

# Computer Assisted Generation of Bas- and High-Reliefs

P. Cignoni, C. Montani, R. Scopigno

*IEI - CNR, Via S. Maria 46, I-56126 Pisa, Italy*

---

## Abstract

The paper proposes a computer-oriented interpretation of the laws ruling the elevation of figures in bas- or high-relief sculptures, and presents some simple algorithms and a prototypal, public-domain software tool for the generation of 3D bas- and high-reliefs starting from 3D surface models.

---

## 1 Introduction

Relief is the sculptural technique where the modeled forms stand out from the surrounding surface. Depending on the shape of the elevation of the modeled objects we can have **bas-relief** (low-relief or *basso-rilievo*, Figure 1), in which the projection from the background surface is slight and no part of the form is undercut, or **high-relief**, in which some of the forms stand out more and may be completely detached from the background.

In this paper we first analyze the geometric properties of the bas-relief technique and we discuss their relationship with the classical computer graphics perspective transformation (Section 2). Based on this relationship, in Section 3 we present simple techniques for the generation of 3D bas-reliefs, coloured bas-reliefs, and cameos starting from a surface-based 3D synthetic model. The generation of 3D high-reliefs is discussed in Section 4. Finally, we conclude in Section 5 giving a short description of our prototypal, public-domain implementation.

## 2 Relief as a distance function

For a better understanding of the technique of bas-relief, let us introduce the example of a real Renaissance masterpiece. Donatello's superb bas-relief



Fig. 1. Donatello (Donato di Niccolo di Betto Bardi), *Miracle of the Ass*, 1447 – 50, Bronze, 57 × 123cm, Basilica di Sant’Antonio, Padua, Italy

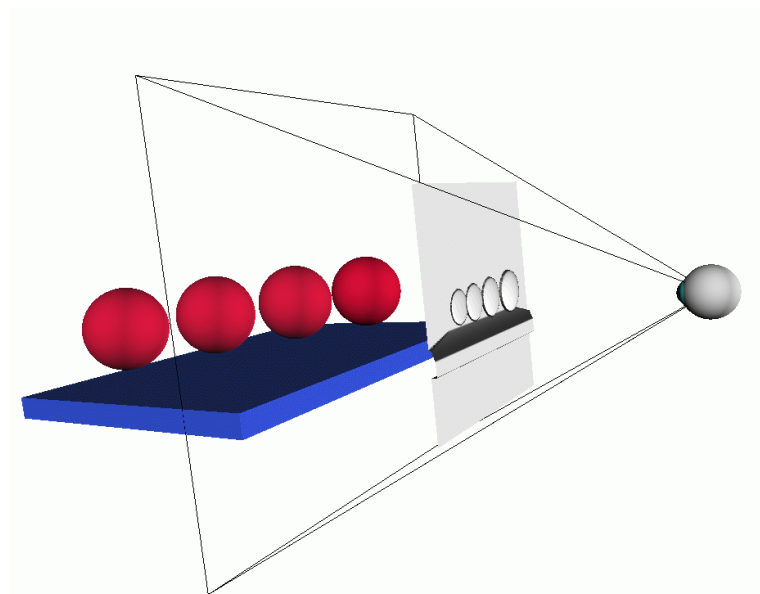


Fig. 2. Embossing the viewplane to create a bas-relief.

in Figure 1 is a good depiction of some of the basic features of this artistic medium:

- *modeling*: the whole scene is modeled following the laws of perspective (note the arches and walls);
- *elevation*: the nearest figures are the most elevated;
- *linearity*: the shape and curvature of the embossed scene still reflect the original shapes; the walls, floor and ceiling are still flat despite being deformed (there are no right-angle walls).

On the basis of the considerations above, given the perspective projection of a 3D scene onto a view plane, a bas-relief can be interpreted as an embossing

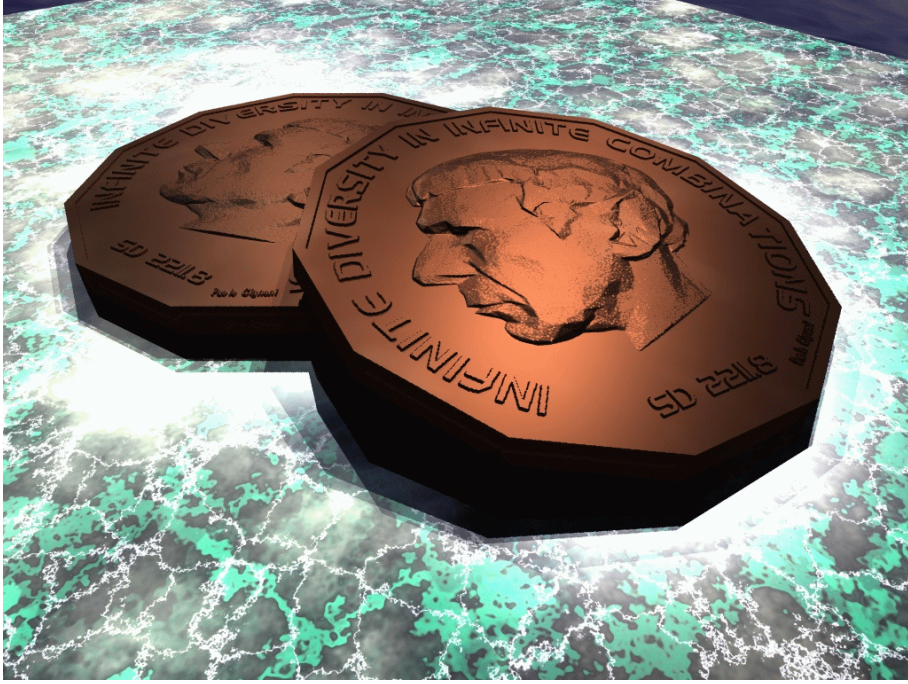


Fig. 3. Bas-reliefs for medals generated from range scanner data.

of the view plane in which the embossing height of each point depends on the distance between the observer and the projected point itself (see Figure 2). In other words, given a point  $p$  in the original 3D scene and the corresponding projected point  $p'$  on the viewplane, to generate a bas-relief we should raise  $p'$  by a quantity  $B(p)$  which depends on the distance  $d(v, p)$  between  $p$  and the viewpoint  $v$ . Recalling the main features of the bas-relief technique, we infer that the elevation function  $B(p)$ :

- *has to be decreasing* ( $\lim_{p \rightarrow \infty} B(p) = 0$ ): objects which are farther away have less relief than those that are nearer; the horizon is the background surface;
- *must preserve linearity*: planar objects in the 3D scene have to be planar in the bas-relief in order to preserve the correct shading of the relief's surfaces.

The standard perspective transformation shares many properties with the elevation function  $B(p)$  we are looking for: it preserves relative depth, straight lines, and planes, and, at the same time, performs the perspective foreshortening. It scales the  $z$ -coordinate of each point  $p$  in the viewing space by a function  $c * \frac{1}{p_z}$  which compresses the elements further away from the observer into a smaller  $Z$  range. In other words, the perspective transformation maps the normalized view volume in a space which is compatible with the bas-relief elevation function  $B(p)$  introduced above, and for this reason it can be efficiently used in the computer-assisted generation of bas-relief models.

### 3 Generating a bas-relief from a 3D model

Let's first assume that a bas-relief can be modeled as a height field, i.e. there are no pairs of points of the relief with the same  $x$  and  $y$  coordinates (i.e. with no detached components).

Based on the previous discussion and by adopting the taxonomy of the visible surface determination (VSD) methods [3], we propose two techniques for the generation of height fields representing the bas-reliefs of 3D scenes: an image precision and an object precision technique. We have implemented the image precision technique and used it for all the examples in this paper. We have not implemented the object precision algorithm.

#### 3.1 Image precision bas-reliefs

Current graphics hardware subsystems solve the visibility problem by adopting a  $z$ -buffer approach [1]. After the perspective transformation has been applied, each pixel of the  $z$ -buffer holds the  $z$ -coordinate of the corresponding pixel of the visible surface in the rendered image, that is the depth value computed by the perspective transformation. These  $z$ -values define a rectangular discrete height field (with a resolution equal to the resolution of the rendered image) that represents the bas-relief of the scene. The bas-relief in Figure 8 was obtained using this technique; the resolution of the height field was  $426 \times 564$ .

The proposed technique operates in a discrete space and therefore presents both the well known problems and benefits of image precision VSD algorithms: *aliasing* in the sampling process versus *simplicity* and *speed*.

The *quality* of the bas-relief (with respect to its shape) directly depends on the resolution of the height field, and therefore a high resolution  $z$ -buffer should be used to render the original 3D scene. However, depending on the rendering technique used, the visualization of large height fields can be quite slow (many graphics libraries manage height fields as large meshes of triangles).

To improve the efficiency of the rendering process, we can simplify the geometric complexity of the triangular mesh representing the height field by means of a surface simplification algorithm<sup>1</sup> [6,4,2]. In this case, the user should pay attention to the choice of the simplification criterion. An anisotropic error

---

<sup>1</sup> For some of the examples in this paper we used our surface decimator *Jade*, available in public domain at our web site <http://miles.cnuce.cnr.it/cg/enhadecimation.html>

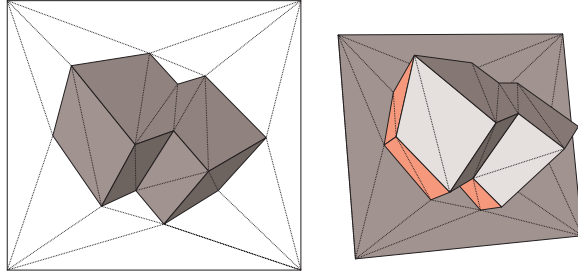


Fig. 4. Object precision algorithm: (left) 2D subdivision of the viewplane; (right) rotated view of the mesh, after associating  $z$  values.

metric has to be provided because the preservation of correct  $z$  values is critical. This can be simply obtained by scaling the  $z$ -coordinates of the bas-relief in a wider domain before the simplification process, and by scaling them back after simplification.

Alternatively, if supported by rendering software, we can directly visualize the generated height fields either as bump maps (for very low elevation bas-reliefs) or as displacement maps. The medals in Figure 3 (from a 3D model of the Spock's Head, courtesy of Cyberware) have been obtained by means of the height field feature of a public domain ray tracer (POV [5]).

The user can easily modify the geometric appearance of the relief by interactively selecting the height of the bas-relief (i.e. the way it embosses) or the background distance.

An alternative to the image precision technique might be to use an object precision algorithm. With respect to the previous approach, object precision reliefs present higher precision and do not need any surface simplification post-processing. On the other hand, they present a slightly more complex implementation. We sketch an object precision approach in the next section.

### 3.2 Object precision bas-reliefs

Object precision bas-reliefs can be generated by using an object precision VSD algorithm [7,9]. VSD algorithms generally return a set of 2D polygons covering only a part of the viewplane. In order to get comprehensive information about the viewplane we add to the starting scene a sufficiently large background polygon, parallel to the viewplane. The VSD algorithm thus returns a 2D subdivision of the viewplane into visible polygons. This subdivision is then triangulated in the postprocessing phase (Figure 4).

Each triangle of the subdivision is part of the projection of a polygon in the original scene. During the VSD process, it is possible to associate the cor-

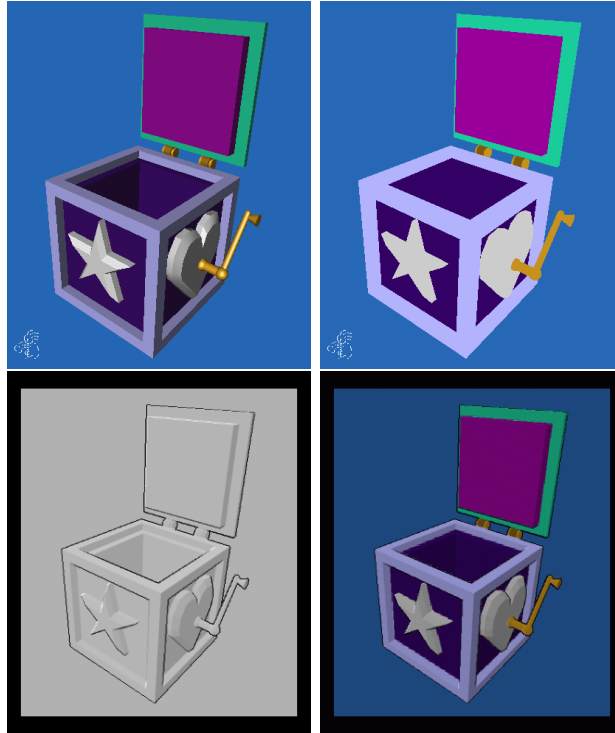


Fig. 5. Stitching color information to a bas-relief. The bas-relief (bottom-left) generated from the 3D model (top-left) can be colored (bottom-right) by simply applying to it the base (not shaded) colors (top-right) of the original rendered scene.

rect (perspective transformed and scaled)  $z$ -coordinate to the vertices of each visible polygon and, therefore, to the vertices of each triangle. To complete the bas-relief modeling we need only to create the faces parallel to the view direction which connect the disjoint triangular faces: for each edge in the 2D subdivision belonging to two triangles without matching  $z$ -coordinates (for example, the red triangles in Figure 4), we create a trapezoidal face to connect the disjoint triangles.

### 3.3 Coloring reliefs and cameos

From the rendering point of view, the image precision technique permits also to obtain colored reliefs and cameos in a very simple way. Coloring is done by applying to the bas-relief a texture which stores only the base colors of each component of the original 3D scene (without any lighting and shading computations). An example is given in Figure 5: the colored bas-relief (bottom-right) was obtained by applying to the original one (bottom-left) the base colors (top-right) of the original 3D scene (top-left). The texture is computed by rendering the scene with the same viewing parameters and without light shading.

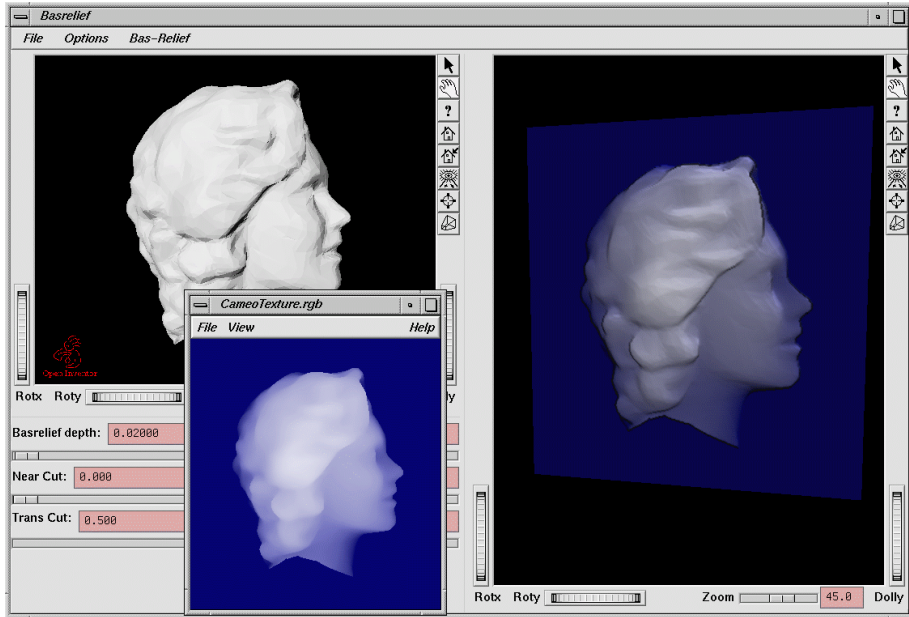


Fig. 6. Exploiting the depth information to build the texture (shown in the small window) for a cameo.

We can also use color textures to create cameos. Cameos are a particular kind of bas-relief. Because they are generally engraved on stones or shells, the embossed parts show different colors (generally white) from the background. Color depends on the layer structure of the engraving material. Moreover, because the color transition of the carving material is not immediate, and/or the material itself is not completely opaque, cameos present the property that the least elevated parts shade into the background color.

We simulate this property by creating a texture in which the bas-relief heights are mapped into different color shades, ranging from the background color to the foreground color (e.g. white).

Figure 6 (right-window) shows a cameo obtained with this technique. It is generated from a 3D model of Marilyn Monroe’s head (left-window, courtesy of 3D Scanners Corporation) using the texture shown in the small window. The figure also shows the interface of our prototype system for the generation of bas-reliefs. Figure 9 shows an image of the Marilyn Monroe’s cameo.

#### 4 High-relief modeling

A high-relief cannot be represented as a simple height field. Generally, however, part of it is still a plain bas-relief and only the nearest forms are full 3D objects. We propose here a technique for the computer-assisted construction of a subset of all the possible high-reliefs, i.e. the reliefs in which the forms can be divided

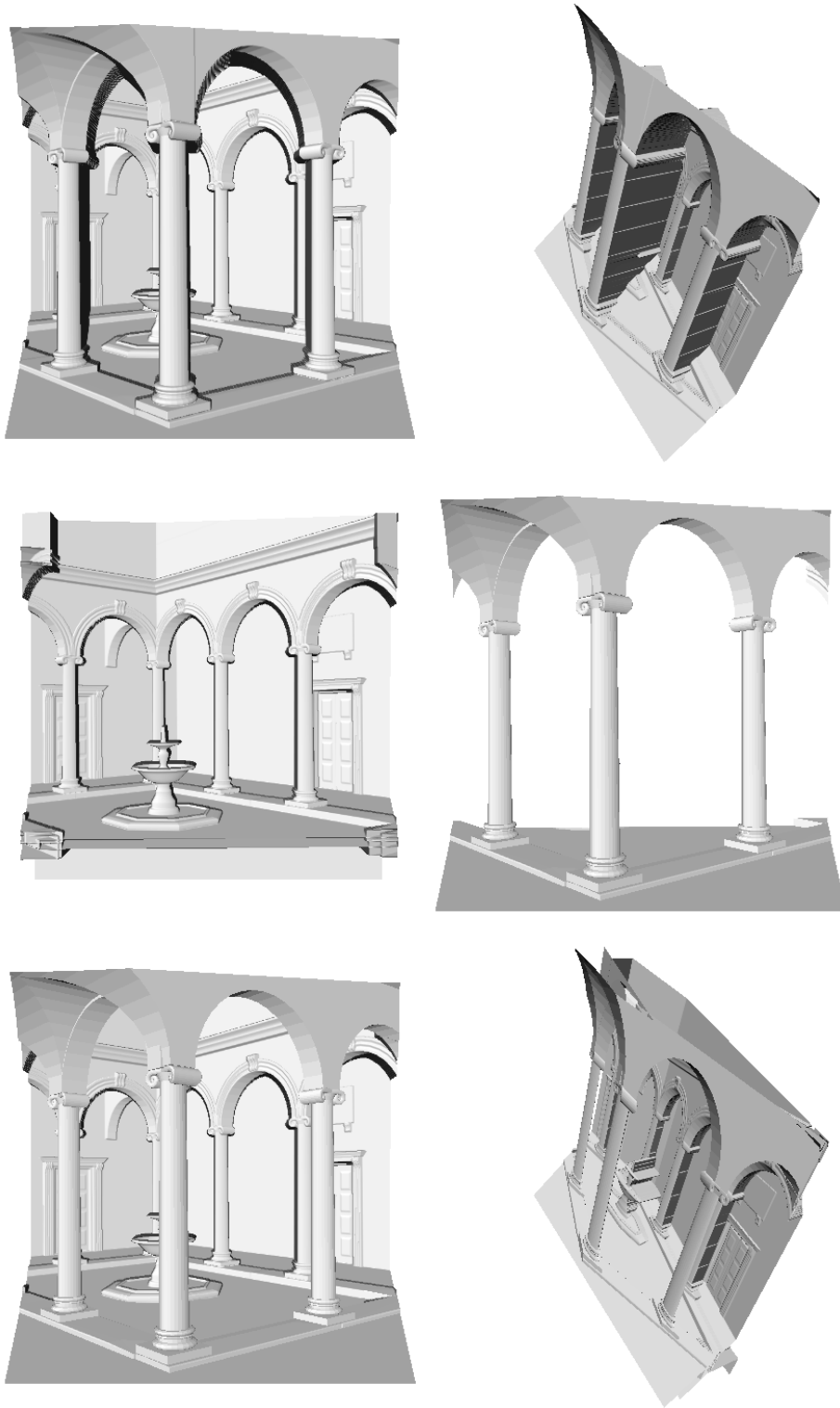


Fig. 7. Generating a high-relief. The bas-relief of the cloister (top-left) and a rotated view of it (top-right); the *bas* (middle-left) and the *high* (middle-right) parts; the high-relief of the cloister (bottom-left) and a rotated view of it (bottom-right).

into two parts by a splitting plane, such that the *high* portion is composed of the 3D protruding components, and the *bas* portion is a height field which represents the farthest objects.

Let us denote with  $z_t$  the clipping plane selected by the user between the *bas* and *high* portions of the relief, then we model the objects farther than  $z_t$  from the observer with a height field, while the objects nearer than  $z_t$  are represented as full 3D objects. The relationship outlined in Section 2 between the elevation function  $B$  and the perspective transformation allows the starting model to be warped into the bas-relief space and thus to obtain the *high* portion.

The full creation process of a high-relief is illustrated in Figure 7. The top-left image in the figure represents the bas-relief of a cloister’s model (Pieve di Cadore, Italy; model by Adriano Del Fabbro). A simple rotation of the bas-relief (Figure 7, top-right image) shows how the height field represented by the front pillar occludes almost completely the fountain. In order to avoid this, we have to select an appropriate threshold  $z_t$  and separately create the *bas* and *high* portions of the relief.

The *bas* portion is generated by adding a clipping plane to the scene (to prevent any detail nearer than  $z_t$  from being written on the  $z$ -buffer) and using the algorithm described in Section 3 (Figure 7, middle-left). The *high* portion is obtained by applying the perspective transformation to the vertices of the 3D model and then clipping every part of it farther than  $z_t$  (Figure 7, middle-right). Finally, the whole process is completed by joining the two portions (Figure 7, bottom-left). A rotation of the relief (Figure 7, bottom-right) now shows its *high-relief* nature. Another example of high-relief is shown in Figure 10.

This technique can also be used for the 3D scenes containing object which give rise to high perspective distortions along the relief direction. In these cases the parts of the relief closer to the observer have to be considered *high* parts.

## 5 Conclusions

Though the results produced by the simple techniques proposed in this paper are generally good, it is important to underline how some modeling tricks can improve the output quality in a sensible way.

In modeling its 3D input scene, the *artist* has to reduce the size of the gaps in the  $z$ -direction between the different figures because these gaps result in excessively high steps between the bas-relief figures and the background. The presence of empty space intervals in the relief transform direction means that a

substantial section of the output relief depth is wasted, and some components of the scene may be excessively squashed, losing most of their 3D appearance (at the extreme case, they become nearly 2D silhouettes). To produce more aesthetically pleasing reliefs, most of the height should be dedicated to showing the slope of the objects in the scene. For simple cases, a possible solution to this problem could be the identification and *cut* of the  $Z$ -depth ranges which have no primitives in them.

A possible improvement of the proposed high-relief technique consists in replacing the clipping plane which separates the *bas* and *high* parts of the model with a more flexible and user-friendly picking selection of the figures or parts of the viewplane to be assigned to the *high* (or *bas*) portion. In this way we could obtain *high* objects at different depths in the output relief. However, we can not go beyond the true limitation of the proposed method: our technique does not allow the generation of high-reliefs in which the *high* portion is full 3D (i.e. with no perspective distortion). This is because we need to join the *bas* and *high* portions of the relief into a unique geometric space, the bas-relief space. Moreover, a smooth transition between perspective transformed and regular 3D objects would imply the loss of the linearity of the transformation.

A prototype implementation of the proposed techniques, based on the image precision approach, has been developed using the Inventor library [10]. The system permits the interactive creation of bas- and high-reliefs out of Inventor 3D models and is available at our web site<sup>2</sup>. The prototype also provides functionalities for coloring reliefs and generating cameos.

The automatic generation of relief models can have interesting applications, for example in the creation of profiles on coins or cameos. Using current 3D range scanners or common CAD modeling techniques we can acquire/design the 3D model from which digital reliefs may be automatically created. Moreover, the availability of stereolithography techniques for the prototyping process [8] facilitates the industrial application of this technique.

## Acknowledgements

This work was supported by the Progetto Finalizzato Beni Culturali (Tema 5.3 - Sistemi Museali) of the Italian National Research Council (C.N.R.). We gratefully acknowledge the editor and the anonymous reviewers for their comments and suggestions.

---

<sup>2</sup> <http://miles.cnuce.cnr.it/cg/basrel.html>



Fig. 8. Bas-relief of a cloister.

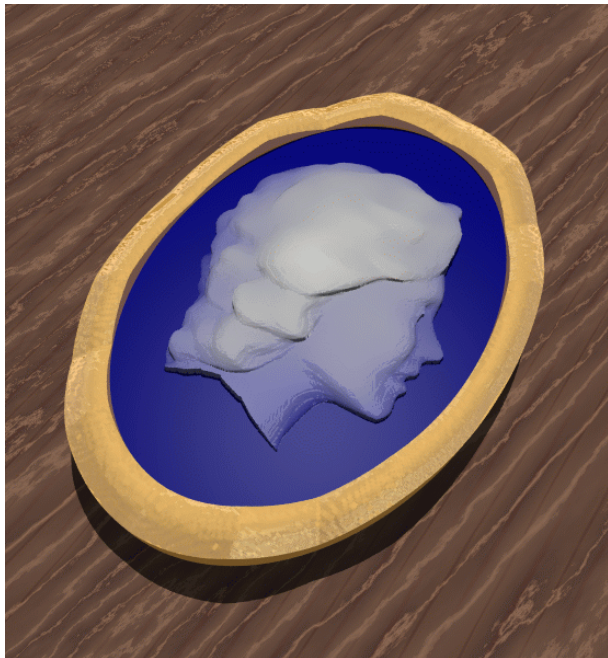


Fig. 9. A cameo of Marilyn Monroe's head.



Fig. 10. High-relief of a dancing skeleton.

## References

- [1] E. Catmull. *A Subdivision Algorithm for Computer Display of Curved Surfaces*. PhD thesis, Computer Science Department, University of Utah, December 1974.
- [2] A. Ciampalini, P. Cignoni, C. Montani, and R. Scopigno. Multiresolution decimation based on global error. *The Visual Computer*, 13(5):228–246, June 1997.
- [3] J. Foley, A. van Dam, S. Feiner, J. Hugues, and R. Phillips. *Introduction to Computer Graphics*. Addison Wesley, 1993.
- [4] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Mesh optimization. In *ACM Computer Graphics Proc., Annual Conference Series, (Siggraph '93)*, pages 19–26, 1993.
- [5] POV-Team. Persistence of vision raytracer 3.0. Publicly available on web: <http://www.povray.org/>, 1996.

- [6] William J. Schroeder, Jonathan A. Zarge, and William E. Lorensen. Decimation of triangle meshes. In Edwin E. Catmull, editor, *ACM Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 65–70, July 1992.
- [7] S. Sechrest and D. P. Greenberg. A visible polygon reconstruction algorithm. *ACM Trans. on Graphics (USA)*, 1:25–42, January 1982.
- [8] P. Stucki, J. Bresenham, and R. Earnshaw. Computer graphics in rapid prototyping technology. *IEEE Computer Graphics and Applications*, 15(6):17–19, Nov 95.
- [9] K. Weiler and P. Atherton. Hidden surface removal using polygon area sorting. *ACM Computer Graphics*, 11(2):214–223, 1977.
- [10] Josie Wernecke. *The Inventor mentor: programming Object-oriented 3D graphics with Open Inventor*. Addison Wesley, 1994.