A Multi-Cellular Orthographic Projection Approach to Image-Based Rendering

Carlos Saraiva	João Fradinho Oliveira	João Madeiras Pereira
IST/INESC-ID	IST/INESC-ID Rua Alves Redol, 9 1000-029 Lisboa	IST/INESC-ID
carlos.saraiva@ist.utl.pt	joao.oliveira@vimmi.inesc-id.pt	jap@inesc.pt

ABSTRACT

Image-based rendering (IBR) techniques take the approach of rendering new images based on existing ones, effectively separating the rendering complexity from the geometric complexity of the scene they represent. With the increasing complexity of scenes currently created, these techniques have gained renewed interest. This paper presents a multi-cellular IBR approach, which uses orthographic rather than perspective projected pre-computed images. Specifically we extend our single cell orthographically projected IBR approach to a multi-cell system, and show how the pixel error can be further reduced with the use of smaller and more numerous cells.

Keywords

Image-based rendering, orthographic projection, multi-cellular, real-time, impostors

1. INTRODUCTION

Despite the fast advance of hardware capabilities, the complexity of the scenes to render has also grown extremely fast. Even with today's modern hardware, complex scenes cannot be rendered in real-time by brute-force methods.

This introduces the need of simplifying the scene to be rendered, reducing its inherent complexity to levels that can be rendered faster, with as little loss in visual quality as possible.

While other efficient techniques have been described, to address the problem of massive data set viewing, they present some problems: they can be sensitive to the type of scene (some techniques don't deal well with CAD, others don't deal well with complex interiors, etc); some require manual intervention (selecting areas of interest) or have navigational constraints (i.e. the camera can only be placed in certain regions). As such, this paper presents an experiment to develop a method that does not suffer the aforementioned problems.

In our approach, we opted for a class of techniques labeled image-based rendering. This class of techniques strives to render the scene from a certain viewpoint given renders of the scene from other viewpoints. From these other renders, information of the scene can be extracted and reprojected into the new viewpoint. In previous work, Saraiva et al. presented a single cell IBR approach based on perspective projection of pre-computed orthographic projected images [Saraiva09] and have shown that the system created less pixel error than perspective projecting perspective projected images (see Figure 1 and 2). This gives the advantage of dealing well with any type of scene as long as it is renderable.

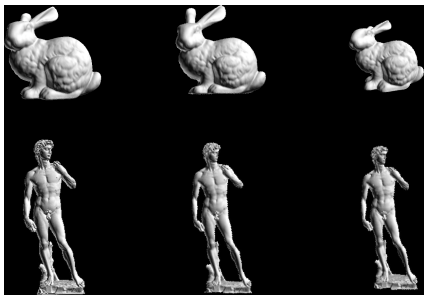

Figure 1. The Bunny and David models. The left shows the original model triangle rendering, the middle is with a texture generated with an orthographic projection and right shows with the texture generated with a perspective projection.

In this article we extend the system to a multi-cell IBR approach and show how the error is further redu-

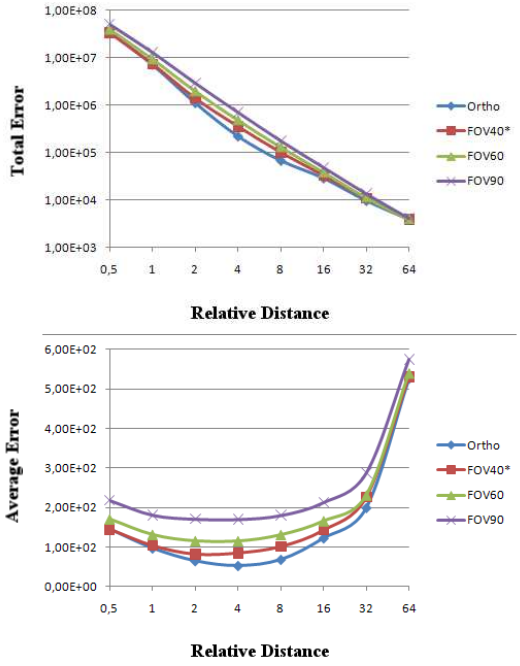

Figure 2. Single cell IBR - Pixel Error: For each increasing relative distance to the model, a series of images were created and subtracted from perspective triangle rendering with a FOV of 40°. Top: the sum of all pixel differences, where p_{dif} is the sum of absolute differences of each R,G,B components between both images. Bottom: the total error divided by the number of foreground bounding rectangle pixels.

ced with more cells using our system. In section 2 we review previous work. In section 3 we present the outline of the multicell system. In section 4 we address implementation issues specific with the reprojection of multi-cell orthographic images. In section 5 we present results. Finally, we draw conclusions in section 6 we conclude and outline future work in section 7.

2. PREVIOUS WORK

In the category of image-based rendering techniques, many methods can be considered, ranging from those that use no geometry at all (like resampling a set of images given the viewing parameters [McMillan95a] to more hybrid approaches that use both geometry and images to represent the underlying scene [Jeschke02].

It is also important to note that image-based rendering can be more useful when dealing with certain types of scenes which are not handled so well with geometric level of detail techniques. An example of such a scenario is the representation of a model constituted by "rough" curves (like a fractal): too much triangle reduction and what is supposed to be a curve can become noticeably linear; too small of a reduction can result in a great number of triangles to draw. Most image-based techniques do not suffer from this problem, as they are independent of the scene.

The techniques can also be classified by how the images are created: some use a predetermined set of possible viewing positions, creating images that represent more distant zones in the scene [Sajadi09]; others create images of objects or subvolumes of the scene and render the scene by representing each of the objects or subvolumes [Schaufler98].

This last category has the advantage that the scene can be represented from any viewpoint and not only a predetermined set. One possible approach useful for viewing models from an outside point of view is to render the model from a series of viewpoints and use those renders as textures placed into billboards, as seen in [Aliaga99].

In [Saraiva09], the camera positions for the generation of the impostors are chosen as the center of the triangles on an *n*-time subdivided icosahedron, enclosing the object (as seen in Figure 3, left), and at a distance (when using perspective) of the bounding sphere radius divided by the sine of half the field of view angle. For orthographic projection no displacement is made.

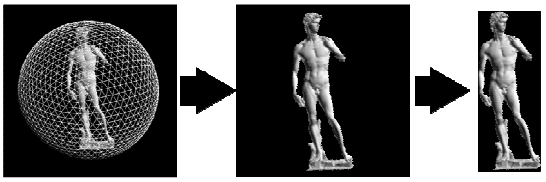

Figure 3. The camera positions; a generated texture; after cropping the texture.

Without correction, this technique can introduce distortion, since the projection of the scene into a plane done during the texture generation can differ from the projection used while viewing the scene with billboards.

This correction is performed by reprojecting an image generated from known viewing conditions into another viewing condition. This reprojection is a function that maps a texel to a number of pixels in the final image. However, several texels can be mapped onto a single pixel, so the texel that has the closest projection is used. Some techniques employ a z-buffer to this end; others are based on analysis of the image [McMillan95b].

Some methods of correction are CPU-based, where each texel in the original image is simply reprojected into the new viewpoint [McMillan97]. While highly parallel in nature, this kind of technique is not trivial

to reproduce in GPU using current graphical APIs, such as OpenGL or DirectX.

In order to exploit the capabilities of modern GPUs, the reverse is usually attempted: finding which source element is the one used in the destination element and sampling it. This problem is not trivial to solve since there is the possibility of the existence of multiple solutions, therefore, a search must be performed for accurate calculation of the source method to use.

These GPU-based methods change the sampled texture coordinate based on a structure that contains the displacement relative to the billboard's normal, typically a texture called a height map (like seen in [Kaneko01]).

Some techniques compensate for this displacement by applying an offset to the texture coordinate based on the view angle relative to the billboard and the height of the billboard at that point [Kaneko01]. While fast, this technique is not accurate and can present incorrect results.

Other techniques strive for absolute correctness, at the cost of performance. Such techniques typically perform a short-distance raycast on the height map, considering points by an increasing order of distance to the camera. If no solutions are skipped due to a large step size, the first solution found is the correct one [Tatarchuk06][Policarpo05]. This raycast process can also be sped up with a pre-computed acceleration structure, as seen in [Baboud06] and [Policarpo07].

3. MULTI-CELL IBR

3.1 Pre-processing

As a first step, the scene is divided into an hierarchy of regular cubic volumes. This can be done by calculating the bounding box of the scene, making it cubic and applying the octree algorithm [Jackins80] to that bounding box, which serves as the root node of the octree. The maximum depth of the octree should be chosen based on the available memory resources, as discussed later.

The second step pre-computes a series of impostor textures for each of these nodes. These impostors are generated using an orthographic projection, since they were found to have the least overall error. The method of generating these impostor textures is similar to the procedure of computing images in the single cell approach; the subdivided icosahedron used for calculating the camera positions is centered in the middle of the octree cell (instead of the centroid of the scene) which is circumscribed by the bounding sphere of the cell.

3.2 Runtime

The octree is traversed and nodes that are not framed by the camera can be culled. If a node that occupies an amount of screen space smaller than a certain threshold is reached, the impostor is used and the traversal stops. At close distances, for full detail, the original model triangle rendering could be used. However, if the choice to not use impostors was only based on distance, that would introduce problems, since then the approach would suffer from high polygonal densities (i.e. areas of the scene that have an extremely high polygon count). Therefore, the nodes that are indeed close enough to be represented geometrically are sorted by priority (distance to the camera) and rendered from the highest priority to the least, until a certain triangle budget is met. All remaining nodes are represented with the impostor with the smallest angle to the view direction.

3.3 Preliminary analysis

By using this approach of creating a representation of the cell as if viewed from the outside as in [Rusinkiewicz00], one gains freedom of navigation, since the camera no longer has to be constrained inside a cell. Also, the algorithm is almost automatic; the only manual intervention needed is setting parameters appropriate to the scene.

Multi-cell IBR solutions, however, are likely to occupy large amounts of disk space. This is accepted due to the relative low cost of disk space in the current days. Also, the algorithm can have an upper limit of disk space usage imposed to it (with a possible loss of performance if the limit is too small for the scene complexity): one can sort the nodes by the amount of primitives they contain. Then, the image generation can be applied to the nodes with the greatest number of primitives, in order, until a disk space budget is reached. It should also be noted that the method should not be used solely with impostors. It uses them to remove most of the scene complexity, but it should be complemented with other techniques (like geometric LODs, for example). However in [Saraiva09] we were somewhat surprised at the visual quality that was possible up close when examining 3D models with the reprojection of orthographic images, hence in this work we did not integrate LOD techniques, and aimed at improving the visual quality of less computationally intensive techniques such as IBR. Indeed in section 5 (results) we show that the combination of smaller IBR cells allows one to achieve this result.

3.4 Summary of steps

In pre-process: 1 – read geometry; 2 – create octree; 3 – calculate camera positions using subdivided icosahedron; 4 – texture creation and cropping.

At runtime: 1 – frustum culling; 2 – node sorting based on distance; 3 – for each node choose image with smallest angle to view direction.

4. MULTI-CELL IBR IMPLEMENTATION ISSUES

In this section we address implementations issues that arise when assembling multiple cell images of orthographic projected images. In section 4.1 we present the problem, in section 4.2 we consider a run-time solution using raycasting, and in section 4.3 our adopted solution.

4.1 Inter cell artifacts

A problem that arose on the multi-cellular approach (that was not present in the single-cell approach of [Saraiva09]) was the presence of "black line" artifacts on the model. An example of this artifact can be seen in Fig 4, which shows vertical and horizontal black lines spanning the Batalha monastery model.

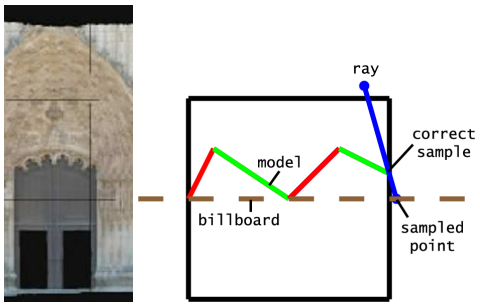

Figure 4. The incorrect sampling problem. The horizontal and vertical black lines artifact visible in the Batalha model (left).

These black lines are actually the edges of the octree node. This is due to the fact that since no correction was performed, the coordinates used to sample the texture were too far away from the center. An illustration of what occurs is depicted in Fig.4 (right).

In the image, the black box represents the region of space delimited by the octree node, and the red and green lines represent the mesh. The brown line represents the billboard plane and the blue line represents a ray from the camera into the volume. If it was done with the original model triangle rendering, the pixel would be colored green. However, since the geometry in the region is effectively flattened by the billboard process, the texture is sampled outside of the region, which is transparent (or non-existent, if it falls outside the billboard), generating the "black line" artifact, where one is actually seeing the background.

This error lessens as the distance to the region increases, since the rays become more and more parallel and, therefore, closer to the orthographic projection that was used to create the image. In practice, one can limit the maximum amount of error in the sampled coordinates by choosing a large enough distance to the billboard.

Attempts were made to correct this distortion. Like previously referred, this sort of correction can be done via CPU (highly undesirable) or the GPU. Attempts to minimize the error with small amounts of computation failed, as all solutions eventually degenerated to something similar to a raycast (presented in the next section).

4.2 The impracticality of raycast solution

Raycasting can be impractical, since it is a somewhat expensive operation, even in modern GPUs when using acceleration structures. On ideal conditions, the performance should depend only on the resolution of the display. However, this proved not to be true, due to the possibility of overdraw and limitations in the current graphics APIs.

Since we have our space divided into regions, without having any sort of occlusion information, we have to render all the regions (at least those that pass the frustum culling operation). In ideal conditions, only one ray would need to be cast per pixel on the screen. Since we desire the first intersection for that pixel, one could think a way to minimize the number of raycasts would be to order the raycasts for that pixel by sub-volume proximity to the camera. Once a ray hit in a region, that pixel would no longer need to have further raycasts performed.

However, the problem lies in how one can determine if that particular pixel has already seen an intersection. Usually to limit this sort of overdraw, two techniques are used: depth-testing and stencil-testing. One could program the GPU to reject fragments with depth greater than the corresponding fragment on the depth buffer. Something similar applies to the stencil tests. This does not solve the problem, though, since these tests are performed *after* the pixel/fragment shader (and raycast) is run, hence the gain would be nullified since the objective is to not do the raycast at all.

To lessen this effect (discarding fragments after an expensive pixel shader is run), graphics card manufacturers have applied a technique called *Early-Z*. This technique does the depth buffer test after the vertex shader is run, but before the pixel shader. Therefore, it lessens the number of fragments that have to go through a possibly expensive pixel shading.

This technique, however, is not part of the typical graphics pipeline, and so is not adjustable via an API. As such, the hardware can only apply this technique if it is sure that the final result is the same. This can only be done if one knows the depth of the fragment before the pixel shader is run. On most applications, this condition is verified; the depth of the fragment is the result of the interpolation of vertex depths.

By performing a raycast, we find out the correct depth of the fragment and can use that correct depth. But by doing so, one effectively is disabling *Early-Z* and increasing the overdraw rate significantly.

Since one needs to know the depth of the fragment in the framebuffer while executing the pixel shader and given that one cannot read and write into the same texture (or a value in a memory array) on a single pass, the use of shaders would not solve the problem. It is possible that more general-purpose APIs (like CUDA) provide a solution, but that goes beyond the purpose of the paper, which is to determine if the approach is viable or if it should be simply discarded.

The problem described, however, has a relatively simple workaround: although one might not know the exact depth of the fragment, one can usually bound that depth between two extreme values. If the Early-Z test fails on both extreme values, any value in between would also fail and the fragment could be discarded. This is not yet possible, however, due to limitations of current graphical APIs such as Direct3D and OpenGL, but is subject to change in the future. Even if we don't change the depth of the fragment (but correct the sampling coordinates), problems can still arise. Without correct depth, interpenetrating objects can display artifacts and techniques that depend on correct depth (like shadow mapping) can become no longer viable. As such, we chose not to correct neither the depth nor the coordinates, to see the viability of the technique with no runtime correction performed at all.

4.3 Cell growing solution

As one can observe, the lines seen due to the artifact are very thin, since the coordinates are only slightly wrong (with an error of one texel, for example). In practice this problem only arises when viewing an object at very close distances.

We found a solution that works well in practice. Instead of using only geometry inside the region to generate the images for the billboard, one can also use the surrounding geometry, which in essence "fills the gap". An illustration of this can be seen in Fig. 6.

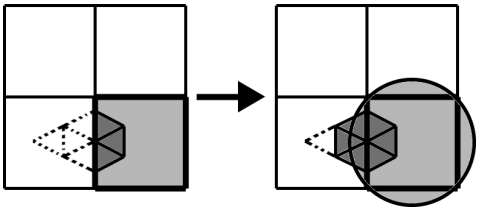

Figure 6. Filling the gap. The thick lines delimit the cell considered. In the left, the geometry is only considered in the area inside the cell (gray), while on the right, the surroundings are also considered.

Although visually incorrect, it tends to be an effective solution since the gap is generally small and color changes are usually smooth enough. To this end, the triangles considered to create the billboards were not only the ones in the cubic volume but the ones that intersect that cubic volume's bounding sphere. In Figure 7 we see a render of the same model after this correction is applied.

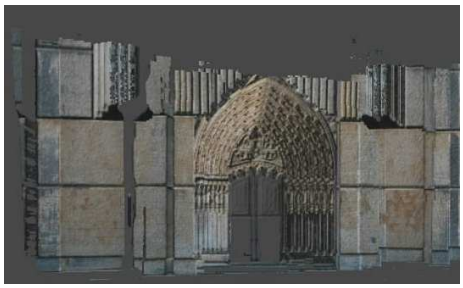

Figure 7. The "black line" artifact invisible, by filling the gap with surrounding geometry.

5. RESULTS

Figure 8 shows that the visual error associated with single cell orthographic projected IBR can be reduced by using more cells.

The test was done with the David model (8 million triangles), using an octree of depth 4, for a total of 218 cells (empty cells are ignored). The positions were chosen with a twice-divided icosahedron, which results in 320 images per cell. At a resolution of 128×128, after cropping, the average size ended up at 11MB per cell (2.41GB total). No compression was implemented, however a fast ZIP compression (using the deflate compression method [Deutsch96], with a 32KB dictionary and a word size of 32) reduced that size to less than a fifth (~18%). A slower and better compression further halved that size. This high compressibility, along with the low cost of disk space on the current days, makes the uncompressed large disk space occupation more acceptable.

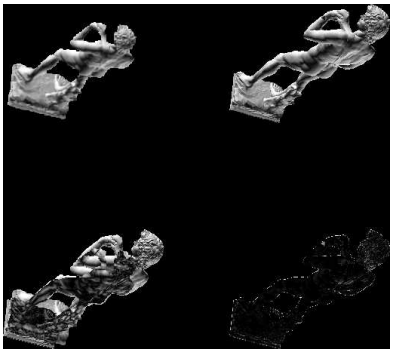

Figure 8. The David model in a single cell (left) and multi cell (right) representation. The top shows the render, the bottom shows the difference to the original model triangle rendering.

To test the effectiveness of this multi-cellular approach in solving the problem seen in the David and Batalha models, due to their large range of depths at certain angles, the camera was placed so that the vertical component was greater (to maximize the error) while still being able to see most of the model, at a relative distance of 1. The result can be seen in Figures 9-13 (cropped to fit the content).

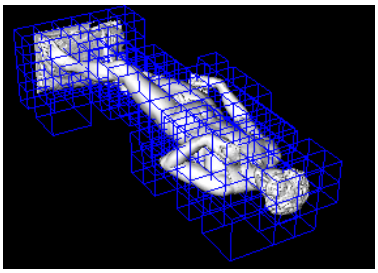

Figure 9. The wireframes of the 218 nodes.

Figure 10. The orthographic impostor approach, from another angle.

Figure 11. From the same angle as figure 10, with the original model triangle rendering.

Figure 12. The difference between figures 10&11.

Figure 13. The outline of the 218 billboards used to generate figure 10.

As one can see, this multi-cellular approach indeed lessened the problem of perspective error. This is due to two facts:

- since there are more cells, each cell occupies a smaller arc in the field of view, bringing it closer to the "original" orthographic projection, the difference with perspective multi-cell IBR can be seen in Figure 14.
- since the cells are smaller, the available gamut of depths is also reduced and so the flattening of the mesh does not change the depth significantly.

In terms of performance, the transfer of texture memory to the graphics card is fast enough to give the viewer 100+ FPS with a C++ implementation using OpenGL on an Intel E7200 with 2GB of main memory and a NVIDIA GeForce 8800GT. To stress the bandwidth limitations between the graphics card and main memory, the graphics card constantly deleted the textures after each render, requiring it to be reuploaded. A problem arises however in disk transfer bandwidth. When the camera rotates around the model too quickly, one texture for each of the 218 cells has to be loaded from disk, bringing performance down to about an average of 5FPS. This problem can be alleviated by caching [Yoon05].

It is hypothesized that this bandwidth problem could be greatly lessened if the textures were compressed and if they were prefetched (like TetraPuzzles [Cignoni04] does). We also note that when rendering triangles over IBR, fewer textures are needed hence further speeding up performance.

In terms of image quality, the lack of correction still brings some visual artifacts, as can be seen in the left wrist area in Figure 10 (with the difference clearly shown in Figure 12). This might not pose a problem, however: as the distance increases, these artifacts get smaller (in number and size), eventually making them unnoticeable. Since the technique should be complemented by some other method for nearby regions, these areas of larger distortion are replaced by correct pixels. An example of the triangle

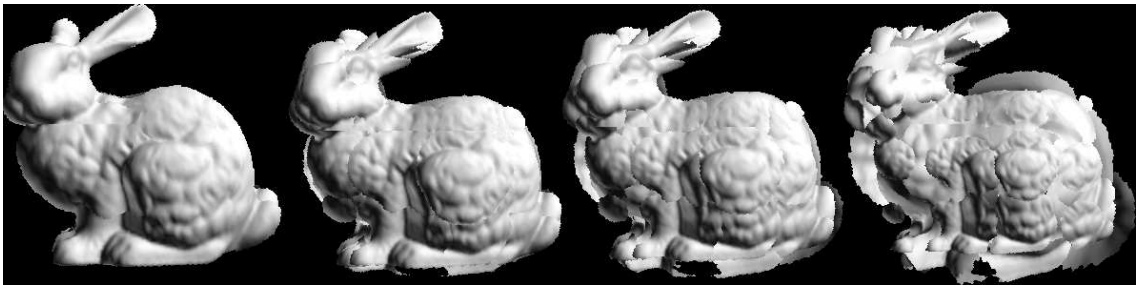

Figure 14. Multi-cell IBR (50cells with 320 images each). From left to right, perspective projection with 60° FOV of: Orthographic projected cells, Perspective projected cells of 40°, 60° and 90° respectively.

representation used at nodes where the center goes below a certain distance threshold to the camera (with impostors used in other cases) can be seen in Fig. 15.

Figure 15. The impostor solution with and without triangles drawn over (right and left, respectively). Notice the artifact on the left wrist disappears.

Figures 16-17 show the David model from another angle, but twice as close as the distance in Figures 9-13.

Figure 16. The David model, from yet another angle, but at a relative distance of 0.5 (impostor on the left and original model triangle rendering on the right).

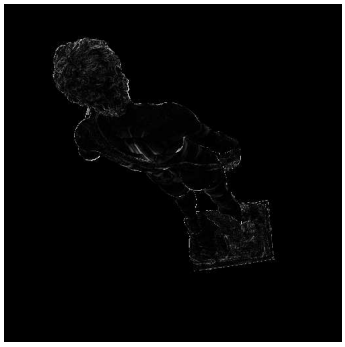

Figure 17. The difference between the two representations in figure 16.

6. CONCLUSIONS

In this paper, an experiment on the usage of simple multiple cell impostors (billboards) was described, in order to determine the feasibility of such a solution for real-time rendering of massive data sets. We have shown that with more cells the visual error is reduced when compared to single cell approaches.

The solution implemented, as described, possesses some potentially great limitations: large use of disk space, bandwidth problems when the camera moves quickly, at close distances visual artifacts can become noticeable. However, being an image-based rendering approach, it also possesses one great quality: the performance of a cell is independent of the number of primitives it contains.

Some of the limitations can be lessened: disk space usage and bandwidth issues can be lessened by the use of compression and caching. Other issues, like visual artifacts at short distances are not so easy to correct, requiring a significant amount of computational power (whether in CPU or GPU). However, if we limit the use of impostors to large distances, the bandwidth and artifact problems are greatly reduced: if an impostor is far away, one would have to move the camera at a very high speed in order to force a large number of texture loads from disk. Also, at great distances the visual artifacts are minimal or non-existing, since the camera "rays" are nearly parallel, like the orthographic projection which gave origin to the image used on the billboard.

We have also concluded that for a multi-cell approach, an orthographic projection produces less error than a perspective projection, as visible in Figure 14.

7. FUTURE WORK

In terms of future work, there are quite a few avenues of possible improvement. In no particular order:

- implementation and analysis of the effects of compression;
- implementation of a caching mechanism;

- researching and implementing some form of correction that is less computationally expensive
- implementing a current form of correction while exploiting future new capabilities given by APIs, like the range-limited Early-Z mentioned;
- integrate a complementary method to the solution described in order to see how well the technique performs in a more ideal situation;
- research some more compact representation of the contents of the cell (like the view-dependent voxel in Far Voxels [Gobbetti08].

8. ACKNOWLEDGMENTS

We would like to thank INESC-ID and its VIMMI research group for the resources made available in the writing of this paper.

We would also like to thank Instituto Português do Património Arquitectónico (IPPAR), now incorporated in Instituto de Gestão do Património Arquitectónico e Arqueológico (IGESPAR) and "Artescan, Tridimensional Digitization" for the scanned model of the Batalha monastery. For the David scanned model, we also thank the Stanford University. The work presented in this paper was funded by the Portuguese Foundation for Science and Technology (FCT, which we also thank), VIZIR project grant (PTDC/EIA/66655/2006).

9. REFERENCES

[Aliaga99] Aliaga, D. Automatically reducing and bounding geometric complexity by using images. PhD thesis, University of North Carolina at Chapel Hill, 1999.

[Baboud06] Baboud, L., Décoret, X. Rendering geometry with relief textures. Proceedings of Graphics Interface 2006, 2006, pages 195-201.

[Cignoni04] Cignoni, P., Ganovelli, F. et al. Adaptive TetraPuzzles – efficient out-of-core construction and visualization of gigantic polygonal models. SIGGRAPH '04: ACM SIGGRAPH 2004 Papers, 2004, 796-803.

[Gobbetti08] Gobbetti, E., Kasik, D. et al. Technical strategies for massive model visualization. SPM '08: Proceedings of the 2008 ACM symposium n Solid and physical modeling, 2008, 405-415.

[Jackins80] Jackins, C., Tanimoto, S. Oct-trees and their use in representing three-dimensional objects. Computer Graphics and Image Processing, 1980, 249-270.

[Jeschke02] Jeschke, S., Wimmer, M., Schuman, H. Layered environment-map impostors for arbitrary scenes. Proc. of Graphic Interface, 2002, 1-8.

[Kaneko01] Kaneko, T. Takahei, T. et al. Detailed Shape Representation with Parallax Mapping. Proceedings of the ICAT 2001, 2001, 205-208.

[McMillan95a] McMillan, L., Bishop, G. Plenoptic Modeling: An Image-Based Rendering System. Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, 1995, 39-46.

[McMillan95b] McMillan, L., Bishop, G. Head-tracked stereoscopic display using image warping. Proceedings of SPIE, 2409 (1995), 21-30.

[McMillan97] McMillan, L. An image-based approach to three-dimensional computer graphics. PhD thesis, Universityof North Carolina at Chapel Hill, 1997.

[Policarpo05] Policarpo, F., Oliveira, M. et. al. Real-time relief mapping on arbitrary polygonal surfaces. Proceedings of the symposium on Interactive 3D graphics and games 2005,155-162.

[Policarpo07] Policarpo, F., Oliveira, M. Relaxed cone stepping for relief mapping. 2007,GPU Gems3.

[Rusinkiewicz00] Rusinkiewicz, S., Levoy, M. Qsplat: a multiresolution point rendering system for large meshes. SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques, 2000, 343-352.

[Sajadi09] Sajadi, B., Huang, Y. et al. A Novel Page-Based Data Structure for Interactive Walkthroughs. Proceedings of the symposium on Interactive 3D graphics and games, 2009, 23-29.

[Saraiva09] Saraiva, C., Oliveira, J., Pereira, J., Araújo, B. A Comparison of Orthographic and Perspecive Projections in the Generation of Textures for Billboards. Proceedings of Encontro Português de Computação Gráfica, (17), 2009.

[Schaufler98] Schaufler, G. Image-based representation by layered impostors. Proceedings of the ACM symposium on Virtual reality software and technology, 1998, 99-104.

[Tatarchuk06] Tatarchuk, N. Dynamic Parallax Occlusion Mapping with Approximate Soft Shadows. Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games, 2006, 63-69.

[Yoon05] Yoon, S.-E. Interactive visualization and collision detection using dynamic simplification and cache-coherent layouts. PhD thesis, University of North Carolina at Chapel Hill,2005.