Efficient Displacement Mapping by Image Warping

Gernot Schaufler
Computer Graphics Group, MIT
Cambridge, MA 02139
gs@graphics.lcs.mit.edu

Markus Priglinger
GUP, JK University
A – 4040 Linz, Austria
mprigl@gup.uni-linz.ac.at

Abstract. While displacement maps can provide a rich set of visual detail on otherwise simple surfaces, they have always been very expensive to render. Rendering has been done using ray-tracing and by introducing a great number of micro-polygons. We present a new image-based approach by showing that rendering displacement maps is sufficiently similar to image warping for parallel displacements and displacements originating form a single point. Our new warping algorithm is particularly well suited for this class of displacement maps. It allows efficient modeling of complicated shapes with few displacement mapped polygons and renders them at interactive rates.

1 Introduction and Motivation

Displacement mapping as introduced by Cook provides rich geometric and visual detail without requiring the user to model them explicitly [6]. In contrast to other texture mapping techniques, not only the appearance of a surface is altered, but the surface itself is displaced by an amount specified in a texture map.

While displacement mapping reduces the burden on the modeling side, it increases the effort necessary to obtain images from the model. Little published work exists on the rendering of displacement mapped surfaces [13] and two approaches are commonly taken: micro-polygons and ray-tracing. Both of these methods are very time consuming and, therefore, more efficient alternatives are desirable.

This paper introduces such an alternative by making the observation that displacement mapping is sufficiently similar to warping an image with depth. The pixel colors describe the optical surface properties and the depth specifies how much the surface deviates from the image plane. These displacements can either be applied perpendicular to the normal of a flat surface using an orthographic projection, or they can be made to emanate from a single point — namely the center of projection in a perspective image. Image warping takes the displaced surface points to the same location in the image as first displacing them in 3D and then projecting them onto the image plane.

Current warping algorithms have difficulties in reconstructing the final image, both in regions of depth discontinuities and where the object's surface was not sufficiently sampled. We introduce a novel warping algorithm that overcomes these problems for the case of images representing displacement maps. In this restricted family of images, depth differences in adjacent pixels are always meant to represent a surface slope, and therefore, must be treated as being connected.

We begin with a discussion of previous work on rendering displacement maps and warping depth images. Next, we describe how the epipolar geometry relating two images can be exploited to implement an efficient warping algorithm well suited for displacement maps. Our interactive system allows the user to cover arbitrary geometry with displacement maps. For rendering, these displacement maps are warped into the final image and composited using a depth buffer, which handles displacement mapped geometry efficiently in traditional scan-line renderers. We then present images and performance measurements in the results section. Finally, we discuss limitations and possible extensions for more general shading.

2 Previous Work

Since the introduction of displacement maps in 1984 by Cook [6], they have been rendered using micro-polygons [7] and using ray-tracing [13, 19, 24]. By subdividing displacement-mapped geometry into micro-polygons and displacing their vertices, a great deal of additional geometry is introduced that makes the model time-consuming to render. A fine subdivision is usually necessary to capture the details of the displacement map. The only system capable of doing this described in the literature is the REYES image rendering architecture [7], but it suffers from additional problems if the displacements excessively increase the size of the triangles. Ray-tracing is a costly rendering technique in itself and ray-tracing inverse displacement maps has been described as only being practical for objects of the complexity of tori or sweeps [13]. The authors of [13] expect the rendering time for more complex surfaces "to become prohibitive." Pharr et al. have presented an approach to handle micro-polygons in ray-tracing [20, 21].

Image-based rendering (IBR), on the other hand, is very efficient for detailed scenes because its algorithmic complexity is independent of the complexity of the rendered scene. The complexity is determined by the number of samples in the reference images and the desired images. IBR evolved from re-projections of environment maps for perspective viewing [3], via image interpolation [4], to image warping [16].

There are two strategies in image warping: forward mapping and backward mapping. Forward mapping loops over the pixels in the reference image and projects each one into the desired image. Splatting can compensate for mismatches in the images' sampling densities. Grossman provides an alternative solution [11]. Forward mapping is usually preferred, because it can be implemented more efficiently. Backward mapping loops over the pixels in the desired image and finds the corresponding samples in the reference image. Since the mapping is not invertible, a search must be performed. The epipolar geometry limits the extent of this search adversely affecting its time complexity [17]. Popescu has accelerated warping by parallelization [22].

Two approaches leading in the direction of our method have been published, namely view-dependent texture mapping [8] and ray-tracing height-fields [1, 5, 10, 15]. Debevec uses stereo algorithms to obtain displacement maps for building facades, and renders them with view-dependent texture mapping by blending between the images taken from different view angles. Height-fields can be rendered efficiently by sweeping a ray upwards in each pixel column of the image keeping track of how the intersection with the terrain recedes into the distance.

3 Forward Image Warping

Assume points in the image space of a pinhole camera (as in [17] and shown in Figure 1) are converted to 3D Euclidean space through multiplication by 3 by 3 matrix P as given in Equation (1):

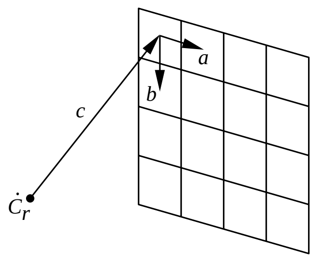

Fig. 1. A pinhole camera with center of projection $\dot{C}_r$.

$$Px = [a\ b\ c] \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{pmatrix} a_x & b_x & c_x \\ a_y & b_y & c_y \\ a_z & b_z & c_z \end{pmatrix} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \tag{1}$$

$$x_d = \delta(x_r) P_d^{-1}(\dot{C}_r - \dot{C}_d) + P_d^{-1} P_r x_r \tag{2}$$

$$\delta(x) = \frac{|Px|}{depth} = \frac{1}{w} \tag{3}$$

The forward warping Equation (2) takes samples $x_r = (u, v)$ described by a color and a disparity $\delta(x_r)$ as its input (see Equation (3)). Equation (2) maps them from a reference image (denoted by subscript r) into a desired image (denoted by subscript d, see Figure 2). The workings of the forward warping equation are shown in Figure 3 (reproduced from [14]). The figure shows a cross-section of Figure 2 known as the epipolar plane — the plane containing the two cameras' centers of projection and the 3D point $\dot{X}$.

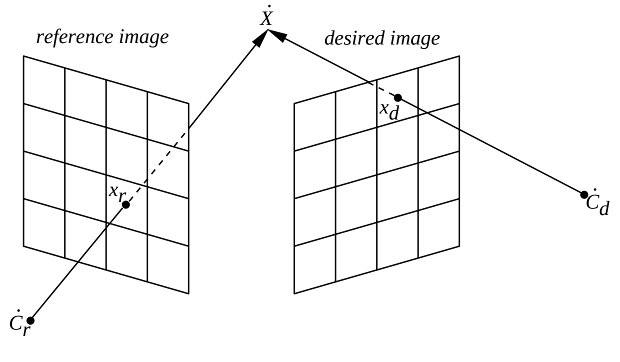

Fig. 2. Point $\dot{X}$ as seen in the reference image and in the desired image.

Subjecting every sample x_r to the warping equation will usually result in holes in the desired image because of different sampling densities in the reference and desired images. This is the main reason why inverse warping is desirable, as it makes sure that every pixel in the desired image is computed.

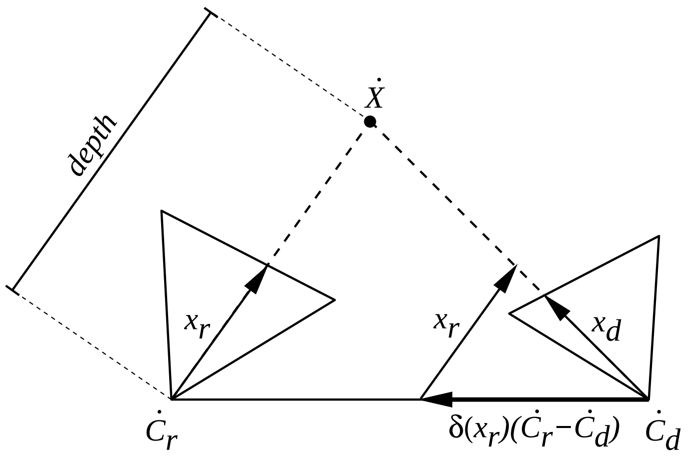

Fig. 3. Illustration of the forward warping equation in the epipolar plane.

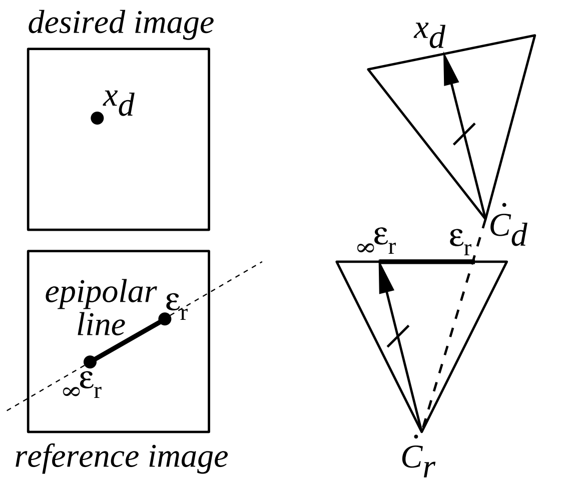

Fig. 4. Inverse warping: the epipolar line's extent in the reference image.

4 Inverse Image Warping

Two points in the reference image play an important role in which samples correspond to a pixel in the desired image — the epipole ϵ_r and the projection of the pixel's point at infinity ${}_\infty\epsilon_r$. They are determined by the viewing ray through a particular pixel in the desired image. Their geometric interpretation is illustrated in Figure 4 and Equation (4).

$$\epsilon_r = P_r^{-1}(\dot{C}_d - \dot{C}_r) \quad {}_\infty\epsilon_r = P_r^{-1} P_d x_d \tag{4}$$

The epipole ϵ_r is the intersection of the baseline connecting the two cameras $\dot{C}_r$ and $\dot{C}_d$ (also the ray's origin) with the reference image plane. The point ${}_\infty\epsilon_r$ is the projection of the ray's point at infinity into the reference image. These two points delimit the segment on the epipolar line, where the point x_r corresponding to the point x_d must lie (see Figure 4). The extent of this line segment must be searched to obtain x_r for every pixel in the desired image [17].

Our approach to inverse image warping (or backward mapping) differs from previous work [8, 14, 17] by making the observation that points x_d on epipolar lines in the desired image share the same corresponding epipolar line in the reference image. Therefore, there is great coherence between the searches for corresponding points when considering adjacent points along the epipolar line in the desired image [23].

This observation is inspired by the ray-casting of height-fields mentioned in the previous work section [1, 5, 10, 15]. In "image-warping terminology," height-fields are orthographic reference images. As long as the viewing direction in the desired image is horizontal, vertical pixel columns lie on epipolar lines in the desired image. The coherence along them can be exploited by sweeping a ray upwards in every pixel column, and the search for the intersection of the next ray's pixel with the terrain can start where the intersection for the last ray was found.

The same is true for warping the points along an epipolar line in an inverse warper. Figure 5 shows how the rays through the points along an epipolar line scan the samples in the reference image for correspondences. Instead of searching the complete segment between x_r and ${}_\infty\epsilon_r$ for every pixel, a match is typically found within a bounded number of steps starting from the match of the last pixel processed. Consequently, the expected runtime complexity of inverse warping is reduced from $O(n^3)$ to $O(n^2)$ for n by n reference and desired images. Instead of warping individual samples we warp epipolar lines as a whole.

5 Implementation

In order to warp a complete perspective image we need to cover the whole image with epipolar lines. For this we locate the epipoles in both images (see Figure 6). Then we trace epipolar lines from the border of the images towards the epipole in the case of a true epipole (the baseline connecting the two cameras pierces the image plane from the back) or from the epipole to the image border in the case of an antipode [9] (the baseline pierces the image plane from the front).

There are nine regions (A-I) in and around the image where the epipole can lie. This results in different numbers of image border-lines (a-d) that must be walked to cover the desired image. Table 1 summarizes these cases and Figure 7 shows the case of the epipole ϵ_d in region I. We use Bresenham's line rasterization algorithm to walk along

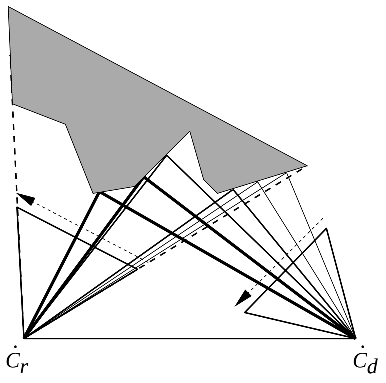

Fig. 5. Rays scanning the displacement map of the reference image along epipolar lines in both the reference and the desired image.

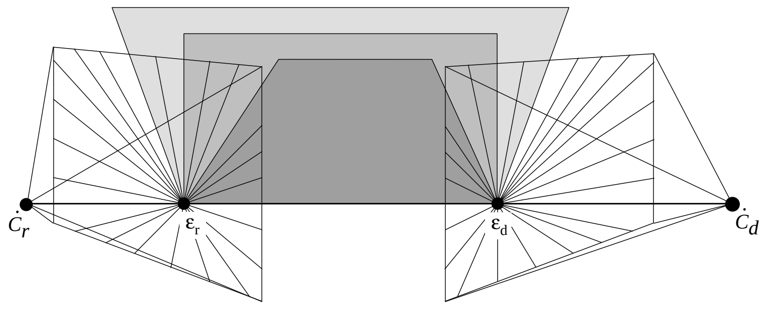

Fig. 6. The family of epipolar planes through the baseline covers both the reference and desired image with epipolar lines.

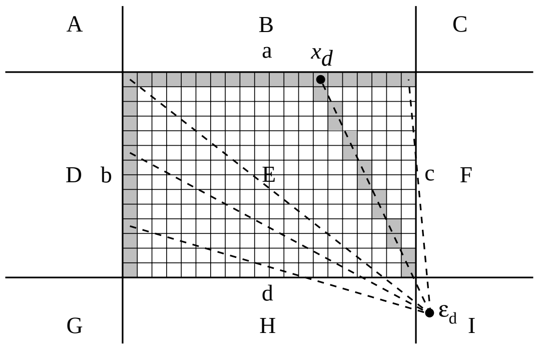

Fig. 7. Covering a raster image with epipolar lines.

the epipolar lines in the raster images [2]. In order to process every pixel in the desired image only once, we mark the processed pixels with a flag.

ϵ_d in region	A	B	C	D	E	F	G	H	I
border walks	2	3	2	3	4	3	2	3	2
borders	c, d	b, d, c	b, d	a, c, d	a, c, d, b	a, b, d	a, c	b, a, c	b, a

Table 1. Different cases depending on which region contains the epipole ϵ_d.

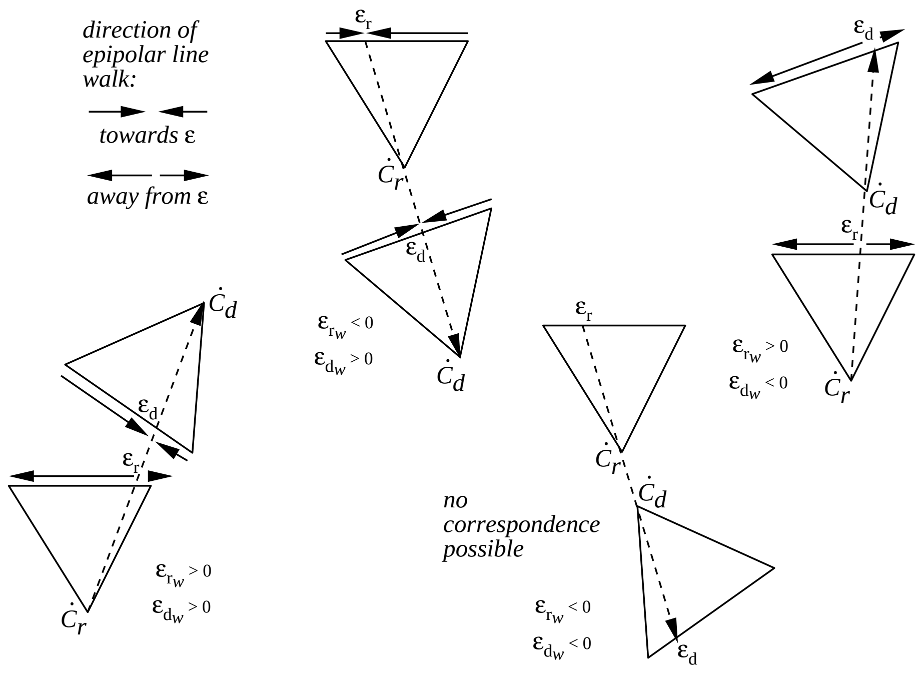

Fig. 8. The four different cases of walking order along epipolar lines.

Figure 8 summarizes the four cases of the walking direction along epipolar lines for both the reference image and the desired image. They are distinguished by the homogeneous coordinate of the images' epipoles (making the point an epipole or an antipode). Figure 9 depicts the five cases when clipping the epipolar line to the reference image results in a non-empty line segment. Notice how in some cases — a), b) and d) — points outside the segment connecting ϵ_r and $_\infty\epsilon_r$ are considered. When warping epipolar lines instead of individual points, the sign of the homogeneous coordinate of $_\infty\epsilon_r$ can change along the line. In warping algorithms, which consider a single point at a time different, the sign of $_\infty\epsilon_r$'s homogeneous coordinates correspond to different walking orders along the epipolar line in the reference image. This behavior is achieved when warping whole epipolar lines by continuing to search for correspondences beyond $_\infty\epsilon_r$. A different sign of the homogeneous coordinate takes $_\infty\epsilon_r$ to the other side of ϵ_r along the epipolar line and the search must proceed in the other direction. However, this is exactly the direction we have started with when the sign was still different.

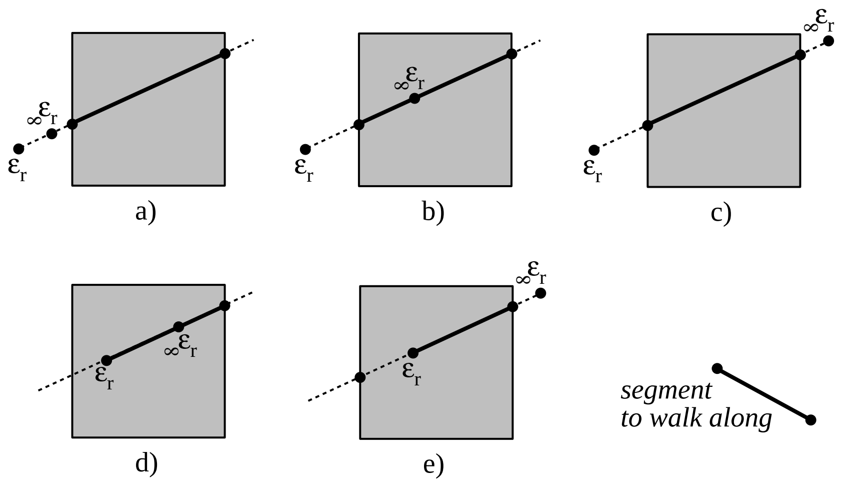

Fig. 9. Clipping the epipolar line to the reference image.

In order to know whether a correspondence of x_d with x_r was found, a minimal disparity $_{min}\delta(x_r)$ is determined from the warping equation. If the disparity $\delta(x_r)$ stored for x_r is larger than $_{min}\delta(x_r)$, then a correspondence exists. Otherwise the search along the epipolar line must continue.

Once a correspondence has been found, we calculate x_{d_w} (the homogeneous coordinate of the point in the desired image) and optionally convert it to depth using Equation (3) so that the depth test can be carried out in the desired image. For a scene entirely made up of displacement maps, disparity values can be compared instead.

6 Displacement Mapping by Image Warping

Displacement mapping by image warping is compatible with two classes of displacement maps: height-fields (orthographic projections) and displacements originating from a single point (perspective projections with the single point being the center of projection). Such displacement maps can be rendered using image warping. This is not as severe a restriction as it sounds because the directions of the displacements are usually irrelevant as long as the surface ends up at the desired 3D location. So even the displacements for a free-form surface could be done in this way — only the displacements may not always be in the direction of the surface normal. The restriction is that every ray in the reference image must not have more than one intersection with the desired surface. Otherwise, the surface will self-occlude and cause undesired rubber-sheets.

Figure 10 shows a simple example of how a displacement-mapped globe is built from a sphere by displacing a cube with six perspective depth images. The center of projection for all the six images is the center of the sphere and all the images have horizontal and vertical fields of view of ninety degrees. From left to right Figure 10 shows the frusta of the six images positioned in the cube, the rays of the reference images originating from the center of the sphere, the color image of the earth's texture, and the displacements.

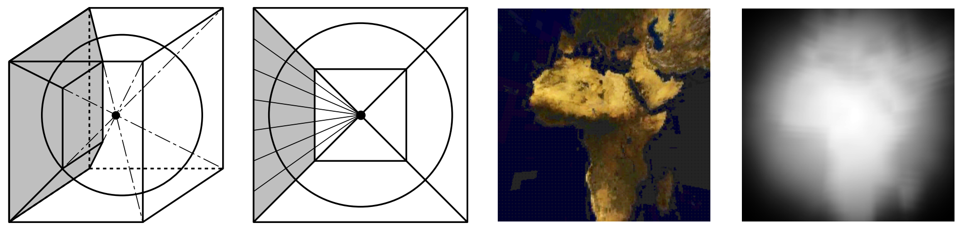

Fig. 10. An example of creating a displacement-mapped globe with six reference images. The frusta of these perspective images are arranged as a cube.

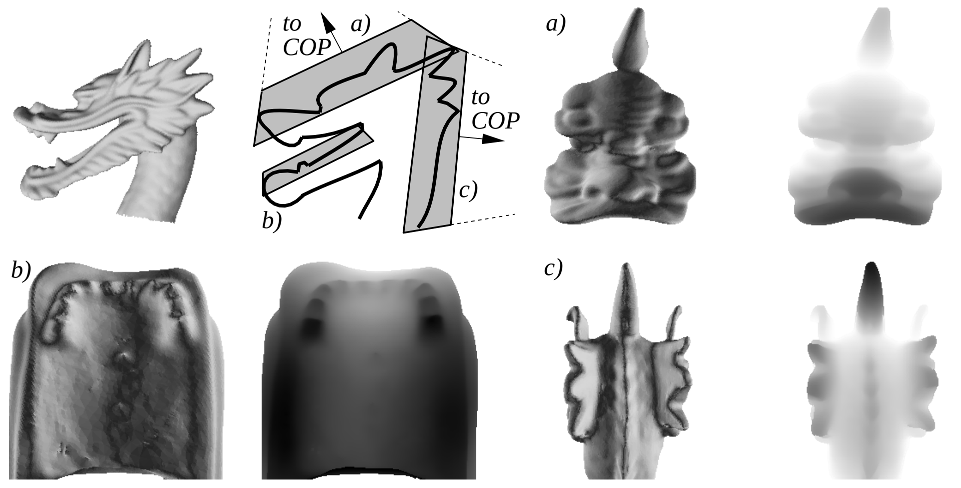

Fig. 11. Covering a dragon head with displacement maps — the arrows point to the centers of projection (COP). Both the texture (left) and the displacements (right) are shown for frusta a) – c).

Figure 11 illustrates how one would plan to cover the complex surface of a dragon head with reference images in order to obtain displacement maps for a hand-full of polygons. The upper left shows the head and a cross-section with three exemplified frusta labeled a) through c). Their corresponding RGB images (left) and depth maps (right) are shown in the rest of the figure. A total of sixteen reference images has been used to cover the surface of the dragon head. Not all points of the displacement-mapped polygon (the image plane) need to be used as samples. The required part is stenciled out by considering only samples in front of the far plane of the view-frustum. The image on the right of Figure 12 shows the polygons to be displaced into the dragon's head (see the color section).

7 Results

We implemented a program for the interactive placement of frusta on the surface of arbitrary geometry and the warper described in sections 4 and 5. The renderer uses the warper to map multiple displacement maps into the final image where they are composited using a depth buffer.

7.1 Interactive Frustum Placement

Figure 12 shows two screenshots of the interactive program used to place frusta on the displacement-mapped globe and on the dragon head (see color section). In this program, the geometry is rendered using OpenGL and the frusta are created and modified using the mouse. In the lower left corner of these images, the view obtained from the current frustum is displayed to guide the user with editing the frustum. Sets of frusta, together with the RGB and the depth images obtained from using them in a perspective projection, can be saved to a file.

The warper distinguishes between pixels pertaining to an object or to the background by examining the associated disparity value: disparity values corresponding to the far clipping plane of the frustum are the background pixels. This would allow us to "paint" far disparity values over unwanted image sections in order to avoid the aforementioned rubber-sheet artifacts. Currently, we do not exploit this possibility.

7.2 Warping Performance

Table 2 shows the performance of warping the reference images of our models taken at resolutions of 128x128, 256x256 and 512x512 into desired images of the same size on a 250 MHz R10000 processor. Note that in image warping for displacement mapping the individual displacement maps cover only a small fraction of the desired image. Therefore, restricting the epipolar line walk to those image sections will avoid a warping cost directly related to the number of displacement maps used.

The table lists the warping performance for the views given in Figure 13 (in the color section). Notice that $O(n^3)$ growth in the rendering time does not occur when increasing both the reference and desired image dimensions by a factor of two. In practice, variations in the warping performance can occur due to the position of the epipoles in the desired image as classified in Table 1. Also, the clipping of epipolar lines and the rejection of epipolar lines for which no correspondence is possible in the reference image account for some variation.

Model	Fig.13	ref. imgs.	resolution	time (sec)
Torus	top	6	128x128	0.0503
			256x256	0.2013
			512x512	0.7752
Globe	n/a	6	128x128	0.0523
			256x256	0.2911
			512x512	0.8362
Dragon	bottom	16	128x128	0.0842
			256x256	0.3218
			512x512	1.2276

Table 2. Warping performance in seconds on a R10000 processor running at 250 MHz.

8 Conclusions and Future Work

In this paper, we described a method to achieve interactive rendering of displacement mapped geometry by inverse image warping. The proposed warping algorithm is particularly suited for rendering displacement maps, because it always treats the samples in the reference images as being connected. Epipolar geometry is exploited to reduce the algorithmic complexity from $O(n^3)$, in previously described inverse warpers, to $O(n^2)$ making the performance comparable to forward warping.

Much work remains to be done in the field of obtaining the displacement maps of interesting objects. Stereo algorithms could be applied to acquire displacement maps for approximate geometry of real-world objects. 3D paint programs [12, 25] could be used by artists to paint displacements directly onto simple geometry to increase their visual richness, much in the way the terrain was created in the game *Myst* [18].

In terms of shading, simply warping color values from the reference images into the desired image restricts the lighting to be entirely static. We would like to apply techniques such as deferred shading and shadow maps in order to obtain a more dynamic lighting. Further, the image quality could be improved by applying interpolation schemes on the reference samples instead of picking the closest one.

We would also like to investigate whether avoiding a much denser sampling of the images around the epipole could significantly speed up the warping. Squares centered around the epipole could be used to define additional border lines from which epipolar lines are traced only up to the next inner square allowing to trade denser sampling for more frequent epipolar line setup.

Acknowledgments

We would like to thank the Stanford University Computer Graphics Group for making the dragon appearing in this paper available in their Stanford 3D Scanning Repository.

References

1. Barr, A., *"Ray-Tracing Deformed Surfaces"*, SIGGRAPH '86, pp 287-296.
2. Bresenham, J. E., *"Algorithm for Computer Control of a Digital Plotter"*, IBM Systems Journal 4(1), July 1965, pp 25-30.
3. Chen, S. E., L. Williams, *"View Interpolation for Image Synthesis"*, SIGGRAPH '93, pp 279-288.

4. Chen, S. E., *"Quicktime VR - An Image-Based Approach to Virtual Environment Navigation"*, SIGGRAPH '95, pp29-38.
5. Cohen-Or, D., E. Rich, U. Lerner, V. Shenkar, *"A Real-Time Photo-Realistic Visual Fly-through"*, IEEE Transactions on Visualization and Computer Graphics, Vol. 2, No. 3, September 1996, pp 255-265.
6. Cook, R. L., *"Shade Trees"*, SIGGRAPH '84, pp 223-231.
7. Cook, R. L., L. Carpenter, E. Catmull, *"The Reyes Image Rendering Architecture"*, SIGGRAPH '87, pp 95-102.
8. Debevec, P. E., C. J. Taylor, J. Malik, *"Modeling and Rendering Architecture from Photographs"*, SIGGRAPH '96, pp 11-20.
9. Faugeras, O., *"Three-Dimensional Computer Vision: A Geometric Viewpoint"*, MIT Press, Cambridge, Massachusetts, 1993.
10. Fishman, B., B. Schachter, *"Computer Display of Height Fields"*, Computers and Graphics, Vol 5 Number 2-4 1980, pp 53-60.
11. Grossman J.P., W. J. Dally, *"Point Sample Rendering"*, Proceedings of the 9^{th} Eurographics Workshop on Rendering, June 29 - July 1, 1998, Vienna, Austria, pp 181-192.
12. Hanrahan, P., P. Haeberli, *"Direct WYSIWYG Painting and Texturing on 3D Shapes"*, SIGGRAPH '90, pp 215-223.
13. Logie J. R., J. W. Patterson, *"Inverse Displacement Mapping in the General Case"*, Computer Graphics Forum, 14 5, December 1995, pp 261-273.
14. Marcato, R.W., *"Optimizing an Inverse Warper"*, Master's Thesis, Massachusetts Institute of Technology, May 1998, pp 35-37.
15. Max, N., *"Vectorized Procedural Models for Natural Terrain: Waves and Islands in the Sunset"*, SIGGRAPH '81, pp 317-324.
16. McMillan, L., G. Bishop, *"Plenoptic Modeling: An Image-Based Rendering System"*, SIGGRAPH '95, pp 39-46.
17. McMillan, L., *"An Image-Based Approach to Three-Dimensional Computer Graphics"*, Ph.D. Dissertation, University of North Carolina, April 1997.
18. Miller, R., Miller, R., *"The Making of Myst"* , Video accompanying the adventure game *"Myst"* by Cyan, Inc.
19. Patterson J. W., S. G. Hoggar, J. R. Logie, *"Inverse Displacement Mapping"*, Computer Graphics Forum, 10 2, June 1991, pp 129-139.
20. Pharr, M., P. Hanrahan, *"Geometry Caching for Ray-Tracing Displacement Maps"*, Proceedings of the 7^{th} Eurographics Workshop on Rendering 1996, pp 31-40.
21. Pharr, M., C. Kolb, R. Gershberg, P. Hanrahan, *"Rendering Complex Scenes with Memory-Coherent Ray-Tracing"*, SIGGRAPH '97, pp 101-108.
22. Popescu, V., A. Lastra, D. Aliaga, M. de Oliveira, *"Efficient Warping for Architectural Walkthroughs using Layered Depth Images"*, IEEE Visualization '98, pp 211-215.
23. Priglinger, M., *"Re-projecting Images with Depth"*, Master's Thesis, Johannes Kepler University Linz, September 1998 (in German).
24. Taillefer, F., *"Fast Inverse Displacement Mapping and Shading in Shadow"*, Graphics Interface '92 Workshop on Local Illumination, Vancouver, May 1992, pp 53-60.
25. Williams, L., *"3D Paint"*, Proceedings of the Symposium on Interactive 3D Graphics 1990, pp 225-234.

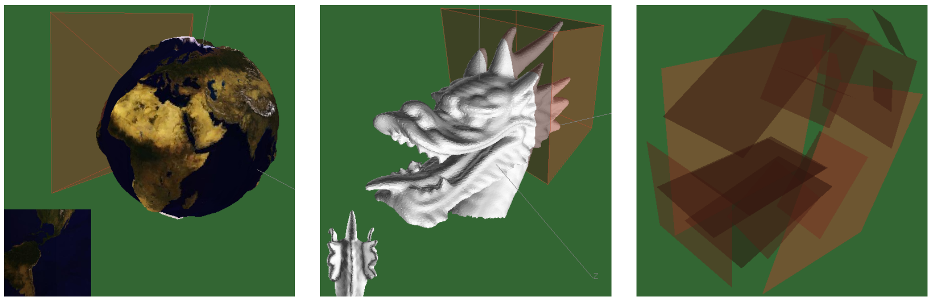

Fig. 12. The system for interactive frustum placement: The main window shows the object together with the frusta. One is highlighted for editing. In the lower left corner the view obtained using this frustum is given. Left: displacement mapped globe. Middle: Scanned dragon head from the Stanford 3D Scanning Repository.
Right: Polygons to be displaced into the dragon head (the image planes of the frusta).

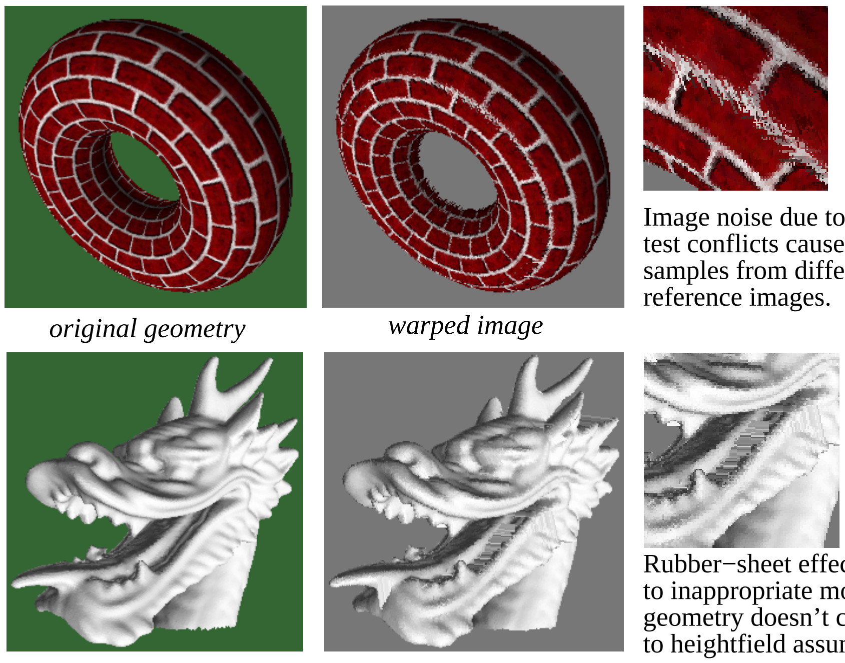

Image noise due to depth test conflicts caused by samples from different reference images.

Rubber−sheet effects due to inappropriate modeling: geometry doesn't conform to heightfield assumption.

Fig. 13. Comparison of images obtained from original geometry and from warped reference images.