Visualizing Building Interiors using Virtual Windows

Norman Moses Joseph Brett Achorn Sean D. Jenkins Hank Driskill
Walt Disney Animation Studios *†

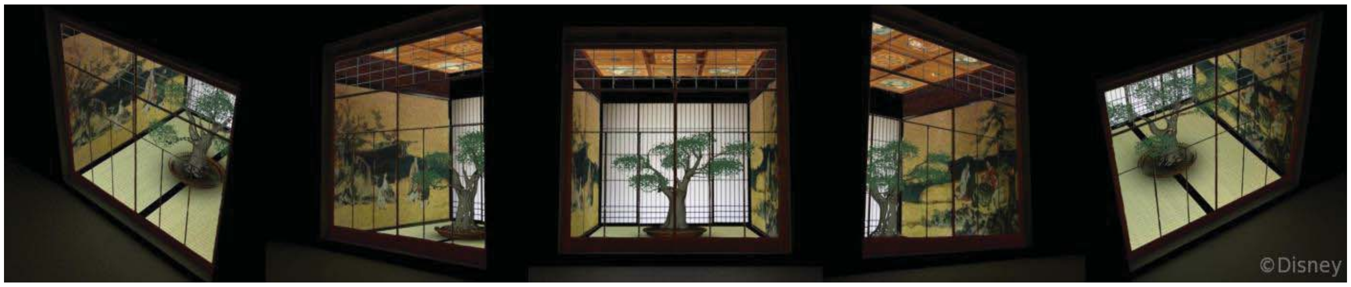


Fig. 1 *Example room used for evaluation of the Virtual Window Shader as seen from different camera views*

1. Abstract

The feature film "Big Hero 6" is set in a fictional city with numerous scenes encompassing hundreds of buildings. The objects visible inside the windows, especially during nighttime, play a vital role in portraying the realism of the scene. Unfortunately, it can be expensive to individually model each room in every building. Thus, the production team needed a way to render building interiors with reasonable parallax effects, without adding geometry in an already large scene. This paper describes a novel building interior visualization system using a Virtual Window Shader (Shader) written for a ray-traced global illumination (GI) multi-bounce renderer [Eisenacher et al. 2013]. The Shader efficiently creates an illusion of geometry and light sources inside building windows using only pre-baked textures.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism; I.6.m [Simulation And Modeling]: Miscellaneous; Game; Production Pipeline; Rendering; Building Interior Visualization.

Keywords: shader, global illumination, building interior visualization, parallax mapping, texture mapping, volume rendering, depth, pre-baked textures.

2. Introduction

Large urban environments present a challenge in terms of size and detail. Most are too big to add geometry for the interiors of the buildings, while maintaining reasonable scene complexity. In the past, various techniques to visualize interiors of buildings have been used to avoid the need to model them in their entirety [Seymour 2005; Seymour 2012].

Since early developments in texture mapping [Catmull 1974], various techniques have been used to add details to geometry including: bump maps [Blinn 1978], normal maps and displacement maps [Oliveira et al. 2000]. To address the change in perspective and show correct parallax, a displacement map can emulate the actual geometry using depth values. However, as mentioned by van Dongen [2008], displacement maps do not support texturing surfaces perpendicular to the original polygon, which can lead to texture stretching. An efficient method for adding detail to geometry, parallax mapping, introduced by Kaneko et al. [2001] and further extended by Welsh [2004], suffers from undesirable distortion as a result of its approximation.

Like the Interior Mapping algorithm [van Dongen 2008], the Shader can be used on procedurally-generated buildings such as those mentioned in Instant Architecture [Wonka et al. 2003] and Procedural Modeling of Buildings [Muller et al. 2006]. Since many of these building systems do not generate interior geometry, the Shader can be used to add virtual interiors without adding more geometry. Furthermore, the Shader can also handle cases of windows at the corners of buildings.

The Shader provides user control by allowing the artist to define, for each building, which virtual interior applies to which window. In addition, the artist has the ability to create interior views that appear to span multiple windows of the same room. The Shader uses emissive textures, allowing the artist to bake-out individual light sources in the room which can affect the surrounding environment.

3. The Objective

The objective of the Shader is to:

1. reduce the work of modeling individual rooms.
2. significantly reduce the amount of geometry that needs to be rendered.
3. create the illusion of a high-quality, furnished, and potentially-lit room that holds up to parallax.
4. provide a large degree of artistic freedom.

4. Implementation

4.1 Rendering Virtual Rooms

The Shader can be applied to a simple plane signifying the window opening. From the dimensions of the plane and artist-provided attributes, the Shader constructs a virtual room with the plane of the window as one side.

For each camera-ray hit on the window plane, the ray direction is noted and the ray is extended until it intersects one of the five virtual walls of the room. Upon intersection, the x,y,z coordinates of the intersection point are converted to texture-coordinates by finding the relative location of the point with respect to the plane (or wall) on which it lies. The textures corresponding to the intersected wall are indexed using these texture-coordinates and the resulting color is returned as the value at the hit point on the window plane. Since the room is a virtual cube, a single ray entering the interior of a building can hit only one of the five

*{ norman.joseph | brett.achorn }@disneyanimation.com
†{ sean.jenkins | hank.driskill }@disneyanimation.com

SIGGRAPH Asia 2014, December 03 – 06, 2014, Shenzhen, China.
2014 Copyright held by the Owner/Author.
ACM 978-1-4503-2895-1/14/12
http://dx.doi.org/10.1145/2669024.2669029

virtual walls of the room. The intersection point of the extended ray and the virtual wall is found by a method similar to the closest-intersection-value method done by van Dongen [2008].

Since the Shader takes the view direction into consideration to calculate the color value for each pixel, the resulting rooms maintain correct perspective. This remains true when rendering stereo images for the 3D version of the movie as well.

4.2 Rendering Room Objects

The objects in the room, including furniture and animated characters, play a vital role in the realistic appearance of a building, especially when in proximity to the camera. To add visual complexity without adding actual geometry, the area in the virtual room is divided into slices parallel to the plane of the window. Each slice has a texture representing the contents of the virtual volume and an alpha texture for opacity. Every camera-ray hit on the window plane is extended and the resultant color from each slice at the intersection point with the extended ray is composited together to get the final color of the pixel. Fig. 2 shows a visual representation of the process similar to the volume-rendering methods highlighted by Drebin et al. [1988] and the image-order technique described by Šrámek [2006]. Also, since geometry is represented by mipmapped textures, it can be easily filtered for wide-diameter rays, avoiding aliasing from high frequency geometry.

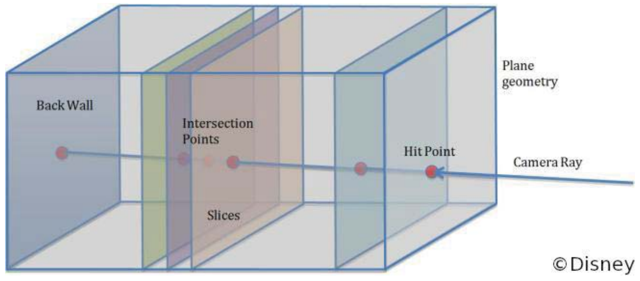


Fig. 2 V*irtual room.*

The textures used on each slice can be generated from painted images or can be rendered ahead of time from a separate scene. The slice textures can then substitute for the room contents. Even large number of objects or characters can be included without increasing the render time. The artist can provide different textures for every frame to create animation using the Shader. Thus, the Shader can be used to depict movement of a character inside the room, adding visual complexity while still not affecting render time.

The Shader allows a user-defined number of slices inside the virtual room. This gives the artist the flexibility to provide as much or as little detail as needed. The artist is able to vary the depth of the room as well as the position of these slices.

4.3 Emission

The Shader calculates the amount of light emitted from the virtual room at the ray hit point. To achieve this, emission values are encoded in the wall and slice textures. For example, the artist can emulate a lamp in the room by painting its emission on the texture and vary the lighting by modifying the Shader attributes such as color intensity. The room-lighting also interacts with the environment in a physically correct manner. This helps create a realistic scene and is particularly useful during nighttime shots, when the rooms in the buildings are more visible.

Unlike a fully-modeled and lit room, which requires subsequent rays to calculate shading, the Shader directly returns emission and does not need to trace further rays. This significantly increases efficiency. Reflections of the room will also carry the correct High Dynamic Range (HDR) intensities. The emission illuminates the rest of the environment. Additionally, the artist can specify an exposure mask to change the light intensities at arbitrary areas in the scene. The artist can add variation by adjusting the exposure mask to simulate turning on one lamp and dimming another.

In the end, the color values and emission values from each intersection point are added and applied at the hit point on the plane as shown in Fig 2.

4.4 Room Variations

Virtual rooms can be varied in many ways. Since the wall and slice textures are independent of each other, different combinations can be used to provide variation. Object looks, emission, slice depth and/or room length can be procedurally varied to give a wider range of choices of floor plans, design and interior illumination. As an added convenience, the slice depth is adjusted automatically when room length is changed. The exposure can be varied across the whole window or just a section by using an exposure mask. Also, the window size does not have to correspond to the outer boundary of the room. The window plane, or "view in", can be smaller than the environment encompassed by the virtual room.

5. Results

Fig. 3 provides just a few of the many variations of the rooms created for the production. It includes both residential-apartment and office-building options.

To evaluate the effectiveness of the Shader, we ran tests to compare renders using it, with renders using original geometry. The Interior Mapping algorithm was slower when rendering a few buildings due to the use of a complex shader. Only when used on more than 90 buildings, each with multiple rooms, did the performance improve over the renders using actual geometry [van Dongen 2008]. The Virtual Window Shader, on the other hand, outperformed actual geometry when comparing renders for even a single room.

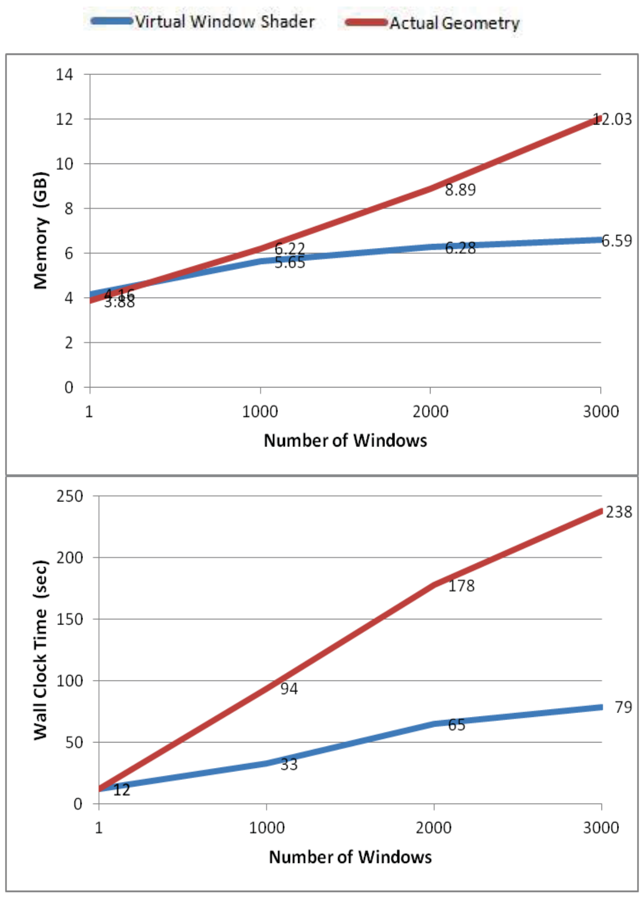
Fig. 1 shows the room used for evaluation of the Shader. The room consists of one object, a tree, along with the five room walls. We used a total of 49 slices in this test-render, 48 to visualize the tree and one extra to represent the window grille. Each of the 49 slices has three texture files: color, alpha and exposure. Each of the five room walls only needs two texture files, since they don't require alpha. Thus, the total number of texture files used for this render is as below:

$$\begin{array}{rl} & 49 \text{ slice texture files} * 3 \\ + & \underline{5 \text{ virtual room wall texture files} * 2} \\ & 157 \text{ texture files total} \end{array}$$

Table 1 shows the number of polygons in the scene using actual geometry as opposed to using the Shader. Graph 1 shows the comparisons of the memory used and the render times. The tests were run using Disney's Hyperion, a new proprietary GI renderer [Eisenacher et al. 2013] with 16 threads, 2 ray bounces and 64 samples per-pixel on a Dell Precision T7600 Unix Workstation with 128 GB RAM. These results only consider render performance; however, we have also seen an increase in performance in scene processing, due to the reduced amount of geometry in the scene.

# of Windows	Actual geometry (# of polygons)	Virtual Window Shader using 49 slices (# of polygons)
1	3124	1
1000	3124000	1000
2000	6251124	2000
3000	9375000	3000

Table 1 *Number of Polygons*



Graph 1 *Render Comparison of Virtual Window Shader vs. Actual Geometry*

6. Extensions - Inverse Virtual Window Shader

Until now, we have considered virtual rooms inside buildings. However, as the Shader creates simplified representations of complex parts of the scene, we could use the Shader to simulate the view outside the window for an indoor render. This can be accomplished by creating the virtual room outside the window to emulate the entire external environment. Thus, we basically replace all the geometry outside the room with the Shader. This can save substantial time and memory in the render and is especially effective for obscured, off-camera or out-of-focus windows, where very-detailed renders of the environment are not required.

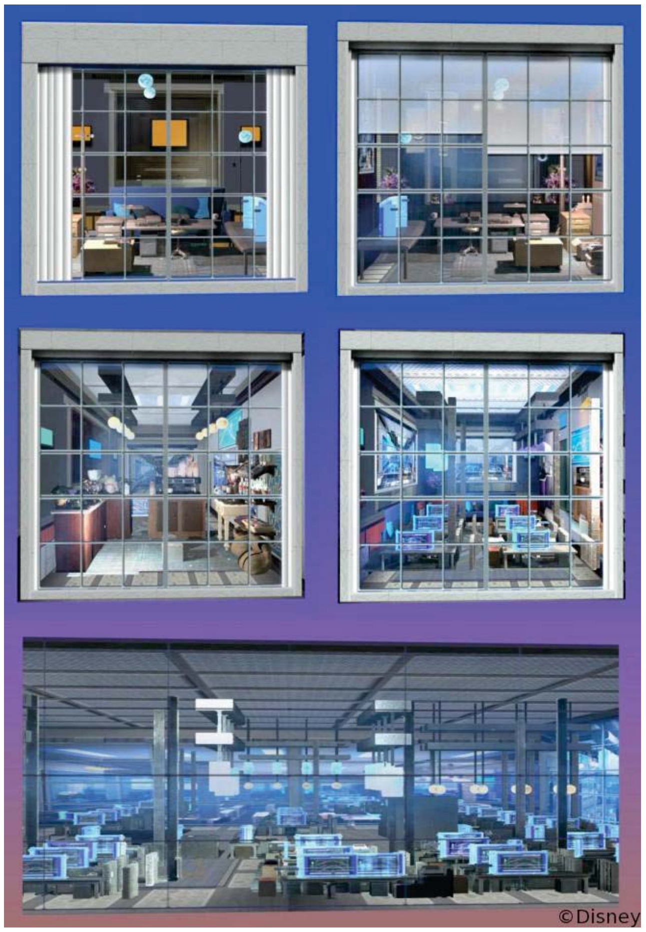


Fig. 3 *Sample Room Variations created using the Virtual Window Shader*

7. Limitations and Future Work

While the Shader is easy to apply to the planes, it takes effort to create the slice textures. To create the slices, the artist could paint them or model the interior in a separate scene and render them. Some modeling effort is also required to place a plane at every window location on the building in order to apply the Shader. We have plans to automate the process of creating slices for given, modeled geometry as well as procedurally placing window planes on buildings.

Though we can render many windows using the same textures and still have variations, the technique does rely on using a significant amount of textures. This may be a limitation in systems constrained by texture memory. In order to reduce the number of texture files, we could combine them to one texture file with different face IDs using a variation of Burley and Lacewell's technique [2008].

As the Shader is inherently a simplification, it could be used as a Level of Detail (LOD) option for far-away geometry. We would like to use stochastic blending to transition from the actual geometry to using the Shader for LOD. This would be particularly useful for real-time applications.

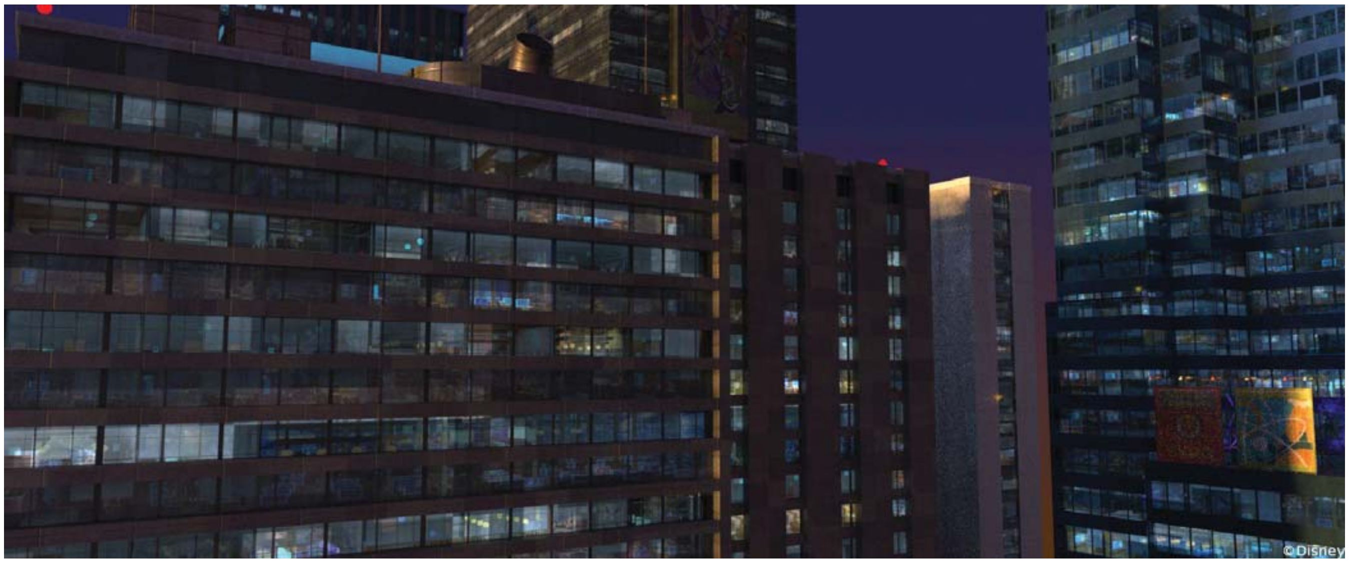


Fig. 4 *Sample Building renders using the Virtual Window Shader.*

8. Conclusions

Since handling a large amount of geometry produces a challenge, removing the need to model and render additional geometry for each room in each building allows for increased visual complexity at lower cost. The artists can use different configurations of emissive textures to create variations. These configurations can also be re-used or varied for multiple buildings to create entire cityscapes, even for large cities full of buildings. This adds character to the buildings and realism of the scene, especially during nighttime shots when room interiors become prominent.

We successfully used the Virtual Window Shader in production for the film "Big Hero 6". It allowed us to create a more complex city than we ever would have been able to represent using actual geometry. Thus, our city scenes are more intricate and realistic than ever, enhancing the look of the movie.

9. References

CATMULL, E. E. 1974. *A Subdivision Algorithm for Computer Display of Curved Surfaces*. Ph.D. Dissertation. The University of Utah. AAI7504786.

BLINN, J. F. 1978. Simulation of wrinkled surfaces. In *Proceedings of the 5th annual conference on Computer graphics and interactive techniques* (SIGGRAPH '78). ACM, New York, NY, USA, 286-292. DOI=10.1145/800248.507101 http://doi.acm.org/10.1145/800248.507101

BURLEY, B., AND LACEWELL, D. 2008. Ptex: Per-Face Texture Mapping for Production Rendering. *In Computer Graphics Forum* 27, 4. pp. 1155-1164.

DREBIN, R. A., CARPENTER, L., AND HANRAHAN, P. 1988. Volume Rendering. *SIGGRAPH Comput. Graph.* 22, 4, 65-74. DOI=10.1145/378456.378484 http://doi.acm.org/10.1145/378456.378484

EISENACHER, C., NICHOLS, G., SELLE, A., AND BURLEY, B. 2013 Sorted deferred shading for production path tracing. In *Proceedings of the Eurographics Symposium on Rendering* (EGSR '13). Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 125-132. DOI=10.1111/cgf.12158 http://dx.doi.org/10.1111/cgf.12158

KANEKO, T., TAKAHEI, T., INAMI, M., KAWAKAMI, N., YANAGIDA, Y., MAEDA, T., AND TACHI, S. 2001. Detailed Shape Representation with Parallax Mapping. *In Proceedings of ICAT*, pp. 205–208.

MULLER, P., WONKA, P., HAEGLER, S., ULMER, A., VAN GOOL, L. 2006. Procedural modeling of buildings. *ACM Trans. Graph.* 25, 3 614-623. DOI=10.1145/1141911.1141931 http://doi.acm.org/10.1145/1141911.1141931

OLIVEIRA, M. M., BISHOP, G., AND MCALLISTER, D. 2000. Relief texture mapping. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (SIGGRAPH '00). ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 359-368. DOI=10.1145/344779.344947 http://dx.doi.org/10.1145/344779.344947

ŠRÁMEK, M. 2006. 20 years of volume rendering. In *Proceedings of the 22nd Spring Conference on Computer Graphics* (SCCG '06). ACM, New York, NY, USA, 7-16. DOI=10.1145/2602161.2602162 http://doi.acm.org/10.1145/2602161.2602162

SEYMOUR, M. 2012. Spider-Man: the detailed vfx of spiders and lizards. *fxguide* http://www.fxguide.com/featured/spider-man-the-detailed-vfx-of-spiders-and-lizards/

SEYMOUR, M. 2005. Double Negative Breaks Down Batman Begins. *fxguide* http://www.fxguide.com/featured/double_negative_breaks_down_batman_begins/

VAN DONGEN, J. 2008. Interior Mapping. In *CGI 2008 Conference Proceedings*

WELSH, T. 2004. Parallax mapping with offset limiting: A per-pixel approximation of uneven surfaces. *Infiscape Corporation*

WONKA, P., WIMMER, M., SILLION, F., AND RIBARSKY, W. 2003. Instant architecture. *ACM Trans. Graph.* 22, 3, 669-677. DOI=10.1145/882262.882324 http://doi.acm.org/10.1145/882262.882324