Geometry-Dependent Lighting

Chang Ha Lee, Xuejun Hao, and Amitabh Varshney, *Member, IEEE*

***Abstract*— In this paper we introduce *geometry-dependent lighting* that allows lighting parameters to be defined independently and possibly discrepantly over an object or scene based on the local geometry. We present and discuss *Light Collages*, a lighting design system with geometry-dependent lights for effective feature-enhanced visualization. Our algorithm segments the objects into local surface patches and places lights that are locally consistent but globally discrepant to enhance the perception of shape. We use spherical harmonics for efficiently storing and computing light placement and assignment. We also outline a method to find the minimal number of light sources sufficient to illuminate an object well with our globally discrepant lighting approach.**

***Index Terms*— Lighting design, scientific illustration, discrepant lighting, light placement, silhouette enhancement, proximity shadows, spherical harmonics**

I. Introduction

OUR ability to generate 3D data, through acquisition and through simulation, has far surpassed our ability to visually comprehend it. As the data sizes continue to increase at a geometric rate of growth, it becomes necessary for us to revisit the traditional visualization pipeline to explore its stages that we can modify to enhance the comprehension of intricate model details. We believe that careful lighting design offers one such avenue of research.

Lighting design has long been considered crucial in conveying the right ambience, emotion, visual complexity, context, and in guiding the viewer's attention in art, scientific illustration, photography, stage lighting, and cinematography. Over two millennia ago Pliny the Elder discussed locally shading a surface fold to make it appear to rise above the background [1], [2]. Since then, artists and illustrators have successfully used local lighting techniques for conveying the object shape. These local techniques convey a powerful impression of geometry, although the lighting across the surface is inconsistent.

C. H. Lee and A. Varshney are with the Department of Computer Science, University of Maryland, College Park, MD 20742. E-mail: {chlee, varshney}@cs.umd.edu

X. Hao is with the Department of Psychiatry, Columbia University, 1051 Riverside Drive, Unit 74, New York, NY 10032. E-mail: xh2108@columbia.edu

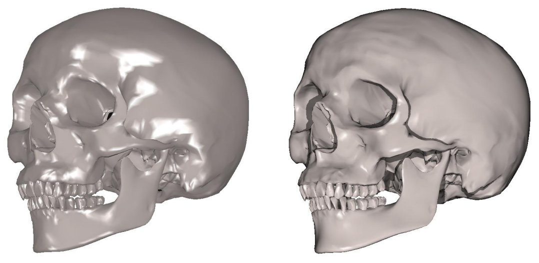


Fig. 1. *(a) Consistent lighting with four lights at the front four vertices of a cube, and (b) a Light Collage rendering with 4 lights. Material properties are the same for both renderings.*

Since the world around us is lit consistently, it was possible that we might have naturally acquired the ability to discern illumination inconsistencies of lighting directions. However, recent research by Ostrovsky *et al.* [3] found that human subjects were largely insensitive to illumination inconsistencies across a set of randomly-oriented 3D cubes. This helps explain why the geometry of consistent lighting is not as meticulously crafted in art as the geometry of perspective [4]. There are also other reasons why artists and illustrators may allow lighting to be inconsistent. First, efforts to ensure consistent lighting in art are usually under-appreciated since they are not visually obvious. Second, artists can use inconsistent lighting to guide the viewer's attention to enhance comprehensibility or convey their message. If one were to apply the inverse lighting models that have been developed recently [5], [6] to most paintings and illustrations, one would find innumerable errors (some admittedly slight, but present nonetheless) in their lighting and shading. However, not only have these lighting errors passed virtually unnoticed by most untrained human observers, lighting for such paintings is visually impressive and sometimes even deeply compelling.

Cavanagh [2] has suggested that our brain perceives the shape-from-shading cues locally and does not use large regions of the visual field for shape-from-shading analysis. In fact, recent work by Akers *et al.* [7] and Agarwala *et al.* [8] has shown the power of such an approach for 2D images. They have shown how image composition can be used with sophisticated, spatially-varying light mattes to create compelling technical illustrations or composite photographs from a set of photographs with different, locally discrepant, lighting. In this paper we explore how the use of discrepant lighting in 3D visualization may allow us to convey a better perception of geometry than consistent lighting.

We have presented our work on an automatic lighting design system, Light Collages [9], for enhancing the visualization of scientific datasets. In this paper we give further details of that approach, present more compelling results with new datasets, and introduce the framework of *geometry-dependent lighting*. Important new contributions in this expanded version of our earlier work are in improving the run-time efficiency of our system by a factor of 20 and reducing the memory footprint by over two orders of magnitude. We discuss how one can achieve this by using a spherical-harmonic-basis representation for light placement and assignment. The benefits of adding more discrepant lights diminish with the total number of lights in the system. Another novel contribution of our work is the notion of minimality of light sources for a given view and geometry and showing how this changes with simplifications of the geometry.

II. Previous and Related Work

In photography, cinematography, and stage lighting, the specification of light position, direction, color, intensity, and type determines the appearance of the resulting scene. Kahrs *et al.* [10] have summarized the lighting design approaches for computer animation. They distinguish between logical and pictorial lights. *Logical* lights are motivated by actual sources of light in a scene that the viewer can see or imply. For example, the *key* light is used in a scene as the primary source of illumination. In addition to logical lighting cinematographers use *pictorial* lighting for enhancing the artistic aesthetics of the scene. For example, *back or rim* lights are used to separate the object from the background, and *fill* lights are used to soften and fill the shadows.

Much of the current work on lighting design in 3D graphics and visualization has focused on determining the parameters for logical lights and has generally overlooked pictorial lighting. We classify the lighting design methods for graphics as either *direct* or *indirect*. Conventional lighting design methods are direct – they require a user to directly specify the lighting parameters. The user starts out by specifying an initial set of lighting parameters and then visually evaluates the results. The lighting parameters are then changed iteratively till the graphics rendering converges to a desired output. Although the visual results from using a direct light specification may be satisfactory, the process itself leaves much to be desired. First, direct lighting design is often iterative and time consuming. Second, it requires a significant expertise on the part of the user to achieve desired visual effects from light placement, such as locations of highlights and shadows. The approach of Design Galleries [11] addresses these shortcomings by using several user-specified lighting parameters (excluding light placement), generating a set of renderings with randomly placed lights, and having a user browse and hierarchically select the renderings that are desirable. The LightKit system [12] allows a user to interactively adjust lighting to enhance visualization. This system allows camera-relative lights that include a dominant light, headlights, and backlights. The system also allows the user to adjust the light color and warmth of lighting.

Indirect lighting design methods use scene properties that are either specified by a user or procedurally estimated. In *user-specified indirect lighting design*, the user specifies the desired highlights or shadows and the system then infers the light placement to achieve them [13]–[17]. In *procedural indirect lighting design*, the system automatically infers light placement and parameters by optimizing a set of perceptual criteria for a given view. Shacked and Lischinski [18] derive light placement for up to two light sources by optimizing a perception-based image quality objective function. Their objective function includes six terms that are based on shading gradients, pixel luminance statistics, and illumination direction. Gumhold [19] has developed a light-placement strategy by maximizing a perceptual entropy objective function as measured from a rendered image.

Although we have not come across prior work on physically-discrepant lighting design for 3D graphics and visualization, there is a sizable literature on physically-implausible lighting models collectively referred as *non-photorealistic lighting*. Gooch *et al.* [20] have developed a lighting model that uses luminance and changes in hue to convey surface orientation, edges, and highlights. Sousa *et al.* [21] have incorporated lighting into adaptive pen-and-ink stroke lengths to convey shape. Hamel [22] has developed a lighting model that incorporates five components – standard lighting with shadows, rim shadow lighting, curvature shading, transparency, and volume illumination. Sloan *et al.* [23] have developed an effective method to transfer the shading from one object to another using a sphere (environment map) as an intermediary. Anderson and Levoy [24] have used curvature- and accessibility-based shading [25] to enhance the visualization of cuneiform tablets. Vicinity lighting [26] improves upon the idea of accessibility shading by using uniform diffuse lighting and occlusion by local occluders.

Previous work, to the best of our knowledge, has not tried to render the same object with multiple light

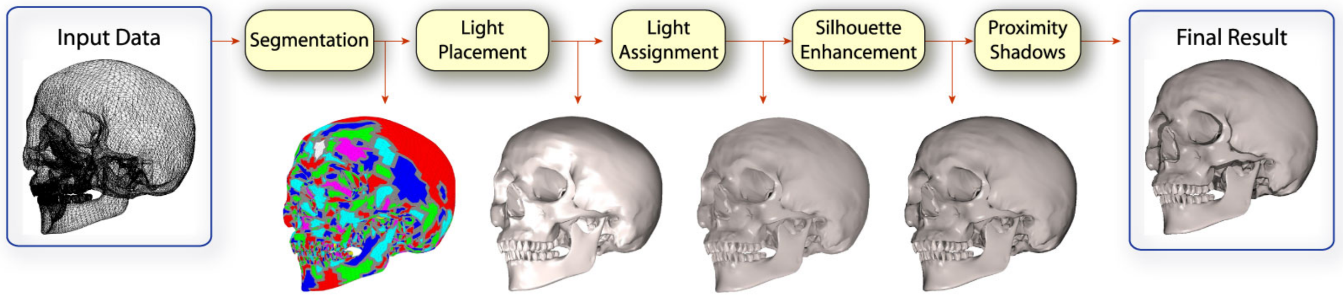


Fig. 2. *Overview of our lighting design pipeline: The input model is segmented using a curvature-based-watershed method into a set of patches. The light placement function models the appropriateness of light directions for illuminating the model. This is done by using the curvature-based segmentation as well as the diffuse and specular illumination at every vertex. Lights are placed and assigned to patches based on the light placement function. Silhouette lighting and proximity shadows are added for feature enhancement.*

sources with each light source lighting a different region of the object. In fact, the general advice seems to have been to illuminate objects with a single light source that is placed above and to the left of the object [27]. In this paper we discuss the idea of geometry-dependent lighting that involves lighting different regions of a 3D object with multiple light sources to render it in a more visually comprehensible manner, while retaining its traditional 3D-graphics-rendered look and feel. Our goal is to provide effective visualization that conveys a large number of data features such as local surface orientation, curvature, silhouettes, and fine texture.

III. Light Collages Overview

The geometry-dependent lighting framework allows local regions to be illuminated by discrepant lights based on their local geometry. Our Light Collages system automatically designs geometry-dependent lighting for a given view by placing directional light sources and assigning them to different regions of an object. Let us define the problem more formally. Consider an object composed of n surface patches $\mathscr{P} = \{p_1, p_2, \ldots, p_n\}$. Let there be a set of m unknown light sources $\mathscr{L} = \{l_1, l_2, \ldots, l_m\}$. The problem we solve here is: *Given $\mathscr{P}$, m, and a viewer position, generate $\mathscr{L}$ and a mapping $\mathscr{M}$ that pairs each light $l_i \in \mathscr{L}, 1 \leq i \leq m$ to a subset of patches $\mathscr{P}_i \subset \mathscr{P}$ that it lights, to best elucidate the local structure of the object*. Here, the subsets $\mathscr{P}_i$ are mutually exclusive and collectively exhaustive: $\mathscr{P}_i \cap \mathscr{P}_j = \emptyset, 1 \leq i, j \leq m$, $\bigcup_{i=1}^{m} \mathscr{P}_i = \mathscr{P}$. Then each patch $p_j \in \mathscr{P}_i$ is assigned a primary light source $l_i = \mathscr{M}(p_j)$. We believe that the idea of *best elucidation* is open to interpretation. There is strong evidence that conveying the local curvature information is important in shape perception. Girshick *et al.* [28] present several compelling visual examples that show that placing line strokes along principal directions of curvature are more effective than other directions. Additionally, user studies on light source placement by Gumhold [19] have indicated that observers tend to select light source directions that favor surface curvature elucidation.

The Light Collages system first segments the input model into a set of patches, then places lights and assigns them to patches, and finally adds silhouette lighting and proximity shadows for feature enhancement as illustrated in Figure 2. In the sections IV–VI, we discuss each stage of the Light Collages pipeline in detail.

IV. Surface Segmentation

We segment the input model into a set of patches to define the local regions which will be lit discrepantly. The segmentation of an object is a classical area of research in computer vision and image processing. Any of the vast number of segmentation algorithms can be used for object segmentation at this stage depending on what the goals of the segmentation-based lighting design are. In this paper, we segment the object into patches based on local curvature. The goal is to make each patch be a collection of triangles with similar curvature values.

We first compute the mean curvature at each vertex of the input mesh as the average of its two principal curva-

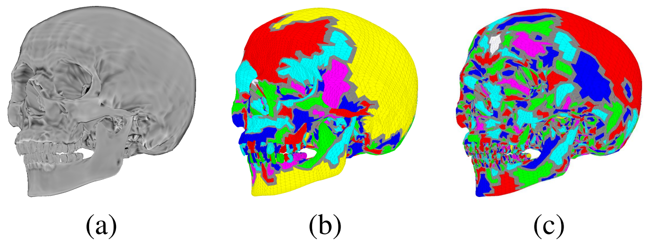


Fig. 3. *(a) Curvature distribution over the Skull model; convex regions are shown brighter and concave regions are darker, (b) coarse segmentation with a high threshold and, (c) fine segmentation with a low threshold. For all models in this paper, we use 7.5% of the range of curvature difference as the threshold and this is shown in (c) for the Skull model.*

tures, which are computed using Taubin's method [29]. Then we segment using a simple watershed algorithm based on Mangan and Whitaker's method [30]. First, their method finds vertices with local curvature minima and uses each of them as a seed for growing a new patch. The method then iteratively assigns vertices to these patches. A path of steepest descent is computed from each unassigned vertex till it reaches a seed vertex with a local curvature minimum. The vertex is assigned to the patch corresponding to this seed vertex. A *watershed depth* is computed for each patch based on the minimum difference in curvature values between a boundary vertex and the seed vertex for that patch. Patches whose watershed depth is below a threshold depth are merged. Figure 3(a) shows the distribution of the curvature over the Skull model and Figures 3 (b) and (c) show how the segmentation can be decreased or increased by raising or lowering the threshold depth, respectively. Figure 3(c) shows the results of our segmentation of the object into multiple surface patches: $\mathscr{P} = \{p_1, p_2, \ldots, p_n\}$.

V. Procedural Lighting Design

We assume that all the lights are white and directional. Our lighting design algorithm proceeds in two interleaved phases. In one phase we identify the placement of a light and in the other phase we assign the light to appropriate patches.

A. Light Placement Function

Curvature influences the illumination gradient across a surface and is an important visual cue to shape. We use a combination of local lighting models to enhance the appearance of high-curvature areas of an object from a given viewpoint. A specular highlight on a shiny surface can easily vanish with even small perturbations of the viewing direction, surface normal, or light direction. For a low-curvature area, the specular highlight hides the subtle geometric changes because of over exposure. However, for a region with high curvature, the specular highlight is useful as it can result in a sharp curvature-based highlight, and thus help illustrate object detail.

As an example, let us consider two points A and B on which we would like to place specular highlights (Figure 4(a)). If we have the freedom to place a directional light source along any direction, we would like to place it in a direction that maximizes the possibility of having highlights on points A and B. We can infer the light directions that will cause specular highlights to appear on points A and B by using the view direction, the shape, and the material properties of an object. Using the reciprocity principal, this is equivalent to shooting a ray of light from the viewpoint to the points A and B, and having that light specularly reflect out to the environment. The specularly reflected rays will result in a distribution around the direction of mirror reflection as shown in Figure 4. The blue and orange blobs on the upper left region of the circle represent the probability density function (PDF) of the reflected ray along those directions. The total probability of a specular highlight can be computed by the sum of the individual PDFs, as shown by the purple curve. Thus, following the reciprocity principal, if we were to place a light source in the direction where the purple curve has the largest value, we would get the best highlights at both the points A and B for the given view position.

We extend the above ideas to define a light placement function $P(\overrightarrow{l})$ that models the appropriateness of placing a light in the direction $\overrightarrow{l}$. Such a light placement function should include contributions from both specular as well as diffuse illumination. Let $\mathscr{P}$ be the set of surface patches for an object. Let $\overrightarrow{v}$ be the view vector, $\overrightarrow{l}$ be the light direction, and $\overrightarrow{h}$ be the halfway unit vector along the direction $\overrightarrow{l} + \overrightarrow{v}$. Further, let κ_i be the mean curvature, $\overrightarrow{n_i}$ be the normal vector, and $\vec{R}$ be the reflection of viewing direction $\vec{v}$ about the normal $\overrightarrow{n_i}$ at a vertex i on the surface. We define the specular weight function S for the vertex i with a shininess s as: $S(i, \overrightarrow{l}) = | \kappa_i | (\overrightarrow{n_i} \cdot \overrightarrow{h})^s$. Given a view direction, we compute $S(i, \overrightarrow{l})$ for each vertex i and for a set of uniformly-distributed light directions $\overrightarrow{l}$. In our implementation we use $12K$ uniformly-distributed directions $\overrightarrow{l}$.

However, the use of specular highlights alone is not desirable, as shown by Gumhold [19]. We have designed the diffuse lighting component of the light placement function to adapt to the local curvature on a patch-by-patch basis. Figure 3(a) shows curvature distribution over the Skull model. We define the *curvature intensity* c_i at a vertex i to be its normalized mean curvature,

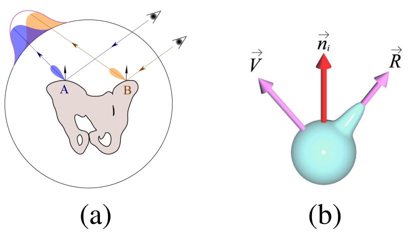


Fig. 4. *(a) A view-dependent weight function for each surface point is added to the light placement function defined in the directional space (shown here by the large circle). The light placement function models the appropriateness of placing a light along a direction. (b) Specular weight function $S(i, \vec{l})$ is defined as the fall-off function around the reflection vector $\vec{R}$ weighted by curvature.*

i.e. $c_i = (\kappa_i - \kappa_{min})/(\kappa_{max} - \kappa_{min})$, where κ_i is the mean curvature at vertex i, and κ_{max} and κ_{min} are the maximum and minimum values of the mean curvature among all the vertices of the input mesh, respectively. For a vertex i with normal vector $\overrightarrow{n_i}$, let $\mathscr{D}$ be the set of light directions whose diffuse color is same as the curvature intensity c_i: $\mathscr{D} = \{\overrightarrow{d} \mid \overrightarrow{d} \cdot \overrightarrow{n_i} = c_i\}$. We define the diffuse weight function $D(i, \overrightarrow{l})$ for vertex i in the direction of $\overrightarrow{l}$ such that the diffuse illumination at vertex i is similar to the curvature intensity c_i. We compute it as the upper envelope (maximum) of the dot product between $\overrightarrow{l}$ and all $\overrightarrow{d} \in \mathscr{D}$ as: $D(i, \overrightarrow{l}) = \underset{\overrightarrow{d} \in \mathscr{D}}{Max}\, \overrightarrow{l} \cdot \overrightarrow{d}$, as seen in Figure 5.

The light placement function can be computed as the sum of specular and diffuse weight functions over all surface points. For any light direction $\overrightarrow{l}$ the value of the light placement function $P(\overrightarrow{l})$ along that direction is given by: $P(\overrightarrow{l}) = \sum_i (S(i, \overrightarrow{l}) + D(i, \overrightarrow{l}))$.

B. Light Placement and Assignment

We select the best m lights $\mathscr{L} = \{l_1, l_2, ..., l_m\}$ by using the light placement function $P(\overrightarrow{l})$, as follows. We identify the light direction $\overrightarrow{l}$ that maximizes $P(\overrightarrow{l})$. We select this to be the direction of the first light l_1. We then identify the patches which will be lit by the light l_1. For any light $l_k \in \mathscr{L}$ and patch $p \in \mathscr{P}$, let $\mathscr{S}_p$ be the set of points that are on p and let $I_i(l_k)$ be the illuminated intensity at vertex $i \in \mathscr{S}_p$ due to light l_k. We define a function $E(p, l_k)$ that measures the similarity of the illuminated intensity $I_i(l_k)$ for vertices i in the patch p to its curvature intensity as: $E(p, l_k) = \sum_{i \in \mathscr{S}_p} (I_i(l_k) - c_i)^2$.

For the first light l_1, we assign l_1 to a patch $p \in \mathscr{P}$ whenever $E(p, l_1)$ is less than a threshold τ (currently we use $\tau = 0.15$), i.e. $\mathscr{M}(p) = l_1$. We deduct the contributions of the vertices in the patches lit by this light l_1 from the light placement function. We repeat this process until m lights are selected. For each unlit patch, the light l_k which minimizes $E(p, l_k)$ is assigned to p : $\mathscr{M}(p) = \arg\min_{l_k \in \mathscr{L}} E(p, l_k)$. Figures 7 (a)–(c) show the lighting with one, two, and eight lights. Patches that are not lit are shown dark without any blending with the neighboring patches.

C. Illumination

Our Light Collages framework allows patches to be assigned different lights even though the patches are adjacent. A straightforward implementation of this idea might result in sharp visual discontinuities across patch boundaries that are lit differently. Such shading discontinuities are disconcerting especially when they occur in absence of shape discontinuities. To alleviate such visual artifacts we blend illumination from neighboring patches. As mentioned earlier, every vertex i in a patch p_j is illuminated by light $\mathscr{M}(p_j)$. The blended illumination at a vertex i is a weighted sum of illuminations from the primary lights for all the patches $\mathscr{N}_j$ that are next to p_j: $\mathscr{N}_j = \{p_k \mid \partial p_j \cap \partial p_k \neq \emptyset\}$, where ∂p_j denotes the boundary of patch p_j. Let the primary light for patch $p_k \in N_j$ be given by $l_k = \mathscr{M}(p_k)$. Let the weight of vertex i with respect to the primary light of patch p_k be based on the distance function $d()$ of vertex i from the boundary ∂p_k and be given by: $w_{ik} \propto \frac{1}{1+d(i,\partial p_k)}$.

We define the distance $d(i, \partial p_j)$ to be zero for a vertex inside or on the boundary of the patch p_j. Therefore, the weight of a vertex i inside patch p_j is $w_{ij} = 1$. The distribution of the blending weights at vertices around a patch is shown in Figure 7(d).

A simple weighted sum of illuminations may increase the overall brightness which tends to result in diminishing the visual discriminability amongst object features. To balance the rendering brightness we normalize the illumination with the blending weights for a given vertex.

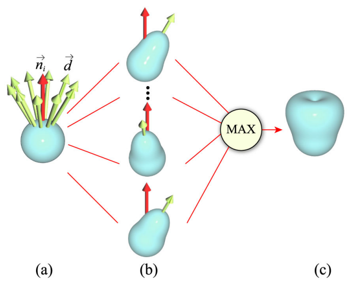


Fig. 5. *Computation of diffuse weight function for a vertex with normal $\vec{n}_i$ and curvature intensity c_i: First, (a) we define the set of light directions $\vec{d} \in \mathscr{D}$ for which $\vec{n}_i \cdot \vec{d} = c_i$. These directions $\vec{d}$ are shown by green arrows. Figure (b) shows the cosine fall-off for each $\vec{d} \in \mathscr{D}$. (c) The diffuse weight function $D(i, \vec{l})$ is the upper envelope (maximum) of the functions shown in Figure (b).*

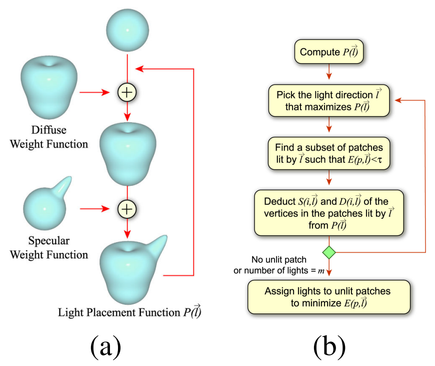

Diffuse Weight Function
Specular Weight Function
Light Placement Function $P(\vec{l})$
Compute $P(\vec{l})$
Pick the light direction $\vec{l}$ that maximizes $P(\vec{l})$
Find a subset of patches lit by $\vec{l}$ such that $E(p,\vec{l})<\tau$
Deduct $S(i,\vec{l})$ and $D(i,\vec{l})$ of the vertices in the patches lit by $\vec{l}$ from $P(\vec{l})$
No unlit patch or number of lights = m
Assign lights to unlit patches to minimize $E(p,\vec{l})$
(a)
(b)

Fig. 6. *The light placement function $P(\vec{l})$ is computed in Figure (a) by adding diffuse and specular weight functions. Figure (b) shows the flowchart of the process for light placement and assignment.*

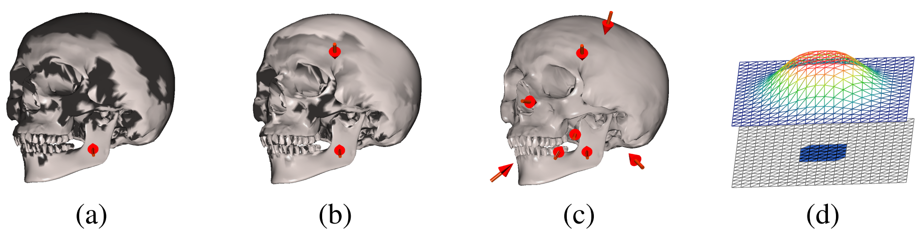


Fig. 7. *Partial surface lighting with (a) first light, and (b) first two lights, and (c) eight lights. The red arrows show the light directions. The dark regions in (a) and (b) are the patches not lit by the current partial lights. No blending is used here. Figure (d) shows the weights for blending illumination. The lower mesh shows the patch (in blue) and its neighborhood. For each vertex of the lower mesh, the vertex of the upper mesh vertically above it represents the blending weight. Note that the weight stays constant over the patch and then gradually falls off.*

Let the illumination at vertex i due to light l_k be given by $I_i(l_k)$ as defined in Section V-B. Then, the final illumination formula for a vertex i in patch p_j with neighbors $\mathcal{N}_j$ is given by:

$$\overline{I_i} = \frac{\sum_k w_{ik} I_i(l_k)}{\sum_k w_{ik}}, p_k \in \mathcal{N}_j, l_k = \mathcal{M}(p_k)$$

VI. Feature Enhancement

A. Silhouette Enhancement

Usually silhouettes characterize large depth discontinuities. Therefore, a well-defined silhouette makes an object easier to comprehend by making it more easily distinguishable from its surroundings. Cinematographers use backlights for separating the foreground from the background. They traditionally place backlights behind an object to generate a thin rim of light around its silhouette. Backlights are also called rim, hair, or separation lights. In particular, the lights at the three-quarters-back position are called as kicker lights [10].

To distinguish an object from its background, we produce a dark silhouette for a bright background and a bright silhouette for a dark background. We use a simple fall-off formula weighted by $\omega_s = (1 - \overrightarrow{n_i} \cdot \overrightarrow{v})^u$, for adding an additional silhouette light at vertex i with normal $\overrightarrow{n_i}$ and view direction $\overrightarrow{v}$. The results of incorporating black silhouette lighting appear in Figure 8. We compute the silhouette-enhanced illumination as the linear blend of the silhouette lighting H_i weighted by ω_s and the existing illumination: $(1-\omega_s)I_i + \omega_s H_i$.

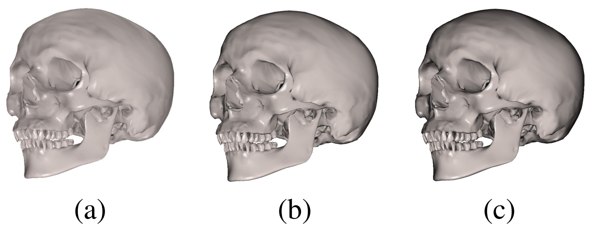


Fig. 8. *Rendering (a) without, and (b), (c) with silhouette lighting. $(1-\vec{n}_i \cdot \vec{v})^u$ is used as the silhouette light's weight factor. (b) and (c) show silhouette enhancement with $u = 4$ and $u = 2$ respectively.*

B. Proximity Shadows

Perception of depth through carefully placed shadows is an important visual cue for comprehending the spatial relationships between objects. As an example, it may be difficult to distinguish two surface patches if they have similar illumination but different distances from the viewer and partially overlap in space as seen by the viewer. However, if the front patch casts a visible shadow on the other patch, their spatial relationship immediately becomes clear. Such pairs of visible patches result in a depth discontinuity that usually occurs along one or more silhouette curves as shown in Figure 9(b). We use proximity shadows to show the relative distance between the two overlapping patches if the eye-space distance between them is within a predefined threshold.

To compute proximity shadows, we first identify the depth discontinuity curves by comparing the value of each pixel in the depth map with its neighbors. We then generate a shadow light direction for each depth discontinuity curve by using the depth gradient. The shadow light direction is determined by rotating the direction vector

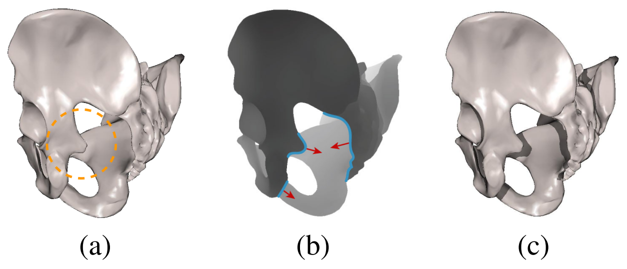


Fig. 9. *Figure (a) shows a pelvis model rendered without proximity shadows. The illumination provides only a weak depth cue for the two overlapping regions inside the circle. Figure (b) shows depth discontinuity curves, where adjacent pixel depths differ by more than a threshold, in blue. The arrows show the average gradients of discontinuity curves. Figure (c) shows the proximity shadows cast by the discontinuity curves in (b).*

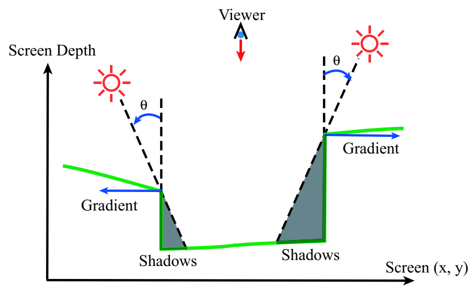


Fig. 10. *The placement of a light for proximity shadow: At each depth discontinuity curve of the depth map, a light for the proximity shadow is placed by rotating a vector to the viewer by an angle θ along the direction of the local gradient.*

to the viewer by a small angle θ towards the average depth-gradient direction as shown in Figure 10. Finally we use the shadow light direction in a shadow map to cast proximity shadow for the depth discontinuity curve.

While casting proximity shadows, we have to be aware that a narrow region might cause a problem if it has depth discontinuities on multiple sides. If we cast shadows of this region in each direction, it can produce a somewhat disconcerting effect as shown in Figure 11(b). For such situations one can use any heuristic that consistently picks one side of the region over the others. Examples of such heuristics may include picking the side of the discontinuity region that is on the left and the top, or pick the side of the discontinuity region that has more surface points on the discontinuity curve (refer Figure 11(c)).

VII. Efficient Computation

The light placement and light assignment stages are the most time consuming in our lighting design pipeline. In this section, we discuss how to speed up the overall system by efficiently computing and updating the light placement function using the spherical-harmonic-basis representation. The Light Collages process described in Section V takes a few hundred seconds for a model with tens of thousands of vertices. It is reasonable running time for one-time image generation, but not fast enough for interactive visualization or generation of a large number of images. Also, we might want to store the precomputed light placement functions for interactive rendering. In that case the current representation will need large amounts of storage.

Spherical harmonics (SH) can encode a function defined over a sphere with orthonormal basis functions. Spherical harmonics can represent any function with representational accuracy related to the number of coefficients used. Since our light placement functions and weight functions are defined on a sphere, we can encode them using spherical harmonics. Moreover, since our light placement function and weight functions are low frequency, we can represent them with a small number of spherical harmonic coefficients resulting in efficient storage and computation.

A. Spherical Harmonics Background

The spherical-harmonic (SH) basis functions with the parametrization $(x,y,z) = (sin\theta cos\varphi, sin\theta sin\varphi, cos\theta)$ are defined as

$$y_l^m(\theta,\varphi) = \begin{cases} \sqrt{2}K_l^m cos(m\varphi)P_l^m(cos\theta), & m>0 \\ \sqrt{2}K_l^m sin(-m\varphi)P_l^{-m}(cos\theta), & m<0 \\ \sqrt{2}K_l^0 P_l^0(cos\theta), & m=0 \end{cases}$$

where P_l^m are the associated Legendre polynomials and K_l^m are defined as: $K_l^m = \sqrt{\frac{(2l+1)(l-|m|)!}{4\pi(l+|m|)!}}$. We can project a scalar function f defined on the sphere into its SH coefficients h, through the integral: $h(m,l) = \int f(s)y_l^m(s)ds$. We approximate the function f with these coefficients h by using n SH bands: $\tilde{f}(s) = \sum_{l=0}^{n-1}\sum_{m=-l}^{l} h(m,l)y_l^m(s)$.

The rotational-invariance property of spherical harmonics enables us to rotate a function by multiplying a rotation matrix to its vector of SH coefficients. We use Blanco *et al.*'s method [31] for fast rotations of spherical harmonic representations. Their method incrementally computes a rotation matrix of real spherical harmonics by using the recursive relations between matrix components for adjacent bands.

B. SH-Based Light Placement and Assignment

We can efficiently represent and compute the light placement function by using spherical harmonics. Let $h(l,m)$ be the spherical-harmonic coefficients for representing the light placement function P: $h(l,m) = \int P(s)y_l^m(s)ds$. We approximate the light placement function P with the coefficients as: $\tilde{P}(s) = \sum_{l=0}^{n-1}\sum_{m=-l}^{l} h_i(m,l)y_l^m(s)$.

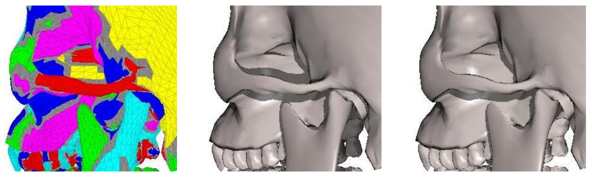


Fig. 11. *Avoiding conflicts in proximity shadows: (a) Depth discontinuity curves arise along both sides of the upper cheek bone. (b) The discontinuity curves result in proximity shadows on both sides of bone that might appear disconcerting. (c) This can be fixed by eliminating one of the two proximity shadows.*

Recall from Section V that the overall light placement function is a sum of specular and diffuse weight functions from each vertex. We observe that given two vertices i and j with the same curvature but different normals, the weight functions (diffuse and specular) of vertex i can be computed by rotating the corresponding weight functions (diffuse and specular) of vertex j. Therefore, we can pre-compute the spherical-harmonic representations of the weight functions for each curvature value and simply rotate them according to the per-vertex normals. This is significantly more efficient than repeatedly projecting every vertex's specular and diffuse weight functions into spherical-harmonic coefficients.

First, we pre-compute the specular and diffuse weight functions for a canonical normal $\overrightarrow{n_0}$ for each curvature intensity c. We currently sample c uniformly in the range 0 to 1. Let the specular and diffuse weight functions for $\overrightarrow{n_0}$ be represented by spherical-harmonic coefficients $f_0(l,m,c)$ and $g_0(l,m,c)$, respectively:

$$f_0(l,m,c) = \int S(0,s) y_l^m(s) ds$$

$$g_0(l,m,c) = \int D(0,s) y_l^m(s) ds$$

Second, for each vertex i, we find the pre-computed specular and diffuse weight functions whose curvature value is closest to the vertex's curvature value c_i. We rotate these weight functions to get the weight functions of the vertex i. Let $R_{0\to i}$ be the spherical-harmonic rotation matrix that is equivalent to the rotation of $\overrightarrow{n_0}$ to the normal $\overrightarrow{n_i}$ of vertex i. Then we compute the spherical-harmonic coefficients $f_i(l,m)$ and $g_i(l,m)$ as:

$$f_i(l,m) = R_{0\to i} f_0(l,m,c_i)$$

$$g_i(l,m) = R_{0\to i} g_0(l,m,c_i)$$

We compute the spherical-harmonic coefficients of the light placement function by adding the $f_i(l,m)$ and $g_i(l,m)$ for all vertices:

$$h(l,m) = \sum_i (f_i(l,m) + g_i(l,m))$$

In Section V-B, we discussed an iterative scheme for identifying the best light source directions using the light placement function. According to this scheme we identify a light l_i and assign it to patches p_j best lit by it. We then deduct from the light placement function, the contributions from the weight functions of all the vertices in the patches p_j lit by light l_i. We do this directly with the spherical-harmonic coefficients of the light placement and per-vertex weight functions. Since the spherical harmonic functions define an orthonormal basis, we simply subtract the spherical-harmonic coefficients of the per-vertex weight functions from the spherical-harmonic coefficients of the light placement function.

VIII. Minimality of the Light Sources

The choice of an appropriate number of discrepant light sources is important and requires trade-offs between quality and efficiency. If we arbitrarily choose the number of discrepant light sources, our rendered image may be of an undetermined quality for different objects. If we choose too many lights, we will pay for the extra run-time lighting costs and if we choose too few lights we may not have an adequate number of lights to show the fine geometric detail.

The Light Collages framework selects light sources incrementally. We examine the incremental improvement in the quality of the image by adding an extra light. If the image improvement (measured as the root-mean-square difference) is small enough we can stop adding light sources. In this paper, we stop adding light sources when fewer than 2% of the screen pixels change by less than 2% of their color range. For example, if we use a screen with a 1024×768 resolution and 8-bit colors, less than $16K$ pixels are allowed to vary by less than 5 out of 256 color values. In this case, the RMSD threshold works out to be 2.8×10^{-3}. The graph in Figure 12 shows that 8 lights suffice for the Skull model. We note that in Figure 12 the image differences are nearly independent of the number of spherical-harmonic coefficients. Therefore, efficient computation by using low-band spherical-harmonic representation is quite appropriate for determining the number of light sources.

Discrepant lighting by more lights increases the geometric detail that we can see. This improvement diminishes for a given geometric level of detail after a certain number of lights have been added (see Figure 13(d)).

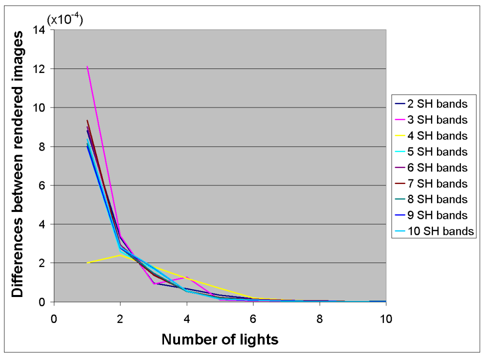
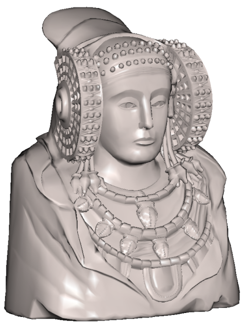
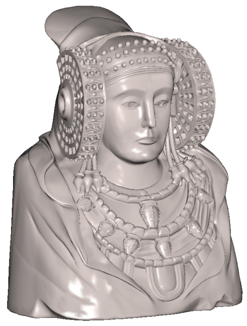


Fig. 12. *The image differences with different numbers of light sources (Skull Model)*

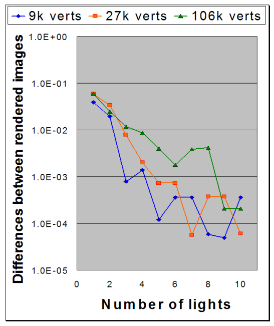
(a) $9K$ verts



(b) $28Kverts$



(c) $106K$ verts



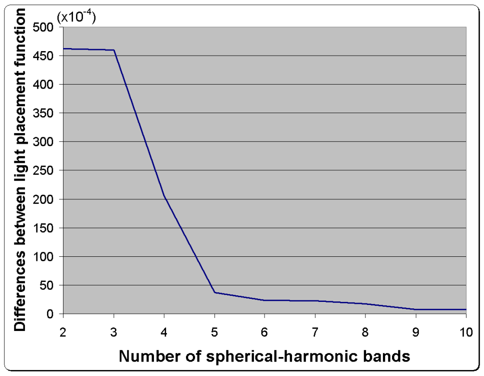
(d)

Fig. 13. *Minimality of light sources for varying levels of detail in geometry: Images (a), (b), and (c) show Light Collages rendering of Dama De Elche model represented with 9K, 27K, and 106K vertices. Image (d) shows the image differences with different numbers of light sources for each level of detail. A less-detailed mesh representation of an object generally requires fewer discrepant lights than a more detailed one. Meshes with 9K, 27K, and 106K vertices select 2, 3, and 5 lights with our system.*

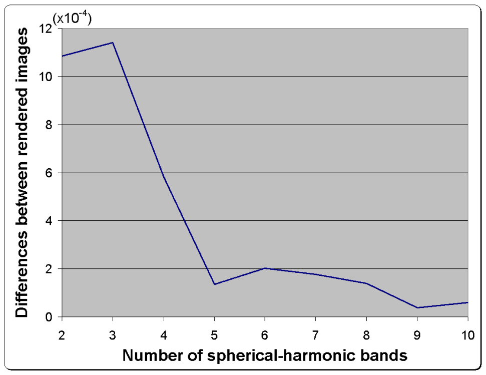
This leads us to believe that it should be possible to relate the level of detail for lighting with the level of detail for geometry. Thus, less-detailed lighting should suffice for less-detailed geometry whereas higher-detailed geometry should require higher-detailed lighting. Just as the geometric level-of-detail systems manage the complexity of geometry based on parameters such as the viewer position relative to the object one should manage the lighting level of detail based on the geometry and viewing parameters. Figures 13 (a), (b), and (c) show Light Collages rendering of the Dama De Elche model at different geometric levels of detail. Figure 13(d) shows that a higher level of detail in geometry requires more lights than a less-detailed geometry.

IX. Results

Figures 16 and 17 show the visualization results using our system. The manuscript dataset used in Figure 16 was provided to us by Paul Debevec at USC and scanned by XYZ RGB Inc. The manuscript is a 177×163mm page from a 15th century "Book of Hours" produced near Rouen in France. The scanned manuscript has an accuracy of $100\mu m$ horizontally and vertically, and $3\mu m$ along the depth. At a depth resolution of $3\mu m$, the scan is detailed enough to lift the impressions of the ink. Naive consistent lighting as shown in Figures 16 (a) and (b) fails to capture the fine details of the characters and the subtle variations and wrinkles in the manuscript. Figure 16(c) nicely shows these subtle variations in the geometry with our geometry-dependent discrepant lighting. We have used the same lighting models and material properties for generating all the three images. As you can see in (a), the specular highlight from consistent lighting will sometimes cause large bright areas on flat regions, while highlights from our method (c) are only on highly curved regions. This helps elucidate geometry details. In Figure 17(a) we show the result from lighting the Pelvis model by consistent lighting with 4 lights, and (b)–(d) show the results by Light Collages. The proximity shadow cast by the sacrum and the right coxal bone in Figure 17(d) nicely illustrates the depth relationship between adjacent regions of the pelvis.



(a)



(b)

Fig. 14. *We show the difference from using spherical harmonics compared to direct evaluation over* $12K$ *uniformly distributed light directions for the* $33K$ *vertex Skull model. Figure (a) shows the root-mean-square difference between direct and spherical-harmonic evaluation of the normalized light placement function. Figure (b) shows the root-mean-square difference between images resulting from lighting design with direct computation and with spherical harmonics.*

We have reduced the computation time for each vertex to be proportional to the number of spherical-harmonic (SH) coefficients instead of the number of directional samples. In this paper we have used 12518 ($\sim 12K$) directions. The important question that remains to be addressed is how many SH coefficients are necessary to give us an acceptable level of accuracy. To address this, we compared the accuracy of the lighting design process with and without spherical harmonic representations. To compare the accuracy in representing the light placement function, we first normalized the light placement function to be in the range 0 to 1. Then, for each of the approximately $12K$ light directions we computed the difference between direct evaluation and the spherical-harmonic evaluation, and used these to compute the overall root-mean-squared difference. This is shown in Figure 14(a) over an increasing number of SH bands for the Skull model. The number of SH coefficients used is the square of the number of bands used. In Figure 14(b) we show the root-mean-square difference between images of the Skull model rendered using lighting design with and without spherical harmonics.

As you can see in Figures 14 and 15, the error is reduced significantly when the number of spherical harmonic bands is five or greater. We report the timings for light placement and assignment in Table I. These times are for a Pentium IV, 1.5 GHz system with 1GB RAM. As one can see, the spherical-harmonic method with 5 bands is almost 20 times faster than the direct computation. Further, since we only need to store 25 spherical-harmonic coefficients per vertex instead of over $12K$ directional samples, our spherical-harmonic-based lighting design approach reduces the required memory by a factor of over 500.

TABLE I
RUN TIMES FOR LIGHT PLACEMENT AND ASSIGNMENT

Model	Skull (33K verts)	Pelvis (17K verts)
SH Bands	**Time (sec)**	**Time (sec)**
2	4.04	2.54
3	5.30	3.41
4	8.05	5.19
5	13.42	7.11
6	19.85	12.05
7	29.58	16.62
8	42.81	23.63
9	60.36	35.51
10	81.30	43.72
Direct Computation	234.17	138.82

X. CONCLUSIONS AND FUTURE WORK

We have introduced the concept of geometry-dependent lighting which allows discrepant lights dependent on local geometry to only affect local regions. Our Light Collages system uses geometry-dependent lighting for automatic lighting design for effective visualization of scientific datasets. Our method relies on using multiple light sources that can be used for accurate local lighting on surfaces, with possible global inconsistencies.

The human visual system is remarkably adept at inferring shape from largely local cues and recent research [3] suggests that inconsistencies in illumination may not be resolved at a low level. However, it is also believed that the human visual system has a strong preference

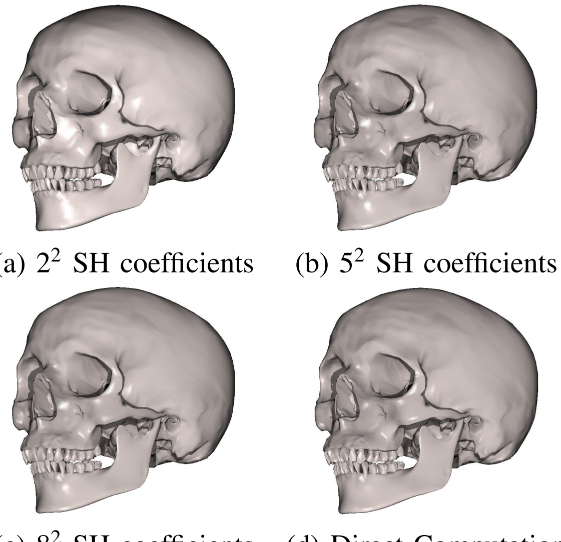

(a) 2^2 SH coefficients (b) 5^2 SH coefficients
(c) 8^2 SH coefficients (d) Direct Computation

Fig. 15. *Lighting for Skull: (a)–(c) show Light Collages rendering with various spherical harmonic coefficients, and (d) shows the result with direct computation.*

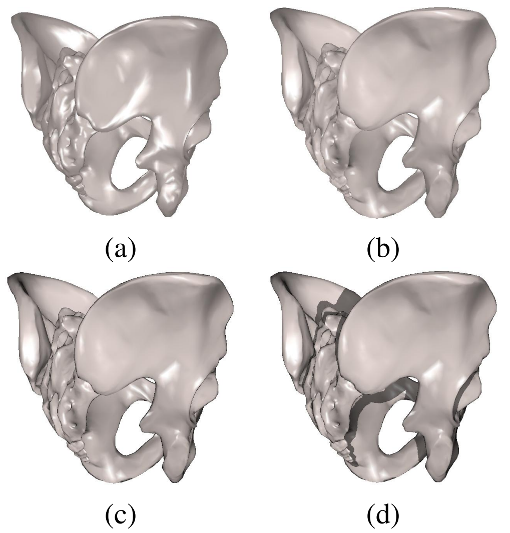

(a) (b)
(c) (d)

Fig. 17. *Lighting design for the Pelvis model. We used 25 SH coefficients for generating images (b)–(d). (a) shows consistent rendering with four lights at the front four vertices of a cube, and (b) shows Light Collages rendering by 4 lights. In image (c), we further added silhouette lighting, and in (d) we further added proximity shadows.*

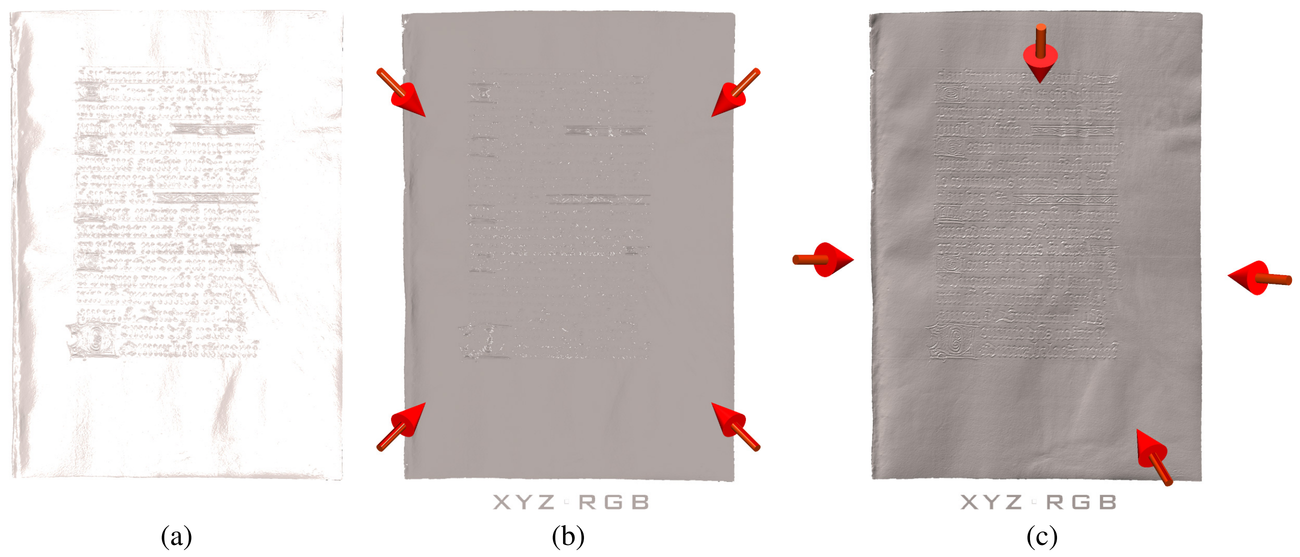


Fig. 16. *Light Collages for the Manuscript: (a) Rendered by one consistent light placed along the view direction, (b) Rendered by four consistent lights arranged at the front four corners of a cube, and (c) Rendered by Light Collages with four lights using* 25 *SH coefficients.*

for a single illumination from above which if violated may lead to incorrect perception of shape [32]. Elder *et al.* [33] reconcile these by suggesting that for simple objects and scenes the low-level human visual system might expect and process consistent illumination but for more complex scenes and objects, with multiple light sources and inter-reflections, discrepancies in illuminations might require higher-level processing. We find it interesting that our results on minimality of the number of light sources derived by our Light Collages system, show an increase in the number of discrepant lights with increasing geometric detail. However, in the absence of an adequate computational model that can reconcile these opposing points of view, it is desirable to allow a user to modify the light source directions for regions in which a system such as Light Collages, causes ambiguous or incorrect shape interpretation.

We have shown how our method can incorporate silhouette lighting as well as proximity shadows to further elucidate the local structure of the scientific datasets. We believe our method is a good start in improving the visualization while retaining the look and feel of traditional 3D graphics rendering. In addition to the visual appearance, interactivity is essential for the perception of 3D shapes. We use spherical-harmonics-based representations to efficiently compute the light placement function for use in a real-time system. Our current run times could be further enhanced by using the vertex shaders on modern graphics processors. We have also presented a method to optimize the number of light sources needed for generating images without loss of image quality. This minimality of the light sources depends on the geometry of various models as well as the geometric level of detail of a single model.

In this paper we have assumed that the lights are directional. Generalizing our approach to point light sources or perhaps even area light sources would be an interesting direction for future work. Our current work does not take into account variations in color or material properties such as albedo and this should be useful to consider in the future. In addition to lighting design for a single object, automatically designing lighting environments for a scene with multiple objects is a highly promising area for future research.

ACKNOWLEDGEMENTS

We would like to thank Patrick Cavanagh, Tony DeRose, and Martin Reddy for inspiring us to explore lighting design with multiple and discrepant lights. David Jacobs, David Mount, and Youngmin Kim have helped in significantly improving the presentation of ideas in this paper. The anonymous referees gave us several valuable suggestions that have greatly improved this paper. This work has been supported in part by the NSF grants: IIS 00-81847, CCF 04-29753, and CNS 04-03313.

REFERENCES

[1] E. H. Gombrich, *The Heritage of Appelles*. Oxford: Phaidon Press, 1976.
[2] P. Cavanagh, “Pictorial art and vision,” *MIT Encyclopedia of the Cognitive Sciences*, pp. 644–646, 1999.
[3] Y. Ostrovsky, P. Cavanagh, and P. Sinha, “Perceiving illumination inconsistencies in scenes,” *Perception*, 2005, in press.
[4] P. Cavanagh, “The artist as neuroscientist,” *Nature*, vol. 434, pp. 301–307, 2005.
[5] R. Basri and D. Jacobs, “Lambertian reflectance and linear subspaces,” in *Proceedings of the Eighth International Conference On Computer Vision*, July 9–12 2001, pp. 383–390.

[6] R. Ramamoorthi and P. Hanrahan, “A signal-processing framework for inverse rendering,” in *ACM SIGGRAPH*, 2001, pp. 117–128.
[7] D. Akers, F. Losasso, J. Klingner, M. Agrawala, J. Rick, and P. Hanrahan, “Conveying shape and features with image-based relighting,” in *IEEE Visualization*, 2003, pp. 349–354.
[8] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin, and M. Cohen, “Interactive digital photomontage,” *ACM Transactions on Graphics (SIGGRAPH 2004)*, vol. 23, no. 3, pp. 294–302, 2004.
[9] C. H. Lee, X. Hao, and A. Varshney, “Light collages: Lighting design for effective visualization,” in *IEEE Visualization*, 2004, pp. 281–288.
[10] J. Kahrs, S. Calahan, D. Carson, and S. Poster, “Pixel cinematography: A lighting approach for computer graphics,” in *ACM SIGGRAPH Course Notes*, 1996.
[11] J. Marks, B. Andalman, P. A. Beardsley, W. Freeman, S. Gibson, J. Hodgins, T. Kang, B. Mirtich, H. Pfister, W. Ruml, K. Ryall, J. Seims, and S. Shieber, “Design galleries: a general approach to setting parameters for computer graphics and animation,” in *ACM SIGGRAPH*, 1997, pp. 389–400.
[12] M. Halle and J. Meng, “LightKit: A lighting system for effective visualization,” in *IEEE Visualization*, 2003, pp. 363–370.
[13] A. C. Costa, A. A. de Sousa, and F. N. Ferreira, “Lighting design: A goal based approach using optimization,” in *Proceedings of Rendering Techniques*, 1999, pp. 317–328.
[14] J. K. Kawai, J. S. Painter, and M. F. Cohen, “Radiooptimization - Goal Based Rendering,” in *ACM SIGGRAPH*, 1993, pp. 147–154.
[15] F. Pellacini, P. Tole, and D. P. Greenberg, “A user interface for interactive cinematic shadow design,” in *ACM SIGGRAPH*, 2002, pp. 563–566.
[16] P. Poulin and A. Fournier, “Lights from highlights and shadows,” in *Symposium on Interactive 3D Graphics*, 1992, pp. 31–38.
[17] C. Schoeneman, J. Dorsey, B. Smits, J. Arvo, and D. Greenberg, “Painting with light,” in *ACM SIGGRAPH*, 1993, pp. 143–146.
[18] R. Shacked and D. Lischinski, “Automatic lighting design using a perceptual quality metric,” *Computer Graphics Forum (Eurographics 2001)*, vol. 20, no. 3, pp. 215–226, 2001.
[19] S. Gumhold, “Maximum entropy light source placement,” in *IEEE Visualization*, 2002, pp. 275–282.
[20] A. Gooch, B. Gooch, P. Shirley, and E. Cohen, “A non-photorealistic lighting model for automatic technical illustration,” in *ACM SIGGRAPH*, 1998, pp. 447–452.
[21] M. Sousa, F. Samavati, and M. Brunn, “Depicting shape features with directional strokes and spotlighting,” in *Proceedings of Computer Graphics International*, 2004, pp. 214–221.
[22] J. Hamel, “A new lighting model for computer generated line drawings,” Ph.D. dissertation, Otto-von-Guericke-Universität Magdeburg, Germany, 2000.
[23] P.-P. Sloan, W. Martin, A. Gooch, and B. Gooch, “The Lit Sphere: A model for capturing NPR shading from art,” in *Proceedings of Graphics Interface*, 2001, pp. 143–150.
[24] S. Anderson and M. Levoy, “Unwrapping and visualizing cuneiform tablets,” *IEEE Computer Graphics and Applications*, vol. 22, no. 6, pp. 82–88, 2002.
[25] G. Miller, “Efficient algorithms for local and global accessibility shading,” in *ACM SIGGRAPH*, 1994, pp. 319–326.
[26] A. J. Stewart, “Vicinity shading for enhanced perception of volumetric data,” in *IEEE Visualization*, 2003, pp. 355–362.
[27] T. Strothotte and S. Schlechtweg, *Non-Photorealistic Computer Graphics: Modeling, Rendering, and Animation.* San Francisco: Morgan Kaufmann, 2002.
[28] A. Girshick, V. Interrante, S. Haker, and T. Lemoine, “Line direction matters: an argument for the use of principal directions in 3D line drawings,” in *Proceedings of the First International Symposium on Non Photorealistic Animation and Rendering*, 2000, pp. 43–52.
[29] G. Taubin, “Estimating the tensor of curvature of a surface from a polyhedral approximation,” in *Fifth International Conference on Computer Vision*, 1995, pp. 902–907.
[30] A. P. Mangan and R. T. Whitaker, “Partitioning 3D surface meshes using watershed segmentation,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 5, no. 4, pp. 308–321, 1999.
[31] M. A. Blanco, M. Flórez, and M. Bermejo, “Evaluation of the rotation matrices in the basis of real spherical harmonics,” *Journal of Molecular Structure (Theochem)*, vol. 419, pp. 19–27, 1997.
[32] V. Ramachandran, “Perception of shape from shading,” *Nature*, vol. 331, no. 6152, pp. 163–166, 1988.
[33] J. Elder, S. Trithart, G. Pintilie, and D. MacLean, “Rapid processing of cast and attached shadows,” *Perception*, vol. 33, no. 11, pp. 1319–1338, 2004.

Chang Ha Lee is currently a Ph.D. student of Computer Science at the University of Maryland at College Park. He received the BS and MS degrees in Computer Science from the Seoul National University, Seoul, Korea, in 1995 and 1997. His research interests include 3D computer graphics, scientific visualization, lighting, perception, and molecular graphics.

Xuejun Hao received the BS degree in physics from the Peking University in 1990, the MA degree in physics and the MS degree in computer science from the State University of New York at Stony Brook in 1996 and 2000, and the PhD degree in computer science from the University of Maryland, College Park, in 2004. He is currently a postdoctoral fellow at Columbia University. His research interests are in interactive 3D graphics, scientific visualization, and medical imaging.

Amitabh Varshney is currently an Associate Professor of Computer Science at the University of Maryland at College Park. He received his B.Tech. in Computer Science from the Indian Institute of Technology, Delhi in 1989 and his M.S. and Ph.D. degrees from the University of North Carolina at Chapel Hill in 1991 and 1994. His current research interests are in interactive 3D graphics, scientific visualization, geometric modeling, and molecular graphics. He received the National Science Foundations CAREER award in 1995 and the IEEE Visualization Technical Achievement Award in 2004.