GENERATING HIGH-RESOLUTION TEXTURES FOR 3D VIRTUAL ENVIRONMENTS USING VIEW-INDEPENDENT TEXTURE MAPPING

Charalambos Poullis, Suya You, Ulrich Neumann

University of Southern California
Integrated Media Systems Center
Charles Lee Powell Hall
3737 Watt Way
Los Angeles, CA 90089

ABSTRACT

Image based modeling and rendering techniques have become increasingly popular for creating and visualizing 3D models from a set of images. Typically, these techniques depend on view-dependent texture mapping to render the textured 3D models in which the texture of novel views is synthesized at runtime according to different view-points. This is computationaly expensive and limits their application in domains where efficient computations are required, such as games and virtual reality. In this paper we present an offline technique for creating view-independent texture atlases for 3D models, given a set of registered images. The best texture map resolution is computed by considering the areas of the projected polygons in the images. Texture maps are generated by a weighted composition of all available image information in the scene.Assuming that all surfaces of the model are exhibiting Lambertian reflectance properties, ray-tracing is then employed, for creating the view-independent texture maps. Finally, all the generated texture maps are packed into texture atlases. The result is a 3D model with an associated view-independent texture atlas which can be used efficiently in any application without any knowledge of camera pose information.

1. INTRODUCTION

Virtual reality and gaming applications often require and employ realistic 3D models of real world scenes. In addition to the 3D geometry, texture information from the images is also used to enhance the visual richness of the models. Advances in image-based modeling and rendering techniques have made it possible to reconstruct a 3D model of the scene using a set of images. The images can be used to texture map the models, thus requiring that the camera poses are known a priori. Different views of the scene can then be synthesized from the images at runtime for each novel viewpoint. The scene can then be viewed from novel view-points by combining the texture information in the images. However, this becomes a problem when the model is intended to be used in applications such as games, virtual reality or feature films. In such applications, the texture maps are required to be precomputed and be independent of the viewing directions of the cameras used to generate the textures and/or models. Hence, no per-pixel calculations should be performed in real-time to determine a pixel color.

The work in this paper introduces a new technique to generate a composite texture map from a set of images. Multiple images taken from different view-points are combined together based on weights computed using different criteria. The texture maps are then packed into texture atlases in order to reduce computation time and improve performance. The result is a standalone 3D model with its associated texture atlases which can be imported in a variety of different applications,without any knowledge of camera pose information.

This paper is organized as follows: In Section 2, we discuss the related work in this area. In Section 3, a brief overview of the proposed system is presented followed by Section 4, the pre-processing. Section 5 and 6 describe how the composite texture maps are generated and finally packed into texture atlases. Experimental results are presented in Section 7 followed by a discussion of our future work.

2. BACKGROUND AND RELATED WORK

One of the most popular and successful techniques is the one introduced by [1] which uses a small set of images to reconstruct a 3D model of the scene. View-dependent texture mapping(VDTM) is then performed for the computation of the texture maps of the model. By interpolating the pixel color information from different images new renderings of the scene can be produced. The contributions of each image to a pixel's color is weighted based on the angle difference between the camera's direction and the novel view-point's direction. For example, if an image was taken from a similar direction as the novel view-point, it will have a greater contribution than another image taken from a direction not close to the view-

1-4244-1017-7/07/$25.00 ©2007 IEEE

point's direction. In [2] the authors present how VDTM can be efficiently implemented using projective texture mapping, a feature available in most computer graphics hardware. Although this technique is sufficient to create realistic renderings of the scene from novel view-points its computation is still too expensive for real-time applications, like games or virtual reality.

A different approach is proposed in [3] to seamlessly map a patchwork of texture images onto an arbitrary 3D model. By specifying a set of correspondences between the model and any number of texture images the system can create a texture atlas. In [4] they order geometry into optimized visibility layers for each photograph. The layers are subsequently used to create standard 2D image-editing layers which become the input to a layered projective texture rendering algorithm. In [5, 6] reflectance properties of the scene were measured and lighting conditions were recorded for each image taken. An inverse global illumination technique was then used to determine the true colors of the model's surfaces. A method for creating renders from novel view-points without the use of geometric information is presented in [7], where densely regularly sampled images are blended together. [8] estimates a set of blending transformations that minimizes the overall color discrepancy in overlapping regions in order to deal with the unnatural color texture fusion due to variations in lighting and camera settings. The authors in [9] propose a method for computing the blending weights based on a local and global component. This results in smooth transitions in the target image in the presence of depth discontinuities. A slightly different approach for texture generation and extraction is proposed in [10]. Given a texture sample in the form of an image, they create a similar texture over an irregular mesh hierarchy that has been placed on a given surface.

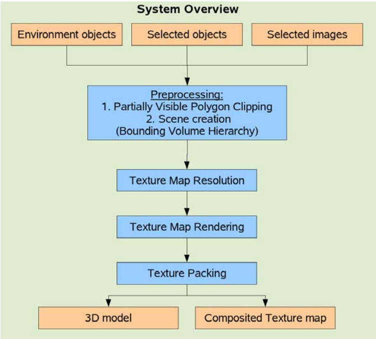


Fig. 1. System Overview

3. SYSTEM OVERVIEW

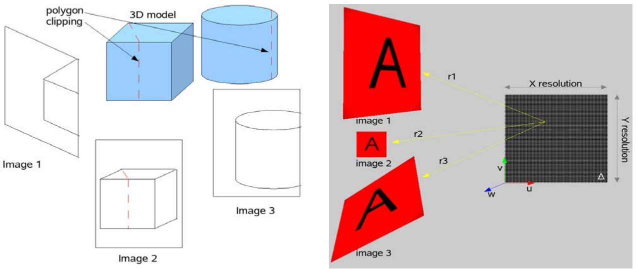
The proposed system begins with pre-processing the input data. The data consists of a 3D model and a set of registered images. Using the image and camera information the texture map resolution for each surface is computed. The weighted contributions from the multiple images are then composed into a single texture map. These texture maps are then packed into texture atlases. Figure 1 shows a visual overview of our system.

An assumption in our work is that the 3D objects exhibit lambertian reflectance properties. In contrast to a specular surface, a lambertian surface is independent of the viewing direction. Thus, if a lambertian surface is viewed from different angles it will respond the same under constant lighting conditions.

4. SCENE PRE-PROCESSING

The input to the system is a set of 3D scene objects and a set of registered images along with their associated camera poses. The user has the option of specifying for which objects to compute texture maps i.e. *selected objects*, and which objects should only be used for visibility and occlusion tests i.e. *environment objects*. Similarly, the user has the option of specifying which images to be used in the process.

The *selected* and *environment* objects are then subdivided into the primitives used by our ray-tracer, namely triangles and quadrilaterals. For each image, a visibility check is performed for all the surface patches of the *selected objects* and the following actions are taken. A surface patch remains unchanged if is entirely visible in the image. A partially visible surface patch is clipped at the boundary of the projection of the image plane, and a non-visible patch remains unchanged. In figure 2(a) surface patches are being projected in different image planes. Once all the surface patches of the selected objects are tested for visibility and clipped accordingly, a bounding volume hierarchy is created as the internal structure for the ray-tracing.



(a) Visibility-based clipping

(b) Map resolution

Fig. 2. Polygon clipping and resolution computation.

5. TEXTURE MAP RESOLUTION

A surface may appear in multiple images with different resolutions. To ensure minimal information loss, the resolution of a texture map is chosen to be the size of the projected polygon with the largest area in image space. To determine this, each surface is projected into all the image planes and the area of each projected polygon is computed. The dimensions of the polygon with the largest area are chosen as the resolution of the texture map for the surface. Surfaces which are backfacing or are not visible in any of the images are automatically assigned a default color and any further texture processing is stopped.

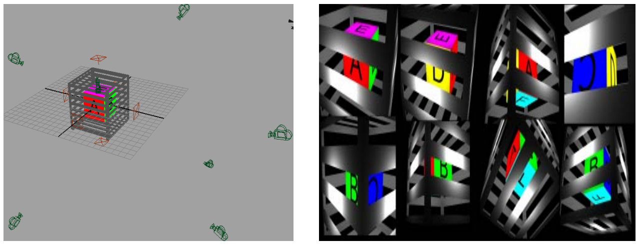
6. TEXTURE MAP RENDERING AND PACKING

Ray-tracing is employed to render the composite texture maps of the surfaces. For each surface Δ in the scene with computed resolution of (X, Y), a local coordinate system $U\hat{V}W$ is created. The surface Δ is then sub-divided into X cells in the $\vec{U}$ axis and Y cells in the $\vec{V}$ axis.

The ray-tracer then casts a ray from each point on the surface to the image planes, in order to determine the point's color. A point P is visible in an image I if there is no other surface intersecting the ray $\hat{r}$ casted from that point P to the image plane of image I. If the point is visible in an image then the corresponding pixel color is retrieved. The contribution of this color to the color of P is then weighted using the following criteria:
1. The distance of the pixel to the image boundary. Pixels which are close to the edges receive a lower weight than pixels located in the middle. This is required in order to hide any stiching effects between images, and reduce the artifacts introduced when blending multiple images.
2. The angle between the camera direction and the surface normal. Images which were taken from oblique angles are down-weighted, since they experience higher degree of perspective distortion.
3. The distance of the pixel from the principal point. Pixels which are further away from the principal point exhibit higher radial distortion. Although the radial distortion coefficients can be closely approximated and used to undistort the image, we have found that it is more likely to have misalignments between images at points further away from the principal point.

Figure 2(b) shows a surface being ray-traced from the original images. The texture map resolution was determined earlier as explained in Section 5.

The computed texture maps are then sorted based on their resolution and packed into a texture atlas. The process changes the texture space of each map therefore requiring the recalculation of texture coordinates for the models in the scene. The texture atlases can also be easily edited by an artist if necessary to add or remove color information. Figure 4(a) shows an example of packed texture maps into five texture atlases.

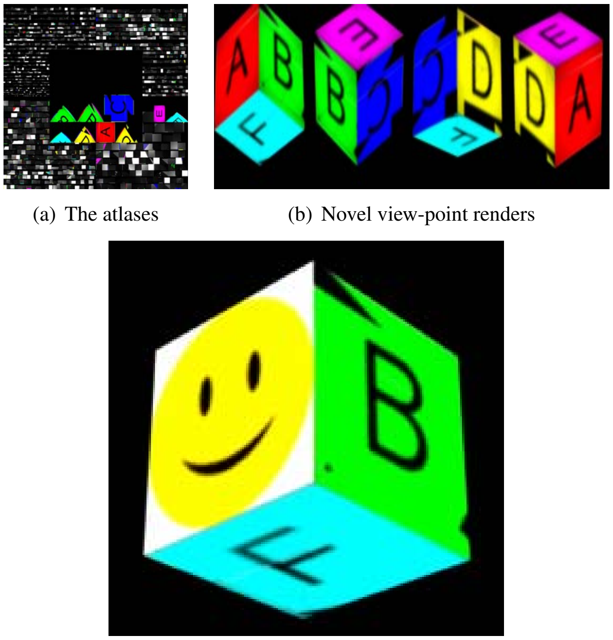

(a) Test scene setup

(b) The 8 input images

Fig. 3. Test case

7. RESULTS

A test scene is setup as shown in figure 3(a). The geometry consists of a cube which is enclosed in a "cage" and a set of images taken from several different angles as shown in figure 3(b). In this example not all surfaces are entirely visible within a single image. The surface materials exhibit lambertian reflectance properties and six area lights are used to lit the scene. The generated texture atlases are shown in figure 4(a). The "cage" is then removed and the cube is rendered from novel view-points using the texture atlases as shown in figure 4(b). As expected, information from all the images is combined together and occlusions created by the cage are filled-in. The atlases can also be edited to add,remove or change color information as shown in figure 4(c).


(a) The atlases

(b) Novel view-point renders

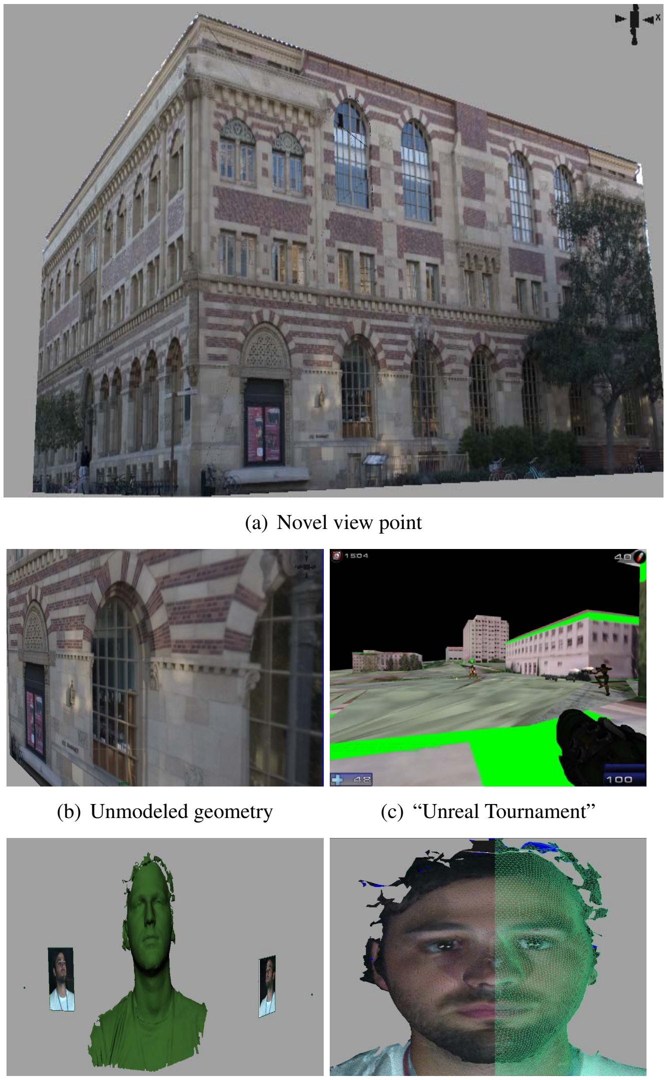
(c) Texture atlas editing

Fig. 4. The output for the test case. The black color corresponds to areas where no texture information was available.

The results using a building model from the USC campus are shown in figure 5(a). The textures were generated using a set of images which was captured and registered to the model. Figure 5(b) shows the perspective distortion effects produced in the case where not all the geometry in the scene is modeled. In such cases the generated texture will not look realistic from all vantage points. This problem can be overcome if the images are taken from a direction perpendicular to the surfaces of the object.

Another example is shown in figure 5(c) where the generated textures of our system and the 3D model are used to create a 'level' in a gaming application. Surfaces which are not visible by any camera and therefore have no texture information, are displayed in green.

The setup for the last example is shown in figure 5(d) with two cameras capturing the geometry from left and right. Figure 5(e) shows the result of a novel viewpoint render.



(a) Novel view point

(b) Unmodeled geometry

(c) “Unreal Tournament”

(d) 3D geometry and cameras

(e) Novel view-point

Fig. 5. Real scene examples.

8. CONCLUSION AND FUTURE WORK

We have presented a technique for creating view-independent texture mapped objects. Unlike other existing techniques,for example view-dependent texture mapping, with our technique it is possible to create a single texture atlas consisting of all available information from the images. The model can then be used as-is without having to perform per-pixel computations based on new viewpoints in order to compute each pixel's color.

In this work we have dealt only with lambertian surfaces. In the future we would like to extend this to work with non-lambertian surfaces by also considering the lighting conditions of the environment.

9. REFERENCES

[1] Paul Ernest Debevec, *Modeling and rendering architecture from photographs*, Ph.D. thesis, University of California, Berkeley, 1996.

[2] Paul Debevec, Yizhou Yu, and George Boshokov, “Efficient view-dependent image-based rendering with projective texture-mapping,” Technical Report CSD-98-1003, University of California, Berkeley, May 20, 1998.

[3] Kun Zhou, Xi Wang, Yiying Tong, Mathieu Desbrun, Baining Guo, and Heung-Yeung Shum, “Texturemontage,” *ACM Trans. Graph.*, vol. 24, no. 3, pp. 1148–1155, 2005.

[4] Alex Reche Martinez and George Drettakis, “View-dependent layered projective texture maps,” in *Pacific Conference on Computer Graphics and Applications*. 2003, pp. 492–496, IEEE Computer Society.

[5] Paul Debevec, Chris Tchou, Andrew Gardner, Tim Hawkins, Charalambos Poullis, Jessi Stumpfel, Andrew Jones, Nathaniel Yun, Per Einarsson, Therese Lundgren, Marcos Fajardo, and Philippe Martinez, “Estimating surface reflectance properties of a complex scene under captured natural illumination,” Technical report, University of Southern California, ICT, 2004.

[6] C.Poullis, A.Gardner, and P.Debevec, “Photogrammetric modeling and image-based rendering for rapid virtual environment creation,” *Proceedings of ASC2004*, 2004.

[7] Marc Levoy and Pat Hanrahan, “Light Field Rendering,” in *Computer Graphics Proceedings, Annual Conference Series, 1996 (ACM SIGGRAPH '96 Proceedings)*, 1996, pp. 31–42.

[8] Nobuyuki Bannai, Alexander Agathos, and Robert Fisher, “Fusing multiple color images for texturing models,” Tech. Rep. EDIINFRR0230, The University of Edinburgh, July 2004.

[9] Ramesh Raskar and Kok-Lim Low, “Blending multiple views,” in *Pacific Conference on Computer Graphics and Applications*. 2002, pp. 145–155, IEEE Computer Society.

[10] Greg Turk, “Texture synthesis on surfaces,” in *SIGGRAPH*, 2001, pp. 347–354.