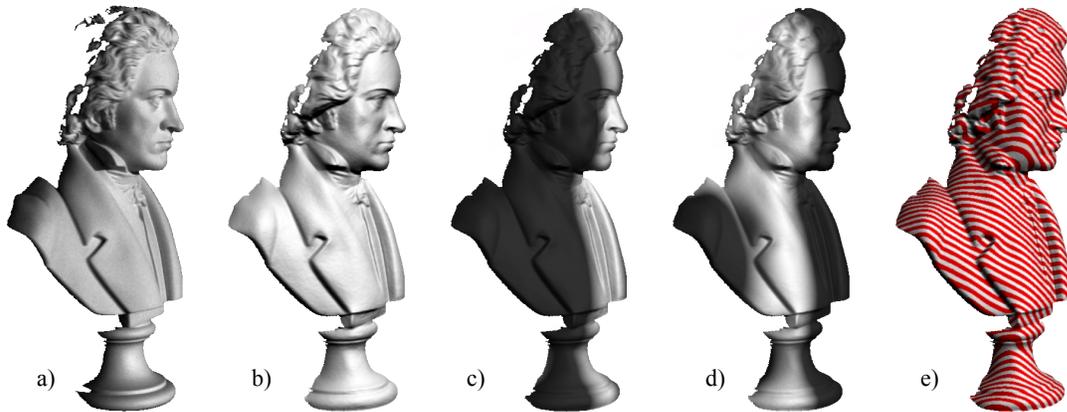


# Digital Inspection: An Interactive Stage for Viewing Surface Details

Daniel G. Aliaga

Department of Computer Science at Purdue University



*Figure 1. Viewing Surface Details. (a) A video-frame of an object under a single light source. (b) A synthesized and re-lit rendering with a different virtual light. (c-d) View using our depth-light visualization tool. (e) Rendering of object using our iso-distance curves from a chosen reference plane.*

## Abstract

In a wide range of applications, we often wish to quickly inspect and visualize historically significant and highly detailed objects. For such scientific illustration applications, often the focus is on high-frequency surface details and on conveying important shape and feature information. In our work, we provide a complete system to visualize objects on the spot. Our approach uses photometric and geometric processing, combined with a set of visualization methods tuned to the interactive inspection and analysis of objects. Highly accurate models are acquired in about 30 seconds using an uncalibrated setup, obtaining both detailed surface geometry and detailed surface normal information. Subsequently, captured objects are visually tracked enabling hand-held manipulation and visualization. In this paper, we demonstrate our system using several real-world objects.

**CR Categories:** I.3 [Computer Graphics], I.3.3 [Picture/Image Generation], I.3.7 [Three-dimensional Graphics and Realism], I.4.1 [Digitization and Image Capture].

**Keywords:** visualization, surface normals, surface heights, structured light, relighting, scientific illustrations.

## 1. Introduction

In many application scenarios, a user, untrained in computer graphics, wishes to quickly and instantly *inspect* a handheld object being observed (e.g., re-light, exaggerate surface details)

and to extract preliminary information about the object before committing to more time consuming tasks. In particular, we seek an approach that meets the design objectives of being

- low-cost, rapid, and easy to use,
- free of any cumbersome calibration tasks, and
- able to capture and inspect highly-detailed handheld objects.

For example, an artist, archaeologist, or historian with a large number of objects or fragments to rapidly inspect might like to digitally magnify surface details, to re-light the objects from numerous orientations, and to create synthetic illustrations on the fly to help with the analysis. In an educational setting, a teacher might be explaining anatomy to young students using a handheld mock-up of a skull. Being able to virtually slice the object on the fly, even approximately, would add powerful interactive visuals to the didactic task. For the inspection of mechanical parts, technical illustrations could be easily generated or a collection of shape profiles of a damaged piece could be preliminarily inspected and used to prime a search through a parts database. Moreover, a low-cost approach would permit easy and widespread dissemination.

Our work performs inspection using a self-calibrating 3D reconstruction. On the one hand, most inspection methods use a collection of high-resolution images to enhance the illustration of an object via careful lighting design and non-photorealistic rendering strategies (e.g., [Akers et al. 2003; Lee et al. 2004; Winnemoller et al. 2005]) without explicitly recovering surface normal and surface geometry information. Thus, these techniques typically depend on previously acquired object data. On the other hand, acquisition methods obtain the overall surface geometry, but need precise calibration and/or extended time to robustly obtain the fine surface details necessary for digital inspection. Passive acquisition methods, such as image-based modeling and stereo vision, need many calibrated images and depend on fragile

correspondence computations. Active acquisition methods, such as lasers or structured-light, are more robust than passive methods but full high-resolution captures can be costly in terms of time and/or monetary expense, and often require geometric calibration and specialized equipment.

In this paper, we present a complete system to first rapidly and robustly acquire a 3D object, including the fine surface details needed for digital inspection, and second an appropriate set of the visualization tools for immediately performing digital inspection of the just captured handheld objects (Figure 1). Our digital inspection stage includes one uncalibrated digital video camera and three uncalibrated digital projectors. For capture, a user simply places a new object in front of the camera and the system projects a short sequence of structured light patterns during several seconds. The system instantly performs a photometric-stereo reconstruction for unknown light sources. The computed surface normals are integrated to produce an approximate surface with high-details but suffering from overall low-frequency deformations. The structured light patterns are used to create a coarse set of corresponded pixels between camera and projectors. Instead of a dense and more time consuming geometric reconstruction, we use the photometrically-computed lighting directions and the coarse correspondence to perform a very fast self-calibrating and low-resolution geometric reconstruction. Then, the highly-detailed photometrically-computed surface is warped to the coarse geometrical model, producing a single high-resolution model. For visualization, the user can physically manipulate the object in front of the camera or control a synthetic rendering. Our interactive inspection tools enable instant relighting, shading exaggeration, depth-based lighting, depth-based detail modulation, object slicing, and iso-distance curves.

The entire capture process requires the object to be static for about ten seconds, completes in about another 20 seconds, provides a sub millimeter point-sampled model, converges to less than a pixel of error, and typically contains over 500,000 triangles in all our examples. The captured models are tracked and visualized at interactive rates using synthetic rendering and/or texture-mapping. We demonstrate our system with the capture and visualization of several objects.

Our contributions include

- a framework for the interactive and low-cost generation of illustrations for the digital inspection of hand-manipulated physical objects,
- a fast, robust, and self-calibrating acquisition process that integrates geometric and photometric data, and
- a set of visualization tools for analyzing 3D objects using surface normals and surface geometry data.

## 2. Related Work

Recently, several significant works have developed visualization and rendering strategies for conveying shape and surface details. These methods enhance the visualization of scientific and historically-important objects by exploiting photorealistic and non-photorealistic rendering strategies in order to inspect desired object features (e.g., [Akers et al. 2003; Barla et al. 2006; Bartesaghi et al. 2005; Rusinkiewicz et al. 2006]). However, most of these methods assume a priori captured 3D models.

Acquiring detailed 3D models of handheld and perhaps moving objects is a challenging problem. On the one hand, passive

methods are unobtrusive, but need to establish robust correspondences using only natural features. On the other hand, active methods explicitly generate correspondences. Laser-based systems acquire high-resolution geometry but often only capture geometry (i.e., no color), can be expensive, and require lengthy cleanup and post-processing to obtain smooth and accurate surface normals and geometry. Structured-light methods use projected patterns to reconstruct a model. However, obtaining a typical high-resolution model is time consuming and does not necessarily produce smooth and accurate normals. While some self-calibrating structured-light systems have been presented [Furukawa and Kawasaki 2005], most require a pre-calibrated setup. One-shot structured-light methods acquire objects quickly but only obtain low-resolution geometry (e.g., Koninckx and van Gool 2006; Zhang et al. 2002) and sacrifice obtaining color information. A few methods have partially addressed simultaneously obtaining color information but need additional cameras and specialized hardware setups (e.g., [Frueh and Zakhor 2005; Waschbüsch et al. 2005]). For slow moving rigid objects, Rusinkiewicz et al. [2002] incrementally build objects by matching projected features from temporally adjacent frames. However, the system requires pre-calibration, is applied to near-white objects, and does not necessarily produce accurate surface normals. Moreover, none of these methods also provide surface-detailing enhancing visualization tools.

Regardless, simply recovering global geometry is often not sufficient to enable many of the aforementioned visualization and rendering strategies. Rather, it has been evidenced in computer graphics that obtaining significant surface detail needs photometric data [Nehab et al. 2005; Rushmeier and Bernardino 1999]. For instance, bump mapping, displacement mapping, and relief textures [Oliviera and Bishop 2000] are examples of methods that show significant surface detail (and do not necessarily use complex geometry).

Methods to obtain surface information from photometric measurements, such as photometric stereo and shape-from-shading, have produced a large amount of literature. Research has generally tackled Lambertian surfaces, sometimes specular surfaces, and treated known and unknown lighting situations. Recovering accurate global surface geometry, however, is a difficult task because of the inherent ambiguities (e.g., [Basri et al. 2007; Zhang et al. 2003]).

Image-based relighting avoids explicitly building geometric models and/or computing surface normals by obtaining a large number of images of the object under different illumination conditions. This has been used to relight images from a stationary viewpoint (e.g., [Winnemoller et al. 2005]) and to enable lighting design (e.g., [Lee et al. 2004]). Debevec [2006] has developed several impressive systems for relighting human actors but requires a very large number of images and a complex and customized lighting stage, including very high frame rate cameras (in excess of 4500 fps).

Most closely related to our work are several interactive surface normal computation systems and surface discontinuity detecting systems. For example, Malzbender et al. [2006] describe a GPU-based real-time system for small objects that assumes fixed and known 16 lighting directions and a 500Hz camera fixed relative to the lights. Masselus et al. [2002] present a lighting stage using a fixed camera and a freely moving light source but need several known calibration objects in the scene. Neither work exploits both surface normal and accurate surface height data to create

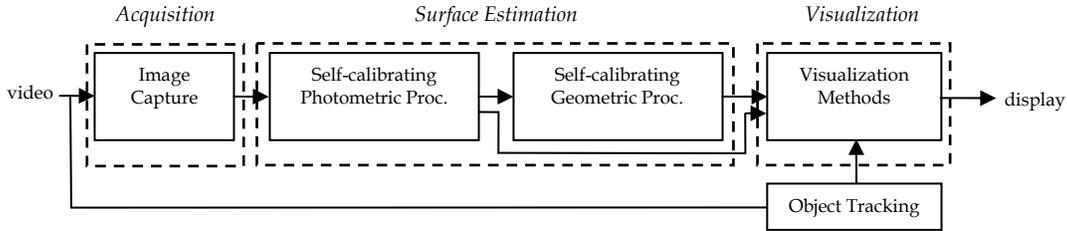


Figure 2. *System Pipeline.* We show a summary of the processing pipeline.

visualizations. Raskar et al. [2004] proposed a method for producing edge-enhanced 2D illustrations. While they illuminate a static scene with a quick sequence of flashes from multiple directions, they do not obtain accurate geometry information.

In contrast, we provide an interactive system that with very little effort on part of the user enables quickly capturing and inspecting the targeted objects. Our approach is fast and fully self-calibrating, thus providing significant flexibility during use. We perform a geometric reconstruction using structured-light patterns (as opposed to relying on fragile feature-tracking; e.g., [Lim et al. 2005]). Further, without needing a highly-customized lighting stage, placing known markers in the scene, pre-calibration, or prior object information, our approach supports freely positioning the camera and light sources and producing surface-detail enhancing illustrations. Furthermore, having the flexibility of freely moving the object, camera, and/or lights enables inspecting objects to which access is difficult.

### 3. Digital Inspection

Our system consists of several major components that interoperate and together form an interactive program (Figure 2). The acquisition component uses each digital projector to project a short sequence of at most 21 structured-light patterns onto the object, over 10 seconds or less, and uses the camera to capture each projected pattern. During this time, the object must be static. The photometric processing component uses the captured images corresponding to all-white patterns to photometrically estimate per-pixel normals and each lighting direction. The geometry processing component uses the additional patterns to correspond up to a thousand pixel between projectors which act as virtual cameras. Since the projectors are both the light sources for photometric processing and the camera for geometric processing, the photometrically-computed lighting directions are used to initialize the camera viewing directions. A fast self-calibration is performed to more accurately estimate camera parameters and to improve surface points from their photometrically-estimated location. Once the low-resolution geometry model is built, the

additional surface point samples from the photometric model are warped to the low-resolution geometry model. Finally, the reconstructed object model is matched to corresponding surface features tracked during visualization. The visualization methods are selected by the user and render novel imagery.

#### 3.1 Photometric Processing

Our photometric stereo processing quickly recovers the surface normals and illumination directions of an object using multiple images in which the viewpoint is fixed but the lighting conditions vary. Our method is based on that of Woodham et al. (1991) which assumes the surface is mostly Lambertian and the lights distant and directional. Under these conditions, the illumination model simplifies to

$$NL^T = C \quad (2)$$

where  $N$  is a  $k \times 3$  matrix of  $k$  outward-facing surface normals,  $L$  is a  $l \times 3$  matrix of  $l$  light directions pointing towards the light, and  $C$  is a  $k \times l$  matrix of the observed pixel intensities. Given at least three images, each illuminated by a different known lighting direction (Figures 3a-c), the matrix  $N$  can be recovered by computing  $N = C(L^T)^{-1}$  using pseudo-inverses.

In the case when both  $N$  and  $L$  are unknown, the system can still be solved using a linear least squares optimization. We look closely at the case of one pixel/normal and three lights, namely  $k=1$  and  $l=3$ . In this case, matrix  $N$  becomes a row vector normal  $n^T$  and the matrix  $C$  becomes a row vector  $c^T$  consisting of the three observations of the pixel under the illumination of the three light sources. Equation (2) can be written as  $n^T L^T = c^T$ . Next, we define a new matrix  $D = (L^T)^{-1}$  and rewrite equation (2) as  $n^T = c^T D$ . Since we desire surface normals of unit length, we express this fact using the quadratic  $n^T n = 1$  which, using the new version of equation (2), can also be written as  $(c^T D)(D^T c) = 1$ , or as

$$c^T E c = 1 \quad (3)$$

where  $E = DD^T$  is symmetric (and positive definite, e.g., positive diagonal elements), thus consisting of six unknowns. This

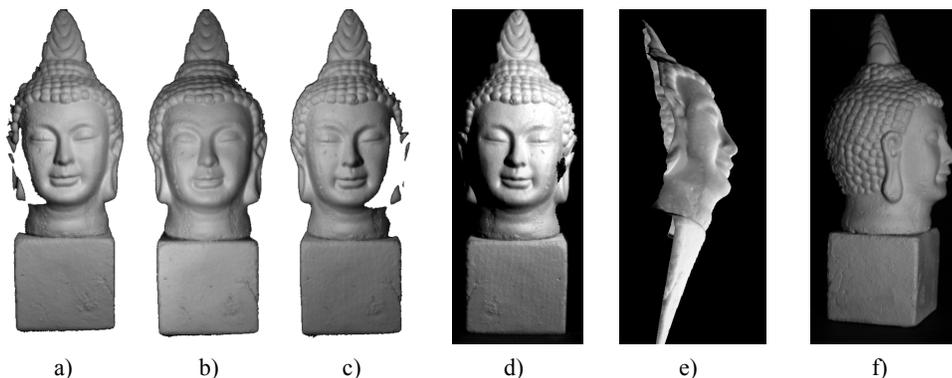


Figure 3. *Photometric Processing.* (a-c) Input pattern images with “zero-stripes” illuminated from three projectors. (d) Synthetically re-lit image of Buddha. (e) Surface obtained by integrated normals. (f) Actual photograph of object; notice the global deformations of the integrated surface.

expression, when expanded, takes the form

$$e_{11}c_1^2 + e_{22}c_2^2 + e_{33}c_3^2 + 2e_{12}c_1c_2 + 2e_{13}c_1c_3 + 2e_{23}c_2c_3 = 1. \quad (4)$$

The equation writes the six unknowns of  $E$  as a linear combination of the known components  $c_i$  of the vector  $c$ . Given at least six pixels and three intensities per pixel, the  $e_{ij}$  variables are over-constrained; thus, the  $e_{ij}$ 's can be solved for using linear least squares. Once  $E$  is recovered, observe that

$$E^{-1} = (DD^T)^{-1} = ((L^T)^{-1})(L^T)^{-1} = LL^T. \quad (5)$$

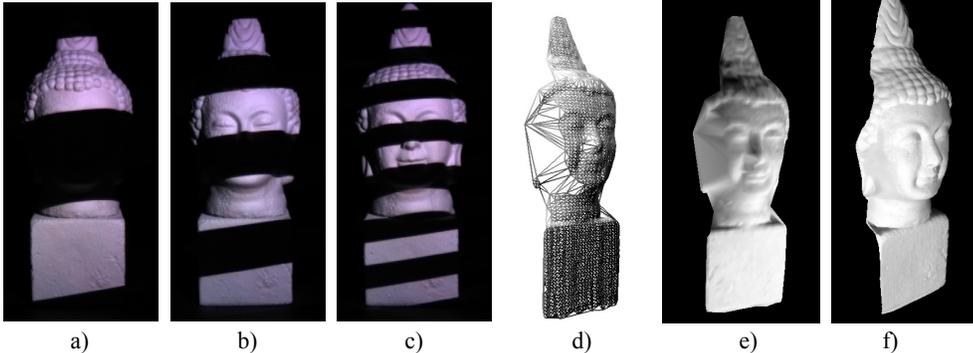
From equation (5) we can recover the magnitude of and angles between light directions, up to an unknown global rotation. Then, we plug  $L^T$  into equation (2) to compute the array of normals  $N$  (Figure 3d). The length of each normal is an approximation of the albedo of the corresponding object point sample.

While a surface height field can be calculated from the surface normals, it can only be done up to an unknown global rotation and to a generalized bas-relief transform (GBR) [Belhumeur et al. 1999]. For lights of different intensities, the generalized bas-relief transform consists of a three-parameter family of transformations that “warp” the normal vectors and the corresponding light vectors yet produce the same observed pixel intensities when viewed under orthographic projection. Without additional knowledge of the scene or multiple calibrated views, these ambiguities cannot be resolved. The GBR transform  $(\lambda, \mu, \nu)$  of a surface has the matrix  $[1 \ 0 \ 0; 0 \ 1 \ 0; \mu \ \nu \ \lambda]$ .

In our system, for each lighting setup we once find the best rotation  $R$  and generalized bas-relief transformation  $G$  that reconstructs an initial frame and then warps all solutions to the same set of lighting directions and GBR transformation. We assume equal light source intensity and thus the GBR transform simplifies to  $(\lambda, 0, 0)$  [Belhumeur et al. 1999]. For an arbitrary image (e.g., the first one), we use our GUI to estimate  $\lambda$  and to determine a 3D rotation that brings the lighting configuration into approximate registration with the  $xy$  image plane (a virtual trackball makes this trivial). In our experience, we found the reconstruction to not be very sensitive to performing a precise  $xy$  plane registration and  $\lambda=0.3$  worked well for all our examples.

To compute a photometrically-estimated surface  $z(x,y)$  (i.e., a height field), we integrate surface normals. The normals of  $z(x,y)$  are approximately given by  $(s, t, -1)$  with  $s = z_x$  and  $t = z_y$ , and  $z_x$  and  $z_y$  denote partial derivatives of  $z$  with respect to  $x$  and  $y$ . On a grid, the values  $s$  and  $t$  can be approximated by

$$\begin{aligned} s &= z(x+1, y) - z(x, y) = -n_x / n_z \\ t &= z(x, y+1) - z(x, y) = -n_y / n_z \end{aligned} \quad (6)$$



**Figure 4. Geometric Processing.** (a-c) Input pattern images with horizontal stripes. (d-e) Wireframe and filled view of coarse self-calibrated model. (f) Final model after up-sampling and merging with photometric data (rendering using OpenGL lighting).

where  $(n_x, n_y, n_z)$  are the components of the estimated normal at pixel  $(x, y)$ . Then, we integrate and construct a surface height field. Surface integration produces an approximation, up to a scale. For an object that is loosely convex in shape (e.g., a single closed “object”), the low-frequency noise, typical with photometric stereo, is tolerable as an initial surface estimate (Figures 3e-f).

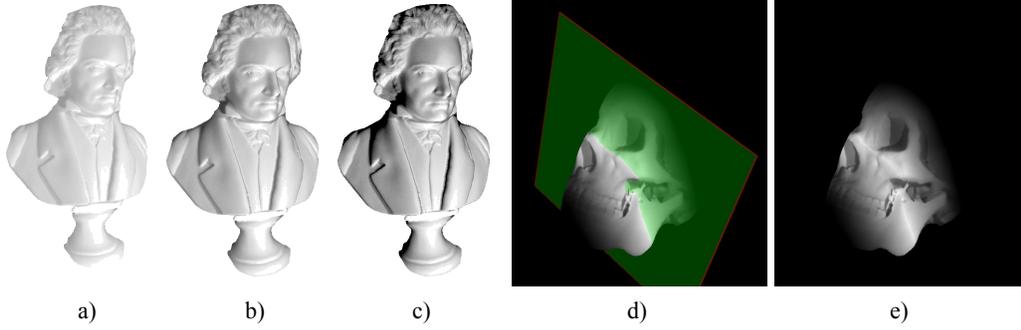
### 3.2 Geometric Processing

To obtain a more precise surface, our system improves the photometrically-computed surface by using our novel method and the additional images containing projected structured-light patterns. In particular, photometric stereo is regarded as using an initial projected pattern with zero-stripes. Projecting  $P$  additional pairs of binary stripe patterns and keeping the object static for at most a few seconds permits robustly corresponding  $(2^P-1)^2$  surface point samples between the camera and the multiple projectors. To yield few point samples and fast processing, we use small values for  $P$  (e.g., 5). To prevent determining surface albedos during structured-light processing, we project the patterns and their inverses as well [Scharstein and Szeliski 2003]. From the three-view correspondence data and the photometrically computed surface normals and lighting directions, we perform a geometric self-calibration (Figures 4a-c).

Given at least two corresponded views, geometric processing seeks to estimate camera parameters and the 3D location of surface points so as to minimize the reprojection error expressed by the following known nonlinear system of equations

$$\sum_j \sum_i \left( \frac{1}{q_{ijz}} \begin{bmatrix} q_{ijx} \\ q_{ijy} \end{bmatrix} - \begin{bmatrix} u_{ij} \\ v_{ij} \end{bmatrix} \right)^2 \text{ where } q_{ij} = F_j(R_j p_i + T_j) \quad (7)$$

where  $p_i$  are the unknown world coordinates of surface point  $i$ ,  $(u_{ij}, v_{ij})$  are the known projections of surface point  $i$  onto view  $V_j$ , and  $R_j, T_j$ , and  $F_j$  are the unknown camera 3x3 rotation matrices, 3D translation vectors, and 3x3 perspective projection matrices, respectively. Since this computation is nonlinear and the number of unknowns (e.g., 3D surface points and camera parameters) is large, convergence depends heavily on having good initial estimates of the surface geometry and camera locations [Pollefeys et al. 2004]. Fortunately, our photometric stereo processing provides us with an initial surface estimate and effective viewing directions. By regarding the three projectors as cameras, photometric stereo effectively provides the initial camera rotation matrices  $R_j$  for  $j=1$  to 3. We then assume a reasonable surface-to-camera distance and provide as input the manufacturer’s specified focal length for the projectors (which only needs to be an approximation and will be refined subsequently). Altogether, this



**Figure 5. Shading Exaggeration and Depth Lights.** (a-c) Mild to exaggerated shaded. (d-e) Depth lights with reference plane in green.

enables performing a robust optimization for computing the 3D location of corresponded surface points.

The photometrically-calculated lighting directions  $l_j$  and an assumed up-vector of  $w=[0 \ 1 \ 0]^T$  are used to create an oriented orthogonal coordinate system for each effective view, represented by a matrix  $M_j$ . The columns of this matrix correspond to the normalized vectors  $l_j \times w$ ,  $(l_j \times w) \times l_j$ , and  $-l_j$ . To bring the re-projection of the surface points into rough alignment with the structured-light observations, we first optimize the following simplified nonlinear system of equations of only 4 unknowns

$$\sum_j \sum_i \left( \frac{\hat{p}_{ij,x} f}{\hat{p}_{ij,z} + z_j} - u_{ij} \right)^2 + \left( \frac{\hat{p}_{ij,y} f}{\hat{p}_{ij,z} + z_j} - v_{ij} \right)^2 \quad (8)$$

where  $\hat{p}_{ij} = M_j p_i$ ,  $j \in [1, 3]$ , and the free parameters are the distances  $z_j$  from the origin to each projector  $j$  along the  $l_j$  vector and a single global focal length  $f$  assumed to be used by all projectors (i.e., we assume all projectors are of the same brand, model, and focal length).

Since we now have a good guess for all scene points and camera parameters, we use a sparse nonlinear bundle adjustment optimization library to refine both camera pose parameters and surface points [Lourakis and Argyros 2004]. Outliers are culled using image-space and world-space criteria. The optimization and culling is repeated several times until no more surface points are removed and the process converges (Figures 4d-e).

### 3.3 Mapping Photometrically-Computed Points to the Geometric Model

Once a coarse geometric model has been computed, we warp the photometrically-computed points to the coarse model. While in general morphing one 3D object to another is a challenging problem, we have per-pixel correspondence between the photometric- and geometric-points; they are both in projector space. Thus, a piecewise linear mapping can be used to warp points from the photometric-surface to the geometric-surface.

To map a point from the photometric surface to its corresponding point on the geometric surface, we use barycentric coordinates to interpolate a displacement between the corresponded displacement vectors of the surrounding triangle of geometrically-calibrated points. In particular, geometric-surface point  $p_{G_i}$  corresponding to photometric surface point  $p_{P_i}$  is computed by

$$p_{G_i} = \alpha_i(a_{G_i} - a_{P_i}) + \beta_i(b_{G_i} - b_{P_i}) + \gamma_i(c_{G_i} - c_{P_i}) \quad (9)$$

$$(\alpha_i, \beta_i, \gamma_i) = \text{barycoords}(a_{P_i}, b_{P_i}, c_{P_i})$$

where  $(a_{P_i}, b_{P_i}, c_{P_i})$  are the vertices of the triangle of geometrically-calibrated points immediately surrounding  $p_{P_i}$ , and  $(a_{G_i}, b_{G_i}, c_{G_i})$  are the corresponding points on the geometric surface. Points on both surfaces are triangulated using 2D Delaunay triangulation. Once all points are mapped to the geometric surface, we re-triangulate, subdivide large triangles, and perform a Laplacian mesh-smoothing operator (Figure 4f).

### 3.4 Object Alignment

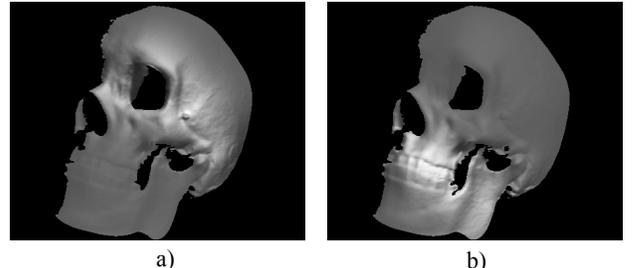
To enable the user to intuitively manipulate the reconstructed object, we track features on the physical handheld object and search for a transformation to bring the reconstructed object into alignment with the current view. Immediately after projecting patterns, our system begins tracking features on the observed object using a Kanade-Lucas-Tomasi feature tracker. Our visualization methods can be applied either to the tracked object or to a virtual-trackball controlled object.

To perform object tracking, we must first calibrate the digital camera (so far it has remained uncalibrated). In a similar fashion to the projectors, the manufacturer's specified focal length and a motion-only bundle-adjustment optimization are used to find a set of camera parameters that best aligns the reconstructed model to the corresponding features of the camera.

Our alignment algorithm assumes that approximately the same portion of the reconstructed object is visible subsequent to geometry processing and attempts to find a 6 degree-of-freedom rigid transformation. In particular, a nonlinear optimization method (i.e., simplex) searches for a 3D translation vector  $T$  and a 3D rotation matrix  $R$ , parameterized by roll  $\varphi$ , pitch  $\phi$ , yaw  $\omega$ , that reduces re-projection error of the tracked surface features. If the object's orientation changes too much, the user is flagged and a new capture is suggested.

## 4. Visualization Methods

Our system provides several visualization tools for interactively focusing on surface details. The algorithms exploit knowledge of



**Figure 6. Depth-based Modulation.** (a-b) Depth modulation rendering, details far from reference plane have less contrast.

surface normal and surface height data. All of our methods can be applied to live video, to frozen video frames, or to recorded video.

#### 4.1 Exaggerated Shading

In order to bring out surface detail, one option is to exaggerate the shading of the surface. Such shading has been explored in various forms in the literature. For instance, shaded relief (e.g., [Rusinkiewicz et al. 2006] employs height exaggeration and, under orthographic projection, produces larger changes between light and dark regions. Similarly, normal exaggeration (e.g., [Malzbender et al. 2006; Willems et al. 2005]) has used been to exaggerate the effect of shading using a single gain parameter  $\delta$  (e.g.,  $n' = (\delta n_x, \delta n_y, (1 - (\delta n_x)^2 - (\delta n_y)^2)^{1/2})$ ). Both of these can be mapped to the GBR transform  $(\lambda, \mu, \nu)$ .

When  $G$  is applied to a surface point, it has the effect of “exaggerating” the z-component (height). When  $(G^{-1})^T$  is applied only to the normals, it scales the z-component of the normal by  $1/\lambda$  and thus, after renormalization, has the effect of exaggerating the normals away from (0,0,1). Our system supports both exaggeration effects. They require simple re-computations of either the surface normals or the surface points (Figure 5a-c).

#### 4.2 Depth-based Shading

In addition to shading varying according to surface normals, surface height can be used to apply different shading strategies to different depths of the object (e.g., [Barla et al. 2006]). We showcase three depth-based shading methods.

##### 4.2.1 Depth Lights

Depth lights shine a virtual light onto the object and emphasize surface details at a particular surface height. The user specifies an imaginary reference plane  $D$  and object points with a small normalized distance  $d \in [0,1]$  from the plane are brightly illuminated. The remaining more distant object points become attenuated using standard or exaggerated Lambertian shading. Although this does not correspond to a physically possible illumination setup (at least not without considerable planning!), it yields an intuitive visualization. The formulation for shading intensity has the simple form

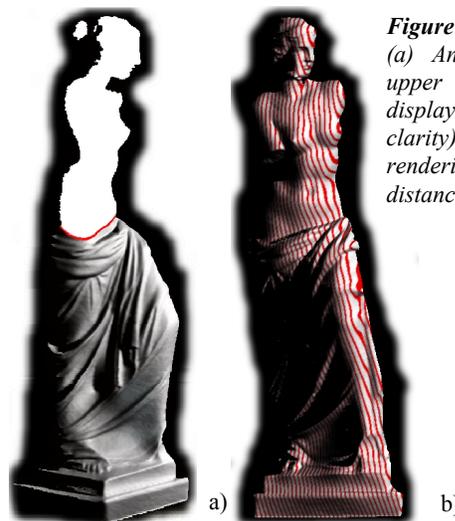
$$(1 - \rho d^\sigma)(n \cdot l) \quad (10)$$

where  $\rho$  and  $\sigma$  are constants controlling the width and fall-off rate of the depth light and  $n$  and  $l$  are the pixel normal and virtual light direction. In our examples, we use  $\rho=6$  and  $\sigma=2$  and clamp the minimum value to an ambient term.

The imaginary reference plane is defined relative to the camera space. The operator can move the object in front of the camera and keep the depth-light in approximately the same spatial location. This enables physically moving the object in order to choose what features to highlight. Figures 5d-e contain example renderings using depth lights. The reference plane is drawn as a semi-transparent green plane. Details near the plane are illuminated while farther away features are more in shadow.

##### 4.2.2 Depth-based Detail Modulation

Another depth-based method is to modulate surface detail based on depth. Thus, as opposed to a spatially-selected region of interest, the operator can choose a depth-range of interest. While the average illumination intensity behaves globally like a diffuse



**Figure 7. Object Slicing.** (a) An object with its upper half clipped (but displayed in white for clarity). (b) A synthetic rendering with iso-distance curves enabled.

surface, high-frequency object details are apparent only in the selected depth range. Similar to depth lights, the depth-range is specified using an imaginary plane and its location remains constant in the surface integration space.

This effect is possible using a two-parameter interpolation ( $g=n \cdot l$ ,  $h=d$ ) between four control points  $c_0$  to  $c_3$ . We select a full dark color (e.g.,  $c_0$ =near black), a full bright color (e.g.,  $c_1$ =white), a dark color for attenuated detail (e.g.,  $c_2$ =dark gray), and a bright color for attenuated detail (e.g.,  $c_3$ =gray). The shading intensity equation is then

$$(1 - g^\kappa)hc_0 + (g^\kappa h^\omega)c_1 + (1 - g^\kappa)(1 - h^\omega)c_2 + g^\kappa(1 - h^\omega)c_3 \quad (11)$$

where  $\kappa$  and  $\omega$  control the fall-off from light to dark and near to far. For standard Lambertian shading and a linear detail fall-off,  $\kappa=\omega=1$ . For mild shading exaggeration and a quick detail fall off,  $\kappa=\omega=2$  performs well. Figure 6 illustrates depth-based detail modulation. Users can interactively move the reference plane through the object and highlight the desired details for inspection.

#### 4.3 Object Slicing

Our system also supports the slicing of objects at selected distances from the reference plane and the rendering of multiple iso-distance curves. Using the reference plane, we can cull object points that are “behind” the reference plane. By performing this task interactively, we in fact obtain a live rendering of the object’s contour intersecting the reference plane (Figure 7a).

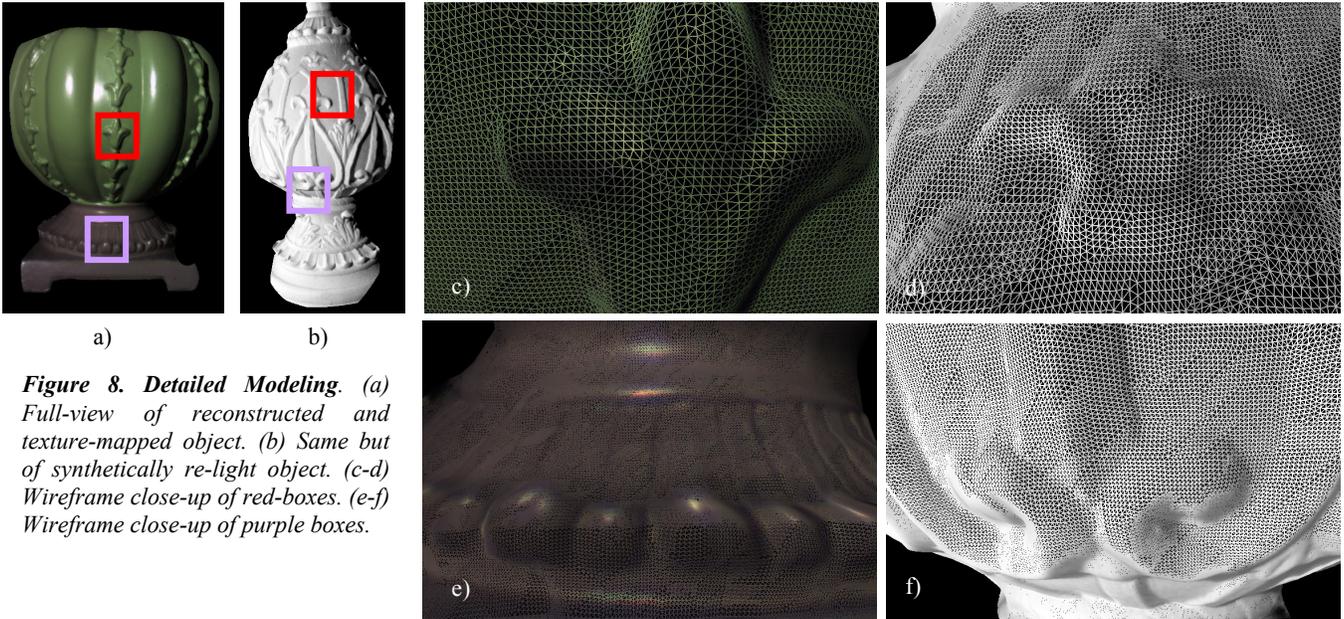
Instead of culling the surface points, we can highlight all points at regularly spaced distances from the reference plane. This produces iso-distances curves on the object itself and serves to visualize its curvature and general shape. Pixels are determined to be on a iso-distance curve whenever

$$(md - \lfloor md \rfloor) < \tau \quad (12)$$

for  $m$  lines and a line thickness of  $\tau \in [0,1]$ . Figure 7b shows an object rendering with multiple iso-distance curves.

## 5. Results

We have implemented a prototype of our digital inspection stage in C/C++ using a 3.6 GHz PC. To project patterns, we use a low-resolution Mitsubishi Handheld PK10 projector of a resolution of



**Figure 8. Detailed Modeling.** (a) Full-view of reconstructed and texture-mapped object. (b) Same but of synthetically re-light object. (c-d) Wireframe close-up of red-boxes. (e-f) Wireframe close-up of purple boxes.

800x600 pixels. Our video camera is a Point Grey Research Flea camera connected to the PC via Firewire, capturing 1024x768 color images. The camera can operate at 30 fps when storing images to CPU memory, but in order to have sufficient projector settle time and camera exposure, we operate at 7.5 fps.

Photometric processing and surface integration uses a custom implementation of the described solution methodology. For a typical full-resolution image, solving for the  $N$  and  $L$  matrix and integrating surface normals, culled to the bounding box of the object, occurs in less than a second.

Geometric processing time varies mostly depending on the number of iterations of optimize-cull. We use Numerical-Recipes in C optimization codes. The geometry up-sampling completes in less than one second. Examples of object details are in Figure 8. Feature tracking and object-pose optimization also operate at interactive rates (Figure 9).

All visualization and rendering occurs at 15+ fps -- it is limited by the number of points that actually need to be rendered. Computations are performed on the CPU and graphics rendering uses OpenGL, GLUT, and GLUI. Shading operations are performed on a triangulation of the surface points and display lists are used whenever possible (Figure 10).

Our system has been used to inspect a variety of objects and to illustrate our visualization methods: *Beethoven*, *Venus di Melo*, *Skull*, *Budda*, *House*, *Column*, *Urn*, *Vase*, and *Pot*. The objects are from handheld to tabletop in size, with a self-calibration

convergence error of less than one pixel, containing about 1000 points in the coarse geometric reconstruction, and about 250k points in the photometric reconstruction. The final meshes contain 252k to 640k triangles and reconstruction time is 8-20 seconds.

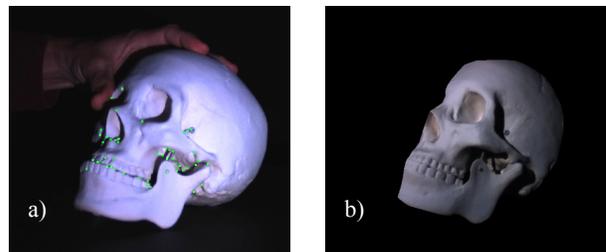
## 6. Conclusions and Future Work

We have described our interactive system for enabling the live inspection and enhancement of 3D objects. Our approach is tailored for quick, easy, and accurate use. While other approaches have used customized hardware, many lights, many images, pre-calibration, and/or synchronized high frame-rate cameras, our method uses only simple off-the-shelf hardware and is fully self-calibrating. This enables a non-trained user to adjust the camera, projectors, and object at will. A new capture can be completed in a seconds and visualizations can be applied to the hand-manipulated object, yielding a complete easy-to-use and intuitive digital inspection stage.

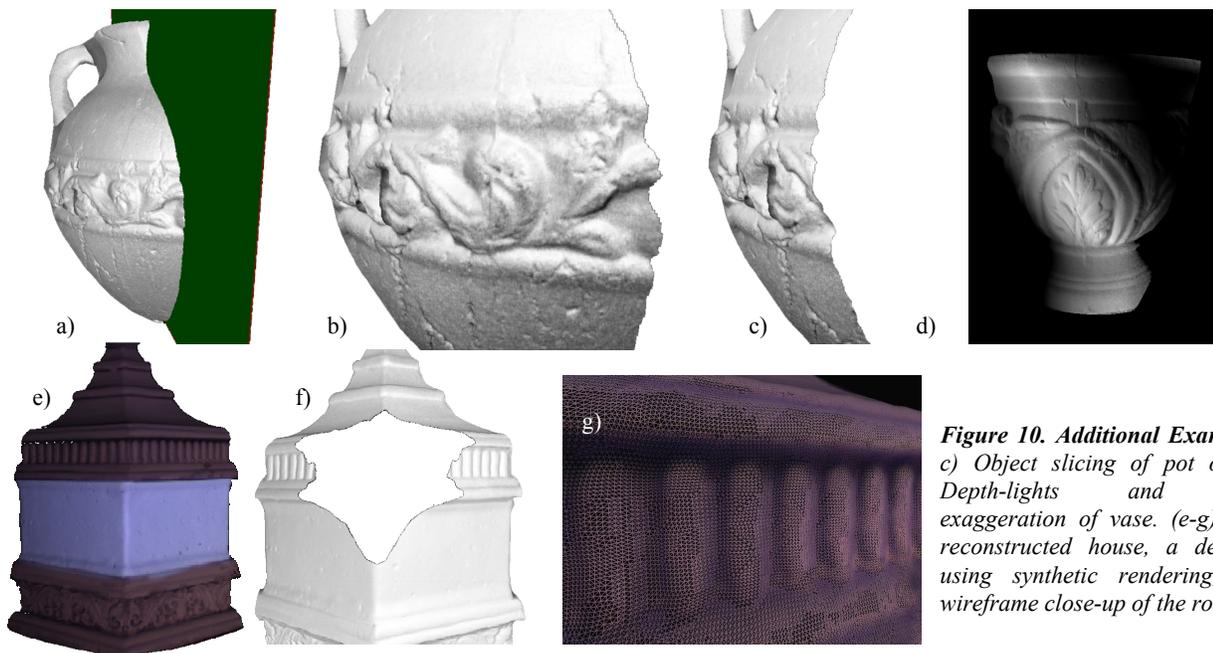
Regarding future work, there are aspects that can further be improved. As is the case with most photometric methods, shadows are problematic. One option is to repetitively move the lights in front of the object and to combine the surface normals computed from more than one image, effectively performing an “or” operation. We are also pursuing methods to exploit the temporal coherence of video sequences. Finally, to further accelerate performance, we are investigating the use of CUDA, a new NVidia-provided SDK for performing linear algebra computations on the GPU.

Model	Total Pts/Tris	Time (sec)	Error (px)
Beethoven	250k/500k	12	0.81
Buddha	204k/409k	14	0.75
Column	279k/558k	16	0.73
House	261k/521k	15	0.69
Pot	320k/640k	16	0.75
Vase	235k/470k	13	0.78
Skull	202k/405k	17	0.65
Statue	227k/454k	14	0.84
Urn	303k/605k	20	0.55
Venus	126k/252k	8	0.93

**Table 1.** Statistics of the captured objects.



**Figure 9. Interactive Manipulation.** (a) The real object is tracked using features in order to intuitively manipulate the reconstructed object, shown in (b), during visualizations.



**Figure 10. Additional Examples.** (a-c) Object slicing of pot object. (d) Depth-lights and shading exaggeration of vase. (e-g) View of reconstructed house, a depth slice using synthetic rendering, and a wireframe close-up of the roof.

## References

- AKERS D., LOSASSO F., KLINGNER J., AGRAWALA M., "Conveying Shape and Features with Image-Based Relighting", *IEEE Visualization*, 46-53, 2003.
- BARLA P., THOLLOT J., MARKOSIAN L., "X-Toon: An Extended Toon Shader", *Int. Symposium on Non-Photorealistic Animation and Rendering*, 127-132, 2006.
- BARTESAGHI A., SAPIRO G., MALZBENDER T., GELB D., "Three-dimensional shape rendering from multiple images", *Graphical Models*, 67(4), 332-246, 2005
- BASRI R., JACOBS D., KEMELMACHER I., "Photometric Stereo with General, Unknown Lighting", *Int. Journal of Computer Vision*, 72(3), 239-257, 2007.
- BELHUMEUR P., KRIEGMAN D., YUILLE A., "The Bas-Relief Ambiguity", *Int. J. of Computer Vision*, 35(1), 33-44, 1999.
- DEBEVEC P., "Virtual Cinematography: Relighting Through Computation", *IEEE Computer*, 2006.
- FATTAL R., AGRAWALA M., RUSINKIEWICZ S., "Multiscale Shape and Detail Enhancement from Multi-light Image Collections", *ACM SIGGRAPH*, No. 51, 2007.
- FRUEH C., ZAKHOR A., "Capturing 2½D Depth and Texture of Time-varying Scenes using Structured Infrared Light", *Proc. of 3DIM*, 318-325, 2005.
- FURUKAWA R., KAWASAKI H., "Uncalibrated multiple image stereo system with arbitrarily movable camera and projector for wide range scanning", *Proc. of 3DIM*, 302-309, 2005.
- LIM J., HO J., YANG M., KRIEGMAN D., "Passive Photometric Stereo from Motion", *IEEE Int. Conference on Computer Vision*, 1635-1642, 2005.
- LEE C., HAO X., VARSHNEY A., "Light Collages: Lighting Design for Effective Visualization", *IEEE Vis.*, 281-288, 2004.
- KONINCKX T.P., GRIESSER A., VAN GOOL L., "Real-time Range Scanning of Deformable Surfaces by Adaptively Coded Structured Light", *Proc. of 3DIM*, 293-300, 2003.
- MALZBENDER T., WILBURN B., GELB D., AMBRISCO B., "Surface Enhancement Using Real-time Photometric Stereo and Reflectance Transformation", *European Symposium on Rendering*, 245-250, 2006.
- MASSELUS V., DUTRE P., ANRYS F., "The Free-form Light Stage", *Eurographics Workshop on Rendering*, 247-256, 2002.
- NEHAB D., RUSINKIEWICZ S., DAVIS J., RAMAMOORTHI R., "Efficiently Combining Positions and Normals for Precise 3D Geometry", *ACM SIGGRAPH*, 536-543, 2005.
- OLIVIERA M., BISHOP G., MCALLISTER D., "Relief Texture Mapping", *ACM SIGGRAPH*, 359-368, 2000.
- RASKAR R., FERIS R., YU J., TURK M., "Non-photorealistic Camera: Depth Edge Detection and Stylized Rendering Using Multi-flash Imaging", *ACM SIGGRAPH*, 678-688, 2004.
- RUSHMEIER H., BERNARDINI F., "Computing Consistent Normals and Colors from Photometric Data", *3DIM*, 99-108, 1999.
- RUSINKIEWICZ S., HALL-HOLT O., LEVOY M., "Real-time 3D Model Acquisition", *ACM SIGGRAPH*, 438-446, 2002.
- RUSINKIEWICZ S., BURNS M., DECARLO D., "Exaggerated Shading for Depicting Shape and Detail", *ACM SIGGRAPH*, 1199-1205, 2006.
- SCHARSTEIN S., SZELISKI R., "High-Accuracy Stereo Depth Maps Using Structured Light", *IEEE CVPR*, 195-202, 2003.
- WASCHBÜSCH M., WÜRMLIN S., COTTING D., SADLO F., GROSS M., "Scalable 3D Video of Dynamic Scenes", *The Visual Computer*, 21(8-10), 629-638, 2005.
- WILLEMS G., VERBIEST F., MOREAU W., HAMEEUW H., VAN LERBEGHE K., VAN GOOL L., "Easy and Cost-Effective Cuneiform Digitizing", *Int. Symposium on Virtual Reality, Archaeology, and Cultural Heritage*, 2005.
- WINNEMOLLER H., MOHAN A., TUMBLIN J., GOOCH B., "Light Waving: Estimating Light Positions from Photographs Alone", *Comp. Graphics Forum (Eurographics)*, 24(3), 433-438, 2005.
- WOODHAM R., IWAHORI Y., BARMAN R., "Photometric stereo: Lambertian reflectance and light sources with unknown direction and strength", *Univ. of B. C., TR-91-18*, 1991.
- ZHANG L., CURLESS B., SEITZ S.M., "Rapid Shape Acquisition using Color Structured Light and Multi-pass Dynamic Programming", *Proc. of 3DPVT*, 24-36, 2002.
- ZHANG L., CURLESS B., HERTZMANN A., SEITZ S.M., "Shape and Motion under Varying Illumination: Unifying Structure from Motion, Photometric Stereo, and Multi-view Stereo", *IEEE Int. Conference on Computer Vision*, 618-625, 2003.