PIECEWISE DESCRIPTIONS OF IMPLICIT SURFACES AND SOLIDS

Luiz Velho

A thesis submitted in conformity with the requirements
for the Degree of Doctor of Philosophy,
Graduate Department of Computer Science, in the
University of Toronto

©Copyright Luiz Velho 1994

Piecewise Descriptions of Implicit Surfaces and Solids

Ph.D. 1994

Luiz Velho

Graduate Department of Computer Science

University of Toronto

Abstract

This thesis presents a complete framework for the description of implicit surfaces and solids. We propose piecewise representations that are based on decomposition models and constructed using methods adapted to the geometry of objects. These models are general, in the sense that they can represent arbitrary shapes, and support variable precision, permitting approximations at any desirable resolution.

The contribution of our work consists of: (i) an original characterization of the implicit description of shapes, (ii) a new method for generating a smooth implicit function corresponding to a solid object, and (iii) two new piecewise representation schemes based on a multiscale decomposition of the implicit function and on an adapted simplicial decomposition of the domain of the implicit function.

We characterize the implicit model through an analysis of the implicit function. We show that the skeleton of a shape and the tubular neighborhood of its boundary are dual structures that relate an object with the space in which it is embedded.

We develop a method that allows the generation of smooth implicit functions from the characteristic function of an object. It employs multiresolution edge detection and reconstruction using dyadic wavelets.

We introduce a functional decomposition model based on B-spline scaling functions that generate nested multiscale approximating spaces. The Laplacian transform is employed to compute a pyramid in terms of these B-spline bases.

We introduce a spatial decomposition model based on adapted simplicial subdivision. Physics based deformation adapts to the boundary of the object a mesh derived from this simplicial complex.

Some of the applications of these methods include: approximate conversion between volumetric, implicit and parametric representations; surface rendering; volume visualization; and animation of implicit objects.

In summary, the relevance of this thesis is twofold: it provides a conceptual, as well as a practical scheme for piecewise shape description. The piecewise implicit representations that we have developed are effective and efficient. They capture the spatial features of objects using composite structures that are constructed from simple elements.

to Noni and Alice

Acknowledgments

I would like to thank my thesis supervisor, Demetri Terzopoulos, for his encouragement and support during this research. He is an outstanding researcher and a great advisor. Demetri was always there when I needed him. He acted as a true mentor pointing me in the right directions without restraining my creativity.

I wish to thank my external committee member, Professor Brian Wyvill of University of Calgary, for his interest in my research. I also thank the other members of my committee, Professors Ken Sevcik, James Stewart, Christina Christara and Alberto Leon-Garcia for their suggestions on the thesis content and form. A special word of recognition goes to Professor Eugene Fiume, who played an important role in my Ph.D. program giving me invaluable advice in many academic matters.

I acknowledge my fellow graduate students, Jos Stam, Dave Tonnesen, Michiel van de Panne, George Drettakis, Victor Ng-Thow-Hing, Michael McCool, Radek Grzeszczuk, Sherry Tu, Dimitris Metaxas, Tim McInerney and Hong Qin for their friendship. They are the heart and soul of the graphics and vision laboratories, two stimulating places where I spent most of my time while at the University of Toronto.

I am indebted to the following people for their inspiration and help: Lance Williams, who brought to my attention the beauty and potential of implicit objects; Stephane Mallat, who introduced me to wavelets; Ken Perlin, who shared with me his wisdom on geometric modeling, and my office mate and friend, Stephen Bellantoni, who gave his companionship and constant emotional support.

I express my gratitude to Professor Jonas de Miranda Gomes of IMPA. He encouraged me to pursue a Ph.D. in the first place and helped me with the mathematical aspects of this research. Jonas, more than a collaborator, became a truly good friend.

A final word of recognition goes to my parents, Luiz Carlos and Teresa Velho, for their affection and determination to give me a good education, and to my wife Noni Geiger and my daughter Alice, for their enduring love. They stood by me every step along the way, sharing the good and bad moments with patience and understanding.

A scholarship from CNPq – Conselho Nacional de Pesquisa e Desenvolvimento, provided financial support during my Ph.D. program.

Contents

Chapter 1

Introduction

The physical universe is populated by objects of astonishing diversity. These objects have one thing in common, a shape. Such concrete embodiment is a fundamental property of material objects which determines their function and allows us to identify them. For this reason, the study of shape is a central theme in many areas of science and technology, from mathematics to engineering. Each of these disciplines is concerned with a particular aspect of the shape of objects, such as its mathematical characterization, its design, its manufacturability, its recognition, its aesthetic content, etc.

This thesis focuses on the geometric representation of shapes and its applications to computer graphics. This chapter introduces the thesis topic, gives the motivation for the thesis research and summarizes its contributions. It presents the structure of the dissertation and relates it to the thesis results.

1.1 Background

This section overviews the main mathematical models used for shape description.

1.1.1 Shape Description

Shape description is a fundamental problem in computer graphics and related areas. In particular, graphics algorithms need a mathematical model of the geometry of objects in a scene. These computational methods require some form of discrete shape representation describing the properties of interest of objects. In three dimensions, one of the main interests is in the surfaces that bound solid objects.

The geometry of a surface can be specified in two ways: through a parametric function or through an implicit function. Although parametric surface models are still the dominant geometric representation in computer graphics, implicit surface models have been gaining importance in recent years.

1.1.2 Parametric versus Implicit

In the *parametric* description, the set of points constituting the object is given directly by a collection of mappings, called parameterizations. These mappings relate a space

of parametric coordinates to the object's surface such that there is a one to one correspondence between points in these two spaces. In the *implicit* description, the set of points belonging to the object is given indirectly through a point-membership classification function. This function defines the relationship of points in the ambient space with the object.

Intuitive understanding of the differences between the parametric and implicit forms is gained by means of a simple example:

Example 1.1 (Unit Circle) The unit circle can be described by the parametric equation $(x, y) = \mathbf{f}(\theta)$, where

$$\mathbf{f}(\theta) = (\cos\theta, \sin\theta), \quad \theta \in [0, 2\pi]$$

This parameterization maps the interval $[0, 2\pi]$ of the real line onto the unit circle S^1. It also allow us to directly enumerate points of S^1, by varying the parameter θ from 0 to 2π.

The unit circle can also be described by the implicit equation $f(x, y) = 0$, where

$$f(x, y) = x^2 + y^2 - 1, \quad x, y \in \mathbb{R}$$

The set of points (x, y) that satisfy $f(x, y) = 0$ constitute the circumference S^1. The function f classifies points on the 2D plane in relation to the unit disk. When we substitute the coordinates of a point $p = (x, y)$ in the equation $f(x, y) = x^2 + y^2 - 1$, the value of f indicates whether the point is inside, outside or on the circle as follows: the sign of f is negative in the interior region delimited by S^1, the sign of f is positive in the exterior region delimited by S^1, and f is zero if the point belongs to S^1. See Figure 1.1.

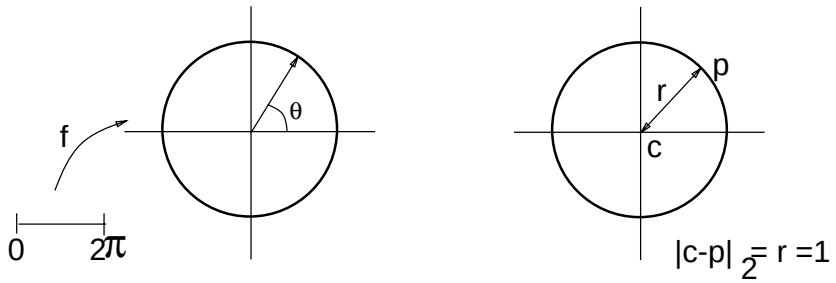


Figure 1.1: Unit circle

The parametric and implicit descriptions are in some sense complementary. One form is better suited to some types of operations than the other, and vice-versa. The

parametric form is suited to generating points on the surface. For example, to draw an approximation of the circle we connect by straight lines an ordered set of points lying on the surface. Using the parametric equation, the points are obtained simply by stepping the value of θ from 0 to 2π. The implicit form is suited to testing the containment of points relative to the surface. For example, to detect interferences between objects it is necessary to determine if points from one object cross the other object's boundary. Using the implicit equation, this is done simply by checking the sign of f.

1.1.3 Volumetric Models

Volumetric models are given by a function whose domain is an n-dimensional region of the ambient n-space. Usually the function is known only at discrete points and is represented by this set of sample values. Typical examples are n-dimensional arrays corresponding to regular sampling grids.

Volume data can be interpreted as samples of an implicit function that are generated by an external process, such as a fluid dynamics simulation, or a sensor, such as an MRI (Magnetic Resonance Imaging) device, as is the case in scientific visualization and medical imaging applications.

These applications deal with complex volumetric objects and the identification of certain features, such as isovalue surfaces, may be as important as the analysis of the function itself. As a consequence, it may be necessary to infer geometric models from volume data. Implicit models provide a powerful framework for this type of problems.

1.1.4 Relevance of Implicit Models

Implicitly defined surface models are important because they give information about the shape, not only at points on the surface, but also at points in the vicinity of the surface. As we have seen, the implicit function classifies points of the ambient space in relation to the volume enclosed by the surface. This explicit dependency of the geometric object with the space in which it is embedded can be exploited in most computational tasks related to modeling and rendering. In fact, this is perhaps the main advantage of implicit models over their parametric counterparts.

Implicit representations are necessary in computer graphics because they unify surface, volume and texture descriptions. The same modeling framework can be used to define surfaces, solids, fuzzy shapes, and spatial data. This is crucial in several application areas, from photorealism to scientific visualization.

1.2 Motivation

Implicit techniques provide the best solution to many problems in computer graphics. The potential of implicit models is largely unexplored. In particular, general

approaches to implicit shape description are lacking. The present research seeks to advance the state of the art of implicit representations in the direction of generality and unification, bridging the existing gap between continuous and discrete models. This section puts the thesis research into context with previous work, discusses its goals and the main strategies adopted.

1.2.1 Previous Work

Implicit surfaces have been used for a long time in computer graphics. Until recently, they were employed mostly as an auxiliary representation for classes of surfaces that allowed both parametric and implicit descriptions. Typical examples are the quadric and superquadric surfaces.

Exceptions to this rule are constructive solid geometry (CSG) models that have simple implicit primitives as their basic building blocks. These models are computationally efficient, but can represent a restricted class of shapes depending on the primitives.

More flexible constructive implicit models were developed as a generalization of algebraic surface models. The implicit function is a density field and the surface of interest is an isosurface. This density field may be constructed through a distance function from a point skeleton.

In Chapter 2 we review these models in more detail.

1.2.2 Piecewise Models

Piecewise models are effective because they are based on simple elements and powerful combination rules. Complex objects can be described by their components in a natural way. They can be simple and yet retain descriptive power.

There are two types of piecewise models: constructive and decomposition. A constructive model is generated by assembling primitive blocks. A decomposition model is generated by subdividing composite structures. Constructive techniques are most appropriate for shape design while decomposition techniques are better suited to shape computation.

Most of the existing piecewise implicit models are constructive. The traditional example is the already mentioned CSG model. Decomposition implicit models are very attractive, because they can provide a balance between generality and efficiency.

1.2.3 Goals and Strategy

This thesis investigates the implicit description of surfaces and solids. Our goal is to devise *effective* shape models that could be used in the development of computational methods for visualization and analysis of complex objects. These models should be *general*, allowing the description of arbitrary geometry. They should be *adapted* to

relevant spatial features and support *variable precision*, permitting approximation at any desired resolution.

The main strategy of this research is to explore *piecewise* representations of the implicit model. Two mechanisms play a key role in the process: *decomposition* and *adaptation*.

These models are obtained through processes that conform the model to geometric features of the object. As a result, we obtain implicit descriptions of surfaces and solids that fulfill the aforementioned goals.

1.2.4 Applications

The success of a representation depends not only on its ability to capture properties of objects and its suitability for computations, but also on the possibility of converting to and from other representations. This makes it more versatile, for example facilitating integration into existing systems. Consequently, this thesis emphasizes conversion methods between implicit, volumetric and parametric forms.

1.3 Contributions

We propose as a solution to the representation of implicitly defined objects the use of models that support effective representations of arbitrary shapes and that are computationally suitable for computer graphics applications. This section states the thesis results and their relevance.

The contributions of this thesis are:

1. A characterization of the implicit definition of surfaces and solids through an analysis of the implicit function.

2. A method to generate a smooth implicit function that approximates a solid object from its characteristic function.

3. An application of the smooth implicit function generation method to the conversion of parametric to implicit surfaces.

4. An investigation of approximate piecewise descriptions of implicit surfaces based on decomposition.

5. An implicit model based on a multiscale decomposition of the implicit function.

6. An application of the multiscale decomposition model to the conversion of volume arrays to piecewise implicit functions.

7. An implicit model based on adapted simplicial decomposition of the domain of the implicit function.

8. An application of the simplicial decomposition model to the conversion of implicit to parametric piecewise surfaces.

Some of these technical contributions have also been reported in (Velho and de M. Gomes, 1991b), (Velho and de M. Gomes, 1991a), (de Figueiredo et al., 1992), (Gomes and Velho, 1993) and (Velho and de M. Gomes, 1993).

Figure 1.2 depicts the relationship among the thesis results. The first column corresponds to theoretical contributions, the second column to computational methods, the third column to geometric representations, and the fourth column to applications.

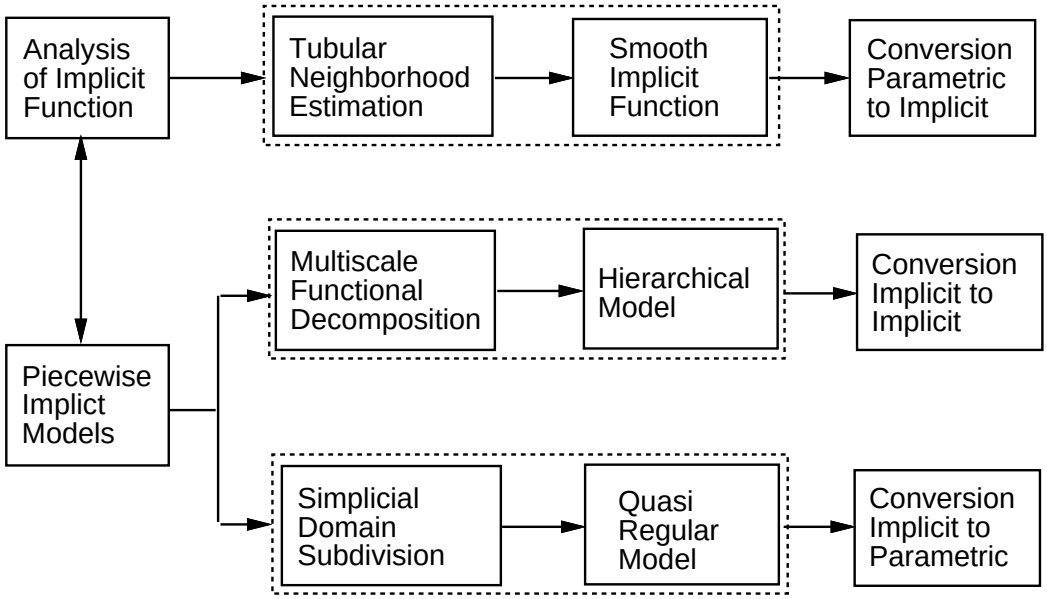


Figure 1.2: Graph with the relationships between the thesis results.

We now describe the contributions of the thesis in more detail.

1.3.1 Analysis of the Implicit Function

We characterize the implicit definition of surfaces and solids through an analysis of the implicit function f. New results include the identification of the skeleton of a shape and the tubular neighborhood of the shape boundary as two fundamental characteristics for implicit models of surfaces and solids. These are extrinsic properties that relate a manifold with the space in which it is embedded. We also show that the skeleton and the maximal tubular neighborhood are dual structures. This theoretical framework provides the required criteria for development and evaluation of the proposed models.

1.3.2 Generation of Smooth Implicit Functions

We develop a method to generate a smooth implicit function that approximates a solid object from its characteristic function. The method is based on multiresolution edge detection and reconstruction using dyadic wavelets. It synthesizes edges at multiple resolutions such that the reconstructed function preserves the shape boundary and the gradient of the function gives the normal field at the boundary. This procedure provides a rough estimate of a tubular neighborhood of the shape.

1.3.3 Conversion from Parametric to Implicit

The method to generate a smooth implicit function from the characteristic function of a shape can be applied to obtain an approximate but accurate conversion from parametric to implicit surfaces. The characteristic function of the region enclosed by the parametric surface is produced by a two step procedure: the surface is rasterized into a volume array and the interior region is filled. The method is then applied to this binary volume array creating a smooth implicit function in volumetric form.

1.3.4 Analysis of Piecewise Implicit Models

We classify implicit models according to their representation structures. The representation can be global or piecewise. Piecewise models are studied further from the point of view of their generation process. As we have mentioned before, we identify two classes of piecewise models: constructive and decomposition. Implicit decomposition models are investigated and the basic strategies used to compute them are discussed.

1.3.5 Multiscale Decomposition Model

We create a multiscale decomposition model for implicit surfaces and solids. The model is based on B-spline scaling functions that are Riesz bases of approximating spaces at multiple scales. The Laplacian transform is employed to generate the model. This model gives a structured representation consisting of a hierarchical stratification of the implicit function.

1.3.6 Conversion from Volumetric to Implicit

The procedure to generate the multiscale implicit model can be used for the conversion of volumetric to implicit models. It transforms an implicit function given in the form of a discrete array of samples into a continuous implicit description expressed as a linear combination of B-spline primitives. The structure of the model also gives a scale-space description of the data.

1.3.7 Simplicial Subdivision Model

We create a simplicial subdivision model [1] for implicit surfaces and solids. The model is generated by partitioning the domain of the implicit function into a simplicial complex and adapting it to the shape boundary. The adaptation method employs physics-based techniques. It uses a spring-mass mesh that deforms under forces derived from the gradient of the implicit function.

The above procedure produces quasi-regular space subdivisions that are subordinate to the implicit surface. The model can be adapted to various properties of the object by changing the deformation forces applied to the mesh.

1.3.8 Conversion from Implicit to Parametric

The procedure to generate the subdivision implicit model can be used for an approximate conversion of implicit to parametric surfaces. The polygonal mesh is derived from the simplicial subdivision structure. This polygonal mesh is the dual of the cell complex approximating the implicit function f.

1.4 Summary

In summary, a complete investigation of piecewise descriptions for implicit surfaces and solids is proposed. This includes: the analysis of the properties of such models, the specification of new representations together with computational methods to construct them and the development of applications related to the conversion between representations.

The claim that a general framework for piecewise implicit shape models has been formulated is supported by the fact that the methods developed in this thesis can be used to convert between the main forms of geometric descriptions, namely: volumetric, implicit and parametric.

This section presents the structure of the thesis and gives an outline of its contents.

1.4.1 Structure

The thesis is structured into four main parts.

The first part, Chapters 1 and 2, introduces the thesis topic, gives the motivation to the thesis research and compares the thesis results with related work.

The second part, Chapters 3 and 5, presents a mathematical analysis of the implicit model of surfaces and solids. The objective of this part is to supply the basic concepts that will be used throughout the thesis. We also define the characteristics of a suitable implicit function and a procedure to construct it.

[1] See Appendix D for a definition of simplicial subdivision

The third part, Chapters 4, 6 and 7, investigates piecewise implicit models and develops two decomposition models. This part discusses applications of the models and the procedures used to generate them.

The fourth part consists of four appendices which provide an overview of the mathematical tools used in the thesis. These appendices are included, in an attempt to make the dissertation self contained, mainly because some of these theoretical results may not be familiar to the graphics community.

1.4.2 Outline

Chapter 2 reviews related work in modeling and computation with implicit surfaces. Previous work in this area is compared with the thesis contributions. Constructive models are discussed and contrasted with the decomposition models developed in the thesis. The use of multiresolution representations in graphics, vision and image processing are reported and the multiscale representations created in the thesis are situated in this context. Polygonization methods of implicit surfaces are examined relative to one of the applications developed in the thesis.

Chapter 3 analyses implicit models of surfaces and solids, discussing the characteristics of the implicit function. The mathematical properties of the implicit model are studied. Skeleton and tubular neighborhoods are defined. Their importance to the characterization of implicit models is shown.

Chapter 4 investigates piecewise representation schemes for describing implicit surfaces and solids. Piecewise representations are divided in two classes: constructive and decomposition. Constructive implicit models are reviewed. Decomposition implicit models are described and the basic strategies to generate them are discussed.

Chapter 5 presents a method to generate a smooth implicit function corresponding to a given closed surface. This method is based on multiresolution edge detection. An algorithm for detection and reconstruction from multiscale edges is described. A method to generate a smooth function that approximates a solid from its characteristic function is developed. Applications of this method are discussed.

Chapter 6 develops a method to produce a piecewise implicit surface representation based on a multiscale decomposition of the implicit function. The theory of multiscale decomposition is presented. Methods based on wavelet analysis are introduced. A procedure to generate a multiscale implicit model is developed. Applications of this procedure are discussed.

Chapter 7 gives a method to construct a piecewise implicit surface representation based on a domain subdivision of the implicit function. The method employs physics based simulation. Deformation techniques using spring-mass meshes are presented. Space partition structures are introduced. A procedure to generate an adapted subdivision of the domain of the implicit function based on space partition and mesh deformation is developed. Applications of this procedure are discussed.

Chapter 8 summarizes the achievements of this research and discusses directions

of future work.

Appendix A is a simple introduction to functional analysis. It includes a review of abstract mathematical spaces, function spaces, linear operators and representation of functions.

Appendix B is a brief overview of wavelets and related topics. It includes a description of classes of wavelets, time-frequency analysis, multiresolution analysis and the wavelet transform.

Appendix C is a review of polynomial splines and a description of spline wavelets. It includes a discussion of spaces of polynomials, computational methods, spline wavelets, B-spline wavelets and time-frequency localization.

Appendix D is a study of space decompositions and spatial data structures. It includes a description of the main types of space subdivision, their properties and representation.

Note: About the Examples

Most of the examples in this thesis will be for objects in two dimensions. We will work with curves and areas in the plane instead of with surfaces and volumes in space. This is done for illustrative purposes and to facilitate the visualization of results. But, for each group of examples we will include at least one example in three dimensions.

Chapter 2

Related Work

This chapter discusses related work with implicitly defined objects and compares it to our research.

The breadth of the thesis topic makes it difficult to exhaustively compare the thesis results with prior work. The main motivation of our research is to investigate general models for implicitly defined shapes, as opposed to the more common approaches of concentrating on a particular class of models or improving an existing technique. To our knowledge, there is no previous comprehensive study of implicit models for arbitrary geometric objects. Most of the research in this area is restricted to specific models, such as implicit algebraic descriptions. For this reason, we believe that there is a need for a systematic investigation of general models.

Furthermore, the methods developed in this thesis are heavily influenced by image processing techniques and shape models used in vision. This is a consequence of the observation that one of the most general implicit representations is the volumetric description. In this form, the implicit function is usually defined by a three dimensional array of samples, or a 3D image. Therefore, it makes sense to extend the machinery developed for image manipulation and analysis to the context of implicit representation. Another advantage of this approach is that it establishes a direct connection between implicit models and volume data, the standard format in scientific applications. In this way, our methods are automatically applicable to scientific visualization and similar problems.

The central ideas in this thesis are that the development and evaluation of implicit models requires an abstract characterization of the implicit description, and that piecewise descriptions are the best choice of implicit models of general type. Furthermore, we argue that two effective ways to create piecewise implicit models are through a spatial decomposition and through a functional decomposition. In the first case, we decompose the domain of the implicit function and in the second, we decompose the implicit function itself.

In the following sections we will discuss these ideas in more detail, talk about related work and draw comparisons with the results of this thesis.

2.1 General Implicit Models

The discussion of a general implicit model includes a mathematical characterization of the implicit description of shapes, as well as the development of methods to generate the model given an object of arbitrary geometry.

2.1.1 Characterization of Shapes

There is a variety of shape models in vision. Among them, multiscale edges and medial axis models are particularly related to implicit descriptions. In the multiscale edge representation, a shape is characterized through the variations of the image intensity function at different resolution levels. The zero crossings of the Laplacian of a Gaussian filter can be used to describe the edges (Marr, 1982). In the medial axis representation, a shape is characterized by a lower dimensional skeleton. The structure of points that are equidistant from its boundary define the skeleton (Blum, 1967).

These two models have been used independently, partially because the connection with implicit descriptions was not made. In our characterization of implicit models, we show that the medial axis representation is related to the skeleton of an implicit solid and that the multiscale edge representation is related to the tubular neighborhood of the boundary of the solid. We also show that these two structures are complementary and that they define the embedding of an implicit object in the ambient space.

2.1.2 Construction of the Model

One of the problems in vision and image processing is the reconstruction of a continuous image function from a discrete set of sample values.This problem is related in many ways with general implicit models. If we are given a volumetric representation, it may be necessary to find a continuous interpolant [1] that describes it. Or, more generally, if we have the characteristic function of a shape, it may be desirable to generate a smooth implicit function defining it.

Although most of the traditional image reconstruction methods used in vision, such as (Terzopoulos, 1986) and (Szeliski, 1989), could be applied to the problem of constructing implicit functions, in this thesis we take a different approach. We propose the use of a method based on reconstruction from multiscale edges. The method employs a wavelet algorithm (Mallat and Zhong, 1992a) to generate a smooth implicit function. It essentially synthesizes a tubular neighborhood corresponding to the boundary of a shape. The benefits of this choice are that it establishes a direct relationship with the implicit model characterization and provides better control over the properties of the resulting function.

[1] Interpolation function

2.2 Piecewise Implicit Descriptions

In this thesis, we advocate that piecewise descriptions solve the representation problem for general implicit models. In addition, we assert that, in order to be useful, these descriptions should allow the conversion between representations.

An analogy with parametric models seems to confirm our arguments. Parametric descriptions are inherently piecewise. This is a consequence of the fact that it is not possible to define a single global parametrization of all but the simplest geometric objects. So, the solution is to use a set of local parametrizations satisfying certain continuity conditions. Some examples are NURBS and Bezier surface patch models.

Parametric descriptions are also easily convertible. In most cases it amounts only to a change of basis operation. As a matter of fact, this is almost transparent in many graphics systems. We are so used to such conversions that they are not noticed.

Let's take a simple case as an example: First specify a surface using a revolution operation, then modify the resulting geometry by moving points of its control mesh and finally render its image using a polygon shader. It may not be apparent, but this example required the use of three different parametric piecewise models and the conversion was done automatically by the system. The initial surface was defined as a surface of revolution, it was then converted to a bicubic mesh surface, and finally to a polygonal surface approximation.

In the case of implicit models, the situation is rather different. Piecewise descriptions are far less ubiquitous. There are fewer types of piecewise implicit representations and the conversion problem has not been satisfactorily solved. We claim that the results in this thesis contribute to improving the capabilities of existing implicit models and provide methods that will help to make the scenario described above a reality for implicit modeling systems.

Next, we review traditional piecewise implicit representation schemes and contrast them with the representation schemes developed in the thesis.

2.2.1 CSG

The CSG (Constructive Solid Geometry) representation scheme was the first piecewise implicit description used in computer graphics. It was introduced as a consistent model for solid objects (Requicha, 1980) and it was the basis of most of the early geometric modeling systems (Boyse and Gilchrist, 1982), (Brown, 1982) (Goldstein and Malin, 1979). The CSG model uses regularized boolean operations on point sets. In this scheme, a CSG object is build from the union, intersection and difference of primitive objects that are usually defined by implicit algebraic equations. A compound CSG object consists of a tree structure in which internal nodes are associated with set operations and leaf nodes with primitives.

Although the CSG scheme is independent of the primitive models (as long as they define a point set) and, therefore, also applies to parametric primitives (Mantyla,

1982), this scheme is best suited to implicit primitives and was initially developed in that context. That is because, implicit models naturally define the boundary, as well as, the interior of solids, an essential property for computation with CSG objects.

The CSG representation has several limitations:

- Its descriptive power depends on the set of primitives available in the system.
- It may be computationally inefficient.
- It is not unique.

These limitations are mostly a consequence of the constructive nature of the CSG description. This makes it suitable for shape design, but since the composite structure is defined solely by the user, it may not be optimal from the computational standpoint. In contrast, the decomposition models proposed in this thesis are adapted to the geometric properties of objects and are computationally efficient.

2.2.2 Skeletons

Primitive implicit solids can be defined by a distance function from lower dimensional geometric elements, such as a point or a curve. As we mentioned, these elements are called *skeletons*. The implicit function is expressed as $f(x, y, z) - c$, where f is some pseudo-distance function and the boundary of the primitive is a level surface c units away from the skeleton in this pseudo-metric. The above scheme has connections with implicit blending operations. This mechanism raises the possibility of representing implicit objects using blend composition of skeleton based primitives [2]. Normally, for simplicity, the composite implicit function is the sum of contributions of each primitive (L^1 metric).

Blinn was the first to introduce skeleton based implicit models in computer graphics (Blinn, 1982). His model was inspired by electron density maps. It uses a point skeleton and the distance function is a Gaussian centered at each point of the skeleton. Blinn's idea was further developed by (Nishimura et al., 1985) and (Wyvill, McPheeters and Wyvill, 1986a). One important aspect of these last two models is that they use functions that drop to zero at a certain distance from the skeleton. This has a practical significance in computational efficiency.

The first skeleton models consisted of point skeletons with simple blending functions. This made them very attractive computationally. In an attempt to develop better mechanisms for shape design, the complexity of the skeleton models was increased making them less efficient computationally.

The multiscale decomposition model that we develop can be considered a point skeleton model that is structured and hierarchical. These two characteristics make it computationally efficient. The decomposition mechanism makes it expressive.

[2] Note that the blending operation does not create a skeleton of the composite shape.

2.2.3 Volumetric Models

Volumetric models are usually represented by 3-dimensional voxel arrays. Such a description can be interpreted as an implicit model in sampled form. In this model, array values are samples of an arbitrary implicit function at regular grid points.

The data may be generated by external processes or by sensors. This type of representation is gaining importance in many application areas of computer graphics, particularly in scientific and medical applications (McCormick, 1987).

Volume data is a redundant representation, because the array values are highly correlated in general. Furthermore, appropriate resolutions demand large memory and computation resources.

The wavelet decomposition methods employed to create our multiscale models eliminate the redundancy from the volumetric input data and give a description that is piecewise analytical.

2.2.4 Comparison

Although point skeletons and volumetric models are very different in many respects, they have some essential characteristics in common. Both models define the implicit function in terms of values at discrete sets of points and use similar blending functions to combine them. However, point skeletons are irregular and sparse representations, while volumetric models are regular and dense. Point skeletons are more compact and computationally efficient, while voxel arrays are inefficient in terms of space and processing.

The multiscale implicit models proposed in this thesis combine some of the best features of point skeletons and volumetric models.

2.3 Functional Decomposition Models

One way to create a piecewise description of an implicitly defined object is to decompose the implicit function into a collection of simpler functions. A particularly effective scheme for this purpose consists in selecting basis functions that are localized in both space and frequency.

2.3.1 Multiscale Representations

The multiscale representation is a description of a function in terms its components at different scales. Recently, research on wavelet theory contributed to the unification of results from many fields and established a rigorous mathematical foundation for multiscale representations (Daubechies, 1992), (Chui, 1992). The intense interest in wavelets also stimulated the development of applications, particularly in image processing (Mallat, 1989).

A wavelet based implicit model was used by (Muraki, 1993) to represent volumetric data sets. His model resorts to orthogonal spline wavelets in 3 dimensions. Muraki's model has two major deficiencies. First, it uses tensor product wavelets which, in three dimensions, require seven different wavelet functions. These basis elements give directional information that is very useful for analysis purposes, but make synthesis computations more complex. Second, it uses basis functions that are not compactly supported. Therefore, the wavelet transform has to be computed approximately, introducing inaccuracy into the model. Our multiscale model is based on a wavelet decomposition of the implicit function, but it employs the scaling function associated with the wavelet and, for this reason, is more adequate for synthesis purposes than models that use the wavelet functions directly.

Another implicit surface model inspired on the wavelet decomposition was proposed by (Perlin and Zhu, 1990). The model, called "surflet", uses a single wavelet-like function that is oriented. This description is constructed by an empirical procedure from samples on a uniform grid. One limitation of the model is that it approximates only one level surface given by the implicit function. Our model describes the implicit function as a whole, in contrast with the surflet model which describes only the level surface.

2.3.2 Multiresolution Representations

The multiresolution representation is a hierarchical description which incorporates versions of the same function corresponding to various scales. It employs a pyramidal data structure that stores this sequence from fine to coarse resolutions.

Multiresolution representations have been used in image processing and computer graphics for a long time. In image processing, it was applied to various computational tasks (Rosenfeld, 1984), including edge detection (Canny, 1986), image compression and transmission (Burt, 1983), and variational solution of visual inverse problems (Terzopoulos, 1984). In computer graphics, it was primarily applied to pre-integration of textures to accelerate anti-aliasing calculations (Williams, 1983). The multiresolution analysis is intrinsically connected with the wavelet decomposition. It provides a structure that makes it possible to construct a class of orthogonal and biorthogonal wavelet functions. Conversely, a wavelet decomposition can be expressed in terms of the scaling function that is a basis of the multiresolution approximation spaces.

In this thesis, we exploit results from multiresolution analysis. We use multiresolution edge estimation to compute a tubular neighborhood of a surface and to generate a smooth implicit function corresponding to this surface. We use the multiresolution analysis to define the biorthogonal B-spline basis of our multiscale implicit model and employ the Laplacian transform to create its representation.

2.4 Physics-Based Spatial Decomposition Models

Another way to create a piecewise description of an implicitly defined object is to decompose the domain of the implicit function into subdomains and approximate it by simpler functions in each subdomain. The standard domain decomposition procedures do not give, in general, a space partition that is adapted to the shape of the object, but a physics based deformation can be used for this purpose.

2.4.1 Spatial Decompositions

The decomposition of the domain of the implicit function provides a piecewise description that serves as a basis for the sampling and structuring operations required in most computations with implicit objects.

A particularly important example is the polygonization of the implicit surface. In this case, the spatial decomposition is used to localize the surface and to generate the corresponding polygon mesh. Therefore, a natural classification of polygonization methods is according to the type of decomposition they use. The main types are *non-simplicial* and *simplicial* methods.

Most non-simplicial methods employ a rectangular tessellation of space. The marching cubes algorithm (Lorensen and Cline, 1987), is the most popular method of this type. Similar algorithms have been developed independently by (Wyvill, McPheeters and Wyvill, 1986b) and (Pasko and Pilyugin, 1988). These methods are fast and simple to implement. The main drawback is that they cannot be used to represent the implicit function unambiguously. Therefore, they require a disambiguating scheme to produce a consistent polygonization (Beier, 1990).

Simplicial methods triangulate the domain of the implicit function. They are theoretically sound, because they rely on classical results from algebraic topology related to the piecewise linear structure of a simplicial complex. Computational work on this type of method was pioneered by Allgower (Allgower and Schmidt, 1985) (Allgower and Gnutzmann, 1987).

Another important characteristic of polygonization methods is whether or not the space decomposition is adapted to the implicit function. Adaptive methods partition space unevenly, such that the decomposition is finer where necessary. Most methods start with an initial space tessellation, and refine it recursively until some adaptation criteria are met. Because these methods subdivide cells adaptively, they must constrain the subdivision process in order to guarantee the compatibility of the cell complex. Adaptive algorithms based on restricted trees have been proposed for simplicial (Hall and Warren, 1990) as well as non-simplicial (Bloomenthal, 1988) decompositions. A more efficient method based on restricting polygon edges was proposed in (Velho, 1990).

In this thesis, we introduce an adaptive method that employs physics-based techniques, rather than recursive subdivision, to generate a simplicial decomposition of

the implicit function domain. This method produces a polygonization that is quasi-regular, in the sense that all polygons are almost equilateral and the transition between different size polygons is smooth. Note that we generate not only a polygonization, but also a piecewise linear description of the implicit function.

2.4.2 Physics Based Methods

Physics based methods have been used in vision and graphics for shape estimation, modeling and animation (Terzopoulos, Witkin and Kass, 1987) (Terzopoulos et al., 1987), (Terzopoulos and Fleischer, 1988).

Discrete physical models constitute a simple and yet powerful mechanism for dynamics simulation (Greenspan, 1973). A spring-mass mesh is a discrete physical model defined by point masses that are interconnected by springs.

We employ physics based deformation to adapt our spatial decomposition model to the boundary of the implicit object. This is done by associating the cell structure describing the space decomposition with a spring-mass mesh. The mesh is then submitted to forces derived from the implicit function. A similar approach was used in image processing for the purposes of adaptive sampling and compression (Terzopoulos and Vasilescu, 1991). The main difference between these methods is that we employ two complementary force fields related to a tubular neighborhood of the implicit surface, while they employ only one force field for the entire image.

Chapter 3

Implicit Models

This chapter studies the description of surfaces and solids in implicit form. It analyses the main characteristics of the implicit function and establishes criteria to define the optimality of this function.

We try to answer the following questions: What is the mathematical characterization of a shape using implicitly defined models? Is this unique? If not, what makes a good implicit model for geometric computation? These questions are very important, especially if our goal is to develop new geometric representations for computer graphics applications. The problems they raise have to be addressed at an abstract level if the intent is a general description of arbitrary shapes.

3.1 Implicit Object Model

An implicit object is described by a pair (f, c), where f is a C^k real function of Euclidean space and c is a real constant.

3.1.1 Definition of Surfaces

A surface $\mathcal{S} \subset \mathbb{R}^n$ has an implicit description if there is a function $f : U \to \mathbb{R}$, $\mathcal{S} \subset U$, and a value $c \in \mathbb{R}$, such that $\mathcal{S} = f^{-1}(c)$. That is,

$$\mathcal{S} = \{p \in U : f(p) = c\}.$$

The function f is called an *implicit function*, U is the domain of f, and c is called the *level* of $\mathcal{S}$.

The implicit surface is the set of points which satisfy the implicit equation $f(p) - c = 0$. Note that this is a root finding problem with infinitely many solutions.

3.1.2 Definition of Solids

If an implicit surface $\mathcal{S} = f^{-1}(c)$ is a closed, codimension one manifold, it separates the space into two connected components having $\mathcal{S}$ as a common boundary. In this case f can also be used to describe an implicit solid.

A solid object $\mathcal{O}$ (with boundary) is the inverse image $f^{-1}(I)$ of the interval $I = (-\infty, c]$ (or $[c, \infty)$, depending on the orientation convention).

$$\mathcal{S} = \{p \in U : f(p) \leq c\}$$

where U and f are defined as in the previous subsection. The implicit solid is the set of points satisfying the inequality $f(p) - c \leq 0$.

3.1.3 Regular Surfaces

The implicit definition is broad enough to include a very large family of surfaces. In fact, this definition is too broad and allows degenerate types of surfaces. Hence, it is necessary to impose restrictions on f and c, so that a sound computational framework can be developed.

An implicit surface is called *regular* if c is a regular value of f. This is true if for all $p \in f^{-1}(c)$ the differential $f'(p)$ is surjective, otherwise c is called a critical value of f. The regularity condition means that the gradient vector

$$\nabla f(p) = \left(\frac{\partial f}{x_i}(p), \ldots, \frac{\partial f}{x_n}(p)\right)$$

does not vanish at the points $p \in \mathcal{S}$.

Any orientable smooth surface may be defined as the inverse image of a regular value c of a function $f : U \to \mathbb{R}$, where U is an open subset of $\mathbb{R}^n$. Moreover, Sard's theorem says that regularity is a stable condition, and a regular value c can be chosen with probability one. Mathematically, this means that the set of regular values is open and dense (Milnor, 1965). We should remark, however, that it is in general very difficult to decide about regularity from a computational point of view. The solution is to design robust numerical methods exploiting Sard's results.

A regular implicit surface $f^{-1}(c)$ is an orientable submanifold of $\mathbb{R}^n$. This implies that $f^{-1}(c)$ is a non-degenerate level surface of f. Moreover, if c is perturbed, we get a foliation of a neighborhood of $f^{-1}(c)$ by regular implicit surfaces, as shown in Figure 3.1.

3.1.4 Geometric Interpretation

Implicitly defined surfaces can be interpreted geometrically as the level set of the graph of a function. This gives a visual intuition of the meaning behind the implicit form.

Consider a function $f : \mathbb{R}^n \to \mathbb{R}$. Its graph is a subset of $\mathbb{R}^{n+1}$ defined by

$$g = \text{graph}(f) = \{(x_1, \ldots, x_{n+1}) : (x_1, \ldots, x_n) \in U \text{ and } x_{n+1} = f(x_1, \ldots, x_n)\}$$

This may be visualized as the height field where, for each point $p \in U$, the value of $f(p)$ gives the elevation of the hypersurface g at that point. The surface $\mathcal{S}$ defined

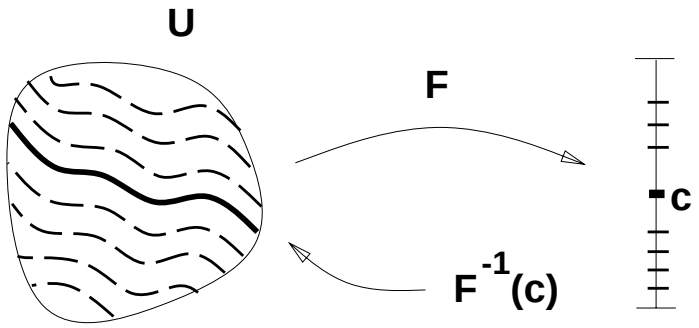


Figure 3.1: Level Contours

by $f^{-1}(c)$ is the intersection of the graph of f with a hyperplane parallel to U at a distance c from it, $f^{-1}(c) = g \cap \{x_{n+1} = c\}$. This is shown in Figure 3.2 for a circle defined implicitly by $x_1^2 + x_2^2 = c$. We use a two-dimensional example for clarity. The graph of f is a paraboloid of equation $x_3 = x_1^2 + x_2^2$, and its intersection with a plane $x_3 = c$ is a circle for $c > 0$.

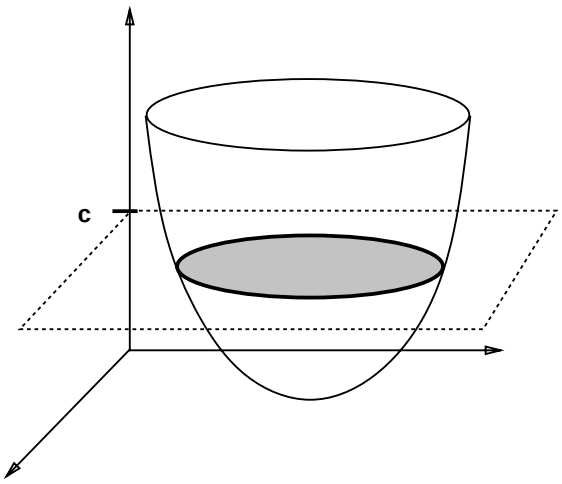


Figure 3.2: Implicit surface as a level set

The regularity condition in the geometric interpretation above is reflected in the fact the hypersurface g and the hyperplane $x_{n+1} = c$ have no tangency along their intersection set. In Figure 3.2, note that $c = 0$ is not a regular value of f because it is a minimum of f and therefore a singular point.

3.2 The Implicit Form

The function f is the main element in the mathematical characterization of implicit surfaces and solids.

3.2.1 Point-Membership Function

The implicit function is a *point-membership* classification function f that returns a value according to the relationship of a point $p \in U$, given as its argument, with the implicit surface $\mathcal{S}$ defined by it:

$$f(p) \begin{cases} > c & p \in \text{positive side of } \mathcal{S}. \\ = c & p \in \mathcal{S}. \\ < c & p \in \text{negative side of } \mathcal{S}. \end{cases}$$

3.2.2 Space Subdivision

The implicit function induces a subdivision of the ambient space into connected components that, depending on the sign of $f(p)-c$, correspond to the interior and exterior regions of the implicit solid $\mathcal{O}$ (Note that interior and exterior are defined by a sign convention which is arbitrary). Figure 3.3 shows an example of a solid object with two connected components, one bounded and the other unbounded.

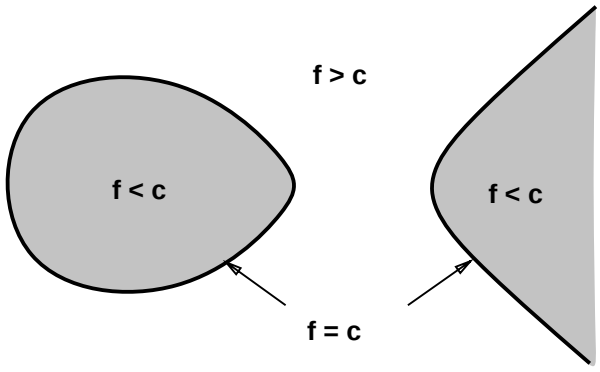


Figure 3.3: Implicit solid

3.2.3 Characteristic Function

The characteristic function $\chi_{\mathcal{O}} : \mathbb{R}^n \to \{0, 1\}$ of a solid object $\mathcal{O}$, defined as $\chi_{\mathcal{O}}(p) = 1$ if $p \in \mathcal{O}$ and $\chi_{\mathcal{O}}(p) = 0$ if $p \notin \mathcal{O}$, can be trivially derived from the implicit equation:

$$\chi_{\mathcal{O}}(p) = \begin{cases} 1 & \text{if } f(p) \leq c \\ 0 & \text{otherwise} \end{cases}$$

3.2.4 The Gradient of the Implicit Function

Given a differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, the gradient vector field ∇f is defined in its domain U. This vector field is orthogonal to each level surface $f^{-1}(a)$, $a \in \mathbb{R}$, associated with the implicit function f. It points in the direction of largest growth of f and indicates, at any point of U, a path to the surface $\mathcal{S} = f^{-1}(c)$.

As pointed out earlier, a regular implicit surface is always orientable. This means that a globally consistent direction of rotation in the tangent planes of the surface can be chosen. The orientation is given by the gradient vector field.

Furthermore, the vector field

$$N(p) = \frac{\nabla f(p)}{\|\nabla f(p)\|}, \quad p \in \mathcal{S}$$

defines the unit normal at a point p of the surface.

3.3 Understanding the Implicit Function

An important aspect of the implicit surface description is the nature of the implicit function f. The understanding of the properties of f gives an indication of the operations that can be performed with the model, as well as their effect on geometry.

3.3.1 The Implicit Function and Metrics

The implicit function can be interpreted as a measure of the proximity from the surface to a point in space. In this sense, $f : U \subset \mathbb{R}^n \to \mathbb{R}$ gives the signed distance, induced by some pseudo-metric $d : \mathbb{R}^n \to \mathbb{R}$, of points $p \in U$ to the level surface $\mathcal{S} = f^{-1}(c)$.

Example 3.1 (Circle) The circle with center o and radius r can be defined as the inverse image $f^{-1}(0)$ of the implicit function $f(p) = d(o, p) - r$, where d is the Euclidean metric.

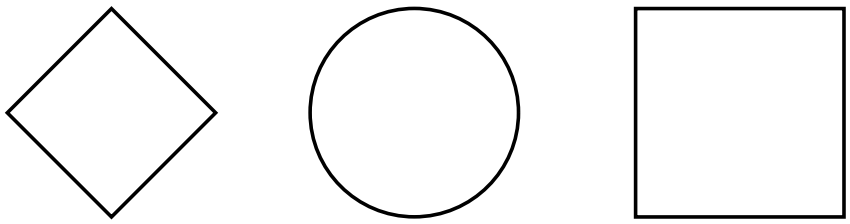
In a finite dimensional vector space all distance functions derived from a norm are topologically equivalent. Particularly, in $\mathbb{R}^n$ we have the following classical metrics

$$d_1(p,q) = |p_1 - q_1| + \cdots + |p_n - q_n|$$

$$d_2(p,q) = \sqrt{|p_1 - q_1|^2 + \cdots + |p_n - q_n|^2}$$

$$d_\infty(p,q) = \max(|p_1 - q_1|, \ldots, |p_n, q_n|)$$

Note that, although these metrics are equivalent from a topological standpoint, they produce different geometric results when used to define an implicit surface. Figure 3.4 shows the circle of Example 3.1 using the metrics above.



d_1 d_2 d_∞

Figure 3.4: A ball in different metrics

The metrics d_1, d_2 and d_∞ can be generalized to form the m-norms, a one parameter family of distance functions:

$$d_m(p,q) = \left[\sum_i \| p_i, q_i \|^m\right]^{1/m}.$$

The fact that altering the metrics causes shape changes may be exploited by the implicit surface model if this is incorporated into the implicit function.

Example 3.2 (Superellipse) The superellipse model allows the control of shape through a modification of the metric. The function $f(x)$ is defined as

$$\left(\left(\frac{x_1}{a}\right)^e + \left(\frac{x_2}{b}\right)^e\right)^{\frac{1}{e}},$$

where e controls the roundness of the shape and a and b are respectively horizontal and vertical scaling factors.

3.3.2 Changing the Implicit Function

Algebraic and arithmetic operations performed on the function f change some characteristic of the implicit function that may or may not alter the surface $\mathcal{S} = f^{-1}(c)$. There are operations that leave $\mathcal{S}$ invariant and others that modify its geometry or topology.

3.3.2.1 Operations that Leave the Surface Invariant

The following operations, depending on the value of c, change the function f without affecting the surface $f^{-1}(c)$.

If a surface $\mathcal{S}$ is defined as the level set $f^{-1}(0)$, then it is also implicitly defined by any function g:

$$g(x) = \alpha f(x),$$

where $\alpha \in \mathbb{R}$, $\alpha \neq 0$.

If $f : U \to (0, \infty)$ and $\mathcal{S}$ is defined as $f^{-1}(1)$, then it is also implicitly defined by any function g:

$$g(x) = f(x)^{\beta},$$

where β is a positive real number.

In these two cases, the surface $\mathcal{S}$ is said to be embedded in both f and g. Note that although these implicit functions describe the same surface, they are not induced by the same pseudo-metric.

3.3.2.2 Operations that Change the Surface

A level surface $\mathcal{S}$ of the graph of f is called an *isosurface* or isovalue surface (in general, we use the name *isocontour* to designate the level sets of an arbitrary implicit function $f : \mathbb{R}^n \to \mathbb{R}^m$). The contour surfaces of f can be traversed by a function h defined as:

$$h(x) = f(x) - \delta,$$

where $\delta \in \mathbb{R}$. The surface $h^{-1}(c)$ is an *offset surface* of $f^{-1}(c)$ in the pseudo-metric induced on h and f.

3.3.3 Canonical Forms

There are two canonical formats for the implicit description of a surface. In the previous subsection, we have seen that the following pairs (R, v) of range and value

$$((-\infty, +\infty), 0)$$
$$((0, +\infty), 1)$$

have special properties in relation to the invariance of $f^{-1}(v)$ for $\text{Range}(f) \in R$.

There exists a simple map that can be used to convert between these two formats as required. Given an arbitrary function $f : U \to (-\infty, \infty)$ the transformation:

$$h(x) = \exp(f(x)),$$

maps the range $(-\infty, \infty)$ into $(0, \infty)$ and maps the level surface $f^{-1}(0)$ into $h^{-1}(1)$.

This transformation has an inverse, defined by taking the logarithm

$$f(x) = \log(h(x)).$$

3.4 Tubular Neighborhoods

The concept of tubular neighborhood is of fundamental importance in the study of differentiable manifolds, because it relates a surface with its normal vector field. Thus, by investigating the tubular neighborhood of a surface $\mathcal{S}$, it is possible to make a connection between the isocontour $\mathcal{S} = f^{-1}(c)$ and the associated implicit function f. This is done indirectly through the vector field normal to $f^{-1}(c)$, given by the gradient of f.

3.4.1 Definitions

A *normal segment* $[p, b]$ to a surface $\mathcal{S}$ at the point p is a line segment from p to b such that $p \in \mathcal{S}$ and $[p, b]$ is contained in $T_p^{\perp}\mathcal{S}$, the orthogonal complement of the tangent space of $\mathcal{S}$ at p. The point p is called the *foot point* of b in $\mathcal{S}$. The 1-dimensional set of all segments normal to $\mathcal{S}$ at the point p with length less than ϵ, is denoted as $B^{\perp}(p, \epsilon)$.

An *admissible normal radius* for a subset $Z \subset \mathcal{S}$ is a real number $\epsilon > 0$ such that any two normal segments, $[p, x]$ and $[q, y]$, with $p \neq q \in Z$ and length $< \epsilon$ do not intersect. See Figure 3.5.

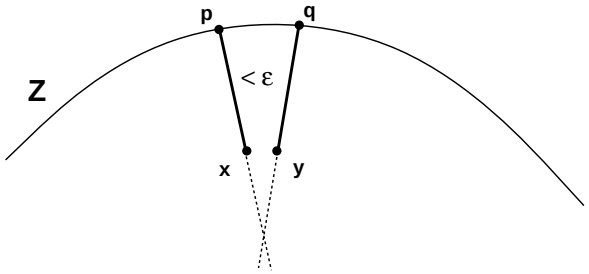


Figure 3.5: Admissible Normal Radius

An *ϵ-tubular neighborhood* V_ϵ of a surface $\mathcal{S}$ is defined as the union of all segments normal to $\mathcal{S}$ with radius $\epsilon(p)$ such that ϵ is an admissible normal radius for $\mathcal{S}$ at p. That is

$$V_\epsilon(\mathcal{S}) = \bigcup_{p \in \mathcal{S}} B^{\perp}(p; \epsilon(p)).$$

See Figure 3.6.

It is possible to prove that any regular implicit surface $\mathcal{S} = f^{-1}(c)$ has a tubular neighborhood. This is a consequence of the fact that the gradient vector field of f does not vanish on $\mathcal{S}$.

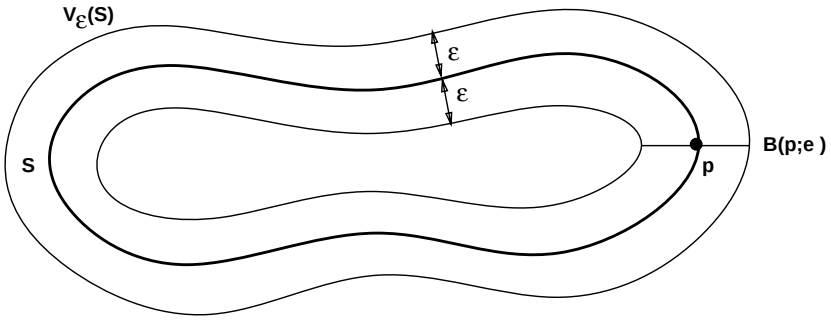


Figure 3.6: Tubular Neighborhood

3.4.2 The Projection on the Surface

The existence of a tubular neighborhood makes possible to define a projection function $\pi : V_\epsilon(\mathcal{S}) \to \mathcal{S}$ which, for each point $x \in V_\epsilon$, associates the unique foot point p of the normal segment that contains x.

This projection is a very powerful mathematical instrument that can be used for many purposes in the study of surfaces. In particular, it implies that a tubular neighborhood is equivalent to the product space $\mathcal{S} \times B(\epsilon)$, where $B(\epsilon)$ is an open interval $\in \mathbb{R}$ with center at the origin and radius ϵ. This corresponds to a topological open cylinder. See Figure 3.7.

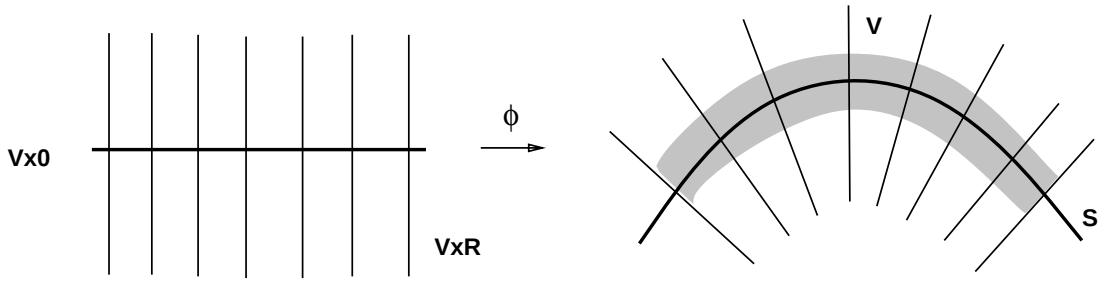


Figure 3.7: Product Space

3.4.3 The Maximal Tubular Neighborhood

A tubular neighborhood of a surface is called *maximal* when it contains all possible ϵ-tubular neighborhoods of that surface.

Intuitively the maximal tubular neighborhood extends the normal fibers of V_ϵ as far as possible without violating the projection conditions above.

The maximal tubular neighborhood $V_{\max}$ of a surface $\mathcal{S} = f^{-1}(c)$ is *unique*. $V_{\max}$ gives the largest open set $\mathcal{S} \subset U \subset \mathbb{R}^n$ where a continuously differentiable distance function $f : U \to \mathbb{R}$, associated with the surface $f^{-1}(c)$, can be defined. In other words, U is the maximal domain in which an implicit function f can be constructed such that f does not have singular points in U (or, equivalently, ∇f does not vanish in U).

Example 3.3 (Unit Circle) The maximal tubular neighborhood of the unit circle is the entire plane minus the origin ($\mathbb{R}^2 - \{(0,0)\}$).

3.5 Skeletons

The skeleton describes the structural essence of a shape. It provides a means to characterize the topology of solids and to construct a geometric model of their boundary. In particular, as suggested earlier, the skeleton may serve as the basis for implicit surface models if it is associated with a suitable distance function.

The skeleton has also been extensively used in computer vision for shape recognition and classification purposes (Blum, 1967), (Hoffmann, 1992), (Nackman and Pizer, 1985).

3.5.1 Definitions

The *distance* from a point p to a surface $\mathcal{S}$ in $\mathbb{R}^n$ is the minimum of the Euclidean distance $d_E(p, s)$, where s is on $\mathcal{S}$.

$$d(p, \mathcal{S}) = \inf_{s \in S} d_E(p, s).$$

Since d is continuous, if $\mathcal{S}$ is compact, for every $p \in \mathbb{R}^n$, there is at least one point $s_0 \in \mathcal{S}$ such that $d(p, \mathcal{S}) = d(p, s_0)$. Such point s_0 is called the *foot point* of p in $\mathcal{S}$ (Note that this definition is equivalent to the one in Section 3.4).

The *skeleton* of a region of $\mathbb{R}^n$ bounded by a surface $\mathcal{S}$ is the closure of the set of points $p \in \mathbb{R}^n$ such that $p \notin \mathcal{S}$ and p has more than one foot point on $\mathcal{S}$.

The *interior skeleton* consists of all skeleton points that are interior relative to $\mathcal{S}$. Similarly, the *exterior skeleton* consists of the set skeleton points that are exterior relative to $\mathcal{S}$.

3.5.2 Intuition

Intuitively, the skeleton is formed by all the points of the ambient space that have more than one geodesic path to the surface $\mathcal{S}$.

An alternative definition of the skeleton with a geometric flavor employs the notion of maximal spheres. A sphere is called *maximal* with respect to a region R, if it is contained entirely in R, but it is not properly contained by any other sphere in R. See Figure 3.8.

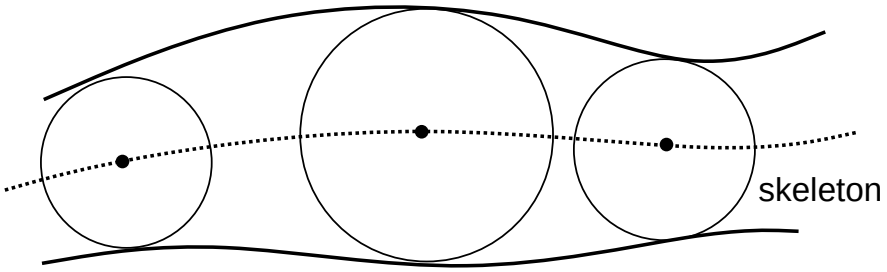


Figure 3.8: Maximal Sphere

The skeleton can be defined as the locus of the centers of all spheres that are maximal with respect to the interior and exterior regions delimited by $\mathcal{S}$.

The definition above was proposed in connection with the *medial axis* transform used in vision (Blum, 1967).

3.5.3 Characteristics

The skeleton of a region of dimension d is formed by the union of elements of dimension $d-1$ or lower.

A solid has a unique skeleton. Figure 3.9 shows the skeleton of a rectangular solid region of $\mathbb{R}^2$.

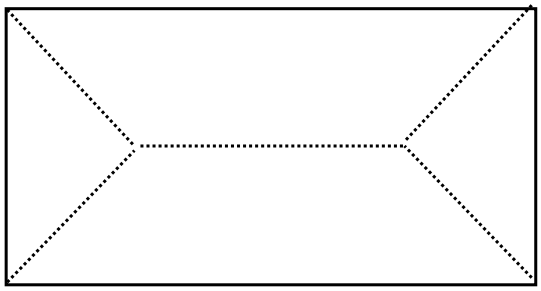


Figure 3.9: Skeleton

3.6 Surfaces in Space

In order to analyze the effectiveness of implicit surface models it is necessary to study the extrinsic properties of the surface. The reason this is so important lies in the fact that implicit surfaces are defined by a function of space. The investigation of how the surface is embedded provides the required criteria to characterize the implicit function.

3.6.1 Surfaces and the Structure of Space

The maximal tubular neighborhood of a surface $\mathcal{S}$ and the skeleton of the region enclosed by $\mathcal{S}$ provide a structure of the ambient space that reveals essential aspects of the implicit surface model.

These two geometric entities are dual structures. More than that, one is the complement of the other in $\mathbb{R}^n$. This is clear from their very definition.

The concepts of tubular neighborhood and skeleton are totally independent of the implicit description. But, in some sense, they capture all properties of a surface which depend on its embedding in the ambient space. This is precisely the reason why they are important: they relate a surface with the space in which it lives.

For implicit regular surfaces:

- The skeleton is contained in the set of singular points.
- The tubular neighborhood is contained in the complement of the set of singular points.

This is a precise statement about the duality of these structures.

These instruments are useful to analyze and construct implicit surface models. Tubular neighborhoods provide a way to investigate the domain of the implicit function; skeletons provide simpler geometric objects from which the implicit function can be defined.

3.6.2 What is a Good Implicit Function?

There are many properties that a good implicit surface model should have. Some of them, such as simplicity, conciseness or completeness, are of qualitative nature and apply to any type of geometric model. Others, are specific to the implicit model.

In simple terms, specific properties of the implicit surface model are related to:

- the information conveyed by the value of f;
- the extent of the domain of f with valid information.

According to these criteria, a good implicit function should provide the desired information over a prescribed region of space.

One possible criterion to specify such a function is faithfulness to a metric of the ambient space. Under this assumption, the optimal implicit function is the Euclidean distance from the surface. This is a linear function that is singular at the skeleton points. Such a function will be briefly described in the appendix to this chapter.

Another optimality criterion is smoothness. In that case, it is not possible to employ a true metric. So, the model has to resort to a pseudo-metric. Also, some control over the domain of the implicit function may be required. The most natural choices are, either the entire $\mathbb{R}^n$, or a prescribed ϵ-tubular neighborhood of the surface. In Chapter 5 we will discuss how to compute smooth implicit functions with these properties.

3.7 Summary

In this chapter, we investigated the mathematical characterization of the implicit definition of surfaces and solids. We showed that the implicit function is the main component of the implicit model and that its variations define level surfaces. We also demonstrated that a solid and an implicit function are related by dual structures: the skeleton of the solid and the maximal tubular neighborhood of its boundary. These fundamental structures define the embedding of a shape in space and, therefore the characteristics of the implicit function. Finally, we discussed a set of criteria which could be used for analyzing implicit models. From an interpretation of the implicit function as a distance function, we identified two classes of implicit models – the ones based on a true metric and on a pseudo-metric.

Although we have built upon many known mathematical concepts from differential geometry, analysis and topology, a general characterization of the implicit definition of shapes has not been attempted before. In particular, the connection between skeletons and the tubular neighborhood is an original result. Previous related work was restricted either to particular types of models or to particular applications; for example, the research in vision on skeleton models (Blum, 1967).

Note: Distance Function Models

Distance function models are closely related to cyclographic maps and to the solution of the eikonal equation (Hoffmann, 1991).

This implicit surface model is based on the Euclidean metric. The surface $\mathcal{S}$ is defined as the zero set $f^{-1}(0)$ of the function $f : \mathbb{R}^n \to \mathbb{R}$,

$$f(x) = d(x, p) - r(p),$$

where $d(x, p)$ is the distance from x to its closest point p on the interior skeleton of $\mathcal{S}$ and $r(p)$ is the distance of p from $\mathcal{S}$ (as defined in Section 3.5).

The value of f gives the signed distance from the point x to its foot point on $\mathcal{S}$. The gradient of f is a unit vector field, defined over the maximal tubular neighborhood of $\mathcal{S}$, that points in the normal direction to the surface.

Observe that this model generalizes the offset surface model with constant radius r given in Example 3.1.

Chapter 4

Implicit Piecewise Representations

This chapter investigates the piecewise representation of implicit surfaces and solids. It discusses the implicit representation, reviews constructive representation schemes and identifies approaches for creating decomposition representation schemes.

In the previous chapter, we presented a characterization of general implicit shape models. The implicit function can be represented numerically by uniform sample values in the form of a volumetric array. Although such a representation is the most general description of a function, it is too redundant and not suitable for computations. The main objective of this chapter is to provide the conceptual foundation for the development of effective implicit representations. Ideally, these new representations should have the same descriptive power as the volumetric array without its shortcomings. Furthermore, there must be a practical method to convert to and from a volumetric description.

4.1 Background

The representation of an implicit surface in a geometric modeling system consists of a symbolic description of the implicit function, along with the numeric parameters required to completely determine it.

4.1.1 The Implicit Form

An implicit surface is the level set of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$. The function f may be given in one of two ways:

- Continuous Form
- Sampled Form

In the first case, f is defined by an analytic equation that allows the computation of its value at any point of the domain. In the second case, the value of f is known only at a discrete set of sample points. An interpolation scheme must be used to reconstruct the continuous function and allow the calculation of its value at other points of the domain.

The ability to convert between these two forms is a key issue in the computation with implicit surfaces. It also provides a link between geometric models and objects of the real world.

4.1.2 The Implicit Representation

The representation of an implicit surface $\mathcal{S} = f^{-1}(c)$ depends on how the function f is specified in a modeling system. In this respect, there are two types of representation schemes:

- Atomic
- Piecewise

In the atomic representation, an implicit surface is associated with a function f that is a primitive to the system. Primitive functions usually describe a family of surfaces. In this case, f has parameters that allows the selection of a member of the family. The representation consists of an identifier (of the primitive) and the parameters values. An example of primitive is the implicit function defining the family of superellipsoids introduced in Chapter 3.

In the piecewise representation, an implicit surface is associated with an expression involving composition operators and primitive implicit functions. The representation consists of a symbolic description of the expression and the numerical values involved. An example of compound implicit object is specified by boolean expressions involving set operations as discussed later in this chapter.

4.1.3 Operations with Implicit Functions

The implicit representation is closed under the following operations:

- Sum: $f_1 + f_2 = f_1(x) + f_2(x)$;
- Product: $f_1 * f_2 = f_1(x) f_2(x)$;
- Maximum: $\max(f_1, f_2) = \max\{f_1(x), f_2(x)\}$;
- Minimum: $\min(f_1, f_2) = \min\{f_1(x), f_2(x)\}$;

In other words, if two functions f_1 and f_2 are valid implicit functions, then their combination using any of the above operations is also a valid implicit function [1].

In general, if $g : \mathbb{R}^k \to \mathbb{R}$ is an algebraic implicit function, it can be used to define the composite implicit function of k implicit functions $\mathbf{f} = (f_1, \ldots, f_k)$, $g \circ \mathbf{f} : \mathbb{R}^n \to \mathbb{R}$,

$$g(f_1, \ldots, f_k).$$

[1] Note that the unary operations, αf and f^α, defined in Chapter 3 are particular cases of the n-ary operations $f_1 + \ldots, f_n$ and $f_1 * \ldots * f_n$, $n = \alpha$, where $\alpha \in \mathbb{N}$.

Also, an arbitrary implicit function can be approximated by a composite implicit function.

These arithmetic operations with implicit functions correspond to geometric operations with the associated implicit objects. For this reason it is important to understand their effects and meaning.

4.1.4 Piecewise Descriptions

Piecewise descriptions can be further subdivided into *constructive* and *decomposition* representation schemes. Constructive schemes assemble the model using different types of primitives and operators. Decomposition schemes take a given shape and form the model using only one type of primitive and operator.

Constructive implicit models have been extensively researched by the solid modeling community (Boyse and Gilchrist, 1982), (Brown, 1982).

In this thesis, we will explore decomposition models of implicit surfaces. This is motivated by the fact that we already have a way to generate a volumetric description of a shape (see Chapter 5).

4.2 Constructive Schemes

This section reviews constructive representation schemes. Its main purpose is to provide the background for a comparison between constructive and decomposition schemes.

In a constructive scheme, the implicit model is created from primitive implicit solids using composition operators.

Function composition combine different implicit functions into a new one using arithmetic operations. This scheme corresponds to shape operations, such as boolean or blends, that are used to create composite implicit objects from primitive ones.

It is worth noting that the algebraic structure of function composition can be exploited in many ways in a modeling system.

4.2.1 Boolean Operations

The max and min functions can be used to define set operations with implicit solids. In such a scheme, compound objects are constructed from the union and intersection of primitives. Other operations, such as difference, are defined using complementation.

An implicit CSG solid is any set of points in $\mathbb{R}^n$ which satisfy $f(x) \leq 0$ for some function f, where f is either a primitive implicit function or constructed from primitives by Boolean set operations (Requicha, 1980).

The Boolean set operations are defined as:

$f_1 \cup f_2 = \min(f_1, f_2)$.

$f_1 \cap f_2 = \max(f_1, f_2)$.

$f_1 \setminus f_2 = f_1 \cap \overline{f_2} = \max(f_1, -f_2)$.

Figure 4.1 shows an example of these operations.

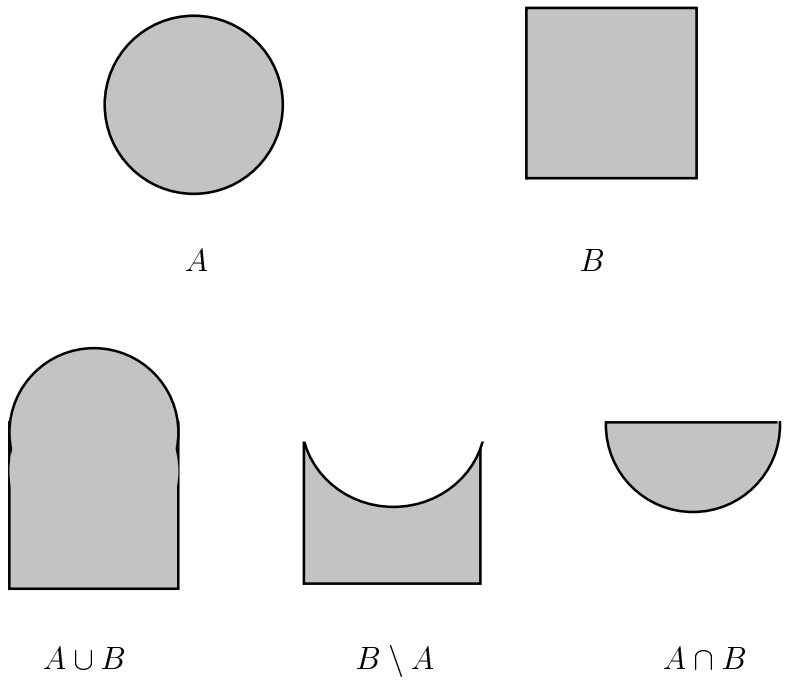


Figure 4.1: CSG operations

One drawback of the CSG scheme is that Boolean operations are implemented using *max* and *min* functions, which are not differentiable everywhere. Consequently, the derivative of the composite implicit function is not defined along the intersection of surfaces.

4.2.2 Blend Operations

A blend of two solids forms a smooth transition between their bounding surfaces. The sum and product operations can be used to define a global blend of implicit solids.

The blending function $b : \mathbb{R}^k \to \mathbb{R}$ creates a transition of isosurfaces based on those operations. The resulting composite function is given as $b(f_1(x), \ldots, f_k(x))$.

Figure 4.2 illustrates a global union blend of two solids.

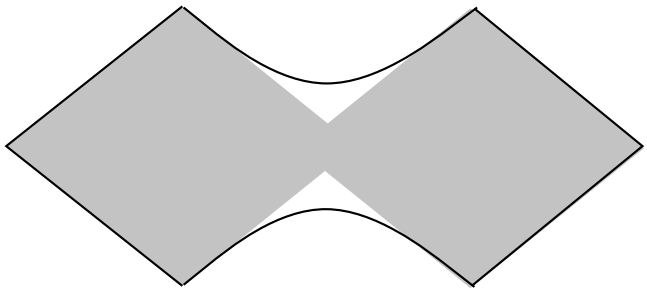


Figure 4.2: Union blend operation

A general form of the function b, known as the superelliptic blend (Rockwood and Owen, 1987), is defined by

$$\left[\sum_{i=0}^{k}(f_i+1)^p\right]^{\frac{1}{p}}-1.$$

The similarity with a metric function is not a coincidence, The blending function gives the weighted distance to the level surfaces [2].

Blend operations may be interpreted as a differentiable approximation of Boolean set operations (Ricci, 1973). Assuming strictly positive implicit functions $g(x,y,z) \in (0,\infty)$ and implicit solids defined by $g(x,y,z) \leq 1$, the following limits are true (Tavares and de M. Gomes, 1989):

$$\lim_{p\to\infty}(g_1^p+\ldots+g_k^p)^{\frac{1}{p}} = \max(g_1,\ldots,g_k)$$
$$\lim_{p\to\infty}(g_1^{-p}+\ldots+g_k^{-p})^{-\frac{1}{p}} = \min(g_1,\ldots,g_k)$$

Essentially, we take the L_n norm on $E \times E$ as an approximation to the maximum, or L_∞, norm on the ring of expressions E defining the implicit function g. When n is small, these operations correspond respectively to smooth global union and intersection blends.

4.2.3 Composite Skeletons

Blending operations are very powerful if used in conjunction with primitives defined by skeletons.

Consider the implicit primitive where f is defined in terms of the distance $d(o,p)$ from a point skeleton o. This model can be extended for a composite skeleton formed

[2] An alternative function is the hyperbolic blend (Kleck, 1989), which is defined in terms of a product instead of a sum.

by a set of isolated points. The implicit function is a blend of the distances from all points

$$f(p) = \left[\sum_{i=1}^{k} d(o_i, p)^q\right]^{1/q} - r.$$

Here each point of the skeleton is associated with a metric space M, and we work with the cartesian product $M_1 \times \cdots \times M_k$ (Blinn, 1982), (Wyvill, McPheeters and Wyvill, 1986b), (Nishimura et al., 1985).

Observe that a primitive defined by higher dimensional skeletons, such as a curve, can be considered as a generalization of the composite point skeleton. Now, the point set is continuous and the summation becomes an integral (Bloomenthal and Shoemake, 1991)

$$f(p) = \int_S d(s, p)ds - r.$$

Alternatively, this type of model can be defined using the concept of distance from a point p to a set S

$$d(p, S) = \inf_{s \in S} d(p, s).$$

We remark that these two schemes do not give necessarily the same geometry.

4.3 Decomposition Schemes

In decomposition schemes, the starting point is a complex implicit function that must be represented in terms of one type of primitive implicit function. Normally, the composite model will give only an approximation of the original object. The importance of such a representation is its good computational properties.

4.3.1 Approaches

There are two general approaches to decomposing a function for computation purposes

- Spatial Decomposition
- Functional Decomposition

The first approach decomposes the domain of the function into subdomains and works separately with each subdomain. The original function is then approximated in each subdomain by a simpler function. An example of the this type of decomposition can be seen in the sampling and reconstruction of functions using linear interpolants.

The second approach decomposes the function itself by a set of simpler functions that together constitute a good approximation of the function. An example of this type of decomposition can be seen in the Fourier series in which a function is represented by a summation of scaled sinusoids.

These two decomposition approaches are present, for example, in the computation of solutions of partial differential equations. Another interesting observation, is that these approaches constitute the basic strategies to obtain theoretical results in different areas of mathematics, such as the integration of functions on manifolds.

There are cases in which these two approaches are equivalent and correspond to the same representation.

One example is the piecewise description based on uniform space subdivision and linear interpolants. This is essentially a representation that is equivalent to the one given by linear B-spline basis of the space of piecewise linear functions. Figure 4.3 illustrates both descriptions for a one-dimensional function.

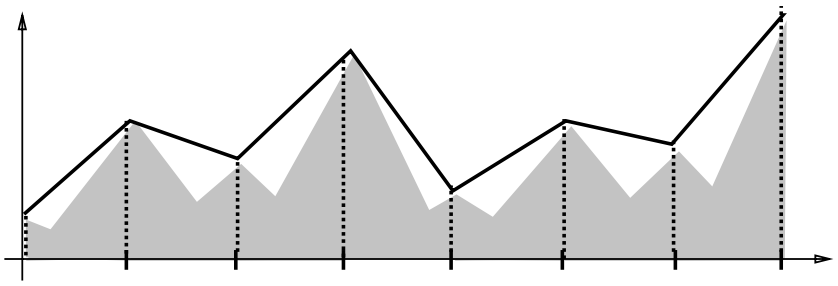


Figure 4.3: Uniform Subdivision and Linear Splines

Another example can be seen in the finite element method in which a function is represented by a collection of elements each corresponding to a subdomain of the function. The elements are constructed from nodal points that are interpolated giving function values at all points of the domain. The element shape functions are polynomials that define a basis.

4.3.2 Adapted Decompositions

In order to be most effective, the function decompositions should satisfy some requirements related to the computational tasks that will be performed on them. They should provide a *suitable approximation* of the original function (i.e. within the required precision).

Also, the representation resulting from the decomposition should be *simple* and *compact*. Intuitively, these goals can be achieved if the decomposition is *adapted* to variations of the function value. In the context of the domain subdivision, this implies that the space partition is finer where the function exhibits large variations and coarser where the function does not vary significantly. In the context of functional decomposition, this implies that high frequency waves represent the highly variable portions of the function and low frequency waves represent portions of lower variation.

One additional requirement that is also important computationally is *spatial locality*. The extent of influence of each piece of the decomposition should correspond to a finite region of the domain of the function. Note that this requirement is not satisfied in the case of a Fourier decomposition.

4.4 Summary

In this chapter we analyzed the representation of implicitly defined shapes. We discussed the forms used to describe the implicit function and their representations. We developed a classification of piecewise representation schemes. Constructive schemes are indicated for shape design and have been extensively researched. Decomposition schemes are suited for computation and will be further investigated in this thesis.

Chapter 5

Smooth Implicit Functions

This chapter studies multiscale edges as a characteristic of the implicit function that is closely related to its level surfaces. We develop a method to generate smooth implicit functions from edge information.

We have seen in Chapter 3, that one of the possible choices of implicit model for a solid object is based on pseudo-metric functions. This type of function vary smoothly and must have as one of its isosurfaces the boundary of the solid. In general, there are many functions satisfying these properties, and the extra degree of freedom can be used to provide control over a prescribed tubular neighborhood of the bounding surface of the object.

The problem of constructing such a function is important because it gives an implicit model for which we have explicit information about the variations of the implicit function. This knowledge is crucial to robust computation with implicitly defined objects.

5.1 Multiscale Edges

An edge of a real valued function is associated with points of sharp variation of the function. These points are among the most fundamental features for analyzing structural properties of the function.

5.1.1 Variation and Regularity

The behavior of a function can be precisely described through the Lipschitz and Hölder conditions.

The variations of the function f are measured by its Lipschitz constant.

A function $f : U \subset \mathbb{R}^n \to \mathbb{R}$ is Lipschitz with constant K if, for all pairs of points p and q in U, f satisfies

$$|f(p) - f(q)| \leq K|p - q|,$$

where $|\ |$ is a norm in $\mathbb{R}^n$ and $K > 0$. This condition implies that f is differentiable and has bounded derivative.

More precise information about the differentiability of f is given by its Hölder exponent. A function f is Hölder continuous at the point p with exponent α if there

exists a constant $\alpha < 0$ such that

$$|f(p) - f(q)| \leq K|p - q|^{\alpha}$$

for a $p \in U$ and all points q in a ϵ-neighborhood of p.

The function f is uniformly Hölder α over the domain U if there exists a constant K such that the above condition holds for any $p, q \in U$.

The Hölder regularity α_p of a function f is the superior bound of all values α such that f is Hölder α at p. Note that the Lipschitz condition is a particular case of the uniform Hölder condition for which $\alpha = 1$.

When $n < \alpha_p < n+1$, $n \in \mathbb{N}$, f is n times differentiable at p but its n-th derivative is singular at that point.

5.1.2 Edges and Scale

The study of a function at different scales gives information about its behavior on neighborhoods of variable size. Particularly, the evolution of edges across scales reveals the nature of function variations over its domain.

The definition of edges at multiple scales requires the auxiliary notion of smoothing. A *smoothing function* is any real function θ whose integral is non-zero. The smoothing function corresponding to scale s is

$$\theta_s(x) = \frac{1}{s}\theta(\frac{x}{s}).$$

In the context of signal processing, it can be interpreted as the impulse response of a low-pass filter. One example of such a function is the Gaussian.

The edges of a function f at scale s, or *multiscale edges*, are defined as the sharp variations of $f(x)$ convolved with $\theta_s(x)$.

For the special case of a Gaussian, the characterization above has a simple physical interpretation. Assuming that the function f measures spatial density, its representation at a given scale s is the result of a diffusion process whose duration is proportional to s. This gives a one-parameter family of functions $F(x; s)$ that is a solution of the *diffusion equation*

$$\frac{d^2F}{dx^2} = \sigma\frac{dF}{ds}$$

with boundary condition $F(x; 0) = f(x)$ (where σ is a diffusion constant).

The kernel of this partial differential equation is the Gaussian. So, instead of solving the diffusion equation, f may be convolved with a Gaussian filter that replaces the density with its average value over a window of diameter s. When s is small, the smoothing of f by θ_s is negligible and its edges provide the locations of sharper function variations. When s is large, the convolution with θ_s removes small oscillations so that edges reveal sharp variations of larger structures.

Edges of a function can be detected from the information provided by its first or second derivatives. In one dimension, the location of multiscale edges are, by convention, at inflection points of the function f smoothed by θ_s. These points correspond to the extrema of the first derivative, or equivalently, to the zero-crossings of the second derivative of the function.

In n dimensions, the edges of a function $f : \mathbb{R}^n \to \mathbb{R}$ at scale s are defined as the set of points where the magnitude of the gradient of f convolved with θ_s is maximum in the direction indicated by the gradient vector. Thus, edge points are the inflection points of the hypersurface $g = \text{graph}(f * \theta_s)$.

5.1.3 Relations with f

In the case of implicit functions, edges are generally located at the boundaries determined by some level surface. This is a consequence of the fact that in order to define a surface $\mathcal{S} = f^{-1}(c)$, the value of f has to go from $c - \epsilon$ to $c + \epsilon$ in the direction normal to $\mathcal{S}$.

Although the function f may oscillate in other parts of its domain without crossing the level c, for implicit surface models the only interest is in the variations producing that isocontour.

Normally, it is reasonable to assume that all variations of an implicit function f are associated with some family of surfaces that is defined by f.

5.2 Edge Detection and Reconstruction

In this section, we describe a method for edge detection and reconstruction with wavelets that was developed by Mallat and Zhong. This method will be used in our process to generate a smooth implicit function from multiscale edges.

The wavelet transform is a mathematical tool for the analysis of function variations at multiple scales (See Appendix B). In particular, an appropriately chosen dyadic wavelet transform allows the detection of multiscale edges very efficiently (Mallat and Zhong, 1992b). From this type of information it is possible to reconstruct a close approximation of the function (Mallat and Zhong, 1992a).

5.2.1 Edge Detection with Wavelets

Multiscale edges of a function f are the inflection points of a smoothed version of f, $f_s = f * \theta_s$, that is obtained convolving it with an appropriate low-pass filter θ_s.

Since the wavelet transform of a one-dimensional function is defined as a convolution with a family of functions $\psi_{s,x}$,

$$(T_\psi f)(s, x) =< f, \psi_{s,x} >= f * \psi_{s,x} \quad ,$$

where s is a scale parameter and x is a translation parameter, it constitutes a good mechanism to compute multiscale edges.

As we have seen in Appendix B, we can define a wavelet ψ which is the derivative of a scaling function ϕ (in the present context, the smoothing function *phi*). If ψ is chosen to be the second derivative of ϕ, then the edges of f at multiple scales are given by the zero-crossings of $(T_\psi f)(s, x)$. Similarly, if ψ is chosen to be the first derivative of ϕ, then the edges of f correspond to the extrema of $(T_\psi f)(s, x)$. This is illustrated in Figure 5.1.

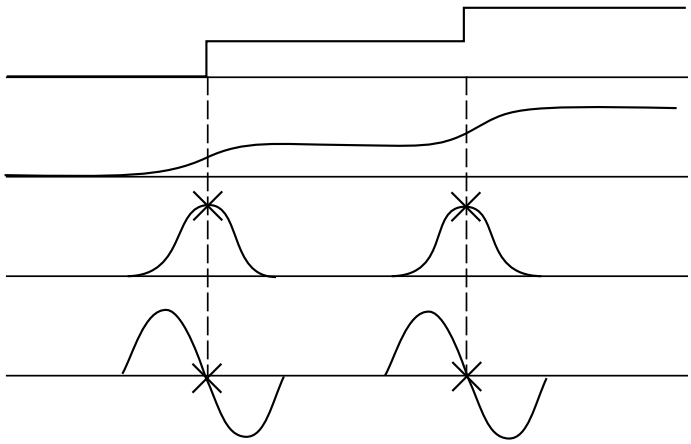


Figure 5.1: Multiscale Edge Detection

When ϕ is a Gaussian, the first procedure is exactly a Marr-Hildreth edge detection (Marr and Hildreth, 1980) and the second is the same as a Canny edge detection (Canny, 1986). In principle, these two procedures are equivalent but the extremum detection has some advantages because it allows to distinguish between different types of edges (Mallat and Hwang, 1991).

It is implied in the above definition of $\psi_{s,x}$ that the parameters s and x vary continuously, therefore $(T_\psi f)(s, x)$ is a continuous wavelet transform. For most purposes, a continuum of scales is not required. The best option is to employ a dyadic wavelet transform in which the scale parameter varies along the sequence $(2^j)_{j\in\mathbb{Z}}$. It provides a natural decomposition of the frequency domain and allows fast numerical implementations (Mallat and Zhong, 1992a).

5.2.2 Analysis of Multiscale Edges

The dyadic wavelet transform allows a complete characterization of the singularities of a function. If the wavelet is the first derivative of a smoothing function, the multiscale information gives a precise description of the function edges as well.

The evolution across scales of the wavelet transform depends upon the Hölder regularity of the function. From the maxima of the wavelet transform it is possible to compute the Hölder exponent at singular points. The decay of $(T_\psi f)$ also allows one to measure smooth variations of the function. This is stated precisely in the theorem below.

Theorem 5.1 *A function $f : \mathbb{R} \to \mathbb{R}$ is Hölder continuous with exponent α over an interval I if there exists a $K > 0$ such that the wavelet transform of f satisfies*

$$|(T_\psi f)(2^j, x)| \leq K(2^j)^\alpha,$$

for all points $x \in I$.

A proof can be found in (Mallat and Hwang, 1991).

The Hölder regularity is computed by finding the constant K and the exponent α such that $K(2^j)^\alpha$ best approximates the decay of $|(T_\psi f)(2^j, x)|$ over all scales 2^j.

This gives a method to discriminate different types of singularities of the function f. When f is not singular, its smoothness at a point x_0 is modeled as the convolution of a function h, that is singular at x_0, with a Gaussian of variance σ^2.

If the wavelet ψ is the derivative of a function ϕ that is close to a Gaussian in the sense that $\phi_{2^j} * g_\sigma \approx \phi_{s_0}$, with $s_0 = \sqrt{2^{2j} + \sigma^2}$, then the wavelet transform of f can be written as

$$(T_\psi f)(2^j, x) = 2^j \frac{d}{dx}(f * \phi_{s_0})(x) = \frac{2^j}{s_0}(T_\psi h)(s_0, x),$$

where $(T_\psi h)(s_0, x)$ is the wavelet transform of h at scale s_0. This shows that the wavelet transform at scale 2^j of a singularity smoothed by a Gaussian of variance σ^2, is equivalent to the wavelet transform of a non-smoothed singularity at scale $\sqrt{2^{2j} + \sigma^2}$.

It is not a coincidence that such an approach is essentially the same as the one used in the definition of multiscale edges (see Section 5.1).

Although the above theorem only characterizes Hölder exponents over intervals, it is sufficient to study isolated singularities and therefore it applies to edge detection (Mallat and Hwang, 1991). The theorem also extends trivially to n-dimensions.

5.2.3 Reconstruction from the Wavelet Maxima

A method to reconstruct a close approximation of a function from the maxima of its wavelet transform was developed by (Mallat and Zhong, 1992a). It relies on the fact that a rich representation of a function f is given by the local extrema of the wavelet transform $(T_\psi f)(2^j, x)$ of f with respect to the wavelet $\psi = \frac{d\phi(x)}{dx}$ and by the values of $(T_\psi f)(2^j, x)$ at the corresponding locations, (i.e. the multiscale edges of f).

The algorithm employs a spline wavelet function that is the first derivative of a cubic spline, similar in shape to a Gaussian. A wavelet with only one vanishing moment is chosen because the main interest is in a characterization of sharp transitions of the function itself.

The reconstruction is accomplished through an iterative procedure based on alternating projections. The derivation of the method follows from a characterization of the set of functions g which have the same wavelet transform maxima as the function f.

It is intuitive that, for any scale 2^j, there exists an infinite set of functions g_j with local extrema equal to $(T_\psi f)(2^j, x)$. However, such a sequence of functions $(g_j)_{j\in\mathbb{Z}}$, is the dyadic wavelet transform of a function if and only if it satisfies the reproducing kernel associated with ψ (for more details see Appendix B).

The reconstruction problem can be broken in two conditions. Let $(x_{j,i})_{i\in\mathbb{Z}}$ be the abscissa where $|(T_\psi f)(2^j, x)|$ is locally maximum. The wavelet transform of g, at each scale 2^j, has to satisfy the following:

1. The local maxima of $|(T_\psi g)(2^j, x)|$ are located at $(x_{j,i})$.

2. For each local maximum at $x_{j,i}$, $(T_\psi g)(2^j, x_{j,i}) = (T_\psi f)(2^j, x_{j,i})$.

Define the set Γ of all sequences $(g_j)_{j\in\mathbb{Z}}$ such that, for all scales 2^j, g_j has the same extrema as $(T_\psi f)(2^j, x)$. Define also the space V of all valid dyadic wavelet transforms $(T_\psi f)(2^j, x)$. Then, the solution of the problem above is expressed as

$$\Lambda = \Gamma \cap V = \left\{ (T_\psi f)(2^j, x) \right\}, \quad j \in \mathbb{Z}.$$

In order to find the solution, it is necessary to compute the intersection of Γ with V. One can prove that, since Γ is an affine space and V a Hilbert space, alternate projections on Γ and V converge to the orthogonal projection on Λ. This is illustrated by Figure 5.2.

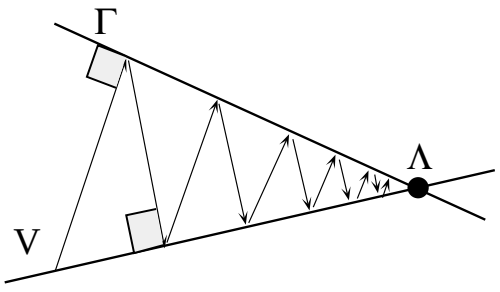


Figure 5.2: Alternating Projections

The orthogonal projection on the space V is implemented by performing a inverse wavelet transform T_ψ^{-1} followed by a direct wavelet transform T_ψ. This is a consequence of the fact that any dyadic wavelet transform is invariant under the operator

$$P_V = T_\psi \circ T_\psi^{-1}.$$

The operator P_Γ is implemented by adding piecewise exponential curves to each function of the sequence (g_j) such that for any index $j \in \mathbb{Z}$ and all maxima positions $x_{j,i}$,

$$g_j(x_{j,i}) = (T_\psi f)(2^j, x_{j,i}).$$

The reconstruction algorithm consists of applying the operator $P = P_V \circ P_\Gamma$ until convergence, because any element of Λ is a fixed point of P.

(Mallat and Zhong, 1992a) conjectured that the local extrema constitute a complete representation, i.e., the intersection of Γ with V has only one element. It has been proved that this conjecture is false (Meyer, 1993), but all functions $g \in \Lambda$ are very close to f and, in practice, the reconstruction algorithm always computes a good approximation of the original function.

5.3 Generating a Smooth Implicit Function

An important problem in modeling with implicit surfaces is the generation of an implicit function having the boundary of a given solid object as one of its level sets.

In this section, we develop a method to compute such an implicit function from the edge information provided by the characteristic function χ of the solid. The method is based on the wavelet edge detection and reconstruction algorithm described in Section 5.2. It exploits multiscale edge information to provide control over the properties of the resulting implicit function. This is done by manipulating the values of the wavelet maxima across scales.

5.3.1 Method

The main idea behind this method is the observation that a function $f : \mathbb{R}^n \to \mathbb{R}$ can be determined from its edges at multiple scales.

The dyadic wavelet transform maxima of f gives the location of its edges at scales 2^j, as well as a measure of how smooth f is in a neighborhood of these edges. Thus, if the maxima values are altered, then the smoothness of f changes. Under certain conditions, it is possible to make f smoother while maintaining the positions of its edges invariant. This means that an implicit function f defining a surface $\mathcal{S} = f^{-1}(c)$ may be manipulated without modifying the isosurface at level c. On the other hand, this procedure may be also used to modify the shape of $\mathcal{S}$ if that is desired.

The method consists of the following steps:

1. Start with a function $\chi_{\mathcal{O}} : U \to \{0, 1\}$, that is the characteristic function of a solid object $\mathcal{O}$ whose boundary is the surface $\mathcal{S}$.

2. Compute the dyadic wavelet transform $(T_\psi \chi_{\mathcal{O}})(2^j, x)$ of $\chi_{\mathcal{O}}$ at scales 2^j, $j = 1, \ldots, J$ as defined in Subsection 5.2.1.

3. Identify the local maxima of $|(T_\psi \chi_{\mathcal{O}})(2^1, x)|$ and record their locations.

4. For each point belonging to an edge of $\chi_{\mathcal{O}}$ at scale 1, analyze the evolution of the wavelet transform across scales and derive new wavelet maxima values for scales $2^j, j = 1, \ldots, J$.

5. Generate a smooth function f from the maxima values using the iterative reconstruction algorithm of Subsection 5.2.3.

5.3.2 Smoothing the Edges

The crucial part of the above method is the determination of how to derive the wavelet maxima to obtain the desired effect on the edges of f. This procedure replaces the function discontinuities of $\chi_{\mathcal{O}}$ corresponding to the boundary of the solid with smoother transitions which still define the same bounding surface.

Let us first investigate the problem in one dimension and consider the wavelet transform maxima of an isolated step edge $g(x)$, range of $g \in \{0, 1\}$. Since the function g is discontinuous at the edge, it has Hölder regularity $\alpha = 0$. Consequently, the value of the wavelet transform at that point is constant at all scales.

$$|(T_\psi g)(2^j, x)| = C$$

Figure 5.3 shows a plot of a step edge and its wavelet transform maxima at scales 2^j, $j = 1, \ldots, 6$.

The goal of the smoothing procedure is to transform the singularity at the edge of g in a regular point which is an inflection point of a smooth function h (i.e. h is differentiable and its first derivative has an extremum at that point). This is equivalent to creating an edge with Hölder exponent $\alpha = 1$. The wavelet transform maxima of such an edge evolves as

$$|(T_\psi h)(2^j, x)| = K 2^j.$$

Figure 5.4 shows a plot of a smooth edge and its wavelet transform maxima at scales 2^j, $j = 1, \ldots, 6$.

In order to produce a smooth edge, select a scale 2^J as the reference and assume that $C = K 2^J$. The wavelet transform maxima at all scales 2^j, $j < J$, should evolve as $C 2^{j-J}$. Take C as the maxima value at scale 2 and set the maxima values at the remaining scales 2^j, $j = 2, \ldots, J$ according to the rule above. When an edge is

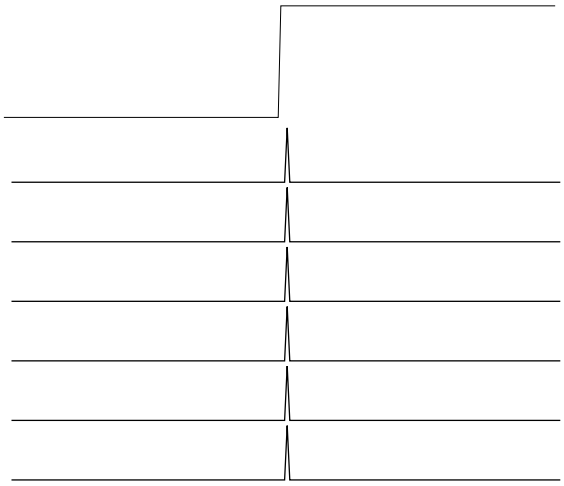


Figure 5.3: Wavelet Transform Maxima of a Step Edge

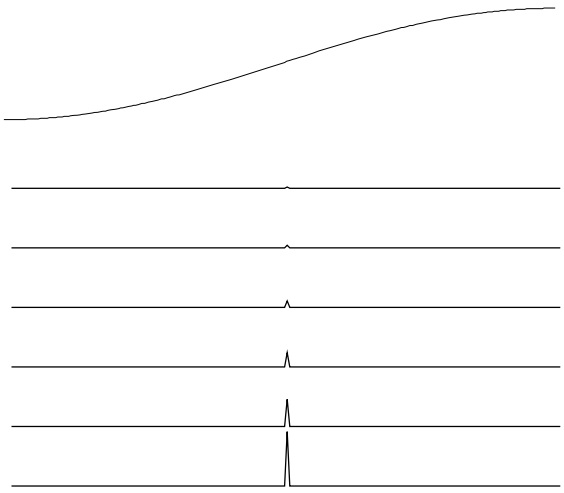


Figure 5.4: Wavelet Transform Maxima of a Smooth Edge

reconstructed from these maxima values, the smoothness of the edge is proportional to the scale 2^J and the value of h at the edge should be 1/2.

The smoothing of a function may cause a change in its edge locations or even make close edges merge together.

When more than one edge is considered, it is necessary to detect at which scale the smoothing of one edge may cause an interference in other edges, and thus, result in an undesirable change in the function. The region of influence of a point p at scale s is equal to the support of the smoothing function ϕ_s centered at p. Two edges do

not interfere at a given scale if their regions of influence have a null intersection at that scale.

In order to be able to preserve the original edge locations in the smoothed function, the edges should be diffused to the extent that they do not interfere with each other.

This type of interference can be detected from the wavelet transform. Since we started with an original function containing only step edges, the value of the wavelet transform maxima should remain constant across scales. This will be true for all scales 2^j, unless there is an interference from the smoothing of another edge at some scale. When processing an edge, this fact is used to determine the scale of reference 2^J for smoothing the edge without interferences of other edges.

More precisely, for each edge point p, of the characteristic function $\chi_{\mathcal{O}}$ at scale 1, we analyze the evolution of the dyadic wavelet transform of $\chi_{\mathcal{O}}$ from scale 2 to scale $2^{J_{\max}}$ in order to determine at which scale the value of $(T_\psi \chi_{\mathcal{O}})(2^j, x)$ fails to remain constant. This scale 2^{J_s} is taken as the reference scale for smoothing at the point p. For scales 2^j, $j = 1, \ldots, J_s$, we create a wavelet maxima point with value $C2^{j-J_s}$, where C is the original value of the wavelet transform of $\chi_{\mathcal{O}}$ corresponding to point p at scale 2^{J_s} (which, as we have seen, is constant for a step edge in this range of scales). For scales 2^j, $j = J_s, \ldots, J_{\max}$, we simply create a maxima point whose value is equal to $(T_\psi \chi_{\mathcal{O}})(2^j, x)$ at $x = p$.

Figure 5.5 shows the smoothing of a function containing several step edges based on the method described above. In the figure, the top row displays the initial and final functions superimposed, for better comparison. The other rows display the maxima values computed using the smoothing algorithm. Since the edges are spaced by different amounts, they are diffused to different degrees by the reconstruction procedure.

To extend the smoothing method to n-dimensions we need to consider the magnitude as well as the direction of the gradient vector field of the smoothed function, as discussed in Appendix B.

As we have seen, in n-dimensions, the edges of a function f are defined as the points where the modulus of the gradient vector of $f * \phi$ is maximum along the direction of $\nabla(f * \phi)$. In Appendix B, we describe the representation of the wavelet transform in spherical coordinates. This form gives exactly the modulus and direction angles of the gradient of f smoothed by ϕ (recall that the wavelet functions ψ_i are the partial derivatives of the smoothing function ϕ).

For an n-dimensional step edge, both the modulus and the direction of the wavelet transform remain constant across scales. For a smooth edge, the modulus of the wavelet transform evolves as in the 1-dimensional case, while its direction remains constant.

Figure 5.6 shows the spherical representation of the wavelet transform of the image of a 2-dimensional smooth function.

Figure 5.7 is a plot of the amplitude of the modulus of wavelet transform corresponding to the central horizontal line of the image in Figure 5.6.

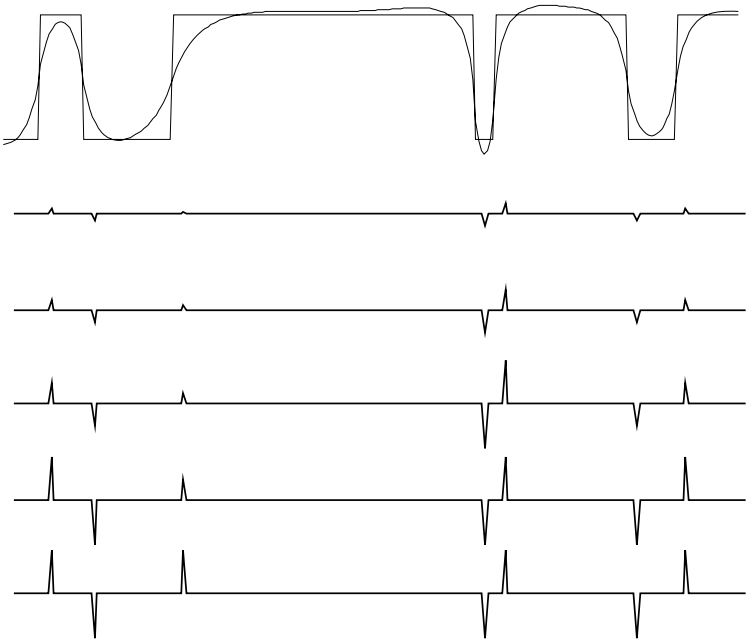


Figure 5.5: Smoothing of Multiple Edges

The method in n-dimensions is essentially the same as in 1-dimension, except that the modulus and the direction of the wavelet transform must be considered. The determination of the reference scale 2^{J_s} is based on the evolution of both the modulus and the direction of $(T_{\psi_i}\chi_{\mathcal{O}})(2^j; x, y)$. To create a wavelet maxima point, the value of the modulus is attenuated as in the 1-dimensional case, while the direction is maintained. This is intuitive, since the goal is to smooth the edge while preserving its location as well as its normal vector field.

Example 5.1 (Diamond Shape) This example illustrates the use of the method in two dimensions.

The original function is the binary image of a diamond shape shown in Figure 5.8(a). The smoothed function is shown in Figure 5.8(b). Note that the original image was blurred by different amounts along the edges. Near the corners there is almost no blur while at the midpoints of the edges there is extensive blurring. Figure 5.8(c) shows the isocontour of the smoothed image corresponding to gray level 0.5.

Example 5.2 (Sphere) This example demonstrates the use of the method in three dimensions.

The shape is a ball defined by a density array of $64 \times 64 \times 64$ samples. A smooth implicit function is generated from this data set using the method. The surface

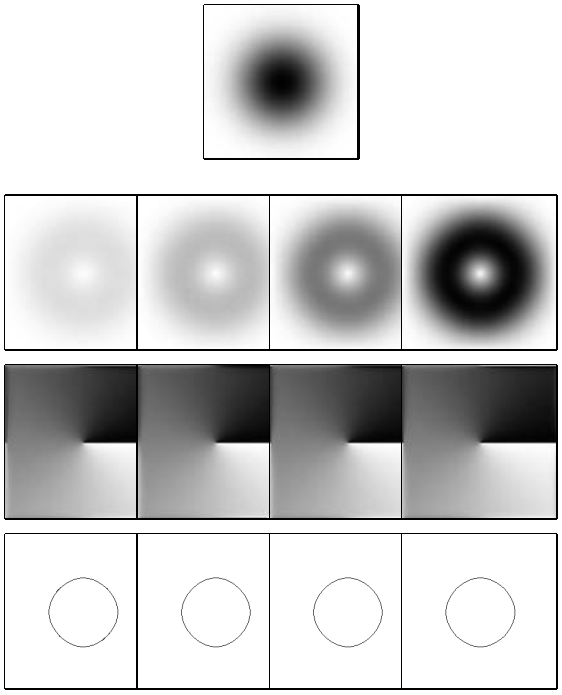


Figure 5.6: Spherical Representation of the Wavelet Transform of a Smooth Edge

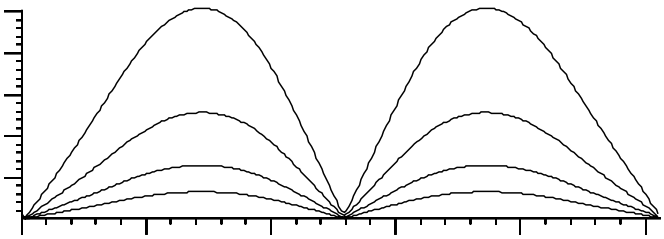


Figure 5.7: Evolution of the Modulus of the 2D Wavelet Transform

corresponding to isovalue 0.5 is inferred from the smooth function and a polygonal approximation is produced. Figure 5.9(a) shows an image of the input density array. Figure 5.9(b) shows the smooth implicit function. Figure 5.10 shows the 64 slices of the density array for the smooth implicit function. Figures 5.11(a) and (b) show a close view of input and output density arrays. Figure 5.12 shows the polygonal approximation of the isosurface extracted from the function in Figure 5.9(b) using a polygonization algorithm. Figure 5.13 shows a shaded image of the surface. Note

that the sphere's surface was completely recovered in spite of the low resolution of the input array.

5.3.3 Controlling Smoothness

The evolution across scales of the wavelet transform maxima can be used to limit the variations of the implicit function and control the tubular neighborhood of the implicit surface.

The smoothness of the edges defining a surface $\mathcal{S}$ is related to a tubular neighborhood of $\mathcal{S}$ in which the implicit function gives useful information about $\mathcal{S}$. Particularly, the gradient of f at a point q in this neighborhood gives a path to the foot point of q in $\mathcal{S}$.

The maximum tubular neighborhood $V_\phi^{\max}(p)$, relative to a smooth function ϕ, of a surface $\mathcal{S} = f^{-1}(c)$ at a point p is given by the largest scale s for which a smooth edge can produce the level surface $f^{-1}(c)$.

There are three useful ways to determine multiscale edges for generating a smooth implicit function:

(A) Produce the maximum tubular neighborhood for ϕ, $V_{\max_\phi}(\mathcal{S})$.

(B) Produce a tubular neighborhood with prescribed inferior and superior limits, V_s, $s \in [s_0, s_1]$.

(C) Produce a uniform tubular neighborhood, V_s, $s = K$.

Note that in option (A), s_0 and s_1, correspond to the the inferior and superior limits of $V_{\max_\phi}(\mathcal{S})$. This is the maximum tubular neighborhood at each point $p \in \mathcal{S}$ that can be generated from the multiscale edges of $f * \phi$ such that the level surface $\mathcal{S} = f^{-1}(c)$ is preserved. In option (B), s_0 and s_1 are set arbitrarily. Option (C) is just a particular case of option (B), in which $s_0 = s_1$.

We remark that options (B) and (C) may cause a change in the shape of the surface if the limits exceed the maximum allowable by the smoothing function ϕ and the surface $\mathcal{S}$.

5.4 Applications

The generation of a smooth function f corresponding to a level surface $\mathcal{S} = f^{-1}(c)$ has various applications in modeling and graphics. It provides control over the properties of the implicit function f, providing at the same time valuable information about the tubular neighborhood of the level surface $f^{-1}(c)$. An implicit function that describes the boundary of arbitrary solid shapes can be created using the method described in Section 5.3. The solids may be defined either in implicit or in parametric form.

5.4.1 Conversion from Parametric to Implicit

An important application of the smoothing method is to the conversion of a parametric surface that is the boundary of a solid object to an implicit surface approximating it. The need for this type of conversion may arise in many practical situations, such as in tolerance analysis in CAD/CAM models.

The conversion is performed by first producing a volumetric representation of the characteristic function of the solid region enclosed by the parametric surface.

The volumetric representation is produced in two steps:

1. Rasterize the parametric surface.

2. Fill the volume enclosed by the surface.

Algorithms for rasterization of parametric objects can be found in (Kaufman, 1987). An algorithm for filling in regions from their boundaries can be found in (Pavlidis, 1978).

The resolution required for the volumetric representation can be estimated based on the smallest surface element, in the case of polygonal surfaces, or on surface curvature, in the case of higher order surfaces.

After the discrete characteristic function has been computed, the smoothing method described in Section 5.3 is applied to generate the implicit function.

Now, we give an example that demonstrates the use of the method in the conversion of parametric curves to a volumetric implicit form.

Example 5.3 (B Shape) We start with a letter "B" described by its outline, in the form of a Bezier curve. The outline is rasterized into an image array of 256×256 resolution. The interior of the "B" shape is filled. The smoothing method is then applied to the solid "B" shape.

Figure 5.14(a) shows the control polygon of the two dimensional Bezier curve that describes the font outline. Figure 5.14(b) shows the image of the solid "B" shape after rasterization of the outline and filling its interior. Figure 5.14(c) shows the implicit function resulting from the smoothing process. Figure 5.14(d) shows the isocontour corresponding to level 0.5.

5.4.2 Modification of Implicit Functions

Another important application of the smoothing method is the modification of an implicit function leaving invariant one of its level surfaces. This may be desirable in situations where it is necessary to perform computations with the level surface in implicit form. In that case, if the original implicit function is very irregular, the smoothing process can generate an implicit function having the same level surface, that is better suited to computation.

The modification is performed by first producing a volumetric representation of the characteristic function of the interior region defined by the level surface. This is a straightforward operation that requires only to evaluate the implicit function at each grid point of the volume and compare its value with the value corresponding to the level surface. The value of the characteristic function at that point is set to 0 or 1 depending on outcome of the test (the function value is greater or less than the isovalue).

This example demonstrates the use of the method for modifying an implicit function given in volumetric form.

Example 5.4 (Head Slice) The data set is an MRI (Magnetic Resonance Imaging) slice of a human head with resolution 256×256. We select an isocontour corresponding to some features of interest (brain, etc). To create the solid regions enclosed by these isocurves, we set to 1 or 0 the sample values whose density is respectively less or greater than the selected isodensity. The resulting binary array gives the characteristic function of the shapes of interest in the data. The smoothing procedure is applied to it.

Figure 5.15(a) shows the original MRI data set. Figure 5.15(b) shows the binary image that defines the features of interest. Figure 5.15(c) shows the smooth implicit function produced by the method. Figure 5.15(d) shows the boundary of the features corresponding to isodensity 0.5 (of the smoothed function).

Note that the method was able to perform a selective smoothing capturing all details of the selected isocurves.

5.4.3 Manipulation of the Tubular Neighborhood

The tubular neighborhood of an implicit solid can be manipulated by restricting the values of the wavelet transform maxima of its characteristic function, as discussed in Section 5.3. This is an important application of the method, because the knowledge of the limits of the tubular neighborhood is the key to robust computation with implicit objects.

The following two examples demonstrate the versatility of the method and the degree of control provided over the characteristics of the smooth implicit function it generates. We use as a starting point the same "B" shape of example 5.3. The multiscale edges are created to limit the tubular neighborhood of the implicit function. Note that, as a consequence, the shape of the font is altered accordingly.

Example 5.5 (Limiting the Tubular Neighborhood) In this example, we limit the tubular neighborhood of the implicit function from above and below. The limits are set by restricting the multiscale edge analysis to scales on the interval 2^j, $j = 3, \ldots, 6$.

Figure 5.16(a) shows the smooth implicit function resulting from the process. Figure 5.16(b) shows a set of evenly spaced isocurves in the tubular neighborhood

of the function. Figure 5.16(c) shows the "B" shape generated by thresholding the function at isolevel 0.5.

Figure 5.17 depicts the two dimensional implicit function as a three dimensional height hypersurface.

Example 5.6 (Uniform Tubular Neighborhood) In this example, we generate an implicit function with uniform tubular neighborhood. The smooth implicit function is reconstructed from the edges at scale 2^j, $j = 6$.

Figure 5.18(a) shows the smooth implicit function resulting from the process. Figure 5.18(b) shows a set of evenly spaced isocurves in the tubular neighborhood of the function. Figure 5.18(c) shows the "B" shape generated by thresholding the function at isolevel 0.5.

Figure 5.19 depicts the two dimensional implicit function as a three dimensional height hypersurface.

5.5 Summary

In this chapter, we developed a method to generate a smooth implicit function from the characteristic function of a solid object. The method is based on multiscale edge synthesis and employs the wavelet edge detection and reconstruction algorithm from (Mallat and Zhong, 1992a). Although, this algorithm has been applied to image processing problems such as compression (Mallat and Zhong, 1992b) and noise removal (Mallat and Hwang, 1991), it has not been applied before to geometric modeling problems.

Our method provides effective control over the variations of the implicit function and of their extent. It also gives information about the tubular neighborhood of the boundary of the generating shape. The method is very flexible and can be used to create an implicit function with the desired characteristics from practically any type of geometric description: parametric, implicit or volumetric. It computes the solution numerically using a fast discrete filtering algorithm. The resulting function is represented in the form of a volumetric array.

One application of this method is to the approximate conversion from parametric to implicit surfaces. This is a general algorithm for arbitrary shapes. Previous algorithms were restricted to particular types of models, such as rational parametric surfaces (Manocha and Canny, 1990), (Kalkbrenner, 1990).

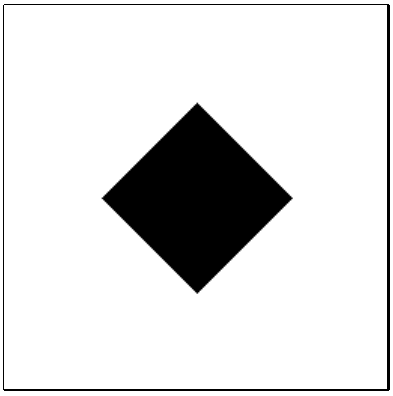
Note: Alternative Methods

An alternative method to generate a smooth implicit function which has as a level surface the boundary of a solid shape is based on a regularization technique used in vision.

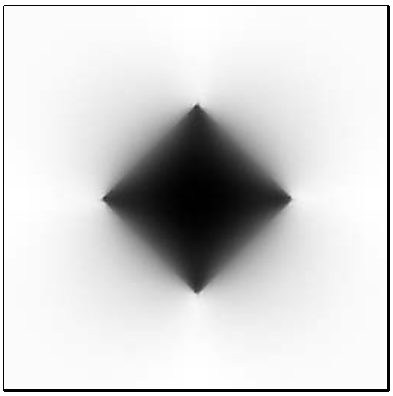
This type of method reconstructs a hypersurface that is a function graph from sparse data (Terzopoulos, 1984). Such a technique allows the generation of an implicit function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ (i.e. a hypersurface in $\mathbb{R}^{n+1}$) that is constrained to have as one of its isocontours the input surface $\mathcal{S}$.

There are several numerical algorithms to implement this technique (Terzopoulos, 1986), (Szeliski, 1989).

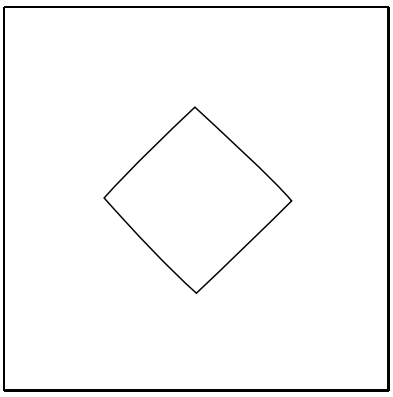
The disadvantage of this method is that it does not provide the same flexibility and control over the properties of the implicit function as provided in the multiscale edge reconstruction.



(a)

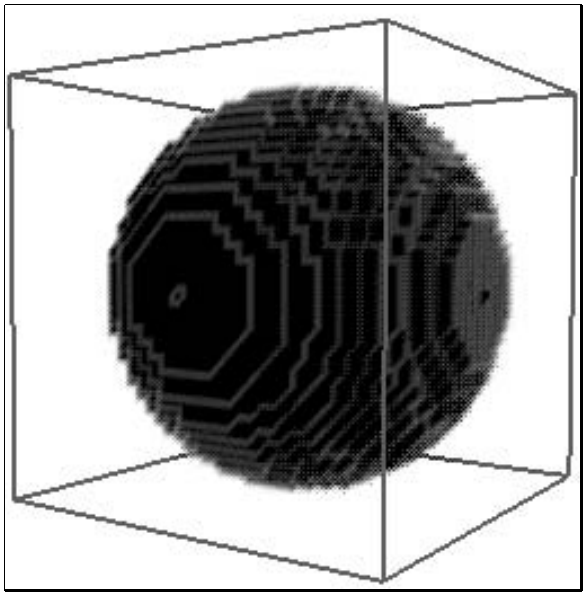


(b)

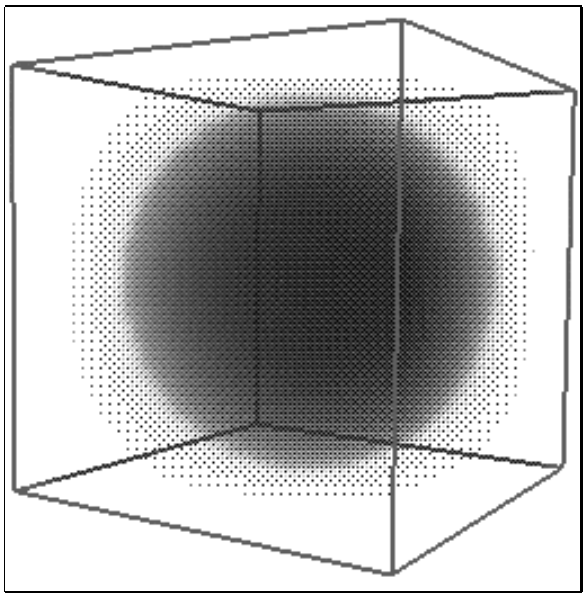


(c)

Figure 5.8: Smoothing in Two Dimensions



(a)



(b)

Figure 5.9: Smoothing of a Ball Shape: (a) input array; (b) output array

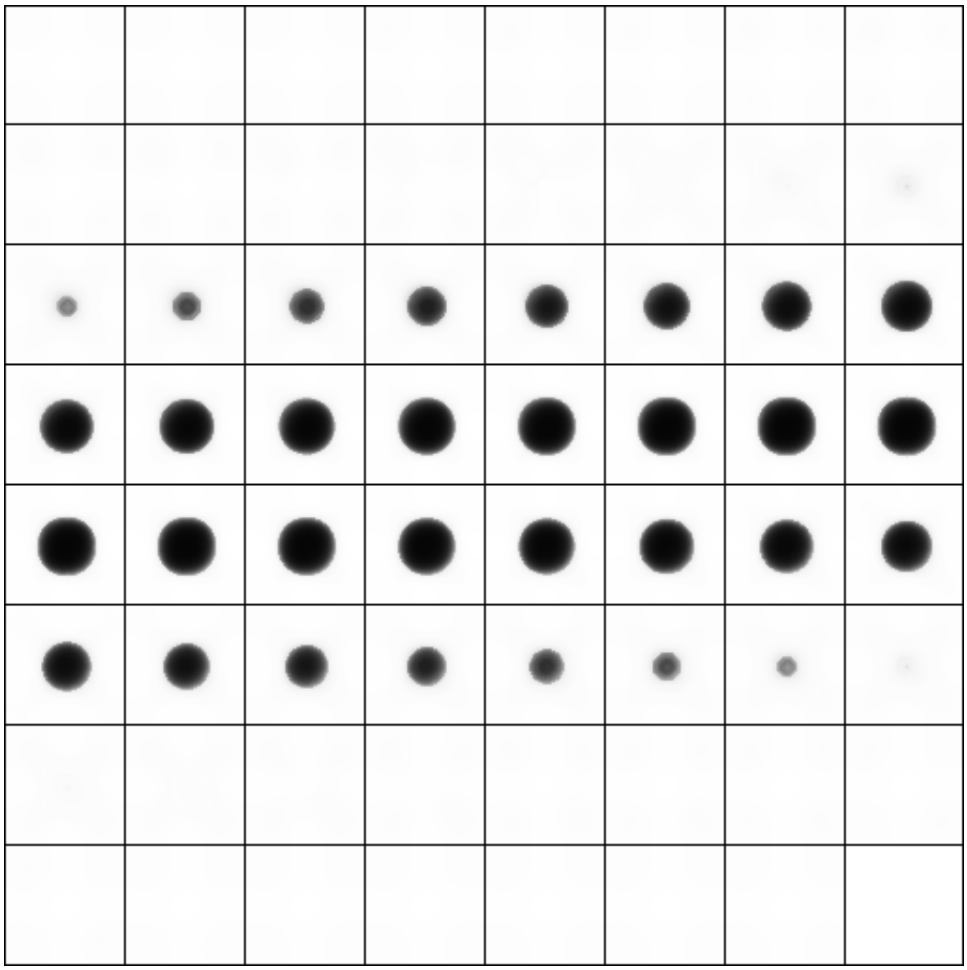
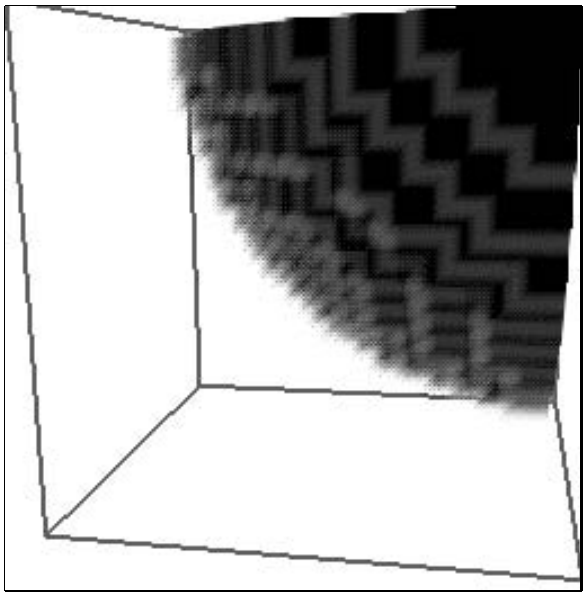
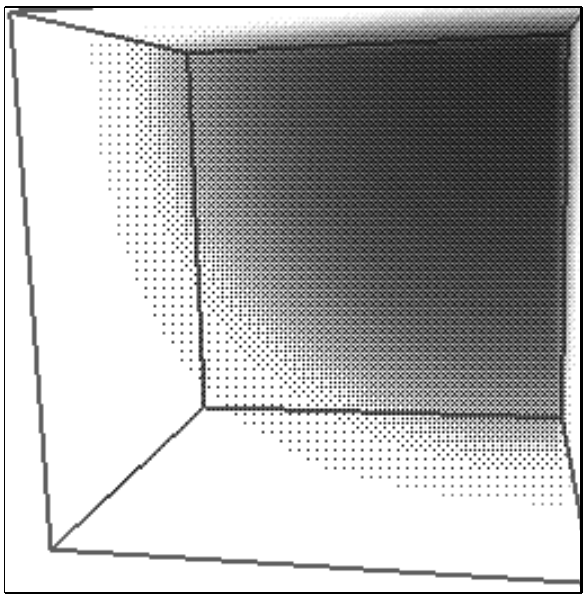


Figure 5.10: Slices of the 3D Array



(a)



(b)

Figure 5.11: Detail of the Input and Output Arrays

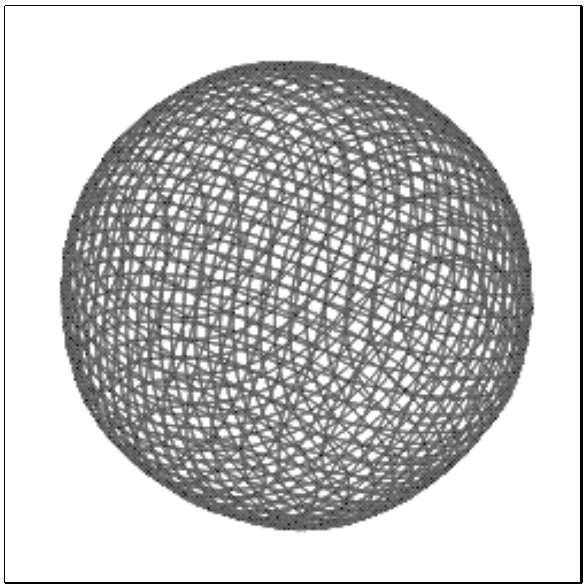


Figure 5.12: Wireframe of the Polygonal Isosurface Approximation

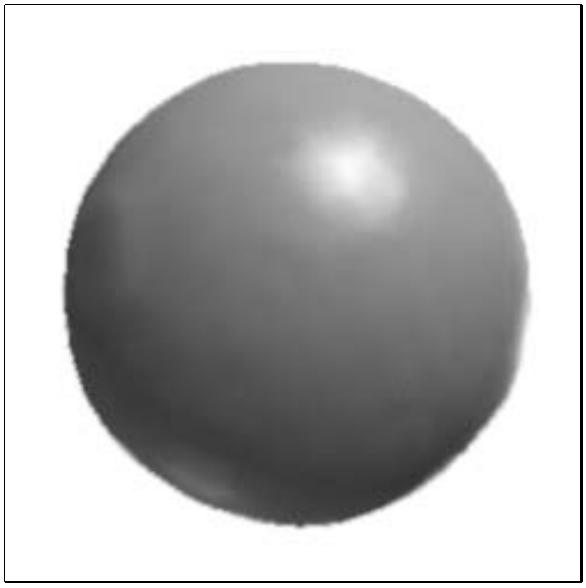
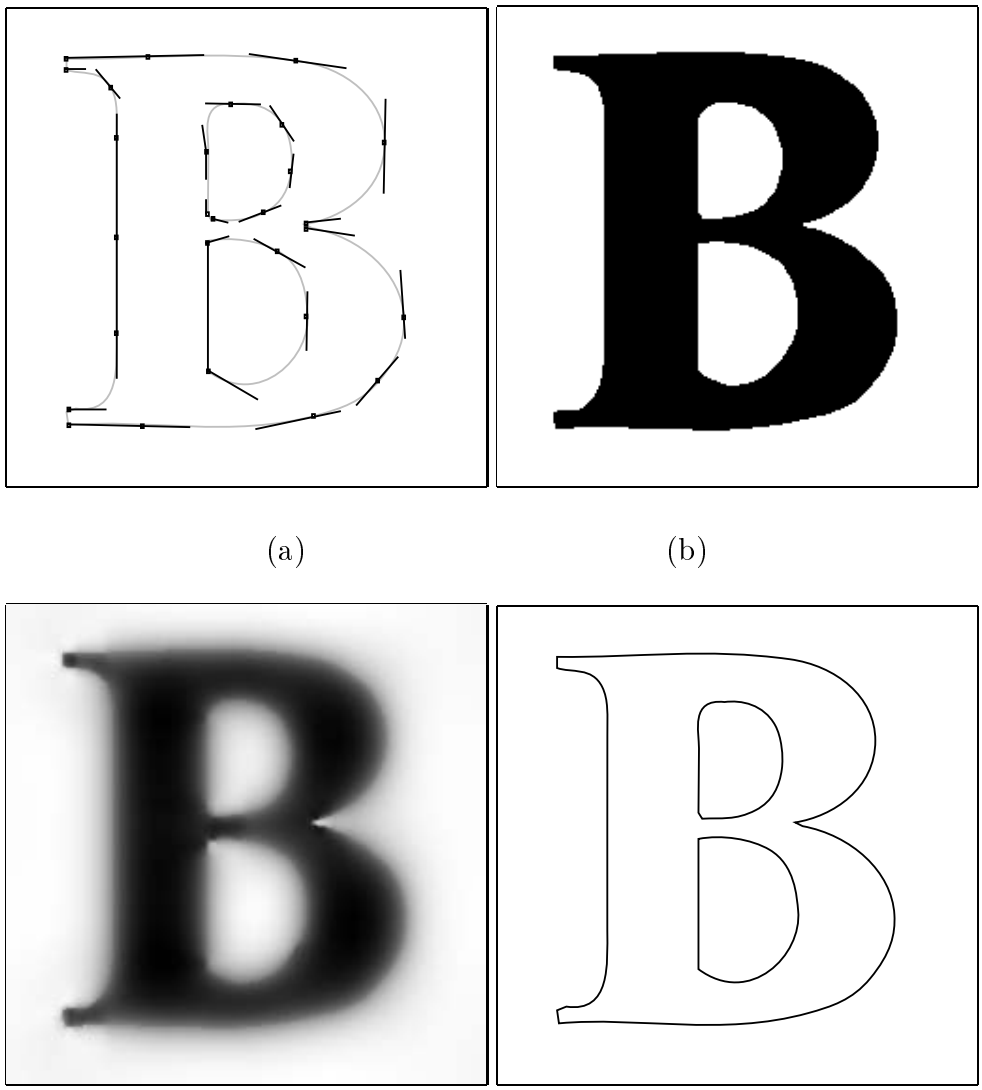


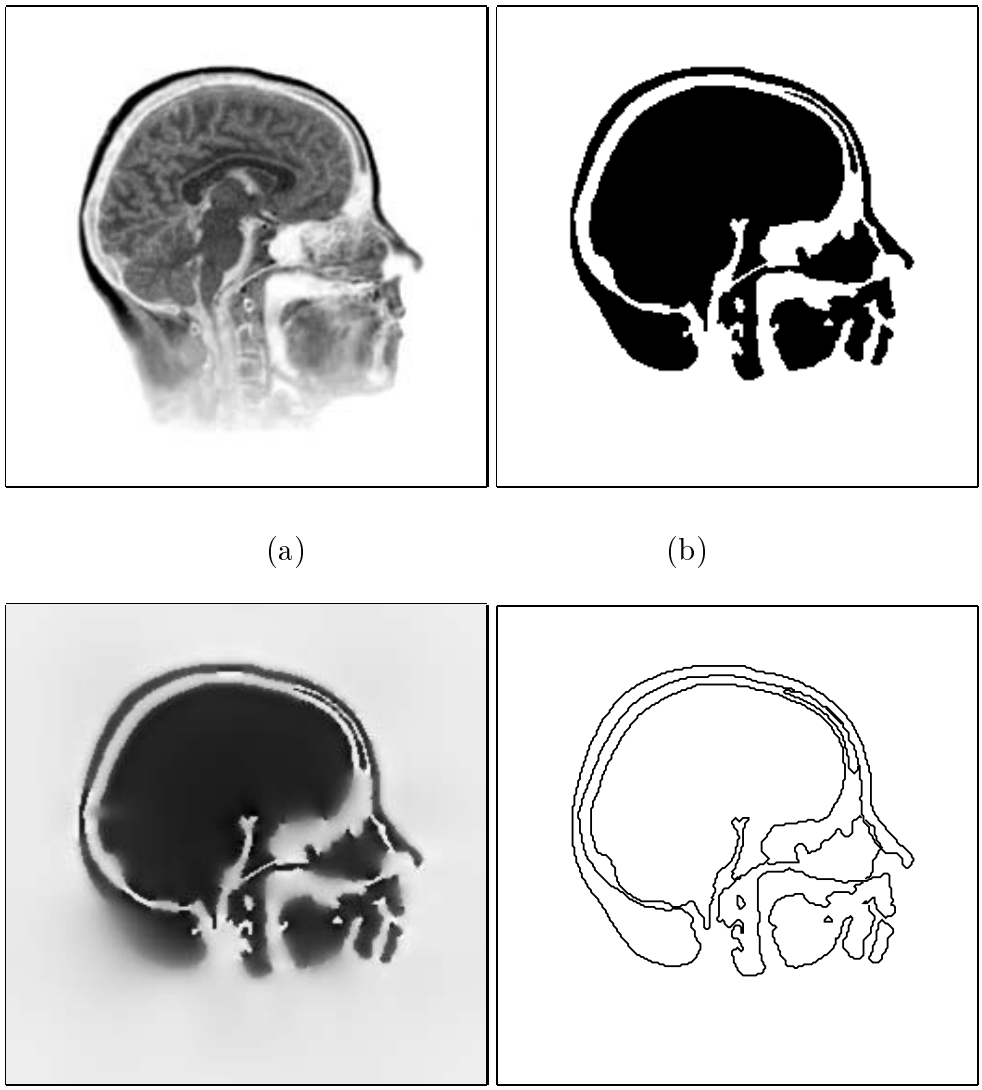
Figure 5.13: Shaded Image of the Polygonal Isosurface Approximation



(a) (b)

(c) (d)

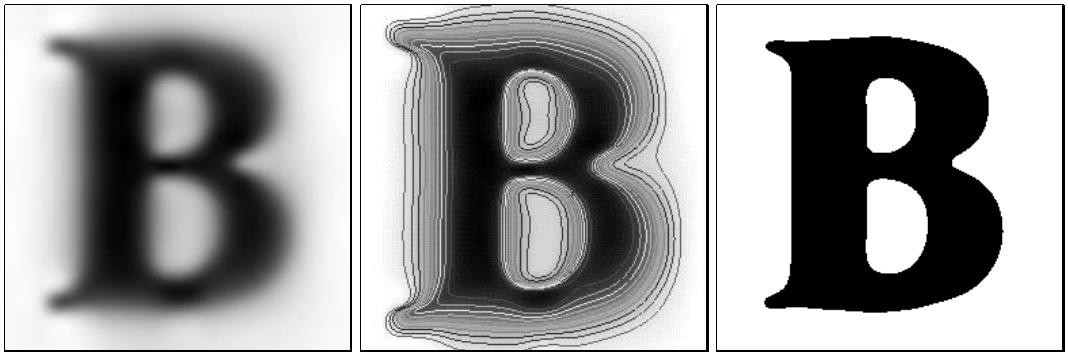
Figure 5.14: Conversion from Parametric to Implicit



(a) (b)

(c) (d)

Figure 5.15: Modification of an Implicit Function



(a) (b) (c)

Figure 5.16: Tubular Neighborhood with Limits

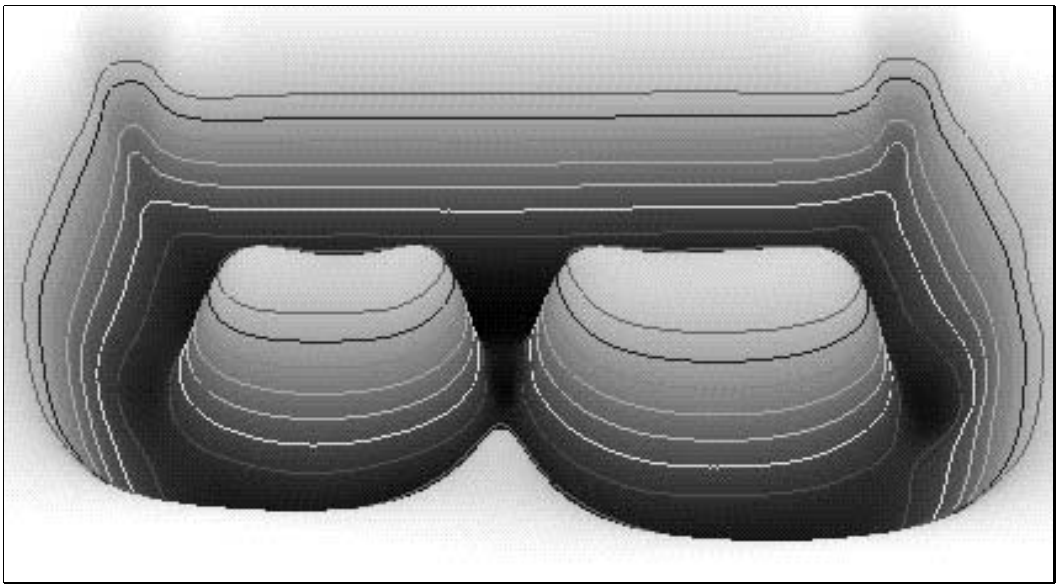
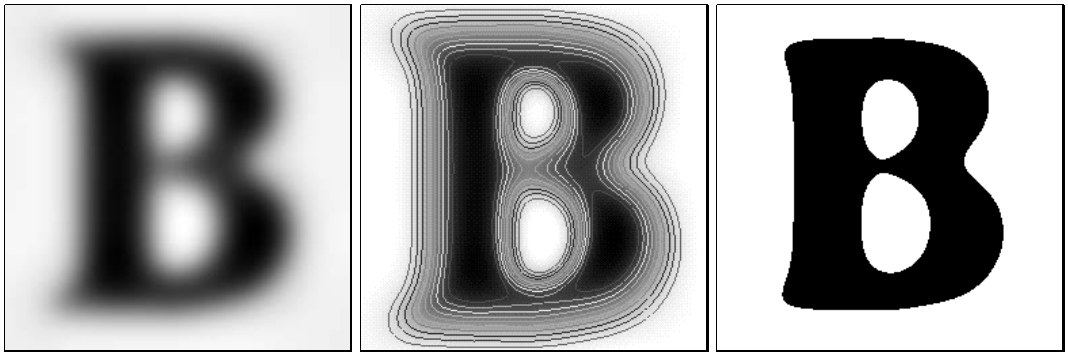


Figure 5.17: Implicit Function as a Height Surface



(a) (b) (c)

Figure 5.18: Uniform Tubular Neighborhood

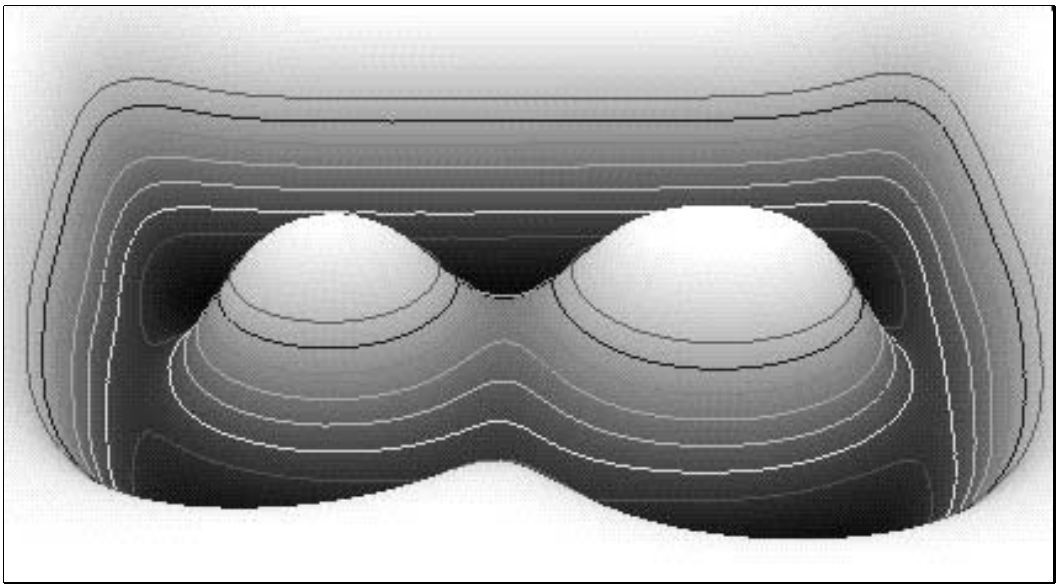


Figure 5.19: Implicit Function as a Height Surface

Chapter 6

Functional Decomposition

This chapter develops a method to produce a piecewise representation of implicit objects based on a multiscale decomposition of the implicit function.

We create a model that is adapted to the local variations of the implicit function at multiple scales. This model is described in terms of simple functions that are suitable for computation.

As mentioned in Chapter 4, decompositions of the implicit function for modeling purposes should reveal relevant aspects of the function, such as its variations. The Fourier transform describes the spectral behavior of a function, discriminating its frequency content. The Fourier series representation of periodic functions gives the best localization in terms of frequency, but its basis functions have infinite support.

It is also desirable that a functional decomposition results in a representation that exhibits spatial locality (an exception to this rule is the case where certain operations are performed more efficiently using some special representation without this property, such as the Fourier series (Totsuka and Levoy, 1993)).

The Fourier representation, unfortunately, does not satisfy the spatial locality requisite. In fact, spatial and frequency localization are incompatible requirements. So, if we want a representation that provides frequency information and has spatial locality, we have to settle for a compromise. Good localization in both space and frequency is achieved by the "short time Fourier transform" (Gabor, 1946), which employs a Gaussian windowed sinusoidal function and by the wavelet transform that uses scaled "small waves" (Daubechies, 1992).

While these decompositions provide suitable representations for image processing, they are not so effective for computer graphics. This will become clear in this chapter where we investigate a functional decomposition that is related to the wavelet transform and is adequate for modeling with implicit surfaces.

6.1 Multiscale Decompositions

The decomposition of a function in terms of elements at multiple scales provides a representation that reflects the function behavior over neighborhoods of variable size. It is a hierarchical structure which is adapted to the function variations.

6.1.1 Definition

A multiscale decomposition is a description based on a set of functions localized both in space and scale. Under this representation a function f is expressed as a linear expansion over this set.

A suitable family of functions for this purpose can be generated by scaling and translating a single function $\phi(x) \in L^2(\mathbb{R})$. A member of this family is denoted by ϕ_γ, where the index $\gamma = (s, u)$ specifies the scaling parameter s and the translation parameter u. Depending upon the properties of ϕ and its parametrization, different types of decompositions are produced.

Here, we assume that ϕ is a scaling function as defined in Appendix B. We restrict the parameters respectively to dyadic scales $s = 2^j$, $j \in \mathbb{Z}$ and integer translations $u = k$, $k \in \mathbb{Z}$. Thus, the family ϕ_γ is

$$\phi_\gamma(x) = \frac{1}{\sqrt{2^j}} \phi(\frac{x}{2^j} - k).$$

The scaling function ϕ is associated with a multiresolution analysis of $L^2(\mathbb{R})$ consisting of a sequence of nested spaces V_j

$$\cdots V_{-1} \subset V_0 \subset V_1 \cdots$$

whose union is dense in $L^2(\mathbb{R})$ and whose intersection is the null space. The collection of functions $\{\phi_{0;k} : k \in \mathbb{Z}\}$ forms a basis of the approximation space V_0 and all spaces in the sequence are scaled versions of this reference space V_0 (i.e. each space V_j has a "natural" scale 2^j).

In the framework of multiresolution analysis, a function f is represented by its approximations simultaneously at all these scales (See Appendix B). Such a structure is composed of several versions of the function f that are computed by projecting f onto the spaces V_j.

In contrast to the above scheme, the multiscale decomposition of a function f is a linear expansion into a countable subset of elements $(\phi_{\gamma_i}(x))_{i \in \mathbb{N}}$, with $\gamma_i = (j, k)$, from the family ϕ_{γ_i}, such that

$$f(x) = \sum_{i=0}^{\infty} a_i \phi_{\gamma_i}(x),$$

where γ_i is an element of the index set $\Gamma = \mathbb{Z}^+ \times \mathbb{Z}$.

The family of functions ϕ_γ plays the role of a dictionary $\mathcal{D} = (\phi_\gamma)_{\gamma \in \Gamma}$ (see Appendix A), from which a subset of elements ϕ_{γ_i} is selected for a multiscale representation.

The coefficients a_i are computed by orthogonal projection on the duals of the selected vectors ϕ_{γ_i} of the dictionary $\mathcal{D}$.

$$a_i = < f, \widetilde{\phi}_{\gamma_i} >$$

Intuitively, this decomposition indicates the features of f that "belong" to each scale 2^j.

A multiscale representation is given by the list of coefficients a_i, together with the corresponding indices $\gamma_i = (j_i, k_i)$ in the dictionary $\mathcal{D}$.

In practice, we use only a finite number of elements from $\mathcal{D}$. For this reason, it is important to obtain a sequence of decompositions with increasing number of elements which converges to the function f.

$$\|f - \sum_{i=0}^{m} a_i \phi_{\gamma_i}\| \leq \epsilon \|f\|$$

This provides a mechanism to approximate f with the desired precision ϵ.

The definition of a multiscale decomposition for n-dimensional functions $f : \mathbb{R}^n \rightarrow \mathbb{R}$ in $L^2(\mathbb{R}^n)$ is a direct extension of the scheme above.

6.1.2 Analysis of the Decomposition

The existence of a multiscale decomposition is simple to prove. It can be shown using a telescoping argument.

Denote by $P_j : L^2(\mathbb{R}) \rightarrow V_j$ the orthogonal projection onto V_j. The properties of the hierarchy of spaces V_j guarantee that:

1. $\lim_{j\rightarrow\infty} P_j f = f$ for all $f \in L^2(\mathbb{R})$, since $\overline{\cup V_j} = L^2(\mathbb{R})$
2. if $f \in V_l$ then f is also in V_j, for all $j > l$.

Condition (1) ensures that every function has a trivial decomposition.

Condition (2) implies that a function $f \in V_l$ can be expressed in the basis of any one of the approximation spaces V_j, $j > l$, which contain V_l. It is just the projection on V_j, $f = P_j f$ (there is no loss of information in this projection if $j > l$).

Conversely, if a function $f \in V_0$ has components g_j in the spaces V_j, $j \geq 0$, then f can be written as as a linear combination of these functions at scales 2^j

$$f = \sum_j g_j = \sum_j \sum_i a_{i,j} \phi_{2^j,k_i}(x).$$

We will see that is always possible to find such components g_j by projecting f on the spaces V_j. Note that in this case, there is loss of information in the projections and the functions g_j have to reflect that.

We stated that a function $f \in V_0$ can be represented by the sum of its components g_j at scales 2^j, $j < 0$, but we have not specified how to compute them. The problem is to construct a multiscale decomposition of f, such that $f = \sum_j g_j$, and it generates a suitable representation of f. (This discussion will be postponed to the next section.)

The dictionary $\mathcal{D}$ is very redundant. This is evident from the nested structure of the spaces V_j. A direct implication of the redundancy of $\mathcal{D}$ is that the multiscale

decomposition is not unique in general. This can be demonstrated in the following way:

A function $g \in V_l$ may be also expressed in terms of the basis of any one of the spaces V_j, $j > l$, without any loss of information, because V_l is also contained in V_j.

It follows that, if $g \in V_l$ then $P_m g = P_n g$ for $m, n > l$.

So, it is easy to see that any function $f \in L^2(\mathbb{R})$ can have infinitely many multiscale representations simply by adding and subtracting different projections, $P_m g$ and $P_n g$ of the same function $g \in V_l$, with $l < m, n$.

Two important issues arise as a consequence of the non-uniqueness of the multiscale representation:

- Which is the best multiscale decomposition of a function? What criteria should be used to characterize it?
- How to select the elements in $\mathcal{D}$ optimally? How to compute efficiently such a decomposition?

6.1.3 Optimal Decompositions

A quantitative or a qualitative criterion could be used to define the optimality of a multiscale decomposition. In the first case, the best representation would be the smallest, i.e. a linear expansion of f with the minimum number of elements from $\mathcal{D}$. In the second case, the best decomposition would be the most adapted to the function f, i.e. the one whose elements would identify features of interest in f. In practice, these criteria are used together. Also, good decompositions usually satisfy both of them to some extent.

6.1.4 Decomposition Strategies

There are two main strategies to generate a multiscale decomposition:

(a) Subdivide the spaces V_j defining a partition W_j whose direct sum is equal $L^2(\mathbb{R})$.

(b) Refine the components f_j belonging the spaces V_j such that $f = \sum_j f_j$

The difference between these two schemes is that while one deals with the structure of the approximating spaces V_j, the other deals with the components f_j. Scheme (a) is independent of f and more general. Scheme (b) is adapted to f and can be more effective.

In the next section we will discuss in detail the first one of these two multiscale decomposition approaches and methods to compute it based on the wavelet transform.

6.2 Wavelet Based Methods

Orthogonal wavelets generate a direct sum decomposition of $L^2(\mathbb{R})$ and are intimately related to multiresolution analysis. They provide a rigorous mathematical framework that can be used to derive multiscale decomposition methods.

The wavelet representation is very effective in describing the variations of a function at different scales. This makes it attractive from the standpoint of image processing, such as edge detection. The main reason for this is that the wavelet is designed to detect changes of the function value at different scales. What it really encodes is the location and scale of these transitions.

From the point of view of graphics applications, a representation in terms of the wavelet coefficients is not so desirable. We are looking for a multiscale representation that is constructive, compact, and expressible in terms of a simple function. Intuitively, we want a description of a function as a summation of "blobs" of different sizes.

This is exactly what is given by the multiscale decomposition. In that sense, we want a representation in terms of a scaling function associated with a wavelet function.

6.2.1 The Wavelet Transform

We saw in Appendix B that the wavelet spaces W_j are defined as the difference spaces between two consecutive approximation spaces V_{j+1} and V_j of a multiresolution analysis. The W_j is the orthogonal complement of V_j in V_{j+1}

$$W_j \oplus V_j = V_{j+1}, \qquad W_j \perp V_j.$$

$L^2(\mathbb{R})$ is naturally decomposed into

$$\bigoplus_j W_j = L^2(\mathbb{R}).$$

The connection of wavelets with multiresolution analysis is the key to the efficient computation of the wavelet decomposition of a function.

The basic process decomposes f recursively from fine to coarse resolutions, exploiting the hierarchy of spaces V_j and W_j.

The core of the algorithm takes a function f_j at resolution 2^j and splits it into two parts, through orthogonal projection onto V_{j-1} and W_{j-1} (a lower resolution space and a detail space). The recursion step is performed by replacing V_j with V_{j-1} and applying the above procedure again.

The algorithm can be illustrated by the graph

$$\begin{array}{ccccccccc} V_j & \rightarrow & V_{j-1} & \rightarrow & V_{j-2} & \rightarrow & V_{j-3} & \cdots \\ & \searrow & & \searrow & & \searrow & & \\ & & W_{j-1} & & W_{j-2} & & W_{j-3} & \cdots \end{array}$$

Note that the spaces V_j are just intermediate elements in the method. We cannot use the wavelet representation directly as a multiscale representation because its coefficients are relative to a wavelet basis, not relative to the scaling function.

The next figures illustrate the wavelet transform in one dimension. Figure 6.1 shows the graph of a function $f : \mathbb{R} \to \mathbb{R}$.

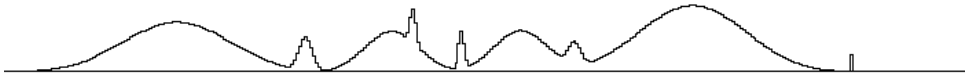


Figure 6.1: One-dimensional Function

Figure 6.2 shows the approximations at different scales of the function in Figure 6.1. It actually shows the scaling function coefficients corresponding to each approximation.

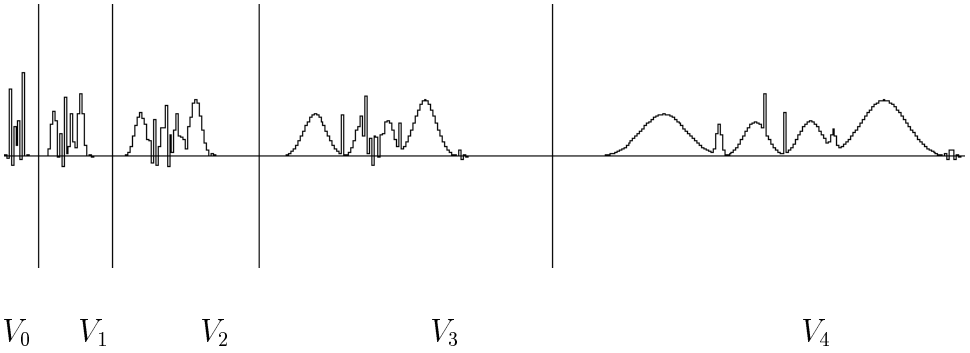


Figure 6.2: Approximations at Different Scales of a Function

Figure 6.3 is a plot of the wavelet coefficients of the function shown in Figure 6.1.

6.2.2 The Laplacian Decomposition

One idea for creating the multiscale representation takes advantage the wavelet transform algorithm. The projection of f onto the spaces V_j gives the multiresolution analysis of f. Unfortunately, we also cannot use this representation directly. The scaling function coefficients represent approximations of the function f at each resolution 2^j.

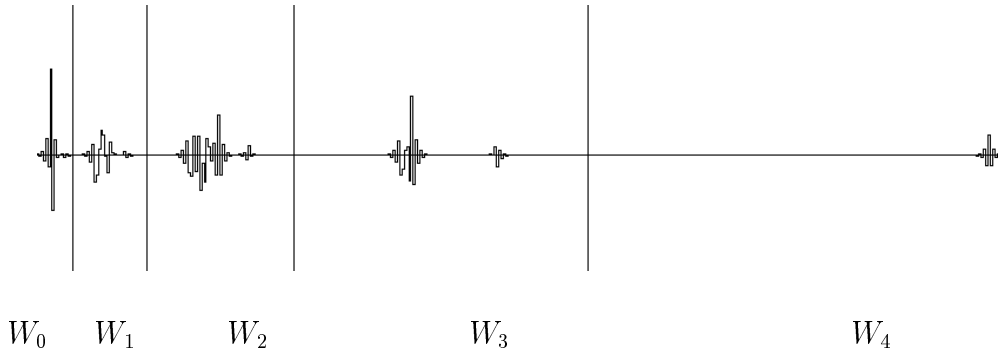


Figure 6.3: Wavelet Decomposition of a Function

It is clear that, as a whole, they do not constitute a linear expansion of f in terms of the scaling function ϕ.

Instead of a multiresolution analysis, we would like to obtain a representation that is a multiscale decomposition of the function f. Such a representation can be constructed from the wavelet decomposition.

Observe that $V_j = V_{j-1} \oplus W_{j-1}$ implies that $W_{j-1} \subset V_j$. Therefore, W_{j-1} can be represented in terms of a basis of V_j without any loss of information. All we need to do in order to produce the desired representation is to project the wavelet coefficients in the subspaces W_{j-1} back to the subspaces V_j.

This method produces a representation that is multiscale and is given in terms of the scaling function. This representation is essentially equivalent to the Laplacian pyramid (Burt, 1983).

Figure 6.4 is a plot of the scaling coefficients of the Laplacian decomposition of the function in Figure 6.1.

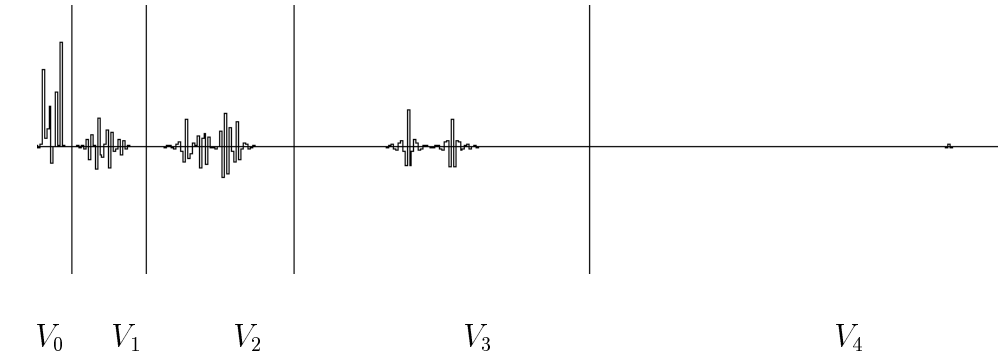


Figure 6.4: Laplacian decomposition of a function

Notice that although the function in Figure 6.1 is positive everywhere, the multiscale decomposition contains negative coefficients. The reason is that the coefficients

reflect the difference in information between approximations of the function at pairs of resolutions 2^j and 2^{j-1}.

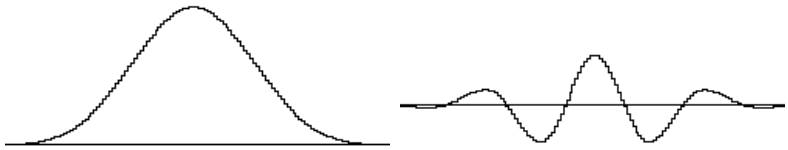
6.2.3 The B-Spline as a Scaling Function

So far we have discussed the multiscale decomposition without defining the scaling function associated with it. The B-spline, introduced in Appendix C, constitutes a suitable scaling function for our purposes.

As we have seen in that appendix, the B-spline has several desirable properties: symmetry, smoothness, compact support, good localization in space and scale, a simple analytical form in both spatial and frequency domains, and efficient implementation.

The only disadvantage is that the B-spline does not generate an orthogonal basis. But it is possible to construct a dual function, which together with the B-spline, defines a pair $(\phi, \widetilde{\phi})$ that is bi-orthogonal and can be used in the multiscale decomposition method. This is described in Appendices B and C.

We will employ in the examples the cubic B-spline. Figure 6.5 shows a plot of the cubic B-spline scaling and wavelet functions.



(a) (b)

Figure 6.5: Cubic B-spline (a) scaling function (b) wavelet function

6.3 The Multiscale Representation

The results in the previous section allow us to create an implicit piecewise representation based on a multiscale decomposition of the implicit function f. This representation is adapted to the variations of f at different scales.

As we mentioned in section 6.1, the multiscale representation contains the coefficients a_i of the multiscale decomposition. The indices $\gamma_i = (j_i, k_i)$ relate these coefficients to elements ϕ_{γ_i} of the dictionary $\mathcal{D}$.

In order to produce the representation, we compute the coefficients a_i of the B-spline multiscale decomposition and convert them to a suitable data structure.

6.3.1 Computing the Representation

There are two equivalent procedures to compute the Laplacian decomposition of a function $f \in V_0$: a modified wavelet transform, and the Laplacian transform.

In both cases, we assume that we have computed the coefficients $\{c\}_0 =< f, \widetilde{\phi}_{1,k} >$ of the representation of f in the basis of V_0, such that f is written as

$$f(x) = \sum_k < f, \widetilde{\phi}_{1,k} > \phi_{1,k}(x).$$

In the modified wavelet transform, as suggested in the previous section, we apply the wavelet decomposition to the coefficient sequence $\{c\}_0$ obtaining the wavelet coefficients $\{d\}_j$, $j = -1, \ldots, -n$, and then for each scale 2^j, we project the wavelet coefficients $\{d\}_j \in W_j$ back to V_j by applying the wavelet reconstruction from level j to $j+1$. This procedure is illustrated in the diagram below.

$$\begin{array}{ccccccc}
\{c\}_j & \overset{\widetilde{H}}{\rightarrow} & \{c\}_{j-1} & \overset{\widetilde{H}}{\rightarrow} & \{c\}_{j-2} & \cdots \\
 & \overset{\widetilde{G}}{\searrow} & & \overset{\widetilde{G}}{\searrow} & & \overset{\widetilde{G}}{\searrow} \\
\{d\}_j & & \{d\}_{j-1} & & \{d\}_{j-2} & \cdots \\
 & \overset{G}{\swarrow} & & \overset{G}{\swarrow} & & \overset{G}{\swarrow} \\
\{a\}_j & & \{a\}_{j-1} & & \{a\}_{j-2} & \cdots
\end{array}$$

where H, G, $\widetilde{H}$ and $\widetilde{G}$ are the pairs of discrete filters associated with the B-spline scaling function, the B-spline wavelet and their duals (see Appendix C).

In the Laplacian transform, we exploit the fact that, since $W_{j-1} = V_j \ominus V_{j-1}$, the coefficients $\{a_j\}$ of the multiscale decomposition can be computed by subtracting the coefficients of the approximations at V_j and V_{j-1}. This procedure is illustrated in the diagram below.

$$\begin{array}{ccccccc}
\{c\}_j & \overset{\widetilde{H}}{\rightarrow} & \{c\}_{j-1} & \overset{\widetilde{H}}{\rightarrow} & \{c\}_{j-2} & \cdots \\
 & \overset{H}{\swarrow} & & \overset{H}{\swarrow} & & \overset{H}{\swarrow} \\
- & & - & & - & \\
\{b\}_j & & \{b\}_{j-1} & & \{b\}_{j-2} & \cdots \\
= & & = & & = & \\
\{a\}_j & & \{a\}_{j-1} & & \{a\}_{j-2} & \cdots
\end{array}$$

In one dimension, these two procedures require roughly the same amount of computation. In higher dimensions, the Laplacian transform is more efficient.

6.3.2 Data Structures

We can use three alternative data structures to encode the coefficients of the multiscale decomposition:

- A list of the coefficients $l = \{a_i, \gamma_i\}$.
- A pyramid data structure, $A_j = (a_{j,k})$.
- A spatial hash table $H = \{\uparrow a_i, \gamma_i\}$.

The list structure is simply an enumeration of the coefficients of the multiscale decomposition a_i and the indices $\gamma_i = (s_i, u_i)$ corresponding to the functions $\phi_{2^j,k}$ of the dictionary $\mathcal{D}$.

The pyramid structure contains all coefficients associated with the basis functions $\phi_{(2^j,k)}$ of the approximating spaces V_j. This is essentially an enumeration of the coefficients corresponding to all elements in $\mathcal{D}$, such that the functions that are not in γ_i have a coefficient $a_{j,k} = 0$. This establishes a one-to-one correspondence between the elements of $\mathcal{D}$ and the coefficients of the decomposition, and eliminates the need for including the indices γ_i in the representation.

The coefficients a_i may also be associated with a spatial hash table in which each cell has pointers to the elements whose support is in the cell. This cell complex can be formed by any adaptive subdivision of space, such as an Octree or a BSP tree (See Appendix D).

The list structure is the most compact, but it requires extra processing to find the elements that should be evaluated at a given point. The pyramid structure has the advantage of simplicity and direct access but, since the multiscale decomposition is usually sparse, it has the disadvantage of using more space than necessary. The hash table structure offers a good compromise between access time and space.

In practice, the choice of a particular data structure will be dictated by the characteristics of the implicit function, as well as the requirements of the application. The list structure is the best for models with a small number of coefficients. The pyramid structure is the best for models with uniform spatial complexity. The hash table structure is the best for models with non-uniform spatial complexity.

6.3.3 Properties of the Multiscale Representation

The multiscale representation has the following properties:

- Hierarchical
- Structured

The multiscale representation describes a function f by its components $f_j \in V_j$. This representation is hierarchical because the spaces V_j form a ladder of approximation spaces. It is structured because the basis functions $\phi_{j,k}$ of V_j are located in a regular rectangular grid.

These two properties are the key for efficient computation with piecewise implicit models.

6.4 Applications

The multiscale decomposition can be applied in the solution of many problems in modeling and graphics. It has the potential to be the basis for the development of efficient computational methods.

The main application of the B-spline multiscale decomposition is in the conversion of volumetric data to an implicit analytic description. Once this representation is obtained, it can be used in other graphics applications.

In this section, we give examples of the conversion from volumetric to implicit and indicate some potential applications, such as models with variable level of detail, surface rendering and volume visualization.

6.4.1 Conversion of Volumetric to Implicit Descriptions

The multiscale decomposition can be applied to the conversion of volumetric to implicit descriptions. The method takes as input a n-dimensional array of discrete samples and generates a piecewise implicit representation in terms of the B-spline scaling basis functions.

The implicit objects in the following examples are given in the form of 2D sample array. This volumetric representation is converted to the B-spline multiscale representation.

In the figures, the implicit function is shown as an image, the boundary of the object as a curve in the plane, and the B-spline pyramid is given a set of circles corresponding to the support of B-spline basis functions with non-zero coefficients.

Example 6.1 (Superquadric) The object is a superquadric rectangular shape with rounded corners. It is defined by a very smooth implicit function. The B-spline pyramid representation has only one level and a total of 36 non-zero coefficients. Figure 6.6(a) shows the implicit function $F(x,y) = \left(\frac{x}{a}^{\frac{2}{e1}} + \frac{y}{b}^{\frac{2}{e1}}\right)^{e1}$. Figure 6.6(b) shows the support (represented as a circle) of the B-spline functions in the pyramid with non-zero coefficients. Figure 6.6(c) shows the boundary curve $F^{-1}(1)$ generated from the original function. Figure 6.6(d) shows the boundary curve generated from an approximation of F obtained using the B-spline pyramid. Figure 6.7 shows the B-spline pyramid.

Example 6.2 (Koch Snowflake) The object is the Koch snowflake, a deterministic fractal object generated by a recursive procedure. The implicit function is discontinuous at the boundary of the object. The B-spline representation is a 6-level pyramid and has non-negligible coefficients at all levels. Level 0 has 61 coefficients, level 1 has 158 coefficients, level 2 has 391 coefficients, level 3 has 910 coefficients, level 4 has 1962 coefficients, and level 5 has 3400 coefficients. Note that while the resolution increases as n^2, the number of relevant coefficients only doubles from one level to the next. This is because amount of detail is proportional to the perimeter of the curve.

Figure 6.8(a) shows the implicit function F. Figure 6.8(b) shows the footprints of the B-spline functions used to reconstruct F. Figure 6.8(c) shows the fractal curve generated from the original function and Figure 6.8(d) shows the curve generated from the B-spline representation. They are visually indistinguishable. Figure 6.9 shows the B-spline pyramid.

Example 6.3 (Free-Form Shape) The object is a free-form implicit shape created with a painting program. The density function was input as gray values with different size brushes. The object was designed to exhibit features in a wide range of scales. The B-spline representation is a 6 level pyramid with a total of 239 non-negligible coefficients. Level 0 has 45 coefficients, level 1 has 65 coefficients, level 2 has 45 coefficients, level 3 has 49 coefficients, level 4 has 33 coefficients, and level 5 has 2 coefficients. Figure 6.10(a) shows an image of its density function. Figure 6.10(b) shows the B-spline functions used in its representation. Figure 6.10(c) shows the boundary curve generated from the density function. Figure 6.10(d) shows the curve generated from the B-spline representation. Figure 6.11 shows the B-spline pyramid. Note how the B-spline functions at finer scales are concentrated in places where the object presents more detail.

In some applications, the interest is exclusively in the boundary of objects. In those cases, the B-spline representation can be used to carry only information contributing to the description of the boundary. Then, the implicit object becomes hollow and is represented by B-spline functions forming a "thick" shell along the boundary.

Example 6.4 (Diamond Shape) Figure 6.12(a) shows the B-spline representation of a solid object and Figure 6.12(b) its boundary. Figure 6.12(c) shows the B-spline functions in the representation whose support is intersected by the boundary curve. Figure 6.12(d) shows the reconstruction of the object using only the functions in Figure 6.12(c).

The next example shows the use of the method in three dimensions. The input is a 3D sample array that is converted to the B-spline multiscale representation.

Example 6.5 (Hypertexture Object) The object is a "noisy sphere", a procedural implicit shape. It is defined by functional composition of an object density

function with density modulation functions (Perlin and Hoffert, 1989). In this example, the object density function is of a soft sphere and the modulation function is a bandlimited noise function.

Figure 6.13 shows the volume density array generated by the hypertexture procedure mentioned above. Figure 6.14 shows the points in the volume corresponding to the boundary and the interior of the object. Figure 6.15 is a ray-traced image of the noisy sphere. It was produced by rendering the B-spline pyramid description of the data in Figure 6.13. Figure 6.16 shows the B-spline pyramid and Figure 6.17 shows one slice from each level enlarged. In these figures, the coefficients are depicted according to the following convention: zero is middle gray, negative is darker and positive is lighter. Note that almost all the information is contained in the bottom level of the pyramid, indicating that most of variations of the implicit function are at that scale.

6.4.2 Variable Level of Detail Models

The multiscale representation describes the model by its components at different scales. This makes it particularly effective to represent implicit objects at variable level of detail.

From Example 6.2, it is clear that a large number of B-spline functions are required to represent accurately the fractal object in Figure 6.8. The B-spline pyramid is very flexible in situations like this, because it allows the generation of approximations of the original object at different resolutions using fewer B-spline functions.

Example 6.6 (Koch Snowflake Approximations) Figures 6.18(a) to 6.18(d) are approximations of the original Koch curve using respectively four, three, two and one levels of the pyramid. Figure 6.9 shows the five levels of the B-spline pyramid used in Figure 6.8.

6.4.3 Ray Tracing and Soft Shadows

The multiscale representation is suitable for ray casting techniques. There are several ray-tracing algorithms for implicit surfaces given by polynomial functions (Tonnesen, 1989), (Wyvill and Trotman, 1990) that could be used to compute the ray intersection with the B-spline primitives. Similarly, the cone tracing algorithm (Amanatides, 1984) can be adapted to handle "fuzzy balls" producing antialiased images as well as soft shadows (Perlin and Zhu, 1990).

6.4.4 Volume Rendering

The multiscale representation is also well suited to volumetric rendering techniques. Direct projection methods, such as splatting, and volume integration methods, such as ray marching, can exploit the multiscale representation in various ways.

6.5 Summary

In this chapter we defined the notion of multiscale decomposition models and discussed the criteria to analyze such models. We identified the strategies to generate a multiscale decomposition and investigated wavelet based methods for producing this kind of description. We presented the wavelet and Laplacian transform algorithms to compute the biorthogonal B-spline pyramid, described alternatives of data structures for this representation and commented on its properties.

The wavelet and Laplacian transform have been used mainly in image processing applications, such as coding and compression (DeVore, Jawerth and Lucier, 1992) (Burt, 1983). Although we employ a multiscale representation which is equivalent to the Laplacian pyramid, the context in which it is applied is very different. In image processing, this type of description is used primarily for storage and transmission, while in geometric modeling, this description is used for computations, such as rendering. Consequently, the problems which have to be solved are of a different nature.

Wavelet based models of implicit surfaces and solids were proposed in (Muraki, 1993) and (Perlin and Zhu, 1990). Muraki's model is a direct extension of the orthogonal 2D wavelet decomposition to three dimensions. Because it uses a tensor product formulation, the representation is given in terms of 7 different wavelet functions, which complicates the computations with the model. Perlin's model uses only one directional spline wavelet, avoiding this problem. His method employs an empirical procedure to construct the representation and model is restricted to a particular level surface.

Our multiscale implicit model is based on the scaling function associated with a wavelet decomposition. It gives a description of the implicit function as a whole in terms of a single B-spline function. For this reason, it is more adequate for image synthesis computations than the wavelet models. Furthermore, this representation can be computed with a fast and exact algorithm developed from a solid theoretical foundation.

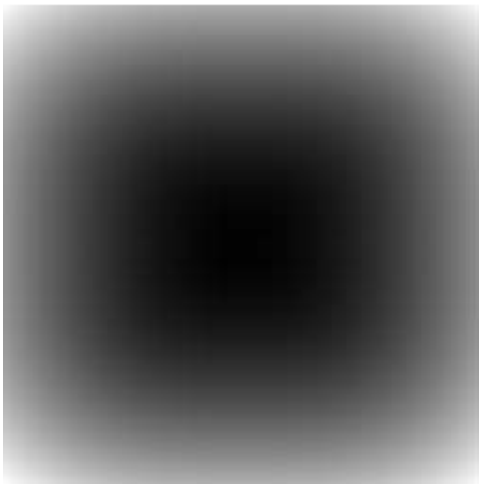
The multiscale decomposition model introduced in this chapter has several applications in the area of computer graphics. Some of these applications are the conversion from volumetric to piecewise implicit representations, variable level of detail models, rendering of implicit surfaces using ray tracing, and volume visualization.

Note: Unstructured Decompositions

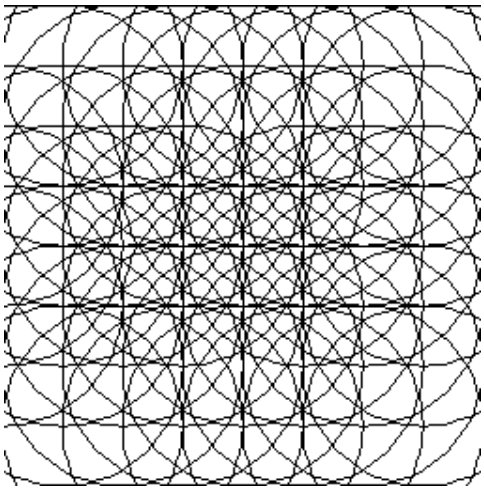
Structured multiscale decompositions have one property that may not be desirable in some situations. They are not invariant under fractional translations. This means that the representation of two identical shapes will be different if one is displaced relative to the other by a non-integer amount.

A possible approach to compute a non-structured multiscale decomposition that

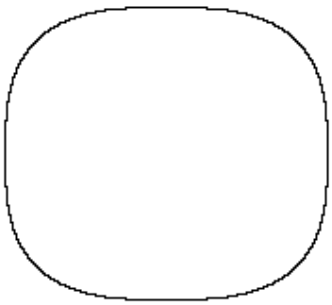
is translation invariant could be to employ a matching pursuit algorithm (Mallat and Zhang, 1993). This algorithm consists of an iterative procedure for minimizing the energy of the residual error between the input function and its linear expansion over a set of elements in the dictionary. In this case, the model would be formulated in terms of a dictionary consisting of functions that are the basis of "continuous" spaces in a dyadic scale sequence. We have already made some preliminary experiments with this type of model. The main problem we have encountered so far is related to the long computation times of the algorithm, due to slow convergence.



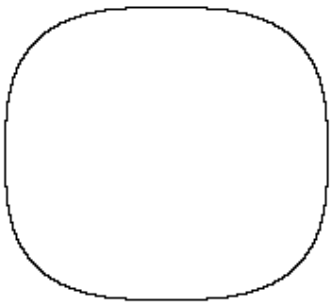
(a)
Density Function



(b)
B-spline Functions



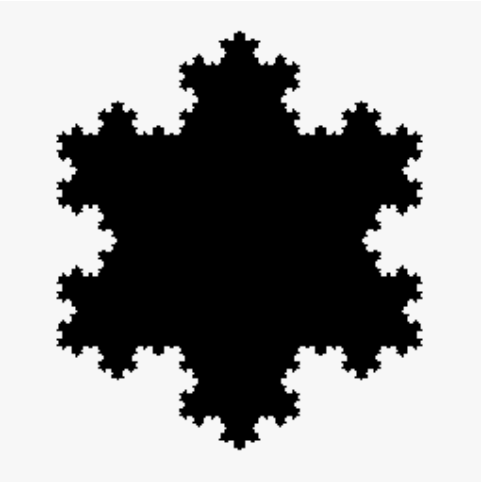
(c)
Boundary derived from (a)



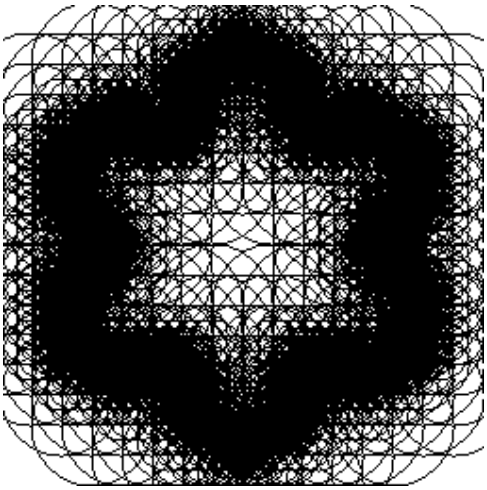
(d)
Boundary derived from (b)

Figure 6.6: Superquadric

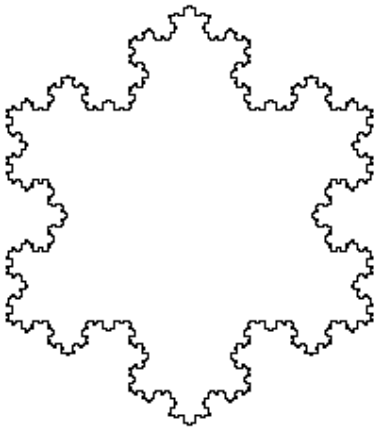
Figure 6.7: B-Spline Pyramid for the Superquadric



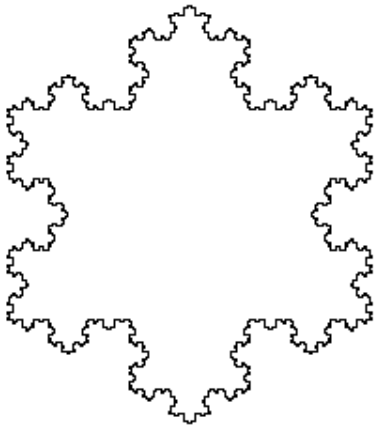
(a)
Density Function



(b)
B-spline Functions



(c)
Boundary derived from (a)



(d)
Boundary derived from (b)

Figure 6.8: Koch Snowflake

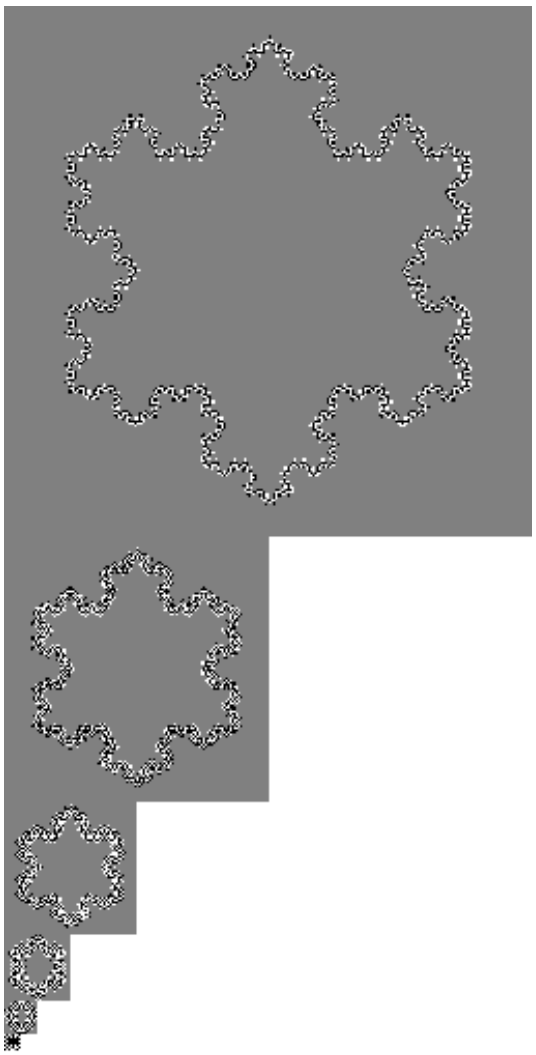
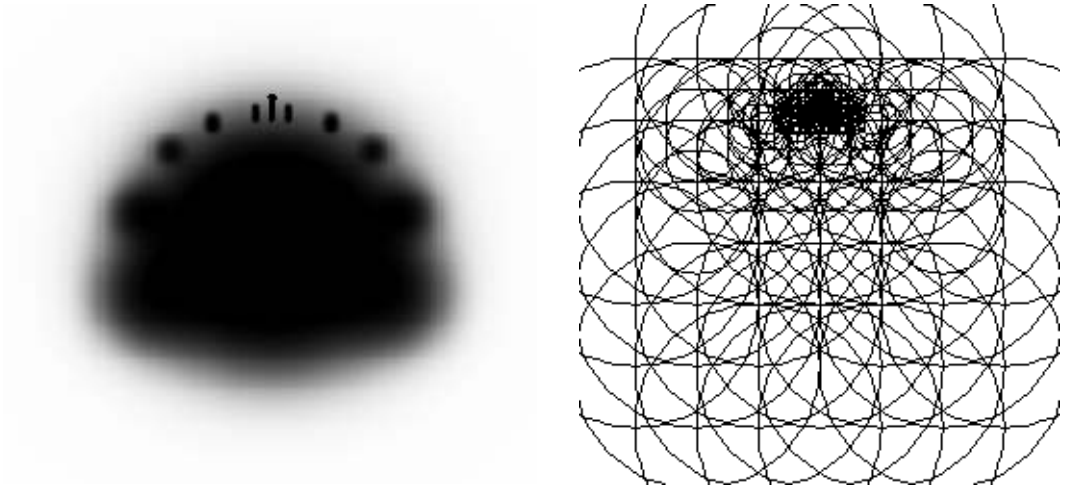
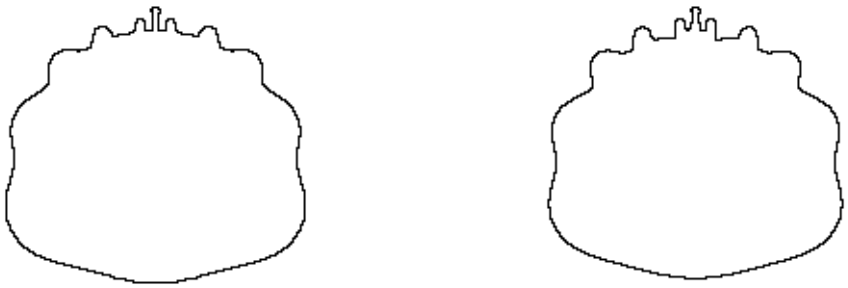


Figure 6.9: B-Spline Pyramid for the Koch Snowflake



(a)
Density Function

(b)
B-spline Functions



(c)
Boundary derived from (a)

(d)
Boundary derived from (b)

Figure 6.10: Free Form Shape

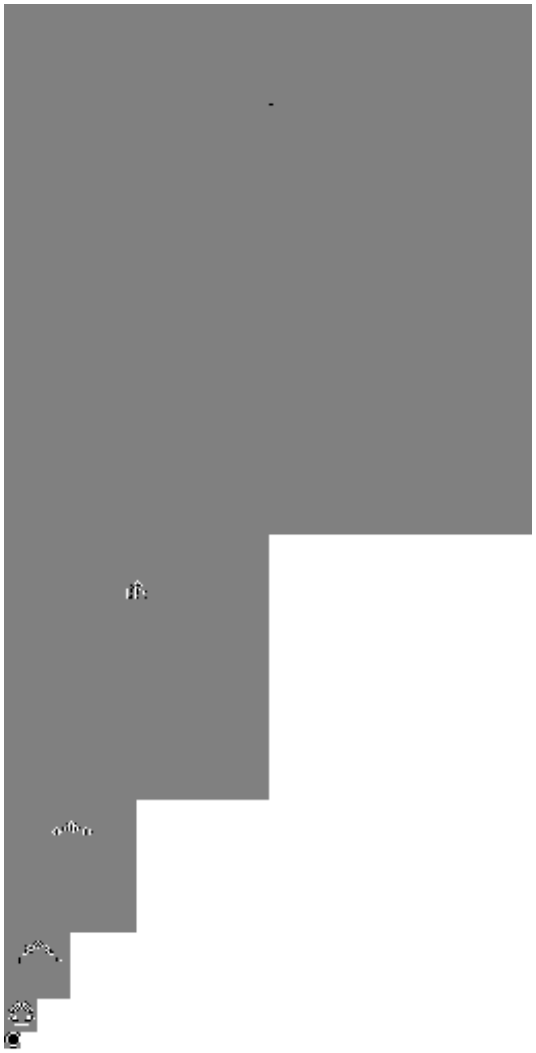
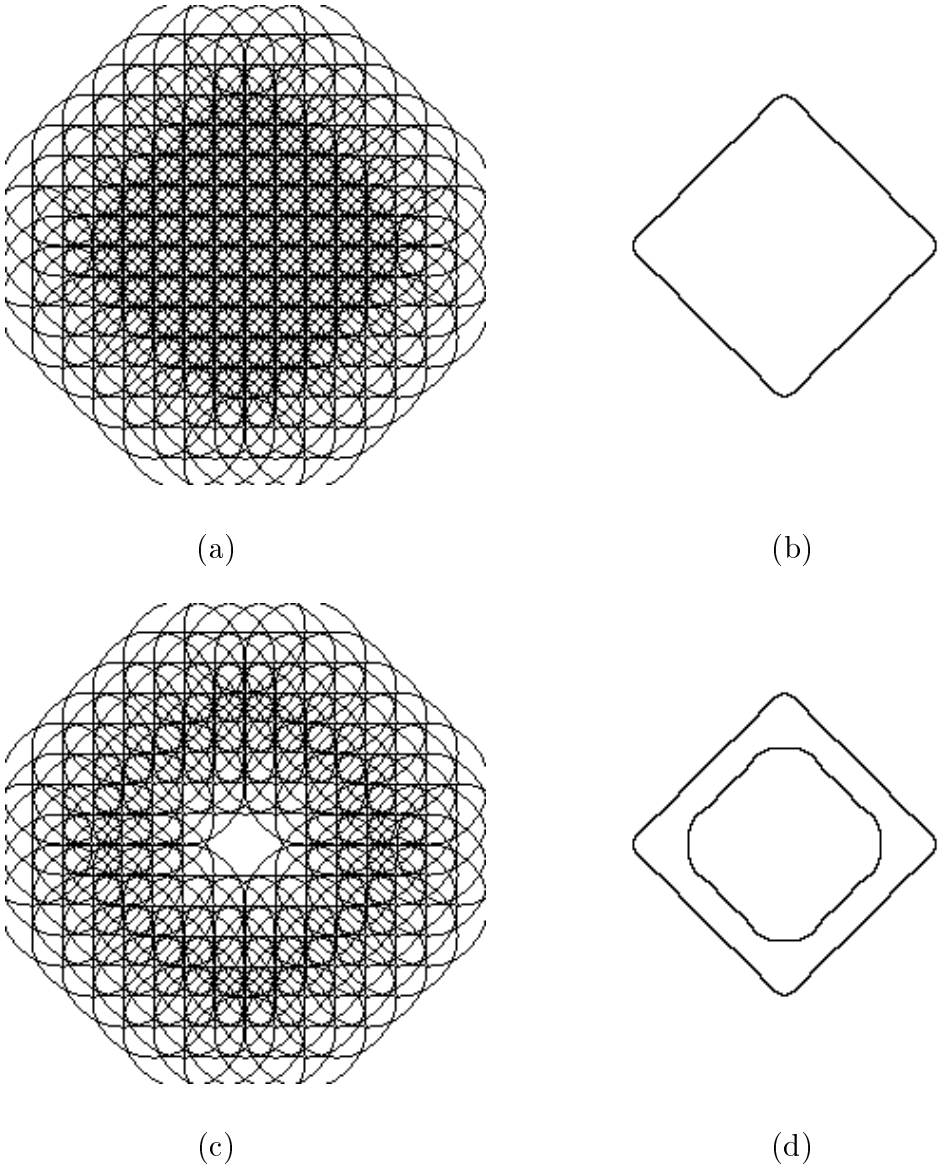


Figure 6.11: B-Spline Pyramid for the Free Form Shape



(a) (b)

(c) (d)

Figure 6.12: Hollow and Solid Objects

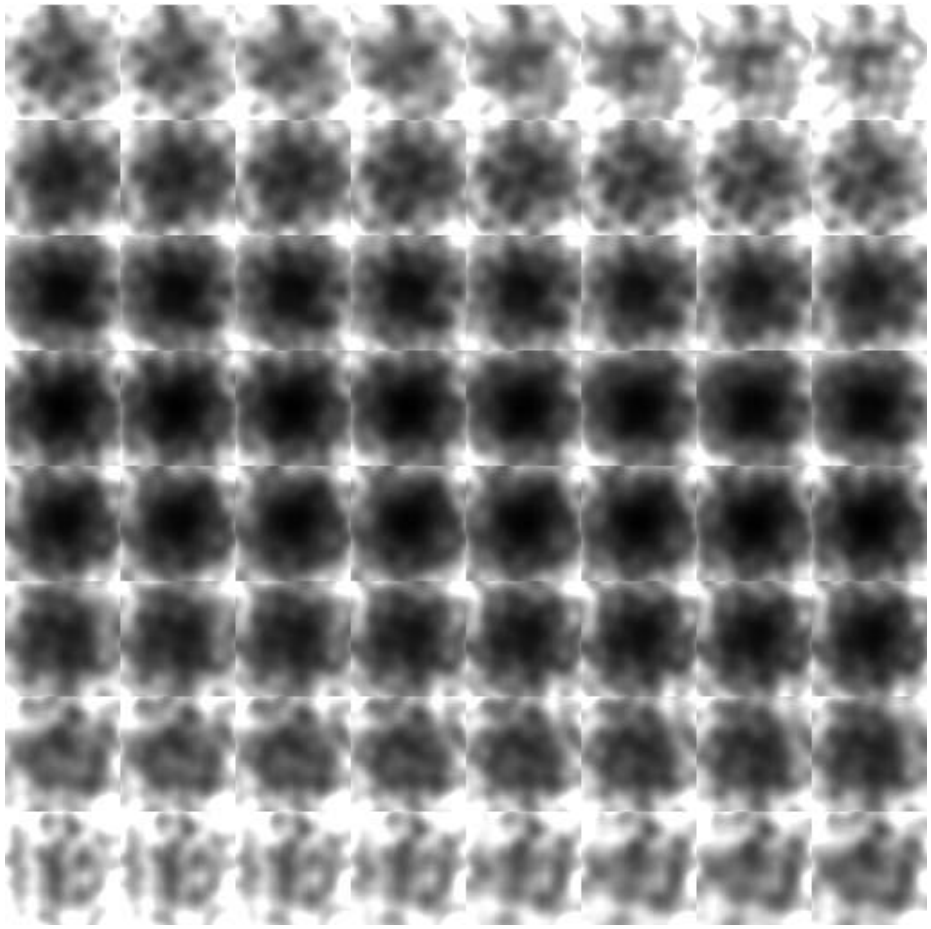


Figure 6.13: Slices of the Volume Density Function for the Noisy Sphere

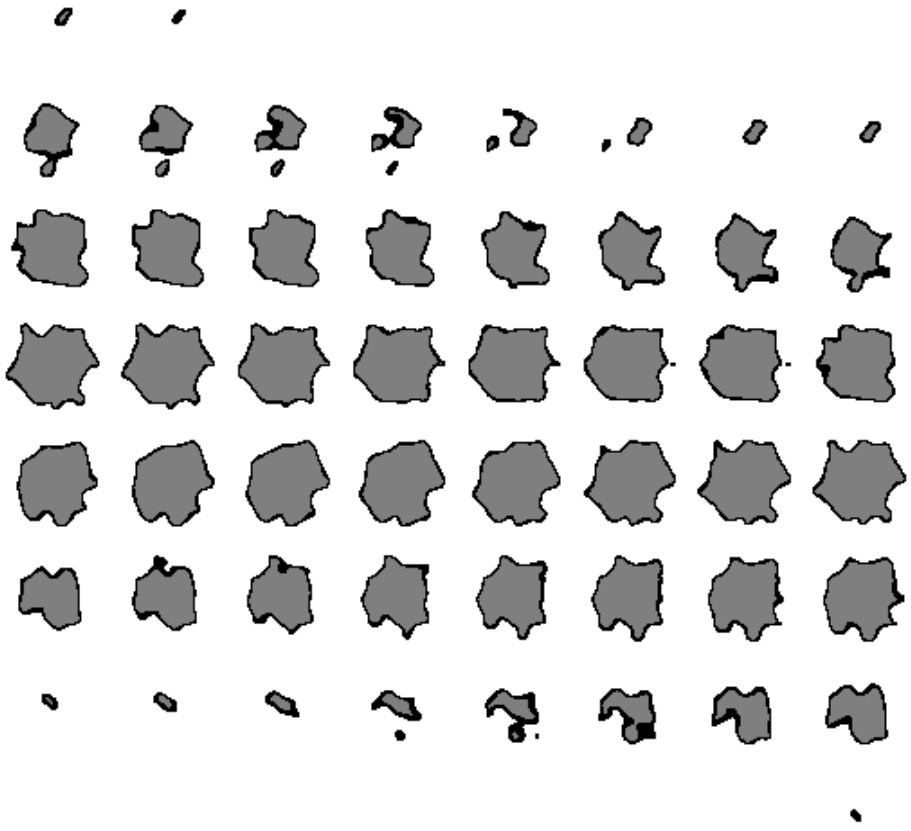
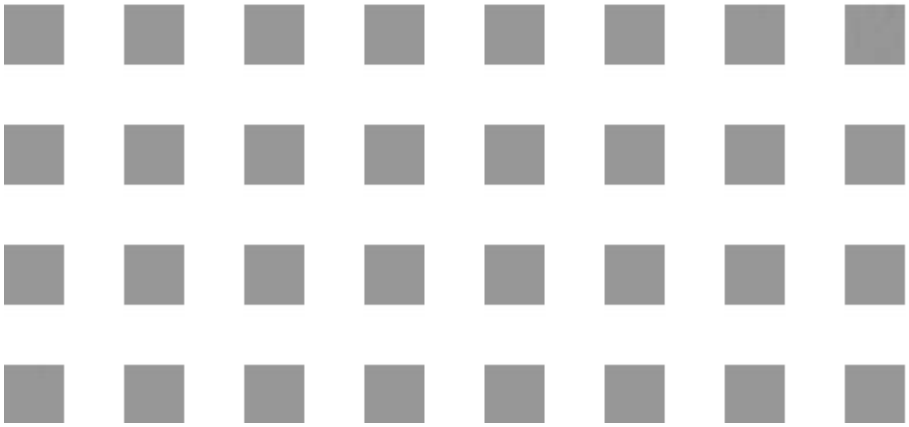


Figure 6.14: Boundary and Interior Points of the Noisy Sphere

Figure 6.15: Noisy Sphere, Raytraced from its B-spline Pyramid

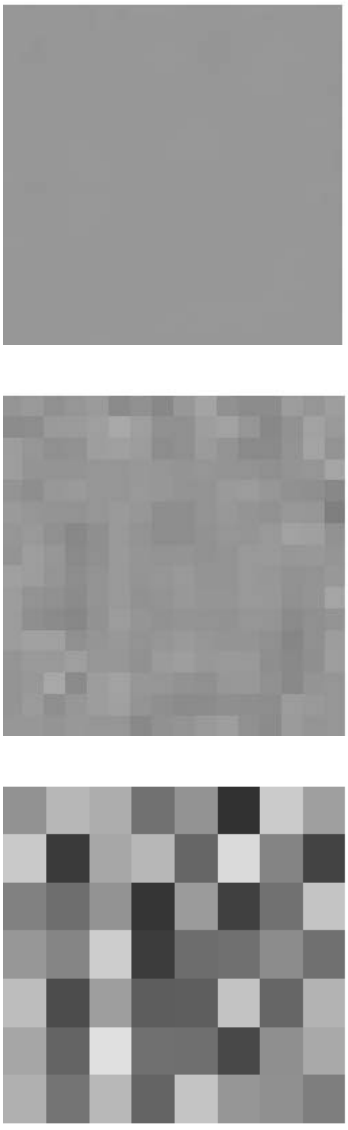


(level 2)

(level 1)

(level 0)

Figure 6.16: B-spline Pyramid of Noisy Sphere

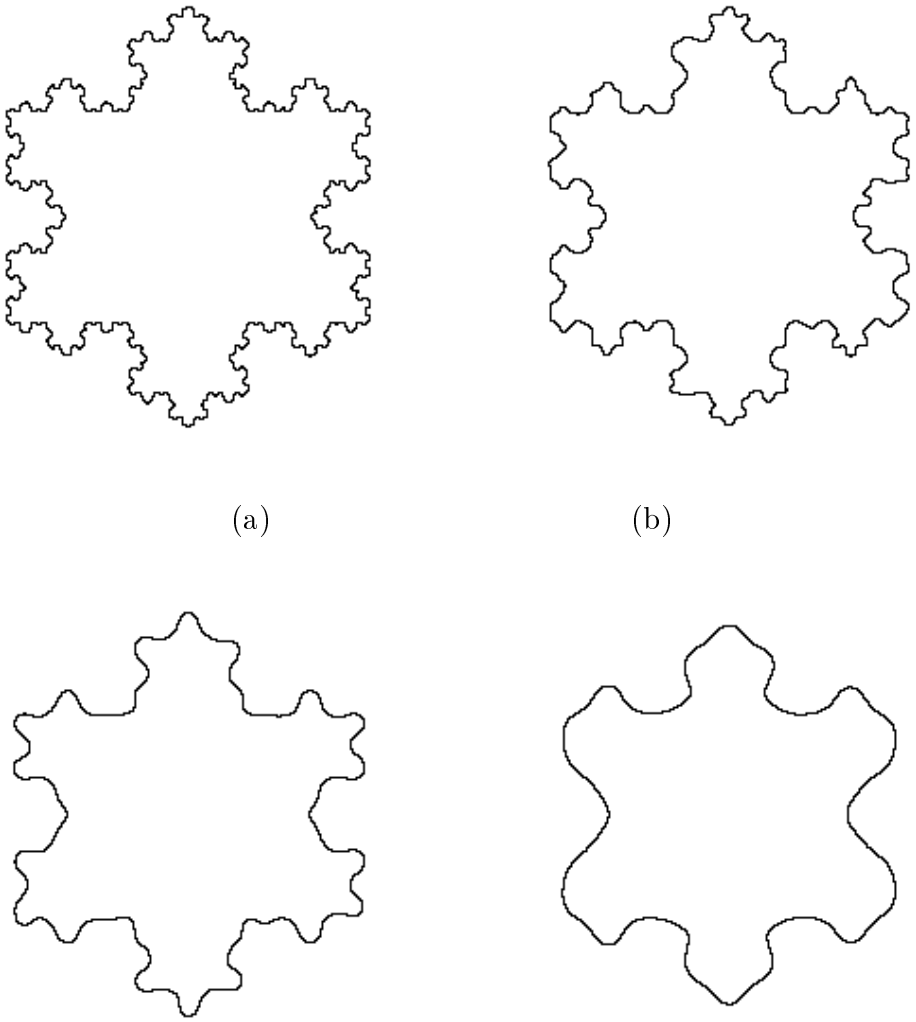


(level 2)

(level 1)

(level 0)

Figure 6.17: One Slice of Each Level (enlarged)



(a) (b)

(c) (d)

Figure 6.18: Koch Snowflake with Different Levels of Detail

Chapter 7

Spatial Decomposition

This chapter develops a method to create a piecewise representation of implicit surfaces and solids based on a simplicial subdivision of the domain of the implicit function f.

We create a model that is adapted to the boundary of an implicitly defined solid. This model is described by a structure composed of simple well-shaped cells.

Piecewise representations that are based on domain subdivision should explore the fact that f is a point-membership classification function which itself induces a partition of the ambient space.

We have seen in Chapter 3 that if $f^{-1}(c)$ is a connected m-dimensional surface, then it is the boundary of an implicit solid. It defines a decomposition of the domain of f consisting of two open sets of dimension $m+1$ (the interior and exterior points) and one open set of dimension m (the boundary points).

When we perform computations with implicit solids we are primarily interested in answering questions relative to their boundary surfaces. For example: given a point, is it on or close to the boundary? Given a line, compute its intersections with the boundary. This is particularly true in the case of visualization where the goal is to render the implicit surface. Secondarily, we may also need information about the interior volume of the object. One example is the computation of the center of mass for dynamics simulation.

One question that might be asked at this point is why not use the space subdivision induced by f as the shape description? Although this is the ideal and is used in the case of primitive implicit surfaces, it may not be feasible in general for arbitrary surfaces. First, the decomposition defined in terms of the entire surface boundary might be too complex. Second, it may not be possible to find a single function that describes the boundary as a whole.

Therefore, based on the above analysis, it is natural to look for a domain decomposition of f that is adapted to the surface $f^{-1}(c)$. Since this boundary is a m dimensional set, an effective solution is to construct a space decomposition that forms a $m+1$ dimensional layer around the surface.

7.1 Implicit Descriptions using Space Subdivision

The idea behind space subdivision is to decompose space into a collection of simpler point sets. As a result a domain is described by a structure that links these pieces together.

There are different decomposition schemes according to the types of structuring and the characterization of associated subsets. (See Appendix D). It is intuitive that more structured decompositions will encode more information but will be more restrictive and difficult to construct. So, it is necessary to find a balance between these two requirements.

A piecewise description of implicit objects using space subdivision consists of a decomposition of the domain of the implicit function f into a structure of cells. The function f is approximated by a simpler function in each cell. Each cell is classified in relation to the partition of space induced by f.

7.1.1 Simplicial Decompositions and Implicit Models

A space decomposition scheme suitable for our purposes is the *Simplicial Cell Decomposition*. It combines flexibility of structuring with simplicity of description. This type of space decomposition results in a *simplicial complex*, in which space is partitioned into cells defined by open simplices (An *n-simplex* is the simplest geometrical object of dimension n).

The cell decomposition induced by a simplicial complex is also called a *triangulation*. Among the various triangulations of the Euclidean space $\mathbb{R}^n$, the Coxeter-Freudenthal is the simplest one: It is constructed by subdividing the space using a uniform cubic grid and the triangulation is obtained by subdividing each cube in $n!$ simplices. More details about the construction of this triangulation are given in the Appendix D.

The triangulation is a particular case of *affine cell decomposition*. This allow us to replace the function f by its affine approximation $\tilde{f}$. The advantage of using the simplicial subdivision is that the interpolation of the function f is unambiguously defined from its sample values at the 0-dimensional simplices of the cell complex (the vertices of the triangulation).

The cells of the triangulation can be classified in relation to the space decomposition defined by the implicit surface $f^{-1}(0)$ by testing the sign of the implicit function f at the vertices of each cell of the triangulation. Assuming that the sampling grid is sufficiently fine, then, if the signs are the same for all vertices, the cell must be totally inside or outside the object. If the signs are different, the cell must intersect the bounding surface. Figure 7.1 shows these three cases.

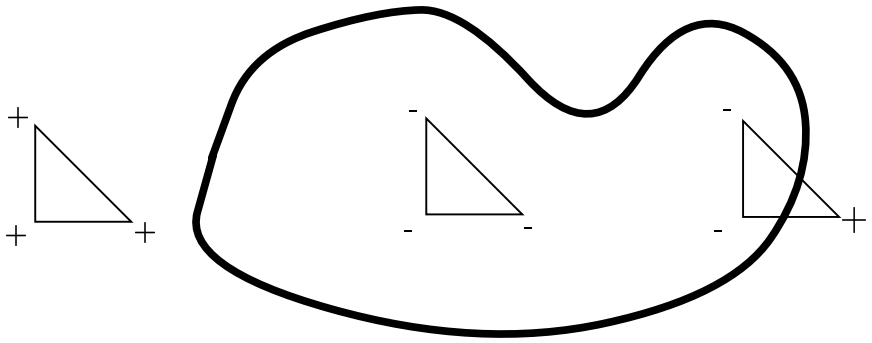


Figure 7.1: Cell classification

7.1.2 Subordinate Triangulations

A triangulation is called *quasi-regular* if its cells can be approximated by regular simplices. Quasi-regular cells are almost equilateral / equiangular. Therefore, this type triangulation is constituted only by well shaped elements, a very desirable property in many applications.

The *meshsize* of a triangulation $\mathcal{T}$ is defined by $\delta = \sup_{\sigma \in \mathcal{T}} \operatorname{diam} \sigma$, where σ represents a simplex of the triangulation. The meshsize depends on the space norm, and its value for different norms differ by a constant. Using the Euclidean norm the meshsize of the unitary Freudenthal triangulation in $\mathbb{R}^n$ is $\sqrt{n}$.

For a given positive real number $\varepsilon > 0$, we say that a triangulation $\mathcal{T}$ is ε-*subordinated* to an implicit surface $\mathcal{S} = f^{-1}(c)$, if the three following conditions are satisfied:

1. $\mathcal{S}$ is transversal to the triangulation;
2. Each n-simplex of $\mathcal{T}$ is quasi-regular.
3. For each n-simplex σ of $\mathcal{T}$ that intersects $\mathcal{S}$, there exists a k-face, $k < 0$, f_σ of σ, and a point $p \in \mathcal{S} \cap \sigma$, at a distance $< \varepsilon$ from the barycenter of σ, such that the tangent plane, $T_p\mathcal{S}$, of $\mathcal{S}$ at p is ε-close to the support plane of f_σ.

Figure 7.2 illustrates condition (3) for the two dimensional case.

The notion of subordinate triangulation allows us to define a space subdivision that is adapted to an implicit surface. Note that the meshsize is related to the tubular neighborhood of the implicit surface. See Chapter 3.

7.2 Physics Based Deformations

A deformation of space is a bijective map $W : \mathbb{R}^n \to \mathbb{R}^n$ that is a least continuously differentiable. The global transformation W warps the spatial domain along with

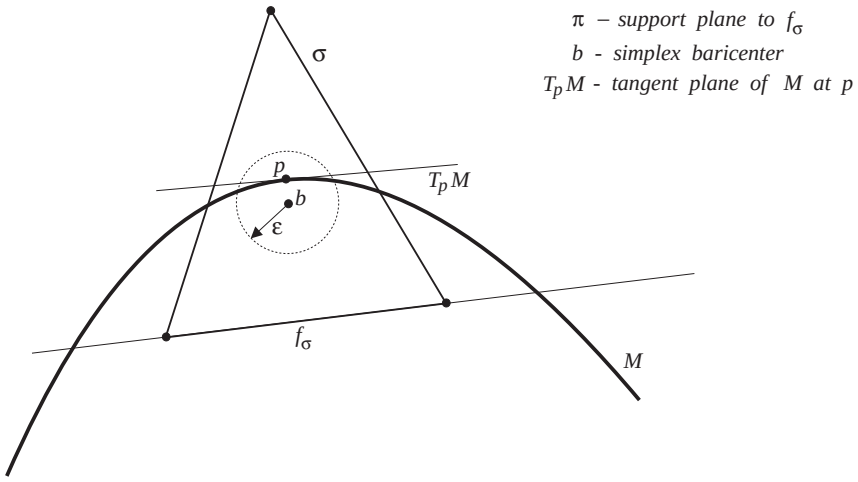


Figure 7.2: ε-subordinated triangulation.

the objects embedded in it. When W is a diffeomorphism it maps surfaces into surfaces. In a physics based deformation, W is derived from the equations of Newtonian mechanics.

A *discrete physical model* abstracts matter as an ensemble of particles related to each other by forces. Several physical phenomena may be naturally modeled using discrete physical systems (Greenspan, 1973). In a discrete physical system the particles interact under the action of internal and external forces. The associated motion equations are easily written as a classical Newtonian equation $F = ma$. Simple numerical integration methods, such as Euler's method, generally produce good results.

7.2.1 Spring-Mass Systems

A *particle system* is a finite set of particles which has an initial position in space and whose behavior is governed by a function of time. In a *physical particle system*, the particles have masses and the Newtonian mechanics dictates their dynamical behavior. The motion of a particle depends on its mass, position and velocity, and on the forces acting on it, either by other particles or by the ambient medium.

A *spring-mass system* is a physical particle system structured by connecting pairs of particles with springs. The springs impose *internal forces* that depend on the distance between these particles and drive the global behavior of the system. The resulting structure can be represented as a graph, where each particle is a node, and two nodes are connected when there is a spring joining the corresponding particles. Conversely, each graph linearly embedded in the space is naturally associated to a spring-mass system — a duality that can be exploited in association with affine decompositions of space.

Spring-mass systems are suitable to create physically-based models of deformable objects for dynamical simulation (Haumann, 1987), (Terzopoulos, Platt and Fleischer, 1989).

7.2.2 The Physical Model

The physical model of spring-mass system consists of a set of equations that balances for the internal forces generated by the springs and the external forces from the environment.

The model is governed by the Lagrangian equation

$$m_i \frac{d^2 x_i}{dt^2} + \gamma \frac{dx_i}{dt} + n_i = 0,$$

where m_i are the node masses and γ is a velocity dependent damping coefficient. The state variables are the node positions x_i and velocities $v_i = dx_i/dt$. The net force n_i is the sum of internal and external forces at node i.

The deformation energy of the mesh is accounted for by the springs that transmit attraction and repulsion forces to the linked nodes. The equation below gives the force s_k, resulting from a Hookean spring connecting nodes i and j, at x_i and x_j

$$s_k = c_k(\|x_i - x_j\| - l_k)$$

where c_k is the spring stiffness and l_k is the natural length of spring.

The physical attributes in the system are assigned to mass nodes and elastic springs.

The attributes of a mass node are:

m_i – mass of node i;

x_i – position of node i;

v_i – velocity of node i;

a_i – acceleration of node i;

n_i – net force on node i at time interval t;

The attributes of an elastic spring are:

l_k – natural length of spring k;

c_k – stiffness of spring k;

s_k – force generated by spring k;

$\{i, j\}$ – pair of nodes linked by spring k.

Some of the attributes above are determined during the course of the simulation of the model. Other attributes are initialized at the beginning of the simulation. The choice of constants, such as the spring stiffness and mass, depend on physical characteristics of the system.

7.2.3 Dynamics Simulation

To simulate the dynamics of the model, the state variables associated with each mass node are computed, according to the Lagrangian equations of motion.

The internal forces are due to the deformation energy of the elastic springs elongation or shrinking. The total internal force on node i comes form all the springs connected to it

$$g_i = \sum_{k \in L(i)} s_k.$$

The external forces are possibly originated from several sources acting on the object during the simulation. Some typical examples are: gravitational forces, forces caused by a viscous fluid, constraint forces and collision forces.

The net force on a node i is the sum of the total internal and external forces influencing the node during a given time interval Δt:

$$n_i = g_i + f_i.$$

The solution of the dynamics equations can be computed through an explicit Euler time integration procedure. This is the core of the simulation loop, and runs continuously calculating the state of the system at different instants in time. The time step may be adjusted according to the magnitude of the forces in the system.

Given the initial positions x_i^0 and velocities v_i^0 of the mass nodes ($i = 1, \ldots N$), the current acceleration, the new velocities and positions at each subsequent time step are determined by:

$$\begin{aligned} a_i^t &= \frac{n_i^t}{m_i} \\ v_i^{t+\Delta t} &= v_i^t + \Delta t a_i^t \\ x_i^{t+\Delta t} &= x_i^t + \Delta t v_i^{t+\Delta t} \end{aligned}$$

This method is only first order accurate, and can be numerically unstable. Numerical stability is improved using higher order methods such as the Runge-Kutta formulation.

7.3 Space Subdivision Representation

In this section we describe the construction of a piecewise representation of implicit objects that is based on adapted space subdivision.

The idea is to produce a "thick" shell around the boundary of the object (Velho and de M. Gomes, 1991b). This is done using a physics based deformation, as explained in section 7.2.

First, a simplicial decomposition of the domain of the implicit function f is created, as discussed in section 7.1. Then, we convert this structure into a spring-mass mesh. We employ dynamics simulation to conform the mesh into the desired shape. The equilibrium position of the mesh gives a triangulation that is subordinate to the surface $f^{-1}(c)$.

7.3.1 Initial Structure

The initial structure is created using a simplicial space decomposition of the domain of f as follows:

1. Compute a Coxeter-Freudenthal triangulation of the ambient space;
2. Replace the function f by its simplicial approximation $\widetilde{f}$ relative to this triangulation;
3. Classify the cells into: inside, boundary and outside, according to the implicit function.

(See Figure 7.3).

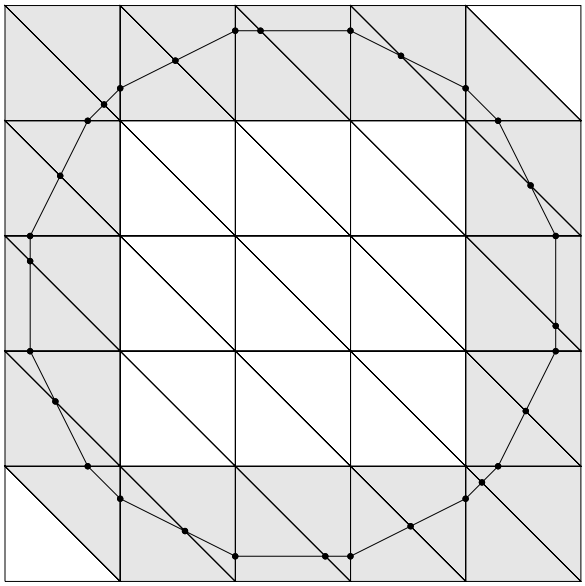


Figure 7.3: Simplicial Approximation of implicit object.

7.3.2 Adaptation

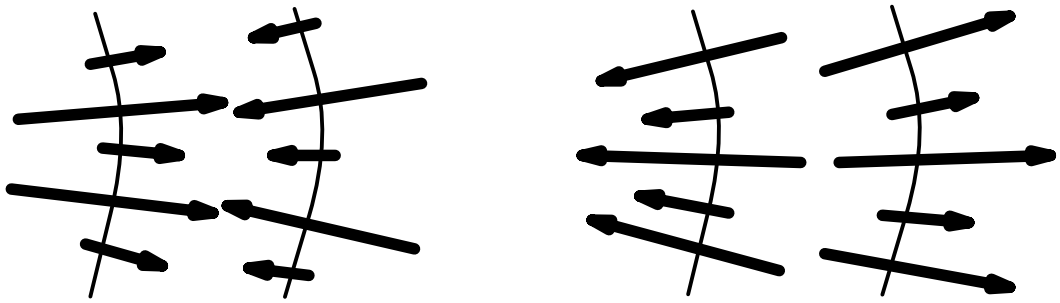
The adaptation method consists of the following steps:

1. Associate the triangulation of f with a spring-mass mesh (i.e., the vertices of the triangulation are identified with mass nodes and the edges of the triangulation are identified with springs).

2. Submit the spring-mass system to deformation forces letting the system relax to an equilibrium position.

We create deformation forces that conform the mesh to the boundary of the implicit object. The simulation takes into account the internal forces produced by the springs as well as the external deformation forces.

The external forces are derived from the implicit function. More specifically, two opposite attracting and repulsing force fields are generated using the gradient vector field ∇f of the function f. One force field defined inside a tubular neighborhood of the implicit surface generates repelling forces that prevent points from being too close to the surface. The other force field, defined outside this neighborhood, generates attraction forces that pulls points towards the surface. These force fields are depicted in Figure 7.4 for the two-dimensional case.



Attraction field Repulsion field

Figure 7.4: Force Fields

In order to facilitate the relaxation of the mesh structure into the desirable configuration, the initial rest length of the springs is made smaller than the initial grid spacing. This means that we start the process with a tensioned mesh that moves to a rest position under the action of internal and external forces. The spring stiffness is set based on the variation of the implicit function at the nodes of the initial mesh.

The spring-mass system described above reaches an equilibrium state and at this position it defines a triangulation that is subordinate to the implicit surface.

Figure 7.5 reveals the evolution of the dynamic simulation. It shows four stages of the mesh deformation process. The initial mesh is shown in Figure 7.5(1). Intermediate configurations can be seen in Figures 7.5(2) and 7.5(3). The final solution

is shown in Figure 7.5(4). Note how the geometry of the adaptive mesh converges to the shape of the object. Note also that the final configuration is very regular.

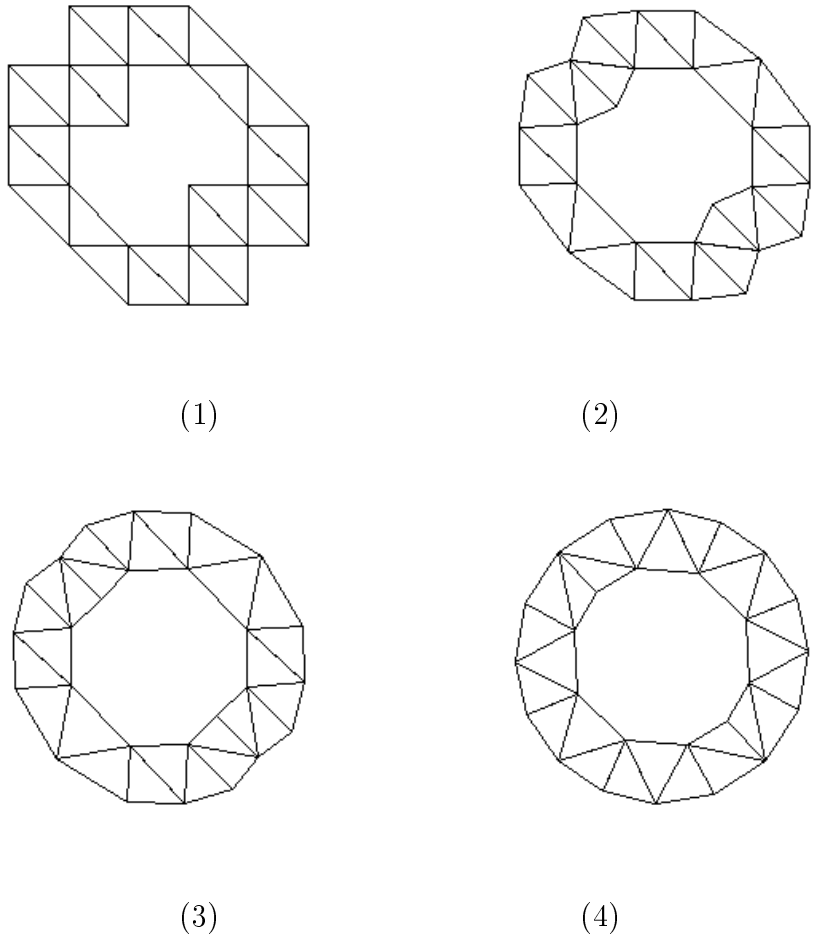


Figure 7.5: Four stages of the mesh deformation process

We conjecture that the force fields can be chosen to produce different types of adapted decompositions. For example, the mesh structure can be made to adapt either to intrinsic surface properties (such as local curvature) or to any other spatial property (such as material resistance).

7.3.3 Representation

The piecewise linear description of the implicit object can be one of three options: only the boundary cells, the boundary plus the inside cells, or all the cells of the adapted simplicial complex. The choice will depend on the kind of computations we want to perform with the implicit object.

In the last two cases, the force fields affect only the point masses belonging to the boundary cells. The internal (and external) nodes are made passive, i.e. they are not driven by external forces.

7.3.4 Examples

The following examples illustrate the use of the method in two and three dimensions.

Hollow and Solid Objects

This example demonstrates the generation of a simplicial subdivision representation for a hollow and solid two dimensional objects.

Example 7.1 (Disk) The implicit object is a disk defined by the equation $x^2 + y^2 - r^2 \leq 0$, $r = 1$.

Figure 7.6 shows examples of the mesh structures for a hollow circular shell and a solid disk.

Different Mesh Sizes

This example demonstrates the result of different grid sizes.

Example 7.2 (Circle) The implicit object is a circle defined by the equation $x^2 + y^2 - r^2 = 0$, $r = 1$.

Figure 7.7 shows the adapted meshes respectively for 2×2, 6×6, 12×12 and 60×60 grids.

Figure 7.8 is a detail of the mesh corresponding to a grid of size 80×80.

3D Examples

The next two examples illustrate the application of the method to construct the spatial subdivision in three dimensions.

Example 7.3 (Sphere) The implicit object is a sphere defined by the equation $x^2 + y^2 + z^2 - r^2 \leq 0$, $r = 1$. Figure 7.9 shows the initial and the final mesh structures.

Example 7.4 (Torus) The implicit object is a torus defined by the equation

$$(x^a + y^2 + z^2 - (a^2 + b^2))^2 - 4a^2(b^2 - z^2) = 0$$

Figure 7.10 shows the initial and the final mesh structures. Note that the meshsize is finer than in the previous example.

7.4 Applications

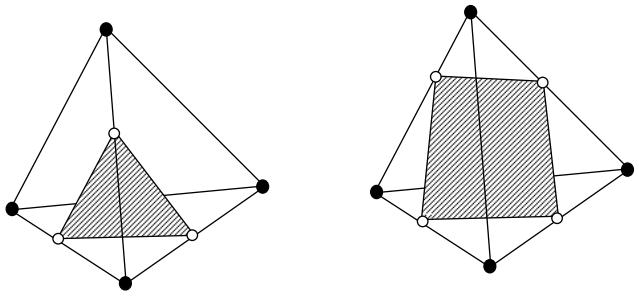
We developed a method that generates adapted triangulations of the domain of the implicit function. The resulting structure is quasi-regular and conforms well to the shape of the implicit surface. The mesh resolution can be easily parameterized. The mesh construction process is totally automatic.

The method has multiple advantages. Besides the adapted triangulation, it constructs in a natural way a spring-mass system associated with the implicit object. This can be exploited in different ways for physics based animation and polygonization of implicit surfaces.

7.4.1 Conversion to PL Parametric Descriptions

An application of the method developed in this chapter is to solve the problem of implicit to parametric conversion (de Figueiredo et al., 1992).

A piecewise linear approximation of the boundary of the implicit object can be easily derived from the simplicial decomposition of the domain of f. The set of boundary simplices of $\widetilde{f}$ form a three dimensional combinatorial manifold that has as its dual a two dimensional manifold that approximates the boundary surface $f^{-1}(c)$. It is constructed from the intersection of the true surface with the edges of the each 3-simplex σ, resulting in either one or two 2-simplices that approximate $f^{-1}(c)$ inside σ. These two cases are illustrated in the Figure below.



Two cases of polygon generation

Since the simplicial complex forms a triangulation that is *subordinated to the surface* $f^{-1}(c)$, it can be shown that the associated polygonization is *quasi-regular*.

Polygonization

The polygonization of an implicit surface $\mathcal{S}$ is computed from the subordinated triangulation of the domain of $\mathcal{S}$. The surface intersects each 3-simplex σ in at most 4 distinct points, each one located on a different 1-dimensional face. Therefore, the linear approximation to $\mathcal{S}$ inside σ is formed by one or two triangles. The set of all these triangles constitute the combinatorial manifold that approximates $\mathcal{S}$.

Example 7.5 (Cylinder) This example illustrates the polygonization of a cylinder defined by the implicit equation $x^2 + y^2 = 1$ Figure 7.11 shows a sequence corresponding to different phases of the mesh deformation process for a cylinder. Figure 7.11(a) depicts the initial mesh created from a Freudenthal triangulation of the ambient space, Figure 7.11(b) reveals the final mesh in its equilibrium position. From the images, it is apparent that the initial mesh was constrained to lie in a tubular neighborhood of the implicit surface, conforming to the cylinder's shape.

The polygonal approximation is derived from the final mesh. Figure 7.12 shows the final polygonal approximations for the cylinder.

Figure 7.13 shows a detail of the polygonization associated with the spatial subdivision representation before and after the deformation process (Figures 7.13(a) and 7.13(b) respectively).

Note how the deformation of the mesh produces a very homogeneous polygon structure, transforming long and thin elements in almost equilateral ones. This is because the triangulation resulting from the dynamical simulation is subordinate to the surface and, as a consequence, the associated polygonization is quasi-regular.

7.4.2 Physics-Based Animation

Another application of the method developed in this chapter concerns the use of physics based methods with implicit objects. For example in the simulation and analysis of deformations (Velho and de M. Gomes, 1991a). We can use this representation to generate a spring-mass mesh or linear finite elements.

This example illustrates the use of the spatial subdivision representation in a physics-based modeling and animation environment. A spring-mass mesh describing the physical properties of the object is automatically generated from this structure. The mesh is incorporated in the physically-based environment as a means of interacting with the implicit object. The visualization of the state of the simulation can be done either using a polygonal approximation of the surface as described previously, or by sampling directly the deformed implicit object.

Example 7.6 (Flexible Tube) We simulate a flexible tube falling under a gravitational field. Figures 7.14 and 7.15 show one frame of this dynamics simulation. Figure 7.14 depicts the spring-mass mesh used in the simulation. Figure 7.15 shows the corresponding polygonization of the tube.

7.5 Summary

In this chapter, we developed a spatial decomposition model for implicit surfaces and solids that is based on the simplicial subdivision of the domain of the implicit function. We studied space decomposition structures and defined the concept of subordinate

triangulation to a surface. We reviewed physics based deformation methods and described in some detail the simulation of spring-mass systems.

The spatial decomposition model consists of a simplicial cell complex that is adapted to the boundary of the solid and composed only of well-shaped elements. This structure is generated from a Coxeter-Freudenthal triangulation that we identify with a spring-mass mesh. This mesh is deformed by forces derived from the gradient of the implicit function to produce a triangulation subordinate to the implicit surface.

This simplicial representation not only gives a piecewise linear description of the implicit function, but also provides a polygonal approximation of the implicit surface.

Our method is the first to combine simplicial decomposition with physics based simulation to generate a piecewise implicit description. Simplicial methods have been used to produce a polygonization of implicit surfaces. Some employ an uniform subdivision of the ambient space (Allgower and Gnutzmann, 1987), and others recursive subdivision (Hall and Warren, 1990). In both cases the resulting structure is not adapted to the implicit object. Physics based techniques have been used in computer graphics to obtain samples points on implicit surfaces (de Figueiredo et al., 1992), this method employs a uncoupled particle system instead of a coupled structure. Adaptive Spring-mass meshes have also been used in vision for image sampling and reconstruction (Terzopoulos and Vasilescu, 1991). In this algorithm the mesh covers the entire image and the deformation forces are applied to all nodes, and not only to a set of boundary cells as in our method.

The space decomposition models introduced in this chapter have several applications in computer graphics. Some of these applications are the polygonization of implicit surfaces, the automatic mesh generation for dynamics simulation and finite element analysis.

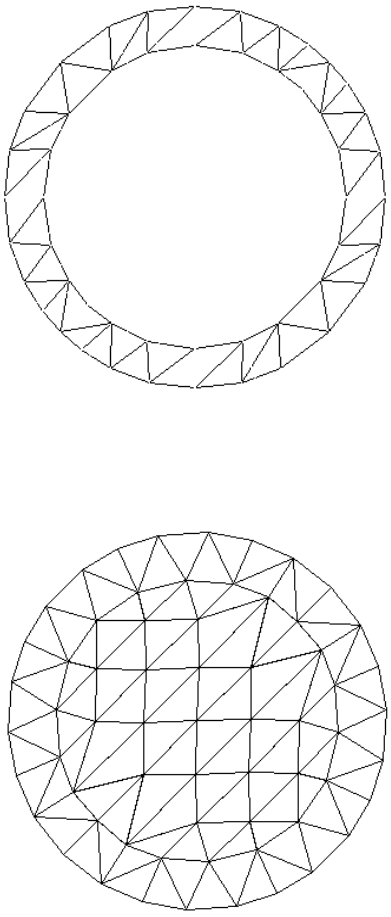


Figure 7.6: Domain Decomposition for Hollow and Solid Objects

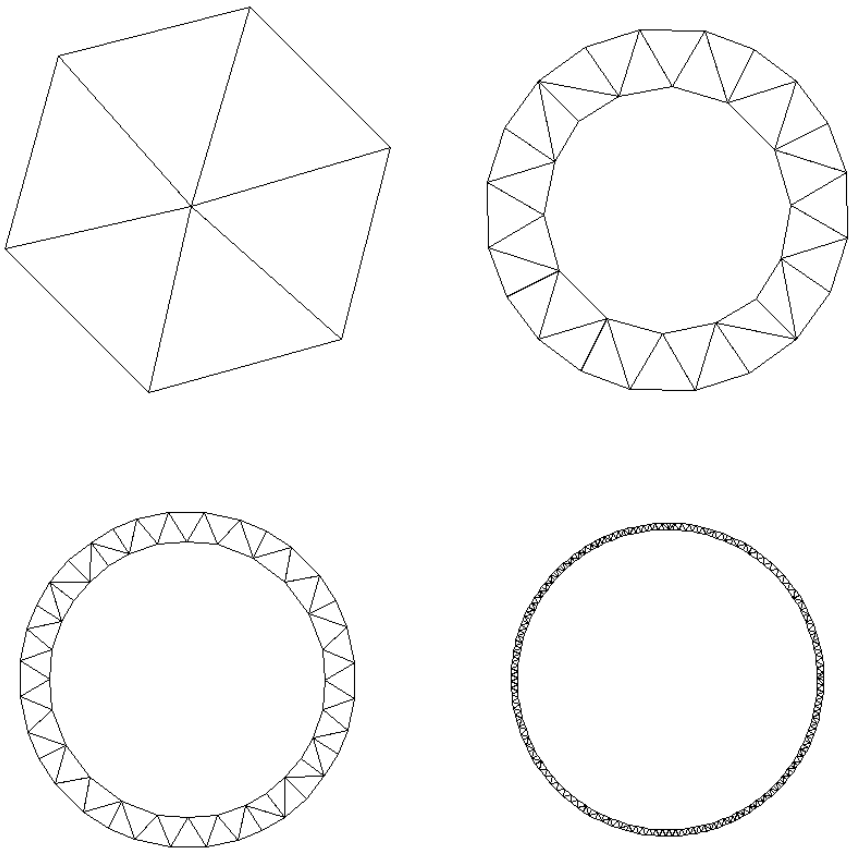


Figure 7.7: Grid sizes of: 2, 6, 12, 60

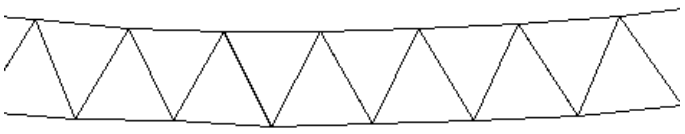
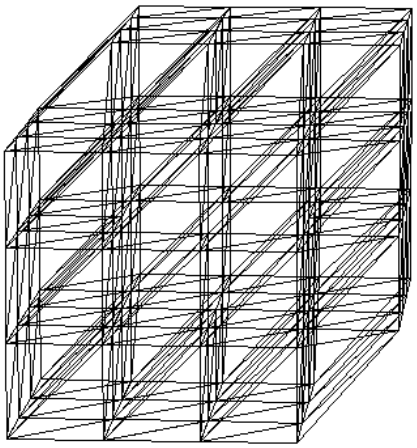
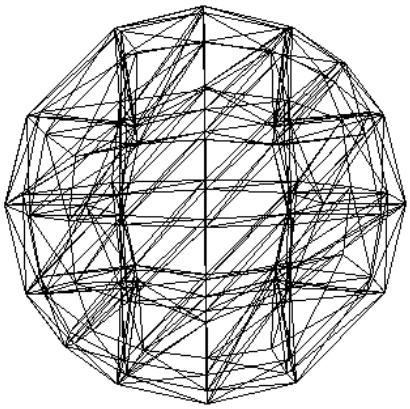


Figure 7.8: Detail of the shell corresponding to a grid of size 80

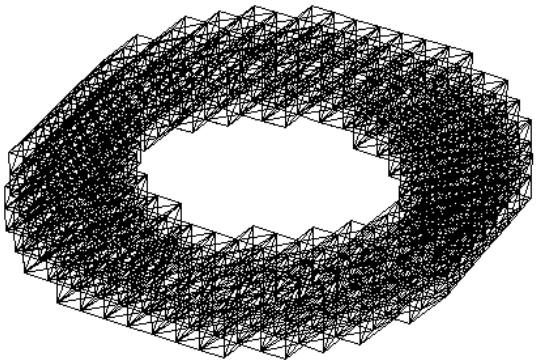


(a)

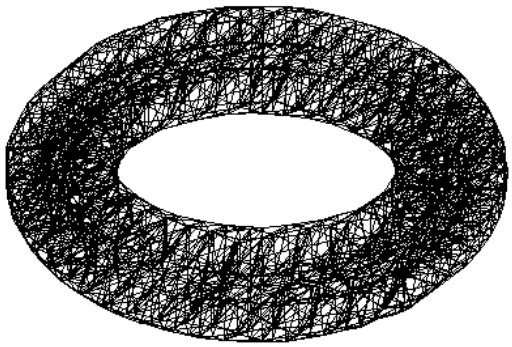


(b)

Figure 7.9: Sphere: Initial and Final Mesh ($4 \times 4 \times 4$)

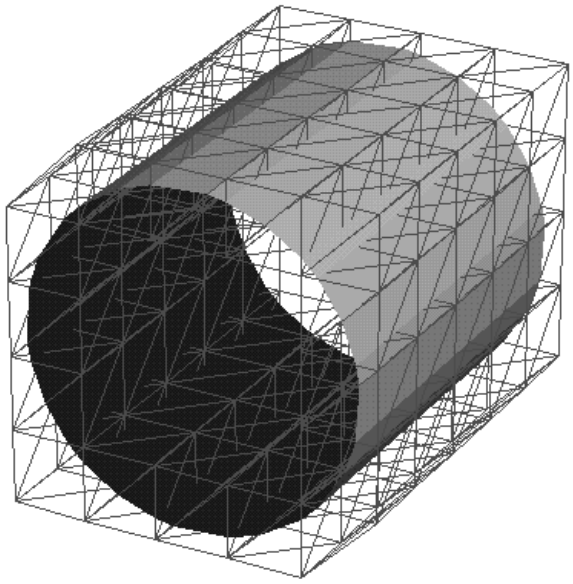


(a)

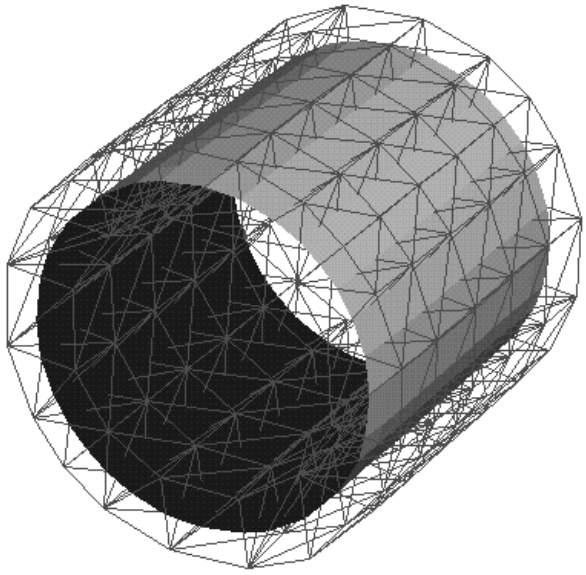


(b)

Figure 7.10: Torus: Initial and Final Mesh ($16 \times 16 \times 4$)



(a)



(b)

Figure 7.11: Grid deformation of the Cylinder

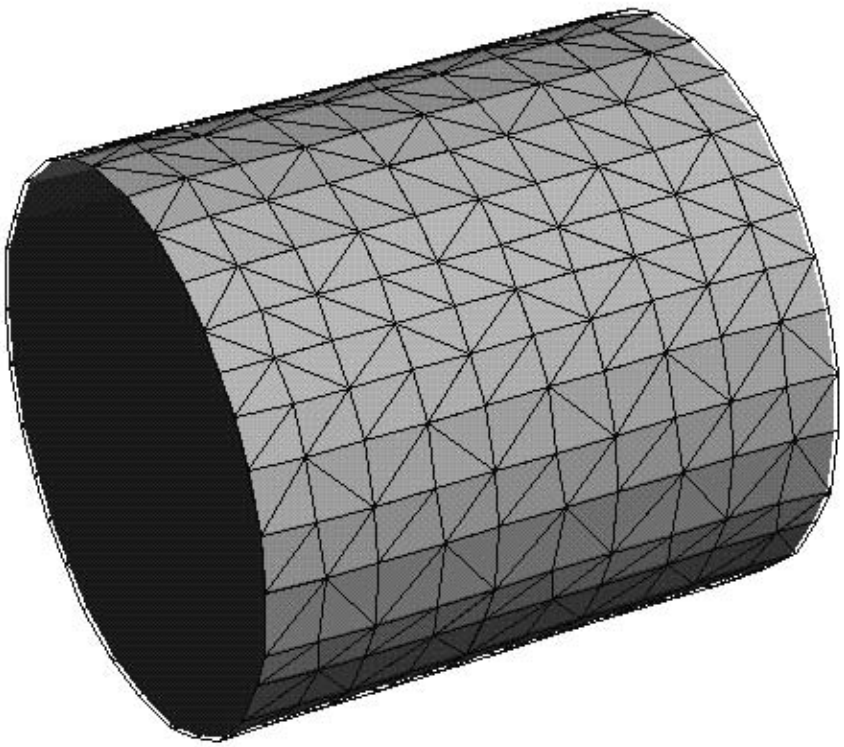
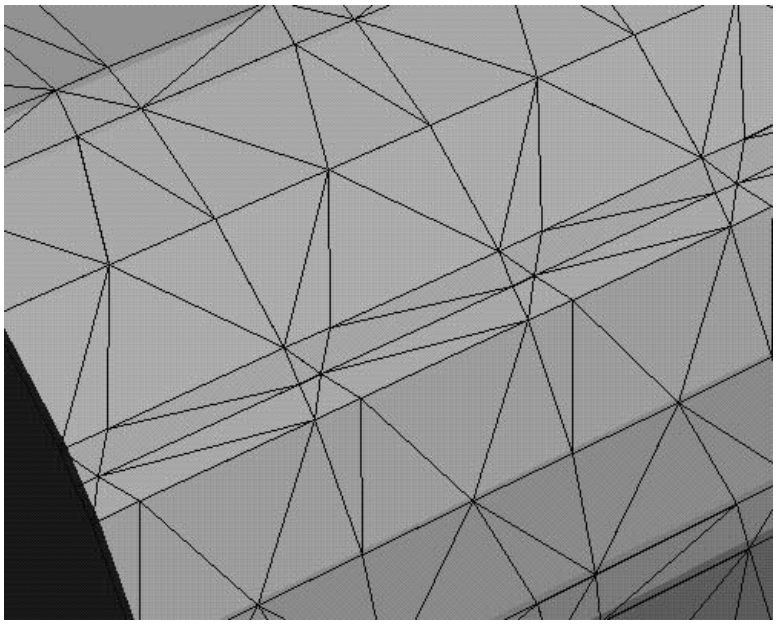
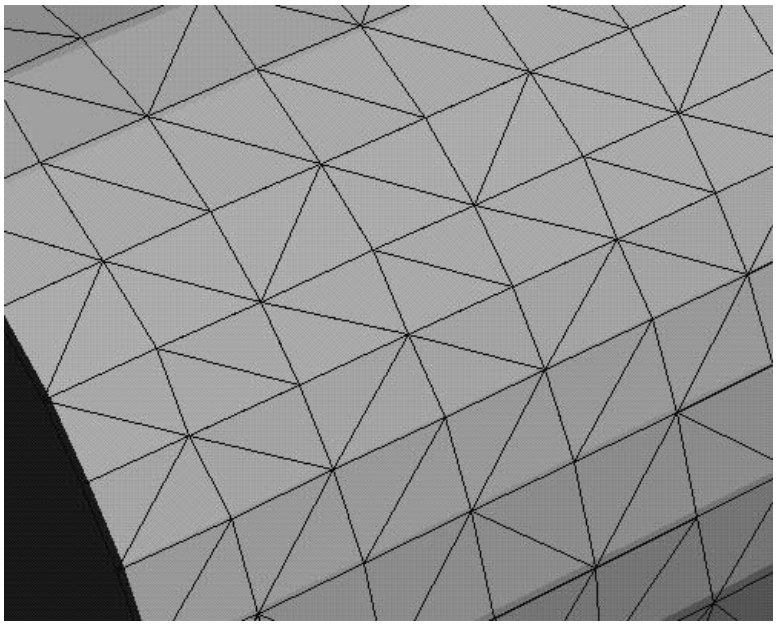


Figure 7.12: Polygonization of the Cylinder



(a)



(b)

Figure 7.13: Detail of the Cylinder's Polygonization

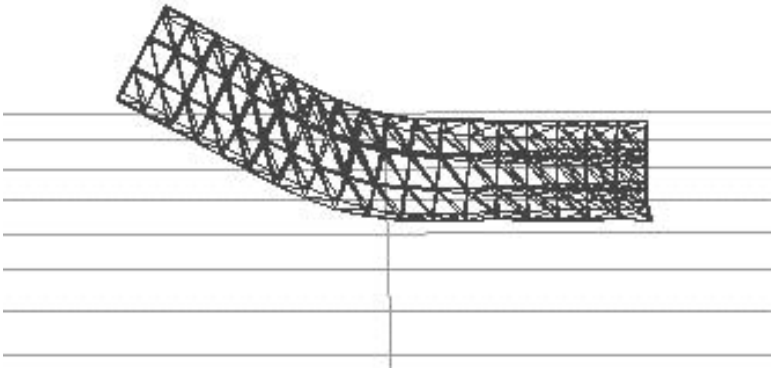


Figure 7.14: Spring-Mass Mesh for Flexible Tube

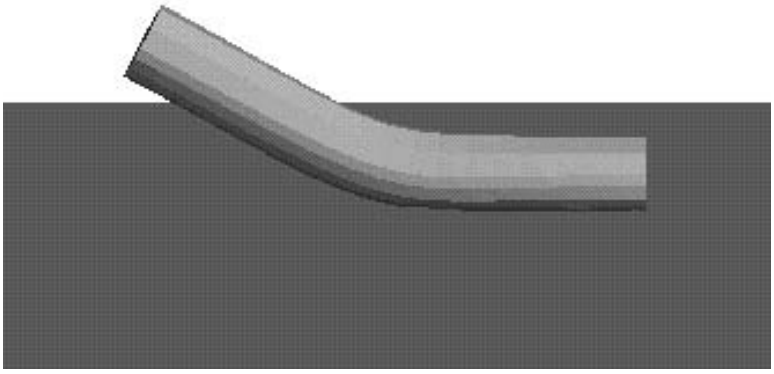


Figure 7.15: Frame of Simulation for Flexible Tube

Chapter 8

Conclusions

This chapter summarizes the thesis, reviews its contributions and proposes directions for future work.

8.1 Summary

In this thesis we investigated the piecewise description of implicitly defined objects. The investigation includes an original study of the implicit model as well as the development of new schemes to represent implicit objects. This study of implicit models provided a theoretical framework that was used in the definition of the proposed representations and in the evaluation of the associated computational methods. The implicit models introduced are general and can be used to describe arbitrary shapes. Their creation is based on decomposition and adaptation processes that produce a compact representation in terms of simple elements. These characteristics make them attractive in computer graphics and related areas. The examples in the thesis demonstrated the use of piecewise models for the solution of many types of problems. Because of their relevance, we emphasized applications to representation conversion and to visualization. As a whole, this research provides a complete modeling paradigm for implicit surfaces using piecewise descriptions. This statement is supported by the fact our methods make possible the approximate conversion between the main forms of geometry specification, as indicated in Figure 8.1.

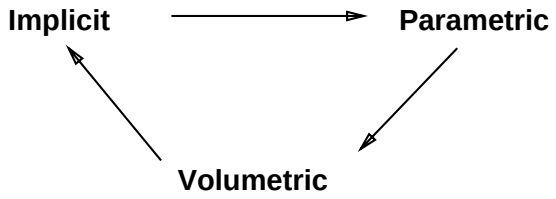


Figure 8.1: Conversion between geometric forms.

The starting point of this research was an analysis of the implicit function and its level surfaces. This made it possible to relate intrinsic and extrinsic properties

of surfaces defined in implicit form. It also provided the necessary arguments to discuss the effectiveness of implicit models and it allowed us to identify the important elements of an efficient representation. Such a framework gives an indication of the best strategies for creating piecewise implicit geometric descriptions.

Using the above concepts, we showed the connection between the domain of the implicit function and the tubular neighborhood of the implicit surface. A good implicit model is given by a smooth function whose variations are subordinate to the surface it defines. We developed a method to estimate the tubular neighborhood of a surface based on multiscale edge detection. The edge information can also be used to reconstruct the implicit function and control its smoothness. One application of this method is in the conversion of parametric to implicit descriptions.

In this thesis we exploited decomposition schemes to produce piecewise models that are adapted to geometric features of the implicit surface. We developed two methods, one based on functional decomposition and other based on spatial decomposition. The functional decomposition method employs the biorthogonal wavelet transform to discriminate the components of the implicit model at multiple scales. It gives a representation that is adapted to function variations and is suited to efficient computation. One application of this method is in the conversion of volumetric to implicit descriptions.

The spatial decomposition method employs a quasi-regular triangulation that subdivides the domain of the implicit function and is subordinate to the implicit surface. In order to adapt it to the surface, the simplicial complex is associated with a spring-mass mesh that deforms using dynamics simulation. The result is a piecewise model composed of well shaped elements. One application of this method is in the conversion of implicit to parametric descriptions.

8.2 A Final Example

The example in this section overviews the thesis results and illustrates their application to the conversion of geometric representations. We apply all the methods developed in the thesis to a single two dimensional shape. This demonstrates how the methods fit together and allows a comparison between the different descriptions that they produce.

We submit the shape to the following changes of representation:

- Parametric $\longrightarrow$ Volumetric
- Volumetric $\longrightarrow$ Implicit
- Implicit $\longrightarrow$ Parametric

The object is a "T" shape. In this first step we convert from a parametric to a volumetric representation using the method developed in Chapter 5. The input

shape is represented by its boundary curve in parametric form. The output is a representation of the shape in implicit form by a sample array. The method generates a smooth implicit function that has the boundary curve as one of its level sets.

Figure 8.2 shows the parametric description of the T-Shape. The boundary is described by a cubic Bezier curve. It is composed of 19 curve segments and contains a total of 54 control points, or 108 coefficients for the x and y coordinates. The input to the method is the characteristic function of the shape in volumetric form. Therefore, we must create this function in a pre-processing stage. The characteristic function is constructed by scan-converting the boundary curve into a 2D pixel array and filling in the interior. The resulting characteristic function is shown in Figure 8.3.

The method of Chapter 5 generates a smooth implicit function by decomposing the characteristic function into multiscale edges using a dyadic wavelet transform. The edges are given by the maxima values of the wavelet transform at each scale. The method modifies these values so that they correspond to edges of a smooth function at the same locations. The function is then reconstructed from these new maxima values. Figure 8.4 shows the modified maxima values for three different scales. Smoother edges are produced at locations where the maxima values exhibit large variations. Sharper edges are produced at locations where the maxima values exhibit small variations. Figure 8.5 shows an image of the two dimensional sample array of the smooth implicit function reconstructed from the multiscale edges in Figure 8.4.

In the second step, we convert the volumetric representation produced in the previous step to an implicit representation using the method developed in Chapter 6. The input is a two dimensional array of values corresponding to uniform samples of an implicit function. The output is a piecewise analytical representation of this implicit function given as a linear expansion over a set of B-spline functions at different scales. The method generates a hierarchical and structured implicit representation of the data.

The method of Chapter 6 constructs this representation by performing a multiscale decomposition of the implicit function. The method consists of three parts: the Laplacian transform is applied to the input sample array producing the coefficients in the B-spline basis; the coefficients with non-negligible magnitude are selected; and these relevant coefficients are stored in a spatial data structure.

Figure 8.6 shows the Laplacian pyramid obtained from the implicit function of Figure 8.5. It is a four level pyramid with a total of 241 non-negligible coefficients. Level 0 (the topmost level) has 60 coefficients, level 1 has 91 coefficients, level 2 has 80 coefficients and level 3 has 10 coefficients. Figure 8.7 shows the B-spline functions in the Laplacian pyramid with relevant coefficients. The support of the B-spline is indicated by a circle. So, larger circles correspond to coarser scales while smaller circles correspond to finer scales. Figure 8.7 shows the boundary curve derived from the multiscale B-spline representation.

In the third step, we convert the implicit representation produced in the previous

steps to a parametric representation using the method developed in Chapter 7. The input is an implicit function given either in volumetric or analytic form. The output is a simplicial cell complex that is subordinate to the shape boundary. A piecewise linear parametric description of the boundary curve can also be generated from such a representation.

The method of Chapter 7 constructs this representation from a spatial decomposition that is adapted to the shape boundary using physics based techniques. In each cell of the decomposition, the implicit function is approximated by interpolating the function values at cell vertices. The method consists of three parts: a regular triangulation of the the domain of the implicit function is created; this triangulation is identified with a spring-mass mesh; the mesh is deformed by forces derived from the gradient of the implicit function. A piecewise linear parametric approximation of the boundary curve is then given by the dual of the cell complex structure.

Figure 8.9-a shows the regular triangulation corresponding to the initial mesh for a solid shape. Figure 8.9-b shows the final adapted mesh. The triangulation was constructed from a 16×16 rectangular grid. It contains a total of 179 cells. Figure 8.10 shows the final mesh for a hollow shape. It contains 179 cells. For each vertex of the triangulation, the (x, y) position and implicit function value are stored. Figure 8.11 shows the approximation to the boundary curve derived from the simplicial complex of Figure 8.10.

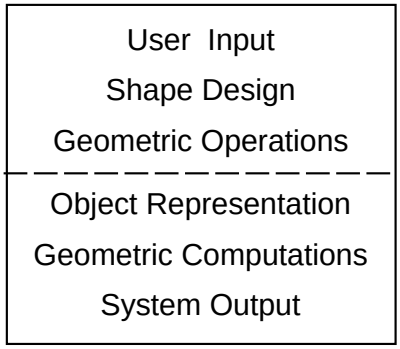
8.3 Future Work

The work reported in this thesis can be extended in two main directions:

- Integration in a modeling and graphics system
- Development of applications and testing.

Both of these items are related to the consolidation of our framework in a concrete, real world, environment.

The implicit decomposition models that we proposed can form the foundation of a graphics system based on implicit objects. There are many issues that need to be considered in such a system. For example, interaction, user interfaces, modeling techniques, architecture, etc. The computational methods and associated geometric representations developed in this thesis are well suited to form the algorithmic substrate of the system. They provide all the machinery required for the internal description of shapes and their processing. However, there are some other issues to be addressed, such as the paradigms for object specification and the class of operations with them. If the graphics system is divided in two parts, as shown in the diagram below, we can say that our work takes care of the lower half of the system's functionality, while the upper half remains to be created.



Graphics System

As we have demonstrated throughout the thesis, implicit decomposition models give an effective representation of geometric objects and allow efficient computation. On the other hand, constructive models are more adequate for shape design and the definition of shape operations. Although it is true that we can convert from constructive to decomposition models, a seamless integration between these two types of models still remains as an important research goal.

8.4 Directions

In almost every area of science and engineering, including computer graphics, we can classify the problems according their level of complexity, relating them to the underlying models and techniques used in the solutions.

For problems of low complexity we can always find an analytic model which is described in closed form and allows an exact solution. For problems of moderate complexity we are usually able to construct a deterministic model which is formulated numerically and provides only an approximate solution. For problems of high complexity we often must resort to a probabilistic model, which is specified using stochastic methods and gives a solution that is valid only in a statistical sense.

The table below summarizes this classification.

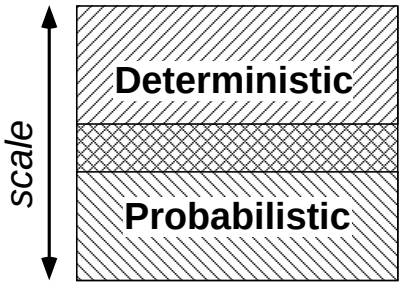
complexity	*model*	*formulation*	*solution*
low	analytic	closed form	exact
medium	deterministic	numerical	approximate
high	probabilistic	stochastic	statistical

This scheme applies, in particular, to the problem of geometric modeling using implicit descriptions. In that context, we can identify the three levels of shape complexity with, respectively, atomic, piecewise and fuzzy implicit models. Examples of atomic descriptions are primitive shape models, such as the quadrics or superquadrics.

Examples of piecewise descriptions are constructive models, such as CSG, or decomposition models, such as the multiscale B-spline pyramid. Examples of fuzzy description are hypertexture models, such as fractal clouds.

An open question concerning the practical application of those models is whether or not it is possible to combine them and, if so, how. This is important for several reasons: First, these models are complementary, in the sense that the same problem can be considered at various levels, i.e., it can be made simpler or more complex depending on the application. Second, from the user's point of view, these models offer different degrees of control that should be available simultaneously, i.e, global, local and procedural control. Third, most shapes are not uniformly complex; they are smooth in some areas and rough in others. Therefore, it would be desirable to model them differently according to their local complexity.

In this thesis, we took a decisive step towards the formulation of powerful deterministic implicit shape models that are adapted to the geometry of objects. A natural direction for future development of our framework would be to consider also probabilistic models. We conjecture that hierarchical representations based on *scale* are the key to a seamless integration of deterministic and probabilistic models. The results in (Szeliski and Terzopoulos, 1989) give a clear indication that this is true. In this way, large scale properties would be described by a deterministic model and small scale properties would be described by a probabilistic model. The problem is how these models can coexist for the same object such that it is possible to make a smooth transition between them. Also, it would be desirable to provide the user with the ability to control the scale and range of the transition region. This situation is illustrated in the following diagram.



Mixed Hierarchical Model

An answer to those questions may be given by multiscale wavelet models with deterministic and stochastic components. Initial steps in this direction have already been taken in the context of image processing (Basseville et al., 1992).

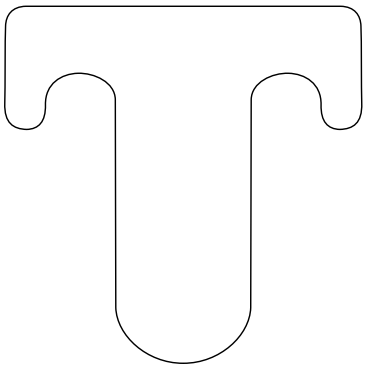


Figure 8.2: T-Shape in Parametric Form, described by a Bezier curve

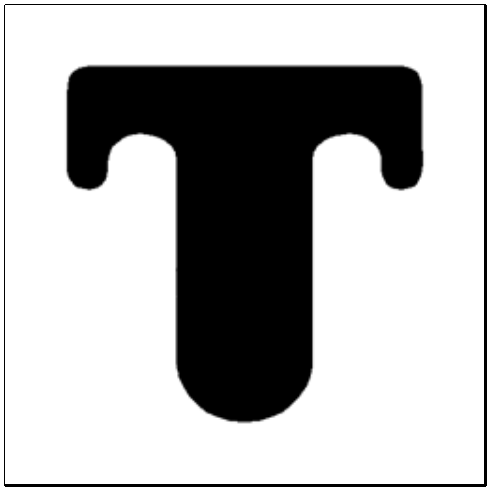


Figure 8.3: Characteristic Function of the T-Shape

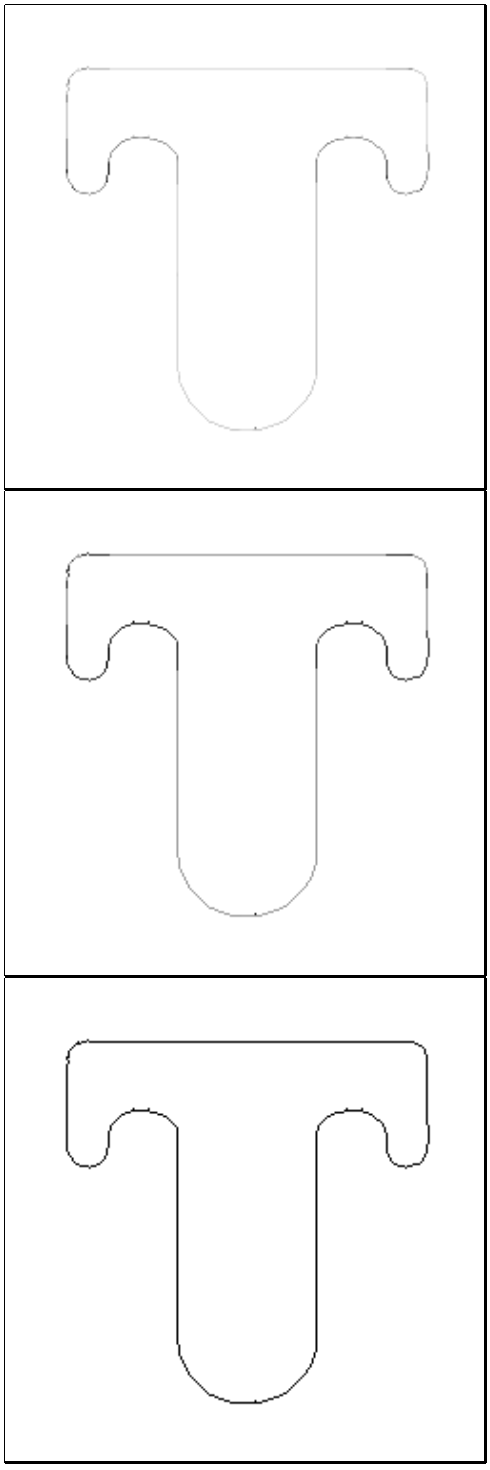


Figure 8.4: Maxima Values corresponding to three different scales (finer scale at the top and coarser scale at the bottom).

Figure 8.5: Smooth Implicit Function reconstructed from multiscale edges

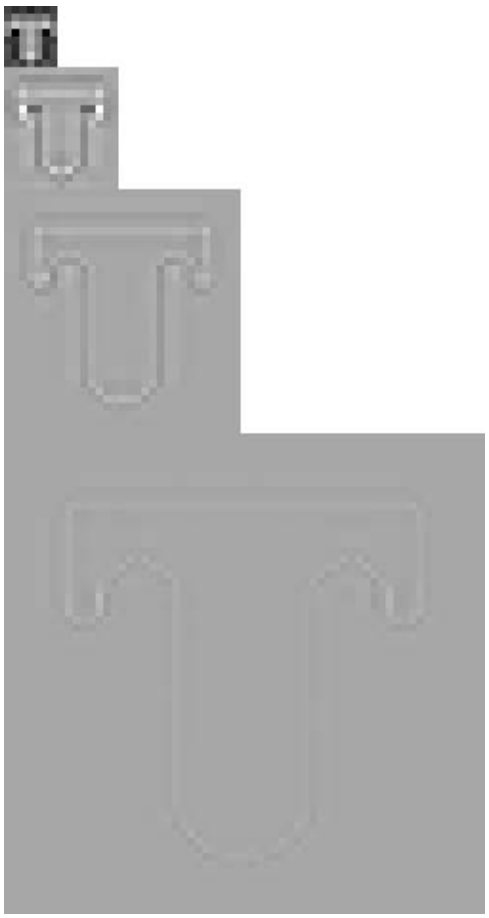


Figure 8.6: B-spline Pyramid of the Implicit Function

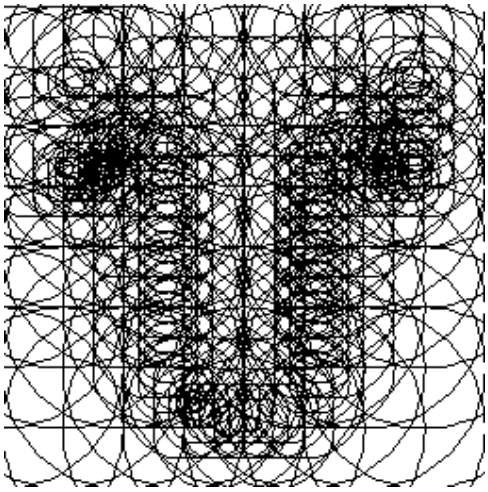


Figure 8.7: B-spline Functions

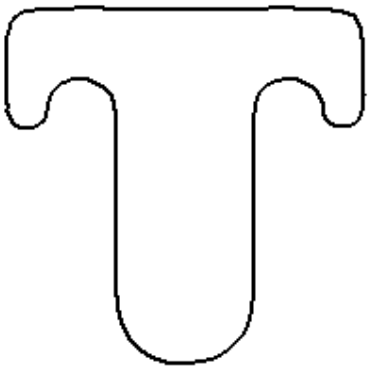
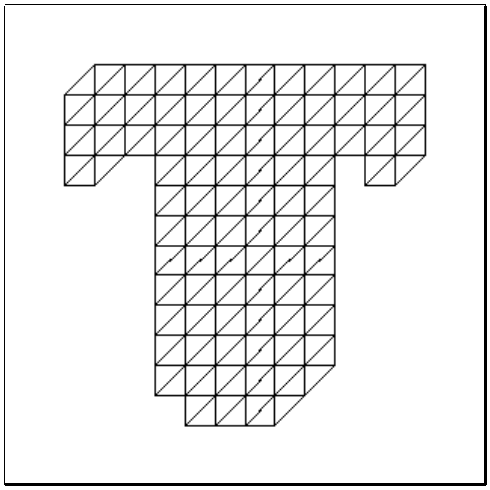
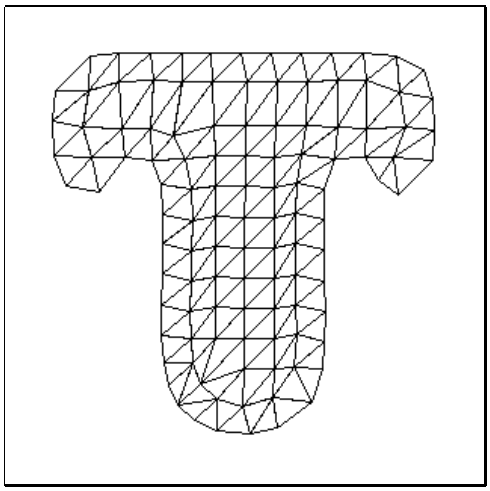


Figure 8.8: Boundary derived from the Multiscale Representation



(a)



(b)

Figure 8.9: Initial (a) and Final (b) Mesh for Solid Shape

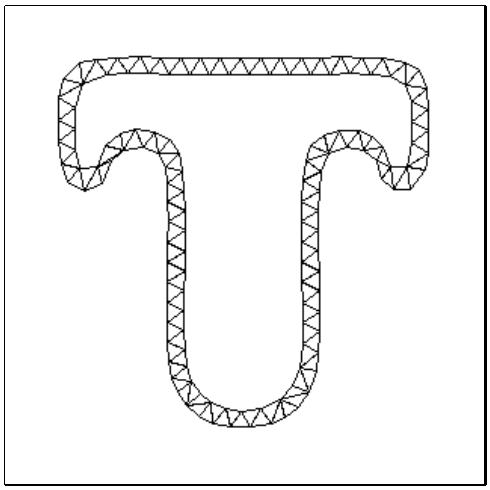


Figure 8.10: Final Mesh for Hollow Shape

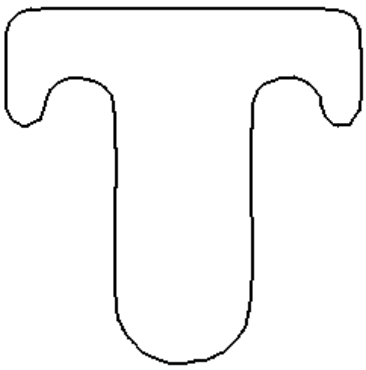


Figure 8.11: Boundary derived from the Mesh above

Appendix A

Functional Analysis

This appendix gives an overview of Functional Analysis introducing some of the main concepts of this discipline required to understand the material in the following appendices.

A.1 Introduction

Functional analysis builds upon the classical branches of mathematical theory, such as analysis, algebra and geometry, to provide new methods for the solution of many problems in pure and applied mathematics. Moreover, it makes possible a unified understanding of the key concepts from these domains in a more general setting. The advance brought by functional analysis influenced the development of new areas such as the theory of real and complex functions, approximation theory, qualitative theory of integral and differential equations, among others.

The essential contribution of functional analysis is the creation of a higher level of abstractions for mathematical objects. While classical analysis studies the properties of individual functions, functional analysis investigates whole collection of functions characterized by some property. While classical algebra deals with operations among variables that represent magnitudes, or numbers, in functional analysis the variables are functions themselves. While classical geometry is restricted to finite dimensional spaces, functional analysis extends this notion to infinite dimensional spaces.

An example will serve to clarify this point.

Example A.1 (Elastic String) Let us consider the problem of studying a simple physical system: the oscillations of an elastic string. Let AB be a weightless flexible string connecting points A and B (See Figure A.1).

Assume that a weight is attached to a point C on the string, as shown in Figure A.1(a). If C is moved from its equilibrium position, the string begins to oscillate. The state of the system at time t is described by the displacement y of the point C from its initial position. Now let us place n weights on the string at the points $C_1, \ldots, C_n$, as shown in Figure A.1(b). The state of the system is given by the displacements $y_1, \ldots, y_n$ of the weights, which can be regarded as a vector $(y_i), i = 1, \ldots, n$ in a n-dimensional space. Such a geometric interpretation is very useful in investigating the motion of this system. Finally, let us imagine a homogeneous string whose mass

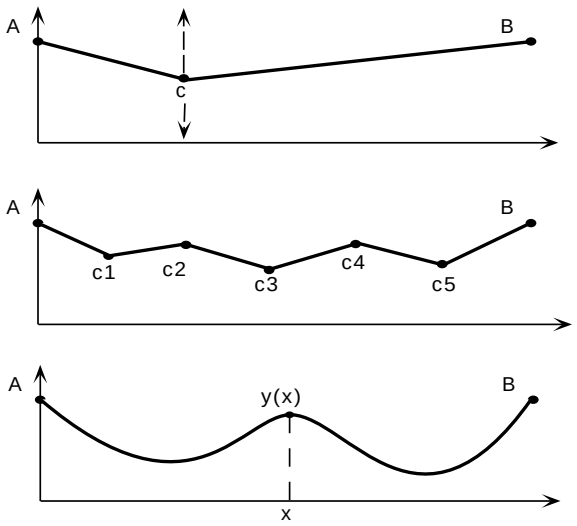


Figure A.1: Elastic String

is continuously distributed along AB, as shown in Figure A.1(c). The state of this system can no longer be described by a finite set of numbers y_i, instead, the displacement $y(x)$ of every point $x \in [A, B]$ has to be given. In other words, the state is represented by a continuous function y on the interval $[A, B]$. It turns out that y can be viewed as a vector on a infinite-dimensional space. This is natural, since when the number of weights C_i is increased, the distance between them tends to zero and in the limit a continuous distribution of mass along the string is obtained.

The above example showed how one can define mathematical objects that represent other objects of arbitrary nature, such as an elastic string. As we have seen, we work with spaces in which the elements describe general properties of objects, such as physical quantities. Furthermore, these elements do not need to be discrete, they may be continuous. Functional analysis provides a rigorous mathematical framework to operate at such general and abstract level.

The methodology of functional analysis has many applications in graphics and modeling. Some examples are the study of shapes, images, and color.

A.2 Mathematical Structures

In this section we describe some of the mathematical structures that may be imposed on the elements of an abstract space.

A.2.1 Linear Structure

A linear structure on a space X consists of the operations of addition and multiplication by a scalar. A scalar is, in general, an element of the field of real numbers $\mathbb{R}$ or of the field of complex numbers $\mathbb{C}$. We will denote a scalar by $\mathbb{F}$. The elements $x \in X$ are called vectors.

For any $x, y \in X$ and $\alpha \in \mathbb{F}$, we define the operations of

vector sum, $x + y$, and

product of a scalar by a vector, αx.

These operations must obey associative laws with respect to both vector sum and product of a vector by a scalar. They relate to each other according to the the distributive laws, as

$$\alpha(x + y) = \alpha x + \alpha y$$

$$(\alpha + \beta)x = \alpha x + \beta x$$

Furthermore, for every $x, y \in X$, there is a $z \in X$ such that $x + z = y$. This implies that there exists for every $x \in X$ an element $z_x \in X$ such that $x + z_x = x$. It turns out that there is exactly one element $z \in X$ satisfying this property. This element is called *zero vector* and is denoted by 0.

A space with a linear structure is called a *vector space*.

A.2.2 Topological Structure

A topological structure Ω on a space X is a family Ω of open sets $O_\gamma \subset X$,

$$\Omega = \{O_\gamma, \gamma \in \Gamma\}$$

such that

1. $\emptyset \in \Omega$ and $X \in \Omega$
2. $O_{\gamma_i} \cap O_{\gamma_j} \in \Omega$, for any O_{γ_i}, $O_{\gamma_j} \in \Omega$
3. $\bigcup_{i=0}^{\infty} O_{\gamma_i} \in \Omega$, for $O_{\gamma_i} \in \Omega$

From (2) it follows that the intersection of a finite number of open sets is open, and (3) states that a denumerable union of open sets is open.

The existence of these open sets O_γ imply that is possible to define the notion of a *neighborhood* for every element $x \in X$. A point $x \in X$ is called an interior point of a set $S \subset X$ if there exists an open set O in X such that $x \in O \subset S$. A *neighborhood* of a point $x \in X$ is a set $V \subset X$ of which x is an interior point.

A set S is *closed* if its complement $\overline{S}$ is open.

The topological structure is of fundamental importance to the definition of convergence, a concept that will be discussed in Section A.3.

A vector space endowed with a topological structure is called a *topological vector space*.

A.2.3 Metric

A metric on a space X is a real-valued function $d(x, y)$, defined on pairs of elements $x, y \in X$

$$d : X \times X \to [0, \infty),$$

satisfying the following properties

1. $d(x, y) = 0$ iff $x = y$
2. $d(x, y) = d(y, x)$
3. $d(x, z) \leq d(x, y) + d(y, z)$

The metric measures the "distance" between two elements of the space X.

A space with a metric is called a *metric space*. Metric spaces are a natural generalization of the space of real (complex) numbers.

A.2.4 Norm

A norm defined on a vector space X is a real-valued function

$$\|\cdot\| : X \to [0, \infty),$$

satisfying the following properties, for any $x \in X$ and $\alpha \in \mathbb{F}$

1. $\|x\| > 0$ for $x \neq 0$
2. $\|\alpha x\| = |\alpha| \|x\|$
3. $\|x + y\| \leq \|x\| + \|y\|$

The norm captures the notion of size of vectors in an abstract vector space.

A vector space equipped with a norm is called a *normed vector space*.

A.2.4.1 P-Norms

There are, in general, many possible norms that can be defined on a given vector space. An important instance is the family of p-norms.

For example, on an n-dimensional Euclidean space the p-norm is defined as

$$\|x\| = \left[\sum_{i=1}^{n} |x_i|^p \right]^{1/p}.$$

Particularly interesting are the cases when p is equal to $1, 2$ and ∞.

A.2.5 Inner Product

An inner product on a (real) vector space X is a function

$$< \cdot, \cdot >: X \times X \to \mathbb{R}$$

with the properties:

1. $< x, x > \geq 0$
2. $< x, y > = < y, x >$
3. $< \alpha x + \beta y, z > = \alpha < x, z > + \beta < y, z >$

From the last two properties we conclude that the inner product *bilinear*.

A vector space with inner product is called an *inner product vector space*.

A.3 Properties of a Space

In this section we discuss some properties associated with abstract spaces.

A.3.1 Dimension

The dimension n of a vector space is the maximal number of linearly independent elements in it. When $n < \infty$ we say that the space is finite dimensional. When $n = \infty$, we say that the space is infinite dimensional.

When dealing with spaces of infinite dimension it is necessary to be very careful because the notions are not as intuitive as in finite dimensions and some additional complications may arise.

For example, it is easy to see that any element x of a finite n-dimensional vector space X may be represented as a combination of a maximal set of linearly independent elements $\{y_i\}$, $i = 1, \ldots, n$, $y_i \in X$:

$$x = \sum_{i=1}^{n} a_i y_i,$$

where $a_i \in \mathbb{R}$.

On the other hand, it is not at all clear that the same applies to infinite dimensional spaces. In general, in order to study infinite dimensional spaces, we must impose some extra structure. We will come back to this topic later.

A.3.2 Convergence

The concept of convergence requires at least the existence of a topological structure in order to be defined.

A sequence $\{x_i\}_{i\in I}$, of elements of a topological space X is *convergent* to $x \in X$ if for each neighborhood V of x, there exists $i_V \in I$ such that $x_i \in V$ for $i \geq i_V$.

Convergence is indicated by $x_i \to x$ or $x = \lim x_i$.

The above concept also applies, in particular, to metric and normed spaces. In these cases, we translate the definition respectively, in terms of the distance function d and the norm $\| \|$. In a metric space, we say that a sequence $\{x_i\}$ of elements of X converges to some $x \in X$ if for every $\epsilon > 0$ there exists a number N such that for every $i \geq N$

$$d(x_i, x) < \epsilon$$

In a normed space, the definition is similar. Two different norms are *equivalent* if they define the same convergence.

A.3.3 Cauchy Sequence

A sequence of vectors $\{x_i\}$ in a metric space X is called a Cauchy sequence if there exists for every $\epsilon > 0$ a number N such that

$$d(x_j, x_i) < \epsilon,$$

for all $i, j > N$.

Note that every convergent sequence is a Cauchy sequence. It is also clear that every Cauchy sequence converges and is bounded.

The concept of a Cauchy sequence makes it possible to define in abstract spaces the equivalent of many important properties of the real and complex numbers.

A.3.4 Closure and Compactness

The *closure* of a subset $S \subset X$, denoted $\overline{S}$, corresponds to the intersection of all closed subsets containing S.

In a normed space, the closure of S is the set of limits of all convergent sequences of elements of S.

$$\overline{S} = \{x \in X : x_i \to x \text{ for all } x_i \in S\}.$$

A subset S of a normed space X is called *compact* if every sequence $\{x_i\}$ in S contains a convergent subsequence whose limit belongs to S.

A subset $S \subset X$ is called *dense* in X if $\overline{S} = X$.

A.3.5 Completeness

A metric space X is complete if and only if every Cauchy sequence in this space converges. Or more generally, every convergent sequence in X, $(x_n)_{n\in\mathbb{Z}}$, always converge to an element of X.

$$\lim_{k\to\infty} d(x, x_n) = 0, \quad x \in X.$$

When a space is complete, the limit points of all convergent sequences are in the space. This guarantees that there is always an element in the space for a sequence to converge to.

Note that completeness is not a topological concept, but a metric concept.

Given a metric space X, there exists a metric space Y, such that $X \subset Y$ and the closure $\overline{X} = Y$. The space Y is called the *completement* of the space X.

From the above, it follows that

Any closed subset of a complete space is also complete.

Every complete set is closed.

A.4 A Hierarchy of Spaces

In Section A.2, we described mathematical structures that that can be imposed on vector spaces. Although it may not be apparent, these structures form a containment hierarchy. This means that we have in fact nested classes of abstract spaces, i.e., spaces belonging to more specific classes are also part of the more general classes.

A.4.1 Causal Relations

Going from most specific to most general, we will show that these structures have indeed a causal relation as illustrated in Figure A.2.

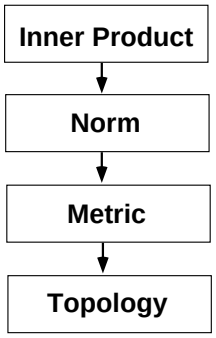


Figure A.2: Causal Relations

Given an inner product $< \cdot, \cdot >$, we can define a norm by

$$\|x\| = \sqrt{< x, x >}.$$

A space with an inner product is automatically a normed space.
Given a norm $\| \cdot \|$, we can define a metric by

$$d(x, y) = \|x - y\|.$$

A normed space is always metrizable.
Given a metric $d(\cdot, \cdot)$ we can construct a topology with the neighborhood structure defined by the open balls with center at x and radius r.

$$B(x, r) = \{d(x, y) \leq r \ ; \ y \in X\}.$$

A metric space has an intrinsic topology.

We remark that the converse of these relations does not hold in general. That is, there are spaces whose topology does not come from a metric, there are metrics that are not generated by a norm, and there are spaces whose norm does not come from an inner product.

A.4.2 Classes of Spaces

The mathematical structures defined in Section A.2 and the relations above produce the following sequence of nested spaces.

The nested sequence of abstract spaces is shown in Figure A.3.

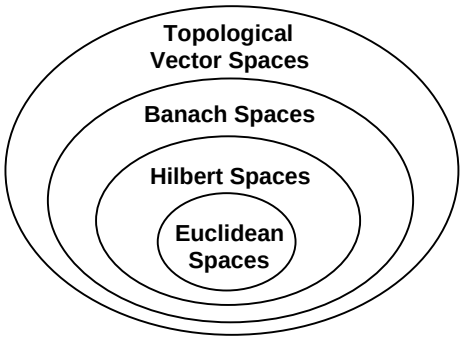


Figure A.3: Hierarchy of Abstract Spaces

A.4.2.1 Topological Vector Spaces

A vector space consists of a set X that is *algebraically closed* under a linear structure. This property means that the result of sums and scalar products must be also in X.

A vector space with a topological structure is called topological vector space.

In practice, the most general type of infinite dimensional space that we work with are the topological vector spaces. Infinite dimensional spaces without extra structure are not interesting at all. In fact, we could say that Functional Analysis studies infinite dimensional vector spaces with the addition of a topology.

A.4.2.2 Banach Spaces

A Banach space is a normed topological vector space that is complete. Note that an infinite dimensional space may be closed in relation to a given norm and not to another.

As we have seen, a closed vector subspace of a Banach space is a Banach space itself.

The fact that Banach spaces have a norm and are complete, means that the result of operations will be in the space and it is possible to quantify error. In other words, it is possible to define a Banach space of operators that is closed under approximation.

A.4.2.3 Hilbert Spaces

A Hilbert space is a Banach space whose norm comes from an inner product.

Hilbert spaces have a geometrical flavor because the inner product leads to the notion of orthogonality. Two vectors x, y of a Hilbert space X are said to be *orthogonal*, denoted $x \perp y$, if

$$< x, y >= 0$$

Similarly, a vector $x \in X$ is orthogonal to a set $S \subset X$, if $x \perp y$ for all $y \in S$.

A.4.2.4 Euclidean Space

An Euclidean n-space is a finite dimensional Hilbert space of dimension n which is naturally isometric to $\mathbb{R}^n$. Furthermore, the Euclidean space has a natural system of coordinates.

At this point, it is imperative to realize how much is taken for granted when we work on Euclidean spaces.

A.4.3 Function Spaces

The elements of an abstract vector space may be of arbitrary nature, as long as they comply with the rules of the mathematical structure imposed on the space.

Some examples of vector spaces are Euclidean spaces, the space of polynomials of one variable, and the space of $n \times n$ matrices.

Function spaces are one of the most fundamental types of spaces in functional analysis. A function space is a space whose elements are functions defined as

$$f : U \to V$$

where U is a open set of a topological space and V is a vector space.

This can be made clear if we recall Example A.1. There, the object of study was a function $f : [a, b] \to \mathbb{R}$.

Example A.2 (Functions on Interval) An example of function space is the collection of all real-valued functions on the interval $[a, b]$, denoted by F. The elements of F form a vector space if addition and scalar multiplication are defined *pointwise*. More precisely, for $f, g \in F$, and each $x \in [a, b]$.

$$(f + g)(x) = f(x) + g(x)$$

$$(\alpha f)(x) = \alpha f(x)$$

Similarly, it is possible define on the space F all other mathematical structures presented in Section A.2.

Next, we present the most important spaces of functions.

A.4.3.1 $L^p(\mathbb{R})$ Spaces

The spaces $L^p(\mathbb{R})$ play an important role in functional analysis. They are Banach spaces of functions $f : \mathbb{R} \to \mathbb{F}$ with the L^p norm.

For $1 \leq p < \infty$, the expression

$$\|f\|_p = \left[\int_{\mathbb{R}} |f(x)|^p dx\right]^{1/p}$$

must be well defined and finite.

For $p = \infty$

$$\|f\| = \sup_{x \in \mathbb{R}} |f(x)|$$

Note that for the p-norms with $p < \infty$, the integral of f must exist. This leads to the question of which integral is used (which relates to the problem of completeness).

A.4.3.2 $L^2(\mathbb{R})$ Space

An important particular case of the L^p spaces is when $p = 2$. In this case, the space $L^2(\mathbb{R})$ is a Hilbert space of functions over $\mathbb{R}$, for which the inner product is defined by

$$< f, g >= \left[\int_{\mathbb{R}} f(x)\overline{g(x)}dx\right]^{1/2}$$

Note that L^2 is the only Hilbert space among the Banach spaces defined by an L^p norm.

A.4.3.3 $l^2(\mathbb{Z})$ Space

Another important case is the vector space $l^2(\mathbb{Z})$ of square-summable sequences

$$l^2(\mathbb{Z}) = \left\{ (\alpha_i)_{i\in\mathbb{Z}} : \quad \sum_{-\infty}^{\infty} |\alpha_i|^2 < \infty \right\}$$

Note that we can consider the space $l^2(\mathbb{Z})$ as the space of functions $f(i) = \alpha_i$ defined on the discrete domain $\mathbb{Z}$.

A.5 Linear Operators

A mapping T between two vector spaces X and Y is linear if it satisfies

$$T(\alpha x + \beta y) = \alpha T(x) + \beta T(y),$$

for any $x \in X$ and $y \in Y$ and $\alpha, \beta \in \mathbb{R}$. If T is linear, it is traditional to write Tx.

When $X = Y$ we call T a *linear operator*.

If $y = T(x)$, then y is called the *image* of x under T and $x = T^{-1}(y)$ is called the *inverse image* of y.

A.5.1 Integral Operator

The integral operator T is defined by

$$(Tx)(s) = \int_a^b K(s,t)x(t)dt,$$

where $K : [a,b] \times [a,b] \to \mathbb{R}$ is a continuous map. The function K is called the *kernel* of the operator. The kernel is essentially an extension of the concept of a matrix to an infinite dimensional space.

A.5.2 Differential Operator

The differential operator Dx is defined by

$$(Dx)(s) = \frac{dx(s)}{ds} = x'(s)$$

Note that the operator Dx is not a continuous map.

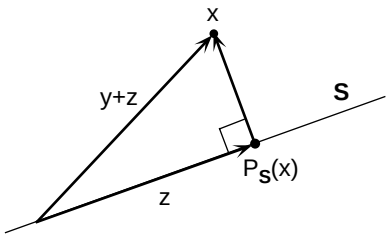


Figure A.4: Projection

A.5.3 Projection Operator

If V is a Hilbert space and $S \subset V$ is a closed subspace of V, the projection operator $P : V \to S$ is defined by

$$P_S(x) = y, \quad \text{for } x = y + z, \quad y \in S, z \in S^{\perp}.$$

This is illustrated in Figure A.4.

The linearity of the projection operator P follows from the uniqueness of the decomposition $x = y + z$.

The *identity* operator I is the projection operator onto the whole space X ($S = X$).

A.5.4 Convolution

The convolution of two functions x and y, denoted by $x * y$, is defined by

$$(x * y)(s) = \int x(s - t)y(t)dt.$$

A.6 Representation

In this section, we discuss the problem of representing the elements of an abstract space.

Basically, the problem consists in establishing an scheme to describe continuous entities (elements of an infinite dimensional space) in terms of a discrete set (a dictionary of elements in the space). This concept is analogous to the process of defining a coordinate system in a finite dimensional space.

The representation problem can also be interpreted as performing a *generalized sampling* relative to a set of basis functions.

We remark that since an exact representation is not always possible we may have to take into account the problem of approximation as well.

Next, we will present several representation schemes from the most restricted to the most general.

A.6.1 Complete Orthonormal Systems

A family of vectors $S = (x_n)$ of an inner product space is called an *orthogonal system* if, for any two distinct vectors x, y of S, $x \perp y$.

In addition, if $\|x\| = 1$ for all $x \in S$ then S is called an *orthonormal system*.

Every orthogonal set of non-zero vectors can be normalized.

$$N = \left\{ \frac{x}{\|x\|} : x \in S \right\}$$

The systems N and S are equivalent in the sense that they span the same space.

In inner product spaces, instead of finite linear combinations, infinite sums are allowed and the condition of linear independence is replaced by orthogonality.

Note that orthogonal systems are linearly independent. A finite or infinite sequence of vectors which forms an orthogonal system is called an *orthogonal sequence*.

An orthonormal sequence $\{x_n\}$ in a Hilbert space H is called *complete* if the condition $< x, x_n >= 0$ implies that $x = 0$ for all $n \in \mathbb{N}$.

A consequence of the above is that a complete orthonormal set $\{x_n\}$ in a space X gives a unique representation of the elements of X. For every $x \in X$, we have

$$x = \sum_{n=1}^{\infty} < x, x_n > x_n.$$

A.6.2 Schauder Basis

A collection of vectors (x_n) is *linearly independent* if

$$\alpha_1 x_1 + \cdots + \alpha_k x_k = 0$$

only if $\alpha_i = 0$, $i = 1, \ldots, k$.

The linear span of a subset S of a vector space X is the space formed by all finite linear combinations of vectors from S

$$\text{span } S = \{ \sum_{i=1}^{k} \alpha_i x_i : \quad x_i \in S, \quad \alpha_i \in \mathbb{R}, \quad i = 1, \ldots, k \}$$

A set of vectors $B \subset X$ is called a *Schauder basis* of X if the vectors $B = (x_n)$ are linearly independent and they span the space X.

In a infinite dimensional (normed) space we require that there exist, for all $x \in X$, a unique sequence (β_n) such that

$$x = \lim_{N \to \infty} \sum_{n=1}^{N} \beta_n x_n$$

The uniqueness requirement of the (β_n) guarantees the linear independence of the x_n.

Note that a Schauder basis is constituted by a *finite* set of vectors x_n. Moreover, the Schauder basis is a *conditional basis*, i.e. the ordering of the x_n may be important.

A.6.3 Riesz Basis

An *unconditional basis* in a Banach space X is a linearly independent set of vectors (x_n), satisfying one of the following two equivalent requirements:

$\sum_{n=-\infty}^{\infty} \alpha_n x_n \in X \Leftrightarrow \sum_{n=-\infty}^{\infty} |\alpha_n| x_n \in X$

if $\sum_{n=-\infty}^{\infty} \alpha_n x_n \in X$, then $\sum_{n=-\infty}^{\infty} \alpha_n \delta_n \; x_n \in X$ for randomly chosen $\delta_n = \pm 1$.

The order in which the basis vectors are taken does not matter for an unconditional basis. We remark that not all Banach spaces have unconditional basis. For example, $L^1(\mathbb{R})$ and $L^\infty(\mathbb{R})$ do not.

In a Hilbert space an unconditional basis is also called a Riesz basis. An equivalent characterization of a Riesz basis is based on the concept of frames.

A *frame* in a Hilbert space H is a collection of vectors x_n satisfying, for all $x \in H$

$$\alpha \|x\|^2 \leq \sum_{n=-\infty}^{\infty} | < x, x_n > |^2 \leq \beta \|x\|^2,$$

with $\alpha > 0$ and $\beta < \infty$.

A *Riesz basis* is defined as a linearly independent frame. Note that we only require that the set of vectors x_n be countable.

A.6.4 Dictionary

A *dictionary* on a Hilbert space H is a denumerable family $\mathcal{D} = (g_\gamma)_{\gamma \in \Gamma}$ of vectors in H, such that $\|g_\gamma\| = 1$.

We denote by V the closed linear span of the vectors g_γ from the dictionary. Finite linear expansions of vectors in $\mathcal{D}$ are dense in the subspace V.

A dictionary $\mathcal{D}$ is said to be *complete* if its elements span the whole space. That is, $V = H$.

Therefore, if a dictionary is complete, it can be used to represent arbitrary elements $x \in V$

$$x = \sum_{n=1}^{\infty} \alpha_n g_{\gamma_n} = \sum_{n=1}^{\infty} < x, g_{\gamma_n} > g_{\gamma_n},$$

where the coefficients α_n of the linear expansion are computed by orthogonal projection on the vectors g_γ of the dictionary $\mathcal{D}$. That is, $\alpha_n =< x, g_{\gamma_n} >$.

In general, the family $\mathcal{D} = (g_\gamma)_{\gamma \in \Gamma}$ is very redundant and to represent effectively any element $x \in V$ we must select an appropriate countable subset of vectors g_γ from $\mathcal{D}$. The goal is to find a linear expansion over the set of vectors selected from the dictionary that best match the inner structures of x.

Depending upon the choice of the vectors g_γ, the expansion coefficients provide explicit information of certain properties of the elements x of V.

A.7 Main Theorems

In this section we present some important theorems in functional analysis.

A.7.1 Fixed Point Theorem

A mapping T from a subset of a normed space $S \subset X$ is a *contraction* if there exists a positive number $\alpha < 1$ such that

$$\|T(x) - T(y)\| \leq \alpha \|x - y\|$$

for all $x, y \in S$.

Theorem A.1 (Fixed Point Theorem) *Let T be a contraction mapping from a closed subset S of a Banach space X into S. There exists a* unique *$z \in S$ such that $T(z) = z$.*

The above theorem has many important applications in the solution of problems in functional analysis.

A.7.2 Riesz Representation Theorem

A linear mapping $F(x)$ from a normed space X into a scalar field $\mathbb{R}$, (or $\mathbb{C}$), is called a (linear) *functional*.

The set of all bounded linear functionals on X, denoted by X^*, is called the *dual space* of X.

If X is an inner product space then for any fixed element $x_0 \in X$, the formula

$$F(x) =< x, x_0 >$$

defines a bounded linear functional on X.

It turns out that in a Hilbert space, every bounded linear functional is of this form.

Theorem A.2 (Riesz Representation Theorem) *Let F be a bounded linear functional on a Hilbert space H. There exists exactly one $x_0 \in H$ such that $F(x) =< x, x_0 >$ for all $x \in H$. Furthermore, $\|F\| = \|x_0\|$.*

The set H^* of all bounded linear functionals on a Hilbert space H is a Banach space. A consequence of the Riesz Representation Theorem is that H and H^* are isomorphic. In other words, for most practical purposes $H = H^*$. The element x_0 corresponding to a functional F is also called the *representer* of F.

A.7.3 Spectral Decomposition

Theorem A.3 (Spectral Theorem) *Let A be a self-adjoint compact operator on a Hilbert space H. There exists a complete orthonormal system in H consisting of the eigenvectors $\{v_n\}$ of A. Every element $x \in H$ has a unique representation in the form*

$$x = \sum_{n=1}^{\infty} \alpha_n v_n + y,$$

where y satisfies the equation $Ay = 0$.

Moreover, for every $x \in H$

$$Ax = \sum_{n=1}^{\infty} \lambda_n < x, v_n > v_n,$$

where λ_n is the eigenvalue corresponding to the eigenvector v_n.

A.8 The Fourier Transform

The Fourier transform of a function $x \in L^1(\mathbb{R})$ is defined by

$$\mathcal{F}\{x(t)\} = \hat{x}(k) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{-ikt} x(t) dt$$

The Fourier transform is a linear operator $\mathcal{F} : L^1(\mathbb{R}) \to L^1(\mathbb{R})$.

The extension of the Fourier transform to $L^2(\mathbb{R})$ is possible but somewhat involved because not all square integrable functions are integrable.

The Fourier transform is an invertible operator. The inverse Fourier transform is defined by

$$\mathcal{F}^{-1}\{\hat{x}(k)\} = x(t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{ikt} \hat{x}(k) dk$$

A.8.1 Properties of the Fourier Transform

The Fourier transform has the following properties:

$\mathcal{F}\{x(s-t)\} = \mathcal{F}\{x(s)\}e^{-ikt}$

$\mathcal{F}\{x(\alpha t)\} = \frac{1}{\alpha}\mathcal{F}\{x(\frac{t}{\alpha})\}, \quad \alpha > 0$

If $x(t)$ is a continuous piecewise n-times differentiable function, $x, x', \ldots, x^{(n)} \in L^1(\mathbb{R})$ and $\lim_{|x|\to\infty} x^{(j)}(t) = 0$ for $j = 0, \ldots, n-1$, then

$$\mathcal{F}\{x^{(n)}\} = (i)^n k^n \mathcal{F}\{x\}$$

Let $x, y \in L^1(\mathbb{R})$. Then

$$\mathcal{F}\{x * y\} = \sqrt{2\pi}\mathcal{F}\{x\}\mathcal{F}\{y\}$$

Theorem A.4 (Parseval Identity) *Let $x \in L^2(\mathbb{R})$ be a continuous function vanishing outside a bounded interval on $\mathbb{R}$. Then $\hat{x}$ is also in $L^2(\mathbb{R})$ and*

$$\|\hat{x}\|_2 = \|x\|_2$$

In physical problems, $\|x\|_2$ is a measure of energy and $\|\hat{x}\|_2$ is the power spectrum of x.

A.8.2 The Fourier Series

The Fourier series of a 2π periodic function x is defined by

$$x = \sum_{k=-\infty}^{\infty} \alpha_k \phi_k$$

where

$$\phi_k(t) = \frac{e^{ikt}}{\sqrt{2\pi}}$$

The numbers α_k are called the *Fourier coefficients* of x and are computed by

$$\alpha_k = \frac{1}{\sqrt{2\pi}} \int_{-\pi}^{\pi} x(t) e^{-ikt} dt$$

The sequence

$$\phi_k(t) = \frac{e^{ikt}}{\sqrt{2\pi}}, \quad k = 0, +-1, \ldots$$

is a complete orthonormal sequence in $L^2([0, 2\pi])$.

A.8.3 Fourier Transform versus Fourier Series

While the Fourier transform is defined on general Banach spaces $L^p(\mathbb{R})$, the Fourier series represents only periodic functions. The simplest way to periodize a function $f \in L^p(\mathbb{R})$ is to consider

$$\Phi_f(t) = \sum_{k=-\infty}^{\infty} f(t + 2\pi k)$$

The first question we need to ask is whether or not Φ_f is a function in L^p. This is affirmative for $p = 1$.

Lemma A.1 *Let $f \in L^1(\mathbb{R})$. Then, the series $\sum_{k=-\infty}^{\infty} f(t + 2\pi k)$ converges to some 2π-periodic function Φ_f. Furthermore $\Phi_f \in L^1([0, 2\pi])$ with*

$$\|\Phi_f\|_{L^1([0,2\pi])} \leq \frac{1}{2\pi} \|f\|_1$$

If the Fourier series of Φ_f converges to Φ_f, then these two quantities can be equated. The problem is that Φ_f is only in $L^1([0, 2\pi])$, and its Fourier series may diverge. In order to ensures that equality is valid we have to impose some conditions on Φ_f.

Theorem A.5 *Let $f \in L^1(\mathbb{R})$ satisfy the conditions*

1. *The series $\sum_{k=-\infty}^{\infty} f(t+2\pi k)$ converges everywhere to some continuous function.*
2. *The Fourier series $\frac{1}{2\pi}\sum_{k=-\infty}^{\infty} \hat{f}(k)e^{ikt}$ converges everywhere.*

Then, the Poisson Summation Formula *holds*

$$\sum_{k=-\infty}^{\infty} f(t + 2\pi k) = \frac{1}{2\pi} \sum_{k=-\infty}^{\infty} \hat{f}(k)e^{ikt}$$

for all $t \in \mathbb{R}$.

In order to comply with the conditions (1) and (2) of the above theorem a function and its Fourier transform must have sufficient decay.

Lemma A.2 *Let f be a mensurable function, and*

$$f(u), ; \ \hat{f}(u) = O\left(\frac{1}{1 + |u|^{\alpha}}\right)$$

for some $\alpha > 1$. Then, the Poisson summation formula applies to f.

Appendix B

Wavelets

In this chapter we introduce the theory of wavelets and the computational methods associated with it.

Wavelets constitute a recent development in applied mathematics. The theory of wavelets is the synthesis of many concepts originated in engineering, physics and mathematics. As a consequence, it provides a rigorous framework for applications in these areas.

The central theme of wavelets is the description of a function in terms of simple building blocks (or atoms). Such atomic decompositions result in an effective representation of complex phenomena and allow an efficient numerical solution in applications.

The wavelet representation is a linear expansion on elements of a family of functions that are based on a single function called the *mother wavelet*. These elementary blocks are generated by scaling and translating the mother wavelet. We will see that in order to be able to describe an arbitrary function in $L^2(\mathbb{R})$ the wavelets must have the following properties:

- oscillation
- fast decay

Therefore, such functions are "small waves" (in French *ondelette*). This is the origin of the name wavelet.

B.1 Classes of Wavelets

In this section we present the main classes of wavelets and the operators associated with them.

B.1.1 Continuous Wavelet

In its most general form, as mentioned above, the wavelet representation is based on a single function $\psi \in L^2(\mathbb{R})$, called mother wavelet (or simply wavelet). This basic

function originates a family $\{\psi_{a,b}\}$ of functions

$$\psi_{a,b}(t) = \frac{1}{\sqrt{|a|}}\psi\left(\frac{t-b}{a}\right),$$

where $a, b \in \mathbb{R}$, and $a \neq 0$. The parameters a and b vary continuously, corresponding respectively to dilations and translations of ψ.

The wavelet transform of a function $f \in L^2(\mathbb{R})$ maps a continuous function of one variable (space) to a continuous function of two variables (scale and location).

$$(T_\psi f)(a,b) = \int_{-\infty}^{\infty} f(t)\overline{\psi_{a,b}(t)}dt = < f, \psi_{a,b} > .$$

The operator $(T_\psi f)(a,b)$ is, therefore, the orthogonal projection of f onto the elements of the family $\{\psi_{a,b}\}$ (up to a normalization). The wavelet transform gives the correlation of f with ψ shifted by b and scaled by a.

The representation of a function by its continuous wavelet transform is redundant and the inverse transform may not be unique.

It can be shown that in order for the wavelet operator to be invertible, the mother wavelet must satisfy the *admissibility condition*

$$C_\psi = \int_{-\infty}^{\infty} \frac{|\hat{\psi}(\omega)|^2}{|\omega|}d\omega$$

where $\hat{\psi}$ is the Fourier transform of ψ.

The inverse of the continuous wavelet transform is then given by the equation

$$f(t) = \frac{1}{C_\psi}\int_{-\infty}^{\infty}(T_\psi f)(a,b)\psi_{a,b}(t)\frac{da\ db}{a^2}$$

The admissibility condition implies that $\hat{\psi}(0)$ has to be 0, and, hence ψ has to oscillate. This is equivalent to

$$\int_{-\infty}^{\infty}\psi(t)dt = \hat{\psi}(0) = 0$$

Also, if $\hat{\psi}$ is continuous at zero, ψ has to decay fast (for example $|\psi(t)| = O(t^{-2})$ as $|t| \to \infty$).

Not every function $(Tf)(a,b)$ is the continuous wavelet transform of a function f. The function $(T_w f)(a,b)$ must satisfy the *reproducing kernel* equation in order to be a valid wavelet transform. For all $(a',b') \in \mathbb{R} - \{0\} \times \mathbb{R}$

$$(T_\psi f)(a',b') = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}(T_\psi f)(a,b)K(a,a',b,b')\frac{da\ db}{a^2}$$

The function

$$K(a,a',b,b') = \frac{1}{C_\psi}\int_{-\infty}^{\infty}\psi_{a,b}(t)\psi_{a',b'}(t)dt$$

is called the *reproducing kernel* and it express the redundancy between $(T_\psi f)(a,b)$ and $(T_\psi f)(a',b')$, for any pairs (a,b) and (a',b').

We remark that there are other ways to define the continuous wavelet transform. For example, the space of parameters (a,b) can be restricted to $\mathbb{R}^+ \times \mathbb{R}$. Nonetheless, this is not essentially different than what was presented above.

The continuous wavelet transform is a very useful tool for the detection and characterization of singularities of functions.

B.1.2 Wavelet Frames

It is often desirable have a discrete representation of a continuous function. This motivates the need for a discretization of the continuous wavelet transform. The goal is to sample the wavelet transform of a function f at a discrete set of points (a,b) in its parameter space $\mathbb{R}\times\mathbb{R}$ such that those values still characterize the function f.

If the (a,b) plane is discretized by restricting the dilation parameter to $a = a_0^m$, with $m \in \mathbb{Z}$ and the translation parameter to $b = nb_0a_0^m$, with $n \in \mathbb{Z}$, then, the discrete wavelet transform becomes a map from $L^2(\mathbb{R})$ to $l^2(\mathbb{Z}^2)$ with basis elements of the form

$$\psi_{m,n}(t) = \frac{1}{\sqrt{a_0^m}}\psi\left(\frac{t - nb_0a_0^m}{a_0^m}\right) = a_0^{-m/2}\psi(a_0^{-m}t - nb_0)$$

We are now faced with two questions:

1. Is it possible to reconstruct a function f from the coefficients $< f, \psi_{m,n} >$?
2. Is there an algorithm to compute the coefficients of the discrete wavelet transform?

If any function $f \in L^2(\mathbb{R})$ can be written as a superposition of $\psi_{m,n}$, then these basis elements are called *atoms* and the representation is called *atomic decomposition*.

The two questions above are *dual* aspects of the same problem. We want to find a solution of the form

$$f = \sum_{m,n} < f, \psi_{m,n} > \widetilde{\psi_{m,n}}$$

In order to meet the above requirements the wavelet family $\{\psi_{m,n}\}$ has to constitute a *frame*.

A family of functions $\{\varphi_k\}_{k\in K}$ is a frame of a Hilbert space H if for all $f \in H$, there exists $0 < A < B < \infty$ such that

$$A\|f\|^2 \leq \sum_{k\in K} | < f, \varphi_k > |^2 \leq B\|f\|^2$$

The numbers A and B are called *frame bounds*.

The frame operator $F : f \to \{< f, \varphi_k >\}_k$ is invertible because it is bounded.

When $A = B$, we call the frame a *tight frame*. In this case, the inversion formula is

$$f(t) = \frac{1}{A} \sum_k < f, \varphi_k > \varphi_k(t)$$

If the vectors $\{\varphi\}$ constituting a tight frame are normalized, then $A = B$ indicates the redundancy of the frame. For $A = B = 1$ the frame is an orthonormal basis.

When the frame is not tight the reconstruction formula becomes

$$f(t) = \sum_k < f, \varphi_k > \widetilde{\varphi_k(t)} = \sum_k < f, \widetilde{\varphi_k} > \varphi_k(t)$$

and we must determine $\widetilde{\varphi_k(t)}$.

This computation amounts to

$$\widetilde{\varphi_k}(t) = (F^*F)^{-1}\varphi_k(t)$$

where

$$(F^*F)f = \sum_k < f, \varphi_k > \varphi_k$$

The amount of computation necessary to determine $\widetilde{\varphi_k}$ to a desired accuracy is proportional to the ratio B/A.

There are some appropriate choices of ψ, a_0, and b_0 such that we have

$$\widetilde{\psi_{m,n}}(t) = \widetilde{\psi}_{m,n}(t) = a_0^{-m/2}\widetilde{\psi}(a_0^{-m}t - nb_0)$$

for a single $\widetilde{\psi}$. This function $\widetilde{\psi}$ is called the *dual* of the wavelet ψ.

B.1.3 Orthonormal Wavelets

When a family of wavelets constitute an orthonormal basis of $L^2(\mathbb{R})$ the associated wavelet representation has no redundancy. The parameters space (a, b) is critically sampled and in some sense the wavelet transform of a function f is the most compact atomic decomposition of f.

It is possible to construct orthonormal basis of wavelets by restricting the parameters a and b to dyadic scalings and shifts in the following way:

$a = 2^{-j}$

$b = \frac{k}{2^j} b_0$

for $j, k \in \mathbb{Z}$. Here b_0 is called the *sampling rate*.

We now have the *dyadic* wavelets of the form:

$$\psi_{j,k}(t) = 2^{\frac{j}{2}}\psi(2^j t - k)$$

where $b_0 = 1$.

The fact the $\{\psi_{j,k}\}$ is an orthonormal basis, means that

$$\|\psi_{j,k}\| = 1, \quad j, k \in \mathbb{Z}$$

$$< \psi_{j,k}, \psi_{j',k'} >= \delta_{j,k} \cdot \delta_{j',k'}, \quad j, k, j', k' \in \mathbb{Z}$$

B.1.3.1 Orthonormalization Procedure

If the family $\{\psi_{j,k}\}$ is a Riesz basis of $L^2(\mathbb{R})$ we can construct from ψ an function $\psi^{\perp}$ which generates an orthonormal basis of $L^2(\mathbb{R})$

$$\hat{\psi}^{\perp}(\omega) = \frac{\hat{\psi}(\omega)}{\left(\sum_{k=-\infty}^{\infty} |\hat{\psi}(\omega + 2\pi k)|^2\right)^{1/2}}$$

B.2 Multiresolution Analysis

Dyadic wavelets can be used to generate a Multiresolution Analysis (MRA) of $L^2(\mathbb{R})$, or conversely, we can derive dyadic wavelets from a multiresolution analysis. The idea is to decompose $L^2(\mathbb{R})$ into a direct sum of closed subspaces W_j, spanned by the functions $\psi_{j,k}(t)$. Consequently, the complementary subspaces

$$V_j = \cdots \oplus W_{j-2} \oplus W_{j-1}, \qquad j \in \mathbb{Z}$$

form a nested sequence of subspaces whose union is dense and whose intersection is the null space. This motivates the investigation of a *scaling function* $\phi(t)$ that generates the spaces V_j, $j \in \mathbb{Z}$, in the same manner that $\psi(t)$ generates the spaces W_j, $j \in \mathbb{Z}$.

B.2.1 Definition

A multiresolution analysis of $L^2(\mathbb{R})$ is a hierarchical decomposition of that space into a sequence of closed subspaces $\{V_j\}_{j\in\mathbb{Z}}$ satisfying

$$\cdots V_{-1} \subset V_0 \subset V_1 \cdots$$

with

$$\begin{aligned} \overline{\bigcup_{j\in\mathbb{Z}} V_j} &= L^2(\mathbb{R}) \\ \bigcap_{j\in\mathbb{Z}} V_j &= \{0\} \end{aligned}$$

and

$$\begin{aligned} f(t) \in V_j &\leftrightarrow f(2t) \in V_{j+1} \\ f(t) \in V_j &\leftrightarrow f(t - 2^{-j}k) \in V_j \end{aligned}$$

The spaces V_j are called *approximation spaces*. In a multiresolution analysis, all approximation spaces are scaled versions of the central space V_0. For this, we require that there exists a function $\phi \in V_0$ of the form

$$\phi_{j,k}(t) = 2^{j/2}\phi(2^j t - k)$$

with $j, k \in \mathbb{Z}$, such that

$$\{\phi_{0,k} : k \in \mathbb{Z}\}$$

is a Riesz basis in V_0. The function ϕ is called a *scaling function* of the multiresolution analysis. A given space V_0 may have many functions that satisfy the above properties, but only one will be an *orthonormal basis* of V_0.

We denote by P_j the orthogonal projection operator onto V_j

$$P_j f = \sum_{k \in \mathbb{Z}} < f, \phi_{j,k} > \phi_{j,k}$$

This corresponds to the approximation of f at the resolution 2^j.

The fact that the limit of the sequence of spaces $\{V_j\}_{j \in \mathbb{Z}}$ is dense in $L^2(\mathbb{R})$ ensures that

$$\lim_{\mathrm{J} \to \infty} P_j f = f$$

for all $f \in L^2(\mathbb{R})$.

Although $\{\phi_{j,k}\}_{(j,k) \in \mathbb{Z}^2}$ spans the space $L^2(\mathbb{R})$, it is not a minimal spanning set, due to the nested nature of the multiresolution analysis. On the other hand, it is possible to construct an orthogonal decomposition of $L^2(\mathbb{R})$ if we exploit the complementary structure of a multiresolution analysis.

We define W_j to be the orthogonal complement of V_j in V_{j+1}. Then we have

$$V_{j+1} = V_j \oplus W_j$$

and

$$W_j \perp W_{j'}$$

if $j \neq j'$.

The spaces W_j are called *detail spaces*, in the sense that they contain the difference in information between the spaces V_j and V_{j+1}. Therefore, W_j contains the detail information needed to go from the approximation at resolution 2^j to the approximation at resolution 2^{j+1}.

Observe that, because the spaces $W_j \subset V_{j'} \perp W_{j'}$ if $j < j'$, then we have

$$V_j = V_J \oplus \bigoplus_{k=0}^{J-j-1} W_{J+k}$$

for $j > J$.

Also, the family of spaces $\{W_j\}_{\mathrm{J}\in\mathbb{Z}}$ constitute an orthogonal decomposition of $L^2(\mathbb{R})$ expressed as

$$L^2(\mathbb{R}) = \bigoplus_{j\in\mathbb{Z}} W_j$$

The spaces W_j inherit from V_j the scaling property of the multiresolution analysis

$$f \in W_j \leftrightarrow f(2^j) \in W_0$$

It is possible to find a wavelet ψ such that, for fixed j, $\{\psi_{j,k} : k \in \mathbb{Z}\}$ constitute an orthonormal basis of W_j. Therefore, the whole family $\{\psi_{j,k}\}_{(j,k)\in\mathbb{Z}^2}$ constitutes an orthonormal basis of $L^2(\mathbb{R})$.

B.2.2 Two Scale Relations

We are now faced with the problem of defining the scaling and wavelet functions ϕ and ψ. This can be done exploiting the inner structure of the multiresolution analysis.

Since $V_0 \subset V_1$, any basis element $\phi_{0,k}$ can be expressed as a linear combination of the basis elements $\phi_{1,k}$ in V_1. In particular,

$$\phi = \sum_k h_k \phi_{1,k}$$

with

$$h_k =< \phi, \phi_{1,k} >$$

Then, we have

$$\phi(t) = \sqrt{2} \sum_k h_k \phi(2t - k)$$

The above equation is called the *dilation equation* or *two scale relation*. It expresses ϕ as a weighted sum of compressed and shifted versions of itself.

By integrating the dilation equation on both sides, and using the fact that the integral of ϕ does not vanish, we see that

$$\sum_k h_k = 1$$

The scaling function is, under very general conditions, uniquely defined by the dilation equation and the normalization requirement

$$\int_{-\infty}^{\infty} \phi(t)dt = 1$$

Because $V_1 = V_0 \oplus W_0$, the wavelet basis $\psi_{0,k}$ of W_0 can also be expressed in terms of the basis of V_1

$$\psi = \sum_k g_k \phi_{1,k}$$

with

$$g_k =< \psi, \phi_{1,k} >$$

Then, we have the dilation equation for the wavelet

$$\psi(t) = \sqrt{2} \sum_k g_k \phi(2t - k)$$

As a consequence of the two scale relation above and of the admissibility condition that requires

$$\int_{-\infty}^{\infty} \psi(t) dt = 0$$

we conclude that

$$\sum_k g_k = 0$$

We have developed a characterization of the scaling function ϕ and of the associated wavelet function ψ using the multiresolution analysis. The structure of these functions is governed by the two-scale relations.

$$\begin{aligned} \phi(t) &= \sqrt{2} \sum_{k=-\infty}^{\infty} h_k \phi(2t - k) \\ \psi(t) &= \sqrt{2} \sum_{k=-\infty}^{\infty} g_k \phi(2t - k) \end{aligned}$$

Note that ϕ and ψ are uniquely determined respectively by the l^2-sequences $\{h_k\}$ and $\{g_k\}$. Also, note that ψ is defined in terms of ϕ.

The next step is to find functions ϕ and ψ satisfying the above conditions.

B.3 Construction of Wavelets

The construction of particular dyadic wavelets is best done in the frequency domain. An indication that this is the way to go is the admissibility condition for continuous wavelets which imposes the finiteness of the Fourier transform of ψ.

B.3.1 Working in the Frequency Domain

If we express the two scale relation for the scaling function ϕ in the frequency domain, we get

$$\hat{\phi}(\omega) = \frac{1}{\sqrt{2}} \sum_k h_k e^{-ik\omega/2} \hat{\phi}(\omega/2)$$

or

$$\hat{\phi}(\omega) = H(\omega/2)\hat{\phi}(\omega/2)$$

with

$$H(\omega) = \frac{1}{\sqrt{2}} \sum_k h_k e^{-ik\omega}$$

H is a 2π- periodic function in $L^2([0, 2\pi])$.

If ϕ is a scaling function that is an orthonormal basis of V_0, it forms a *partition of unity*

$$\sum_k \phi(t-k) = 1$$

for all $t \in \mathbb{R}$. Consequently, using the Poisson summation formula (see Appendix A) we have that

$$\begin{aligned} \hat{\phi}(0) &= 1 \\ \hat{\phi}(2\pi k) &= 0, \quad 0 \neq k \in \mathbb{Z} \end{aligned}$$

Since $\hat{\phi}(0) = 1$, by applying the two scale relation recursively, we get

$$\hat{\phi}(\omega) = \prod_{j=1}^{\infty} H(2^{-j}\omega)$$

This product formula allows us to construct ϕ from the sequence h_k.

As a consequence of the above we conclude that

$$|H(0)| = 1$$

$$|H(\omega)|^2 + |H(\omega + \pi)|^2 = 1$$

Furthermore, $H(\pi) = 0$ and

$$\sum_k (-1)^k h_k = 0$$

Similarly, If we express the two scale relation for the wavelet function ψ in the frequency domain, we get

$$\hat{\psi}(\omega) = \frac{1}{\sqrt{2}} \sum_k g_k e^{-ik\omega/2} \hat{\phi}(\omega/2)$$

or

$$\hat{\psi}(\omega) = G(\omega/2)\hat{\phi}(\omega/2)$$

with

$$G(\omega) = \frac{1}{\sqrt{2}} \sum_k g_k e^{-ik\omega}$$

where G is 2π-periodic.

The requirement that ψ integrates to zero together with the fact that it is defined in terms of ϕ implies that

$$G(0) = 0$$

For a orthogonal wavelet ψ

$$\sum_l |\hat{\psi}(\omega + 2\pi l)|^2 = 1$$

or

$$|G(\omega)|^2 + |G(\omega + \pi)|^2 = 1$$

From the two previous conditions we conclude that

$$G(\pi) = 1$$

The constraint that $W_0 \perp V_0$ implies that $\psi \perp \phi_{0,k}$ and

$$\int_0^{2\pi} \hat{\psi}(\omega + 2\pi l)\overline{\hat{\phi}(\omega + 2\pi l)}e^{-ik\omega}d\omega = 0$$

or, in terms of the Fourier series

$$\sum_l \hat{\psi}(\omega + 2\pi l)\overline{\hat{\phi}(\omega + 2\pi l)} = 0$$

for all $\omega \in \mathbb{R}$.

Substituting in the above equation the expressions of $\hat{\phi}$ and $\hat{\psi}$ in terms of, respectively, H and G, we obtain after regrouping the sums for odd and even l

$$G(\omega)\overline{H(\omega)} + G(\omega + \pi)\overline{H(\omega + \pi)} = 0$$

A possible choice of G satisfying the above equation is

$$G(\omega) = e^{-i\omega}\overline{H(\omega + \pi)}$$

Note that G is defined in terms of H, as expected.

From this, we can construct an orthogonal wavelet from an orthogonal scaling function by choosing

$$g_k = (-1)^k h_{-k+1}$$

B.3.2 Quadrature Mirror Filters

The functions H and G can be interpreted as the discrete Fourier transform of a pair of discrete filters. The function H is a low pass filter for the interval $[0, \pi/2]$, and the function G is a band pass filter for the interval $[\pi/2, \pi]$.

From these observations and from the definition of ϕ and ψ in the frequency domain, we conclude that the main part of the energy of $\hat{\phi}$ and $\hat{\psi}$ is concentrated respectively in the intervals $[0, \pi]$ and $[\pi, 2\pi]$.

The fact that $H(0) = G(\pi)$ and $G(0) = H(\pi)$, together with the relation $G(\omega)\overline{H(\omega)} + G(\omega + \pi)\overline{H(\omega + \pi)} = 0$ makes them a pair of filters that are complementary. These filters are called *quadrature mirror filters*. This means that the wavelet transform essentially splits the frequency space into dyadic blocks $[2^j\pi, 2^{j+1}\pi]$ with $j \in \mathbb{Z}$.

Note that is possible to construct discrete filters H and G that satisfy the above properties, but do not correspond to any functions ϕ and ψ in $\mathrm{L}^2(\mathbb{R})$ as defined in this section.

B.4 Computing the Wavelet Transform

An efficient method to compute the coefficients $\{c_{j,k}\}$ of the dyadic wavelet transform and reconstruct a function from these coefficients exploits the hierarchical structure of the multiresolution analysis. It is based on the two scale relations and the associated pair of discrete filters H and G.

We start with a function $f_J \in V_J$. The multiresolution analysis guarantees that every function $f \in L^2(\mathbb{R})$ can be approximated as closely as desired by an $f_J \in V_J$ for some $J \in \mathbb{Z}$. We assume that we have computed the inner products of f with $\phi_{J,k}$. Since $V_j = V_{j-1} \oplus W_{j-1}$, f_J has a unique decomposition

$$f_J = f_{J-1} + g_{J-1}$$

where $f_{J-1} \in V_{J-1}$ and $g_{J-1} \in W_{J-1}$. By applying this recursively N times we have

$$f_J = f_{J-N} + g_{J-N} + \cdots + g_{J-2} + g_{J-1}$$

for $f_j \in V_j$ and $g_j \in W_j$. This is called the *wavelet decomposition*.

The goal of the fast wavelet transform algorithm is to compute the above decomposition of a function and obtain the reconstruction of a function from it efficiently.

B.4.1 Algorithm

Since, $f_J \in V_J$ it can be represented as

$$(P_J f)(t) = \sum_k c_{J,k}\phi_{J,k}(t)$$

with $c_{J,k} =< f_J, \phi_{J,k} >$.

But, as we have seen,

$$(P_J f)(t) = (P_{J-1} f)(t) + (Q_{J-1} f)(t)$$

where P_j and Q_j are respectively the orthogonal projections on the spaces V_j and W_j.

Therefore,

$$< f_J, \phi_J >=< f_J, \phi_{J-1} > + < f_J, \psi_{J-1} >$$

The efficiency of the fast wavelet transform algorithm is due to the fact that, because of the structure of the multiresolution analysis, the lower resolution coefficients can be computed directly from the higher resolution coefficients, and vice-versa, without resorting to the $L^2(\mathbb{R})$ inner products.

From the two scale relations we have

$$< f, \phi_{j,k} >= \sum_n \overline{h_{n-2k}} < f, \phi_{j+1,n} >$$

and

$$< f, \psi_{j,k} >= \sum_n \overline{g_{n-2k}} < f, \phi_{j+1,n} >$$

Or in terms of the coefficients $c_{J,n}$

$$c_{J-1,k} = \sum_n \overline{h_{n-2k}} c_{J,n} = \overline{H} c_{J,n}$$

and

$$d_{J-1,k} = \sum_n \overline{g_{n-2k}} c_{J,n} = \overline{G} c_{J,n}$$

Again, applying the above formulas recursively on the coarse sequence of approximation coefficients $\{c_{j,k}\}$ we get an algorithm to decompose the finer approximation coefficients into coarser approximation and detail coefficients. This is illustrated in Figure B.1.

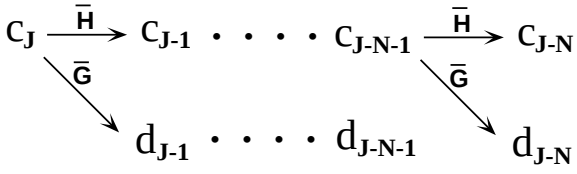


Figure B.1: Wavelet Decomposition Algorithm

Going the other way, we have the reconstruction algorithm. Using again the relation $(P_{j+1}f)(t) = (P_j f)(t) + (Q_j f)(t)$

$$f_{j+1} = \sum_k c_{j,k}\phi_{j,k} + \sum_k d_{j,k}\psi_{j,k}$$

Also

$$c_{j+1,n} =< f_{j+1}, \phi_{j+1,n} >$$

Substituting the previous equation we get

$$\begin{aligned} c_{j+1,n} &= \sum_k c_{j,k} < \phi_{j,k}, \phi_{j+1,k} > + \sum_k d_{j,k} < \psi_{j,k}, \phi_{j+1,k} > \\ &= \sum_k (h_{n-2k} c_{j,k} + g_{n-2k} d_{j,k}) \end{aligned}$$

The above formula gives an algorithm to reconstruct coefficients of finer approximations recursively from the coarse approximation and detail coefficients. This is illustrated in Figure B.2.

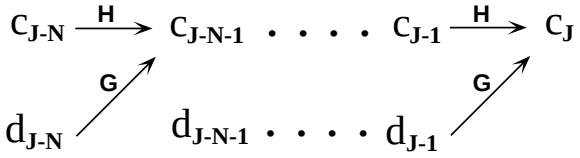


Figure B.2: Wavelet Reconstruction Algorithm

B.4.2 Connection with Subband Filtering

The wavelet decomposition and reconstruction algorithms correspond respectively to the analysis and synthesis steps of an exact subband filtering scheme. In this case we have a discrete two-channel subband filtering. The input sequence $(c_{J,k})_{k\in\mathbb{Z}}$ is convolved with a low-pass filter H and a band-pass filter G. The two resulting sequences $(c_{J-1,k})_{k\in\mathbb{Z}}$ and $d_{J-1,k})_{k\in\mathbb{Z}}$ are then subsampled, i.e., only the even (or only the odd) elements are retained. The process is repeated recursively on the result of the low pass part.

The whole scheme is illustrated in the diagram of Figure B.3.

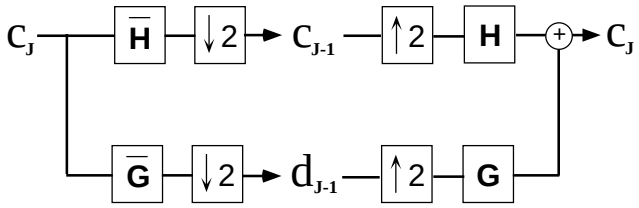


Figure B.3: Subband Filter Bank Analysis and Decomposition Scheme

B.5 Time-Frequency Analysis

In this section we discuss the localization properties of wavelets in the time and frequency domains.

The wavelet functions are localized in time (or space) and frequency because of its oscillation and decay characteristics. Consequently, the wavelet transform of a function provides valuable information about the variations of the function and their locations.

B.5.1 Time-Frequency Plane

It is convenient to analyze the wavelet transform in the time-frequency plane.

Time-Frequency Window

A function $w \in L^2(\mathbb{R})$ is a window function if it satisfies the decay requirement that $tw(t) \in L^2(\mathbb{R})$ and $|t|^{1/2}w(t)$ is also in $L^2(\mathbb{R})$, so that it is possible to identify its center and radius.

The center c_w and the radius Δ_w of the window w are defined respectively by

$$c_w = \frac{1}{\|w\|^2} \int_\infty^\infty t|w(t)|^2 dt$$

and

$$\Delta_w = \frac{1}{\|w\|^2} \left\{ \int_\infty^\infty (t - c_w)^2 |w(t)|^2 dt \right\}^{1/2}$$

The window function w has most of its energy concentrated around the mean c_w with standard deviation Δ_w. In other words, w is localized in the window

$$[c_w - \Delta_w, c_w + \Delta_w]$$

A time-frequency window is a window w that has a sufficiently fast decay such that both w and its Fourier transform $\hat{w}$ are window functions.

Theorem B.1 (Uncertainty Principle) *Let $w \in L^2(\mathbb{R})$ be such that w is a time frequency window. Then*

$$\Delta_w \Delta_{\hat{w}} \geq \frac{1}{2}$$

The uncertainty principle says that a function cannot be concentrated simultaneously in time and frequency, i.e., the function and its Fourier transform cannot be both compactly supported.

The orthogonal projection $< f, w >$ of a function f onto a time-frequency window function w gives information about the function inside the rectangular region of the time-frequency plane

$$[c_w - \Delta_w, c_w + \Delta_w] \times [c_{\hat{w}} - \Delta_{\hat{w}}, c_{\hat{w}} + \Delta_{\hat{w}}]$$

It is of fundamental importance to realize that the elements represented in the time-frequency plane must comply with the uncertainty principle. Consequently, it does not make sense to speak about a single point in the time-frequency plane.

It is customary to represent a function in the time-frequency plane by its Wigner distribution. The Wigner distribution of a function $f(t)$ denoted by $Wf(t,\omega) = W[f,f](t,\omega)$ is defined by

$$W[f,f](t,\omega) = \frac{1}{2\pi}\int f(t+\xi/2)\overline{f}(t-\xi/2)e^{-i\omega\xi}d\xi$$

This gives an energy distribution which reflects the joint time-frequency localization of the function f.

Figure B.4 shows the Wigner distribution of a function in the time-frequency plane.

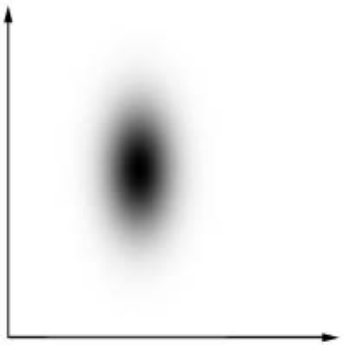


Figure B.4: Wigner Distribution of a Function

B.5.2 Short Time Fourier Transform

The standard Fourier transform of a function f gives information about the frequency content of f, but this information corresponds to oscillations of the function over the entire real line (the time domain). Therefore the Fourier transform does not provide any indication of the location of function variations, such as a high frequency burst.

Time localization can be achieved by first windowing the function f in order to cut off a localized slice of f and then taking the Fourier transform. Equivalently, we can define the *windowed Fourier transform*, also called *short time Fourier transform* (or Gabor transform), by windowing the Fourier operator itself

$$(G_w f)(\omega,s) = \int_{-\infty}^{\infty} f(t)\overline{(w(t-s)e^{-i\omega s})}dt$$

If both w and $\hat{w}$ are concentrated around zero, then $(G_w f)(\omega,s)$ can be interpreted as the content of f near time s and near frequency ω. The short time Fourier transform

produces a description of f in the time-frequency plane. The time-frequency window associated with $(G_w f)(\omega, s)$ for each point (ω, s) is defined by

$$[s - \Delta_w, s + \Delta_w] \times [\omega - \Delta_{\hat{\omega}}, \omega - \Delta_{\hat{\omega}}]$$

(assuming that $c_w = c_{\hat{\omega}} = 0$).

Note that the area of the time-frequency window is constant (equal to $4\Delta_w\Delta_\omega$) and remains unchanged for localizing a function, since the parameters (ω, s) only translate the window in the time and frequency axis respectively.

It is clear that we can discretize the short time Fourier transform by assigning to ω and s regularly spaced values $\omega = m\omega_0$ and $s = ns_0$, where $m, n \in \mathbb{Z}$ and $\omega_0 s_0 > 0$ are fixed.

$$(G_w f)(m, n) = \int_{-\infty}^{\infty} f(t)\overline{(w(t - ns_0)e^{-im\omega_0 s})}dt$$

For appropriately chosen window function w and parameters ω_0,s_0 the discrete values of $(G_w f)(m, n)$ are sufficient to completely characterize the function f.

The windowed Fourier transform induces an uniform partition of the time-frequency plane, as shown in Figure B.5.

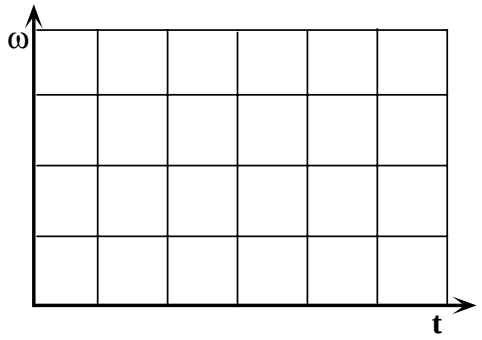


Figure B.5: Partition of the Time-Frequency Plane Induced by the Short Time Fourier Transform

The discrete windowed Fourier transform is invertible only if this tiling covers the time-frequency plane completely.

B.5.3 Wavelets

The wavelet transform

$$(T_\psi f)(a, b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} f(t)\overline{\psi(\frac{t - b}{a})}dt$$

localizes a function$f$ with a time window

$$[b + ac_\psi - a\Delta_\psi, b + ac_\psi + a\Delta_\psi]$$

where the center of the window is at $b + ac_\psi$ and its width $2a\Delta_\psi$.

On the other hand, setting $\eta(\omega) = \hat{\psi}(\omega - c_{\hat{\psi}})$, then η is a window function with center at zero and radius Δ_ψ. The wavelet transform can also be written as

$$(T_\psi f)(a,b) = \frac{a}{2\pi\sqrt{|a|}} \int_{-\infty}^{\infty} \hat{f}(\omega) e^{ib\omega} \overline{\eta(a(\omega - \frac{c_{\hat{\omega}}}{a}))} d\omega$$

using the Parseval identity.

Hence, $(T_\psi f)(a,b)$ also gives localized information of the spectrum $\hat{f}(\omega)$ with a frequency window

$$[\frac{c_\omega}{a} - \frac{1}{a}\Delta_{\hat{\omega}}, \frac{c_\omega}{a} + \frac{1}{a}\Delta_{\hat{\omega}}]$$

where the center of the window is at c_ω/a and its width is $(2\Delta_{\hat{\psi}})/a$.

We then have a time frequency window

$$[b + ac_\psi - a\Delta_\psi, b + ac_\psi + a\Delta_\psi] \times [\frac{c_\omega}{a} - \frac{1}{a}\Delta_{\hat{\omega}}, \frac{c_\omega}{a} + \frac{1}{a}\Delta_{\hat{\omega}}]$$

Note that the scaling parameter a makes the window change its shape while it is translated, but maintaining a constant area $(4\Delta_\psi\Delta_{\hat{\psi}})$. The window narrows for large center-frequency c_ω/a and widens for small center-frequency c_ω/a. This is a very desirable property in the time frequency analysis of functions.

The discrete wavelet transform can be chosen such that it covers the entire time frequency plane. The dyadic wavelet transform does a recursive binary partition of the frequency domain. Figure B.6 shows the tiling induced by the discrete wavelet transform in the time frequency plane.

B.6 Properties of Wavelets

In this section we discuss briefly some important properties of the wavelet function and their impact on applications of the wavelet transform.

B.6.1 Orthogonality

With orthogonal wavelets the fast wavelet transform yields a perfect numerical condition ensuring stable computation of the decomposition and reconstruction algorithms. If the multiresolution analysis is orthogonal, the projection operators produce optimal approximations in the $L^2(\mathbb{R})$ sense.

Orthogonality is very a desirable property, but it imposes a severe restriction on the candidate wavelet functions.

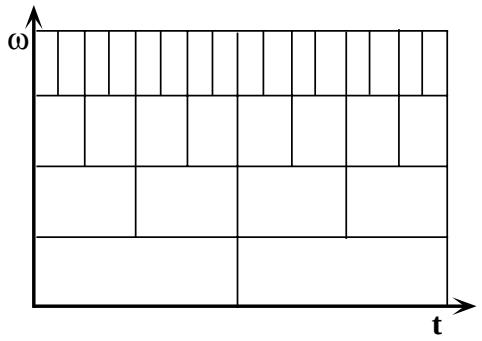


Figure B.6: Partition of the Time-Frequency Plane Induced by the Wavelet Transform

B.6.2 Support and Decay

If the scaling function and wavelet are compactly supported, the filters H and G have finite impulse response, which is desirable in the implementation of the fast wavelet transform. If these functions are not compactly supported, fast decay is desirable. In this case, the filters H and G have infinite impulse response, but can be approximated reasonably well by truncation.

B.6.3 Smoothness

Smooth basis functions are desired in applications where derivatives are involved. Smoothness also corresponds to better frequency localization of the filters.

B.6.4 Vanishing Moments

The number of vanishing moments of the wavelet is connected to its smoothness.

Vanishing moments are important in singularity detection using wavelets and in the rate of convergence of wavelet approximations of smooth functions.

B.6.5 Analytic Form

The analytic expression for the scaling function and wavelet is, in general, not available. These functions are defined indirectly through the filter coefficients $\{h\}$ and $\{g\}$.

Nonetheless, the definition of the scaling function and wavelet in analytic form is very useful in many applications.

B.7 Biorthogonal Wavelets

The orthogonality condition is a very strong constraint that limits the construction of wavelets. For example, it is not possible to create an orthogonal wavelet that has both compact support and is symmetric.

A practical solution, that allows more flexibility on the choice of wavelet functions with desirable properties, is to replace the orthogonality requirement by a biorthogonality condition.

In this case, we have a family of *biorthogonal* scaling functions and wavelets that are dual basis of the approximating and detail spaces of a multiresolution analysis.

More precisely, we define a pair of scaling functions ϕ_j and $\widetilde{\phi}_j$ that are, respectively, Riesz basis of the subspaces V_j and $\widetilde{V}_j$. Similarly we define a pair of wavelet functions ψ_j and $\widetilde{\psi}_j$ that are, respectively, Riesz basis of the subspaces W_j and $\widetilde{W}_j$.

These functions generate dual multiresolution analysis

$$\cdots \subset V_{-1} \subset V_0 \subset V_1 \subset \cdots$$
$$\cdots \subset \widetilde{V}_{-1} \subset \widetilde{V}_0 \subset \widetilde{V}_1 \subset \cdots$$

We impose that

$$V_j \perp \widetilde{W}_j \quad \text{and} \quad \widetilde{V}_j \perp W_j$$

Consequently,

$$W_j \perp \widetilde{W}_l$$

for $j \neq l$.

The above biorthogonality condition implies that

$$< \phi(t), \widetilde{\psi}(t-l) >=< \widetilde{\phi}(t), \psi(t-l) >= 0$$

and

$$< \phi(t), \widetilde{\phi}(t-l) >= \delta_l \quad \text{and} \quad < \psi(t), \widetilde{\psi}(t-l) >= \delta_l$$

Which can be extended to the multiresolution analysis by a scaling argument giving

$$< \phi_{j,k}, \widetilde{\phi}_{j,m} >= \delta_{k,m}, \quad j,k,m \in \mathbb{Z}$$

and

$$< \psi_{j,k}, \widetilde{\psi}_{l,m} >= \delta_{j,l}\delta_{k,m}, \quad j,k.l.m \in \mathbb{Z}$$

In the frequency domain, the biorthogonality condition is equivalent to

$$\sum_k \widehat{\widetilde{\phi}}(\omega + k2\pi)\overline{\widehat{\phi}(\omega + k2\pi)} = 1$$
$$\sum_k \widehat{\widetilde{\psi}}(\omega + k2\pi)\overline{\widehat{\psi}(\omega + k2\pi)} = 1$$

$$\begin{aligned}\sum_k \widehat{\widetilde{\psi}}(\omega + k2\pi)\overline{\widehat{\phi}(\omega + k2\pi)} &= 0 \\ \sum_k \widehat{\widetilde{\phi}}(\omega + k2\pi)\overline{\widehat{\psi}(\omega + k2\pi)} &= 0\end{aligned}$$

for all $\omega \in \mathbb{R}$.

This means that the filters H, G and their duals $\widetilde{H}$, $\widetilde{G}$ have to satisfy

$$\begin{aligned}\widetilde{H}(\omega)\overline{H(\omega)} + \widetilde{H}(\omega+\pi)\overline{H(\omega+\pi)} &= 1 \\ \widetilde{G}(\omega)\overline{G(\omega)} + \widetilde{G}(\omega+\pi)\overline{G(\omega+\pi)} &= 1 \\ \widetilde{G}(\omega)\overline{H(\omega)} + \widetilde{G}(\omega+\pi)\overline{H(\omega+\pi)} &= 0 \\ \widetilde{H}(\omega)\overline{G(\omega)} + \widetilde{H}(\omega+\pi)\overline{G(\omega+\pi)} &= 0\end{aligned}$$

This can be written in matrix form as

$$\widetilde{M}(\omega)\overline{M^T(\omega)} = I$$

where

$$M(\omega) = \begin{bmatrix} H(\omega) & H(\omega+\pi) \\ G(\omega) & G(\omega+\pi) \end{bmatrix}$$

and I is the identity matrix.

Interchaging the left side, we get

$$\begin{aligned}\overline{H(\omega)}\widetilde{H}(\omega) + \overline{G(\omega)}\widetilde{G}(\omega) &= 1 \\ \overline{H(\omega)}\widetilde{H}(\omega+\pi) + \overline{G(\omega)}\widetilde{G}(\omega+\pi) &= 0\end{aligned}$$

Note that this condition is equivalent to a perfect decomposition/reconstruction subband filtering scheme.

The projection operators onto the approximation and detail spaces are respectively of the form

$$(P_j f)(t) = \sum_k < f, \widetilde{\phi}_j, k > \phi_{j,k} = \sum_k < f, \phi_j, k > \widetilde{\phi}_{j,k}$$

and

$$(Q_j f)(t) = \sum_k < f, \widetilde{\psi}_j, k > \psi_{j,k} = \sum_k < f, \psi_j, k > \widetilde{\psi}_{j,k}$$

where the role of the primary functions ϕ, ψ and their duals $\widetilde{\phi}$, $\widetilde{\psi}$ can be interchanged.

From the relation of these functions with the respective filters we derive that the coefficients are

$$\widetilde{h}_{k-2l} =< \widetilde{\phi}(t-l), \phi(2t-k) >$$

and

$$\widetilde{g}_{k-2l} =< \widetilde{\psi}(t-l), \phi(2t-k) >$$

By writing $\phi(2t-k) \in V_1$ in terms of the bases of V_0 and W_0, we get the two-scale relation

$$\phi(2t-k) = \sum_l \widetilde{h}_{k-2l}\phi(t-l) + \sum_l \widetilde{g}_{k-2l}\psi(t-l)$$

The fast biorthogonal wavelet transform uses the above decomposition/reconstruction relation. The algorithm employs the two pairs of primary and dual filters and except for this difference, it is essentially similar to the orthogonal case. The pair of filters H-G is employed in the decomposition step and the pair of filters $\widetilde{H}$-$\widetilde{G}$ in the reconstruction step. Remember that, as we already noted, the roles of these two filter banks can be interchanged.

B.8 Multidimensional Wavelets

We can extend the one-dimensional wavelets presented in this appendix to higher dimensions using a tensor product formulation. In this section we will describe the two-dimensional case, the general case is analogous.

B.8.1 Tensor Product Wavelets

In this construction, we use the tensor product of two one-dimensional multiresolution analyses. As a consequence, the dilations of the resulting basis functions are applied to all the variables simultaneously.

We define the spaces $\mathbf{V}_j$, $j \in Z$

$$\mathbf{V}_0 = V_0 \otimes V_0$$

such that

$$F \in \mathbf{V}_j \leftrightarrow F(2^j x_1, 2^j x_2) \in \mathbf{V}_0$$

The sequence of spaces

$$\cdots \mathbf{V}_{-1} \subset \mathbf{V}_0 \subset \mathbf{V}_1 \cdots$$

forms a multiresolution analysis of the $L^2(\mathbb{R}^2)$ with

$$\overline{\bigcap_{\mathrm{J}\in\mathbb{Z}}} = L^2(\mathbb{R}^2), \quad \bigcup_{j\in\mathbb{Z}} \mathbf{V}_j = \{0\}$$

The orthonormal basis of $\mathbf{V}_j$ is given by the functions

$$\Phi_{j;k_1,k_2}(x_1,x_2) = 2^j \phi(2^j x_1 - k_1)\phi(2^j x_2 - k_2)$$

with $(k_1,k_2) \in \mathbb{Z}^2$.

We define, as in the one-dimensional case, the spaces $\mathbf{W}_j$ as the orthogonal complement of $\mathbf{V}_j \in \mathbf{V}_{j+1}$. Expanding the one-dimensional multiresolution analysis, we have

$$\begin{aligned}
\mathbf{V}_{j+1} &= V_{j+1} \otimes V_{j+1} \\
&= (V_j \oplus W_j) \otimes (V_j \oplus W_j) \\
&= (V_j \otimes V_j) \oplus [(V_j \otimes W_j) \oplus (W_j \otimes V_j) \oplus (W_j \otimes W_j)] \\
&= \mathbf{V}_j \oplus \mathbf{W}_j
\end{aligned}$$

Therefore, $\mathbf{W}_j$ is the direct sum of three subspaces $(V_j \otimes W_j)$, $(W_j \otimes V_j)$ and $(W_j \otimes W_j)$, with orthogonal basis given respectively by the three wavelets below

$$\begin{aligned}
\Psi^h(x_1, x_2) &= \phi(x_1)\psi(x_2) \\
\Psi^v(x_1, x_2) &= \psi(x_1)\phi(x_2) \\
\Psi^d(x_1, x_2) &= \psi(x_1)\psi(x_2)
\end{aligned}$$

The wavelet functions

$$\{\Phi^l_{j;k_1,k2} \quad : \quad k_1, k_2 \in \mathbb{Z}, \quad l = h, v, d\}$$

is an orthonormal basis of $\mathbf{W}_j$.

The whole family of wavelets $\{\Phi^l_{j;k_1,k2}\}$, $j \in \mathbb{Z}$ constitute an orthonormal basis of the entire space

$$L^2(\mathbb{R}^2) = \overline{\oplus_{j\in\mathbb{Z}} \mathbf{W}_j}$$

Note that this construction gives in two dimensions not just one but three wavelets. In the general case, the tensor product construction results in $2^n - 1$ wavelets.

B.8.2 Multidimensional Wavelet Transform Algorithm

The tensor product structure of the multiresolution analysis implies that the scaling function and the corresponding wavelet functions are separable, and so are the associated filters H and G. An important consequence of this property is that the fast wavelet transform algorithm can be implemented very efficiently as a multi-pass transformation. In the two-dimension algorithm, one-dimensional filtering is performed on rows and columns of the two-dimensional arrays. This means that at each step of the algorithm the data is processed in the horizontal direction first and in the vertical direction after that. Figure B.7 shows a schematic illustration of the two-pass wavelet transform algorithm.

Due to the subsampling, if the original function consists of an $N \times N$ array, after the first decomposition, it will be decomposed into four arrays of $N/2 \times N/2$ elements, one corresponding to the coarse resolution component and the other three corresponding to the detail components. Because the recursion is performed on the coarse resolution residual, the whole transformation can be done "in place". This scheme is depicted in Figure B.8.

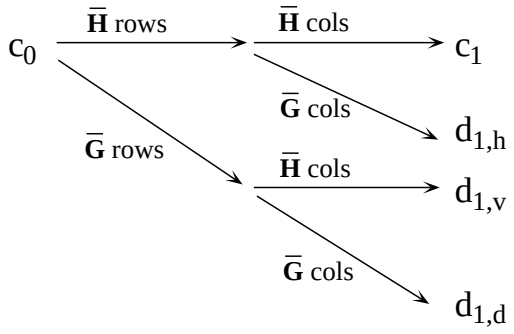


Figure B.7: Two Pass Algorithm for the Wavelet Transform

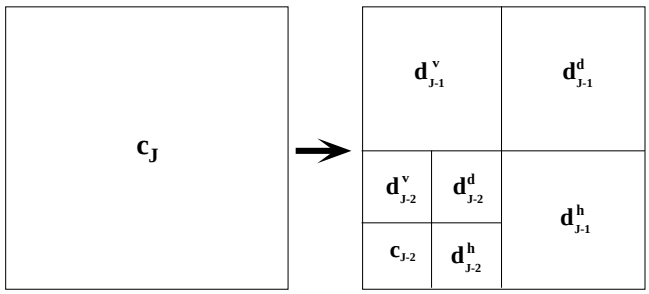


Figure B.8: n-dimensional Wavelet Decomposition

B.9 Derivative Wavelets

In this section we describe a wavelet which is defined as the derivative of a scaling function. This special type of wavelet is useful in many applications, such as edge detection and estimation of regularity of functions.

B.9.1 1-dimensional Wavelet Transform

To construct a derivative wavelet, we start with a scaling function $\phi(x)$ with the following two properties

$\int_{-\infty}^{\infty} \phi(x)dx = 1$

$\lim_{x\to\pm\infty} \phi(x) = 0$

If ϕ is differentiable, we can define the function

$$\psi(x) = \frac{d\phi(x)}{dx}$$

The function $\psi(x)$ is a wavelet because it satisfies the admissibility condition (i.e., its integral is zero)

$$\int_{-\infty}^{\infty} \psi(x)dx = 0$$

The families of scaling and wavelet functions associated with ϕ and ψ are defined respectively by

$$\phi_s(x) = \frac{1}{s}\phi(\frac{x}{s})$$

and

$$\psi_s(x) = \frac{1}{s}\psi(\frac{x}{s}) = s\frac{d\phi_s(x)}{dx}$$

where ϕ and ψ are dilated by the scaling factor s.

The wavelet transform of a function $f(x)$ at scale s and position x is defined by

$$(T_\psi f)(s,x) = f * \psi_s(x)$$

where $*$ denotes convolution.

From the definition of ψ, this is equivalent to

$$(T_\psi f)(s,x) = f * (s\frac{\phi_s}{dx})(x) = s\frac{d}{dx}(f * \phi_s)(x)$$

In other words, the wavelet transform $(T_\psi f)(s,x)$ is the first derivative of f convolved with ϕ scaled by s. This makes the this type of wavelet particularly indicated for analyzing the regularity of functions as well as detecting their variations at multiple scales.

B.9.2 N-dimensional Wavelet Transform

In order to extend the wavelet transform discussed above to n dimensions, we define n wavelet functions that are the partial derivatives of an n-dimensional scaling function ϕ

$$\psi_1 = \frac{\partial\phi}{\partial x_1}, \ldots, \psi_n = \frac{\partial\phi}{\partial x_n}$$

As before, these prototype functions are defined at scale 1. At an arbitrary scale s, the scaling function is

$$\phi_s(x_1, \ldots, x_n) = \left(\frac{1}{s}\right)^n \phi(\frac{x_1}{s}, \ldots, \frac{x_n}{s})$$

and the wavelets are

$$\psi_{s,k}(x_1, \ldots, x_n) = \left(\frac{1}{s}\right)^n \psi_k(\frac{x_1}{s}, \ldots, \frac{x_n}{s})$$

for $k = 1 \ldots n$.

The wavelet transform of a function $f \in L^2(\mathbb{R}^n)$ is a n-dimensional vector function where each component is

$$(T_{\psi,k}f)(s, x_1, \ldots, x_n) = f * \psi_{s,k}(x_1, \ldots, x_n).$$

From the definition of ψ_k, it follows that

$$\begin{pmatrix} (T_{\psi,1}f)(s, x_1, \ldots, x_n) \\ \vdots \\ (T_{\psi,n}f)(s, x_1, \ldots, x_n) \end{pmatrix} = s \begin{pmatrix} \frac{\partial}{\partial x_1}(f * \phi_s)(x_1, \ldots, x_n) \\ \vdots \\ \frac{\partial}{\partial x_n}(f * \phi_s)(x_1, \ldots, x_n) \end{pmatrix}$$

This shows that the wavelet transform is equal to $s\nabla(f * \phi_s)$, i.e. it is proportional to the gradient vector field of f convolved with ϕ_s.

For the purposes of edge detection it is convenient to convert the wavelet transform from rectangular coordinates to spherical coordinates. In this way, the wavelet transform is expressed in terms of the modulus of T_ψ

$$(Mf)(s, x_1, \ldots, x_n) = \sqrt{|(T_{\psi,1}f)(s, x_1, \ldots, x_n)|^2 + \ldots + |(T_{\psi,1}f)(s, x_1, \ldots, x_n)|^2}$$

and the spherical angles A_l, $l = 1 \ldots n-1$, that give respectively the magnitude and direction of the n-dimensional vector function T_ψ.

Figure B.9 shows the two dimensional wavelet transform of the image of a disk in rectangular coordinates and spherical coordinates.

The spherical representation is also important in studying the smoothness of a function through its wavelet transform. The local regularity of a function f must be estimated from the evolution of all components $T_{\psi,k}$, $k = 1 \ldots n$, of its wavelet transform. But, since the value of each $T_{\psi,k}$ is bounded by M, the computations need only be performed on the modulus of the wavelet transform.

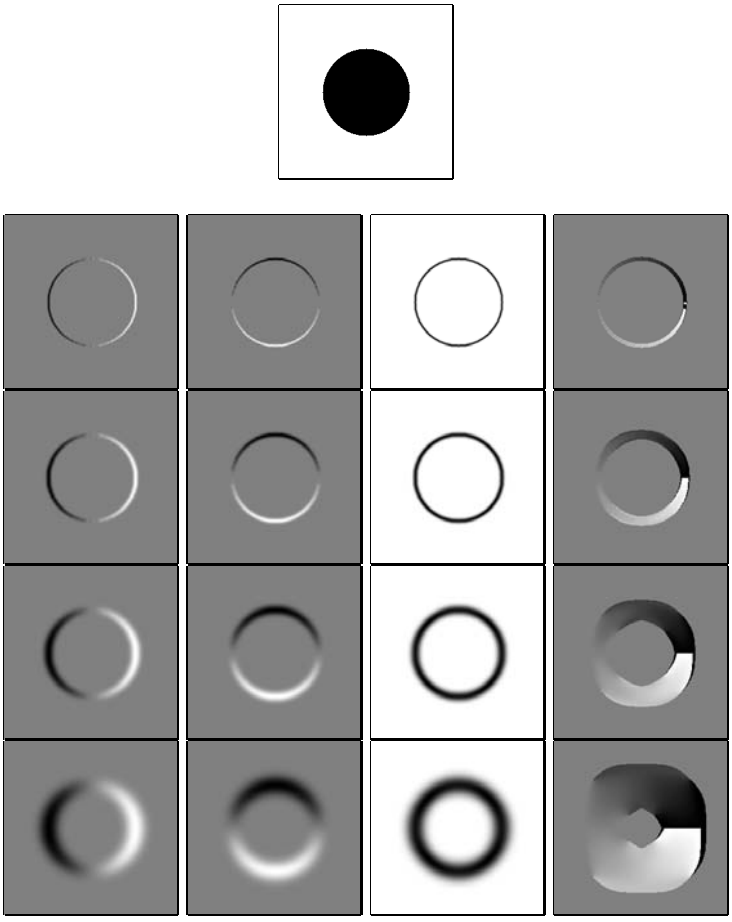


Figure B.9: 2D Wavelet Transform in Rectangular and Spherical Coordinates

Appendix C

Splines

In this chapter we review some concepts related to splines as well as the application of splines to multiresolution analysis.

C.1 Spaces of Polynomials

Algebraic polynomials are probably the best choice of functions in many problems of mathematics and its applications. They have many desirable properties that make then effective as a representation and efficient for computations. The Stone-Weierstrass theorem guarantees that every continuous function may be approximated uniformly with arbitrary precision by an algebraic polynomial.

An algebraic polynomial in n variables, $x_1, \ldots, x_n$, over a field $\mathbb{F}$ is a function

$$f(x_1, \ldots, x_n) = \sum_{i=0}^{k} a_i x_1^{p_{1,i}} \cdots x_n^{p_{n,i}}$$

where the coefficients $a_i \in \mathbb{F}$ and the exponents $p_{j,i} \in \mathbb{N}$. The degree of the polynomial is the maximum combined degree of the non-zero terms.

A polynomial scalar function in $\mathbb{R}^n$ is a map $f : \mathbb{R}^n \to \mathbb{R}$, where $f(x_1, \ldots, x_n)$ is a polynomial. A polynomial vector function in $\mathbb{R}^n$ is a map $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$, defined by m polynomial functions

$$\mathbf{f}(x_1, \ldots, x_n) = (f_1(x_1, \ldots, x_n), \ldots, f_m(x_1, \ldots, x_n))$$

C.1.1 Parametric and Implicit Descriptions

Piecewise algebraic polynomials can be used equally well in parametric and implicit descriptions.

In the parametric case, we have a function

$$f : V \subset \mathbb{R}^l \to \mathbb{R}^n$$

where f is a piecewise polynomial whose domain is the space of parameters V and whose range is the the Euclidean space $\mathbb{R}^n$. In terms of a geometric description, this

means that we have a representation given by the parametric function $f(v) = x$, which specifies the point set

$$X = \{x = f(v) : \quad v \in V\}$$

of dimension l embedded in the n-dimensional ambient space. Typically, we deal with surfaces in the three dimensional space (i.e, $l = 2$ and $n = 3$). Note that in general, f is a vector function defined on the parameter space.

In the implicit case, we have a function

$$f : U \subset \mathbb{R}^n \to \mathbb{R}^m$$

where f is a piecewise polynomial whose domain is a subset U of the ambient space $\mathbb{R}^n$ and whose range is the space $\mathbb{R}^m$. In terms of geometric description, this means that we have a representation given by the implicit function $f(x) = c$, which defines as the inverse image of the value $c \in \mathbb{R}^m$, the point set

$$X = \{x = f^{-1}(c) \quad : \quad x \in U\}$$

of dimension $n - m$ (the codimension of f) embedded in the n-dimensional ambient space. Again, the usual situation is related to surfaces in three space (i.e., $m = 1$ and $n = 3$). Note that in this case, f is a scalar function defined on the ambient space.

C.1.2 Piecewise Polynomials

Here, for simplicity and for practical reasons we will concentrate on piecewise polynomial functions.

Let $P^n([a, b])$ denote the space of all algebraic polynomials of degree at most n defined on the interval $[a, b]$.

Let $C^n(\mathbb{R})$ denote the collection of all functions f such that $f, f', \ldots, f^{(n)}$ are continuous everywere.

The space S^m of piecewise polynomials of order m with equally spaced simple knots $k \in \mathbb{Z}$ is the collection of all functions $f \in C^{m-2}$, such that the restrictions of f to any interval $[k, k + 1)$, are in $P^{m-1}([a, b))$.

Note that the space S^m is constituted of piecewise algebraic polynomials of degree $m - 1$, with continuous derivatives (i.e., $f|_{[k,k+1)} \in P^{m-1}$ and $f \in C^{m-2}$).

A direct consequence of the above definition is that an element $f_m \in S^m$ is constituted by a sequence of adjacent polynomial pieces $p_{m,k} = f|_{[k,k+1)}$, $k \in \mathbb{Z}$. In order to guarantee derivative continuity we have to enforce

$$(p^{(l)}_{m,k} - p^{(l)}_{m,k+1})(i) = 0$$

for $l = 0, \ldots, m - 2$ and $i \in \mathbb{Z}$. This means that the derivatives of contiguous pieces have to be equal up to order $m - 2$.

Furthermore, adjacent pieces $p_{m,k}$ of f_m are related by the identity

$$p_{m,k}(t) = p_{m,k-1}(t) + \frac{c_k}{(m-1)!}(t-k)^{m-1}$$

where $c_k = p_{m,k}^{(m-1)}(k) - p_{m,k-1}^{(m-1)}(k)$ (The above relation is derived from the difference of the Taylor expansion of adjacent pieces).

C.1.3 Truncated Power Basis

In order to construct a basis for the space of piecewise polynomials, we first concentrate on a interval $[-N, N]$, where $N \in Z_+$ and then extend it to $\mathbb{R}$.

If we restrict the attention to functions $f \in S^{m,N}$ defined on the interval $[-N, N]$ we have for all $t \in [-N, N]$

$$f(t) = \sum_{k=-N+1}^{N-1} \frac{c_k}{(m-1)!}(t-k)_+^{m-1}$$

where $x_+ = \max(0, x)$.

Consequently, the collection of functions

$$\{1, \ldots, t^{m-1}, (t-N+1)_+^{m-1}, \ldots, (t+N-1)_+^{m-1}\}$$

is a basis of the space $S^{m,N}$.

Since f is restricted to the interval $[-N, N]$, we can replace the monomials $1, \ldots, x^{m-1}$ with

$$(t+N)_+^{m-1}, \ldots, (t+N+m-1)_+^{m-1}$$

This defines the basis of $S^{m,N}$ in terms of the truncated powers $(t-k)_+^{m-1}$, that is generated by a single function and is independent of N.

The basis of $S^{m,N}$ can be extended to the space S^m of piecewise polynomials over the entire real line $\mathbb{R}$, by taking the union of the basis over the intervals $[-N, N]$

$$S^m = \bigcup_{N=1}^{\infty} S^{m,N},$$

Hence the family

$$\mathcal{T} = \{(t-k)_+^{m-1} \quad : \quad k \in Z\}$$

constitutes a basis of S^m.

C.2 Types of Splines

In this section, we discuss the different types of splines.

C.2.1 B-Spline

One problem with the truncated power basis is that it is formed by functions which do not belong to $L^2(\mathbb{R})$. This is due to the fact that these functions grow without bounds as $t \to \infty$ (this didn't cause problems in the construction of the basis because these functions were used only inside the intervals $[-N, N]$).

In order to construct a practical basis of S^m, we tame the polynomial growth of the functions $(t-k)_+^{m-1}$ by taking differences of elements of the truncated power basis. These new functions are made to vanish outside the intervals of interest because they cancel when combined with the other elements of $\mathcal{T}$. Note that, in spite of the fact that we are not in a Hilbert space, (finite) linear combinations are allowed in a general vector space.

Let

$$B_m(t) = \frac{1}{(m-1)!} \sum_{k=0}^{m} (-1)^k \binom{m}{k} (x-k)_+^{m-1}$$

It can be shown that the collection

$$\mathcal{B} = \{B_m(t-k) \quad : \quad k \in \mathbb{Z}\}$$

of integer translates of B_m is a Riesz basis of S^m.

The functions B_m are called *B-splines of order* m and $\mathcal{B}$ is called *B-spline basis.*

A function $f \in S^m$ is represented by its *B-spline series*

$$f(t) = \sum_{k=-\infty}^{\infty} c_k B_m(t-k)$$

where the coefficient c_k is associated with the k-th knot, of the integer sequence $k \in \mathbb{Z}$.

C.2.2 Orthogonal Spline

The B-spline basis does not constitute an orthonormal basis of S^m.

If an orthonormal basis is required, we can use the fact that the family of B-spline functions is a Riesz basis of S^m and apply the orthonormalization formula given in Appendix B to get

$$\widehat{O}_m(\omega) = \frac{\widehat{B}_m(\omega)}{\left(\sum_k |\widehat{B}_m(\omega + 2\pi k)|^2\right)^{1/2}}$$

The basis function $O(t)$ is obtained by computing the inverse Fourier transform of $\widehat{O}(\omega)$.

C.2.3 Cardinal Spline

In the B-spline representation, the values of $f(k)$, $k \in \mathbb{Z}$ approximate the knot sequence $\{c_k\}_{k\in\mathbb{Z}}$. For some applications it is desirable to have a representation which interpolates the values of its knot sequence.

With this goal in mind we can formulate a basis of S^m with interpolatory properties. We can define an m-th order *cardinal spline* function

$$L_m = \sum_{k=-\infty}^{\infty} a_k B_m(t + \frac{m}{2} - k)$$

such that $L_m(k) = \delta_k$.

In order to determine this function, we solve the bi-infinite system above in the frequency domain.

Using the fact that, because the properties of the B-spline basis, the sequence $\{a_k\}$ decays to zero exponentially as $k \to \pm\infty$, we can rewrite this system of linear equations using the z notation as

$$\sum_k c_k z^k \sum_k B_m(\frac{m}{2} + k) z_k = 1$$

or

$$\tilde{C}_m(z) = \frac{1}{\tilde{B}_m(z)}$$

where $z = e^{-i\omega}$.

Applying the Poisson Summation Formula (see Appendix A) we rewrite the above equation as

$$\tilde{B}_m(z) = \sum_{-\infty}^{\infty} \widehat{P}(\omega + 2\pi k)$$

where $(\widehat{P})(\omega)$ is the Fourier transform of $B_m(t + m/2)$.

Now, taking the Fourier transform of L_m and substituting $\tilde{C}_m$

$$\widehat{L}_m(\omega) = \frac{\widehat{P}(\omega)}{\sum_{-\infty}^{\infty} \widehat{P}(\omega + 2\pi k)}.$$

This formula can be used to compute L_m for specific values of m.

Alternatively, we can solve the linear equation above directly in the time domain. This is relatively simple for small values of m.

C.2.4 Other Splines

There are many other types of spline functions used in computer graphics. To mention a few, we have

- Hermite spline
- Bezier spline
- Beta spline

- Catmull-Rom spline

among others.

These spline functions are constructed by specifying a set of constraints on the knot sequence $\{c_k\}_{k\in\mathbb{Z}}$, similarly to the construction of the Cardinal splines. The constraints result in a system of linear equations which must be solved to define a set of basis functions. Note that, differently from the previous examples, the basis is not formed by the integer translates of a single function anymore.

C.3 Computation

Once we have defined a spline basis of the space S^m of piecewise algebraic polynomials, there are two fundamental computational operations we need to perform.

Given a function $f \in S^m$ and a spline basis $\mathcal{S}$,

1. Compute the representation of f relative to $\mathcal{S}$.
2. Reconstruct f from its representation in terms of $\mathcal{S}$.

C.3.1 Computing the Spline Series Coefficients

In order to determine the representation of a function $f \in S^m$ in the spline basis $\mathcal{S} = \{s_k : k \in \mathbb{Z}\}$ we have to find a decomposition of f in terms or the basis elements s_k. This amounts to compute the coefficient sequence $\{c_k\}$ of the spline series associated with each function s_k in the series, such that f is given by the linear expansion

$$f(t) = \sum_{k=-\infty}^{\infty} c_k s_k(t-k)$$

If the basis is orthonormal, the coefficients c_k are computed through the orthogonal projection of f onto the basis elements s_k

$$c_k =< f, s_k >$$

C.3.2 Computing F from the Spline Series

The reconstruction of a function $f \in S^m$ represented in terms of a spline basis $\mathcal{S} = \{s_k : k \in \mathbb{Z}\}$ consists in computing the value of $f(t)$ at all points $t \in \mathbb{R}$ of its domain.

The basic operation, then, consists in computing the value of $f(t)$ at a point t from its spline representation.

C.3.2.1 Direct Method

The direct computation of the value of f from the spline basis representation associated with it is the most straightforward reconstruction method.

In this method we simply substitute the parameter t into the spline series

$$f(t) = \cdots + c_{-1}s_{-1}(t+1) + c_{-0}s_0(t) + c_1 s_1(t-1) + \cdots$$

and compute directly the value of f at point t. A practical concern is the number of basis functions evaluations in the doubly infinite sum. If the spline basis functions have compact support, then the series above will have only a finite number of non-zero terms (namely, the ones corresponding to basis elements whose support contains t). When the spline basis functions do not have compact support, they usually have a sufficiently fast decay so that the series can be truncated with marginal error.

C.3.2.2 Matrix Formulation

The direct reconstruction method is simple, but not very efficient. The computation amounts to evaluating several polynomials of degree $m-1$ and then summing the resulting values. One problem with the direct method is that, because the spline basis functions are of the form $s_k(t-k)$, when we are evaluating f in a given interval $[k, k+1]$, the individual basis elements s_k have to be considered separately.

More precisely, the spline series takes the form

$$f(t) = \sum_{k=-K}^{K} c_k(a_{k,m-1}(t-k)^{m-1} + a_{k,m}(t-k)^{m-2} + \cdots + a_{k,1}(t-k) + a_{k,0}),$$

where we assume that at a point t the non-zero terms of the spline series correspond to basis elements s_k, $k = -K, \ldots K$.

This operation can be simplified, using a matrix formulation. In order to do this we need to transform the polynomials of the form $s_k(t-k)$ into polynomials of the form $s_k(t)$ This can be accomplished by a simple algebraic manipulation.

We redefine the spline series on the canonical interval $t \in [0,1]$ as

$$\begin{aligned} f(t) &= c_{-K}(\widetilde{a}_{-K,m-1}t^{m-1} + \widetilde{a}_{-K,m-2}t^{m-2} + \cdots + \widetilde{a}_{-K,1}t + \widetilde{a}_{-K,0}) \\ &+ \\ &\vdots \\ &+ \\ &\quad c_{\ K}(\widetilde{a}_{\ K,m-1}t^{m-1} + \widetilde{a}_{\ K,m-2}t^{m-2} + \cdots + \widetilde{a}_{\ K,1}t + \widetilde{a}_{\ K,0}) \end{aligned}$$

by replacing the coefficients $a_{k,l}$ of the original algebraic polynomials $s_k(t-k)$ with new coefficients $\widetilde{a}_{k,l}$ of algebraic polynomials $s_k(t)$ for $t \in [0,1]$.

The above equation can be written in matrix notation as

$$f(t) = (t^{m-1} t^m \cdots t \;\; 1) \begin{pmatrix} \widetilde{a}_{-K,m-1} & \widetilde{a}_{-K+1,m-1} & \cdots & \widetilde{a}_{K-1,m-1} & \widetilde{a}_{K,m-1} \\ \widetilde{a}_{-K,m-2} & \widetilde{a}_{-K+1,m-2} & \cdots & \widetilde{a}_{K-1,m-2} & \widetilde{a}_{K,m-2} \\ \vdots & \vdots & & \vdots & \vdots \\ \widetilde{a}_{-K,1} & \widetilde{a}_{-K+1,1} & \cdots & \widetilde{a}_{K-1,1} & \widetilde{a}_{K,1} \\ \widetilde{a}_{-K,0} & \widetilde{a}_{-K+1,0} & \cdots & \widetilde{a}_{K-1,0} & \widetilde{a}_{K,0} \end{pmatrix} \begin{pmatrix} c_{-K} \\ c_{-K+1} \\ \vdots \\ c_{K-1} \\ c_K \end{pmatrix}$$

or simply as

$$f(t) = \mathbf{t}\mathbf{A}\mathbf{c}^T$$

Note that this pre-process step needs to be performed just once for a given spline basis and it does not depend on the knot coefficients.

Obviously, to evaluate $f(t)$ at a point t in an arbitrary interval $[k, k+1]$ we first have to express it in terms of the canonical interval $[0, 1]$ as $t - k$.

The matrix form has several advantages:

- Separates the three main components $\mathbf{t}$, $\mathbf{A}$ and $\mathbf{c}$, where $\mathbf{A}$ depends on the spline basis being used and $\mathbf{c}$ contains the coefficients of the knot sequence.
- The powers of the variable t can be computed once for a given value of t.
- Special purpose hardware can be use to compute the matrix products.
- The formulation generalizes naturally to n dimensions

C.3.2.3 Forward Differences

When several values of $f(t)$ need to be computed at uniformly spaced points, the reconstruction operation can be made even more efficient by employing a forward difference polynomial (or matrix) and computing $f(t)$ incrementally.

C.3.3 Change of Basis

If two sets of functions $\mathcal{S}_1$ and $\mathcal{S}_2$ are basis of the same vector space S, it is possible to find a linear transformation that maps $\mathcal{S}_1$ into $\mathcal{S}_2$ and vice-versa.

Therefore, we can convert between different spline representation by an appropriate basis change.

C.4 The B-Spline Representation

The B-spline has a number of properties that characterize it as the standard basis for the space S^m of piecewise algebraic polynomials of order m.

C.4.1 Recursive Formulation

The B-spline basis function

$$B_m(t) = \sum_{k=0} m \frac{(-1)^k}{(m-1)!} \binom{m}{k} (t-k)_+^{m-1}$$

satisfies the property that for any value of m

$$B_m(t) = (B_0 * \cdots * B_0)(t)$$

where the convolution is applied m times.

Because B_0 is the characteristic function of the unit interval

$$B_0(t) = \chi_{[0,1)}(t) = \begin{cases} 1 & 0 \le t < 1 \\ 0 & \text{otherwise} \end{cases},$$

we can define B_m recursively from B_0 as

$$B_m(t) = (B_{m-1} * B_0)(t) = \int B_{m-1}(t-s)ds$$

C.4.2 Properties of the B-spline

The recursive definition is important because it allows us to derive useful properties of the B-spline function.

1. The B-spline function B_m is compactly supported with

   $$\text{supp } B_m = [0, m]$$

   In fact, the support is minimal among all polynomial splines of order m.

2. The B-spline function B_m is positive

   $$B_m(t) > 0$$

   for $0 < t < m$

3. The derivative of the B-spline function B_m can be defined by taking backward differences

   $$B_m'(t) = B_{m-1}(t) - B_{m-1}(t-1)$$

4. The inner product of a B-spline function with itself shifted by integers can be defined by sampling a higher order B-spline

   $$< B_m(t), B_m(t-k) >= B_{2m+1}(k)$$

   for $t \in \mathbb{R}$ and $k \in \mathbb{Z}$.

5. For every $g \in C^m$,

$$\int_{-\infty}^{\infty} g^{(m)}(t) B_m(t) dt = \sum_{k=0}^{m} (-1)^{m-k} \binom{m}{k} g(k)$$

6. The B-spline basis $\mathcal{B} = \{B_{m,k}\}$ forms a partition of unity

$$\sum_{k=-\infty}^{\infty} B_m(t-k) = 1$$

for all $t \in \mathbb{R}$.

C.4.3 B-spline in the Frequency Domain

The Fourier transform of the B-spline function has a simple formula that can be derived from its recursive definition.

$$\widehat{B}_m(\omega) = \text{sinc}^{m+1}(\omega)$$

where $\text{sinc}(\omega) = \frac{\sin \pi\omega}{\pi\omega}$.

Also, it can be shown that

$$\widehat{B}_m(\omega) = \frac{1}{2} \sum_{k=0}^{m} 2^{-m+1} \binom{m}{k} e^{-ik\omega/2} \widehat{B}_m(\omega/2)$$

a formula that will be important in the definition of the two-scale relation for the B-splines.

C.5 Splines as Scaling Functions

The spline functions can be interpreted as a scaling function associated with a multiresolution analysis. Below we develop the theory for the B-spline basis, but the same can be done for other spline basis that are generated by the integer translates of a single function, such as the Cardinal spline basis and the Orthogonal spline basis.

C.5.1 B-splines and Multiresolution Analysis

The B-spline function can be used to construct a hierarchy of nested subspaces that forms a multiresolution analysis of $L^2(\mathbb{R})$. This is given by the fine-to-coarse sequence of dyadic dilations of the basic spline space S^m.

More precisely, for a fixed m, the approximation space V_j, piecewise polynomials of order m, is defined at scale 2^j as

$$V_j = \{f_j \quad : \quad f_j(t) = \sum_{k \in \mathbb{Z}} c_k \phi_{j,k}(t), \quad c_k \in l^2(\mathbb{R})\}$$

where the basis functions of V_j are the normalized order m B-spline functions

$$B_{j,k}^m(t) = 2^{j/2} B_m(2^j t - k)$$

As we have seen, the integer translations of the B-spline function $B_m(t)$ constitute a Riesz basis of V_0. Therefore, the collection of functions $B_{j,k}^m(t)$ forms an unconditional basis of the spaces V_j.

C.5.2 Two-Scale Relation

The two-scale relation for the B-splines of order m is

$$B_m(t) = \sum_{k=0}^{m} 2^{-m+1} \binom{m}{k} B_m(2t - k)$$

So, the coefficients of the discrete filter H are given by the sequence $\{h_k\}$

$$h_k = 2^{1-m} \binom{m}{k}$$

for $k = 0, \ldots m$.

C.6 Spline Wavelets

Spline wavelets are polynomial functions whose dilations and translations constitute a basis of $L^2(\mathbb{R})$. It is possible to construct different types of spline wavelets with distinct properties. In this section, we discuss the most important spline wavelets with some emphasis on the biorthogonal B-spline wavelets.

The method of choice to construct a spline wavelet is to start with a multiresolution analysis generated by a specific spline basis function. The spline wavelet is then constructed from the spline scaling function using one of the methods discussed in Appendix B.

C.6.1 Orthogonal Spline Wavelet

The orthogonal spline wavelet is associated with the orthogonal spline scaling function specified in Section C.2.1.

In the frequency domain

$$\widehat{\phi}(\omega) = \widehat{O}(\omega)$$

Because ϕ is an orthonormal basis, it follows that ψ is given by

$$\widehat{\psi}(\omega) = e^{-i\omega/2} \overline{H(\omega/2 + \pi)} \widehat{O}(\omega/2)$$

The orthogonal wavelet functions are also called *Battle-Lemarie wavelets.*

The orthonormal scaling functions and the corresponding wavelet functions are C^k with exponential decay.

C.6.2 B-spline

The B-spline wavelet is associated with the B-spline scaling function that was analyzed in the previous section.

The B-spline functions are not orthogonal to its integer translates and constitute only a Riesz basis of the approximating spaces V_j of the multiresolution analysis. For this reason, we need to determine the B-spline wavelet as well as the corresponding duals to the scaling and wavelet functions, as discussed in Appendix B. Note that once the B-spline scaling function is fixed, there are many choices of associated wavelets and dual functions.

Here we will describe a formulation due to Ingrid Daubechies in which all the functions involved are compactly supported.

The low-pass filter H associated with the B-spline scaling function of order m is given by

$$H(\omega) = \left(\frac{1+e^{-i\omega}}{2}\right)^m$$

To find the high-pass filter G and the corresponding dual filters $\widetilde{H}$ and $\widetilde{G}$ we recall the biorthogonality condition

$$H(\omega)\overline{\widetilde{H}(\omega)} + H(\omega+\pi)\overline{\widetilde{H}(\omega+\pi)} = 1$$

Because we are looking for functions with compact support, we also impose that these functions are trigonometric polynomials of the form

$$\left(\cos\frac{\omega}{2}\right)^{2l} q(\omega)$$

if the order m of the spline is even, or

$$e^{-i\omega/2}\left(\cos\frac{\omega}{2}\right)^{2l+1} q(\omega)$$

if m is odd, and where q is a polynomial in $\cos\omega$.

Combining both conditions gives

$$(\cos\omega/2)^{2L} q(\cos\omega)\overline{\widetilde{q}(\cos\omega)} + (\sin\omega/2)^{2L} q(-\cos\omega)\overline{\widetilde{q}(-\cos\omega)} = 1$$

with $L = l + \widetilde{l}$ for m even and $L = l + \widetilde{l} + 1$ for m odd.

Defining $q(\cos\omega)\overline{\widetilde{q}(\cos\omega)} = p(\sin^2\omega/2)$, then we obtain the Bezout equation

$$(1-x)^L p(x) + x^L p(1-x) = 1$$

whose general solution is given by

$$p(x) = \sum_{n=0}^{L-1} \binom{L-1+n}{n} x^n$$

Since $H(\omega)$ is already fixed, it is straightforward to find $\widetilde{H}$. The factorization of p into q and $\widetilde{q}$ is done by setting $q \equiv 1$ and solving for $\widetilde{q}$. Then, we have

$$H(\omega) = \left(\cos\frac{\omega}{2}\right)^{2l}$$

and

$$\widetilde{H}(\omega) = \left(\cos\frac{\omega}{2}\right)^{2l} \sum_{n=0}^{l+\widetilde{l}-1} \binom{l+\widetilde{l}-1+n}{n} \left(\sin^2\frac{\omega}{2}\right)^n$$

for m even, and

$$H(\omega) = e^{-i\omega/2}\left(\cos\frac{\omega}{2}\right)^{2l+1}$$

and

$$\widetilde{H}(\omega) = e^{-i\omega/2}\left(\cos\frac{\omega}{2}\right)^{2l+1} \sum_{n=0}^{l+\widetilde{l}} \binom{l+\widetilde{l}+n}{n} \left(\sin^2\frac{\omega}{2}\right)^n$$

form m odd.

The filters G and $\widetilde{G}$ are derived from H and $\widetilde{H}$ using the biorthogonality condition

$$G = e^{-i\omega}\overline{\widetilde{H}(\omega+\pi)}$$

and

$$\widetilde{G} = e^{-i\omega}\overline{H(\omega+\pi)}$$

The scaling function, the wavelet function and their duals are defined recursively in the frequency domain as

$$\begin{aligned}
\widehat{\phi}(\omega) &= H(\frac{\omega}{2})\phi(\frac{\omega}{2}) \\
\widehat{\widetilde{\phi}}(\omega) &= \widetilde{H}(\frac{\omega}{2})\widetilde{\phi}(\frac{\omega}{2}) \\
\widehat{\psi}(\omega) &= G(\frac{\omega}{2})\phi(\frac{\omega}{2}) \\
\widehat{\widetilde{\psi}}(\omega) &= \widetilde{G}(\frac{\omega}{2})\widetilde{\phi}(\frac{\omega}{2})
\end{aligned}$$

The result is a family of biorthogonal spline basis functions with compact support. Note that, for a preassigned order m of the B-spline (i.e. fixed l), there exists infinitely many choices of $\widetilde{l}$. The function ϕ is completely determined by l alone, while the functions $\widetilde{\phi}$, ψ and $\widetilde{\psi}$ are determined by both l and $\widetilde{l}$. For increasing values of $\widetilde{l}$, we have different ψ with larger support and different $\widetilde{\phi}$ and $\widetilde{\psi}$ with more regularity.

An important question is how to find an optimal value of $\widetilde{l}$. For large m, it can be shown that $\widetilde{\psi}(t) \in C^k$ if $m > 4.165n + 5.165(k+1)$ where $n = l + \widetilde{l}$ or $n = l + \widetilde{l} + 1$ depending on the parity of m.

Recall that the coefficient sequences $\{h_k\}$, $\{\widetilde{h}_k\}$, $\{g_k\}$ and $\{\widetilde{g}_k\}$ are defined by the filters H, $\widetilde{H}$, G and $\widetilde{G}$

$$\begin{aligned} H(\omega) &= \frac{1}{\sqrt{2}} \sum_k h_k e^{-ik\omega} \\ G(\omega) &= \frac{1}{\sqrt{2}} \sum_k g_k e^{-ik\omega} \end{aligned}$$

and similarly for $\widetilde{H}$ and $\widetilde{G}$.

The sequences $\{g_k\}$ and $\{\widetilde{g}_k\}$ are derived respectively from $\{\widetilde{h}_k\}$ and $\{h_k\}$

$$\begin{aligned} g_k &= (-1)^{k+1} \widetilde{h}_{1-k} \\ \widetilde{g}_k &= (-1)^{k+1} h_{1-k} \end{aligned}$$

C.7 Space-Frequency Localization

In this section we analyze more closely the space-frequency localization properties of the spline and wavelet spline functions.

C.7.1 Uncertainty Principle Revisited

The Uncertainty Principle states that the area in the time-frequency window function w has a fixed lower bound. It says that the product of the variances (or uncertainties) in the time (Δ_w) and the frequency ($\Delta_{\widehat{w}}$) domains has to be greater or equal to $1/2$.

Moreover, it also asserts that the only function for which the equality is attained is the window function

$$w(t) = ce^{i\alpha t} g_\alpha(t-b)$$

where $g_\alpha(t)$ is the Gaussian function

$$g_\alpha(t) = \frac{1}{2\sqrt{\pi\alpha}} e^{-t^2/4\alpha}$$

with $c \neq 0$, $\alpha > 0$ and $a, b, \in \mathbb{R}$.

C.7.2 Convergence

The convolution property of the B-splines suggests that these functions converge optimally to a Gaussian as the order of the spline m tends to infinity.

The observation above indicates that the B-spline wavelet transform gives near optimal information in terms of time-frequency localization.

Appendix D

Subdivision of Space

This appendix gives an overview of the concepts related to space decompositions and discusses the data structures for their representation. The existence of certain subdivisions of a space allows us to obtain valuable information about the geometry and topology of that space. This plays an important role in the representation and computation with implicit objects. A more extensive treatment of the subject in the context of graphical applications is given in (Carvalho, Gomes and Velho, 1992).

D.1 Types of Space Decompositions

The intuitive idea of a space decomposition is to subdivide the space in a collection of disjoint connected subsets. Subdivision of the space into simpler pieces, along with a structure that links these pieces together, allows us to obtain valuable information about the geometry and topology of the space. This strategy is related to the "divide and conquer" paradigm. We get smaller, easier-to-understand pieces of the space, and structure them together in order to get information about the space as a whole. There is a trade-off involved when obtaining a decomposition of a given space: a more structured decomposition certainly will give us more information about the space, but it is harder to construct and may not even exist in general.

D.1.1 Space Partition

A *space partition* of a set U is a collection U_α, $\alpha \in I$, of subsets of U such that

- $\bigcup U_\alpha = U$;
- $U_{\alpha_1} \cap U_{\alpha_2} = \emptyset$ if $\alpha_1 \neq \alpha_2$.

Every set has a trivial space partition, that consists of the collection of all of its points. In fact it is easy to see that, for any infinite set U, there always exists an infinite number of finite partitions. Space partition is also called in the literature *space decomposition*.

A natural way to define space partitions is by using some equivalence relation defined on the points of the set U. If R is such a relation, for each point $p \in U$ we

define $U_p = \{q \in U \ ; \ (p,q) \in R\}$. If R is an equivalence relation, then it is clear that the sets U_p, $p \in U$ define a partition of the set U. Conversely, every partition of the space induces in an obvious way an equivalence relation by defining two elements to be equivalent if they are in the same set of the partition.

A partition is the most general decomposition scheme that can be used to subdivide a space. All decomposition schemes studied below are partitions with some additional structure, which may impose requirements on the geometry or topology of each set of the partition or on the relationships among these sets. This additional structure enables us to represent geometrical and topological properties of the underlying space.

D.1.2 Cell Decompositions

We now discuss a family of finite space decompositions of a set U, in which further structuring is imposed. Each set of the partition is now required to be a k-dimensional cell, that is, a set which is homeomorphic to an open disk of some dimension k. Furthermore, we require the boundary of each cell to be a (finite) union of lower-dimensional cells.

More precisely, a *cell complex* is a finite collection of subsets c_j^q (where $q = 0, 1, 2, \ldots, d$ represents the dimension of the cell and j ranges over some index set J_q) such that:

- each c_j^q is homeomorphic to the open q^{th}-dimensional disk for $q > 0$ and is a single point if $q = 0$.
- For each q in $0, 1, 2, \ldots, d$ and each j in J_q, the boundary of c_j^q is equal to the union of all lower dimensional cells that intersect that boundary.

D.1.3 Affine cell decomposition

Affine cell decompositions are examples of special cases of cell decompositions obtained by restricting the geometry of the cells. For this decomposition scheme, one requires each cell to be a convex polytope.

An *affine cell decomposition* is a cell decomposition such that every cell is affine. A cell in $\mathbb{R}^n$ is called *affine* if it is implicitly defined by the equations

$$L_i(x) < b_i, i = 1, \ldots, m,$$

where each L_i is a linear function $L_i : \mathbb{R}^n \to \mathbb{R}$.

Moreover, in order to avoid unnecessary fragmentation, each face of a cell (that is, a set obtained by turning some of the defining inequalities into equalities) must also be a cell. Note that, this concept generalizes to $\mathbb{R}^n$ the concept of a convex polygon on the plane.

The very special cell geometry present in this type of decomposition allows one to design very efficient algorithms to deal with it. On the other hand, due to its restrictive geometric nature, exact affine cell decompositions are not generally available for a given set. However, important families of subsets of the euclidean space can at least be approximated by an affine cell complex, which is sufficient for a huge number of applications.

Although the conditions imposed on the definition of affine cell decompositions concern mainly the geometry of the cells, they also imply additional combinatorial structure. In particular, the boundary of a d-dimensional cell has dimension $d-1$. As a consequence, affine cell decompositions have cells of all dimensions from 0 to the maximum dimension d. This is not true for arbitrary cell complexes. For instance, if p is a point on a 2-sphere S, then $\{p\}$ and $S-\{p\}$ determine a cell decomposition of S in which there are only cells of dimension 0 and 2.

D.1.4 Simplicial Decompositions

Simplicial decompositions are special cases of affine cell decompositions in which cell geometry is as simple as possible: each cell is a (relatively) open *simplex*. Given $d+1$ points v_0, v_1, ..., v_d not belonging to the same d-dimensional hyperplane, the set $\{\sum_i x_i v_i; 0 < x_i < 1, i = 0, \ldots, d\}$ is an open d-dimensional simplex. Furthermore, single points are considered to be 0-dimensional open simplices.

Every affine cell decomposition can be turned into a simplicial decomposition by using a refinement operation that consists in triangulating in a standard way each of its convex affine cells (see (Munkres, 1966)). Thus, the sets which admit an affine cell decomposition are exactly those who have a simplicial decomposition. There is a natural trade-off between the two types of decomposition. Simplicial decompositions are particularly easy to represent due to the fact that each cell of a given dimension has a fixed number of lower-dimensional bounding cells. On the other hand, general affine decompositions are more concise.

A *simplicial complex* $\mathcal{T}$ over a domain $D \in \mathbb{R}^n$ is a family of simplices with the following properties:

1. $D = \cup_{\sigma \in \mathcal{T}}$
2. $\sigma_1 \cap \sigma_2$ is either empty or a common face (lower dimensional simplex) of both simplices.
3. If D is a compact subset of $\mathbb{R}^n$ then it intersects only finitely many simplices.

Triangulations

A triangulation of a subset $U \in \mathbb{R}^n$ is a homeomorphism $h : \mathcal{T} \to U$ from some simplicial complex C to U. The complex C induces a cell decomposition on U, which

is called a *triangulation* of U. A subset U of the Euclidean space is *triangulable* if it admits this homeomorphism. Triangulable sets are also called *topological polyhedra.*

There are many possible triangulations of the space $\mathbb{R}^n$. Some important types of triangulations are the *Coxeter-Freudenthal* triangulation and the J_1 triangulation of Todd (Allgower and Georg, 1990). Figure D.1 shows these triangulations in two dimensions.

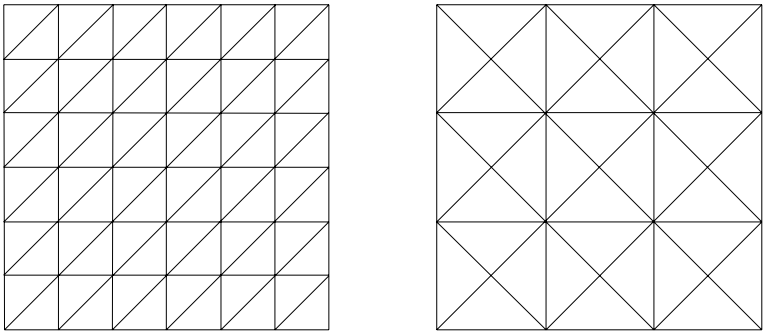


Figure D.1: Freudenthal and J_1 triangulations

It is desirable to have a small number of cell types, e.g., *congruent cells* – that differ only by orientation or reflection. If all the cells are identical, computations are very simple. It can be shown, (Coxeter, 1963), that in three dimensions the only type of cell that fills space is the *cube.*

A hypercube in $\mathbb{R}^n$ is the cartesian product of n non-degenerate intervals

$$\prod_{i=1}^{n}[a_i, b_i]$$

It is very easy to obtain a cell decomposition of $\mathbb{R}^n$, where each cell is a hypercube of appropriate dimension. Therefore, an easy way to obtain a triangulation of $\mathbb{R}^n$ is to use a triangulation of the hypercube. A classical triangulation obtained with this method is the Coxeter-Freudenthal triangulation.

This triangulation can be defined as follows: For the square (the hypercube in $\mathbb{R}^2$), we take its diagonal, and the triangulation obtained has two simplices of dimension 2. For the cube in $\mathbb{R}^3$ with vertices $p_0, \ldots, p_7$, we take the diagonal p_0p_7 and project it onto each face of the cube. We obtain in this way the Coxeter-Freudenthal triangulation of the faces. The simplices of dimension 3 of the triangulation of the cube are constructed by adding to each 2-simplex ρ of the cube's faces the vertex of the diagonal p_0p_7 that does not belong to ρ. (See Figure D.2).

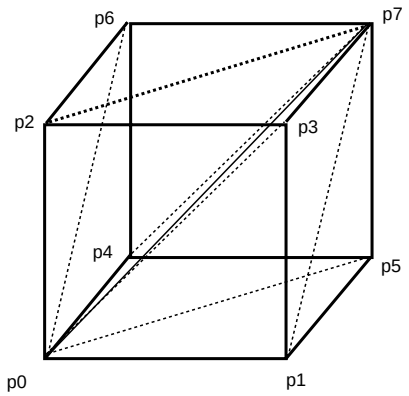


Figure D.2: Triangulation of the cube in $\mathbb{R}^3$

We obtain in this way 6 simplices of dimension 3.

$$\begin{aligned}
\rho_0 &= (p_0, p_1, p_3, p_7) \\
\rho_1 &= (p_0, p_1, p_5, p_7) \\
\rho_2 &= (p_0, p_2, p_3, p_7) \\
\rho_3 &= (p_0, p_2, p_6, p_7) \\
\rho_4 &= (p_0, p_4, p_5, p_7) \\
\rho_5 &= (p_0, p_4, p_6, p_7)
\end{aligned}$$

It is not too difficult to see that the Coxeter-Freudenthal triangulation of the hypercube in $\mathbb{R}^n$ has $n!$ simplices of dimension n. [1]

It is also important to observe that not all triangulations of $\mathbb{R}^3$ are obtained by replicating a triangulation of the cube. An example where this does not occur is in the already mentioned triangulation of Todd, shown in Figure $D.1$ for the two-dimensional case.

In three dimensions another interesting triangulation is based on the so called *cubic* tetrahedra, that are formed by slicing off the corners of a cube (Hall and Warren, 1990).

[1] This is not a problem if we are working in 3-space, but this combinatorial explosion can be a serious issue if we need to work in higher dimensional spaces.

D.2 Spatial Data Structures

The spatial decompositions defined on the previous section possesses natural graphs associated to them. They describe the adjacency relations between the several elements of the decomposition. Spatial data structures represent these graphs. They are used to structure interesting subsets of geometric objects and to provide a means for operating on them.

There are two ways to describe a region in a space decomposition: either by the point set that the region represents, or by the boundary that defines the region.

Some space decompositions are so simple and structured that it is not necessary to specify their regions explicitly. They are defined indirectly through a *canonical model*. This can be seen in some tessellations of space where the space subdivision is composed by the repetition of a single type of cell.

Spatial data structures can be divided in two main classes: flat and hierarchical. The former consists of an enumeration of cells, while the latter is defined by a recursive decomposition of cells.

Spatial data structures are designed to encode both the geometry and the topology of space decompositions. Below we analyze these data structures from the most general to the most specific type.

The most general data structure that can be used to represent space partitions is a *list* of cells. This cannot be really considered a spatial data structure because it does not encode any topological or hierarchical information. It is just a way to enumerate a set of cells. Note that this is the only type of data structure that can be used to describe arbitrary partitions of space.

D.2.1 Topological Graphs

The basic data structure that represent space decompositions is a topological graph in which the nodes of the graph contain information about the geometry of each decomposing region and the links of the graph give topological information reflecting adjacency relationships between regions. This type of spatial data structure is suitable to describe a large number of space decomposition schemes.

The fundamental topological information in a space decomposition is the association between a cell and its boundary elements. Note that this is a binary relation which is sufficient to completely specify the topology of a space decomposition. Since the boundary of a decomposing region is composed of geometrical elements of dimension lower than the dimension of the region, the graph in our data structure can be layered in a hierarchical manner such that each layer contains only regions of the same dimension. The links of the graph representing incidence relations occur only between layers. Figure D.3 shows a diagram of a topological graph.

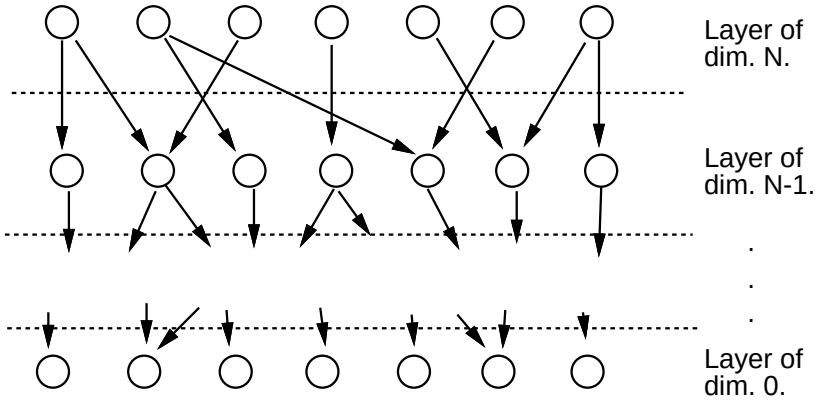


Figure D.3: Topological graph

D.2.2 Trees

Trees encode the hierarchical geometry of nested subdivisions of space. They consist of a set of nodes linked recursively starting from the top (or root) node. In this scheme we say that parent nodes are linked to children nodes. Terminal nodes are the leaves of the tree and do not have links. Each parent node corresponds to a cell which is subdivided into siblings according to some prescribed rule. For example, a n-dimensional cube can be subdivided into 2^n cubes.

The tree structures are classified according to the type of cell subdivision performed at the nodes:

N-trees (Quadtrees, Octrees) – Cells are subdivided n subcells by hyperplanes at regular intervals that are aligned with the axis of the reference frame.

K-d trees (K-dimensional trees) – Cells are subdivided in two subcells by a hyperplane arbitrarily positioned and aligned with each axis of the reference frame cyclically in succession (i.e. $x_1, x_2, \ldots, x_k$).

BSP-trees (Binary Space Partition trees) – Cells are subdivided in two subcells by a hyperplane arbitrarily positioned and oriented.

Restricted trees – are trees that correspond to a balanced partition of space. In this structure adjacent cells differ at most by a factor of 2 in terms of the level of refinement. This type of tree is important in adaptive subdivision schemes (von Herzen and Barr, 1987).

All trees can be reduced to binary trees that are constructed by recursively subdividing n-dimensional space into two regions by a $n-1$ dimensional hyperplane.

D.2.3 N-dimensional Arrays

N-dimensional arrays reflect the topology of regular packings of n-dimensional cells. They consist of a set of records specifying: the dimension of the array and the number of elements in each dimension, followed by the sequence of elements in a prescribed order. Arbitrary elements can be accessed directly through an n-tuple of indices corresponding respectively to each spatial dimension. Note that the spatial location of each cell relative to a frame of reference can be derived from the index of the element in the array and vice versa.

Arrays are flat data structures in which the cell geometry is given by a canonical model. This data structure tends to be quite large, because all the elements have to be stored.

It is interesting to note that n-dimensional arrays produce uniform subdivision that can be also represented by N-trees in which the leaves are all at the same depth.

References

Allgower, E. L. and Georg, K. (1990). *Introduction to Numerical Continuation Methods*. Springer-Verlag, Berlin, Heildelberg.

Allgower, E. L. and Gnutzmann, S. (1987). An algorithm for piecewise linear approximation of implicitly defined two-dimensional surfaces. *SIAM Journal of Numerical Analysis*, 24(2):2452–2469.

Allgower, E. L. and Schmidt, P. H. (1985). An algorithm for piecewise linear approximation of an implicitly defined manifold. *SIAM Journal of Numerical Analysis*, 22:322–346.

Amanatides, J. (1984). Ray tracing with cones. *Computer Graphics*, 18(3):129–136.

Basseville, M., Benveniste, A., Chou, K., Golden, S., Nikoukhah, R., and Willsky, A. (1992). Modeling and estimation of multiresolution stochastic processes. *IEEE Transactions on Information Theory*, 38(2):766–784.

Beier, T. (1990). Practical uses for implicit surfaces in animation. ACM Siggraph Course Notes. Modeling and Animating with Implicit Surfaces.

Blinn, J. F. (1982). A generalization of algebraic surface drawing. *ACM Transactions on Graphics*, 1(3):235–256.

Bloomenthal, J. (1988). Polygonization of implicit surfaces. *Comp. Aid. Geom. Des.*, 5(4):341–355.

Bloomenthal, J. and Shoemake, K. (1991). Convolution surfaces. *Computer Graphics*, 25(4):251–255.

Blum, H. (1967). A transformation for extracting new descriptors of shape. In Whaten-Dunn, W., editor, *Models for the Perception of Speech and Visual Form*, pages 362–380. MIT Press, Cambridge, MA.

Boyse, J. and Gilchrist, W. (1982). Gm solid: Interactive modeling for desing and analysis of solids. *IEEE Computer Graphics and Applications*, 2:209–216.

Brown, C. M. (1982). Padl-2 a technical summary. *IEEE Computer Graphics and Applications*, 2:69–84.

Burt, P. J. (1983). The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31:532–540.

Canny, J. (1986). A computational approach to edge detection. *IEEE Trans. on Patt. Anal. and Mach. Intell.*, 8:679–698.

Carvalho, P. C., Gomes, J. M., and Velho, L. (1992). Space decompositions: Theory and practice. *IMPA (preprint).*

Chui, C. K. (1992). *An Introduction to Wavelets.* Academic Press.

Coxeter, H. (1963). *Regular Polytopes.* Macmillan, New York.

Daubechies, I. (1992). *Ten Lectures on Wavelets.* Number 61 in CBMS-NSF Series in Applied Mathematics. SIAM Publications, Philadelphia.

de Figueiredo, L. H., de M. Gomes, J., Terzopoulos, D., and Velho, L. (1992). Physically-based methods for polygonization of implicit surfaces. In *Proceedings of Graphics Interface 92*, pages 250–257.

DeVore, R. A., Jawerth, B., and Lucier, B. J. (1992). Image compression through wavelet transform coding. *IEEE Trans. on Inf. Theory*, 38(2):719–746.

Gabor, D. (1946). Theory of communication. *J. Inst. Elec. Eng.*, 93(part III):429–457.

Goldstein, R. and Malin, L. (1979). 3-d modeling with the synthavision system. In *Proc. First Annl. Conf. Comp. Graphics in CAD/CAM Systems*, pages 244–247.

Gomes, J. and Velho, L. (1993). *Implicit Objects in Computer Graphics.* IMPA.

Greenspan, D. (1973). *Discrete Models.* Addison-Wesley, Reading, MA.

Hall, M. and Warren, J. (1990). Adaptive polygonization of implicitly defined surfaces. *IEEE Computer Graphics and Applications*, 10(6):33–43.

Haumann, D. (1987). Modeling the physical behavior of flexible objects. *Siggraph Tutorial Notes in Animation*, pages 1–10.

Hoffmann, C. M. (1991). Skeletons, cyclographic maps, and shock waves. manuscript.

Hoffmann, C. M. (1992). Computer vision, descriptive geometry, and classical mechanics. In Falcidieno, B. and Herman, I., editors, *Proc. Eurographics Workshop on Computer Graphics and Mathematics*, Eurographics Series, pages 229–244. Springer Verlag.

Kalkbrenner, M. (1990). Implicitization of rational parametric curves and surfaces. Technical Report RISC Linz, Johannes Kepler Univ., Linz, Austria.

Kaufman, A. (1987). Efficient algorithms for 3d scan-conversion of parametric curves, surfaces, and volumes. *Computer Graphics (SIGGRAPH '87 Proceedings)*, 21(4):171–179.

Kleck, J. (1989). Modeling using implicit surfaces. Master's thesis, University of California at Santa Cruz, Computer and Information Sciences.

Lorensen, W. E. and Cline, H. E. (1987). Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics*, 21(4):163–169.

Mallat, S. and Hwang, W. L. (1991). Singularity detection and processing with wavelets. Preprint Courant Institute of Mathematical Sciences, New York University.

Mallat, S. and Zhang, Z. (1993). Matching pursuits with time-frequency dictionaries. Technical report, Courant Institute, New York University.

Mallat, S. and Zhong, S. (1992a). Characterization of signals from multiscale edges. *IEEE Trans. on Patt. Anal. and Mach. Intell.*, 14:710–732.

Mallat, S. and Zhong, S. (1992b). Wavelet transform maxima and multiscale edges. In et. al., M. B. R., editor, *Wavelets and Their Applications*, pages 67–104. Jones and Bartlett Publishers.

Mallat, S. G. (1989). Multifrequency channel decompositions of images and wavelet models. *IEEE Trans. on Acoust. Signal Speech Process.*, 37(12):2091–2110.

Manocha, D. and Canny, J. (1990). Implicitization of rational parametric surfaces. In *The Mathematics of Surfaces*. Oxford University Press, England.

Mantyla, M. (1982). Boolean operations of 2-manifolds through vertex neighborhood classification. *ACM Transactions on Graphics*, pages 1–29.

Marr, D. (1982). *Vision*. Freemann.

Marr, D. and Hildreth, E. (1980). Theory of edge detection. *Proc. Roy. Soc. Lon*, 207:187–217.

McCormick, B. H. e. a. (1987). Visualization in scientific computing. *Computer Graphics*, 21(3).

Meyer, Y. (1993). *Wavelets, Algorithms and Applications*. SIAM.

Milnor, J. (1965). *Topology from the Differentiable Viewpoint*. The University Press of Virginia.

Munkres, J. (1966). *Elementary Differential Topology*. Princeton University Press.

Muraki, S. (1993). Volume data and wavelet transform. *IEEE Computer Graphics and Applications*, 13(4):50–56.

Nackman, L. R. and Pizer, S. M. (1985). Three-dimensional shape description using the symmetric axis transform I. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI 7:187–205.

Nishimura, H., Hirai, M., Kawai, T., Kawata, T., Shirakawa, I., and Omura, K. (1985). Object modeling by distribution function and a method of image generation. *Japan Electronics Communication Conference 85*, J68-D(4):718–725.

Pasko, A. and Pilyugin, V. (1988). Geometric modeling in the analysis of trivariate functions. *Computer and Graphics*, 12(3-4):457–465.

Pavlidis, T. (1978). Filling algorithms for raster graphics. *Computer Graphics (SIGGRAPH '78 Proceedings)*, 12(3):161–166.

Perlin, K. and Hoffert, E. (1989). Hypertexture. *Computer Graphics*, 23(3):253–262.

Perlin, K. and Zhu, B. (1990). Surflets. ACM Siggraph Course Notes. Photorealistic Volume Modeling and Rendering Techniques.

Requicha, A. (1980). Representation for rigid solids: Theory, methods, and systems. *ACM Computing Surveys*, 12(4):437–464.

Ricci, A. (1973). A constructive solid geometry for computer graphics. *The Computer Journal*, 16(2):157–160.

Rockwood, A. P. and Owen, J. (1987). Using implicit surfaces to blend arbitrary solid models. In Farin, G., editor, *Geometric Modeling: Algorithms and Trends*. SIAM.

Rosenfeld, A. (1984). *Multiresolution Image Processing and Analysis*. Springer–Verlag, New York.

Szeliski, R. (1989). Fast surface interpolation using hierarchical basis functions. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'89)*, pages 222–228, San Diego, California. IEEE Computer Society Press.

Szeliski, R. and Terzopoulos, D. (1989). From splines to fractals. In Lane, J., editor, *Computer Graphics (SIGGRAPH '89 Proceedings)*, volume 23, pages 51–60.

Tavares, G. and de M. Gomes, J. (1989). Concordance operations for implicitly-defined manifolds. In *First SIAM Conference on Geometric Design*, page 10.

Terzopoulos, D. (1984). *Multiresolution Computation of Visible-Surface Representations*. PhD thesis, Massachusetts Institute of Technology.

Terzopoulos, D. (1986). Image analysis using multigrid relaxation methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(2):129–139.

Terzopoulos, D. and Fleischer, K. (1988). Modeling inelastic deformation: viscoelasticity, plasticity and fracture. *Computer Graphics*, 22(4):269–278.

Terzopoulos, D., Platt, J., Barr, A., and Fleischer, K. (1987). Elastically deformable models. *Computer Graphics*, 21(4):205–214.

Terzopoulos, D., Platt, J., and Fleischer, K. (1989). Heating and melting deformable objects. In *Proceedings of Graphics Interface 89*, pages 219–226.

Terzopoulos, D. and Vasilescu, M. (1991). Sampling and reconstruction with adaptive meshes. In *Computer Vision & Pattern Recognition Conference (CVPR-91)*, pages 70–75, Lahaina, HI.

Terzopoulos, D., Witkin, A., and Kass, M. (1987). Symmetry-seeking models and 3D object reconstruction. *International Journal of Computer Vision*, 1(3):211–221.

Tonnesen, D. (1989). Ray tracing implicit surfaces resulting from the summation of bounded polynomial functions. Technical report, Rensselaer Polytechnic Institute.

Totsuka, T. and Levoy, M. (1993). Frequency domain volume rendering. In Kajiya, J. T., editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 271–278.

Velho, L. (1990). Adaptive polygonization of implicit surfaces using simplicial decomposition and boundary constraints. In *Proceedings of Eurographics 90*, pages 125–136. Elsevier Science Publisher.

Velho, L. and de M. Gomes, J. (1991a). A dynamical simulation environment for implicit objects using discrete models. In *Proceedings of 2nd Eurographics Workshop on Animation and Simulation*, pages 183–190.

Velho, L. and de M. Gomes, J. (1991b). Regular triangulations of implicit manifolds using dynamics. In *Proceedings of Compugraphics 91*, pages 57–71.

Velho, L. and de M. Gomes, J. (1993). A multiscale piecewise representation for implict objects. In *Third SIAM Conference on Geometric Design*, page 14.

von Herzen, B. and Barr, A. (1987). Accurate triangulations of deformed intersecting surfaces. *Computer Graphics*, 21(4):103–110.

Williams, L. (1983). Pyramidal parametrics. In *Computer Graphics (SIGGRAPH '83 Proceedings)*, volume 17, pages 1–11.

Wyvill, B., McPheeters, C., and Wyvill, G. (1986a). Animating soft objects. *The Visual Computer*, 2(4):235–242.

Wyvill, G., McPheeters, C., and Wyvill, B. (1986b). Data structure for soft objects. *The Visual Computer*, 2(4):227–234.

Wyvill, G. and Trotman, A. (1990). Ray tracing soft objects. ACM Siggraph Course Notes. Modeling and Animating with Implicit Surfaces.