Computational Aesthetics in Graphics, Visualization, and Imaging (2010)
O. Deussen and P. Jepp (Editors)

Discretization of 3D models using voxel elements of different shapes

Davide Cavagnino and Marco Gribaudo

Computer Science Department, University of Turin, Italy

Abstract

This paper presents a voxelization algorithm that uses elements of various shapes that are not larger than a single voxel. Elements are chosen from a palette in order to minimize the deviation from the actual object according to a given metric. An analytical framework is presented and then an implementation based on a statistical sampling on the object is proposed. The result obtained is used to guide the recreation of the original model using small components that resemble the palette elements but emphasize the artistic nature of the reconstruction. Several examples where the proposed technique has been used to create some models with artistic and advertising purposes are presented to show the effectiveness of the technique.

1. Introduction

Voxelization is a process with which a solid object is approximated by a set of elements over a regular grid and can be considered the 3D counterpart of pixelization.

Usually voxels are considered of cubic shape and voxelization associates a value 0 or 1 to each voxel. This value represents whether that portion of the space is occupied by the object. A description of voxelization can be found in [CK95] and references therein. In [Kau87] are presented 3D scan conversion algorithms that transform an object into contiguous cubic voxels placed on a regular grid. The discussion is focused on objects described by means of parameters. The paper [BSC00] compares two techniques for voxelization and states some morphological conditions that an object must satisfy to be voxelized without exceeding a specific error. A Voxelization model, called V-model, is introduced in [SK99] with the aim of creating an intermediate representation of an object to obtain an alias-free voxelization.

Voxelization is used in both computer graphics and physical, geometrical and medical analysis to describe objects characterized by complex surfaces. For example, in [KCD*04] voxelization is applied to microcomputed tomography studying the effect of the error induced by the digitalization process. Earthquake simulation with the finite-element technique over a voxel mesh is proposed in [KFI04]. A more artistic application is presented in [Lam] that uses this technique to automatically build LEGO® models of an object.

In this paper we extend the conventional voxeling algorithm to support elements with shapes different from the cubical one. Each element is bounded by the conventional voxelization grid. The aim of the proposed extension is to improve the characterization of the original object by diminishing the approximation error without increasing the memory occupation.

We differentiate from the approach proposed in [QL96] based on the extended cuberille model [QD92], and from the marching cubes algorithm [LC87] because we focus on the construction of the approximated object rather than on the surface representation for visualization. We also allow voxels of generic shapes and we are not limited to having elements with only one external face.

The problem will be first considered analytically by proposing a matching function whose maximization allows the choice of the best element. Different definitions of the matching function can be used to achieve different voxelization goals such as computing the best approximation entirely contained inside an original model or the approximation that minimizes the mean difference with the object. For specific metrics and object encodings we propose an optimized implementation of the proposed algorithm.

The aim of the procedure proposed in this paper is not an accurate reconstruction of the object but an artistic vi-

© The Eurographics Association 2010.

DOI: 10.2312/COMPAESTH/COMPAESTH10/091-098

delivered by EUROGRAPHICS DIGITAL LIBRARY www.eg.org diglib.eg.org

sion of the parts that compose the original shape. Therefore the voxelized object is not directly represented using the elements of the palette but the information computed during the voxelization is used to appropriately choose other (small) objects that resemble the palette elements.

This approach prevents the application of standard voxel visualization techniques like marching cubes. For the rendering of the final model we developed a simple Blender script that reconstructs the original object using *geometry instancing*.

Section 2 defines the basic notation and the analytical specification of the framework. The main algorithm and implementation is introduced in Section 3. In Section 4 some results obtained with the presented algorithm are shown. Finally, some conclusions are drawn in Section 5.

2. Generalized voxelization

In this work we propose an algorithm to approximate a three-dimensional polygonal object using elements of different shapes. The use of voxels of different shapes allows some artistic exploitations that are not possible with rectangular prism voxelization.

We start with a classical voxelization problem that considers an object O in three dimensional space and tries to reconstruct it using a set of equal rectangular prism volumes. Those volumes are referred as *volume elements*, or *voxels*.

To perform the reconstruction, first the space is partitioned into a set of voxels and then the intersection between each voxel V and the object is considered [KCY93].

In the classical voxelization algorithm the procedure determines whether a given voxel should be either completely empty or completely full.

If a voxel is completely contained in the object ($V \cap O = V$), or has an empty intersection ($V \cap O = \emptyset$), then the choice is trivial. However, if there is only a partial intersection between the voxel and the object ($V \neq V \cap O \neq \emptyset$), the algorithm requires a technique, possibly based on some metric, to determine the most suitable choice.

The idea developed in this work is to enlarge the set of possible choices when partial intersections are considered. In particular, instead of having only the full (V) or empty ($\emptyset$) voxel, we consider a set of possible elements $P = \{E_0 \dots E_n\}$, $n \geq 1$, with $E_0 = \emptyset$ and $E_1 = V$. We call P the *Palette*. The algorithm then chooses the palette element that more closely resembles the intersection ($O \cap V \approx E_v$) within the considered voxel, where *more closely* is intended as the element that maximizes a similarity function.

This similarity function is based on some metric that measures the difference between the object and the approximating element. The metric will be considered in Section 2.2.

In our approach, we assume that each element E_v belonging to the palette P is completely contained in the voxel, that is $E_v \cap V = E_v$. Examples of this kind of elements are shown in Figure 1 (with the voxel mesh depicted). In [SR00] the properties of sets of intersecting convex voxels are exhaustively analyzed; conversely, in this paper, elements with any shape are considered with the only constraint to be contained in a volume of the grid. In Figure 2 we show an example of an element not considered in this work.

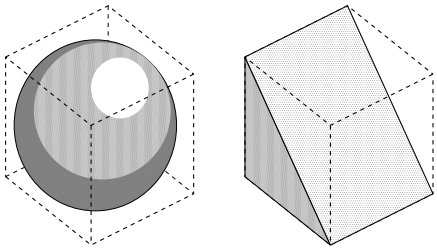

Figure 1: *Elements that are acceptable for our algorithm since contained within the voxel defined by dashed lines.*

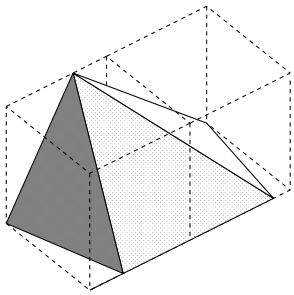

Figure 2: *Example of an element that is not acceptable for our algorithm because it extends beyond a single voxel.*

An example of a palette of elements could be the one that includes standard polyhedrons with rectangular and diagonal faces such as the ones depicted in Figure 3. Beside the basic elements, the palette is usually extended to take into account the main transformations that can be obtained by mirroring and swapping the principal axes. For example, the palette in Figure 4 is composed by the main transformations of the four original elements.

2.1. Notation

In the following we call $o(x,y,z)$ the *object description function*; in particular, for each point of coordinates (x,y,z), we have $o(x,y,z) = 0$ if the point does not belong to the object and $o(x,y,z) = 1$ if the point is part of the object. In the same way, for each element $E_v \in P$ we define the *element description function* $e_v(x,y,z)$ such that $e_v(x,y,z) = 0$ if the point (x,y,z) does not belong to E_v, and $e_v(x,y,z) = 1$ if the point is part of it.

© The Eurographics Association 2010.

Figure 3: *The basic elements making up the palette.*

Figure 4: *A global view of the* 46 *element palette we consider in the paper.*

We assume that every voxel is a box of size x_l, y_l, z_l having center coordinates x_c, y_c, z_c (see Figure 5). Each voxel is identified by three integer numbers (i, j, k) expressing its position in the 3D space. Thus the points (x, y, z) contained in voxel (i, j, k) are such that $i \cdot x_l \leq x < (i+1) \cdot x_l$, $j \cdot y_l \leq y < (j+1) \cdot y_l$, $k \cdot z_l \leq z < (k+1) \cdot z_l$.

The "element-in-voxel" constraint depicted in Figures 1 and 2 may be expressed as $e_v(x,y,z) = 0 \ \ \forall\ (x < 0) \vee (x \geq x_l) \vee (y < 0) \vee (y \geq y_l) \vee (z < 0) \vee (z \geq z_l)$. We call $\Omega_V = x_l y_l z_l$ the volume of the voxel and Ω_O the volume of the object defined as

$$\Omega_O = \iiint o(x,y,z)\, dx\, dy\, dz \tag{1}$$

To simplify the notation in the following we will restrict our attention to the voxel with $(i, j, k) = (0, 0, 0)$. Other voxels can be considered defining $o_{i,j,k}(x,y,z) = o(x - ix_l, y - jy_l, z - kz_l)$ and using it instead of $o(\cdot)$.

2.2. The similarity measure

We use a measure B_v to compute the degree of similarity of an element E_v to the part of the object contained in a voxel. This measure uses a similarity metric

$$g : \{0,1\}^2 \times R^3 \rightarrow R$$

that computes the difference between the point (x, y, z) of the object and the corresponding point of the element. In particular B_v can be defined as:

$$B_v = \frac{\int_0^{z_l}\int_0^{y_l}\int_0^{x_l} g(e_v(x,y,z), o(x,y,z), x, y, z)\, dx\, dy\, dz}{G} \tag{2}$$

where G is a normalization constant such that

$$G = \int_0^{z_l}\int_0^{y_l}\int_0^{x_l} \underset{(\varepsilon,\omega)\in\{0,1\}^2}{\arg\max} (g(\varepsilon, \omega, x, y, z))\, dxdydz$$

Note that G depends only on the similarity metric $g(\cdot)$.

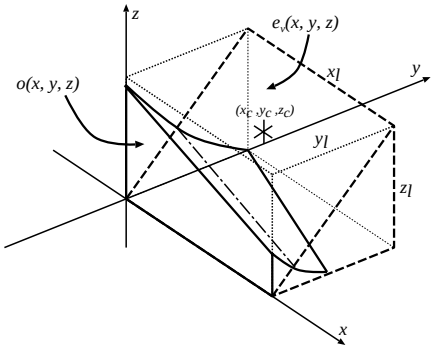

Figure 5: *Functions and coordinates.*

The dependency of g on x, y, z allows to consider anisotropic matches: that is, the total matching depends also on the position of the points in the volume considered. Moreover, due to the definition of the normalization constant G we have that $B_v \leq 1$.

We have considered the three possible metrics *Nearest Approximation*, *Best Outer Box* and *Best Inner Box* described in the following.

The *Nearest Approximation* metric chooses the element that has the least mean difference between itself and the object; it is defined as:

$$g_N(e, o, x, y, z) = 1 - (e - o)^2 \tag{3}$$

The squared difference between e and o is such that it is 0 when both the element and the object have the same density (both empty or both full); instead it is 1 when one is empty and the other is full.

The *Best Outer Box* metric, computes the smallest approximation that completely contains the polygonal object. It is defined as

© The Eurographics Association 2010.

$$g_O(e,o,x,y,z) = 2-(e-o)^2 - \frac{1}{\text{step}(e-o)}$$

where $\text{step}(x)$ is the indicator function ($\text{step}(x)=0$ if $x<0$, $\text{step}(x)=1$ otherwise). Note that this function adds to the *Nearest Approximation* (Equation 3) metric a term that tends to $-\infty$ when the object is full and the element is empty in the corresponding point. This ensures that elements having a point with this characteristic will not be considered.

The *Best Inner Box* metric computes the largest approximation that is contained in the object. It is defined as:

$$g_I(e,o,x,y,z) = 2-(e-o)^2 - \frac{1}{\text{step}(o-e)}$$

This definition is analogous to the one of the *Best Outer Box* metric.

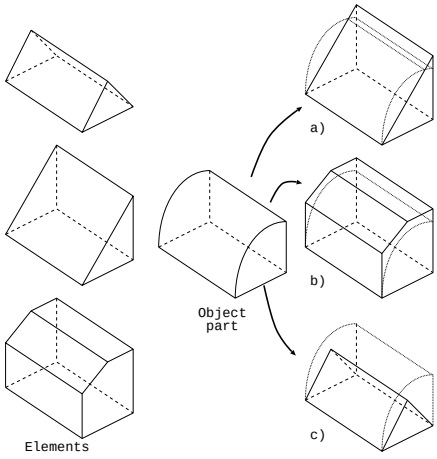

Figure 6: *Matching examples of the object in the centre using the palette shown in the left column. The right column shows the results using: a)* Nearest Approximation, *b)* Best Outer Box, *c)* Best Inner Box.

A visual representation of the choices made by the *Nearest Approximation*, *Best Outer Box* and *Best Inner Box* is represented in Figure 6. In Figure 7 the three different types of approximation *Best Inner Box*, *Nearest* and *Best Outer Box* are shown applied to a model of the head of a cow. This figure is presented for subjective visual quality comparison purposes.

Table 1 summarizes the values for the three matrics for all the $e(\cdot)$ and $o(\cdot)$ combinations.

3. Voxelization algorithm

In the following we will present an algorithm that performs an optimized version of the approach considering only objects with closed two-manifold surfaces (i.e. objects that do not have laminar faces).

Table 1: *Match metrics w.r.t. the position of the sample point.*

		Object	
	Element	$o(\cdot)=1$	$o(\cdot)=0$
Nearest Approx.	$e_v(\cdot)=1$	1	0
	$e_v(\cdot)=0$	0	1
Outer Box	$e_v(\cdot)=1$	1	0
	$e_v(\cdot)=0$	$-\infty$	1
Inner Box	$e_v(\cdot)=1$	1	$-\infty$
	$e_v(\cdot)=0$	0	1

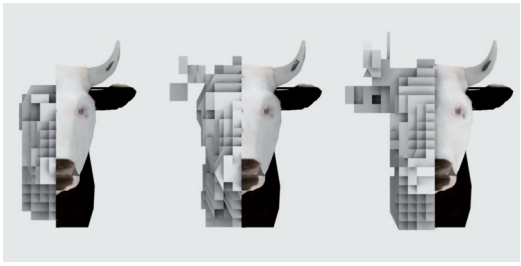

Figure 7: *Front view of the model of a cow as reproduced by Best Inner Box, Nearest Approximation and Best Outer Box.*

We base our algorithm on the voxelization method presented in [TGR04]. In particular, we use a Monte Carlo method to compute the similarity measure in (2). In this case we choose

$$\hat{B_v} = \frac{1}{|U|}\sum_{(x,y,z)\in U} g(e_v(x,y,z),o(x,y,z),x,y,z) \approx G\cdot B_v \quad (4)$$

where $U \subset [0,x_l]\times[0,y_l]\times[0,z_l]$ is a finite set of random points inside the voxel space V. Indeed $\lim_{|U|\to\infty}\hat{B_v} = G\cdot B_v$.

The construction of U requires the generation of $3|U|$ random numbers. Moreover the evaluation of $g(\cdot)$ requires the computation of $e_v(x,y,z)$ and $o(x,y,z)$ for each point. To optimize the computation of Equation (4) we follow the approach proposed in [TGR04]. Voxels (i,j,k) are considered in columns, that is we fix i and j, and vary only k. For each column we cast a set U_r of t randomly positioned *rays*. Each ray is parallel to the direction of the voxel's column and it is identified by its x and y coordinates: $(x,y)\in U_r \subset [0,x_l]\times[0,y_l]$. Then, the intersections of the ray with the object are computed and collected into a set $\{z_0,\ldots z_{2n-1}\}$ with $z_m < z_{m+1}$. Note that the number of intersections is always even thanks to the **two-manifold closed surface** assumption. In this way, $o(x,y,z)$ can be eas-

© The Eurographics Association 2010.

ily determined by checking the position of z in the ordered list: if $z_{2m} < z < z_{2m+1}, o(x,y,z) = 1$, otherwise $o(x,y,z) = 0$.

We then consider each voxel in the column. We call *segment* the intersection of each ray with the voxel under consideration. We identify p randomly positioned points of each segment (*sample points*) and we use them to compute the set U used to approximate B_v. In this way $|U| = t \times p$, where t groups of p points share the same (x,y) coordinates. Figure 8 shows a visual representation of the sampling process.

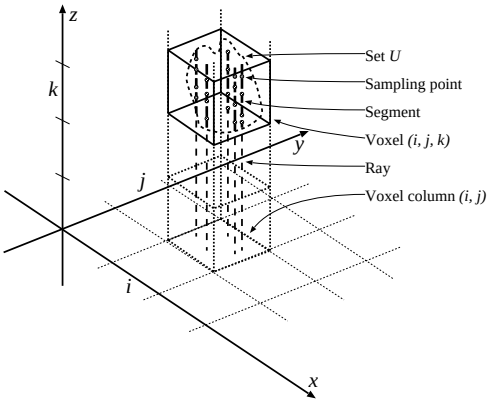

Figure 8: *Representation of the sampling process in a voxel.*

The process of approximation of B_v is repeated for each element $E_v \in P$. The algorithm chooses the element E_w which maximizes $\hat{B_v}$. That is, $B_w = \arg\max_{E_v \in P} \hat{B_v}$. In this case, the complexity of the whole voxelization process is $O(I\ J\ K\ p\ t\ |P|)$ where I, J, K represent the total number of voxels considered in each direction.

3.1. Optimization

The complexity could be greatly reduced by treating the empty and full voxels (which are the most likely in common cases) in a special way. We add two tests to determine if we are in one of such cases that are executed before considering the remaining pieces of the palette.

The first check verifies that the segments obtained from the intersection of the rays with the voxel are all completely outside (or all completely inside) the object. Let's call z_l and z_u the two end points of each segment (i.e. the intersections of the ray with the upper and lower faces of the voxel). If, for all segments $z_{2m} < z_l < z_u < z_{2m+1}$, then the voxel is full. Otherwise, if for all segments $z_{2m+1} < z_l < z_u < z_{2m+2}$ then the voxel is empty.

The second optimization, which can be used only with the *Nearest Approximation* metric, does not compute the g function, but determines if the voxel should be E_0 or E_1 based on a statistic of the generated points. If we call $\#o_1$ the number of points (x,y,z) where $o(x,y,z) = 1$ then we use the empty voxel if $\#o_1 < \alpha|U|$ or the full voxel if $\#o_1 > \beta|U|$. Here α and β are two constants with $0 < \alpha << p_{min}$ and $p_{max} << \beta < 1$, and p_{min} (p_{max}) is the fraction of the voxel occupied by the element with the smallest (largest) volume in the set $P - \{E_0, E_1\}$. The effect of these optimizations will be more meaningful as the number of approximating elements used increases, since the tests on the remaining elements in P will be skipped.

Several other optimizations have been implemented to enhance the computation of $o(\cdot)$, $e_v(\cdot)$ and $g(\cdot)$.

As introduced before, the evaluation of $o(\cdot)$ is replaced by the computation of the intersection list. When the surface of the object is described by a closed triangle mesh, the quad-tree based search technique presented in [TGR04] can be used. The computation of the list requires the search for possible intersections between the ray and each triangle of the surface. The quad-tree structure limits the search to only the triangles whose projection intersects the leaf of the quad-tree that contains the ray. Note that in [TGR04] rays were cast to perform anti-aliasing oversampling whereas in our work they are used to compute the similarity measure with the Monte Carlo technique.

In principle, the computation of $e_v(\cdot)$ may be optimized using the same technique previously mentioned for $o(\cdot)$. However, in most cases, the geometry of every E_v is quite simple. For example, most of the useful elements can be defined by a convex polyhedron, or the union of convex polyhedrons, or the complement of a convex polyhedron. In this case the membership of a point to the element may be evaluated using only a combination of a few linear expressions. In the example provided in this paper two linear expressions were enough to describe most of the elements, the most complex ones being defined by four linear constraints.

The *Nearest Approximation* can be improved by incrementing the similarity measure when $e_v(x,y,z) = o(x,y,z)$ for the considered point (x,y,z). This test corresponds to adding to the similarity counter the result of the logic operation $e_v(x,y,z)$ NEX-OR $o(x,y,z)$ (equivalence).

The *Best Outer Box* (*Best Inner Box* respectively) performs like the optimized version of the *Nearest Approximation*. However if $e_v(x,y,z) = 0 \wedge o(x,y,z) = 1$ ($e_v(x,y,z) = 1 \wedge o(x,y,z) = 0$) the similarity value is set to $-\infty$ and the check is stopped since other points will not be able to affect it. Note that even imposing this condition does not create any problems of not being able to find a good approximating element, because at least the full (empty, respectively) element will have a finite and positive value for the counter (even if very small).

In Figures 9 and 10 we show the results of the voxelization algorithm on a sphere varying both the size of the voxel (Figure 9) and the number of approximating elements (Figure 10) using the *Nearest Approximation* metric. As can be seen the visual quality of the results improves as either the

© The Eurographics Association 2010.

palette enlarges or the voxel size decreases. The 78 element palette considered in Figure 10 is composed by the elements belonging to the 46 element palette plus their complements.

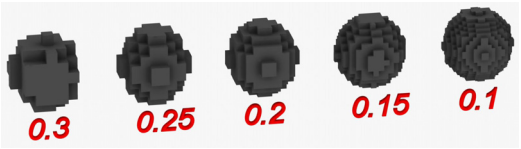

Figure 9: *Voxelization of a radius 1 sphere with a 2 element palette (empty and full voxels) and different indicated voxel sizes.*

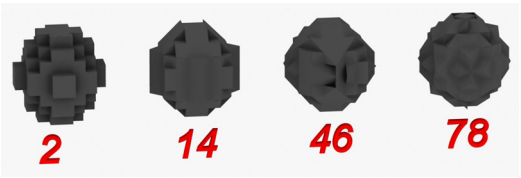

Figure 10: *Voxelization of a radius 1 sphere with 0.25 voxel size and different indicated numbers of approximating elements in the palette.*

4. Application

The methodologies and algorithms presented in the previous sections were used to perform voxelizations of objects using palettes with different numbers of pieces.

After the algorithm produced a voxelization, we postprocessed the data to obtain an artistic representation based on the same voxelization data. Namely, we associated with every element in the original palette a set of mini-objects that were (possibly) real objects whose shape, in some way, could recall the corresponding element in the palette. In other words, we associated with every element E_v an ordered list of objects whose shape could approximate E_v:

$$E_v \rightarrow \{R_{v,1}, R_{v,2}, \ldots R_{v,T_v}\}$$

Then, we used the similarity measure B_v (defined in Equation 2) to choose which object in the associated list to select to generate the rendering.

So we assigned to every $R_{v,j}$ a *threshold* $D_{v,j}$, $D_{v,j} < D_{v,j-1}$ (with $D_{v,0} = 1$, $D_{v,T_v} = 0$) and rendered E_v with $R_{v,j}$ if $D_{v,j} \leq B_v < D_{v,j-1}$.

Note that the choice of using the similarity measure to select the graphical object allows a consistent and deterministic output (other choices like random object selection would not have this property).

For example, we could associate with the pyramidal element E_{pyr} a list composed of two elements ($T_{pyr} = 2$): a small Egyptian pyramid ($R_{pyr,1}$) and an ice cream cone shape ($R_{pyr,2}$). Then we can set $D_{pyr,0} = 1$, $D_{pyr,1} = 0.6$ and $D_{pyr,2} = 0$. In this way the Egyptian pyramid will be used if the similarity measure is in the range $0.6 - 1$, and the ice cream cone otherwise. Note that, in principle, we could have used the Egyptian pyramid and ice cream model directly as palette elements. This, however, would have increased the complexity of computing the similarity measure without achieving sensible improvements.

In the following we present three possible applications of the voxelization technique for artistic and advertisement purposes.

In the first example we create a sample advertisement text, whose characters are composed by small letters. Each letter is painted according to a texture that represents a different national flag. The text that is voxelized is the word “Europa”. We used a 10 element palette where we associated with each of the elements one or more letters or numbers as reported in Figure 11. The size of the voxel was $x_l = y_l = 1$, $z_l = 0.4$. In Figure 12 the original model is compared with the corresponding voxelized version. The render presented in Figure 13 proposes a view from above the surface of the model. This image could be a frame of an animation that “walks” over the writing to show how the different letters, representing different nations, work together to create a greater entity: the text “Europa” symbolizing the cooperation among the people.

Figure 11: *The* 10 *element palette (the empty element is not shown) in the back row. In each column are presented the real elements associated with the element at the top of the column.*

In the second application of our method, we wanted to express the idea of a world made of different objects. We employed the 46 element palette as shown in Figure 4 where the four basic elements (the ones depicted in Figure 3) have been associated with several common-use objects, as shown in Figure 14. Here the size of the voxel was $x_l = y_l = z_l = 1$. The peculiarity of this experiment is the use of a single spherical mapped earth texture projected over all the voxels. A comparison of the overall vision of the Earth along with

© The Eurographics Association 2010.

Figure 12: *Comparison of the original model and the voxelized model using the real elements in Figure 11.*

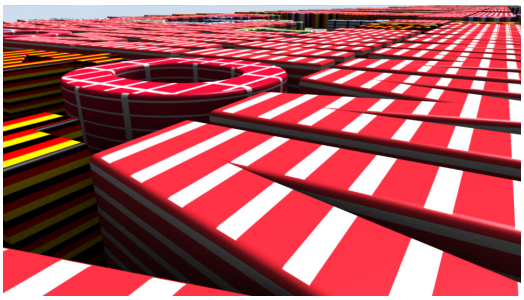

Figure 13: *View from the surface of the rendered Europa model.*

its voxelized version is presented in Figure 15. A view of the voxelized planet of Figure 15 from the surface is shown in Figure 16.

Figure 14: *Four basic elements (rightmost column) and to the left of each element the real objects used for the rendering.*

In the third experiment, we mixed the "Leviathan" by T. Hobbes with an artistic expression as the dance. We used the same 10 element palette employed in the first experiment but we represented it with two dancers mimicing the corresponding shapes. Figure 17 and Figure 18 show the different elements not taking into account horizontal symmetries. Figure 19 compares the original model (a ballerina moving

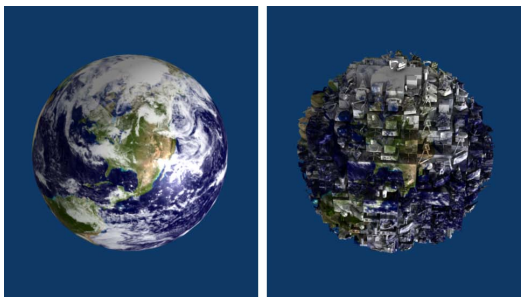

Figure 15: *Comparison of the Earth and a voxelized sphere with a* 46 *element palette, rendered with the objects in Figure 14 and successively textured.*

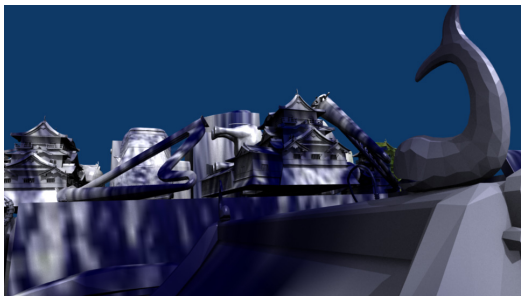

Figure 16: *View of the voxelized sphere of Figure 15 from the surface.*

to the "arabesque" position) with the voxelized version. Figure 20 shows the view which can be experienced by a dancer taking part to the performance.

Figure 17: *First part of the palette elements and the real objects used to approximate them.*

5. Conclusions

In this paper we have presented an analytic voxelization framework capable of producing different approximations of an original object for artistic and advertising purposes. The approximation is composed of elements chosen from a finite palette. Elements are selected through the maximization of

© The Eurographics Association 2010.

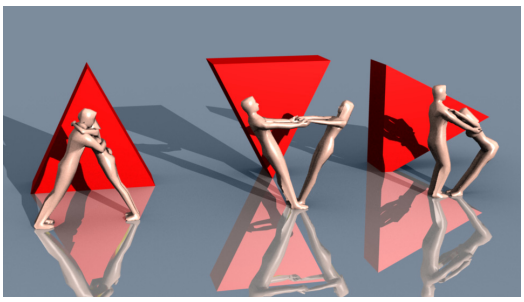

Figure 18: *Second part of the palette elements and the real objects used to approximate them.*

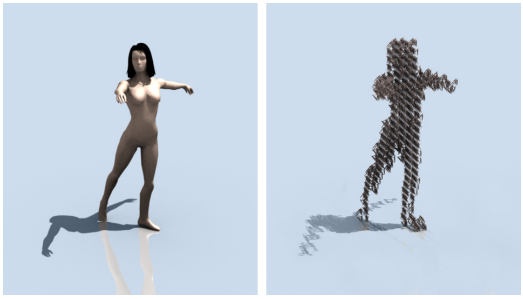

Figure 19: *Original model and its voxelization rendered with the* 10 *element palette shown in Figure 17 and Figure 18.*

a matching measure. To simplify the search procedure we differentiate a rendering palette from the one used during voxelization.

The validity of the approach is demonstrated by providing three simple possible applicative examples.

We are planning to extend the proposed algorithm to consider elements larger than a voxel and to apply smoothing techniques to produce less accurate but smoother and visually appealing surfaces.

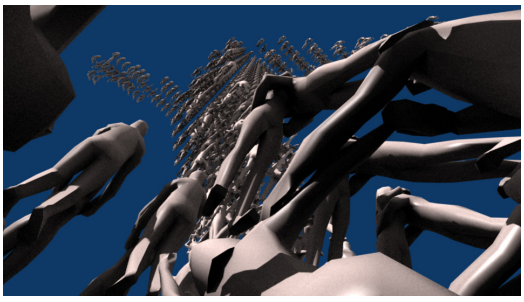

Figure 20: *View from the inside of the dancer model.*

Acknowledgements

The authors wish to thank Professor Albert E. Werbrouck for his help in the preparation of this paper.

References

[BSC00] BÆRENTZEN J. A., SRAMEK M., CHRISTENSEN N. J.: A morphological approach to the voxelization of solids. In *The 8-th International Conference in Central Europe on Computer Graphics, Visualization and Digital Interactive Media* (Feb. 2000), pp. 44–51.

[CK95] COHEN D., KAUFMAN A.: Fundamentals of surface voxelization. *Graphical Models and Image Processing 57*, 6 (1995), 453–461.

[Kau87] KAUFMAN A.: Efficient algorithms for 3d scan-conversion of parametric curves, surfaces, and volumes. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1987), ACM, pp. 171–179.

[KCD*04] KIM D.-G., CHRISTOPHERSON G. T., DONG X. N., FYHRIE D. P., YENI Y. N.: The effect of microcomputed tomography scanning and reconstruction voxel size on the accuracy of stereological measurements in human cancellous bone. *Bone 35*, 6 (2004), 1375–1382.

[KCY93] KAUFMAN A., COHEN D., YAGEL R.: Volume graphics. *Computer 26*, 7 (1993), 51–64.

[KFI04] KOKETSU K., FUJIWARA H., IKEGAMI Y.: Finite-element simulation of seismic ground motion with a voxel mesh. *Journal of Pure and Applied Geophysics 161*, 11-12 (2004), 2183–2198.

[Lam] LAMBRECHT B.: Voxelization of boundary representations using oriented LEGO® plates. http://lego.bldesign.org/LSculpt/lambrecht_legovoxels.pdf Downloaded on 13th August 2008.

[LC87] LORENSEN W. E., CLINE H. E.: Marching cubes: A high resolution 3D surface construction algorithm. *SIGGRAPH Computer Graphics 21*, 4 (1987), 163–169.

[QD92] QU X., DAVIS W. A.: An extended cuberille model for identification and display of 3D objects from 3D gray value data. In *Proceedings of the conference on Graphics interface '92* (San Francisco, CA, USA, 1992), Morgan Kaufmann Publishers Inc., pp. 70–77.

[QL96] QU X., LI X.: A 3D surface tracking algorithm. *Computer Vision and Image Understanding 64*, 1 (1996), 147–156.

[SK99] SRAMEK M., KAUFMAN A. E.: Alias-free voxelization of geometric objects. *IEEE Transactions on Visualization and Computer Graphics 5*, 3 (1999), 251–267.

[SR00] SAHA P. K., ROSENFELD A.: The digital topology of sets of convex voxels. *Graphical Models and Image Processing 62*, 5 (2000), 343–352.

[TGR04] THON S., GESQUIÈRE G., RAFFIN R.: A low cost antialiased space filled voxelization of polygonal objects. In *International Conference Graphicon* (2004), pp. 71–78.

© The Eurographics Association 2010.