EUROGRAPHICS '0x / N.N. and N.N.
(Guest Editors)

Volume xxx, (200x), Number yyy

Multiresolution for Algebraic Curves and Surfaces using Wavelets

Jordi Esteve Pere Brunet Alvar Vinacua

Universitat Politècnica de Catalunya, Barcelona, Spain
e-mail: [jesteve, brunet, alvar]@lsi.upc.es

Abstract

This paper describes a multiresolution method for implicit curves and surfaces. The method is based on wavelets, and is able to simplify the topology. The implicit curves and surfaces are defined as the zero-valued piece-wise algebraic isosurface of a tensor-product uniform cubic B-spline. A wavelet multiresolution method that deals with uniform cubic B-splines on bounded domains is proposed. In order to handle arbitrary domains the proposed algorithm dynamically adds appropriate control points and deletes them in the synthesis phase.

Keywords: *geometric modeling, algebraic surfaces, wavelets, multiresolution, topological simplification, conversion algorithms.*

1. Introduction

In this paper a planar curve and surface multiresolution method that simplifies the topology is presented. We will work with algebraic curves and surfaces defined as the zero-valued algebraic isosurface of a tensor-product uniform cubic B-spline. Thus, the cubic B-spline has one dimension more than the dimension of objects to be represented (bivariate B-splines for curves and trivariate B-splines for surfaces). The B-spline approximation will produce the simplification of the objects and the change of their topology in a simple way.

A preprocess is required to convert the initial object model in a way that B-spline algebraic curve/surfaces can be used. After a voxelization of the initial object (a polyhedral BRep, a surface model, etc.), the conversion method from voxel and octree representations to B-spline algebraic surfaces described in Vinacua et al.[1] can be used to obtain the initial data for the wavelet decomposition.

There are a lot of efforts around the solid multiresolution or simplification, due to the interest of interacting with simplified versions for complex models. However, because of the complexity of the topological relations inside the classical models used in CAD (triangular meshes, BReps, CSGs) it is actually difficult to make changes directly in their topology.

Many algorithms simplify triangular meshes doing edge collapses, for example Hoppe et al.[2] and Hoppe[3]. Some also make topological changes joining the nearest vertices of different triangles: Rossignac et al.[4], Ronfard et al.[5], Popovic et al.[6], Garland et al.[7]. Although they do topological simplification, they produce non-regular surfaces (isolate edges and vertices appear) or non-manifold objects (not every surface point has a neighbourhood topologically equivalent to a disk).

The algorithms that get topological changes keeping the solid correctness are based on space decomposition models, usually voxels and octrees. It is the case of He et al.[8] with voxels and Andújar et al.[9] with octrees. Nevertheless, they have several drawbacks: they must do two conversion steps to and from the space decomposition model, the second one is very hard if we want to compress the large number of generated faces and it is complicated to save information needed for the reconstruction of the initial object.

In our multiresolution algorithm we use a very powerful tool: the wavelets. The wavelet methods allow doing a fast decomposition that minimizes the error data (the detail lost in the decomposition) and, later, we can reconstruct the original solid. Furthermore, the decomposed solid plus the error data use the same storage space as the original solid.

Other curve and surface multiresolution methods based on wavelets exist, but they do not use algebraic curves/surfaces and they cannot simplify the topology either. Reissell[10] implements a parametric curves/surfaces multiresolution with a type of wavelets called coiflets. The method works over the co-ordinates of the surface points and the approximations are only linear interpolations of the scaling coefficients. Frequently, the wavelets multiresolution methods have been applied over B-spline surfaces (see Section 3). Using the ideas of Lyche et al.[11], Kazinnik et al.[12] presents a multiresolution over a parametric surface defined as a tensor-product of non-uniform B-splines. Finkelstein et al.[13] does the same over tensor-products of end-interpolating uniform cubic B-splines. In Lounsbery et al.[14] the wavelets are applied over recursive subdivision surfaces with any topology, although they cannot simplify this topology.

In Vinacua et al.[1], a conversion method from a voxel or octree representation to an algebraic surface defined as the zero-valued algebraic isosurface of a tensor-product uniform cubic B-spline is described. The multiresolution is based on an octree level selection and a subsequent conversion to the algebraic surface. However, this conversion has an important cost. The goal of our implementation is a quick multiresolution thanks to directly handling the B-spline control points that define the algebraic surface. Furthermore, our algorithm produces better approximations due to the use of wavelet methods.

In the next section we present the formal definition of the algebraic curves and surfaces used in the application. In Section 3 we discuss how to fit the wavelet multiresolution to decompose/reconstruct models of arbitrary size voxel spaces (i.e. not necessary powers of 2). Section 4 exposes the particularities of our algorithm: how to set the unknown control points in the decomposition and how we can do the reconstruction process. Also, it describes a convenient data structure for the model. Section 5 includes several examples in 2D and 3D domains. Finally, we point out some conclusions and possible improvements of our work.

2. B-Spline-Based Algebraic Surface Models

Let be the voxel model of an object surface. We do not need all the data contained in the voxel model, it is enough to store the set of voxels (nodes) stabbed by the object surface. This set of nodes is called Node-Collection.

The Node-Collection can be obtained easily from voxel and octree representations. The Node-Collection can also be calculated from polyhedral BReps and surface models through a voxelization or 3D scan conversion algorithm[15].

Using the Node-Collection to represent the surface offers us flexibility. Any surface stabbing all nodes in the Node-Collection is an acceptable representation of the initial object. This allow us to get smoother surfaces. As always, this discretization comes at the price of ignoring detail below its node's resolution. Therefore, the resolution of the Node-Collection must be chosen according to the size of the features that we are interested in.

In our case, a functional B-spline has been chosen to define the surface stabbing the Node-Collection. The surface is the zero-valued algebraic isosurface of the functional B-spline.

To model a planar curve, for example, we consider a 3D B-spline function defined on the rectangular grid (now the nodes are 2D). The intersection of its graph with the zero-valued plane defines the curve (see Figure 1). The Node-Collection is the set of the 2D nodes on the rectangular grid stabbed by the curve (see Figure 2). To model a surface, we define a B-spline function on the 3D regular grid such that the intersection of its graph with the zero-valued hyperplane ($w = 0$) produces the surface.

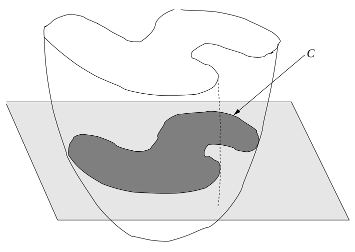

Figure 1: *The intersection of the 3D B-spline function with the zero-valued plane defines the object boundary curve*

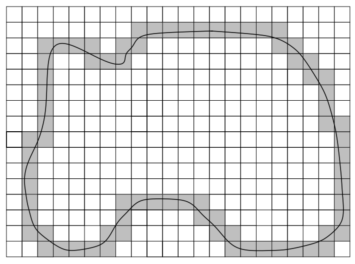

Figure 2: *The Node-Collection is the set of shaded nodes*

Let us now define G as the set of spatial indices of the nodes of the Node-Collection

$$G = \{(i, j),\ N_{ij} \in \{Node{-}Collection\}\} \qquad (1)$$

where N_{ij} represents a unit cube with the vertices $(i+\delta_1, j+\delta_2)$, with δ_1, δ_2 being either 0 or 1. Observe that the cardinality of G grows in the same manner as the size of the object does (quadratically if we are modeling a surface).

On the other hand, let us define I as the set of spatial indices of the nodes in the immediate vicinity of the Node-Collection:

$$I = \{(i,j)\ with\ (i+\delta_1, j+\delta_2) \in G,\ \delta_1, \delta_2 \in \{-1, 0, +1, +2\}\} \tag{2}$$

The cardinality of I is of the order of 4 times the cardinality of G. A similar expression is easily obtained in a 3D case. Let us now consider the uniform, tensor-product B-spline function F defined on the regular grid. F is an integral, functional spline and we are only interested in evaluating it on the Node-Collection, to find its zero-isosurface.

$$F(x,y) = \sum_{(i,j)\in I} w_{ij}\ N_i(x)\ N_j(y) = 0 \tag{3}$$

F is a valid representation if its zero-isosurface stabs all nodes in the Node-Collection. Let us consider that $F < 0$ inside the object and $F > 0$ outside the object (only in nodes belonging to the Node-Collection). Since F is defined on a regular grid (induced by the voxels), we choose to represent it as a tensor-product uniform cubic B-spline.

Our multiresolution algorithm consists of two main steps:

1. The B-spline algebraic surface is obtained. The B-spline functional F must be calculated from an initial Node-Collection. This is an iterative algorithm that tries to minimize the curvature of the final B-spline algebraic surface. Our implementation is based on Vinacua et al.[1]. We initially assign weights of $+1$ to the vertices in the set I (Equation 2) that lie outside the object and -1 to the interior vertices in that set. We then filter these weights using a discrete Laplacian instead of the more elaborated method described in[1].
2. A surface multiresolution is calculated applying a wavelet multiresolution method to the B-spline functional F. The object surface approximations are the zero-isosurfaces of the wavelet approximations of F.

This paper focuses on this last point. Before presenting a detailed discussion in Section 4 we include for completeness and notational convenience an introduction to wavelets-based B-spline multiresolution in the next section.

3. Wavelet Multiresolution

Wavelets are a mathematical tool with a wide variety of applications in several fields. A lot of B-spline wavelet methods exist. In Finkelstein et al.[13] a method for end-interpolating uniform cubic B-splines has been presented. Chui[16] contains a study of the uniform B-spline wavelets on unbounded domains. In Lyche et al.[11] and, later, in Kazinnik et al.[12] the use of wavelets with non-uniform B-splines is described.

In our case, due to the kind of algebraic surface to decompose, we will use uniform cubic B-splines on bounded domains (their knot sequence is uniformly spaced, with all knots of multiplicity 1 and the domain is finite). We develop a wavelet multiresolution method, similar to the end-interpolating B-splines in Finkelstein et al.[13], implemented with pre-calculated band matrices. This will allow us to perform the analysis and synthesis processes of a row of coefficients with linear cost. Our method produces similar results as Chui[16] does, but using wavelets defined on bounded domains. In Chui[16] the same filter is applied to all coefficients in the analysis/synthesis process (a moving average algorithm). The disadvantage of the wavelets in Chui[16] is that the analysis filters are infinite and must be truncated (causing precision errors).

Before describing our algorithm, we briefly review the main ideas behind wavelet multiresolution and define the notation used. For a more complete description and implementation details see, for example, Finkelstein et al.[13] and Esteve et al.[17]. It is important to remark that, in this paper, the vector and matrix indices represent the number of intervals where the B-spline function is defined instead of a level counter in the multiresolution process.

A B-spline on a bounded domain with 2^{n+1} intervals can be represented as a column vector of control points $C^{2^{n+1}}$ (made up of $2^{n+1} + degree$ control points). The goal is to approximate the B-spline with another B-spline function of half resolution (its domain will have half number of intervals). This function can be represented as a column vector of fewer control points C^{2^n}. Classical wavelets use a matrix A^{2^n} to compute

$$C^{2^n} = A^{2^n} C^{2^{n+1}} \tag{4}$$

and an accompanying matrix B^{2^n} to capture detail

$$D^{2^n} = B^{2^n} C^{2^{n+1}} \tag{5}$$

These two matrices (called analysis filters) have independent columns, and the process can be inverted using the synthesis filters P^{2^n} and Q^{2^n}:

$$C^{2^{n+1}} = P^{2^n} C^{2^n} + Q^{2^n} D^{2^n} \tag{6}$$

without loss of information. Furthermore, the space required to store C^{2^n} and D^{2^n} is the same as that required for $C^{2^{n+1}}$.

This process can be iterated yielding a multiresolution scheme (with each level using half as much detail), which can be stored using the same space as one approximation C^1 and a sequence of detail vectors $D^1, D^2, D^4, ..., D^{2^n}$.

We shall need to apply this level of analysis in contexts where the number of intervals of the B-splines involved is arbitrary, not a power of two, and we will thus introduce modifications to this scheme in the following section.

Before dealing with that aspect of our problem, we need to borrow one more ingredient of classical wavelets. The synthesis matrices P^{2^n} and Q^{2^n} for our bounded-domain uniform B-splines are banded and, therefore, yield to economic numerical handling. However, their "inverses" A^{2^n} and B^{2^n} are full, which means that the analysis phase using (4) and (5) has a quadratic cost (in the number of control points). Quak et al.[18] shows how to improve on this. Noting by Φ^{2^n} the vector of B-spline basis functions (the scaling functions) and by Ψ^{2^n} the vector of basis functions of the orthogonal of the resolution 2^n B-splines in the space of resolution 2^{n+1} B-splines (the wavelet functions) one can form the matrices I^{2^n} and J^{2^n} of inner products of these, which satisfy[18, 17]

$$I^{2^n} C^{2^n} = [P^{2^n}]^T I^{2^{n+1}} C^{2^{n+1}} \tag{7}$$

$$J^{2^n} D^{2^n} = [Q^{2^n}]^T I^{2^{n+1}} C^{2^{n+1}} \tag{8}$$

and are banded. Thus, the analysis amounts to computing the right-hand sides of these and solving two banded linear systems, which gives a linear cost[19]. Our technical report[17] discusses this in more detail and also include the computation of the synthesis matrices P^{2^n} and Q^{2^n} and the inner product matrices I^{2^n} and J^{2^n} for the uniform cubic B-splines defined on a bounded domain.

Next, we move on to discuss our adaptation of these algorithms to our problem, extending in Section 4.1 their applicability to an arbitrary even number of intervals.

4. The Proposed Wavelet Algorithm

We have adapted the wavelet multiresolution to our application, since the B-spline function is not defined on the whole domain, but only on the nodes of the 3D grid that belong to the "Node-Collection". The main contributions of the proposed wavelet algorithm are next summarized:

1. The family of analysis and synthesis matrices has been extended. This is necessary, as the B-spline functions must be defined on a domain more general than domains having 2^n intervals. We restrict the B-splines to have an even number of intervals. This generalizes the usual analysis/synthesis process.
2. At each iteration, only B-spline control points in the vicinity of the zero-isosurface are stored.
3. We must set some unknown control points (control points whose indices do not belong to the set I) before decomposing the B-spline. The reasons are:
   - We need that the B-splines have an even number of intervals to decompose them, as indicated above.
   - Intervals which are close together can be joined in the decomposition. In this case, we must set the control points in between.
   - The decomposed B-spline plus the produced detail should use the least space possible (ideally the same as the original B-spline).
4. In the reconstruction we must recover the control points whose indices belonged to the set I using the information obtained in the analysis process and keeping a linear time cost. We will also reject the control points that we have set in the decomposition process.

Figure 3 shows the changes in the Node-Collection during the analysis process. The area that is occupied by the Node-Collection grows. We must take into account the unknown control points around these new areas of the Node-Collection.

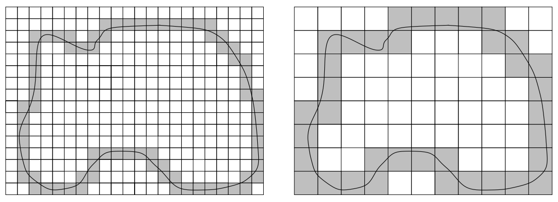
(a) Level 2n (b) Level n

Figure 3: *Changes in the Node-Collection*

4.1. Wavelet Multiresolution on Domains of Even Length

The single condition we will impose on the multiresolution is that the B-spline must have an even number of intervals where it is defined (the number of control points must be odd since we are using cubic B-splines). In the decomposition process, the number of intervals will be divided by 2.

Thus, the decomposition of a B-spline defined on $2n$ intervals C^{2n} in one defined on n intervals C^n and its detail D^n and the inverse reconstruction can be written (analogously to Equations (4), (5) and (6)) as

$$C^n = A^n C^{2n} \tag{9}$$

$$D^n = B^n C^{2n} \tag{10}$$

$$C^{2n} = P^n C^n + Q^n D^n \tag{11}$$

In the iterative application of the analysis process we can obtain a B-spline with an odd number of intervals. Then, we must set additional control points at each end of the B-spline to return to an even interval B-spline and, in this way, continue the decomposition. Subsection 4.2 discusses how to set these additional control points. Therefore, we need to add an assignment process between the decomposition steps. Also, during the reconstruction phase, we must reject the control points thus introduced at each level:

Analysis

$$\begin{tikzcd}
C^{2n} \arrow[r, "A^n"] \arrow[dr, "B^n"'] & C^n \arrow[r, "set"] & C^{2m} \arrow[r, "A^m"] \arrow[dr, "B^m"'] & C^m \dots \\
& D^n & & D^m \dots
\end{tikzcd}$$

Synthesis

$$\begin{tikzcd}
C^{2n} & C^n \arrow[l, "P^n"'] & C^{2m} \arrow[l, "reject"'] & C^m \dots \arrow[l, "P^m"'] \\
& D^n \arrow[ul, "Q^n"] & & D^m \dots \arrow[ul, "Q^m"]
\end{tikzcd}$$

The synthesis matrices P^n and Q^n and the inner product matrices I^n and J^n for the uniform cubic B-splines defined on a domain with an arbitrary even number of intervals are listed in Esteve et al.[17].

4.2. Analysis Process and the Assignment of Extra Control Points

As previously discussed, before applying the analysis process we must set the value of some unknown control points. Hence, we must decide which control points must be assigned and which will be their value. We have studied several alternatives about how to set these control points.

In the following examples we detail and explain the decomposition of the 10th row (we work on unidimensional examples that represent the decomposition/reconstruction in one specific direction of the tensor product).

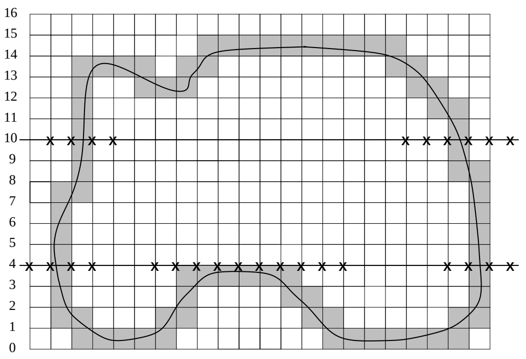

Figure 4: *Horizontal Decomposition*

First (see Figure 5), the B-spline functional $C^{initial}$ is defined on the intervals [2, 3] and [19, 22]. We have to set control points with odd indices (in positions 5 and 17) to get

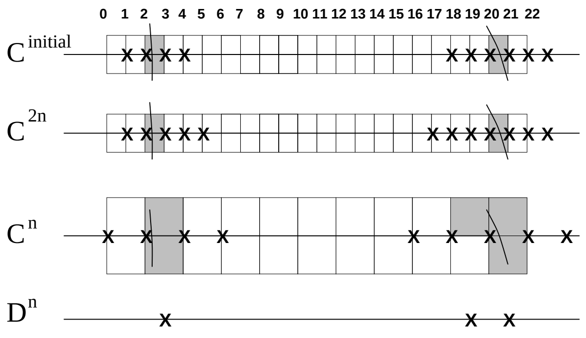

Figure 5: *10th row decomposition*

the B-spline C^{2n} defined on two domains with even indices ([2, 4] and [18, 22]). Then we can apply the decomposition process described in the previous section to obtain C^n and D^n.

However, there is a problem: when working with a tensor product of the basis functions (on a 2 or 3D grid), after several steps of decomposition artifacts do appear (empty space inside the solid and/or solid inside the empty space). The reason is that the basis functions of the uniform cubic B-splines on a bounded domain are similar except at the endpoints, which are truncated. To minimize the error produced, the wavelet decomposition plays with the endpoint basis functions, that have less energy than the others, giving them high weights and sign changed with respect to the neighbours basis functions. The control points at each end in one direction can become interior control points when we change the direction of the decomposition process. In short, the value of the control points are weights of basis functions with different energy depending on the direction we are decomposing.

To overcome this problem, the B-spline function has been extended at each end before applying the decomposition process. In this way, the truncated basis functions have no influence in the region of interest. Setting 3 additional control points at each side is enough because the uniform cubic B-splines on bounded domains have 3 truncated basis functions at each end. Our implementation, however, sets 4 additional control points at each side which yields simpler algorithms.

For example, Figure 6 shows a piece of the 10th row decomposition. First, as we described before, the control points with odd indices are assigned (the control point in position 5) to get C^{2n}. Then, 4 additional control points at both sides are assigned (C^{set}). We shall call these *accessory control points*. And finally $C^{2n} + C^{set}$ is decomposed in C^{n+4} and D^{n+4}.

Obviously, this proposal produces 8 additional control points in C^{n+4} plus D^{n+4} than those in C^{2n}. To overcome this drawback, first observe that only the control points of C^{n+4} and D^{n+4} needed to reconstruct the initial B-spline C^{2n} must be stored. Since P^n is a band matrix such that its middle rows have 2 or 3 non-zero coefficients, all the C^{2n} control

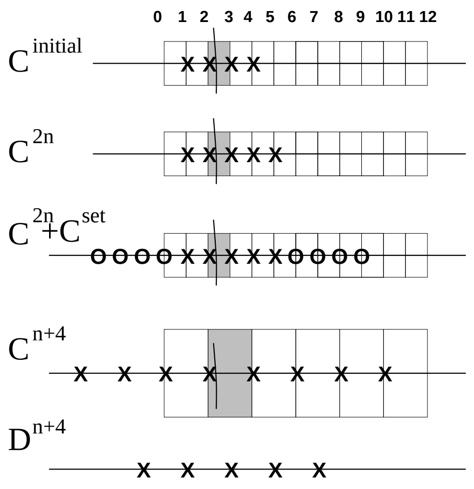

Figure 6: *Piece of 10th row decomposition. 4 additional control points at both sides are assigned to C^{2n}*

points (in positions 1, 2, 3, 4 and 5) are calculated from the C^n control points in positions 0, 2, 4 and 6 (for example — the position is indicated by subindices— C_1^{2n} is recovered form C_0^n and C_2^n, C_2^{2n} is recovered form C_0^n, C_2^n and C_4^n, etc). Therefore we can reject 2 control points at both sides of each interval of C^{n+4}. We cannot do the same with D^{n+4} because Q^n has the middle rows wider than P^n (Q^n has 5 or 6 non-zero coefficients in the middle rows).

In this way, each interval decomposition produces 4 additional control points in C^n plus D^{n+4} than those in C^{2n}. Next observe that if in the reconstruction process we know the values assigned to the accessory control points of C^{2n} during the decomposition process we may do without these extra 4 control points. In the present work we set the accessory control points of C^{2n} in the following way: they can be $+K$ or $-K$ (K is a model constant whose value is in the same order of magnitude as the control points) and, further, all correlative control points have the same sign and they coincide in sign with the functional B-spline at the endpoints of the domain where it is defined. This is consistent with the semantics of our model: negative values correspond to the interior whereas positive values correspond to the exterior. In this way, we don't need keep so many control points of D^{n+4} in the decomposition, because we take advantage of knowing several fragments of C^{2n}. The computational procedure to take advantage of this information is discussed in Section 4.3. As can be seen in Figure 7, we achieve that the number of control points stored in C^n plus D^n is the same as the initial number in C^{2n}.

We have tested other ways to set the accessory control points. For example, setting the accessory control points as the average value of the closest known control point at the left side and the closest known control point at the right side. Or setting the accessory control points as the average value

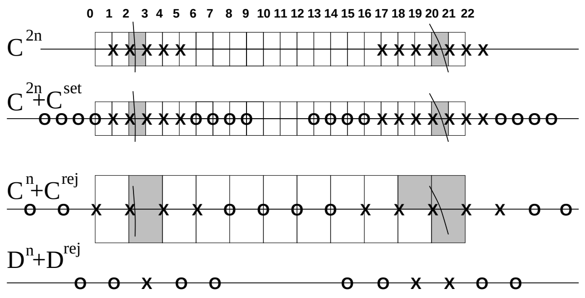

Figure 7: *10th row decomposition. 4 additional control points at both sides are assigned to C^{2n} and, later, 2 control points at both sides are rejected from C^n and D^n*

of the functional B-spline evaluated at the ends of the Node-Collection at the left side and right side. We have also tested the assignment of accessory control points that minimizes the error data in the wavelet decomposition. None of them can take advantage of knowing several fragments of C^{2n} to reduce the number of control points stored in the decomposition. And none of them has produced so good results as the assignment of $+K$ or $-K$ we have previously described. This assignment produces B-spline functions with strong slopes around their zeroes in the nodes of the Node-Collection, leading to better approximations.

Note that, in the analysis process, if two intervals of known control points of C^{2n} are very close, they can be converted into one interval in C^n. This happens when the distance of the nearest endpoint control points of two intervals (situated on odd positions) is lower or equal than 4 (see Figure 8). In these cases we cannot use the economies discussed above, and our scheme will need a few more control points to represent the multiresolution (see Section 5 for figures measuring the impact of this in several examples).

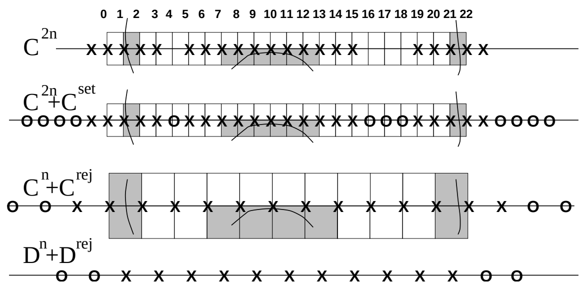

Figure 8: *4th row decomposition. Joining of 3 close intervals*

4.3. Synthesis Process and the Recovery of Control Points

Let us see how to do the reconstruction $C^{2n} = P^n\,C^n + Q^n\,D^n$ under these conditions. We start by knowing the previous accessory control points of C^{2n} (with value $+K$ or $-K$) and those stored in C^n and D^n. From the reconstruction point of view the known control points of C^{2n}, C^n and D^n of the example in Figure 7 are marked in Figure 9.

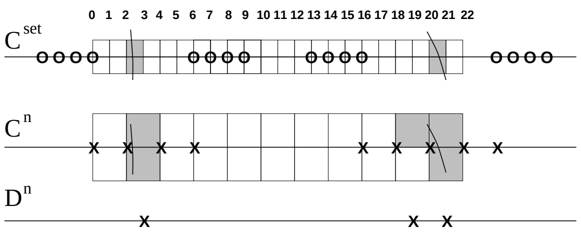

Figure 9: *10th row reconstruction. Known control points of C^{2n}, C^n and D^n*

Splitting each column vector C in two vectors with the known C_k and unknown C_u control points (C, C_k and C_u have the same dimension. C_k is filled with zeroes in the entries where there are unknown control points. C_u is filled with zeroes in the entries where there are known control points.) and applying the synthesis filter:

$$C^{2n} = C_k^{2n} + C_u^{2n} \tag{12}$$

$$C^n = C_k^n + C_u^n \tag{13}$$

$$D^n = D_k^n + D_u^n \tag{14}$$

$$I(C_k^{2n} + C_u^{2n}) = P^n(C_k^n + C_u^n) + Q^n(D_k^n + D_u^n) \tag{15}$$

$$IC_k^{2n} - P^nC_k^n - Q^nD_k^n = P^nC_u^n + Q^nD_u^n - IC_u^{2n} \tag{16}$$

The left-hand side in the last equation can be evaluated: it is the independent term V of the linear equation system. The right-hand side can be expressed with a single matrix and a single vector of unknown control points if we eliminate the matrix columns that are multiplied by zero (the vectors and matrices simplified are marked with ')

$$V = [P'^n|Q'^n| - I'][C_u'^n|D_u'^n|C_u'^{2n}]^T \tag{17}$$

Reordering the columns of matrix $[P'^n|Q'^n| - I']$, we transform it into a band matrix. So, we have a banded linear system that can be solved in linear time doing an LU decomposition[19].

In the reconstruction process we must reject the accessory control points of C^{2n} that have been assigned. So, we must store additional information about which control points are assigned during the analysis process (in the present implementation we store the intervals where there are defined control points, see next section).

4.4. Data Structure

Because our model only works with the control points with spatial indices contained in I, a set usually much smaller than the possible control points that can be defined inside the domain's grid, we think it is suitable to store the control points in a hash table. We also keep some additional data about the intervals where the control points are defined in one direction (a vector of interval lists in 2D and a matrix of interval lists in 3D). The hashing table enables the storage of the known control points in a compact way (the storage space is proportional to the known control points, not to the domains's grid) and to consult them in an almost constant time. The interval information about where the defined control points are, avoids searching for unknown control points in the hashing table. Thus, it will be feasible to do fast computations in the direction of the interval data in the analysis and synthesis process. Furthermore, this information will be very useful in the reconstruction process since it will allow us, in one multiresolution level, to distinguish between the known and the accessory control points.

Due to the alternation of the direction of the evaluations in both the analysis and the synthesis processes we must calculate the interval information in one direction from the same information in other direction. This costs $O(n)$ in 2D and $O(n^2)$ in 3D (it needs to visit only once each element in the interval vector/matrix: we assume the same n dimension in all directions of the data domain). This cost has the same order of magnitude as each decomposition and reconstruction step if we consider that the control points in the Node-Collection are sparse in relation to the domain's grid and, therefore, the analysis and synthesis process of each row of few control points has a constant cost. This is only true at high resolution. It is there, however, that the question of cost if meaningful.

5. Examples and Discussion

We have tested the proposed algorithm over several curves and surfaces. Figures 10 through 13 show examples of curves, whereas Figures 14 and 15 show two 3D solids. Figure 10 shows a multiresolution of the African continent, Figure 11 of a camera, Figure 12 of a skeleton and Figure 13 of a tiger. The images correspond to the resolution levels after doing the decomposition process in the two directions. Therefore, there is an intermediate resolution between two consecutive images. The initial curve is obtained from a bitmap image. We have developed a simple algorithm for that conversion. Table 1 lists the dimensions of the domain, the number of control points stored (spatial indices of I) and the number of defined nodes (same order of magnitude as the Node-Collection G) for each resolution level. The number of coefficients in the wavelet transform of Africa (Africa in the lowest resolution level plus all the detail data produced in the decomposition process) is 27116, 22.4% more than the original curve. The wavelet transform of tiger has 152062 coefficients, 8.3% more than the original curve (140397 coefficients). This increase is due to set the accessory control points in the decomposition process.

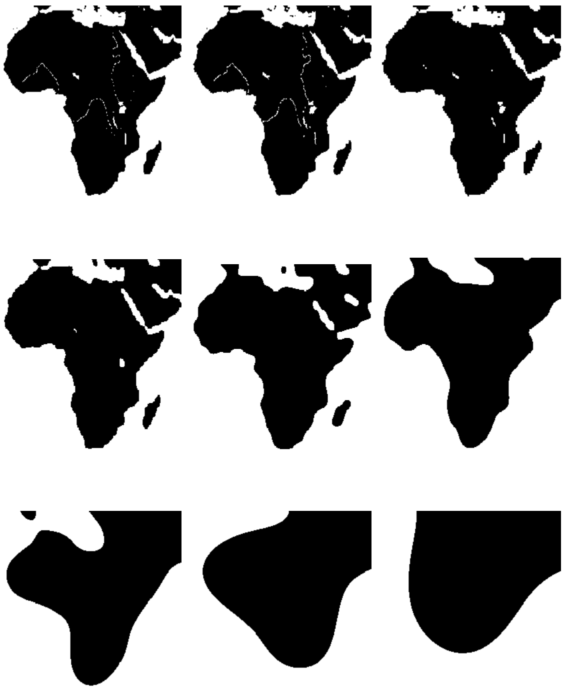
Figure 10: *Africa multiresolution*

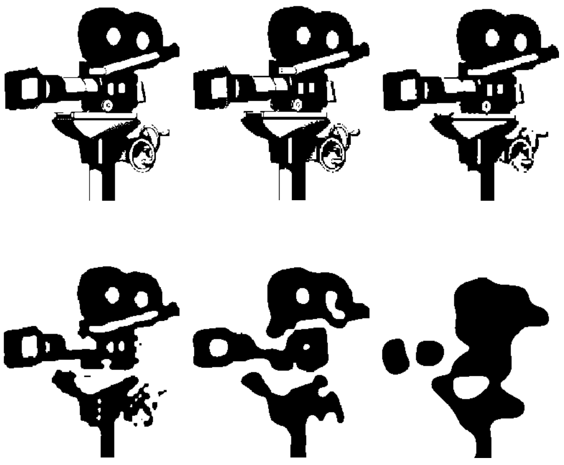
Figure 11: *Camera multiresolution*

Figure 14 shows a multiresolution of a medical model

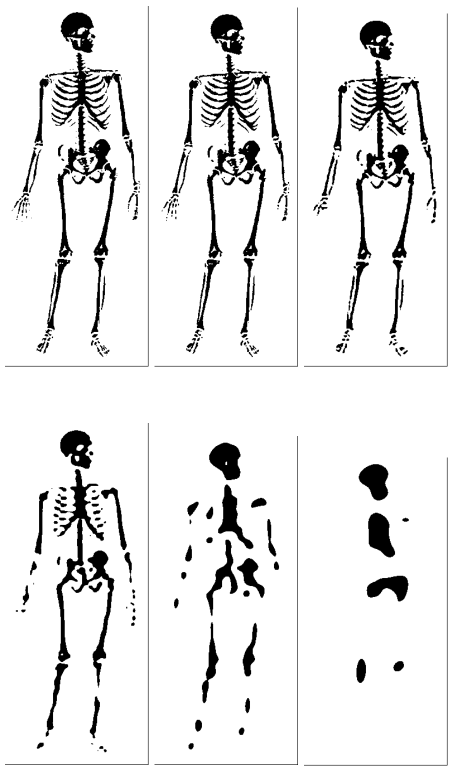
Figure 12: *Skeleton multiresolution*

(an image of the voxelization of a skull from a computerized tomography). Finally, Figure 15 shows a multiresolution of a mechanical part's boundary. The initial surface is obtained from an octree and using the conversion algorithm presented in Vinacua et al.[1]. The images correspond to the resolution levels after doing the decomposition process in the three directions. Therefore, there are two intermediate resolutions between two consecutive images. Tables 2 and 3 list the same parameters as the previous table, but here for the surface models. The wavelet transform of the skull has 178067 coefficients, 14.4% more than the original surface. The wavelet transform of the mechanical part has 50473 coefficients, 17.2% more than the original.

Because of the small thickness of the skull, the surface topology becomes more complicated (holes and solid fragmentation) in the intermediate resolutions.

Figure 16 shows the same mechanical surface as Figure 15, but now the simplification is achieved pruning one or more levels of the octree structure and using the conversion algorithm of Vinacua et al.[1]. From the images we realize that our wavelet multiresolution produces results closer to the initial object than pruning the octree, surely because of

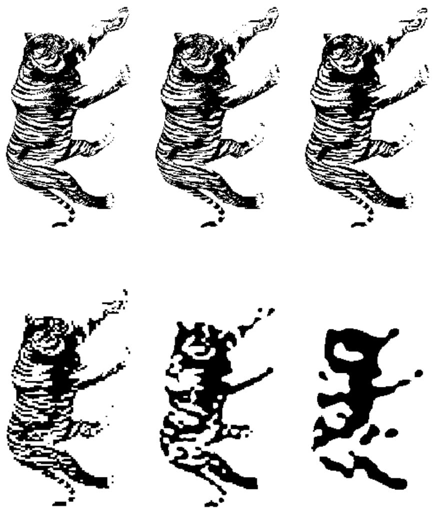
Figure 13: *Tiger multiresolution*

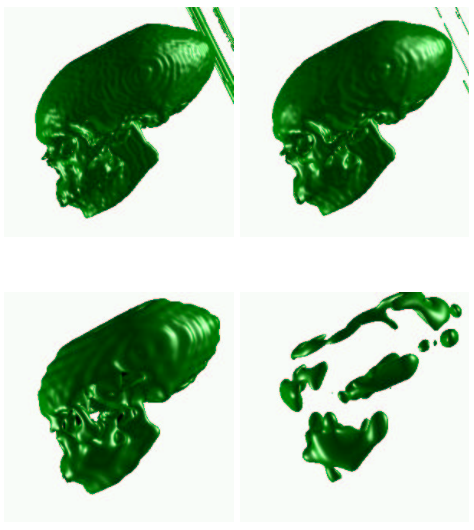
Figure 14: *Skull multiresolution*

the error minimization property of wavelet analysis and the good approximation properties of B-splines.

Note that in the first simplification steps we get objects very similar to the originals along with an important data reduction (if the detail data is thrown out). In fact, we could use the wavelet method as a compression algorithm if we only preserve the most simplified solid along with the significant detail coefficients. Due to the linear time cost in 2D and the quadratic cost in 3D, the multiresolution is calculated quickly. For example, the run-times of each decomposition step for the mechanical part are 6.067, 1.571, 0.465, 0.111, 0.037, 0.019, 0.012 seconds respectively. Each decomposition step consists in doing three decomposition substeps and changing the direction X, Y, Z of decomposition. These times have been obtained on a personal computer with a 350Mhz AMD-K6-2 CPU and 64 Mb of main memory.

Table 1: *Curve multiresolution: Africa*

Resolution level	Domain dimension	#defined coeff. (I)	#defined nodes $\cong$ (G)
1	367x343	22159	12380
1a	173x367	14077	7349
2	185x173	8624	4362
2a	88x185	5599	2816
3	94x88	3453	1785
3a	46x94	2225	1174
4	49x46	1345	780
4a	25x49	853	544
5	26x25	488	320
5a	14x26	293	182
6	15x14	175	97
6a	9x15	121	58
7	9x9	76	31
7a	6x9	52	16
8	6x6	35	8
8a	5x6	30	6
9	5x5	25	4
9a	4x5	20	2
10	4x4	16	1

6. Conclusions and Future Work

We have presented a multiresolution method to simplify the geometry and also the topology of curves and surfaces. The proposed scheme uses wavelet decomposition and obtains a set of multiresolution algebraic surface models.

By raising the problem's dimension and viewing the object as a level curve/surface of a function in one more dimension, we allow topological changes in the object when we analyse this function. The proposed algorithm calculates the simplification and reconstruction directly over the known control points of the function using wavelet multiresolution methods. This allows obtaining an object decomposition (the lowest resolution version plus several detail at different levels) that needs almost the same space as the initial object. Further, the detailed data (error) of the B-spline func-

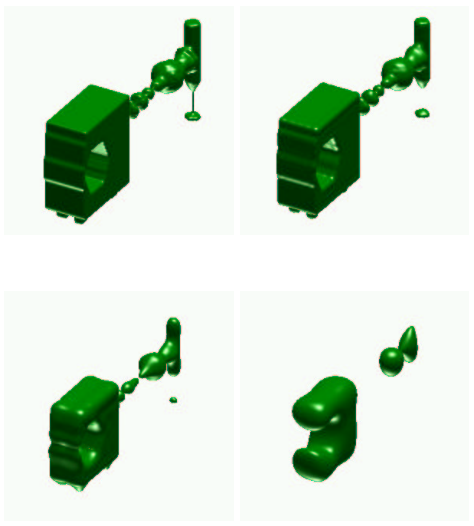

Figure 15: *Mechanical part multiresolution*

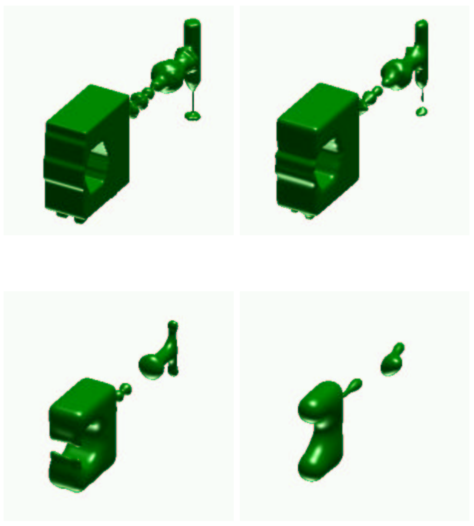

Figure 16: *Mechanical part multiresolution pruning the octree levels*

Table 2: *Surface multiresolution: Skull*

Resolution level	Domain dimension	#defined coeff. (I)	#defined nodes $\cong (G)$
1	97x97x70	155659	76649
1a	97x37x97	95641	45979
1b	37x50x97	56031	25582
2	50x50x37	32547	14022
2a	50x20x50	20214	8567
2b	20x27x50	12234	4949
3	27x27x20	7420	2877
3a	27x12x27	4679	1875
3b	12x15x27	2973	1062
4	15x15x12	1802	582
4a	15x8x15	1194	318
4b	8x9x15	811	190
5	9x9x8	517	112
5a	9x6x9	374	59
5b	6x6x9	270	36
6	6x6x6	180	18
6a	6x5x6	144	9
6b	5x5x6	120	6
7	5x5x5	100	4
7a	5x4x5	100	4
7b	4x4x5	80	2
8	4x4x4	64	1

tion that defines the algebraic curve/surface is minimized according to the inner function product $< f(u),\ g(u) >= \int f(u)\ g(u)\ du$.

It is not easy to describe how minimizing the B-spline error affects the B-spline zeroes that define the algebraic curve/surface. For example, we can get better approximations using B-spline functions with strong slopes around their zeroes, at least in the first simplification steps.

The algorithm uses wavelets on general domains (not restricted to 2^n). A specific wavelet scheme for analysis/synthesis on general even domains has been developed, including strategies for addition and rejection of control points. At each iteration, only B-spline control points in the vicinity of the zero isosurface are stored. As a consequence, the number of defined control points in general for the 3D case is proportional to n^2 - surface area of the object - and not to the dimension n^3 of the complete voxel domain. The wavelet algorithms present linear cost in 2D and quadratic cost in 3D, and 2D and 3D cases are based on repeated application of one-dimensional analysis and synthesis. The complexity of one simplification step is of the order of the total surface area of the object.

Working with algebraic isosurfaces of cubic B-splines is well suitable for applications that deal with smooth curves and surfaces. The main drawback of these algebraic isosurfaces is the inability to represent sharp features.

To improve the results and avoid shrinking/dilating the solid we are exploring the possibility to change the isosurface value or, on the other hand, to start with a initial ob-

Table 3: *Surface multiresolution: Mechanical part*

Resolution level	Domain dimension	#defined coeff. (I)	#defined nodes $\cong (G)$
1	129x129x129	43114	13204
1a	129x66x129	26973	8760
1b	66x66x129	17421	5451
2	66x66x66	10184	3080
2a	66x35x66	6459	2148
2b	35x35x66	4342	1612
3	35x35x35	2626	871
3a	35x19x35	1634	473
3b	19x19x35	1239	285
4	19x19x19	781	154
4a	19x11x19	504	102
4b	11x11x19	414	60
5	11x11x11	276	30
5a	11x7x11	198	18
5b	7x7x11	168	12
6	7x7x7	140	8
6a	7x5x7	100	4
6b	5x5x7	100	4
7	5x5x5	80	2
7a	5x4x5	64	1
7b	4x4x5	64	1
8	4x4x4	64	1

ject that has been modified with the same energy where the B-spline function is positive (solid) as where the B-spline function is negative (empty space).

It is interesting in complex models to save memory space. Thus, we are comparing several compression techniques to reduce the amount of detail or error. We are also studying other ways to set the accessory control points that, generating correct simplifications, could reduce the total error.

Also, we are working on directly computing a B-spline algebraic surface from a sufficiently dense set of points in 3D space.

In the future, we will work on the multiresolution editing of the curves and surfaces defined by this model. This is a very useful tool in multiresolution environments: we can edit in one multiresolution level and, later, add the other levels to get the object with local or global changes. To make this editing more useful, we are investigating ways to localize the simplification.

References

1. A. Vinacua, I. Navazo, and P. Brunet. Octtrees meet splines. In G. Farin, H. Bieri, G. Brunett, and T. DeRose, editors, *Geometric Modelling, Computing [Suppl]*, volume 13, pages 225–233. Springer-Verlag, 1998.
2. H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. *SIGGRAPH. Computer Graphics Proceedings, Annual Conference Series*, pages 19–26, 1993.
3. H. Hoppe. Progressive meshes. *SIGGRAPH. Computer Graphics Proceedings, Annual Conference Series*, pages 99–108, 1996.
4. J. Rossignac and P. Borrel. Multi-resolutions 3d approximations for rendering complex scenes. In B. Falcinedo and T. L. Kunii, editors, *Modelling in Computer Graphics*, pages 455–465. Springer-Verlag, 1993.
5. R. Ronfard and J. Rossignac. Full-range approximation of triangulated polyhedra. *Computer Graphics Forum (Proceedings of Eurographics)*, 15(3):67–76, 1996.
6. J. Popović and H. Hoppe. Progressive simplicial complexes. *SIGGRAPH. Computer Graphics Proceedings, Annual Conference Series*, pages 217–224, 1997.
7. M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. *SIGGRAPH. Computer Graphics Proceedings, Annual Conference Series*, pages 209–216, 1997.
8. T. He, L. Hong, A. Kaufman, A. Varshney, and S. Wang. Voxel based object simpliication. In G. M. Nielson and D. Silver, editors, *Visualization'95*, pages 296–303, Atlanta, GA, 1995.
9. C. Andújar, D. Ayala, P. Brunet, R. Joan-Arinyo, and J. Solé. Automatic generation of multiresolution boundary representations. *Computer Graphics Forum*, 15(3), 1996.
10. L.-M. Reissell. Wavelet multiresolution representation of curves and surfaces. *Graphical Models and Image Processing*, 58(3):198–217, 1996.
11. T. Lyche and K. Mørken. Spline-wavelets of minimal support. *Numerical Methods of Approximation Theory*, 9:177–194, 1992.
12. R. Kazinnik and G. Elber. Orthogonal decomposition of non-uniform bspline spaces using wavelets. *Computer Graphics Forum (Proceedings of Eurographics)*, 16(3):27–38, 1997.
13. A. Finkelstein and D. H. Salesin. Multiresolution curves. *SIGGRAPH. Computer Graphics Proceedings, Annual Conference Series*, pages 261–268, 1994.
14. M. Lounsbery, T. DeRose, and J. Warren. Multiresolution analysis for surfaces of arbitrary topological type.

Technical Report 93-10-05b, Dept. of Computer Science and Engineering. University of Washington, 1994.

15. M. Sramek and A. E. Kaufman. Alias-free voxelization of geometric objects. *IEEE transactions on visualization and computer graphics*, 5(3):251–267, 1999.
16. Charles K. Chui. *An introduction to Wavelets*. Academic Press, 1992.
17. J. Esteve, P. Brunet, and A. Vinacua. Multiresolution for algebraic curves and surfaces using wavelets. Technical Report LSI-98-60-R, http://www.lsi.upc.es/dept/techreps/1998.html, Dept. L.S.I., Universitat Politècnica de Catalunya, 1998.
18. E. Quak and N. Weyrich. Decomposition and reconstruction algorithms for spline wavelets on a bounded interval. Technical Report CAT 294, Center for Approximation Theory, Texas A&M University, 1993.
19. W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Fetterling. *Numerical Recipes*. Cambridge University Press, Cambridge, second edition edition, 1992.