EUROGRAPHICS 2002 / I. Navazo Alvaro and Ph. Slusallek
(Guest Editors)

Normal Enhancement for Interactive Non-photorealistic Rendering

P. Cignoni†, R. Scopigno‡ and M. Tarini§

Istituto di Scienza e Tecnologia dell'Informazione – Consiglio Nazionale delle Ricerche, Pisa, Italy

Abstract

We present a simple technique to improve the perception of shape of an object. Bump mapping is well-known in the computer graphics community for providing the impression of small-scale geometrical features (which actually are not there). Here, we propose a similar approach (variation of normals) for the purpose of enhancing perception of the given geometry. Our approach is based on a simple modification of the surface normals in order to enhance the geometric features of the object during the rendering. The enhanced normals produced by this approach can be used in any rendering technique. The technique presented is particularly well suited to improve the rendering of mechanical parts where common cheap shading techniques can often generate shading ambiguities.

1. Introduction

In this paper we consider a class of 3D objects, including but not limited to typical mechanical parts used in CAD systems, that have a common set of features: flat surfaces, many of which facing the same direction, sharp straight edges, overall regularity. Straightforward rendering of such objects often results in visually unsatisfactory, dull, flat looking, or even unclear and ambiguous images (see Figure 1 and 2).

Adding enough realism, the problem could disappear: complex realistic effects (common in off line rendering), like cast soft-shadows, inter-reflections, radiosity, local (as opposed to at infinite) light positions, and so on, can produce a much less *flat* result, and are known to provide many intuitive visual hints to the viewer.

In graphic design illustrations (either totally hand made, or made with vector based drawing programs) the problem has been solved also in a different, simpler yet effective way: professional illustrators can reduce flatness (or unclarity) "by hand", shading surfaces according to their esthetic sense (see for example Figure 1) rather then solving difficult physical problems (shadow projection, light diffused by surfaces, etc).

† cignoni@iei.pi.cnr.it
‡ roberto.scopigno@cnuce.cnr.it
§ tarini@iei.pi.cnr.it

The implicit idea behind this is that appropriate shading supplies a kind of information that is more *qualitative* than *quantitative* in the perception of an image. Conversely, the shape of the silhouette and the shading discontinuity bring us the most significant information about the real shape of the object. Moreover, to obtain an improved perception shading does not have to be physically correct (see Figure 1).

Along these lines, we designed a new perception-oriented, non-realistic, automatic technique for interactive rendering systems. We aim at synthesizing images that are qualitatively similar to the illustration style visible, for example, in Figure 2. It is based on enhancing high frequency components of the model; the key issue is that, rather than working on the geometry (vertex positions) of the digital model, we apply the enhancing to the surface orientation alone, leaving the silhouette unchanged. This technique, hereafter called *normal enhancement*, is done on the mesh in a preprocessing stage: the enhanced normals are integrated into the model, making this technique view-independent.

In contrast with most non-photorealistic techniques, this approach is de-coupled from the rendering algorithm used to effectively produce the image. For this reason the enhanced normals can be used into any rendering subsystem that support user-specified normals, like for example the standard VRML browsers. Moreover, a visualization tool or a geometry browser that uses this technique can easily allow the user

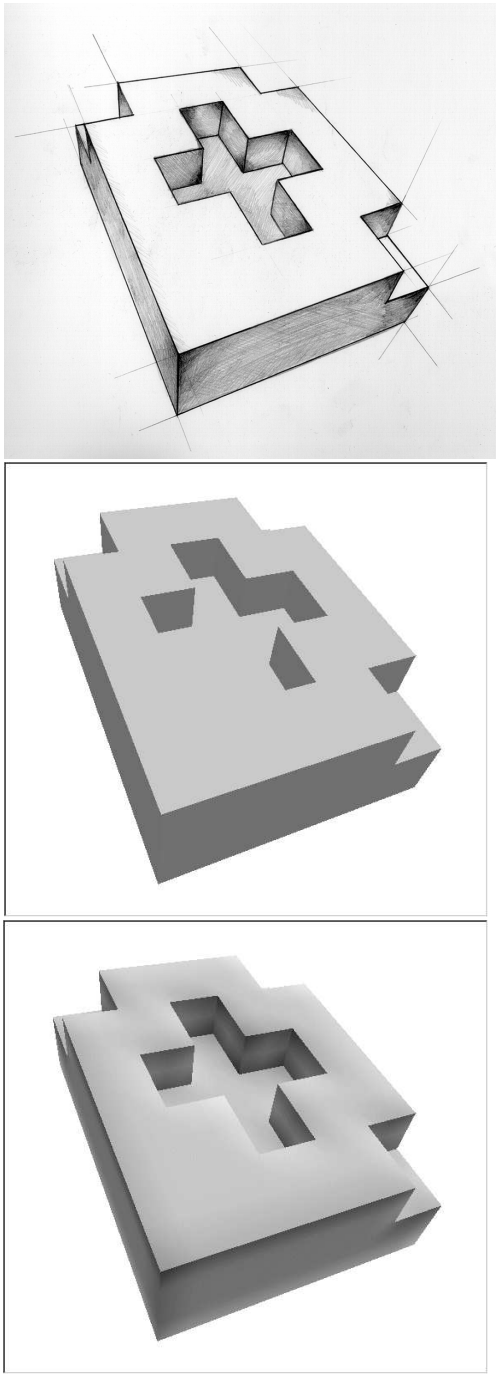

Figure 1: *A hand-drawn pencil drawing with a non-photorealistic shading that enhances the mesh features (top), courtesy of Alessandro Briglia; real-time rendering of a similar object without (center) and with the proposed method (bottom).*

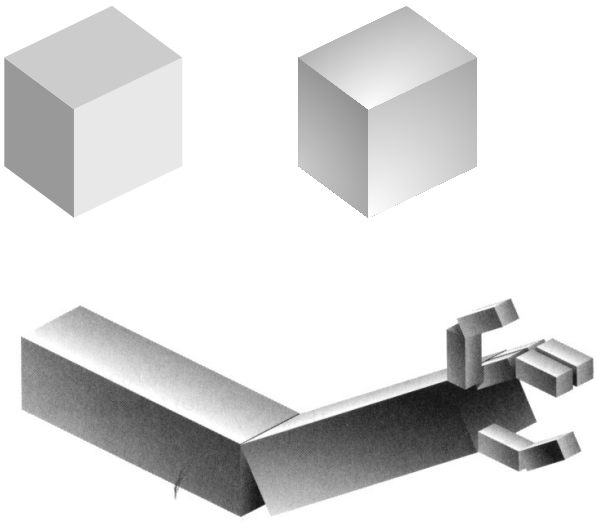

Figure 2: *Examples of non-synthetic, perceptual-oriented drawings of two simple 3D objects: the cube above is drawn in a non-constant, non-realistic manner on the right; the robot arm is a drawing published in Figure 3.26 of the red book on OpenGL* [24].

to toggle between the normal-enhanced and standard rendering modes.

2. Related Work

Computer graphics algorithms and techniques that aim to imitate non-photographic illustration styles are usually referred to as *non-photorealistic rendering* [6] (NPR). NPR techniques greatly differ in style, visual appearance, and objectives. Many of the presented approaches aim to mimic/imitate some existing artistic techniques or styles like watercolour [2], pen-and-ink [23, 16], Charcoal[11] or while other works take inspiration from the field of technical and professional illustration. The latter techniques have the main goal of providing a better comprehension of a given three dimensional structure. Many NPR approaches have been proposed in the last few years making NPR a new branch of Computer Graphics. An extensive annotated survey of online resources on NPR has been presented by C. Reynolds [14].

Here, for sake of conciseness, we limit ourselves to review only the papers more related to the field of computer generated rendering of technical illustrations. This kind of problem was probably first faced by Saito and Takahashi [15], who proposed some techniques to enhance the visual comprehensiveness of a 3D images by means of some post-rendering image-based processing applied to the final image.
This problem has been faced from a more abstract point of

view in some papers [22, 18] where, without introducing new rendering techniques, the problem of the perception of various kind of information through the use of computer generated illustrations was discussed.
Gooch et al. presented a non-photorealistic lighting model that provides a better shape comprehension by mapping the change in surface orientation into variations of hue instead of brightness variations [5]. This technique can also be efficiently implemented using current graphics hardware [7].
Another common way, pioneered by the work of [4], consist in detection and outlining of certain elements of the model (like silhouettes and sharp edges). Recently Raskar [13] proposed a graphic accelerated approach these elements are drawn in real time without being explicitly detected in any preprocessing phase.

In the sense of the above contributions, the normal enhancement technique here presented can be helpful in generating sharper and less ambiguous images. Moreover an advantage of this technique is a seamless integration with existing rendering systems.

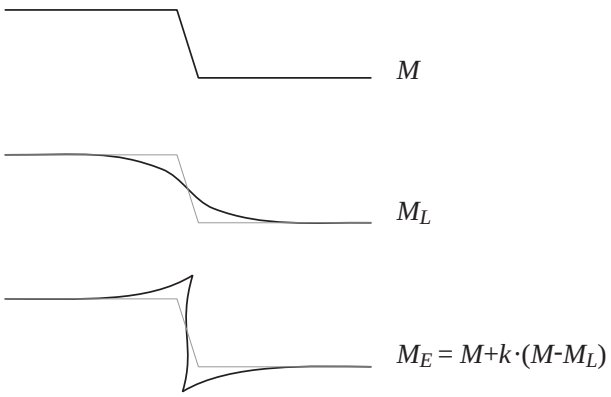

Figure 3: *High frequency components of a mesh M can be enhanced by summing to M the weighted difference between M and a M_L where M_L is a smoothed representation of M.*

3. Enhancement of Mesh Features

Using a rather informal signal processing terminology, we can say that to *sharpen* a 3D mesh M we must enhance the *high frequency* components of that mesh. A simple way to compute these components is to make a low pass filtering of M by means of a Laplacian smoothing kernel [20, 21], obtaining a *low frequency* mesh M_L (see Fig. 3). Then the *high frequency* component can be recovered by computing the difference between the original mesh and the smoothed one, i.e. $M - M_L$. The desired result of an *high frequency* enhancement can be obtained by adding this component to the original mesh, scaled by a user specified constant factor k: $M_E = M + k(M - M_L)$.

This high frequency enhancement technique works only if the starting mesh is rather densely and uniformly tessellated or, in other words, if the triangles are small with respect of the size of most of the features of the mesh and the ratio between the largest and the smallest triangle edges is not too large. Otherwise the smoothing process leads to erroneous results. Common smoothing techniques for triangular meshes [20, 21] do not work if the mesh lacks the above properties. For this reason hereafter we assume that if the starting mesh exhibits a large disparity of triangle sizes it is preprocessed and all the faces larger than a given threshold are split.

It must be noted that a more formal and correct signal mesh processing could be done on a generic mesh, as presented for example in a paper by Guskov et al. [8], but our goal is much less elaborate: we want only to enhance the visual presentation of an object in order to improve the perception of some features.

An example of the results of this technique is shown in Figure 4: given a simple mesh M, we apply a scale-dependent Laplacian filter [20, 3] and obtain a smoothed representation M_L; the enhanced mesh M_E is obtained by summing to each vertex of M the difference vector between itself and the corresponding vertex of M_L. But we have introduced in Section 1 that we do not want to modify the geometry of the input mesh, so let us describe how we perform a similar transformation by working just with normal vectors and rendering.

4. Normal Enhancement

The effects obtained by modifying the geometry of the mesh using the technique discussed in the previous section are interesting and, in some situations, they can be useful. On the other hand, the mesh produced is an object that the users in some sense perceive as inherently different from the original one (at least because the silhouette is changed, see Figure 4). As noted before, the shading of the surface conveys a lot of *qualitative* information, and we may try to make use of it to improve the perception of surface features.

This is the reason why we propose to modify only the normals of the object instead of the coordinates of the mesh. In this way, affecting only the shaded appearance of the object but keeping untouched its silhouette and its geometry extent, we achieve the desired enhancement in the rendered images without the drawback of having a distorted objects in our rendering. The methodology remains analogous to the one described in the previous section, but it is applied only to the surface normals (see Figure 5) as follows:

1. For each face, compute a new normal n_L as a low pass filtering of the normals of the mesh; this is done by iteratively substituting each face normal with a re-normalized

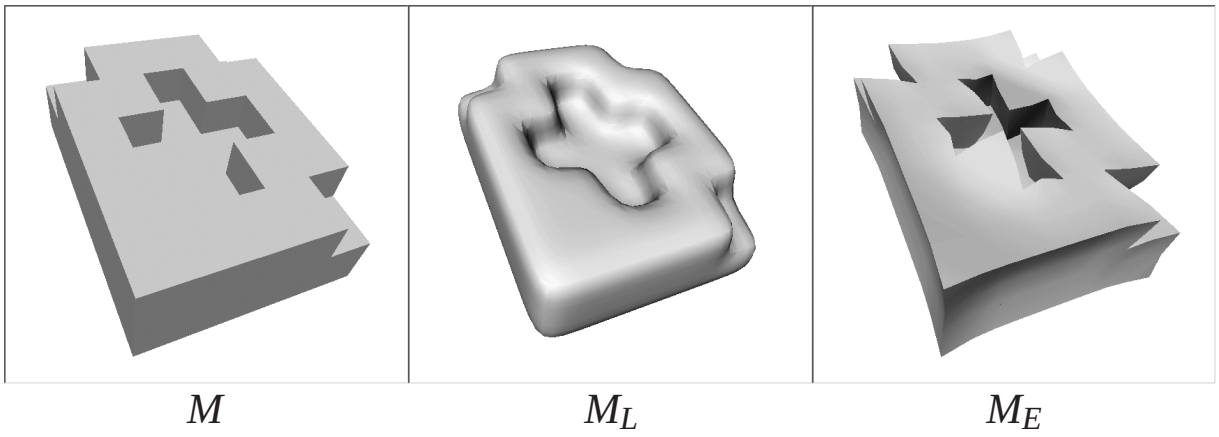

Figure 4: *Enhancement of high frequency geometry components of a mesh.*

area-weighted average of the normals of the adjacent faces.

2. For each normal vector, enhance it by pushing back the original normal n in the opposite direction of its averaged counterpart n_L and then re-normalize the resulting vector, obtaining an enhanced normal $n_E = n + k \cdot (n - n_L)$.

Note that the same approach could be obtained also by applying the geometry enhancement technique presented in Section 3 and replacing the normals of the vertices of M with the normals of the vertexes of M_E. This can be done in a straightforward manner because there is a one-to-one vertex correspondence between M, M_L and M_E. This technique is somewhat more expensive because it requires to store both the original and the modified vertex positions. Even if, from an abstract point of view, this approach could be more *correct*, because it allows the exploitation of better smoothing or fairing techniques, we have found that it is more sensible to the tesselation quality of the starting mesh. We have performed empirical tests and we have not found such an improvement in the final quality of the result to justify the adoption of this latter approach.

The result of the above procedure is a new set of per-face normals. In order to obtain a high quality shaded rendering it is necessary to correctly compute per-vertex normals. This can be done by using the standard approach of averaging together those face normals of adjacent faces whose normals differs less than a user specified crease angle. In this way the sharp discontinuity of the mesh are preserved while regions with low curvature exhibit a smoother shading.

4.1. Bump Mapping

When the original object is composed by a small number of faces with many sharp features, our technique needs an initial refinement step. This refinement is needed to ensure the correctness of the smoothing pass. In some cases, this refinement step can heavily increase the initial complexity of the object (even by a factor of ten or more). In these cases, it may be unacceptable to render, let us say, ten times more triangles to produce a smarter rendering. But because the technique presented in this paper leaves the surface geometry untouched and modifies just the surface normals, we can encode the resulting new normals into a new normal texture map which can be mapped onto the original mesh. In this case the detail recovering technique presented by Cignoni et al. [1] and further improved by Sander et al. [17] can be applied to re-sample a normal map from the refined mesh produced by of our normal-enhancement algorithm and the associated set of normal vectors. We can then use common graphics hardware to render the synthesized map very efficiently [9].

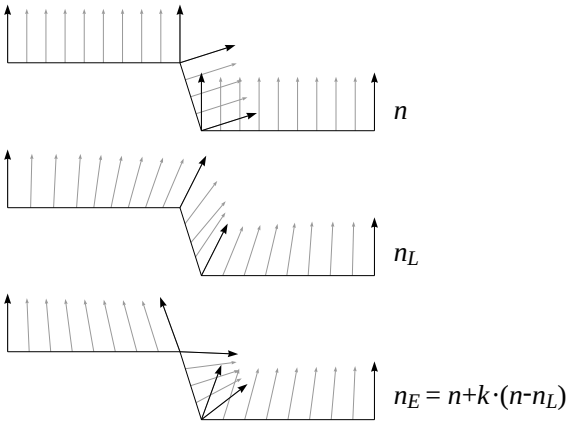

Figure 5: *Enhancement of high frequency components of a mesh normals vectors.*

Note that this approach could lead to yet another normal enhancement approach: if there exists a *face-continuous*

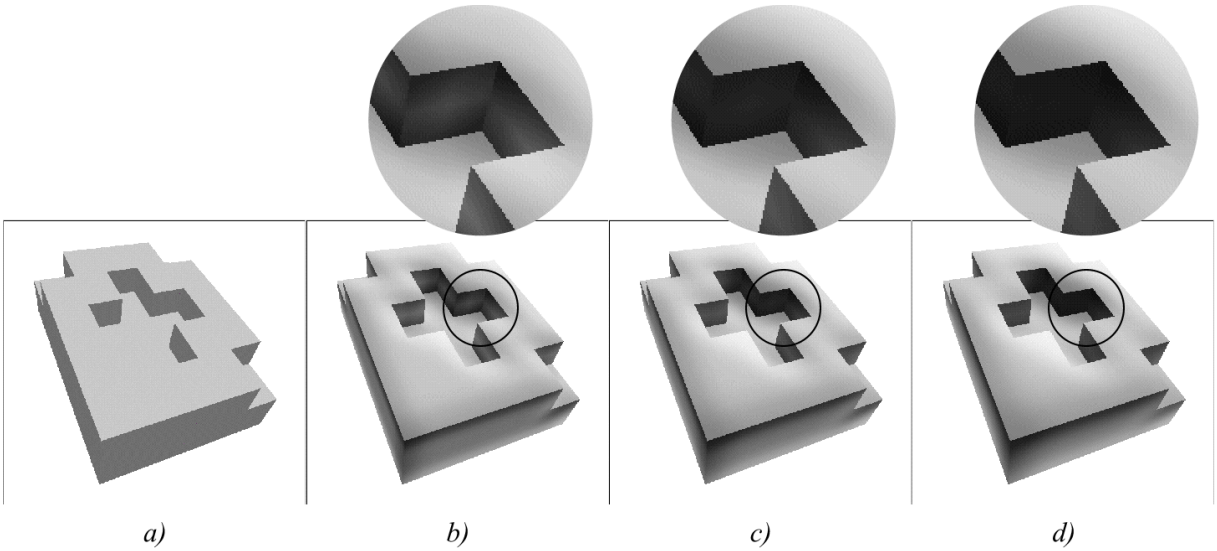

Figure 6: *The amount of normal smoothing affects the normal enhancement of the smallest features.*

mapping from surface to texture space (e.g. a good texture parameterization of the mesh) then we could build the bitmap representing the normal map of the mesh and then apply on it standard image processing techniques. In most cases, this approach is not feasible because, with some trivial exceptions, it does not exist a simple way to build such a texture parameterization; existing techniques [1, 12, 19, 10] either build discontinuous mappings or exhibit deformations that make an image processing approach either difficult or not feasible.

5. Results

The normal enhancement effect that can be obtained with the application of the formula $n_E = n + k \cdot (n - n_L)$ depends mainly on two parameters: the amount of low-pass filter that we use to generate the smooth normals n_L and the value of the weighting constant k used in the perturbation of the original normal vectors. By tuning these two parameters we can obtain slightly different visual results.
The weighting constant k affects the *intensity* of the normal enhancement effect; as a rule of thumb we have found, for this parameter, reasonable values in the range [0.2..0.7].
As introduced in Section 4, low pass filtering is performed by adopting a simple Laplacian kernel: we iteratively average each face normal with the normals of adjacent faces. The number of iterations of this averaging process affects the extent of the smoothing process. By using a large number of smoothing steps the smallest features of a mesh can totally disappear in the smoothed representation and, for this reason, their enhancement can become uniform and therefore less detectable. On the other hand, by using a large number of smoothing steps, we obtain a larger extension of the shaded section that can be useful for large features. Figure 6 shows this situation: (a) is the original mesh, while $(b), (c)$ and (d) show the effect of normal enhancement by using, respectively 10, 20 and 30 normal averaging iterations. The enhancement is visible in the cross shaped hole: the central section of the hole faces is rendered with a slightly lighter shading, that almost vanish in the last two instances (c) and (d). On the other hand, in all cases shown in figure the shading produced by the proposed technique resolve the shading ambiguity occurring in the original mesh.

The use of a *scale-dependent* Laplacian filtering of normals allows to reduce the problems arising from uneven tessellation by weighting the normal influence on the neighboring triangles with their size.

Some other examples of the application of the proposed technique are shown in Figure 7. For each object we show on the left the mesh rendered using standard OpenGL shading of the input mesh, while on the right are the results rendered using the enhanced normals (OpenGl shading with bump mapping). For each pair of images, all the rendering parameters (lights, materials, etc) except surface normals remain unchanged. Note that, in most cases, the normal enhancement technique allows to resolve many shading ambiguities.

The technique can be applied over irregularly shaped objects, e.g. the Stanford bunny in Figure 8, rather than the highly regular object we focused on: the result is a sort of high frequency detail enhancement (the fact that the silhouette and the shape of the object is unchanged is hardly notable in that case). Figure 8 could suggest that the effect of the technique proposed is very similar to what can be obtained by a simpler contrast enhancement (e.g. adopting standard 2D image-processing filters on a rendered image).

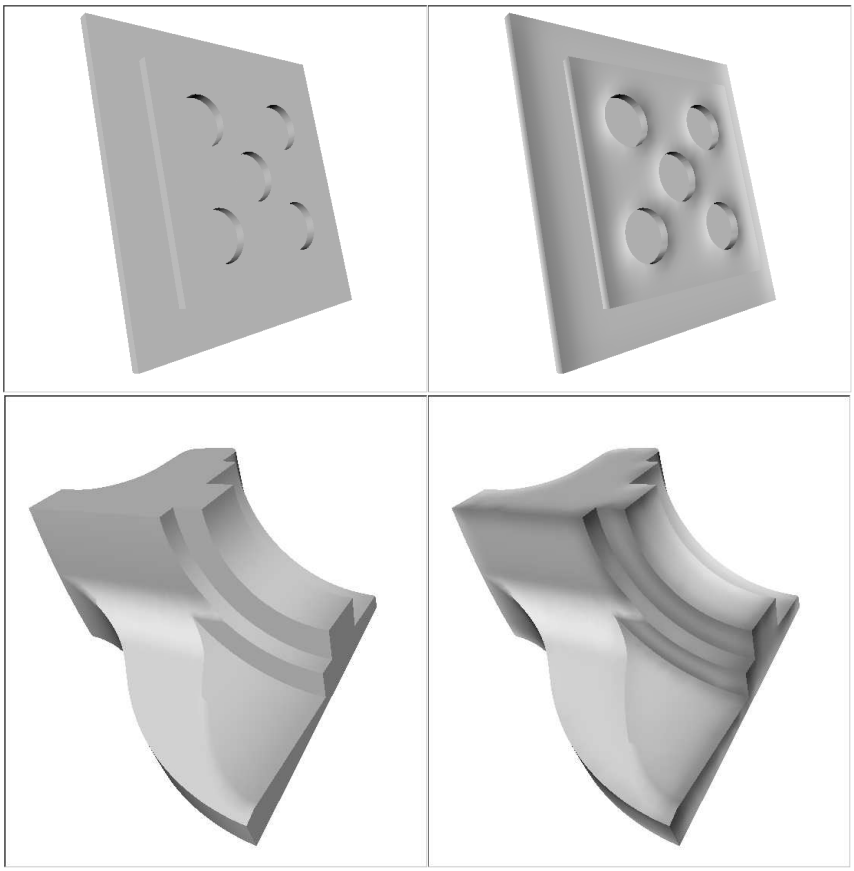

Figure 7: *Two examples of normal enhancement.*

This is true just in the case of a purely diffuse material lit by a single light source positioned on the same location of the viewer. Conversely, if we have a more complex lighting environment or the object is rather specular, it emerges the intrinsic 3D nature of the enhancement. As an example, see Figure 8 which shows also some images of a shiny bunny.

Since the simplicity of the proposed technique, its implementation is straightforward. The expected running time of the normal enhancement and resampling process is usually very low, in the orders of a few seconds for any mesh that can be rendered interactively.

5.1. Side effects

Under particular rendering conditions, some enhanced model may suggest an artifact concavity of flat surfaces (for example, see the top-right image in Figure 7). This effect however can be kept under control using low values for either of the two parameters (still getting most of the visual improvements). Nor it is always the case that this represents a real disadvantage: there are a number of applications where the context clearly determines the object's regularity or planarity, such as mechanical CAD parts, architectural and interior design etc. In all these cases, the effect of our technique can be considered some sort of "artistic" license. There are many applications (e.g. assembly instructions) where the geometry of the objects in the scene is known, and the focus of the visual presentation is to clarify the respective positions and inter-relations between parts.

On the other hand, it has to be noted that we do not modify the geometry, and thus our approach can be used as a non-permanent modification of the object obtained just using a different rendering modality, which could be toggled on/off by the user in the inspection of a given object.

6. Conclusions

We have presented a technique that enhances the shading and the perception of its features by modifying the normals of an object. This *normal enhancement* technique is done on the mesh in a preprocessing stage; the enhanced normals are integrated into the model either by assigning new normal values per vertexes, or through resampled normal maps.

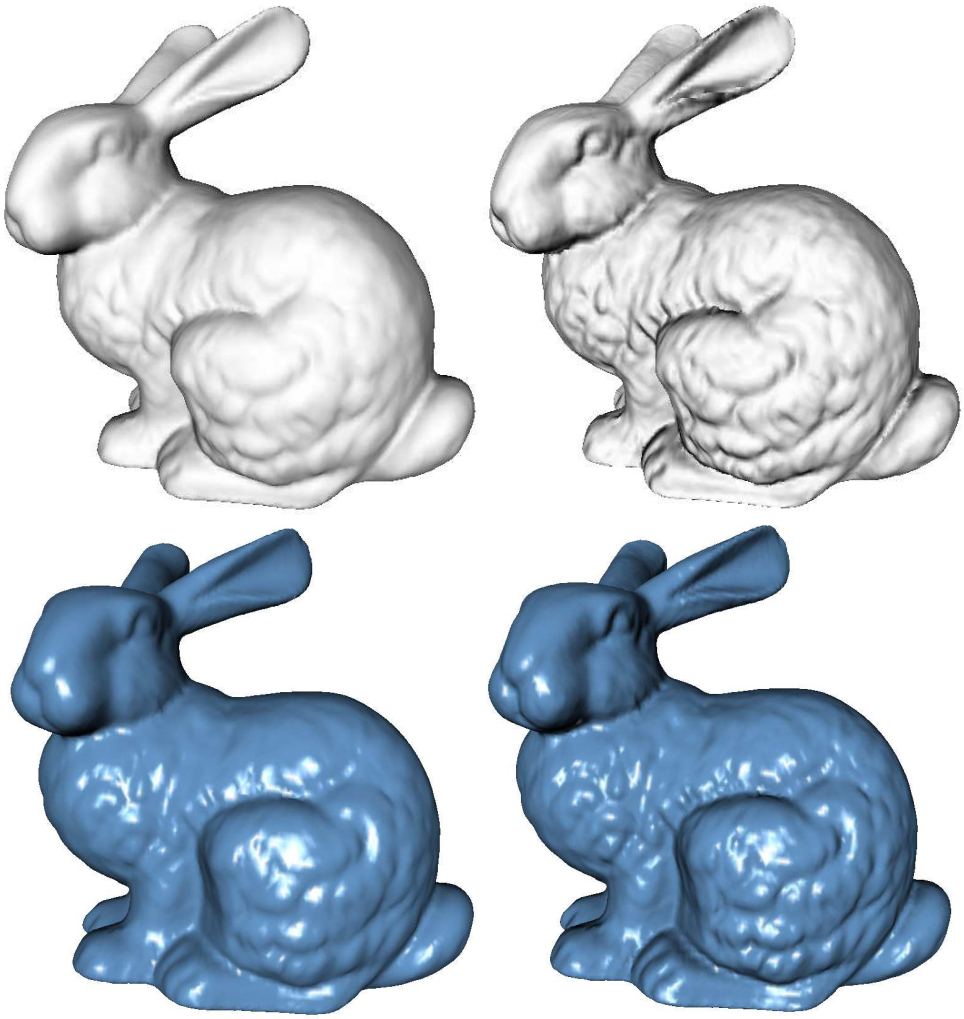

Figure 8: *Enhancement of high frequency geometry components of an irregularly shaped object (left: original mesh); a diffuse shading is in the top images, while a more specular material is adopted in the ones on the bottom.*

This approach is thus decoupled from the rendering algorithm used to effectively produce the image. The enhanced normals, mapped to the input geometry using a standard texture mapping approach, can then be used into any rendering subsystem that support user-specified normals, or interactive bump mapping.

The technique is especially well suited for regular objects (such as CAD models), but can also be used on any 3D mesh: the enhanced normals, once used in a rendering process, results in images that look more “sharp" and intuitive in the sense that they support a better perception of the shape of the represented object and present less ambiguities. These synthetic images are not quite realistic, but closely resemble a style commonly used by illustrators for the same category of objects.

References

1. P. Cignoni, C. Montani, C. Rocchini, R. Scopigno, and M. Tarini. Preserving attribute values on simplified meshes by re-sampling detail textures. *The Visual Computer*, 15(10):519–539, 1999. (preliminary results appeared in IEEE Visualization ’98 Proceedings). 4, 5

2. C.J. Curtis, S.E. Anderson, J.E. Seims, K.W. Fleischer, and D.H. Salesin. Computer-Generated Watercolor. In Turner Whitted, editor, *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series, pages 421–430. ACM SIGGRAPH, Addison Wesley, August 1997. ISBN 0-89791-896-7. 2

3. M. Desbrun, M. Meyer, P. Schröder, and A.H. Barr. Implicit fairing of irregular meshes using diffusion and

Figure 9: *Some more results on two CAD-like meshes.*

curvature flow. In Alyn Rockwood, editor, *Proceedings of the Conference on Computer Graphics (Siggraph99)*, pages 317–324, N.Y., August8–13 1999. ACM Press. 3

4. D. Dooley and M.F. Cohen. Automatic Illustration of 3D Geometric Models: Surfaces. *IEEE Computer Graphics and Applications*, 13(2):307–314, 1990. 3

5. A. Gooch, B. Gooch, P. Shirley, and E. Cohen. A Non-photorealistic Lighting Model for Automatic Technical Illustration. In *Computer Graphics*, July 1998. 3

6. B. Gooch and A. Gooch. *Non-Photorealistic Rendering*. A. K. Peters Ltd, 2001. 2

7. B. Gooch, P.-P.J. Sloan, A. Gooch, P. Shirley, and R. Riesenfeld. Interactive technical illustration (color plate S. 220). In Stephen N. Spencer, editor, *Proceedings of the Conference on the 1999 Symposium on interactive 3D Graphics*, pages 31–38, New York, April 26–28 1999. ACM Press. 3

8. Igor Guskov, Wim Sweldens, and Peter Schroeder. Multiresolution signal processing for meshes. In *Proceedings of SIGGRAPH 99(Los Angeles, CA, August 10-14). In Computer Graphics Proceedings, Annual Conference series, ACM SIGGRAPH*, pages 325–334, 1999. 3

9. M.J. Kilgard. Pratical and robust bump mapping technique for today's GPU's. In *Advanced Opengl Game development*, 2000. 4

10. J. Maillot, H. Yahia, and A. Verroust. Interactive texture mapping. In *ACM Comp. Graph. Proc., Annual Conf. Series (Siggraph'93)*, pages 27–34, 1993. 5

11. A. Majumder and M. Gopi. Hardware accelerated real time charcoal rendering. In Gregory M. Nielson and Dan Bergeron, editors, *to appear in Proceedings of the Non-Photorealistic Animation and Rendering*, pages 291–299, Annecy, France, June 2002. 2

12. M. Maruya. Generating texture map from object-surface texture data. *Computer Graphics Forum (Proc. of Eurographics '95)*, 14(3):397–405, 1995. 5

13. Ramesh Raskar. Hardware support for non-photorealistic rendering. Apr 2001. 3

14. C. Reynolds. Stylized depiction in computer graphics non-photorealistic, painterly and 'toon rendering. Publicly available on web: `www.red3d.com/cwr/npr/` , 2001. 2

15. T. Saito and T. Takahashi. Comprehensible Rendering of 3-D Shapes. In *Computer Graphics (SIGGRAPH '90 Proceedings)*, volume 24, pages 197–206, August 1990. 2

16. M.P. Salisbury, M.T. Wong, J.F. Hughes, and D.H. Salesin. Orientable Textures for Image-Based Pen-and-Ink Illustration. In *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series, pages 401–406. ACM SIGGRAPH, Addison Wesley, August 1997. ISBN 0-89791-896-7. 2

17. Pedro V. Sander, Xianfeng Gu, Steven J. Gortler, Hugues Hoppe, and John Snyder. Silhouette clipping. In *SIGGRAPH 2000, Computer Graphics Proceedings*, Annual Conference Series, pages 327–334. Addison Wesley, 2000. 4

18. S. Schlechtweg and T. Strothotte. Generating Scientific Illustrations in Electronic Books. In *Smart Graphics. Papers from the 2000 AAAI Spring Symposium (Stanford, March, 2000)*, pages 8–15, Menlo Park, 2000. AAAI Press. 3

19. M. Soucy, G. Godin, and M. Rioux. A texture-mapping approach for the compression of colored 3d triangulations. *The Visual Computer*, 12(10):503–514, 1996. 5

20. G. Taubin. A signal processing approach to fair surface design. In *Comp. Graph. Proc., Annual Conf. Series (SIGGRAPH 95), ACM Press*, pages 351–358, Aug. 6-12 1995. 3

21. J. Vollmer, R. Mencl, and H. Müller. Improved Laplacian smoothing of noisy surface meshes. *Computer Graphics Forum*, 18(3):131–138, 1999. 3

22. L.R. Wanger, J.A. Ferwerda, and D.P. Greenberg. Perceiving Spatial Relationships in Computer-Generated Images. *IEEE Computer Graphics and Applications*, 12(3):44–58, May 1992. 3

23. G. Winkenbach and D.H. Salesin. Computer–Generated Pen–And–Ink Illustration. In A. Glassner, editor, *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, Computer Graphics Proceedings, Annual Conference Series, pages 91–100. ACM SIGGRAPH, ACM Press, July 1994. ISBN 0-89791-667-0. 2

24. M. Woo, J. Neider, T. Davis, and D. Shreiner. *OpenGL Programming Guide - Third Edition*. Addison Wesley, 1999. 2