Image-Based Objects

TR98-028
July 1998

Manuel M. Oliveira and Gary Bishop

Department of Computer Science
University of North Carolina at Chapel Hill
Chapel Hill, NC 27599-3175

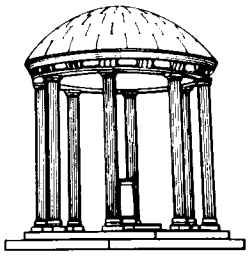

Image-Based Objects

Manuel M. Oliveira and Gary Bishop
UNC Computer Science Technical Report TR98-028

Abstract

We present a compact, image-based representation for three-dimensional objects with complex shapes that can be rendered with correct perspective from arbitrary viewpoints using a list priority algorithm. Objects are represented by six layered depth images sharing a single center of projection. They can be freely translated, rotated, and scaled, being used as primitives to construct complex scenes. We also present a new list priority algorithm for rendering such scenes, and a back face culling strategy for a class of image-based objects.

We demonstrate these concepts by constructing image-based representations from both synthetic and real objects, and rendering them at interactive rates on a PC. Due to their minimum storage requirements and rendering simplicity, image-based objects can find potential uses in games, virtual museums applications, and web catalogs.

1. Introduction

Image-based representations of objects are currently used in computer games and virtual museum applications, and there is some potential demand for web-based shop catalogs. An ideal object representation for such applications should preserve the original appearance of the objects, be able to be manipulated interactively, and visualized from arbitrary viewpoints. Also it should be compact enough to be sent through a network, and rendered at reasonable frame rates using non-specialized graphics hardware.

This paper presents a new compact image-based representation for three-dimensional objects with complex shapes that can be rendered with correct perspective from arbitrary viewpoints using a list priority algorithm based on McMillan and Bishop's [9] occlusion compatible order. In our approach, each object, called an Image-Based Object (IBO), is represented by six layered depth images (LDIs) [15] that share a single center of projection (COP). IBOs can be scaled, and arbitrarily translated and rotated, and can be regarded as primitives to construct more complex scenes. We also present a new list priority algorithm for rendering dynamic scenes composed of IBOs, given some spatial constraints among the objects. We demonstrate these concepts by constructing image-based representations from both synthetic and real objects, and implementing a system prototype that can render IBOs at interactive rates on a PC.

The remainder of this paper is organized in the following way: section 2 surveys previous work in image-based object representation. Section 3 describes the process of building and rendering IBOs. Section 4 discusses how scenes can be constructed from IBOs and presents a list priority algorithm for rendering such scenes. Section 5 discusses some results, while section 6 presents some conclusions and directions for future work.

CB #3175 Sitterson Hall, Chapel Hill, NC, 27599-3175
oliveira@cs.unc.edu, gb@cs.unc.edu

2. Previous and Related Work

Dally et al [3] use a sampling sphere around a target object to take multiple views of it. Such images are stored in a data structure called a delta-tree that divides the (θ, ϕ) space into square regions. During rendering time, the four corners of the region enclosing the desired viewpoint are used for reconstruction [3]. Since the COPs of all reference images are constrained to be on the sphere, it is not possible to completely sample the surface of objects with arbitrary shapes. The multiple views have different centers of projection and a z-buffer is required to eliminate hidden surfaces.

Levoy and Hanrahan [8] and Gortler et al [5] represent objects as collections of images obtained from rectangular grids placed around the objects. At rendering time, the images in the database are re-sampled to produce an interpolated view of the object. This approach requires a large number of images, and putting multiple such representations in the same scene is not straightforward.

Pulli et al [13] use color images and dense range maps to reconstruct sparse triangle meshes associated with real objects. The color images are used as texture maps and applied to the meshes. In order to reconstruct a new view of an object, the meshes corresponding to the three closest original viewpoints are blended on a per pixel basis and z-buffered in software. The use of sparse triangle meshes introduces artifacts at the object silhouettes that look polygonal.

Multiple-center-of-projection images [14] can represent objects as sequences of one-dimensional images acquired along a continuous path. Such a representation provides connectivity information among adjacent samples, and allows different parts of the object/scene to be sampled at different resolutions. Since samples are acquired from different COPs, visibility is determined using a z-buffer.

3. Rendering Objects using a List Priority Algorithm

McMillan and Bishop [9] presented a list priority solution for the visibility problem in the context of their image warping framework. Unfortunately, their algorithm cannot be used to warp multiple images acquired from different COPs simultaneously. An important observation, however, is that their algorithm can still be used if all images share a common COP. In this case, one only needs to specify the order in which

the images must be warped, which changes with the desired viewpoint. Unfortunately, it is not possible to sample the whole exterior surface of a three-dimensional object from a single COP.

3.1. Building Image-Based Objects

One solution to this problem is to put the shared COP inside the object. Although this is not directly realizable for real objects, this idea gives us a good framework to think about the problem. A similar result can be obtained by acquiring multiple views of the object, and resampling them from a single COP, as illustrated in figure 1. The depth values associated with the pixels are used to project all samples back to 3D. Once these samples have been registered, they are reprojected into perpendicular image planes sharing the same COP (figure 1b). Notice that, although a cubic arrangement is shown in figure 1, any parallelepiped would work as well. Also, the object does not need to be completely inside the parallelepiped. The described arrangement is topologically equivalent to a spherical image and, therefore, can be warped in visibility preserving order. During the resampling process, multiple non-redundant samples falling along the same ray are preserved. Thus, each image-based object is represented by six LDIs, which are stored as linear arrays for efficient warping.

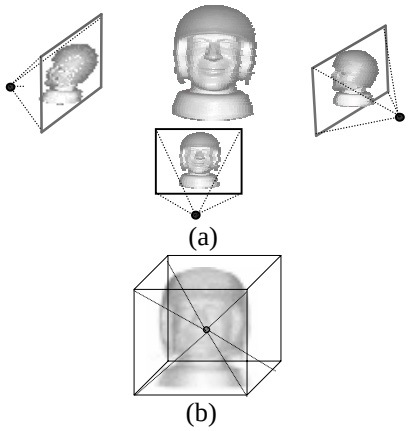

Figure 1. (a) Views of an object acquired from multiple COPs. (b) The samples are registered and will be reprojected onto perpendicular image planes (represented by the faces of the cube) sharing a single COP.

Notice that, although our representation makes use of LDIs, the two concepts are quite different. LDIs were introduced to minimize disocclusion problems that occur when warping depth images [15]. This is achieved by allowing a view of the scene to contain multiple samples along each ray. While LDIs can be warped in occlusion compatible ordering, they are only effective if the desired view is in a certain neighborhood of the LDI's COP. An apparent similarity between both concepts is the use of a cube when generating LDIs from ray traced scenes. In this case, the purpose of such a cube is to define the region of interest in which the viewer will be allowed to move when exploring the scene [15]. In our approach, the goal is to produce arbitrary views of a three-dimensional object from a fixed set of six images warped in occlusion compatible order. The parallelepiped (shown in figure 1) is used for two reasons: first, to provide a surface topologically equivalent to a sphere, for which an occlusion compatible ordering is known to exist [10]. Secondly, it can be decomposed into parameterized planar regions (faces of the parallelepiped), for which a warper can be implemented efficiently.

Given the resolutions (possibly different) of the planar images associated with the faces of the parallelepiped, each original sample is mapped to the closest pixel of the image covering the corresponding region of space. The higher the resolution the smaller the reprojection error. Alternatively, the surfaces of the objects can be reconstructed and resampled using a regular grid at the corresponding image planes. In all examples shown in this paper, samples were mapped to the center of the closest pixel they project to.

A sample is considered to be redundant if it is closer than a pre-defined threshold in 3D (Euclidean distance) to another sample from a different original image, and both have similar colors. Redundant samples are eliminated during the construction of the LDIs. The preprocessing time associated with the construction of the six LDIs corresponding to the IBO shown in figure 13 was about 5 seconds on a HP workstation, after which the LDIs are saved in disk and are ready for future use. In our current implementation, the choice of which samples are preserved (among the redundant ones) is arbitrary. A better solution seems to base such a decision on the angle between the ray from the IBO COP to the sample, and the sample normal. Such a normal can be approximated using the neighborhood of the sample in the corresponding original image [12], as part of this preprocessing.

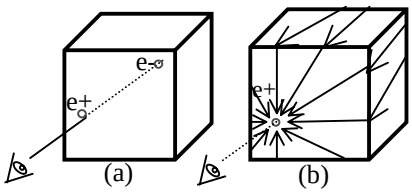

Figure 2. Epipolar Geometry of an IBO. (a) The segment containing the desired COP and the IBO COP (center of the cube) intersects opposite faces at e+ (positive epipole), and e- (negative epipole). (b) Projected flow lines (arrows) defining an occlusion compatible ordering to the whole object.

3.2. List Priority Rendering

Since the IBO representation is topologically equivalent to a sphere, each IBO can be warped using an adaptation of McMillan and Bishop's algorithm for spherical images. The IBO COP is defined as the intersection of the two diagonals of the parallelepiped. Given such a configuration, the following properties can be observed:

- Although the six planar images share a common COP, they are still independent from one another. Therefore, the definition of sheets is independently established for each image, using the regular sheet split procedure [10];

- Since the IBO COP is at the intersection of the diagonals, the positive and the negative epipoles fall within opposite faces;
- There is no redundancy among images, i.e., no sample is seen in more than one face;
- The whole field of view is covered;

Figure 2 illustrates the epipolar geometry for a cubic IBO. The line connecting the desired and the IBO COPs intersects the cube at opposite faces[1], and defines an occlusion compatible order for warping the whole object. The face containing the negative epipole (e-) (figure 2) must be warped first, while the face containing the positive epipole (e+) must be warped last. The arrows in figure 2b are the projected flow lines representing the occlusion compatible order. Consider the cube split into six pyramids with apices at the IBO COP, as shown in the figure 3. Let's call the faces containing the positive and negative epipoles, F, and K, respectively. Notice that this classification is relative to desired view position. The other two pairs of opposite faces are called (A, A'), and (B, B') (figure 3). The following theorem defines orders in which the six faces can be independently warped in visibility preserving order. A proof of the theorem is presented in Appendix A.

Theorem: Let B' be the base of pyramid $P_{B'}$ that is intersected by the segment connecting the positive epipole and its parallel projection into K. Then, warping the faces of the cube in the order (K, B, A, A', B', F), or (K, B, A', A, B', F) produce correct visibility from the desired view position.

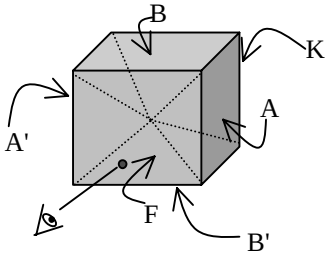

Figure 3. Faces of the parallelepiped are labeled with respect to the desired viewpoint. F contains the positive epipole (e+), while K contains the negative epipole (e-).

Since the set of rays emanating from the object COP covers a solid angle of 4π steradians, some care should be taken in order to guarantee that multiple samples along a ray are always warped from back to front with respect to the desired view. One way to define such an order is to compute the smallest angle between the two vectors from both COPs to the furthest sample along the ray in question. If the angle is less than 90 degrees, the samples are warped from farthest to closest (with respect to the IBO COP); otherwise, they are warped from closest to farthest. Notice, however, that such a procedure requires the knowledge of the ray direction associated with the projection of the sample in the desired image plane, which is only known after the actual warping. In order to avoid an extra warping step just to compute such a direction, we approximate the desired ray using the desired image plane normal. This way, the order in which samples are warped is established by a dot product. Although this is only an approximation, it works very well in practice.

In the case of objects whose representations are topologically equivalent to spheres (genus zero) and present good aspect ratio, the warping of the image containing the negative epipole can be omitted (figure 15). This optimization is analogous to back face culling used in polygonal computer graphics. In practice, we observed speedups varying from 19% to 22% due to the its utilization.

3.3. Transformations

Geometric transformations such as translation, rotation, and scaling can be easily applied to IBOs. Since all six LDIs are defined with respect to a single COP, translations are obtained simply by translating the object COP. Rotations are obtained by rotating the three vectors that define the pinhole camera parameters [11] of each LDI around the desired axis. The generalized disparity value associated with each sample is computed as d/z, where d is the distance from the COP to the image plane, and z is the z coordinate of the sample with respect to a frame of reference whose XY-plane contains the image plane. Therefore, to scale an object by a factor s is equivalent to multipling the disparity of all its samples by $1/s$.

4. Scenes from IBOs

Image-based objects can be combined to generate more complex scenes. Given some constraints on the spatial relationship among the objects, the whole scene can rendered using an occlusion compatible order. Thus, if there are no inter-penetrations between any pair of IBOs bounding spheres, then for any desired view there is at least one serial order in which the objects can be warped that produces correct visibility and does not require depth comparison.

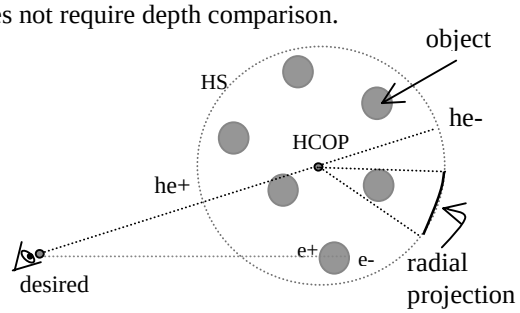

Figure 4. Epipolar geometry of a scene with respect to the desired view. Small circles represent IBOs bounding spheres. HS (hypothetical sphere), HCOP (hypothetical COP). Non-overlapping radial projections.

The algorithm presented here in 2D, for simplicity, provides a way to obtain one such order. Given a set of objects, compute a hypothetical COP for the scene (HCOP) as the average of all objects COPs. If the derived HCOP does fall inside any object's bounding sphere, move it to avoid this

[1] The cases in which the line intersects edges or vertices are treated similarly.

situation. Define HS, a hypothetical sphere (circle in 2D) whose center is at HCOP. Given an arbitrary desired view, compute he- and he+, the hypothetical negative and positive epipoles on HS induced by the desired view. Next, radially project all objects into HS (figure 4). At this point, there are two possible configurations: (a) none of the radial projections overlap, and (b) at least two projections overlap.

The case in which no projections overlap is shown in figure 4. We radially scan one of the hemispheres from he- towards he+ and every time we reach an object it is added to the end of a list (originally empty). If the segment HCOP/he+ crosses an object, that object should appear last in the final list. Then the other hemisphere can be processed the same way. The order of the objects in the list is an order that produces correct visibility from the desired COP. Each individual object is warped using the order defined by the theorem presented in section 3.2.

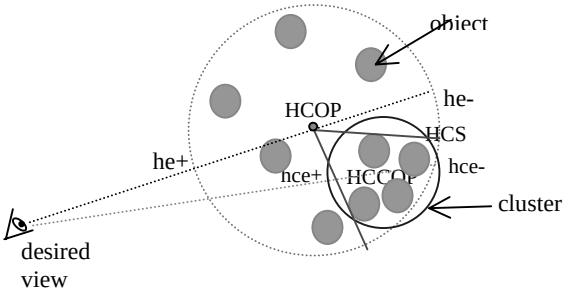

Figure 5. Recursive application of the algorithm to solve visibility inside a cluster of overlapping projections.

If at least two projections overlap (figure 5), group the overlapping objects in clusters. The criterion for defining clusters is transitive, i.e., if the projection of object A overlaps the projection of object B, and the projection of object B overlaps the projection of object C, then A, B, and C belong to the same cluster.

We start scanning one of the HS hemispheres from he- towards he+ and every time we reach an object that does not belong to a cluster it is added to the end of a list (originally empty). If the object belongs to a cluster, the algorithm is recursively applied to the cluster itself, i.e., an HCCOP (hypothetical cluster COP - see figure 5), an HCS (hypothetical cluster sphere), and a pair of epipoles (hce- and hce+) are defined for the cluster, and the projections of the objects into HCS are computed, with sub-clusters possibly defined. Then, one of the hemispheres of HCS is scanned from hce- towards hce+. If an object of the cluster is found that does not belong to a sub-cluster it is added to the end of the list, and removed from the cluster. If, however, the object belongs to a sub-cluster, this process is recursively reevaluated. As soon as the objects belonging to the first hemisphere have been completely examined, the next hemisphere can be processed in a similar way. At the end, the constructed list defines an order that produces correct visibility from the desired COP.

A cluster covers a whole angular range in HS. The order in which its elements should be rendered is completely specified by them and the desired COP. Therefore, when the angular scanning reaches a cluster, only its elements need to be considered, in the context of the desired COP.

4.1. Additional Remarks

When applying the algorithm to warp a series of objects, it is strictly correct to warp only the parts of the objects that are allowed by the (h)e- (h)e+ configurations, i.e., the parts that fall in the "working" hemisphere. However:

1. If an object is crossed by the segment from the HCOP to e- and the radial projection of the object into HS does not overlap any other projection (figure 6a), the object can be warped completely, although its parts fall in the both hemispheres.
2. If the projections involving at least one object crossed by segment from HCOP to e- overlap (figure 6b), keep applying the algorithm to this cluster recursively, and this situation will be reduced to case 1.
3. If at least one object is crossed by the segment from HCOP to e+ (figure 6c), wait until the other hemisphere also reaches this cluster before deciding its order and warping its components.

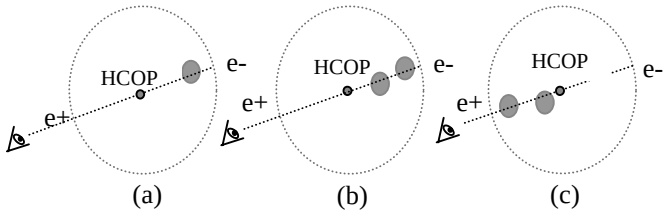

Figure 6. (a) Object crossed by the segment connecting HCOP and the negative epipole. (b) Two objects with overlapping projections crossed by the segment HCOP-e-. (c) Two overlapping objects crossed by the segment HCOP-e+.

4.2. Further Considerations

The notion of planes dividing the space into two half spaces, one of which contains the desired COP, is a general concept from which the algorithm described in section 4 is a special case. Such a notion has been explored by Schumacher's list priority algorithm [16], as well as by BSP trees [4]. The main difference between these two approaches and ours is the fact that they make use of a preprocessing stage and are constrained to static scenes (dynamic BSP trees [17] provide some extra flexibility but do not completely eliminate the need to update the tree structure). In the presented algorithm, the rendering primitives are IBOs. This implies a much coarser granularity and, therefore, a smaller number of comparisons to decide the final order. On the other hand, the comparisons are computationally more expensive than the ones used when traversing a BSP tree. One advantage of our approach is the ability to handle dynamic scenes, since no preprocessing is involved. Its main disadvantage is it does not support object interpenetration. In the BSP tree case, polygon interpenetrations are eliminated during the pre-processing phase.

Our current implementation of the algorithm is obtained by computing the projections of the objects' bounding spheres onto a plane perpendicular to the view plane, and using the procedure described. The use of bounding spheres allows for a 2D implementation to be applied to 3D scenes. For the cases in which two projections overlap on the plane, the ambiguity is solved by computing the projections of the conflicting objects onto planes perpendicular to the original one. If the conflict persists, for each pair of conflicting objects we compute the plane orthogonal to the vector connecting the centers of the two objects and tangent to one of the bounding spheres. The coefficients *a*, *b*, and *c* of the plane are the coefficients of the orthogonal vector. *d* is obtained by plugging in the coordinates of the vector scaled by the radius of the first sphere. The sign of the desired view position with respect to the computed plane is then used to order the conflicting objects.

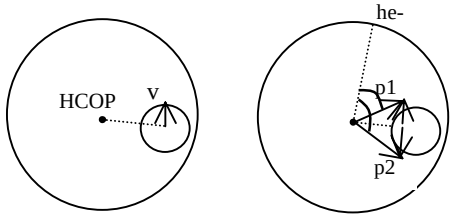

Figure 7. Using angular range to check for radial projection overlapping. (a) v is the vector orthogonal to the segment connecting the center of the two circles. (b) The vectors p_1 and p_2 are used to compute a conservative angular range for the object bounding sphere. Angular ranges are used to check for overlapping projections.

The check for radial projection overlapping is implemented conservatively. For each object we compute a vector *v*, orthogonal to the vector from HCOP to the center of the object's bounding sphere. *v*'s length equals the radius *r* of the object's bounding sphere (figure 7a). Then, we compute vectors p_1 and p_2 (with tails at HCOP), by translating *v* and –*v* by *r*/2 towards HCOP (figure 7b). Finally, compute the angles between p_1, and p_2 and the vector from HCOP to he- (figure 7b). The angular range comprising each object is used to check for overlappings.

4.3. An Approximation Algorithm

Given the restriction that spherical bounding boxes of objects do not interpenetrate, an approximation algorithm can be used to produce correct visibility in almost all cases. A priority list is constructed simply by sorting the objects according to the decreasing distance from the desired COP to the center of each bounding sphere. Despite its simplicity, this heuristic works very well in practice. Figure 8 illustrates this point for two views of the same scene produced with an interactive tool used to verify the heuristic. The numbers associated with the circles are distances from the desired COP to their corresponding centers. The heuristic breaks for configurations involving spheres at highly different scales and tangent to each other. Figure 9a depicts such a configuration. Notice that this is a conservative heuristic, and the rendering of the actual objects in the specified order can be correct even for such configurations (figure 9b).

Figure 8. The use of an approximation algorithm to define a priority list. The numbers represent the distances from the desired COP to the center of the spheres. Objects are sorted by decreasing distance.

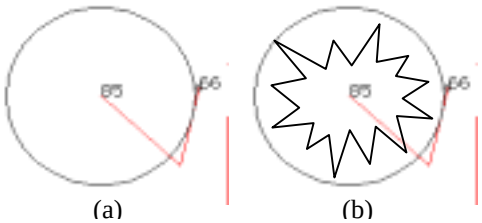

Figure 9. (a) The configuration that breaks the heuristic: two spheres at extremely different scales and very close to each other. Although closest, the smallest sphere is hidden by the biggest one. (b) This order can still produce correct results depending on the actual geometry of the objects.

4.4. Adding Geometric Objects to a Scene

If there are no interpenetrations involving IBOs or geometric models bounding speheres, then for any desired view there is at least one serial order obtained by interleaving the rendering of IBOs and geometric models that produces correct visibility and does not require depth comparison to be established. Such a situation is similar to the one involving only image-based objects.

Given such an order, both IBOs and geometric models can be safely rendered to the same buffer. Geometric models should be rendered using a z-buffer to solve visibility among the polygons that constitute each model (i.e., the z-buffer can be reset after the rendering of each geometric model).

5. Results

We built a system prototype in C++ that implements the algorithms described for construction and rendering of IBOs. In our system, IBOs can be built using 4 different approaches: images with depth obtained from 3D Studio MAX, images with depth obtained from the OpenGL depth buffer [18], images acquired with a laser range finder, and a modified ray tracer [7] that keeps all intersections along a ray. In all examples shown (figures 10, 11, 13, and 14), visibility was solved using our occlusion compatible order algorithm, and no anti-aliasing technique has been used.

Figure 10. Views of an IBO constructed from 6 images and rendered using 2x2 splats.

Figure 11. Views of an IBO constructed from 4 images of a highly specular Venus statue. The left image rendered with points, while right one was rendered using 3x3 splats.

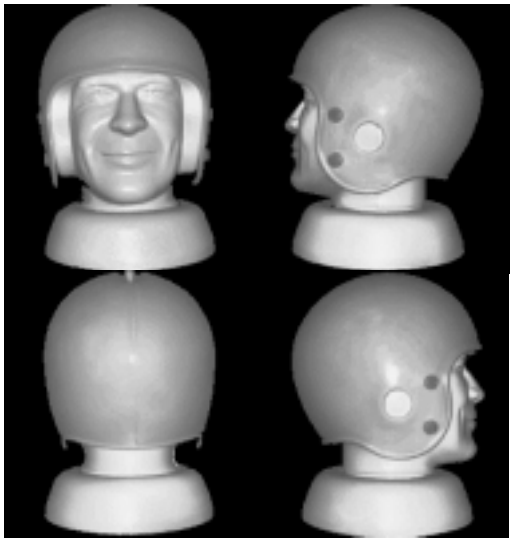

Figure 12. Reflected intensities of a real object obtained with a laser range finder. From top to bottom, left to right: 0, 90, 180, and 270 degrees, respectively. Background color represents zero intensity.

The old clock shown in figure 10 was generated from 6 synthetic images rendered with 3D Studio MAX. Notice that, in this case, the registration process is extremely easy because we have exact camera calibration. The generalized disparitity values were obtained using a plug-in. Its final representation is composed of 6 150x150 LDIs, with a total of 230,102 samples. This is equivalent to a regular depth image with 480x480 pixels. The Venus statue in figure 11 was generated from 4 images rendered with 3D Studio MAX, resulting in 6 150x150

.

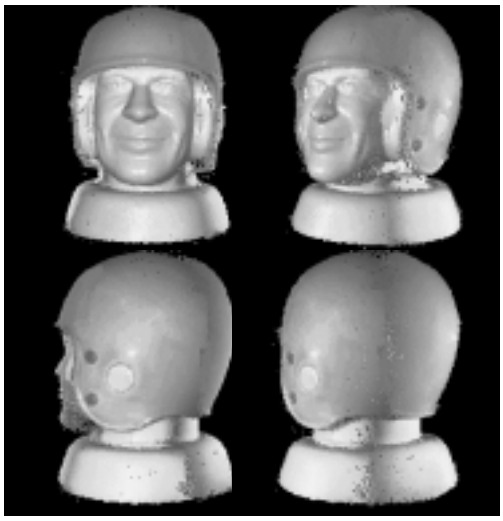

Figure 13. Views of an IBO constructed from the four range images shown in figure 12. The visibility problem is correctly solved, but some artifacts due to differences in shading in the original images, and to areas not sampled by the scanner can be noticed.

Figure 14. Scene rendered using the approximation algorithm described: Venus with an old clock (2x2 splats).

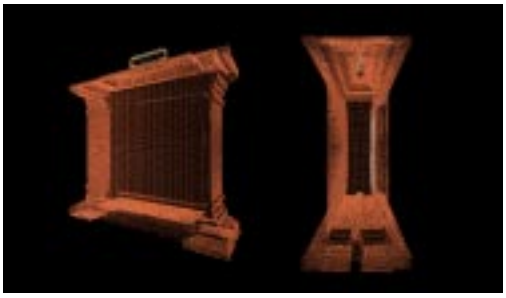

Figure 15. Close-ups of the old clock rendered as point clouds to illustrate back face culling (front and opposite side culled).

LDIs with a total of 220,324 samples. This is equivalent to a regular depth image with 470x470 pixels. Notice how this extremely complex shape is faithfully reconstructed from a relatively small number of samples.

An Acuity Research AccuRange4000 time-of-flight laser range finder was used to create IBOs from real objects. It outputs intensity reflected gray scale images, and range maps. The samples are acquired on a spherical grid (θ,ϕ) and need to be converted to Euclidean coordinates.

Figure 12 shows four views of an object acquired using a rotational platform. Each image subtends 30 degrees in both vertical and horizontal field-of-view, and is 240x240 pixels in size. Specularity is a problem common to all laser range scanners [2] and, in order to reduce its effect, the specular helmet was sanded. However, specular highlights are still visible in the lateral views (figure 12). Another major sources of error during range acquisition are the discontinuities involving the boundaries of the object and the background. In those regions, laser range finders usually receive two returns and average them, producing wrong measurements. In order to avoid this problem, a planar specular reflector oriented approximately 30 degrees with respect to the vertical was used as background. This way, as some portion of the laser beam missed the object, it was reflected away from the sensor. The background color in figure 12 corresponds to regions where no light (zero intensity) returned to the sensor, and illustrates the effectiveness of our solution. However, wrong measurements are still caused by discontinuities along the surface of the object (for instance, see discontinuities between the face and the helmet), as well as inaccuracies of the device. In all such cases, the error appears as noisy data (outliers).

Since the reference images were acquired from virtually different COPs (the scanner COP was kept still and the platform was manually rotated), the images needed to be registered. The approach used for registration is very simple: a white pin was scanned at the center of the rotational platform. From the range data, the (x,z) coordinates of the pin with respect to the scanner were recovered (our system can provide the 3D coordinates associated to any pixel of a range image). The samples were then rotated around the vertical axis passing through (x,z). This simple procedure led to good initial positioning that required little user intervention to achieve satisfactory (although not perfect) registration.

The shading in the intensity images tends to cause seams in the reconstructed model. The surfaces facing the scanner appear brighter than the ones seen by the laser beam at grazing angles. When samples from all reference images are put together, seams become very noticeable. Such distracting effects were reduced (but again not completely eliminated) with the use of a 3D painting tool that is part of our system.

Figure 13 shows some views of the IBO reconstructed from the range images presented in figure 12, and rendered using 2x2 splats. A total of 112,865 valid samples were stored in 4 150x150 and 2 100x100 LDIs, and are equivalent to a regular depth image with 336x336 pixels. Despite its small size, all major features of the original object are preserved.

The original images contain no information about the top, nor the bottom of the object. However, if views from these areas are not going to be explored, it is possible to take advantage of back face culling by not warping the face that contains the negative epipole, as discussed in section 3.2.

Figure 14 shows a scene demonstrating the potential of using IBOs to construct complex scenes. The visibility between the two objects was solved using the approximation list priority algorithm described. Notice that the two objects do not interpenetrate (although their bounding spheres do), showing how conservative the algorithm is. For applications that can constrain objects' spatial relationships, this algorithm can be an attractive alternative. The accompanying video tape illustrates the use of the algorithm in a dynamic scene.

We measured the frame rates associated with rendering of IBOs on a Pentium II PC running at 400MHz. Resampling was performed using 2x2 splats. During the measurements, all objects were completely inside the user's field of view. Table 1 summarizes the results for the IBOs shown in figures 10, 11 and 13.

Table 1. Average frame rates for rendering IBOs on a PC

Description	old clock	Venus	Helmet
Frames/sec.	7.13	6.26	8.29

6. Conclusions and Future Work

We presented a new compact, image-based representation for three-dimensional objects with complex shapes that can be visualized from arbitrary viewpoints using an extension of McMillan and Bishop's visibility algorithm. We demonstrated that even very complex shapes can be faithfully reconstructed from a relatively small number of samples. We showed that IBOs can be rendered at interactive rates on a PC. We also described how such objects can be translated, rotated, and scaled to produce more complex scenes that can possibly contain polygonal objects. For such scenes, we presented a new list priority algorithm for non-interpenetrating objects, whose rendering primitives are the objects themselves. Since no preprocessing is involved, the algorithm can be used for dynamic scenes. A much simpler approximation algorithm was also presented. We showed how back face culling can be applied to IBOs, leading to a 16.6% expected speedup for objects whose samples are evenly distributed among the faces.

We demonstrated these concepts implementing a prototype for building and warping image-based objects and scenes. We implemented different approaches for object construction: from images for which their corresponding depth buffers are available, from data acquired using a laser range finder, and from a modified ray tracer

The use of better acquisition strategies (for instance, based on Cyberware scanners) can provide registered seamless color images to greatly improve the appearance of IBOs constructed from real objects. The use of variable splat sizes and anti-aliasing techniques can also improve the final appearance of IBOs in general.

An interesting problem is how to guarantee consistent illumination in a scene composed of multiple IBOs. In this work, objects were rendered using their original shading during acquisition time. Reconstruction of normal approximations from point clouds [1] [6] and from polygonal meshes [12] is a

relatively straightforward task. If specular effects are factored out (e.g., acquiring images with ambient light only), conventional shading models can be applied to IBOs. Casting shadows, however, would require more elaborated solutions.

A possible way to speedup the rendering of IBO scenes is to use levels of detail for rendering distant objects. The authors have successfully implemented a mipmap version of an image warper by filtering both color and disparity values on the fly. Its adaptation to layered depth images seems to be relatively straightforward.

The recent interest in web-based applications such as virtual museums and shopping catalogs, puts some importance on the user's ability to interact with compact three-dimensional object representations whose rendering requires no specialized graphics hardware. For these applications, and also for the game industry, the image-based object approach presented in this paper seems to be an attractive alternative.

Acknowledgements

We would like to thank Nick England, Anselmo Lastra, and the UNC IBR group for their assistance and support. Special thanks to Lars Nyland for his enthusiasm and help with the laser experiments. Discussions with David McAllister resulted in the approximation algorithm. The old clock model was provided by Amazing 3D Graphics, and the Venus model was provided by REM Infográfica.

This work was sponsored by CNPq/Brazil – Process # 200054/95. Additional support provided by DARPA under order # E278 and NFS under grant # MIP-961. Intel generously donated equipment used in this research.

References

[1] Amenta, N., et al.. *A New Voronoi-Based Surface Reconstruction Algorithm*. Proc. SIGGRAPH 98 (Orlando, FL, July 19-24, 1998). In Computer Graphics Proceedings. Annual Conference Series, 1998, ACM SIGGRAPH, pp. 415-422.

[2] Arman, F., Aggarwal, J. Model-Based Object Recognition in Dense Range Images – A Review. ACM Computing Surveys, Vol. 25, No. 1, March 1993. pp. 5-44.

[3] Dally, W., et al. The Delta Tree: An Object-Centered Approach to Image-Based Rendering. MIT AI Lab Technical Memo 1604.

[4] Fuchs, H. et al. *On Visible Surface Generation by A Priori Tree Structures*. Proc. SIGGRAPH 80. In Computer Graphics Proceedings. Annual Conference Series, 1980, ACM SIGGRAPH, pp. 124-133.

[5] Gortler, S., et al.. *The Lumigraph*. Proc. SIGGRAPH 96 (New Orleans, LA, August 4-9, 1996). In Computer Graphics Proceedings. Annual Conference Series, 1996, ACM SIGGRAPH, pp. 43-54.

[6] Hoppe, H., et al.. *Surface Reconstruction from Unorganized Points*. Proc. SIGGRAPH 92 (Chicago, IL, July 26-31, 1992). In Computer Graphics Proceedings. Annual Conference Series, 1992, ACM SIGGRAPH, pp. 71-78.

[7] Kolb, C. Rayshad User's Guide and Reference Manual. Draft 0.4, January 10, 1992.

[8] Levoy, M., Hanrahan, P. *Light Field Rendering*. Proc. SIGGRAPH 96 (New Orleans, LA, August 4-9, 1996). In Computer Graphics Proceedings. Annual Conference Series, 1996, ACM SIGGRAPH, pp. 31-42.

[9] McMillan, L., Bishop, G. *Plenoptic Modeling: An Image-Based Rendering System*. Proc. SIGGRAPH 95 (Loas Angeles, CA, August 6-11, 1995). In Computer Graphics Proceedings. Annual Conference Series, 1995, ACM SIGGRAPH, pp. 39-46.

[10] McMillan, L. Computing Visibility Without Depth.UNC Computer Science Technical Report TR95-047, University of North Carolina, October 1995.

[11] McMillan, L., Bishop, G. Shape as a Perturbation to Projective Mapping. UNC Computer Science Technical Report TR95-046, University of North Carolina, April 1995.

[12] Oliveira, M., Bishop, G. *Dynamic Shading in Image-Based Rendering*, UNC Computer Science Technical Report TR98-023, University of North Carolina, May, 1998. http://www.cs.unc.edu/~oliveira/TR98-023.pdf

[13] Pulli, K., et al. *View-Based Rendering: Visualizing Real Objects from Scanned Range and Color Data*. Proceedings of the 8th Eurographics Workshop on Rendering. St. Ettiene, France, June 1997.

[14] Rademacher, P., Bishop, G. *Multiple-Center-of-Projection Images*. Proc. SIGGRAPH 98 (Orlando, FL, July 19-24, 1998). In Computer Graphics Proceedings. Annual Conference Series, 1998, ACM SIGGRAPH, pp. 199-206.

[15] Shade, J., et al. *Layered Depth Images*. Proc. SIGGRAPH 98 (Orlando, FL, July 19-24, 1998). In Computer Graphics Proceedings. Annual Conference Series, 1998, ACM SIGGRAPH, pp. 231-242.

[16] Sutherland, I. et al. A Characterization of Ten Hidden-Surface Algorithms. Computing Surveys, Vol. 6, No. 1, March 1974.

[17] Torres, E.. *Optimization of the Binary Space Partition Algorithm (BSP) for the Visualization of Dynamic Scenes*. Proc. EUROGRAPHICS'90 (Montreux, Switzerland, September 4-7, 1990), pp. 507-518.

[18] Woo, M., et al. *OpenGL Programming Guide*. 2nd edition. Addison Wesley, 1997.

Appendix A

Theorem: Let B' be the base of pyramid $P_{B'}$ intersected by the segment connecting the positive epipole and its parallel projection into K. Then, warping the faces of the cube in the order (K, B, A, A', B', F), or (K, B, A', A, B', F) produce correct visibility from the desired view position.

Proof: Consider the four planes that split the cube into pyramids (figure 3). The pyramid containing K is the only one that is always separated from the half space, defined by each of the four planes, that contains the desired view position. Thus, its samples are the only ones that cannot occlude samples from any of the other five pyramids. Therefore, K must be warped first. On the other hand, the pyramid containing F is the only one that always falls into the same half space, defined by each of the four planes, that contains the desired view position. Thus, its samples are the only ones that can potentially occlude samples from all the other five pyramids. Therefore, F must be warped last.

After removing the pyramids containing K, and F, only two of the four planes are necessary to divide the space into four disjoint subspaces, such that the pyramids containing A, A', B, and B' all fall into different subspaces. The pyramid containing B is the only one that is always separated from the half space, defined by each of the two planes, that contains the desired view position. Thus, its samples cannot occlude samples from any of the other three pyramids. Therefore, B should be warped second. On the other hand, the pyramid containing B' is the only one that always falls into the same half space, defined by each of the two planes, that contains the desired view position. Its samples can potentially occlude samples from pyramids containing A and A'. Therefore, B' must be warped fifth.

Two opposite pyramids not containing the epipoles cannot occlude each other, since there is a plane passing through the desired COP and the apices of the pyramids that separates them into different half spaces (actually, there are infinite many such planes). Thus, two opposite pyramids project into distinct portions of the desired view plane, and therefore cannot occlude each other. Thus, it is safe to warp either A third and A' fourth, or A' third and A fourth.

Notice that, at first glance, it seems that A, A', B, and B' can be warped in any order, given that K is warped first, and F is warped last. However, this is true only when the positive epipole falls exactly at the center of face F.