una representación gráfica de la determinación de las claves de identificación para un *voxel* actual (indicado por la flecha roja) y una responsabilidad de procesamiento de la celda en dirección NE (indicada por la flecha verde). Los *voxels* vecinos, hipotéticos como se puede comprobar, están resaltados en amarillo. La ordenación de los vértices de la futura celda, establecida a priori, es la que se muestra asociada al *voxel* actual y a sus vecinos.

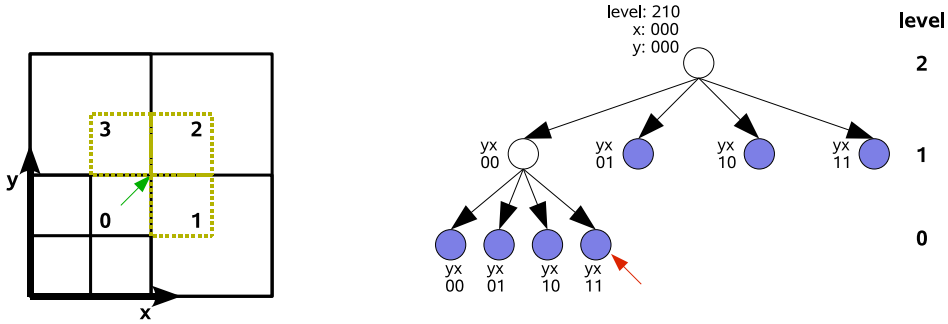

Figura A.3: Cálculo de las claves de identificación de los *voxels* vecinos al *voxel* actual (flecha roja) en la dirección NE (flecha verde).

Para ilustrar el procedimiento se muestran a continuación las claves de identificación asociadas al ejemplo de la figura A.3. La posición de la región activa es $x = 001, y = 001$ y el nivel $l = 2$. Para calcular la posición de cada vecino hay que sumar (en este caso) el incremento adecuado a cada coordenada, pero teniendo en cuenta que dicho incremento hay que sumarlo en la posición de la coordenada adecuada al nivel del *voxel* actual. De esta forma, como el nivel es 2, tendremos que sumar 001. Si el nivel hubiese sido igual a 1, habríamos tenido que sumar 010, para obtener los *voxels* de la misma resolución que la activa.

$$\begin{array}{ll} R_{0x} = 001 & R_{0y} = 001 \\ R_{1x} = 010 & R_{1y} = 001 \\ R_{2x} = 010 & R_{2y} = 010 \\ R_{3x} = 001 & R_{3y} = 010 \end{array}$$

Ya solo resta procesar cada clave de identificación para obtener la localización del nodo que representa el *voxel* vecina real en el árbol. La idea es simplemente buscar desde el nodo raiz la clave de identificación hasta llegar a un nodo hoja, independientemente de que se procese completamente la clave o no se llegue a procesar porque el *voxel* vecino real tiene agregado el *voxel* correspondiente a la clave. Debido a que en el *IDDG-Octree* está búsqueda se realiza en multitud de ocasiones, se ha desarrollado un algoritmo eficiente que minimiza el acceso a memoria principal, realizando éste solamente cuando es extrictamente necesario. Por tanto, se utilizan explícitamente registros de la CPU y operadores a nivel de *bit*: $RAL(\cdot)$, que desplaza un *bit* hacia la izquierda el contenido de un registro y $RAR(\cdot)$, que desplaza el contenido un *bit* hacia la derecha. El pseudocódigo del algoritmo se muestra a continuación.

1. Cargar la clave de identificación en tres registros de la CPU: `x,y,z`

2. Cargar en un registro, a partir de una tabla de búsqueda, la máscara de bits que presenta un bit a 1 en la posición que se corresponde con el primer nivel a partir del nodo raiz del árbol y el resto de bits a 0: `selec`

3. Cargar un registro e inicializarlo a 0. Este registro se usa para localizar uno de los hijos de un nivel determinado del árbol. `offset=`$0x0$

4.
```
node=root; flag=0;
while(!flag) {
    offset= z & selec;
    RAL(offset);
    offset & (y & selec);
    RAL(offset);
    offset & (x & selec); //offset= zyx de level actual.
    if(Heap[(node=Heap[node]+offset)] == 0) // Nodo hoja
        flag=1;
    else {
        offset=0x0;
        RAR(selec);
    }
}
// ‘‘offset’’ contiene la posición del \emph{voxel} vecino
//  real en el árbol.
// ‘‘Data[node]’’ permite accecer a los datos del nodo
//  que representa al \emph{voxel} vecino.
```

Referencias bibliográficas

[A.87a] Kaufman A.
An algorithm for 3d scan conversion of polygons.
In *Proc. Eurographics 87*, pages 197–208, August 1987.
North Holland, Amsterdam.

[A.87b] Kaufman A.
Efficient algorithms for 3d scan conversion of parametric curves, surfaces, and volumes.
Proc. Siggraph 87ACM Computer Graphics, 21(4):171–179, July 1987.

[AFH81] E. Artzy, G. Frieder, and G.T. Herman.
The theory, design, implementation and evaluation of a three-dimensional surface detection algorithm.
Computer Graphics and Image Processing, (15):1–24, 1981.

[AS96] R.S. Avila and L.M. Sobierajski.
A haptic interaction method for volume visualization.
In *Proceedings Visualization'96*, pages 197–204, San Francisco, CA, USA, October 27–November 1 1996.

[Bær98] A. Bærentzen.
Octree-based volume sculpting.
In *Short Presentations Proceedings of Visualization'98*, 1998.

[Bai00] M. Bailey.
Manufacturing isovolumes.
In Arie E. Kaufman Min Chen and Roni Yagel, editors, *Volume Graphics*, pages 79–93. New York: Springer, 2000.

[BBB91] J.A. Broekhuijsen, R.P. Burton, and W.A. Barrett.
Interactive editing of volumetric objects with 3d input and output devices.
Journal of Imaging Technology, 17(6):269–274, 1991.

[BC02] A. Bærentzen and N.J. Christensen.
Volume sculpting using the level-set method.
In *Shape Modeling International (SMI'02)*, pages 175–182, Banff, Alberta, Canada, May 17–22 2002.

[Ben75] J.L. Bentley.
Multidimensional binary search trees used for associative searching.
Communications of the ACM, 18(3):509–517, 1975.

[BFCG04] Renaud Blanch, Eric Ferley, Marie-Paule Cani, and Jean-Dominique Gascuel.

Non-realistic haptic feedback for virtual sculpture.
Technical Report RR-5090, INRIA, U.R. Rhone-Alpes, January 2004.
Projets ARTIS et EVASION, theme 3.

[BL95] J.R. Bill and S.K. Lodha.
Sculpting polygonal models using virtual tools.
In *Proceedings Graphics Interface'95*, pages 272–278, Quebec, Quebec, Canada, May 17–19 1995.

[Bli82] James F. Blinn.
A generalization of algebraic surface drawing.
ACM Trans. on Graphics, 1(3):235–256, July 1982.

[Blo88] J. Bloomenthal.
Polygonization of implicit surfaces.
Computer Aided Geometric Design, 5(4):341–355, 1988.

[Blo94] J. Bloomenthal.
Graphic Gems, volume IV, chapter IV.8.
Academic Press, Cambridge, 1994.

[Blo97] Jules Bloomenthal, editor.
Introduction to Implicit Surfaces.
Morgan Kaufmann Publishers, inc., 1997.

[BM00] D. Bartz and M. Meissner.
Volume Graphics, chapter Voxels versus polygons: a comparative study for volume graphics, pages 171–184.
New York: Springer, 2000.

[BMM+07] I. Braude, J. Marker, K. Museth, J.Ñissanov, and D. Breen.
Contour-based surface reconstruction using mpu implicit models.
Graphical Models, 69(1):139–157, 2007.

[BMTB02] L. Balmelli, C.J. Morris, G. Taubin, and F. Bernardini.
Volume warping for adaptive isosurface extraction.
In *Proceedings IEEE Visualization 2002*, pages 467–474, Boston, MA, USA, October 27–November 1 2002.

[BS98] C.L. Bajaj and D.R. Schikore.
Topology preserving data simplification with error bounds.
Computer & Graphics, 22(1):3–12, 1998.

[BW01] K. Brodlie and J. Wood.
Recent advances in volume visualization.
Computer Graphics forum, 20(2):125–148, 2001.

[CFM+94] P. Cignoni, L.D. Floriani, C. Montani, E. Puppo, and R. Scopigno.
Multiresolution modeling and visualization of volume data based on simplicial complexes.
In *1994 Symposium on Volume Visualization*, pages 19–26, Tysons Corner, Virginia, USA, October 17–18 1994.

[CFM+04] P. Cignoni, L. Floriani, P. Magillo, E. Puppo, and R. Scopigno.
Selective refinement queries for volume visualization of unstructured tetrahedral meshes.

IEEE Transactions on Visualization and Computer Graphics, 10(1):29–45, Jan-Feb 2004.

[CGMS00] P. Cignoni, F. Ganovelli, C. Montani, and R. Scopigno.
Reconstruction of topologically correct and adaptive trilinear surfaces.
Computers and Graphics, 24(3):399–418, 2000.

[Che95] Evgeni V. Chernyaev.
Marching cubes 33: Construction of topologically correct isosurfaces.
Technical report, Institute for High Energy Physics, Moscow, Russia, 1995.

[CLL88] H.E. Cline, W.E. Lorensen, and S. Ludke.
Two algorithms for the three-dimensional reconstruction of tomograms.
Medical Physiscs, 15(3):320–327, May/June 1988.

[CLLC90] Shiuh-Yung Chen, Wei-Chung Lin, Cheng-Chung Liang, and Chin-Tu Chen.
Improvement on dynamic elastic interpolation technique for reconstructing 3-d objects from serial cross sections.
IEEE Trans. on Medical Imaging, 9(1):71–83, 1990.

[CMPS97] P. Cignoni, C. Montani, E. Puppo, and R. Scopigno.
Multiresolution representation and visualization of volume data.
IEEE Transactions on Visualization and Computer Graphics, 3(4):352–369, 1997.

[CMSS96] P. Criscione, C. Montani, R. Scateni, and R. Scopigno.
Discmc: an interactive system for fast fitting isosurfaces on volume data.
In *Proceedings Eurographics WS on Virtual environments and scientific visualization'96*, pages 178–190, Monte Carlo, Monaco, February 19–20 1996.

[Com85] P.C. Come.
Diagnostic Cardiology, noninvasive imaging techniques.
J.B. Lippincott Company, 1985.

[CP98] S.L. Chan and E.O. Purisima.
A new tetrahedral tesselation scheme for isosurface generation.
Computer & Graphics, 22(1):83–90, 1998.

[CS78] H.N. Christiansen and T.W. Sederberg.
Conversion of complex contour line definitions into polygonal element mosaics.
Computer Graphics, 12:187–192, 1978.

[CTM03] H. Carr, T. Theuβl, and T. Möller.
Isosurfaces on optimal regular samples.
In *Proceedings Symposium on Data Visualization 2003*, pages 39–48, Grenoble, France, May 26–28 2003.

[CYH+97] T. Chiueh, C. Yang, T. He, H. Pfister, and A. Kaufman.
Integrated volume compression and visualization.
In *Proceedings IEEE Visualization'97*, pages 329–336, Phoenix, AZ, USA, October 19–24 1997.

[DCH88] R.A. Drebin, L. Carpenter, and P. Hanrahan.
Volume rendering.

ACM Computer Graphics (Proceedings of SIGGRAPH'88), 22(4):65–74, 1988.

[Die88] R. Diehl.
Conversion of boundary representations to bintrees.
In *Eurographics'88*, pages 117–127, Nice, France, 12–16 September 1988.

[DNFM90] R.A. Drebin, D.R. Ney, E.K. Fishman, and D. Magid.
Volumetric rendering of computed tomography data: Principles and techniques.
IEEE Computer Graphics and Applications, 10(2):24–32, 1990.

[ea89] B. Friedman et al.
Principles of MRI.
McGraw Hill, 1989.

[FAG83] H. Fuchs, G.D. Abram, and E.D. Grant.
Near real-time shaded display of rigid objects.
ACM Computer Graphics, 17(3):65–72, 1983.

[FCG99] Eric Ferley, Marie-Paule Cani, and Jean-Dominique Gascuel.
Practical volumetric sculpting.
In *Implicit Surfaces'99*, Bordeaux, France, September 1999.

[FCG00] Eric Ferley, Marie-Paule Cani, and Jean-Dominique Gascuel.
Practical volumetric sculpting.
the Visual Computer, 16(8):469–480, dec 2000.
A preliminary version of this paper appeared in Implicit Surfaces'99, Bordeaux, France, sept 1999.

[FCG02] Eric Ferley, Marie-Paule Cani, and Jean-Dominique Gascuel.
Resolution adaptive volume sculpting.
Graphical Models (GMOD), 63:459–478, march 2002.
Special Issue on Volume Modelling.

[Fer04] Randima Fernando, editor.
GPU Gems. Programming Techniques, Tips, and Tricks for Real-Time Graphics.
Addison-Wesley, 2004.

[FGR85] G. Frieder, D. Gordon, and R.A. Reynolds.
Back-to-front display of voxel-based objects.
IEEE Computer Graphics and Applications, 5(1):52–60, 1985.

[FK85] K. Fujimura and T. Kunii.
A hierarchical space indexing method.
In *Computer Graphics: Visual technology and Art, Proceedings of Computer Graphics Tokyo'85 Conference*, pages 21–34, New Orleans, Louisiana, USA, July 23–28 1985.

[FKN80] H. Fuchs, Z. Kedem, and B.Ñaylor.
On visible surface generation by a priori tree structures.
ACM Computer Graphics, 14(3):124–133, 1980.

[FKU77] H. Fuchs, Z.M. Kedem, and S.P. Uselton.
Optimal surface reconstruction from planar contours.
Communications of the ACM, 20(10):693–702, 1977.

[FPRJ00] S. F. Frisken, R.Ñ. Perry, A. P. Rockwood, and T. R. Jones.
Adaptively sampled distance fields: A general representation of shape for computer graphics.
In *Proceedings SIGGRAPH 2000, ACM Press*, pages 249–254, New Orleans, Louisiana, USA, July 23–28 2000.

[FvDFH95] J.D. Foley, A. van Dam, S.K. Feiner, and J.F. Hughes.
Computer Graphics: Principles and Practice in C (2nd ed.).
Addison-Wesley Professional, 1995.

[FY94] J. Fowler and R. Yagel.
Lossless compression of volume data.
In *1994 Symposium on Volume Visualization*, pages 43–50, Tysons Corner, Virginia, USA, October 17–18 1994.

[FYK84] K. Fujimura, K. Yamaguchi, and T. Kunii.
Octree-related data structures and algorithms.
IEEE Computer Graphics and Applications, 4(1):53–59, 1984.

[Gar82a] I. Gargantini.
An effective way to represent quadtrees.
Communications of the ACM, 25(12):905–910, 1982.

[Gar82b] I. Gargantini.
Linear octrees for fast processing of three-dimensional objects.
Computer Graphics and Image Processing, 20(4):365–374, 1982.

[GH91] T.A. Galyean and J.F. Hughes.
Sculpting: An interactive volumetric modeling technique.
ACM Computer Graphics, 25(4):267–274, 1991.

[GH95] A. Guéziec and R. Hummel.
Exploiting triangulated surface extraction using tetrahedral decomposition.
IEEE Transactions on Visualization and Computer Graphics, 1(4):328–342, 1995.

[Gib98] S. Gibson.
Using distance maps for accurate surface representation in sampled volumes.
In *Proceedings IEEE Symposium on Volume Visualization '98*, pages 23–30, 1998.

[GK04] A. Gress and R. Klein.
Efficient representation and extraction of 2-manifold isosurfaces using kd-trees.
Graphical Models, 66(6):370–397, November 2004.

[Gos94] M.E. Goss.
An adjustable gradient filter for volume visualization image enhancement.
In *Proceedings Graphics Interface'94*, pages 67–74, 1994.

[GP00] T. Gerstner and R. Pajarola.
Topology preserving and controlled topology simplifying multiresolution isosurface extraction.
In *Proceedings Visualization 2000*, pages 259–266, 2000.

[GR00] T. Gerstner and M. Rumpf.
Volume Graphics, chapter Multi-resolutional parallel isosurface extraction based on tetrahedral bisection, pages 267–278.
New York: Springer, 2000.

[H3D] *H3D API Manual*.
http://www.h3dapi.org.

[HB94] C.T. Howie and E.H. Blake.
The mesh propagation algorithm for isosurface extraction.
Computer Graphics Forum (Eurographics'94), 13(3):65–74, 1994.

[HK03] W. Hong and A. Kaufman.
Feature preserved volume simplification.
In *Proceedings ACM Symposium on Solid Modeling and Applications 2003*, pages 334–339, Seattle, Washington, USA, June 16–20 2003.

[HL79] Gabor T. Herman and Hsun Kao Liu.
Three-dimensional display of human organs from computed tomograms.
Computer Graphics and Image Processing, (9):1–21, 1979.

[HQ02] J. Hua and H. Qin.
Haptics-based volumetric modeling using dynamic spline-based implicit functions.
In *Proceedings IEEE Symposium on Volume Visualization and Graphics*, pages 55–64, 2002.

[HTF97] B. Hamann, I.J. Trotts, and G. Farin.
Approximating contours of the piecewise trilinear interpolant using triangular rational quadratic bézier patches.
IEEE Transactions on Visualization and Computer Graphics, 3(3):215–227, 1997.

[HU83] Gabor T. Herman and Jayaram K. Udupa.
Display of 3-d digital images: Computational foundations and medical applications.
IEEE Computer Graphics and Applications, 3(5):39–46, 1983.

[HW90] M. Hall and J. Warren.
Adaptive polygonalization of implicitly defined surfaces.
IEEE Computer Graphics and Applications, 10(6):33–42, 1990.

[HZB92] G.T. Herman, J. Zheng, and C.A. Bucholtz.
Shape-based interpolation.
IEEE Computer Graphics and Applications, 12(3):69–79, 1992.

[IK94] T. Itoh and K. Koyamada.
Isosurface generation by using extrema graphs.
In *Proceedings of Visualization'94*, pages 77–83, Washington, DC, USA, October 17–21 1994.

[JBS06] M.W. Jones, J.A. Baerentzen, and M. Sramek.
3d distance fields: A survey of techniques and applications.
IEEE Transactions on Visualization and Computer Graphics, 12(4):581–599, 2006.

[JC94] M.W. Jones and M. Chen.

A new approach to the construction of surfaces from contour data.
Computer Graphics Forum, 13(3):75–83, 1994.

[JLSW02] T. Ju, F. Losasso, S. Schaefer, and J. Warren.
Dual contouring of hermite data.
ACM Transactions on Graphics, 21(3):339–346, 2002.

[JS01] M.W. Jones and R. Satherley.
Using distance fields for object representation and rendering.
In *Proceedings Eurographics'01*, pages 37–44, 2001.

[JX96] E.K.-Y. Jeng and Z. Xiang.
Moving cursor plane for interactive sculpting.
ACM Transactions on Graphics, 15(3):211–222, 1996.

[Kau90] A.E. Kaufman.
Volume Visualization.
IEEE Computer Society Press, August 1990.
Los Alamitos, Calif.

[KBSS01] L. P. Kobbelt, M. Botsch, U. Schwanecke, and H. P. Seidel.
Feature-sensitive surface extraction from volume data.
In *SIGGRAPH 2001 ACM Press/ACM SIGGRAPH*, pages 57–66, August 2001.

[KCY93] A. Kaufman, D. Cohen, and R. Yagel.
Volume graphics.
IEEE Computer, 26(7):51–64, 1993.

[Kep75] E. Keppel.
Approximating complex surfaces by triangulation contour lines.
Technical Report 19, IBM J. Res. Develop., 1975.

[KFDB07] P. Komma, J. Fischer, F. Duffner, and D. Bartz.
Lossless volume data compression schemes.
In *Proceedings Simulation and Visualization 2007*, pages 1–14, 2007.

[KR04] Reinhard Klette and Azriel Rosenfeld.
Digital Geometry. Geometric Methods for Digital Picture Analysis.
Morgan Kaufmann, 2004.

[KS86] A. Kaufman and E. Shimony.
3d scan-conversion algorithms for voxel-based graphics.
In *Proceedings of 1986 Workshop on Interactive 3D Graphics*, pages 51–64, 1986.

[LB03] A. Lopes and K. Brodlie.
Improving the robustness and accuracy of the marching cubes algorithm for isosurfacing.
IEEE Transactions on Visualization and Computer Graphics, 9(1):16–29, 2003.

[LC87] William E. Lorensen and Harvey E. Cline.
Marching cubes: A high resolution 3d surface construction algorithm.
SIGGRAPH'87– ACM Computer Graphics, 21(4):163–169, July 1987.

[LCN98] B. Lichtenbelt, R. Crane, and S.Ñaqvi.
Introduction to Volume Rendering.

Hewlett-Packard Professional Books, 1998.

[Lev88] M. Levoy.
Display of surfaces from volume data.
IEEE Computer Graphics and Applications, 8(3):29–37, 1988.

[Lev90] M. Levoy.
Efficient ray tracing of volume data.
ACM Transactions on Graphics, 9(3):245–261, 1990.

[LGK97] L. Lippert, M. Gross, and C. Kurmann.
Compression domain volume rendering for distributed environments.
In *Proceedings EUROGRAPHICS'97*, pages 95–107, 1997.

[LLC88] Wei-Chung Lin, Cheng-Chung Liang, and Chin-Tu Chen.
Dynamic elastic interpolation for 3-d medical image reconstruction from serial cross sections.
IEEE Trans. on Medical Imaging, 7(3):225–232, 1988.

[Lop99] Adriano Lopes.
Accuracy in Scientific Visualization.
PhD thesis, University of Leeds, Leeds, UK, 1999.

[LSJ96] Y. Livnat, H. Shen, and C.R. Johnson.
A near optimal isosurface extraction algorithm using the span space.
IEEE Transactions on Visualization and Computer Graphics, 2(1):73–84, 1996.

[LTV06] A. León, J.C. Torres, and F. Velasco.
Interacción en tiempo real para un sistema de escultura virtual.
In *Proceedings (Short Papers), Ibero-American Symposium in Computer Graphics, SIACG 2006*, pages 84–87. Technological Research Institute, University of Santiago de Compostela, July 2006.

[LTV07a] Alejandro León, Juan Carlos Torres, and Francisco Velasco.
Octree de volumen con celdas duales implícitas.
In Eva Cerezo Bagdasari and Francisco R. Feito Higueruela, editors, *Actas del XVII Congreso Español de Informática Gráfica, CEIG'07 (CEDI 2007)*, pages 189–198. Eurographics, s.e., International Thomson Editores, S.A., September 2007.

[LTV07b] Alejandro León, Juan Carlos Torres, and Francisco Velasco.
Representación Compacta de Volúmenes Discretos, chapter VI, pages 127–140.
Universidad de Jaén, 2007.

[LTV08] A. León, J.C. Torres, and F. Velasco.
Volume octree with an implicitly defined dual grid.
Computer & Graphics, 32(4):393–401, August 2008.

[LTVS07] Alejandro León, Juan Carlos Torres, Francisco Velasco, and Francisco Soler.
Escultura Virtual sobre un Octree con Celdas Duales Implícitas, chapter VII, pages 141–154.
Universidad de Jaén, 2007.

[LVS08] A. León, F. Velasco, and F. Soler.

Interacción en tiempo real para un sistema de escultura virtual.
In Luis Matey and Juan Carlos Torres, editors, *Actas del XVIII Congreso Español de Informática Gráfica, CEIG'08*, pages 105–111. Universitat Politècnica de Catalunya. Eurographics, s.e., Eurographics Association, Sep 5–8 2008.

[Mar03] Carlos Sánchez Martín.
Microscopía confocal.
Technical report, Servicio de Microscopía Optica y Confocal, 2003.

[Mea80] D. Meagher.
Octree encoding: a new technique for the representation, manipulation and display of arbitrary three dimensional objects by computer.
Technical report, IPL-TR-80-111, Polytechnic Inst., Revisselaer, 1980.

[Mea82] D. Meagher.
Geometric modelling using octree encoding.
Computer Graphics and Image Processing, 19(2):129–147, 1982.

[ML94] S.R. Marschner and R.J. Lobb.
An evaluation of reconstruction filters for volume rendering.
In *Proceedings Visualization'94*, pages 100–107, 1994.

[MOT98] S. Mizuno, M. Okada, and J. Toriwaki.
Virtual sculpting and virtual woodcut printing.
The Visual Computer, 14(2):39–51, 1998.

[MQW01] K.T. McDonnel, H. Qin, and R.A. Wlodarczyk.
Virtual clay: A real-time sculpting system with haptic toolkits.
In *Proceedings Symposium on Interactive 3D Graphics*, pages 179–190, 2001.

[MS93] H. Muller and M. Stark.
Adaptive generation of surfaces in volume data.
The Visual Computer, 9(4):182–199, 1993.

[MSS94] Claudio Montani, Riccardo Scateni, and Roberto Scopigno.
A modified look-up table for implicit disambiguation of marching cubes.
The Visual Computer, 10(6):353–355, December 1994.

[MTY00] M. Matsumiya, H. Takemura, and N. Yokoya.
An inmersive modeling system for 3d free-form design using implicit surfaces.
In *Proceedings ACM Symposium on Virtual Reality Software and Technology*, pages 67–74, 2000.

[MW96] J. Menon and B. Wyvill.
Implicit surfaces for geometric modeling and computer graphics.
In *Proceedings SIGGRAPH'96 (course notes)*, New Orleans, Louisiana, USA, August 4–9 1996.

[Mä88] Martti Mäntylä.
An introduction to Solid Modeling.
Computer Sciencie Press, 1988.
Rockville, Maryland.

[NAT90] N.Ñaylor, J. Amanatides, and W. Thibault.

Merging bsp trees yields polyhedral set operations.
ACM Computer Graphics, 24(4):115–124, 1990.

[Nat94] B.K. Natarajan.
On generating topologically consistent isosurfaces from uniform samples.
The Visual Computer, 11(1):52–62, 1994.

[Nay90a] B.Ñaylor.
Binary space partitioning trees as an alternative representation of polytopes.
Computer Aided Design, 22(4):250–252, 1990.

[Nay90b] B.Ñaylor.
Sculpt: an interactive solid modeling tool.
In *Proceedings Graphics Interface'90*, pages 138–148, 1990.

[NB93] P.Ñing and J. Bloomenthal.
An evaluation of implicit surface tilers.
IEEE Computer Graphics & Applications, 13(6):33–41, November 1993.

[NH91] G.M Nielson and B. Hamann.
The asymptotic decider: resolving the ambiguity in marching cubes.
In *Proceedings IEEE Visualization'91*, pages 83–91, 1991.

[NH93] P.Ñing and L. Hesselink.
Fast volume rendering of compressed data.
In *Proceedings IEEE Visualization '93*, pages 11–18, 1993.

[Nie03] G.M. Nielson.
On marching cubes.
IEEE Transactions on Visualization and Computer Graphics, 9(3):283–297, 2003.

[Nie04] Nielson:2004.
Dual marching cubes.
In *Proceedings IEEE Visualization'04*, pages 489–496, 2004.

[NY06] T.S. Newman and H. Yi.
A survey of the marching cubes algorithm.
Computer & Graphics, 30(5):854–879, 2006.

[OPC96] J-M Oliva, M. Perrin, and S. Conquillart.
3d reconstruction of complex polyhedral shapes from contours using a simplified generalized voronöi diagram.
Computer Graphics Forum, 15(3):397–408, 1996.

[OR97] M. Ohlberger and M. Rumpf.
Hierarchical and adaptive visualization on nested grids.
Computing, 59:269–285, 1997.

[Paw90] James B. Pawley.
Handbook of Biological Confocal Microscopy.
Plenum Press, 1990.

[PF01] R.Ñ. Perry and S. F. Frisken.
Kizamu: A system for sculpting digital characters.
In *Prodeedings SIGGRAPH 2001, ACM Press*, pages 47–56, August 2001.

[Pho75] B.T. Phong.
Illumination for computer generated pictures.
Communications of the ACM, 18(6):311–317, 1975.

[PT90] B.A. Payne and A.W. Toga.
Surface mapping brain function on 3d models.
IEEE Computer Graphics and Applications, 10(5):33–41, 1990.

[PT92] B.A. Payne and A.W. Toga.
Distance field manipulation of surface models.
IEEE Computer Graphics and Applications, 12(1):65–71, 1992.

[PTWH90] A. Pommert, U. Tiede, G. Wiebecke, and K.H. Hohne.
Surface shading in tomographic volume visualizacion.
In *Proceedings First Conference on Visualization in Biomedical Computing*, pages 19–26, 1990.

[PWFO01] K. Perng, W. Wang, M. Flanagan, and M. Ouhyoung.
A real-time 3d virtual sculpting tool based on modified marching cubes.
In *Proceedings International Conference on Artificial Reality and Tele-existence*, pages 64–72, 2001.

[PY90] M.S. Paterson and F.F. Yao.
Optimal binary space partitions for orthogonal objects.
Journal of Algorithms, 13(1):99–113, 1990.

[RBB90] A.E. Richardson, R.P. Burton, and W.A. Barret.
Sculpt-box - a volumetric environment for interactive design of 3d objects.
In *Proceedings 1990 SPIE/SPSE Symposium on Electronic Imaging Science and Technology*, pages 198–209, 1990.

[RE00] A. Raviv and G. Elber.
Three-dimensional freeform sculpting via zero sets of scalar trivariate functions.
Computer Aided Design, 32(8-9):513–526, 2000.

[Req77] A. A. G. Requicha.
Mathematical models of rigid solid objects.
Technical report, Tech. Memo. 28, Production Automation Project, Univ. Rochester, 1977.

[Req80] A. A. G. Requicha.
Representations for rigids solids: Theory, methods and systems.
ACM Comput. Surveys, 12(4):437–465, December 1980.

[RT77] A. A. G. Requicha and R.B. Tilove.
Mathematical foundations of constructive solid geometry: General topology of regular closed sets.
Technical report, Tech. Memo. 27, Production Automation Project, Univ. Rochester, 1977.

[RV77] A. A. G. Requicha and H.B. Voelcker.
Constructive solid geometry.
Technical report, Tech. Memo. 25, Production Automation Project, Univ. Rochester, 1977.

[Sam84] H. Samet.

The quadtree and related hierarchical data structures.
ACM Computing Surveys, 16(2):187–260, 1984.

[Sam90] Hanam Samet.
Applications of spatial data structures : computer graphics, image processing, and GIS.
Addison Wesley, 2nd edition, 1990.

[SFF91] M.R. Stytz, G. Frieder, and O. Frieder.
Three-dimensional medical imaging: Algorithms and computer systems.
ACM Computing Surveys, 23(4):421–499, 1991.

[SFYC96] R. Shekhar, E. Fayyad, R. Yagel, and J. F. Cornhill.
Octree-based decimation of marching cubes surfaces.
In *Proceedings Visualization'96*, pages 335–344, 1996.

[SJ95] H. Shen and C.R. Johnson.
Sweeping simplices: A fast iso-surface extraction algorithm for unstructured grids.
In *Proceedings Visualization'95*, pages 143–150, Atlanta, Georgia, USA, October 29–November 3 1995.

[SK94] L.M. Sobierajski and A.E. Kaufman.
Volumetric ray tracing.
In *Proceedins Volume Visualization Symposium*, pages 11–18, 1994.

[SK99] M. Sramek and A. Kaufman.
Alias-free voxelization of geometric objects.
IEEE Transactions on Visualization and Computer Graphics, 5(3):251–267, 1999.

[SM98] S. Sethia and S. Manohar.
Minkowski operators for voxel based sculpting.
Computers and Graphics, 1998.

[SML96] W. Schroeder, K.M. Martin, and B. Lorensen.
The visualization toolkit: an object-oriented approach to 3D graphics.
Prentice-Hall, Inc, 1996.

[SML06] W. Schroeder, K. Martin, and B. Lorensen.
The visualization toolkit: an object-oriented approach to 3D graphics, 4th ed.
Kitware, Inc, 2006.

[SPS01] S. Schkolne, M. Pruett, and P. Schröder.
Surface drawing: creating organic 3d shapes with the hand and tangible tools.
In *Proceedings SIGCHI conference on human factors in computing systems*, pages 261–268, 2001.

[Sra01] M. Sramek.
High precision non-binary voxelization of geometric objects.
In *Spring Conference on Computer Graphics*, pages 220–229, 2001.

[ST85] H. Samet and M. Tamminen.
Bintrees, csg trees and time.
ACM Computer Graphics, 19(3):121–130, 1985.

[Sta05] William Stallings.
Sistemas Operativos. Aspectos internos y principios de diseño.
Pearson-Prentice Hall, 2005.

[SW88a] H. Samet and R.E. Webber.
Hierarchical data structures and algorithms for computer graphics. part i: Fundamentals.
IEEE Computer Graphics and Applications, 8(3):48–68, 1988.

[SW88b] H. Samet and R.E. Webber.
Hierarchical data structures and algorithms for computer graphics. part ii: Applications.
IEEE Computer Graphics and Applications, 8(4):59–75, 1988.

[SW05] S. Schaefer and J. Warren.
Dual marching cubes: Primal contouring of dual grids.
Computer Graphics Forum, 24(2):195–201, 2005.

[SZK95] R. Shu, C. Zhou, and M. S. Kankanhalli.
Adaptive marching cubes.
The Visual Computer, 11(4):202–217, 1995.

[The02] H. Theisel.
Exact isosurfaces for marching cubes.
Computer Graphics Forum, 21(1):19–31, 2002.

[TN87] W.C. Thibault and B.Ñaylor.
Set operations on polyhedra using binary space partitioning trees.
ACM Computer Graphics, 21(4):153–162, 1987.

[TPG99] G.M. Treece, R.W. Prager, and A.H. Gee.
Regularised marching tetrahedra: improved iso-surface extraction.
Computers and Graphics, 23(4):583–598, 1999.

[vGW94] A. van Gelder and J. Wilhelms.
Topological considerations in isosurface generation.
ACM Transactions on Graphics, 13(4):337–375, 1994.

[VKKM03] G. Varadhan, S. Krishnan, Y. Kim, and D. Manocha.
Feature-sensitive subdivision and iso-surface reconstruction.
In *Proceedings IEEE Visualization'03*, pages 99–106, 2003.

[VT01] F. Velasco and J.C. Torres.
Cells octree: A new data structure for volume modelling and visualisation.
In *6th International Fall Workshop on Vision, modeling and visualization 2001*, pages 151–158, Stuttgart, Germany, November 2001.

[VTC02] F. Velasco, J.C. Torres, and P. Cano.
Marching edges: A method for isosurface extraction.
In *Proceedings SIACG 2002*, pages 199–208, 2002.

[VTLS08] F. Velasco, J.C. Torres, A. León, and F. Soler.
Adaptive cube tessellation for topologically correct isosurfaces.
Journal of Virtual Reality and Broadcasting, 5(3), mar 2008.

[Wei08] Eric W. Weisstein.
Polyhedron.

Technical report, From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/Polyhedron.html, 2008.

[Wes90] L. Westover.
Footprint evaluation for volume rendering.
ACM Computer Graphics. (Proceedings SIGGRAPH'90), 24(4):367–376, 1990.

[WG94] J. Wilhelms and A.V. Gelder.
Multi-dimensional trees for controlled volume rendering and compression.
In *1994 Symposium on Volume Visualization*, pages 19–26. ACM SIGGRAPH, 1994.

[WK93] S. Wang and A. Kaufman.
Volume-sampled voxelization of geometric primitives.
In *Visualization'93*, pages 78–84, 1993.

[WK95] S.W. Wang and A.E. Kaufman.
Volume sculpting.
In *Proceedings Symposium on Interactive 3D Graphics*, pages 151–156, April 1995.

[WKE99] R. Westermann, L. Kobbelt, and T. Ertl.
Real-time exploration of regular volume data by adaptive reconstruction of isosurfaces.
The Visual Computer, 15(2):100–111, 1999.

[WMW86a] B. Wyvill, C. McPheeters, and G. Wyvill.
Animating *soft* objects.
The Visual Computer, 2(4):235–242, 1986.

[WMW86b] G. Wyvill, C. McPheeters, and B. Wyvill.
Data structure for *soft* objects.
The Visual Computer, 2(4):227–234, 1986.

[WvG90] J. Wilhelms and A. van Gelder.
Topological considerations in isosurface generation extended abstract.
ACM Computer Graphics, 24(5):79–86, 1990.

[WvG92] J. Wilhelms and A. van Gelder.
Octrees for faster isosurface generation.
ACM Transactions on Graphics, 11(3):201–227, 1992.

[ZBS03] Y. Zhang, C. Bajaj, and B.-S. Sohn.
Adaptive and quality 3d meshing from imaging data.
In *Proceedings ACM Symposium on Solid Modeling and Applications 2003*, pages 286–291, Seattle, Washington, USA, June 16–20 2003.

[ZCK97] Y. Zhou, B. Chen, and A. Kaufman.
Multiresolution tetrahedral framework for visualizing regular volume data.
In *Proceedings Visualization'97*, pages 135–142, 1997.

[ZCT95] Y. Zhou, W. Chen, and Z. Tang.
An elaborate ambiguity detection method for constructing isosurfaces within tetrahedral meshes.
Computers & Graphics, 19(3):355–364, July 1995.

[ZHK04] N. Zhang, W. Hong, and A. Kaufman.
Dual contouring with topology-preserving simplification using enhanced cell representation.
In *Proceedings IEEE Visualization'04*, pages 505–512, Austin, Texas, USA, October 10–15 2004.