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

October 23, 1998

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 algebraic isosurface of a tensor-product uniform cubic Bspline. A wavelet multiresolution method that deals with uniform cubic Bsplines on bounded domains has been constructed. Further, the report explains how to set the unknown coefficients to produce the most compact object, how to recover the initial object, a suitable data structure and, finally, points out several improvements that might produce better results.

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

1 Introduction

In this paper a 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 Bspline. Thus, the cubic Bspline has one dimension more than the space where we are representing the objects. The Bspline approximation will produce the simplification of the objects and the change of their topology in a simple way.

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 [HDD$^+$93, Hop96]. Some also make topological changes joining the nearest vertices of different triangles [RB93, RR96, PH97, GH97]. 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 bases in space decomposition models, usually voxels and octrees. It is the case of [HHK$^+$95] with voxels and [AAB$^+$96] 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 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. [Rei96] 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 Bspline surfaces (see section 3). Using the ideas on [LM92], [KE97] exposes a multiresolution over a parametric surface defined as a tensor-product of non-uniform Bsplines. [FS94] does the same over tensor-products of end-interpolating uniform cubic Bsplines. In [LDW94] the wavelets are applied over recursive subdivision surfaces with any topology, although they cannot simplify this topology.

In [VNB98], 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 Bspline 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 Bspline coefficients 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 any dimensional object. Section 4 exposes the particularities of our algorithm: how to set the unknown coefficients 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 Algebraic curves/surfaces in terms of Bsplines

For simplicity, we consider a scalar function defined on a rectangular grid and consider the intersection of its graph with the zero-valued plane as the curve (the boundary of the area). Because of the nature of the domain, we set out to construct this function as a tensor-product uniform cubic Bspline. The curve obtained is the zero-valued algebraic isosurface of the functional Bspline. See figure 1. It is a smooth, C^2, piecewise algebraic curve. To model a surface, we define the scalar function on a 3D regular grid and its intersection with the zero-valued hiperplane produces the boundary of the solid. The rectangular domain also allows straightforward conversion algorithms from voxel and octree models [VNB98].

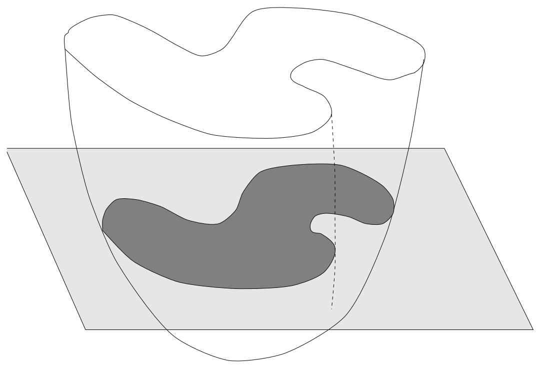


Figure 1: The intersection of the 3D Bspline function with the zero-valued plane defines a curve or area

It is only necessary to define the algebraic object on the nodes —cells of the grid— stabbed by the isosurface. This will form a set of nodes named Node-Collection. Let us now define G as the set of spatial indexes of the nodes of the Node-Collection

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

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 indexes of the nodes in the immediate vicinity of the Node-Collection:

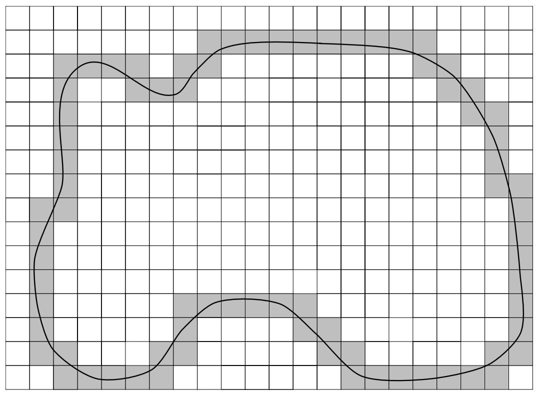


Figure 2: The Node-Collection is the set of shaded nodes

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

The cardinality of I is of the order of 4 times the cardinality of G. Let us now consider the uniform, tensor-product Bspline function F defined on the regular grid. F is an integral, functional spline and we are only interested to evaluate it on the Node-Collection, to find its zero-isosurface.

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

3 Wavelet multiresolution

Wavelets are a mathematical tool with a wide variety of applications in several fields. A lot of Bspline wavelet methods exist due to the variety of Bspline functions. In [FS94] there is a method for end-interpolating uniform cubic Bsplines. [Chu92] contains a study of the uniform Bspline wavelets on unbounded domains. In [LM92] and, later, in [KE97] the use of wavelets with non-uniform Bsplines is described.

In our case, due to the kind of algebraic surface to decompose, we will use uniform cubic Bsplines on bounded domains. We want to develop a wavelet multiresolution method, similar to the end-interpolating Bsplines in [FS94], implemented with pre-calculated band matrixes. This will allow us to do the analysis and synthesis processes with linear cost. Our method produces similar results as [Chu92] does, but uses wavelets defined on bounded domains. In [Chu92] the same filter is applied to all coefficients in the analysis/synthesis process (a moving average algorithm). The disadvantage of [Chu92] is that the analysis filters are infinite and we must truncate them causing precision errors.

Appendix A discusses the wavelet method for uniform cubic Bsplines on bounded domains. The direct use of this multiresolution has an important restriction: It can only work on uniform cubic Bsplines that have $2^i + 3$ control points. Our application will usually deal with Bsplines that have any number of control points and we would also want to decompose and reconstruct them. Section 4.1 explains how we do it.

4 The proposed algorithm

We have adapted the wavelet multiresolution to our application, since the Bspline function is not defined on the whole domain. It is defined on the nodes of the 3D grid that belong to the "Node-Collection". These are the contributions of our algorithm:

1. To decompose the Bspline functions we must define them on a domain more general than domains having 2^i intervals. We restrict the Bsplines to have an even number of intervals. Thus, we have extended the family of used analysis and synthesis matrices. This generalizes the usual analysis/synthesis process.
2. We must set some unknown coefficients (coefficients whose indexes do not belong to the set I) before decomposing the Bspline. The reasons are:
    - It is necessary that the Bsplines have an even number of intervals to decompose them, as per the previous observation.
    - Intervals which are close together can be joined in the decomposition. In this case, we must set the coefficients in between.
    - We are interested that the decomposed Bspline plus the produced detail use the least space possible (ideally the same as the original Bspline).
3. In the reconstruction we must recover the control points whose indexes belonged to the set I using the least information possible and keeping a linear time cost. We will also reject the coefficients that we have assigned before.

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 coefficients around these new areas of the Node-Collection.

4.1 Wavelet multiresolution on domains of even length

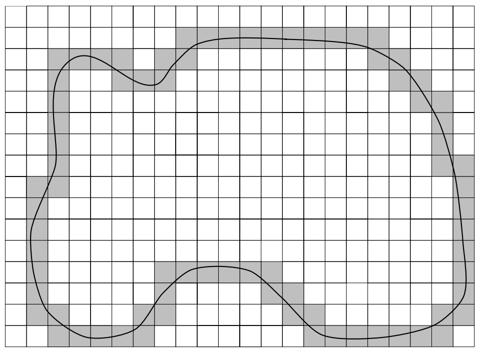
The single condition we will impose to the multiresolution is that the Bspline must have an even number of intervals where it is defined (the number of control points must be odd). In the decomposition process, the number of intervals will be divided by 2.

We change the indexes used in the equations of the analysis and synthesis process described in the appendix A to represent the decomposition of a Bspline defined on $2n$ intervals C^{2n} in one defined on n intervals C^n and its detail D^n and the inverse reconstruction. So,

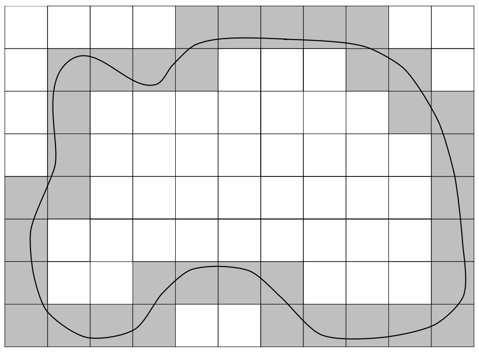
$$C^n = A^n C^{2n}$$

$$D^n = B^n C^{2n}$$

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



(a) Level 2n



(b) Level n

Figure 3: Changes in the Node-Collection

and the relations between the scaling functions and wavelet functions will be

$$\Phi^i(u) = \Phi^{2i}(u)P^i$$

$$\Psi^i(u) = \Phi^{2i}(u)Q^i$$

In the iterative application of the analysis process we can obtain a Bspline with an odd number of intervals. Then, we must set additional control points on the Bspline endpoints to return to an even interval Bspline and, in this way, continue the decomposition. Therefore, we need to add an assignment process between the decomposition steps. Also, between the reconstruction phases we must reject some unnecessary control points.

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

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

Appendix B explains how to achieve the decomposition process in linear time despite that the analysis matrices A^n and B^n are not band matrices. This can be achieved using the inner product matrices of the scaling and wavelet functions I^i and J^i and solving a band linear system.

Appendix C lists all the synthesis matrices P^i and Q^i and the inner product matrices I^i and J^i for the uniform cubic Bsplines defined on a domain with an even number of intervals.

4.2 Analysis process and the assignment of extra coefficients

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 the coefficients. Our purpose is to do the decomposition over the several sets of known control points separately.

For example, in the next case we decompose the 10th row (we work on unidimensional examples that represent the decomposition/reconstruction in one specific direction of the tensorial product).

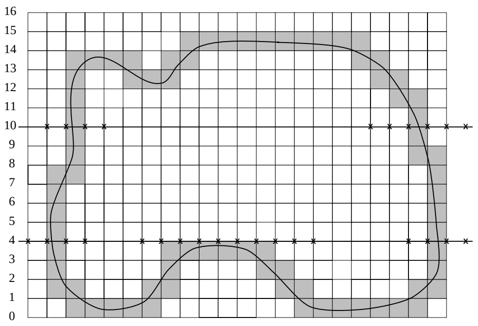


Figure 4: Horizontal Decomposition

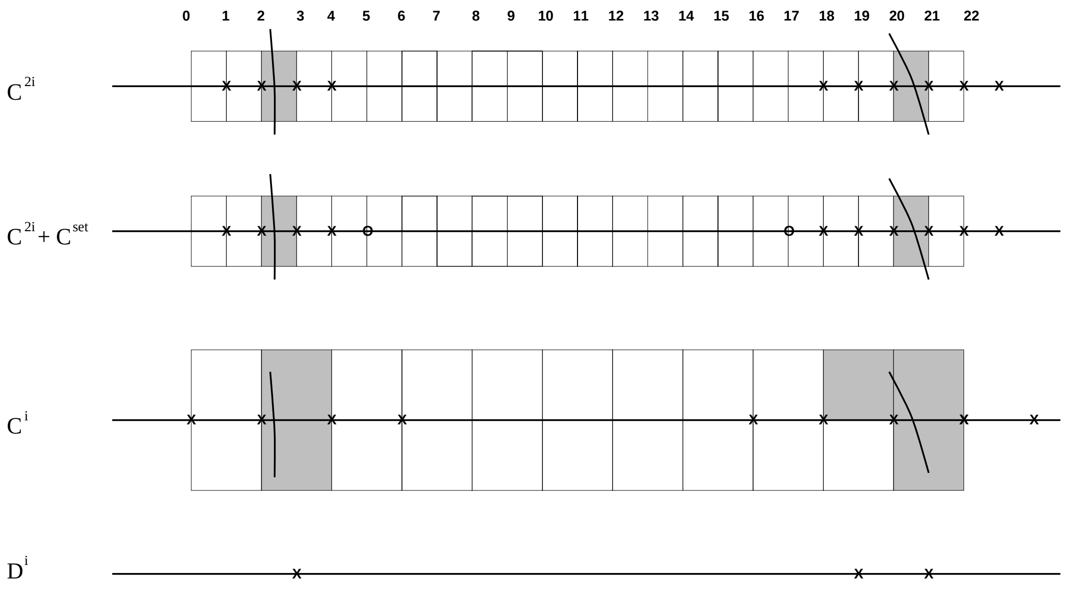


Figure 5: 10th row decomposition

First (see figure 5), the Bspline functional C^{2i} is defined on the intervals [2, 3] and [19, 22]. We have only assigned control points with odd indexes to get the Bspline defined on several domains with even indexes ([2, 4] and [18, 22]). Then we can apply the decomposition process described in the previous section to obtain C^i.

However, there is a problem: when working with a tensorial product of the basis functions on quadratic or cubic domains, after several steps of decomposition artifacts do appear (empty space inside the solid and/or solid inside the empty space). The reason is the basis functions of the uniform cubic Bsplines on a bounded domain are similar except on 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 endpoint control points in one direction can become inside control points in the other directions since we are working with a tensorial product of the basis functions and frequently changing 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.

Thus, our implementation sets 4 additional coefficients at both sides of each set of known coefficients. In fact, 3 additional coefficients at both sides are enough because the uniform cubic Bsplines on bounded domains have 3 truncated basis functions on each end.

We will keep only the coefficients of C^i and D^i needed to reconstruct the initial Bspline. Thanks to P^i being a band matrix (the middle columns have 5 non-zero coefficients) we can reject 2 coefficients at both sides of each interval of C^i. We can not do the same with D^i because Q^i has the middle columns wider than P^i (Q^i has 11 non-zero coefficients in the middle columns).

Therefore, each interval decomposition produces 4 additional coefficients in C^i plus D^i than the initials in C^{2i}. This problem can be avoided if in the reconstruction process we know the values assigned to the unknown coefficients of C^{2i} during the decomposition process. In the present work we set the unknown coefficients of C^{2i} 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 coefficients have the same sign and they coincide to the sign of the functional Bspline on 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 coefficients of D^i in the decomposition, because we take advantage of knowing several fragments of C^{2i}. As it can be seen in figure 6, we achieve that the number of coefficients stored in C^i plus D^i are the same as the initial number in C^{2i}.

Note, in the analysis process, if two intervals of known coefficients of C^{2i} are very close, they can be converted in one interval in C^i. This happens when the distance of the nearest endpoint coefficients of two intervals (situated on odd positions) is lower or equal than 4 (see figure 7).

4.3 Synthesis process and the recovery of coefficients

Let us see how to do the reconstruction $C^{2i} = P^i\ C^i + Q^i\ D^i$ under these conditions. We start by knowing the previous assigned coefficients of C^{2i} (with value +K or -K) and those stored in C^i and D^i. From the reconstruction point of view the known coefficients of C^{2i}, C^i and D^i of the second last example are marked in figure 8.

Splitting the known and unknown coefficients and applying the synthesis filter:

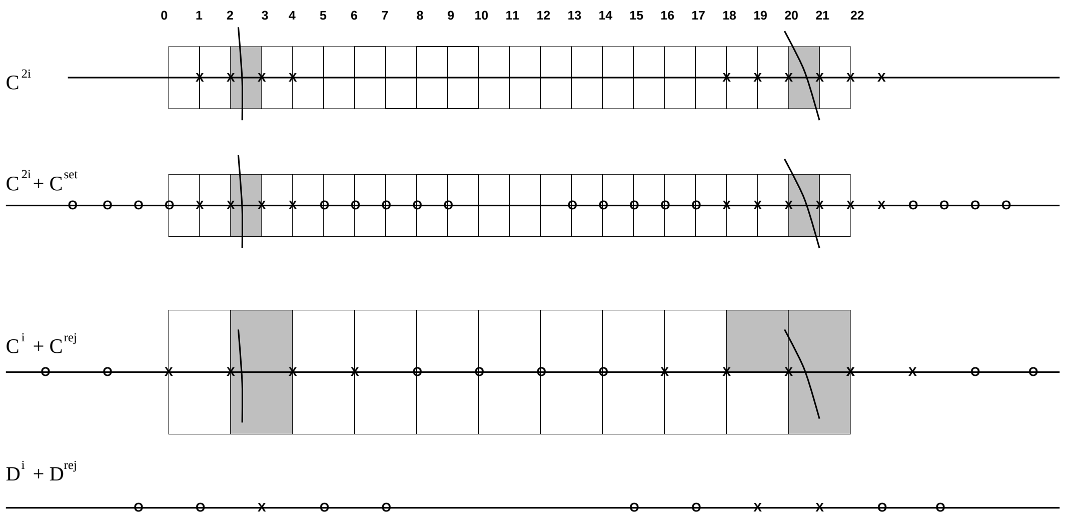


Figure 6: 10th row decomposition. 4 additional coefficients at both sides are assigned to C^{2i} and, later, 2 coefficients at both sides are rejected from C^i and D^i

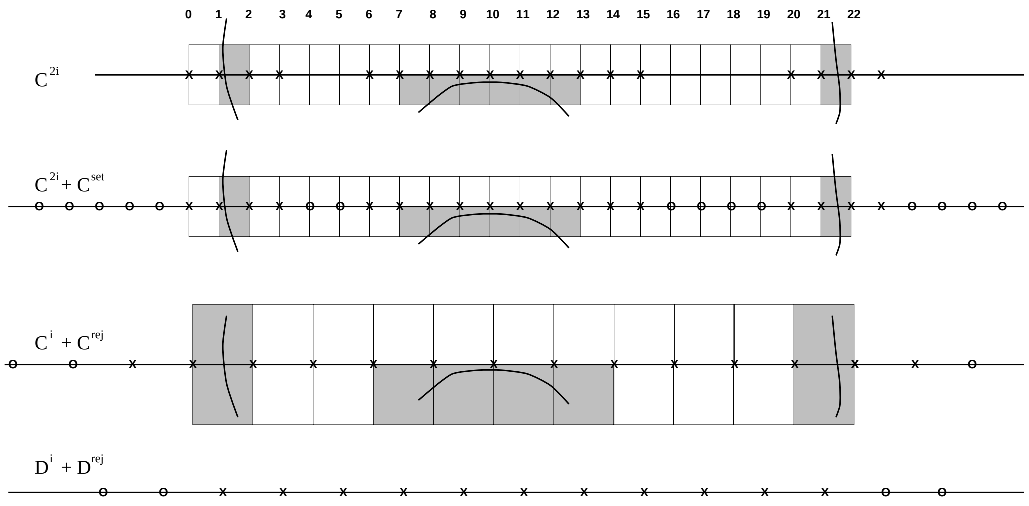


Figure 7: 4th row decomposition. Joining of 3 close intervals

$$C^{2i} = C^{2i}_k + C^{2i}_u$$
$$C^i = C^i_k + C^i_u$$
$$D^i = D^i_k + D^i_u$$
$$I(C^{2i}_k + C^{2i}_u) = P^i(C^i_k + C^i_u) + Q^i(D^i_k + D^i_u)$$
$$IC^{2i}_k - P^iC^i_k - Q^iD^i_k = P^iC^i_u + Q^iD^i_u - IC^{2i}_u$$

The left term in the last equation can be evaluated: it is the independent term V of the linear equation system. The right term can be expressed with a single matrix and a single vector

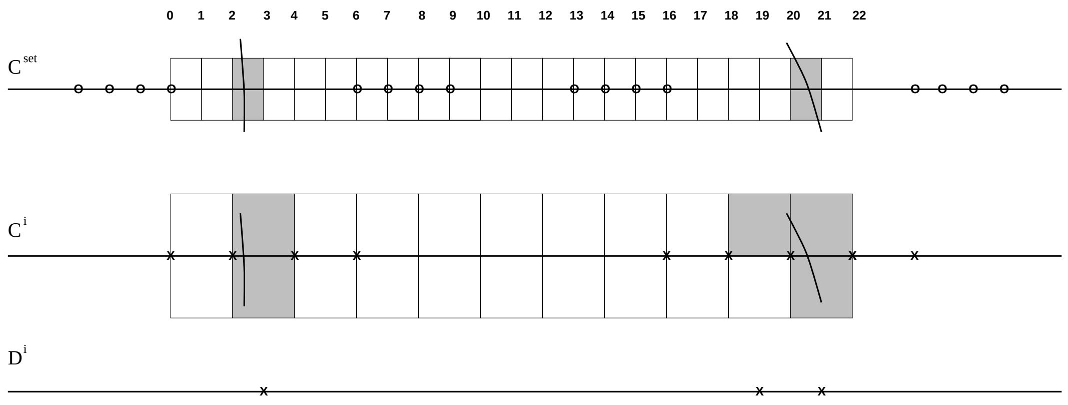


Figure 8: 10th row reconstruction. Known coefficients of C^{2i}, C^i and D^i

of unknown coefficients if we eliminate the matrix columns that are multiplied by zero (the vectors and matrices simplified are marked with ')

$$V = [P'^i | Q'^i | - I'][C'^i_u | D'^i_u | C'^{2i}_u]^T$$

Reordering the columns of matrix $[P'^i|Q'^i|-I']$, we transform it to a band matrix. So, we have a diagonal-band linear system that can be solved in linear time doing a LU decomposition [PFTF92].

In the reconstruction process we must reject the coefficients of C^{2i} 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 coefficients, see next section).

4.4 Data structure

Because our model only works with the control points that define the Node-Collection (the control points with spatial indexes contained in I, a set usually lower 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 hashing 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 coefficients in a compact way and to consult them in an almost constant time. The interval information about where the defined coefficients are, avoids searching for unknown coefficients 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 coefficients and the assigned ones.

Due to the alternation of the direction of the evaluations in both the analysis and the synthesis processes we must calculate the interval information where there are the known coefficients in one direction from the same information in other direction. To do so costs $O(n)$ in 2D and $O(n^2)$ in 3D (it is a run of the interval vector/matrix, we assume the same n dimension in all directions of the data domain and the number of intervals in a row is lower than n). This cost has the same order of magnitude as each decomposition and reconstruction step.

5 Examples and discussion

We have tested the proposed algorithm over several curves and surfaces. Figure 9 shows a multiresolution of the African continent, figure 10 of the American continent, figure 11 of a cinema camera, figure 12 of a skeleton and figure 13 of a tiger. 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 coefficients stored (spatial indexes of I) and the number of defined nodes (same order of magnitud as the Node-Collection G) for each resolution level.

Table 1: Curve multiresolution

	Africa			Tiger		
Resolution level	Domain dimension	#defined coeff. (I)	#defined nodes $\cong$ (G)	Domain dimension	#defined coeff. (I)	#defined nodes $\cong$ (G)
1	367x343	22159	12380	800x507	140397	110443
1a	173x367	14077	7349	255x800	76978	61349
2	185x173	8624	4362	402x255	41906	34557
2a	88x185	5599	2816	129x402	22676	18958
3	94x88	3453	1785	203x129	12088	10183
3a	46x94	2225	1174	66x203	6589	5390
4	49x46	1345	780	103x66	3566	2804
4a	25x49	853	544	35x103	2023	1474
5	26x25	488	320	53x35	1150	793
5a	14x26	293	182	19x53	691	454
6	15x14	175	97	28x19	393	258
6a	9x15	121	58	11x28	249	141
7	9x9	76	31	16x11	145	73
7a	6x9	52	16	7x16	104	44
8	6x6	35	8	10x7	66	24
8a	5x6	30	6	5x10	49	13
9	5x5	25	4	7x5	34	7
9a	4x5	20	2	4x7	28	4
10	4x4	16	1	5x4	20	2
10a	-	-	-	4x5	20	2
11	-	-	-	4x4	16	1

Figure 14 shows a multiresolution of a mechanical part's boundary. The initial surface is obtained from an octree and using the conversion algorithm explained in [VNB98]. Finally, figure 15 shows a multiresolution of a medical model (an image of the voxelization of a skull from a computerized tomography). Table 2 lists the same parameters as the previous table, but here for the surface models.

Table 2: Surface multiresolution

Resolution level	Mechanical part Domain dimension	Mechanical part #defined coeff. (I)	Mechanical part #defined nodes $\cong (G)$	Skull Domain dimension	Skull #defined coeff. (I)	Skull #defined nodes $\cong (G)$
1	129x129x129	43114	13204	97x97x70	155659	76649
1a	129x66x129	26973	8760	97x37x97	95641	45979
1b	66x66x129	17421	5451	37x50x97	56031	25582
2	66x66x66	10184	3080	50x50x37	32547	14022
2a	66x35x66	6459	2148	50x20x50	20214	8567
2b	35x35x66	4342	1612	20x27x50	12234	4949
3	35x35x35	2626	871	27x27x20	7420	2877
3a	35x19x35	1634	473	27x12x27	4679	1875
3b	19x19x35	1239	285	12x15x27	2973	1062
4	19x19x19	781	154	15x15x12	1802	582
4a	19x11x19	504	102	15x8x15	1194	318
4b	11x11x19	414	60	8x9x15	811	190
5	11x11x11	276	30	9x9x8	517	112
5a	11x7x11	198	18	9x6x9	374	59
5b	7x7x11	168	12	6x6x9	270	36
6	7x7x7	140	8	6x6x6	180	18
6a	7x5x7	100	4	6x5x6	144	9
6b	5x5x7	100	4	5x5x6	120	6
7	5x5x5	80	2	5x5x5	100	4
7a	5x4x5	64	1	5x4x5	100	4
7b	4x4x5	64	1	4x4x5	80	2
8	4x4x4	64	1	4x4x4	64	1

In the lowest resolutions the whole object disappears (there is only empty space). We think the reason is the solid space is smaller than the empty space in the highest resolution level (there is more empty space than solid space and on average the empty space wins).

Because of the little thickness of the skull, the surface topology becomes more complicated (holes and solid fragmentation) in the intermediate resolutions. However, in lower resolutions the topology is drastically simplified.

Figure 16 shows the same mechanical surface as figure 14, but now the simplification is achieved pruning one or more levels of the octree structure and using the conversion algorithm. From the images we realize that wavelets multiresolution produces results closer to the initial object than pruning the octree, due to the error minimization property of wavelet analysis.

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 can 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 of the one dimensional analysis, the multiresolution is calculated quickly.

6 Conclusions and future work

We have presented a multiresolution method to simplify the geometry and also the topology of curves and surfaces.

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 algorithm proposed 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) is minimized according to the inner function product $< f(u),\ g(u) >= \int f(u)\ g(u)\ du$.

We are currently studying another type of inner product that, instead of evaluating the function in the whole domain, gives more importance around their zeroes. Thus, we could get zero-valued isosurfaces (the objects) closer to the initials ones.

To avoid the solid disappearance in the lowest resolution levels we are exploring the possibility to change the isosurface value or, on the other hand, to start with a initial object that has been modified with the same energy where the Bspline function is positive (solid) as where the Bspline 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 unknown coefficients that, generating correct simplifications, could reduce the total error.

In the future, we will work on the multiresolution edition 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.

References

[AAB$^+$96] C. Andújar, D. Ayala, P. Brunet, R. Joan-Arinyo, and J. Solé. Automatic generation of multiresolution boundary representations. *Computer Graphics Forum*, 15(3), 1996.

[Chu92] Charles K. Chui. *An introduction to Wavelets.* Academic Press, 1992.

[Far92] G. Farin. *Curves and surfaces for computer aided geometric design.* Academic Press, 3rd edition edition, 1992.

[FS94] A. Finkelstein and D. H. Salesin. Multiresolution curves. *SIGGRAPH. Computer Graphics Proceedings, Annual Conference Series*, pages 261–268, 1994.

[GH97] M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. *SIGGRAPH. Computer Graphics Proceedings, Annual Conference Series*, pages 209–216, 1997.

[HDD$^+$93] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. *SIGGRAPH. Computer Graphics Proceedings, Annual Conference Series*, pages 19–26, 1993.

[HHK$^+$95] 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.

[Hop96] H. Hoppe. Progressive meshes. *SIGGRAPH. Computer Graphics Proceedings, Annual Conference Series*, pages 99–108, 1996.

[KE97] 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.

[LDW94] 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.

[LM92] T. Lyche and K. Mørken. Spline-wavelets of minimal support. *Numerical Methods of Approximation Theory*, 9:177–194, 1992.

[Mal89] S. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE transactions on pattern analysis and machine intelligence*, 11(7):674–693, 1989.

[PFTF92] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Fetterling. *Numerical Recipes.* Cambridge University Press, Cambridge, second edition edition, 1992.

[PH97] J. Popović and H. Hoppe. Progressive simplicial complexes. *SIGGRAPH. Computer Graphics Proceedings, Annual Conference Series*, pages 217–224, 1997.

[RB93] 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.

[Rei96] L.-M. Reissell. Wavelet multiresolution representation of curves and surfaces. *Graphical Models and Image Processing*, 58(3):198–217, 1996.

[RR96] R. Ronfard and J. Rossignac. Full-range approximation of triangulated polyhedra. *Computer Graphics Forum (Proceedings of Eurographics)*, 15(3):67–76, 1996.

[VNB98] A. Vinacua, I. Navazo, and P. Brunet. Octrees meet splines. *Technical report. Dept. L.S.I., U.P.C.*, 1998.

A Wavelets for uniform cubic B-splines on bounded domains

In the next sections we will describe the basic ideas about multiresolution analysis on wavelets and we will apply them it in the specific case of uniform cubic B-splines on bounded domains. Rather than showing the classical multiresolution analysis developed by Mallat [Mal89], we will work on a more generic multiresolution analysis following [LDW94] and [FS94] where it doesn't matter that the basis functions are constructed translating and scaling a single function. This is more convenient for our application, because the purpose is working on bounded domains of scaling and wavelet functions.

A.1 Wavelets multiresolution on bounded domains

We will deal with B-spline functions $f^n(u)$ defined as a column vector of m control points C^n. The B-spline functions can be evaluated weighting up m basis functions with the values of the control points C^n. Let $\Phi^n(u)$ be the row vector of B-spline basis functions with parameter u, the function $f^n(u)$ can be expressed as

$$f^n(u) = \Phi^n(u)C^n$$

Our goal is to approximate the B-spline $f^n(u)$ with another function of lower resolution $f^{n-1}(u)$. This second B-spline will depend on fewer base functions $\Phi^{n-1}(u)$. It will be defined as a vector of m' control points C^{n-1}, with $m' < m$.

$$f^{n-1}(u) = \Phi^{n-1}(u)C^{n-1}$$

Let's assume that our approximation of $f^n(u)$ with $f^{n-1}(u)$ is linear. That is, the approximation process can be represented as an $m' \times m$ matrix A^n

$$C^{n-1} = A^n C^n$$

If the matrix A^n is of maximal rank, we can capture the details lost in the filtering process in another vector D^{n-1}, computed by an $(m - m') \times m$ matrix B^n

$$D^{n-1} = B^n C^n$$

The process of splitting the B-spline C^n into a low resolution version C^{n-1} and detail D^{n-1} is called analysis or decomposition. A^n and B^n are the analysis filters. If they are chosen so that their columns are linearly independent, the original B-spline C^n can be recovered from C^{n-1} and D^{n-1} by the synthesis filters P^n and Q^n in this following way:

$$C^n = P^n C^{n-1} + Q^n D^{n-1}$$

This process is called synthesis or reconstruction.

The analysis process can be applied recursively to the B-spline C^{n-1} and thus obtain a set of lower resolution B-splines $C^{n-1}, ..., C^0$ and details $D^{n-1}, ..., D^0$.

$$\begin{tikzcd}
\textit{Analysis} & C^n \arrow[r, "A^n"] \arrow[dr, "B^n"'] & C^{n-1} & \cdots & C^1 \arrow[r, "A^1"] \arrow[dr, "B^1"'] & C^0 \\
 & & D^{n-1} & \cdots & & D^0
\end{tikzcd}$$

We can also recover the higher resolution B-spline C^n from the lower resolution C^0 and the details $D^{n-1}, \dots, D^0$. We have a multiresolution of the original B-spline C^n.

$$\begin{tikzcd}
\textit{Synthesis} & C^n & C^{n-1} \arrow[l, "P^n"'] & \cdots & C^1 & C^0 \arrow[l, "P^1"'] \\
 & & D^{n-1} \arrow[ul, "Q^n"] & \cdots & & D^0 \arrow[ul, "Q^1"]
\end{tikzcd}$$

The information $C^0, D^0, ..., D^{n-1}$ is the wavelet transform of the B-spline C^n. We don't need additional storage to save the wavelet transform (the total size of the $C^0, D^0, ..., D^{n-1}$ is the same as the size of the original B-spline C^n).

To compute the wavelet transform we need an appropriate set of analysis and synthesis filters A^n, B^n, P^n and Q^n. To this end, let us look at the space of B-splines. As stated, each vector C^n represents a B-spline function as a linear combination of a set of basis functions also called scaling functions: $f^n(u) = \Phi^n(u)C^n$. The scaling functions are refinable, that is: each scaling function at level $i-1$ —lower resolution function— is a linear combination of scaling functions at level i —higher resolution functions— because the higher resolution levels are obtained via knot insertion. So, for $i \in [1, n]$ we may set P^i so that

$$\Phi^{i-1}(u) = \Phi^i(u)P^i.$$

This relation defines our synthesis filters P^i.

Let V^i be the B-spline function space spanned by the set of scaling functions at level i, $\Phi^i(u)$. The fact that the scaling functions are refinable implies that these spaces are nested: $V^0 \subset V^1 \subset \ldots \subset V^n$. If we choose an inner product in the space V^{i+1}, we can define the space W^i as the orthogonal complement of V^i in V^{i+1}. The elements of a basis $\Psi^i(u)$ for W^i are called wavelet functions. The basis functions $\Psi^{i-1}(u)$ can also be expressed as a linear combination of the scaling functions $\Phi^i(u)$. Thus we define (for a certain choice of basis in W^i) the synthesis matrix Q^i as that satisfying

$$\Psi^{i-1}(u) = \Phi^i(u)Q^i$$

The last two equations can be expressed as a single equation joining the matrices together

$$[\Phi^{i-1}(u)|\Psi^{i-1}(u)] = \Phi^i(u)[P^i|Q^i]$$

The analysis matrices can be calculated as the matrices that satisfy the inverse relation

$$[\Phi^{i-1}(u)|\Psi^{i-1}(u)] \left[\frac{A^i}{B^i}\right] = \Phi^i(u)$$

In fact, the matrices $[P^i|Q^i]$ and $\left[\frac{A^i}{B^i}\right]$ are square matrices. So

$$\left[\frac{A^i}{B^i}\right] = [P^i|Q^i]^{-1}$$

A.2 Wavelet multiresolution for uniform cubic B-splines

As previously discussed, we want to work with multiresolutions of uniform cubic B-splines that will permit us to define a family of implicit curves and surfaces. Thus, we will handle B-splines that have their knot sequence uniformly spaced and with all knots of multiplicity 1. For a deeper analysis of B-splines, see for instance [Far92].

First, we must choose the scaling functions $\Phi^i(u)$ for $i \in [0, n]$ to determine the synthesis filters P^i. We will select the basis functions of the uniform cubic B-splines defined with 2^i interior segments. There are $2^i + 3$ scaling functions at each level $i \in [0, n]$. In one level, all basis functions are translations of the same basis function, except at the endpoints where the basis functions are truncated. The matrix P^i can easily be derived from the Cox-de Boor recursion formula which shows how a B-spline can be expressed as a linear combination of B-splines whose knot intervals are half as wide. In the case of the uniform cubic B-splines, each scaling function of V^i is a combination of 5 scaling functions of V^{i+1} with weights $1/8, 4/8, 6/8, 4/8$ and $1/8$ (except for the endpoints). Thus, the matrix P^i has only non-zero coefficients around the main diagonal (it is a band matrix).

Next, to determine the orthogonal complement spaces W^i of V^i, we must define the inner product for any two functions $f(u)$ and $g(u)$. We will select the L^2 inner product: $< f(u),\ g(u) >= \int f(u)\ g(u)\ du$. To achieve an efficient reconstruction process we want the Q^i matrices to be band matrices as the P^i are. This means that we want the 2^i wavelet functions of the W^i space to have minimal support. The Q^i matrix can be calculated in the following way

Let $[< F(u)|G(u) >]$ denote the matrix of inner products $< f_j(u), g_k(u) > \forall j, k$. Since scaling functions and wavelet functions at the same level $i-1$ are orthogonal, we have

$$[< \Phi^{i-1}(u)|\Phi^i(u) >]Q^i = [< \Phi^{i-1}(u)|\Psi^{i-1}(u) >] = 0$$

Hence the columns of Q^i are a basis of the null space of $[< \Phi^{i-1}(u)|\Phi^i(u) >]$. We construct this basis by selecting the columns of Q^i that have the shortest number of successive non-zero coefficients. This is equivalent to having minimal support wavelet functions (since in the relation $\Psi^{i-1}(u) = \Phi^i(u)Q^i$, the scaling functions $\Phi^i(u)$ of the uniform cubic B-splines have minimal support and each function is a translation of the previous one). Q^i has 2^{i-1} columns, which is the dimension of the null space of matrix $[< \Phi^{i-1}(u)|\Phi^i(u) >]$ (the rank of this $2^{i-1} + 3\ \times\ 2^i + 3$ matrix is always $2^{i-1} + 3$ because of the properties of the scaling functions discussed before).

In appendix C we list the synthesis matrices P^i and Q^i for the uniform cubic B-splines. Due to a change in the matrix numeration described in section 3, the synthesis matrices for uniform cubic B-splines on 2^i intervals are the ones that have powers of 2 ($P^1, P^2, P^4, \ldots$; $Q^1, Q^2, Q^4, \ldots$)

B linear time decomposition

The reconstruction has a linear cost since the matrices P^i and Q^i are band matrices. Rather, the matrices A^i and B^i, calculated as the inverse of the previous ones, are dense. So, if we use directly the matrices A^i and B^i, the decomposition has a quadratic cost. To avoid it, we can rewrite the analysis equations as

$$I^i C^i = [P^i]^T I^{2i} C^{2i}$$
$$J^i D^i = [Q^i]^T I^{2i} C^{2i}$$

being I^i and J^i the inner product diagonal-band matrices of the scaling and wavelet functions respectively

$$I^i = [< \Phi^i(u) | \Phi^i(u) >] = [\Phi^i(u)]^T \Phi^i(u)$$
$$J^i = [< \Psi^i(u) | \Psi^i(u) >] = [\Psi^i(u)]^T \Psi^i(u)$$

Since P^i, Q^i and I^i are band matrices, the right term can be calculated in linear time. To get C^n and D^n we must solve two diagonal-band linear systems. It can be done in linear time through a LU decomposition [PFTF92].

Proof:

If the function $f^{2i}(u)$ described by the C^{2i} coefficients or by the approximation C^i and detail D^i

$$f^{2i}(u) = \Phi^{2i}(u) C^{2i} = \Phi^{2i}(u) P^i C^i + \Phi^{2i}(u) Q^i D^i = \Phi^i(u) C^i + \Psi^i(u) D^i$$

is multiplied by $[\Phi^i(u)]^T$

$$[\Phi^i(u)]^T \Phi^{2i}(u) C^{2i} = [\Phi^i(u)]^T \Phi^i(u) C^i + [\Phi^i(u)]^T \Psi^i(u) D^i$$

we get the first equation (since the basis $\Phi^i(u)$ and $\Psi^i(u)$ are orthogonal and the $\Phi^i(u)$ basis can be represented by the $\Phi^{2i}(u)$ basis and the P^i matrix):

$$[P^i]^T [\Phi^{2i}(u)]^T \Phi^{2i}(u) C^{2i} = [\Phi^i(u)]^T \Phi^i(u) C^i$$
$$[P^i]^T I^{2i} C^{2i} = I^i C^i$$

The second equation can be obtained in a similar way: multiplying the function $f^{2i}(u)$ by $[\Psi^i(u)]^T$.

C Multiresolution matrices

Following are listed the synthesis matrices P^i and Q^i for uniform cubic Bsplines that allow reconstruction of the initial Bspline doubling the number of intervals where it is defined. P^i is a $(2i+3) \times (i+3)$ matrix whose central columns, for $i \geq 2$, are vertical translations of the third column, shifted down by 2 places for each column. Q^i is a $(2i+3) \times i$ matrix whose middle columns, for $i \geq 7$, are vertical translations of the fourth column, shifted down by 2 places also.

The matrices I^i and J^i used in the linear system to do the analysis in linear time are symmetric band-diagonal matrices. I^i is a $(i+3)\times(i+3)$ matrix whose central columns, for $i \geq 4$, are vertical translations of the fourth column, shifted down by 1 place for each column. Further, each I^i matrix is multiplied by the $1/(10080i)$ factor. The matrices I^i for $i < 4$ and the matrices J^i (matrices with dimensions $i \times i$) can be found with the following relations

$$I^i = [P^i]^T I^{2i} P^i$$
$$J^i = [Q^i]^T I^{2i} Q^i$$

The matrices J^i, $i \geq 19$, may be also calculated from J^{19} matrix: the middle columns are similar, each one shifted down by 1 place and the whole matrix must be multiplied by a scaling factor.

Thanks to these repetitions, all matrices needed for the analysis and synthesis of any dimensional uniform cubic Bspline are precalculated and the memory space to store them is small.

$$P^1 = \frac{1}{8}\begin{bmatrix} 4 & 4 & 0 & 0 \\ 1 & 6 & 1 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 1 & 6 & 1 \\ 0 & 0 & 4 & 4 \end{bmatrix}$$

$$P^{i\geq 2} = \frac{1}{8}\begin{bmatrix} 4 & 4 & 0 & \\ 1 & 6 & 1 & \\ 0 & 4 & 4 & \\ 0 & 1 & 6 & \cdots \\ 0 & 0 & 4 & \\ 0 & 0 & 1 & \\ & \vdots & & \ddots \end{bmatrix}$$

$$Q^1 = \frac{1}{2}\begin{bmatrix} 14 \\ -3 \\ 2 \\ -3 \\ 14 \end{bmatrix}$$

$$Q^2 = \begin{bmatrix} 1 & 0 \\ \frac{-42821}{241982} & \frac{-1775}{120991} \\ \frac{9421}{120991} & \frac{2838}{120991} \\ \frac{-39}{908} & \frac{-39}{908} \\ \frac{2838}{120991} & \frac{9421}{120991} \\ \frac{-1775}{120991} & \frac{-42821}{241982} \\ 0 & 1 \end{bmatrix}$$

$$Q^3 = \begin{bmatrix} 1 & 0 & 0 \\ \frac{-33780837}{194145502} & \frac{-19660304}{58856313} & 0 \\ \frac{56856795}{776582008} & \frac{10217118}{19618771} & \frac{195139}{776582008} \\ \frac{-218744309}{6212656064} & \frac{-195888875}{235425252} & \frac{-17036965}{6212656064} \\ \frac{20003959}{1553164016} & 1 & \frac{20003959}{1553164016} \\ \frac{-17036965}{6212656064} & \frac{-195888875}{235425252} & \frac{-218744309}{6212656064} \\ \frac{195139}{776582008} & \frac{10217118}{19618771} & \frac{56856795}{776582008} \\ 0 & \frac{-19660304}{58856313} & \frac{-33780837}{194145502} \\ 0 & 0 & 1 \end{bmatrix}$$

$$Q^4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{-157179839}{903732526} & \frac{-1075797843315}{3028371961174} & 0 & 0 \\ \frac{33030599}{451866263} & \frac{836687879553}{1514185980587} & \frac{1355598789}{216312282941} & 0 \\ \frac{-633094403}{18074650520} & \frac{-5275614974873}{6056743922348} & \frac{-8993481151}{131668346138} & \frac{-27877}{18074650520} \\ \frac{57250023}{4518662630} & 1 & \frac{469776758380}{1514185980587} & \frac{864187}{4518662630} \\ \frac{-4681957}{1807465052} & \frac{-1103604194298}{1514185980587} & \frac{-1103604194298}{1514185980587} & \frac{-4681957}{1807465052} \\ \frac{864187}{4518662630} & \frac{469776758380}{1514185980587} & 1 & \frac{57250023}{4518662630} \\ \frac{-27877}{18074650520} & \frac{-8993481151}{131668346138} & \frac{-5275614974873}{6056743922348} & \frac{-633094403}{18074650520} \\ 0 & \frac{1355598789}{216312282941} & \frac{836687879553}{1514185980587} & \frac{33030599}{451866263} \\ 0 & 0 & \frac{-1075797843315}{3028371961174} & \frac{-157179839}{903732526} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$
Q^5 = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
\frac{-157179839}{903732526} & \frac{-3647617410915}{10251422561674} & 0 & 0 & 0 \\
\frac{33030599}{451866263} & \frac{2836718912970}{5125711280837} & \frac{3787764705}{566651652857} & 0 & 0 \\
\frac{-633094403}{18074650520} & \frac{-17880242830937}{20502845123348} & \frac{-660465038201}{9066426445712} & \frac{-451866263}{11715911499056} & 0 \\
\frac{57250023}{4518662630} & 1 & \frac{747737600591}{2266606611428} & \frac{14007854153}{2928977874764} & 0 \\
\frac{-4681957}{1807465052} & \frac{-59562600894529}{82011380493392} & \frac{-6921777645829}{9066426445712} & \frac{-2652557686421}{41005690246696} & \frac{-27877}{18074650520} \\
\frac{864187}{4518662630} & \frac{6270590805851}{20502845123348} & 1 & \frac{6270590805851}{20502845123348} & \frac{864187}{4518662630} \\
\frac{-27877}{18074650520} & \frac{-2652557686421}{41005690246696} & \frac{-6921777645829}{9066426445712} & \frac{-59562600894529}{82011380493392} & \frac{-4681957}{1807465052} \\
0 & \frac{14007854153}{2928977874764} & \frac{747737600591}{2266606611428} & 1 & \frac{57250023}{4518662630} \\
0 & \frac{-451866263}{11715911499056} & \frac{-660465038201}{9066426445712} & \frac{-17880242830937}{20502845123348} & \frac{-633094403}{18074650520} \\
0 & 0 & \frac{3787764705}{566651652857} & \frac{2836718912970}{5125711280837} & \frac{33030599}{451866263} \\
0 & 0 & 0 & \frac{-3647617410915}{10251422561674} & \frac{-157179839}{903732526} \\
0 & 0 & 0 & 0 & 1
\end{bmatrix}
$$

$$Q^{i\geq 6} = \begin{bmatrix} 1 & 0 & 0 & 0 & \\ \frac{-157179839}{903732526} & \frac{-3647617410915}{10251422561674} & 0 & 0 & \\ \frac{33030599}{451866263} & \frac{2836718912970}{5125711280837} & \frac{751756320}{112272017197} & 0 & \\ \frac{-633094403}{18074650520} & \frac{-17880242830937}{20502845123348} & \frac{-32770468855}{449088068788} & \frac{-1}{24264} & \\ \frac{57250023}{4518662630} & 1 & \frac{37097805865}{112272017197} & \frac{31}{6066} & \\ \frac{-4681957}{1807465052} & \frac{-59562600894529}{82011380493392} & \frac{-171639244051}{224544034394} & \frac{-559}{8088} & \\ \frac{864187}{4518662630} & \frac{6270590805851}{20502845123348} & 1 & \frac{988}{3033} & \\ \frac{-27877}{18074650520} & \frac{-2652557686421}{41005690246696} & \frac{-85413071471}{112272017197} & \frac{-9241}{12132} & \\ 0 & \frac{14007854153}{2928977874764} & \frac{36513354727}{112272017197} & 1 & \cdots \\ 0 & \frac{-451866263}{11715911499056} & \frac{-15492929849}{224544034394} & \frac{-9241}{12132} & \\ 0 & 0 & \frac{572783931}{112272017197} & \frac{988}{3033} & \\ 0 & 0 & \frac{-18476901}{449088068788} & \frac{-559}{8088} & \\ 0 & 0 & 0 & \frac{31}{6066} & \\ 0 & 0 & 0 & \frac{-1}{24264} & \\ & \vdots & & & \ddots \end{bmatrix}$$

$$I^{i\geq 4} = \frac{1}{10080i} \begin{bmatrix} 40 & 258 & 120 & 2 & \\ 258 & 2416 & 2124 & 240 & \\ 120 & 2124 & 4792 & 2382 & \\ 2 & 240 & 2382 & 4832 & \cdots \\ 0 & 2 & 240 & 2382 & \\ 0 & 0 & 2 & 240 & \\ 0 & 0 & 0 & 2 & \\ & \vdots & & & \ddots \end{bmatrix}$$

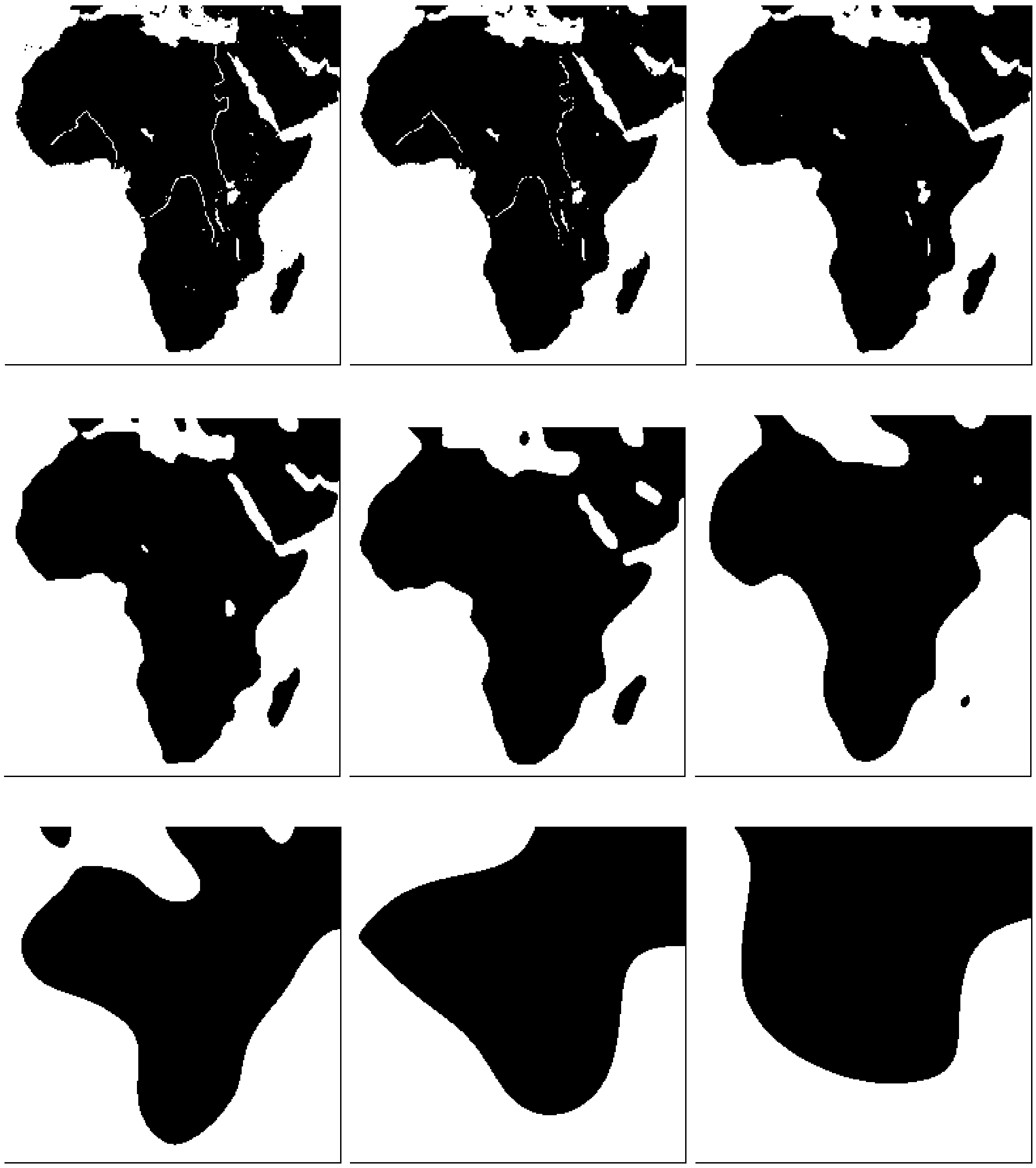


Figure 9: Africa multiresolution

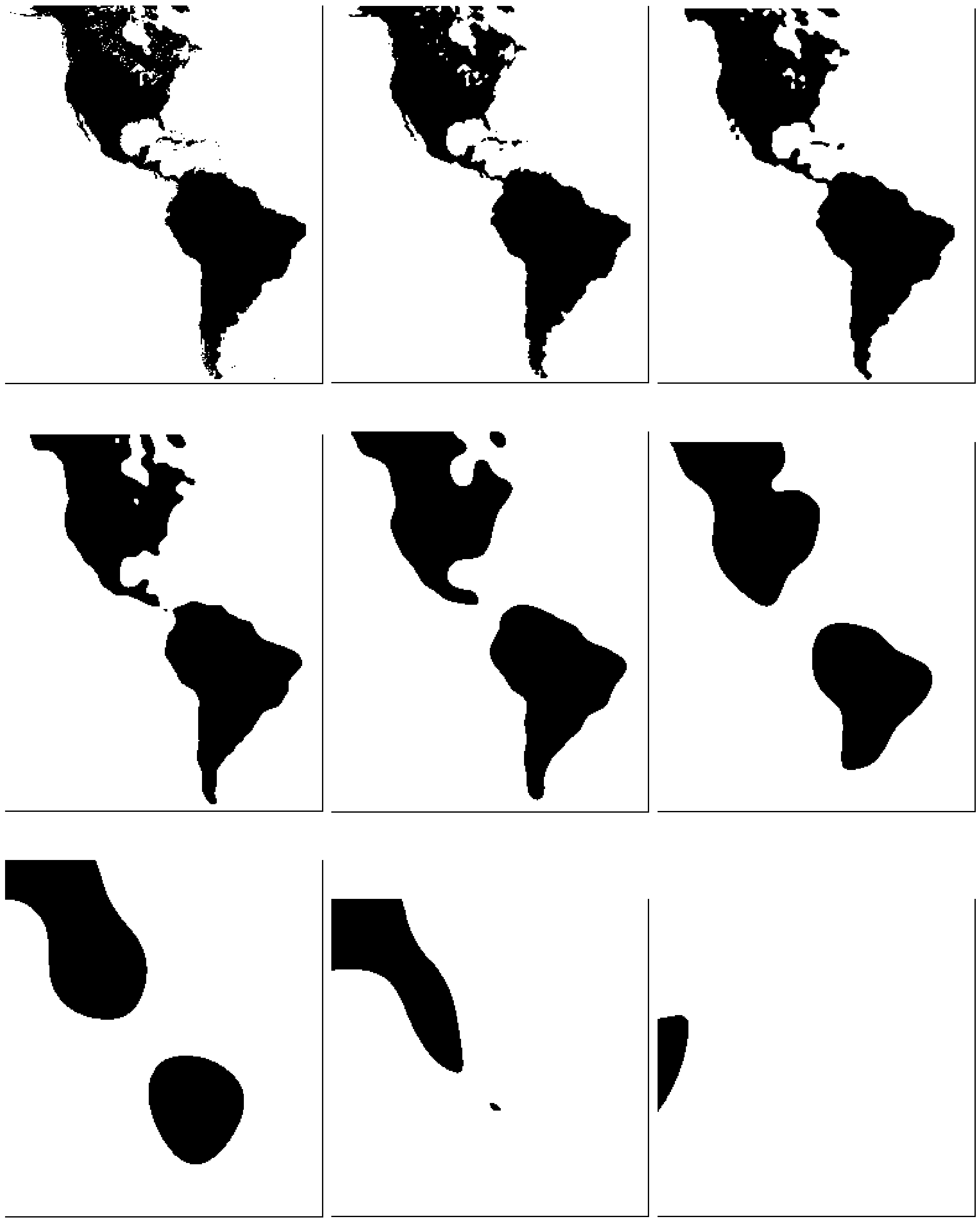


Figure 10: America multiresolution

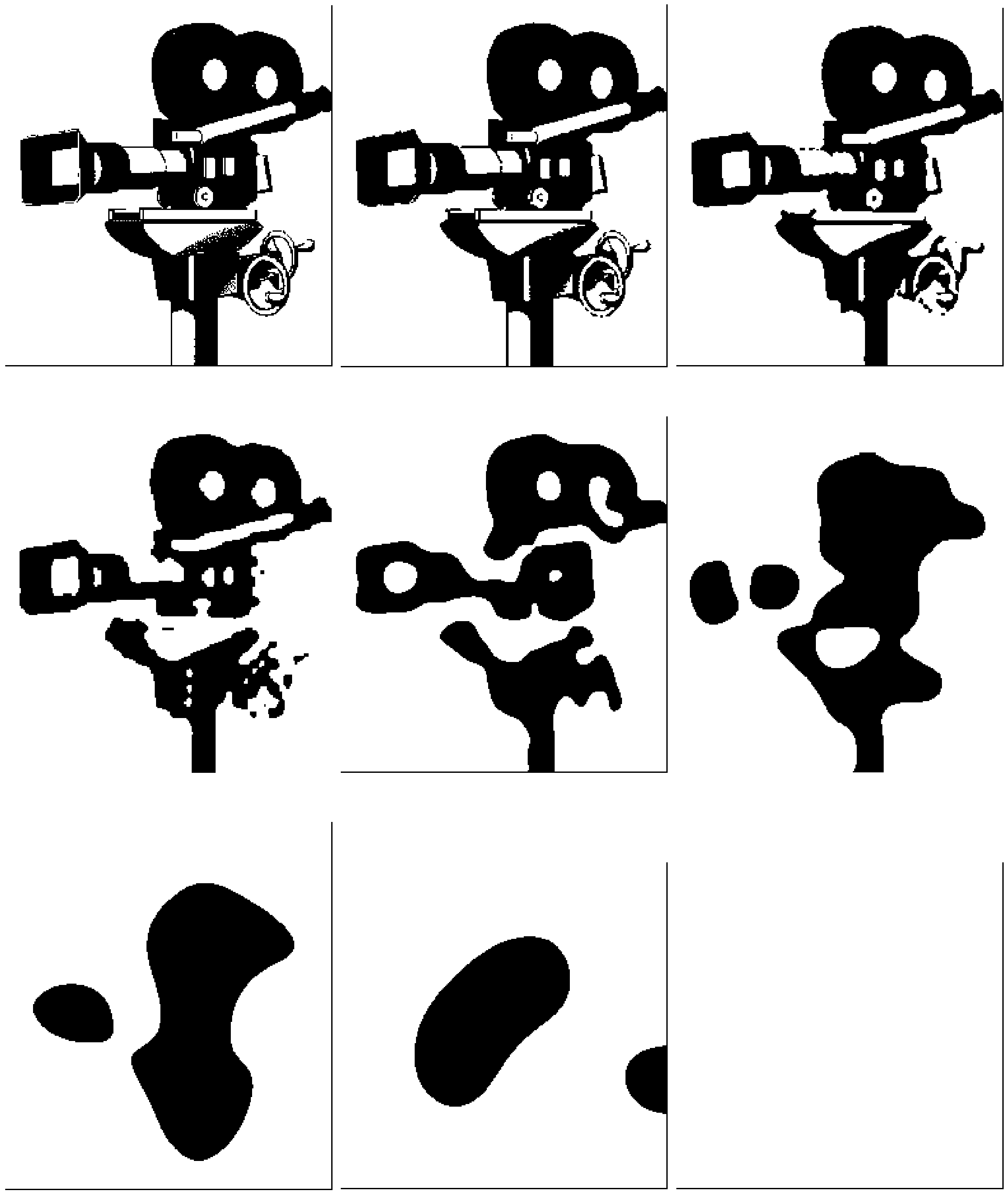
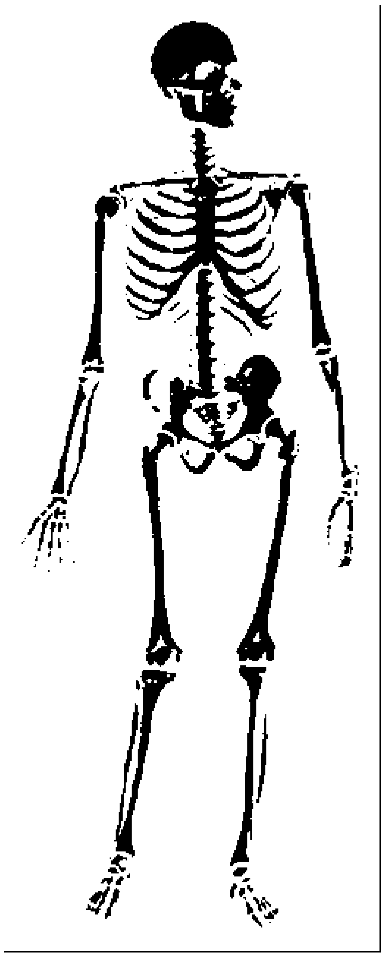
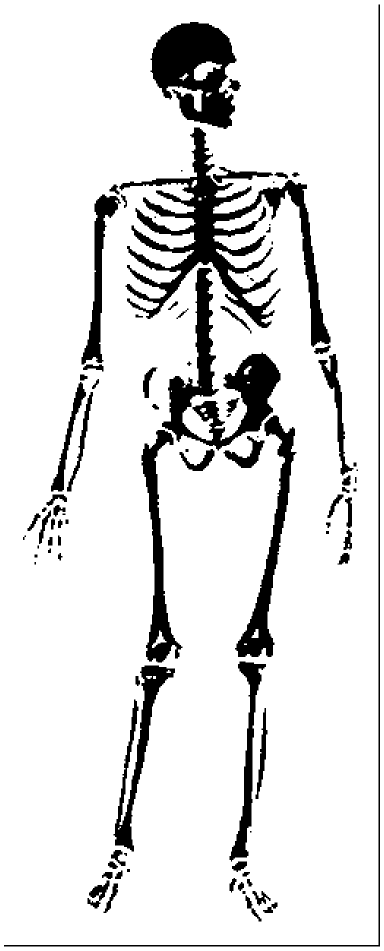
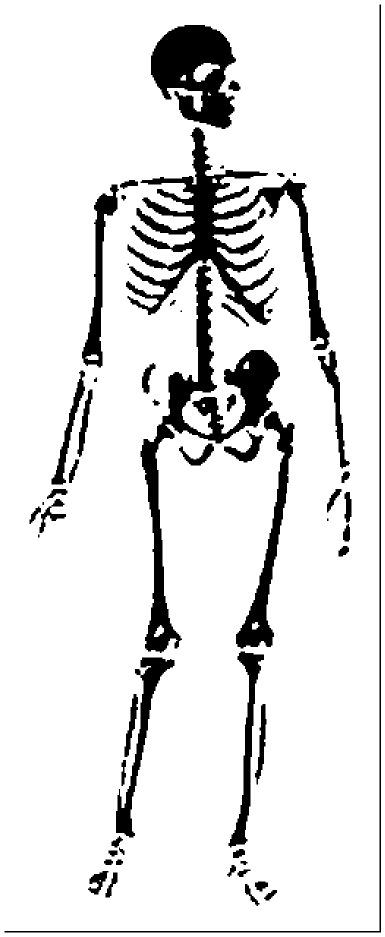
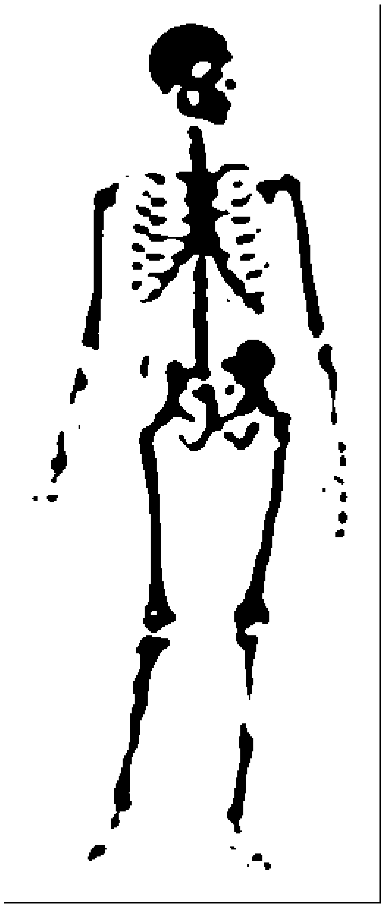
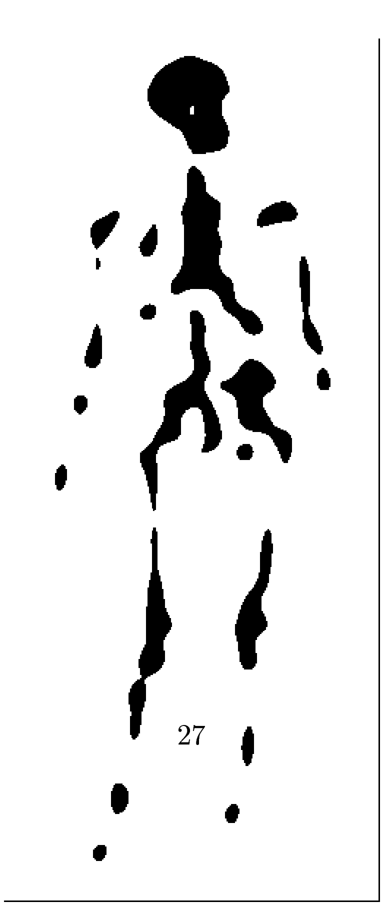
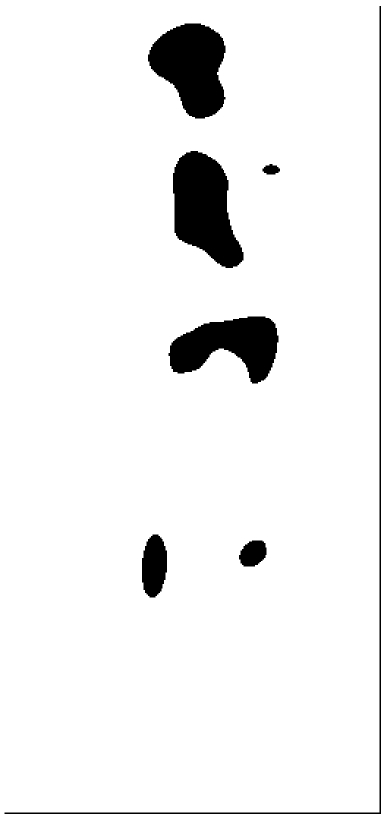


Figure 11: Cinema camera multiresolution

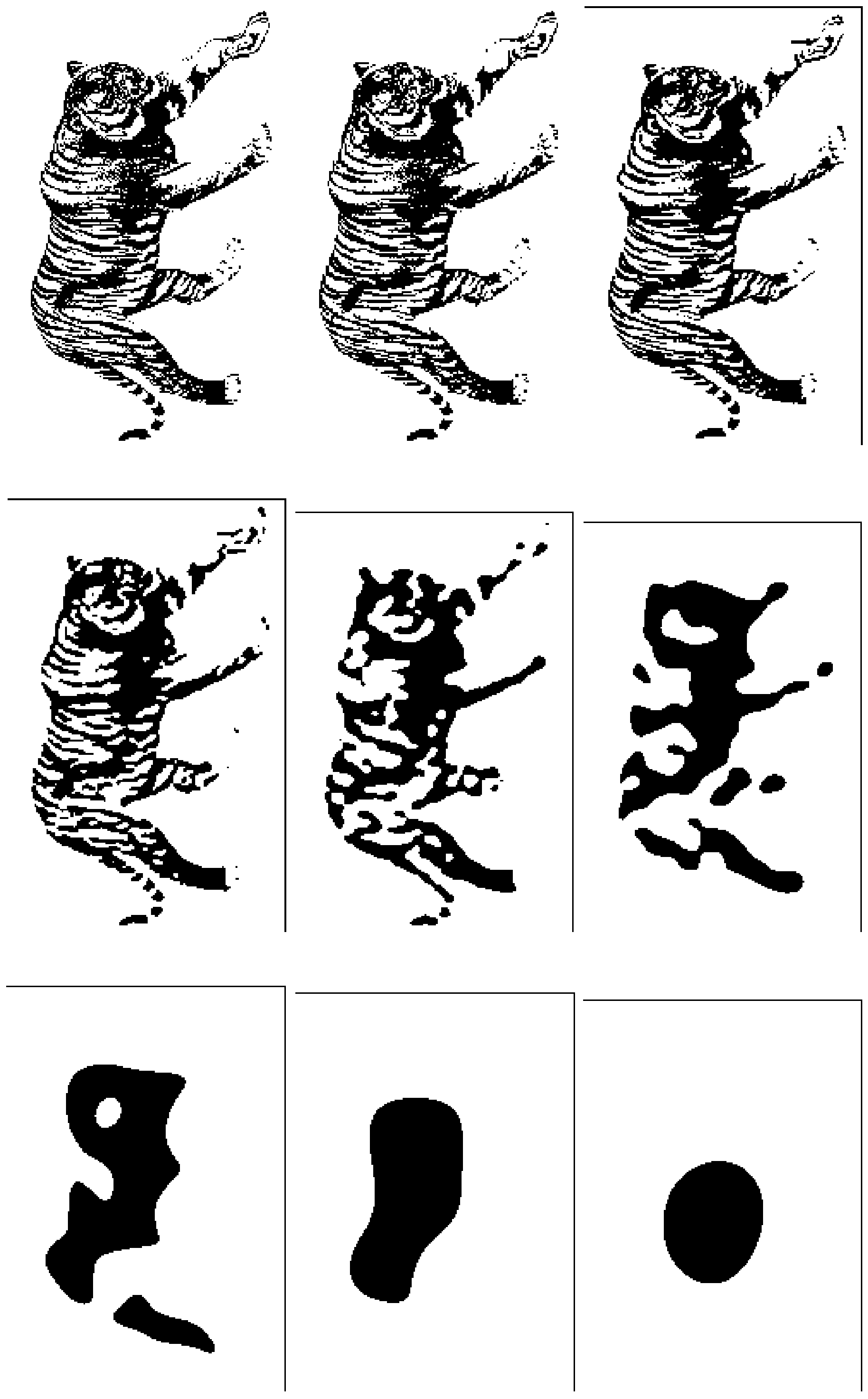
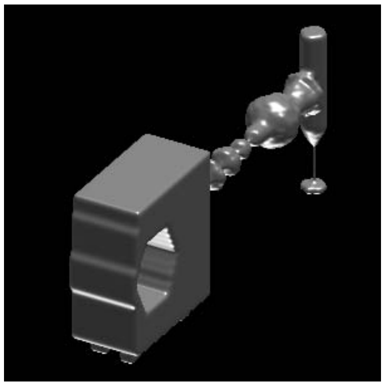
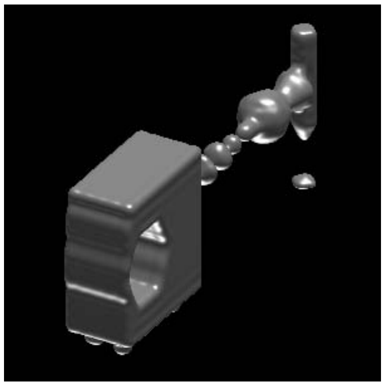
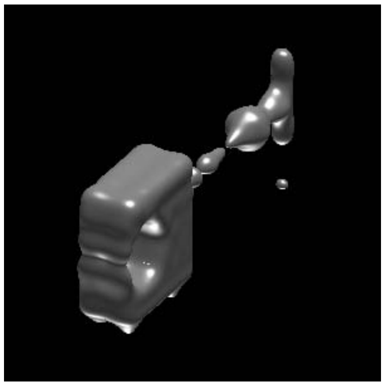
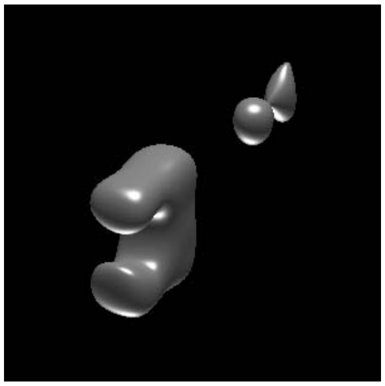


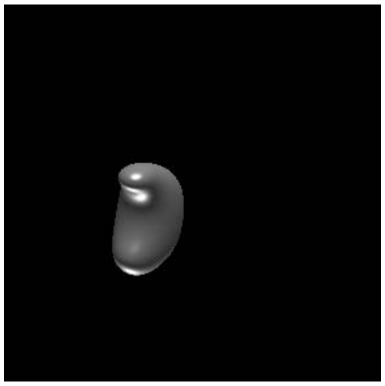
Figure 13: Tiger multiresolution


(a) Level 1


(b) Level 2

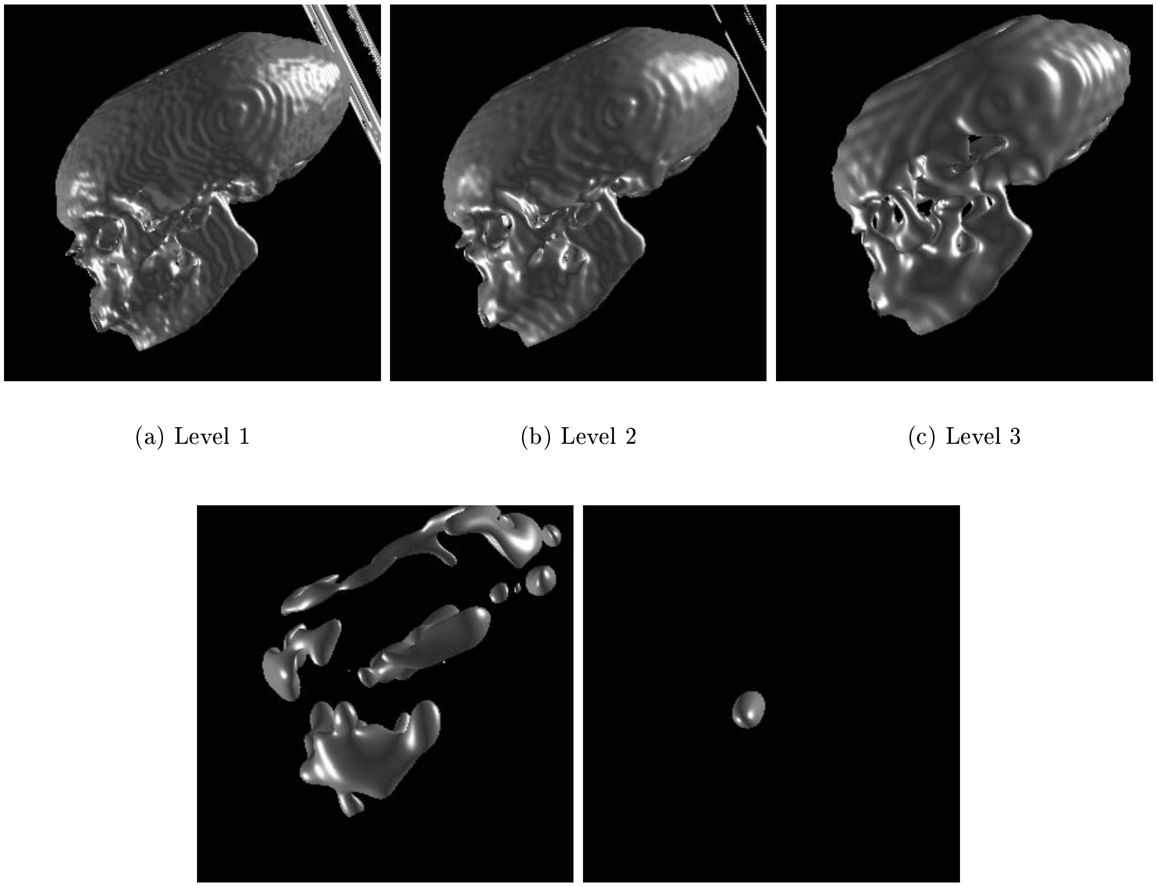

(c) Level 3


(d) Level 4


(e) Level 5

(f) Level 6

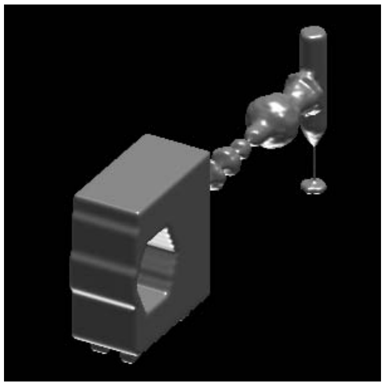
Figure 14: Mechanical part multiresolution



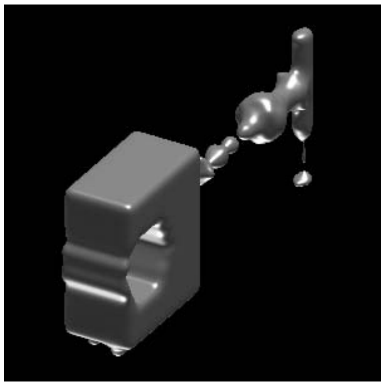
(a) Level 1 (b) Level 2 (c) Level 3

(d) Level 4 (e) Level 5

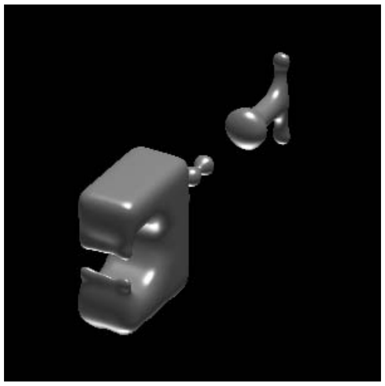
Figure 15: Skull multiresolution



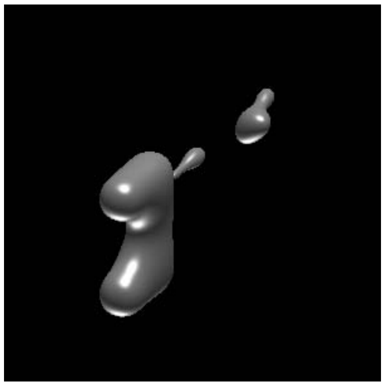
(a) Level 1



(b) Level 2



(c) Level 3



(d) Level 4

Figure 16: Mechanical part multiresolution pruning the octree levels