Exemplar Based Surface Texture

Antonio Haro and Irfan Essa

GVU Center / College of Computing
Georgia Institute of Technology
http://www.cc.gatech.edu/cpl/projects/surfacetexture/

Abstract

Realistic rendering of computer modeled three dimensional surfaces typically involves estimation of the reflectance properties of the material to be used for rendering the surface, or use of photographs of the material for texturing instead. Bidirectional texture functions (BTFs) can be used for this purpose, however, full coverage of all viewing and lighting directions desired must be acquired or estimated. We present a computationally inexpensive method for rendering a 3D surface using a single image. We use a photograph of a sphere or ellipsoid of the source material as the exemplar and sample from it non-uniformly and coherently to produce a realistic rendering of an input 3D model from a fixed viewpoint and lighting position. Our method is very simple, yet fills the niche between pure 2D texture synthesis methods and full BTF synthesis.

1 Introduction

Rendering objects realistically has been one of the primary goals of computer graphics research for many years. Typically, generating results that look realistic requires knowledge of light-transport and accurate models of materials' bidirectional reflectance distribution functions (BRDFs). Recently, image based rendering techniques have come into favor since using real images for rendering facilitates the synthesis of realistic renderings provided that the images are sampled appropriately. However, these methods could be complicated as well. Our goal is to allow users to create realistic renderings without much difficulty.

This paper presents a technique for making untextured and unlit 3D models look perceptually realistic. The algorithm starts with a single photograph of a sphere or ellipsoid of the desired source material. This photograph is then used to produce realistic renderings of models from specified static viewpoints as if they were made of the same material. We sample from the photograph of the material coherently and with respect to the 3D surfaces of the source material and the model to be rendered to produce a rendering from a desired viewpoint. This rendering is produced without any knowledge of surface material properties or scene lighting. Our technique, in essence, treats the input photograph of the material as a bidirectional texture function (BTF) sample, and produces a rendering of the model from the same lighting direction as a result. While BTFs have usually been used to render surfaces from multiple viewpoints and lighting directions, we have found this representation useful for single computationally inexpensive renderings from static viewpoint and lighting directions as well. Such renderings would be valuable to users as they could easily produce realistic renderings with no knowledge of computer graphics principles using only a digital camera and a simple user interface.

Figure 1: *The Stanford bunny rendered as kiwi*

2 Previous Work

Our work treats a single photograph of a source material as a BTF and samples from it to synthesize a rendering. Liu *et al.*[7] sample from BTFs for rendering as well, however, in their work, a set of BTF examples is used to approximate the complete 6D space to produce rectangular surfaces that can be texture mapped. Recently, Tong *et al.*[12] showed a method for sampling a BTF example set to cover arbitrary 3D surfaces instead. However, since they are interested in rendering models from arbitrary lighting/viewing directions, their technique is time intensive. Our approach uses only a single BTF exemplar for producing renderings that are computationally inexpensive. However, as a result, we can only produce renderings from fixed lighting/viewing directions.

Polynomial texture maps [8] (PTMs) approximate material appearance with photographs as BTFs do. However, the viewing direction is held constant to reduce the dimensionality of the space from 6D to 4D. A special apparatus is used to capture many images of a material under different lighting conditions. These images are then interpolated using a polynomial at each texel to yield a continuous space. PTMs are great for texturing models but they are hard to acquire. PTMs are most useful in enhancing the contrast of captured materials and in interpolating image parameters smoothly (*e.g.* images with different focus or taken at varying times of day).

Our work is also similar to the lit sphere work of Sloan *et al.*[11]. In their work, a drawing is sampled by hand to produce a drawing of a sphere lit from a fixed direction. They match the normals from the 3D model and lit sphere using nearest neighbor at each rendered pixel to produce a rendering of the model in the same style. The lit sphere produced by the user in their system is essentially a BTF exemplar for a single lighting/viewing direction. However, nearest neighbor matching alone is not sufficient for realistic rendering since the material may contain high-frequency albedo that must be present in the rendering as well. We sample from the surface coherently while matching the curvature of the material and object to produce a realistic rendering so that high frequency albedo is not lost while producing the rendering. Hertzmann and Seitz [6] have worked on the inverse version of this problem. They match reflectance from illuminated spheres to infer the 3D shape of a real object from an image. Spheres can also be used to compute inverse global illumination to synthesize realistic images for compositing. Debevec [2] uses high dynamic range images of a chrome sphere to measure scene radiance and then uses the measured radiance to add new objects to the scene with correct lighting.

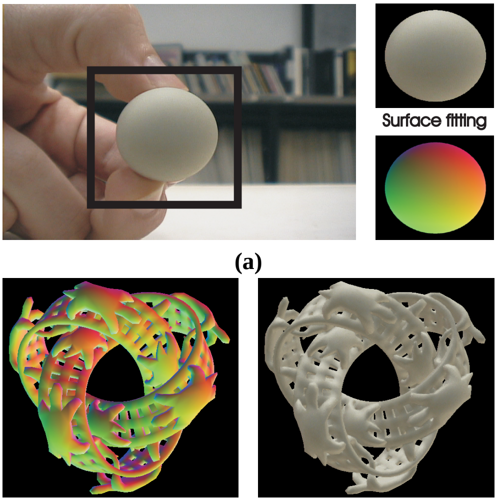
Recently, there has been work on performing texture synthesis in 3D over a model's surface [9, 13, 16, 15]. These methods use an input texture image to provide seamless coverage of the entire 3D surface. However, even if the input image is of a photograph, these methods may not produce highly realistic results. The lighting of the textured model may not be coherent across the entire model since the surface curvature of the model may not match that of the texture patches used to cover it.

Our results appear similar to the 2D texture transfer results presented by Efros and Freeman [3]. However, the major difference is that we can achieve similar results on arbitrary 3D models instead of images. Our approach uses unlit, untextured 3D models and does not require the lighting in the scene to be known. Since the 3D models are unlit, we must simultaneously match the input material's albedo and lighting coherently over the surface of the model in the rendered view. Our approach is also similar to the texture by numbers application of image analogies presented in [5]. In the texture by numbers application, landscapes are the input so they can be approximated by gradient 'planes'. However, arbitrary 3D surfaces cannot be described that simply.

It is also worth noting the work of Hamel and Strothotte [4] since our work is similar in spirit. In their work, a model is rendered with a non-photorealistic renderer, and then has its rendering 'style' transferred to a new model. Various 2D representations of both 3D models such as curvature and shadows, as in G-buffers [10], are used to perform the transfer . We use similar feature vectors to transfer material appearance rather than style.

3 Motivation

The material image used by our system is in essence a single bidirectional texture function (BTF) image. A BTF is a mapping from the 4D space of viewing and lighting directions (each parameterized by



(a)

(b)

Figure 2: (a) Target material is cropped from photograph, fitted with 3D ellipsoid, and rendered with encoded normals, (b) Input 3D surface, left: encoded normals, right: result.

a pair of tilt and azimuth angles) to the space of images:

$$\tau : L \times V \rightarrow I \qquad (1)$$

where L and V are lighting and viewing directions respectively. Since the photograph provides an exemplar of a material's appearance under a particular viewing and lighting direction, if it is mapped properly onto a model as texture, realistic renderings would be producible. However, such a mapping would cause a warping/loss of detail in areas of high surface curvature, and could cause high frequency albedo in the image to become incoherent. Sampling from a sphere or ellipsoid addresses these issues. The key intuition of our approach is that since these shapes provide coverage of the complete set of unit normals and are assumed to be the same scale as the 3D models, they can be used as a good estimate for how a complex surface would be illuminated if made of the same material.

Our approach works because the lighting, but not the albedo, varies with the surface normal. However, the albedo must be correct in synthesized renderings. For simple untextured material, such as colored plastic without specular highlights, matching normals alone results in plausible renderings and the lit sphere technique [11] would be sufficient. For a textured material, such as an orange, this approach will not yield good results. This is because matching normals alone will cause the high frequencies of the orange albedo to mismatch, producing a result that resembles noise.

4 Method

Our algorithm uses a digital photograph of a sphere or ellipsoid of the source material for the input 3D model. Our pipeline is shown in Figure 2. Since ellipsoids are such smoothly varying surfaces, we can fit 3D surfaces through the boundaries of the material photograph. The ellipsoids are fit by hand by warping a rendering of a sphere with encoded normals so that the width and height match those of the photograph using a commercial image processing program.

We then encode the normals of the fitted 3D ellipsoid and the 3D model as colors to produce two images; one of the fitted surface material, and one of the 3D model from the desired viewpoint. Finally, we coherently sample from the photograph of the ellipsoid material by using the two encoded normal images.

4.1 Image synthesis

We use normals to match in rendering since a sphere or ellipsoid provides coverage of the complete set of unit normals. In addition, if it is of a similar scale as the 3D model to be rendered, using a sphere or ellipsoid as a 'stand-in' for the reflectance of a more complex object is justified.

We call the rendering of a model to be the rendering of its normals color-coded, *i.e.* (R,G,B) maps to (X,Y,Z) and the range (0, 255) maps to (-1,1). We perform this encoding so that we can perform matching in 2D (the encoded normal image) instead of over the 3D surface itself to avoid having to parameterize the surface.

The normals are matched between the rendered 3D model and the rendered sphere or ellipsoid model. Simple nearest neighbor is not sufficient for realistic rendering since the material may contain high-frequency albedo that must be present in the rendering as well. We use small local surface neighborhoods and multi-resolution pyramids in match-

ing. This ensures that we preserve both small and large scale surface and albedo details while matching.

We match the normals of the input model with those of the material ellipsoid while respecting the local albedo that has already been matched at the current and previous pyramid levels while matching from coarse to fine levels. We use image analogies [5] as our matching algorithm since it alternates between nearest causal neighbor matching [14] and coherence matching [1] if the nearest neighbor would not be coherent with previously matched results. In terms of analogy, the 3D rendering of the material ellipsoid is to the photograph of it as the 3D rendering of the model is to the output rendering.

We describe the image analogies matching algorithm briefly now, but refer readers to the image analogies paper for more details. Feature vectors are constructed for each pixel at each pyramid level of the training images. The feature vector for a pixel (x, y) on pyramid level i consists of the 3x3 neighborhood at level $i-1$, and the full 5x5 neighborhood or causal [14] (half of a 5x5) L-shaped 12 pixel neighborhood centered around pixel (x, y) at level i, depending on whether the input (encoded normals) or output (photograph) image is being sampled. These features are computed for both the training input and output images and concatenated, resulting in a 165 dimensional feature vector ((3x3 + 5x5 + 3x3 + 12) pixels $*$ 3 channels).

The matching algorithm then proceeds as follows. A Gaussian pyramid is constructed for the input rendering (with encoded normals) of the model to be textured. For each pixel at each pyramid level, progressing in scanline order beginning at the coarsest pyramid level, a feature vector is constructed for the pixel. At the topmost level of the pyramid, the nearest neighbor training feature vector is selected and its corresponding pixel value in the training output image is copied to the appropriate pixel position in the output rendering that is being synthesized. At every other pyramid level, the nearest neighbor is computed along with the most coherent candidate pixel [1], and the minimum is chosen. The training feature vector that is chosen is used to select which pixel is copied from the training output image to the current pixel position in the synthesized output image at the current pyramid level. This is done for every pixel at every level, resulting in a synthesized textured rendering of the model.

Using coherent matching, we can handle textured materials without distorting the albedo when sampling. We also use tree structured vector quantization to speed-up the feature vector matching. In addition, background pixels are skipped in matching and are not part of the training set to speed up computation. The steps of our approach are:

1. Render a 3D model with its normals encoded as colors along with a visibility mask
2. Render a 3D sphere or ellipsoid with its normals also encoded as colors
3. Warp the rendered material surface image so that it lines up with the surface in the photograph of the source material
4. Match all normals in the 3D model rendering with the material's normals coherently with respect to the albedo in scanline order
5. Use the visibility mask to crop out any regions where the rendering does not match the silhouette of the 3D model

Figure 2 pictorially describes the pipeline. We use the visibility mask produced by our renderer to crop out regions where image analogies does not follow the rendered object's silhouette. It is possible for the matching algorithm to not follow the silhouette perfectly because the coherence matching is pixel and not silhouette based. We apply the mask to correct the model's silhouette and holes if necessary, though this is not usually a problem. We can then composite the result onto a photograph of the source material's scene if desired using any image manipulation program (*e.g.* Adobe Photoshop, Gimp, etc.)

5 Results

We have run the algorithm on different combinations of lighting conditions, materials, and models. Our algorithm produces results in 5-10 minutes on a 2Ghz Intel Xeon on images with 16-20k pixels. In Figure 3 we show several results with different models, materials, and lighting. The method produces realistic results for these textured target materials. Details such as lumps in the clay and the groove in the tennis ball are preserved. The lumps in Figure 3 (d) are not exact copies of entire lumps; some are combinations of several lumps. This prevents the result from looking like a rearranged version of the source material. The high frequency ma-

terials are sampled coherently while matching the varying surface curvature and holes in the models.

Figure 4 demonstrates that the algorithm is able to light the rendered models plausibly without any knowledge of the lights in the scene. In addition, the results are very different for the two materials. The specular highlights are successfully transferred from the source materials onto the model. Coherent matching instead of straight nearest neighbor results in highlights that are similar but not exact copies of those in the source images. We used a stand-in object to produce a shadow that we then composited the renderings over to show the degree of realism achievable with this approach. The method produces images which composite very well onto the background plate. In Figure 5 we show results taken from materials with no high frequency content. Our technique is able to produce good renderings of these materials, even when some of the source photographs are out of focus.

These results show that high-frequency albedo is preserved as expected. Our approach does well with a large number of materials, however some, such as the tennis ball material in Figure 3 (a) can be problematic. These materials are difficult because multiple properties must be matched simultaneously on the material object and only a single image is present. As a result, the algorithm trades off between following some high-frequency detail like the tennis ball grooves and following the model's surface. This could be remedied by using multiple reference images of the same object from different orientations but with the same illumination to provide additional samples for each normal in matching.

6 Conclusion and Future work

We have presented a method for rendering 3D models realistically that fills the niche between pure texture synthesis and full BTF synthesis. Our method takes advantage of the fact that the geometry of a photographed sphere or ellipsoid material sample is known. Since the geometry is known, we can sample from the photograph in rendering a realistic image of the 3D model from a given viewing position.

We have shown results demonstrating that our approach works for a number of different material types, lighting conditions, and models. In addition, our approach implicitly captures light direction as well, without explicitly knowing where light sources are or how many are present in the scene. Our method combines ideas from rendering, texture synthesis, and machine learning to easily create imagery that looks very realistic, even by novice users.

There are many exciting avenues of future research that we are interested in. We are interested in using this approach to produce animations. Currently, any animations produced by our approach would be temporally incoherent because temporal variations of the surface are not taken into account. We would also like to be able to move the models and light sources such that the models continue looking realistic. This will require more information in the feature vector to encode other important information such as light source positions. The matching algorithm will probably have to be modified as well so that specular highlights do not pop on or off the model's surface and so that cast shadows are correct in addition to attached shadows, which our approach already handles. It might also be useful to have additional information in the feature vector, such as distance to the camera (or other G-buffer properties), user specified constraints for highly textured material, or skeleton information of the model [4]. By making the feature vector more descriptive, its dimensionality will be increased, but with the introduction of multiple viewpoints, such information could make it possible to use this algorithm as a basis for 3D interaction.

References

[1] M. Ashikhmin. Synthesizing natural textures. *2001 ACM Symposium on Interactive 3D Graphics*, pages 217–226.

[2] Paul Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proceedings of SIGGRAPH 98*, pages 189–198.

[3] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. *Proceedings of SIGGRAPH 2001*, pages 341–346.

[4] J. Hamel and T. Strothotte. Capturing and re-using rendition styles for non-photorealistic rendering. *Computer Graphics Forum*, 18(3):173–182, September 1999.

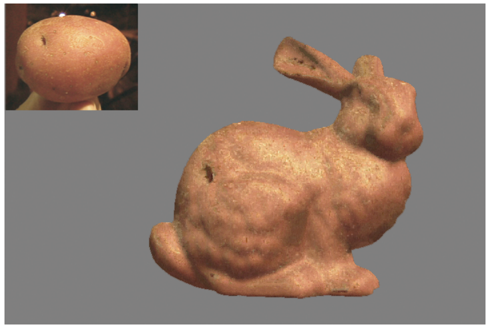
[5] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin. Image analogies. *Proceedings of SIGGRAPH 2001*, pages 327–340.

[6] A. Hertzmann and S.M. Seitz. Shape and materials by example: A photometric stereo approach. In *CVPR03*.

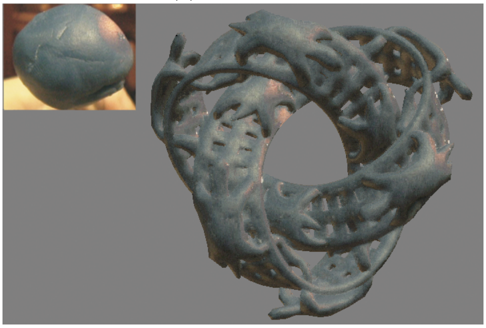
[7] X. Liu, Y. Yu, and H. Shum. Synthesizing bidirectional texture functions for real-world surfaces. *Proceedings of SIGGRAPH 01*, pages 97–106.

[8] T. Malzbender, D. Gelb, and H. Wolters. Polynomial texture maps. *Proceedings of SIGGRAPH 01*, pages 519–528.

(a) Tennis ball

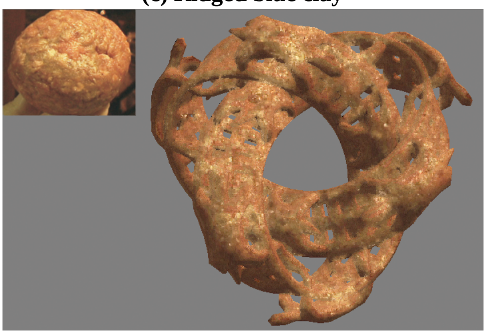


(b) Red potato

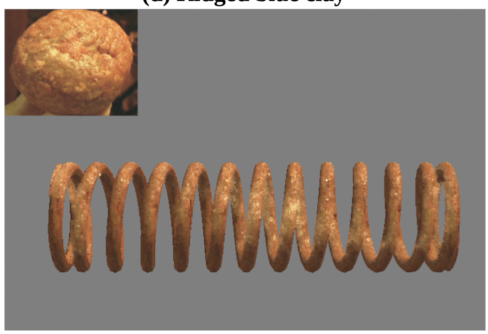


(c) Ridged blue clay

(d) Ridged blue clay



(e) Ground beef



(f) Ground beef

Figure 3: Results of the algorithm run on different materials, lighting conditions, and 3D models.

[9] E. Praun, A. Finkelstein, and H. Hoppe. Lapped textures. *Proceedings of SIGGRAPH 1998*, 465-470.

[10] T. Saito and T. Takahashi. Comprehensible rendering of 3-d shapes. *Proceedings of SIGGRAPH 90*, pages 197–206.

[11] P.-P. J. Sloan, W. Martin, A. Gooch, and B. Gooch. The lit sphere: A model for capturing npr shading from art. In *Graphics Interface 2001*, pages 143–150.

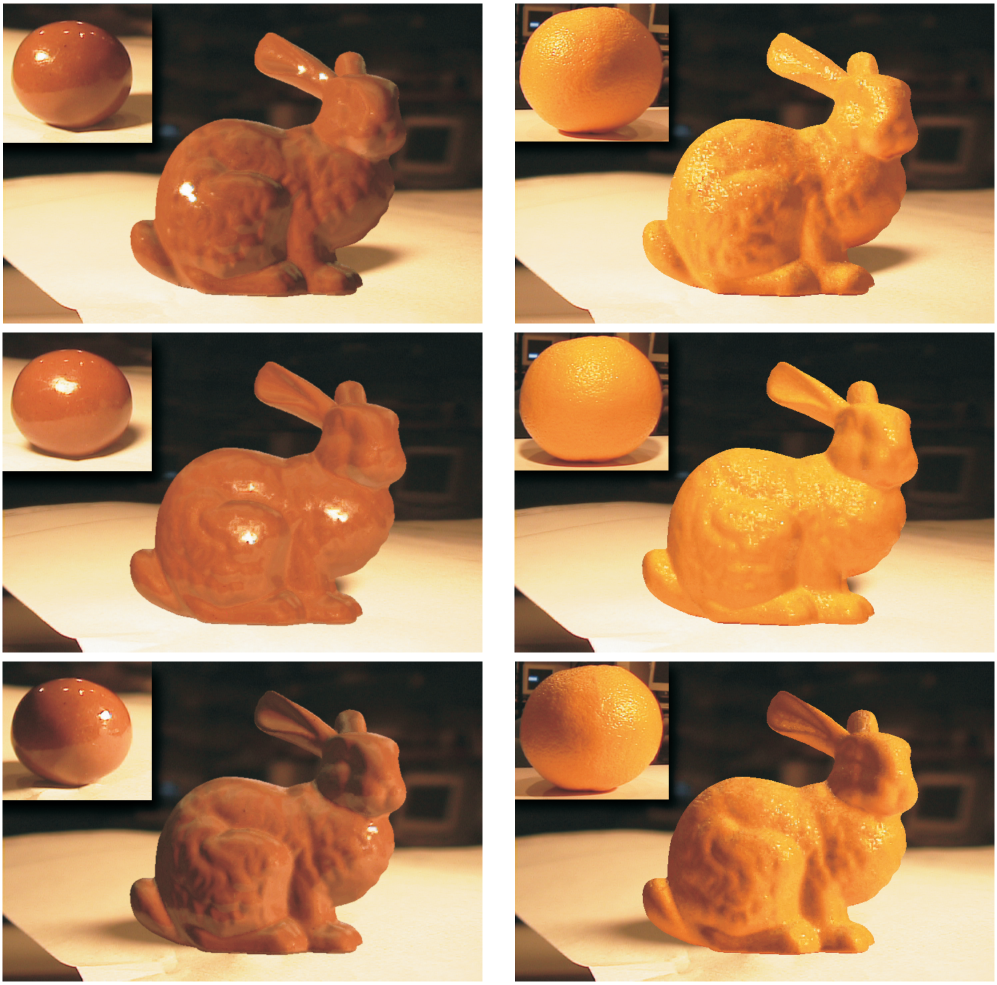
[12] X. Tong, J. Zhang, L. Liu, X. Wang, B. Guo, and H. Shum. Synthesis of bidirectional texture functions on arbitrary surfaces. *ACM Transactions on Graphics*, 21(3):665–672, July 2002.

[13] G. Turk. Texture synthesis on surfaces. *Proceedings of SIGGRAPH 2001*, pages 347–354.

[14] L. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. *Proceedings of SIGGRAPH 2000*, pages 479–488.

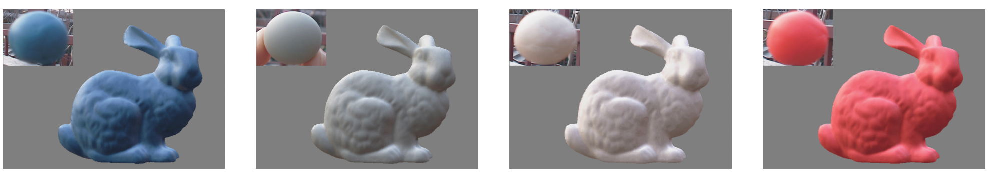
[15] L. Wei and M. Levoy. Texture synthesis over arbitrary manifold surfaces. *Proceedings of SIGGRAPH 2001*, pages 355–360.

[16] L. Ying, A. Hertzmann, H. Biermann, and D. Zorin. Texture and shape synthesis on surfaces. In *12th Eurographics Workshop on Rendering*, pages 301–312, June 2001.



(a) Chocolate **(b) Orange**

Figure 4: Renderings of the Stanford bunny using chocolate and orange as materials under varying illumination. The shadow is of the original object in the scene, not of the model. Inset in each image is the material photograph that was sampled to produce the rendering.



(a) Blue clay **(b) Mouse ball** **(c) White clay** **(d) Red clay**

Figure 5: Renderings of the Stanford bunny using various materials photographed outdoors.