Chapter 7

Results

I have compared face cluster radiosity to my own implementations of progressive radiosity, and hierarchical radiosity with volume clustering, all of which are handled by a program I call “Radiator”. My implementation of volume clustering follows those of Bekaert and Christensen [Chri95, Beka]. I also compared my results to Greg Ward’s “Radiance” package [Ward94, Warda], a ray-tracing-based renderer which uses diffuse sample caching to perform radiosity calculations quickly. It is the most sophisticated of the various monte carlo-based sample-caching renderers, with respect to diffuse illumination. Finally, to validate my own implementation of hierarchical radiosity, I also used “RenderPark” for some experiments [Beka]. RenderPark is a freely available global-illumination renderer that, amongst other things, performs volume-clustering radiosity with the M2 wavelet basis.

Both the radiosity packages were set to use three gather iterations for hierarchical radiosity, and Radiance was likewise set to calculate to three diffuse bounces of light. The approximate visibility strategy discussed in **Section 6.3.3** was not used, both to provide a fairer comparison to the other methods and so it would not obscure the effects of using face clusters as finite elements. (See, however, **Section 7.4**.) Other settings used for the various renderers, along with the steps necessary to produce consistent output amongst them, are outlined in **Appendix B**.

7.1. Simple Reflectance Tests

I carried out a number of experiments using variations of a simple scene containing three test patches; see **Figure 85**[1]. This was done to test that small-scale interreflection was being handled consistently, and as a validation test amongst the three renderers. We would obviously like to ensure that face-cluster radiosity produces the same results as other global illumination methods.

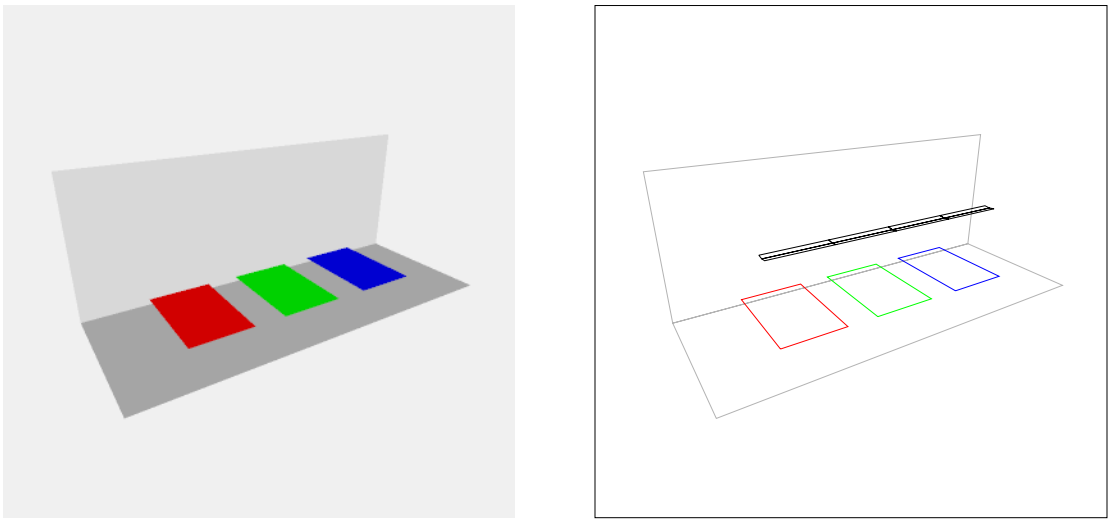


Figure 85: The reflectance test scene. Three patches corresponding to the primary colours reflect light from a long horizontal overhead luminaire onto a grey rear wall.

7.1.1. Flat Test Patches

We start by using a single, flat quadrilateral for the test patch. The main purpose of this test is to show that all three renderers produce consistent results, and that the irradiance vector-based face cluster method produces the same results as ordinary, scalar radiosity. The results are shown in **Figure 86**. As we can see, all images look nearly identical. The Radiance picture suffers from noise artifacts,

1. All results were collected on a Dell Precision Workstation 610 with a 450MHz PIII processor and 256MB of main memory, running RedHat Linux 6.1.

giving the rear wall a dappled look, especially near where the plane of the test patches intersects the rear wall. However, it has high-quality shadows.The two Radiator methods produce good strong diffuse reflections, although the shadows cast by the patch are a little softer than the Radiance solution. The RenderPark picture is probably the cleanest of the three, its M2 basis being well suited to this scene. We can conclude that all methods produce consistent results for this simple test.

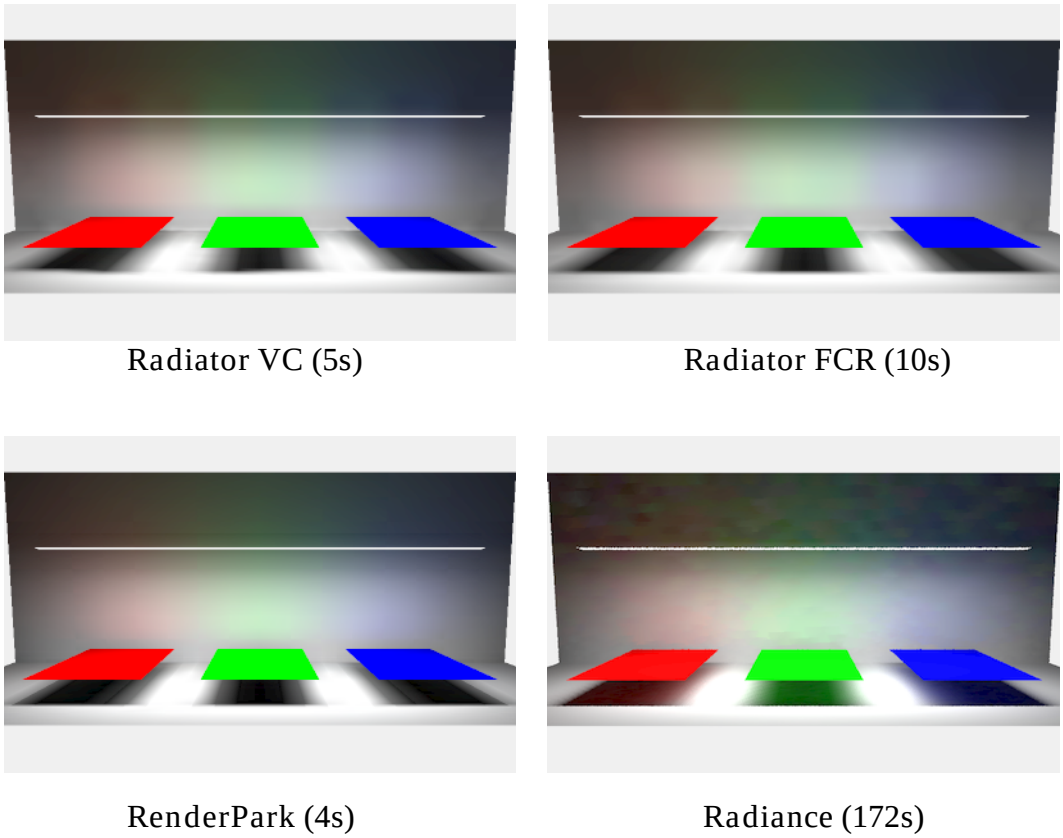


Figure 86: The flat patch scene. RenderPark does the best in terms of quality and speed. The face cluster method takes twice as long as the volume clustering methods due to the extra overhead of the irradiance vector. Radiance does the worst in terms of time and quality.

Radiance takes by far the longest of the methods, but this is partly because it has to ray-trace the final image, and it has the advantage that, as we shall see, its rendering time doesn't rise much with added scene complexity. Face-cluster radiosity takes twice as long as the two volume-clustering methods, because of the extra overhead of using irradiance vectors.

7.1.2. Bumpy Test Patches

The second test exchanged the flat patches for displacement-mapped sawtooth surfaces; these are the three-dimensional equivalent of the surfaces in **Figure 39**; an example is shown in **Figure 87**. Each bumpy surface contains 8,000 polygons. For diffuse light transfer, these patches are practically equivalent to the previous flat patches; solving the radiosity problem for this scene should ideally be no more complicated than the flat-patch scene. The results are shown in **Figure 88**.

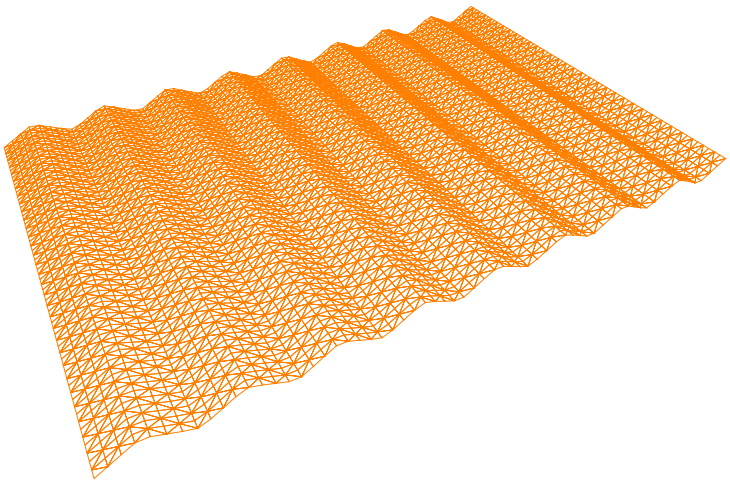
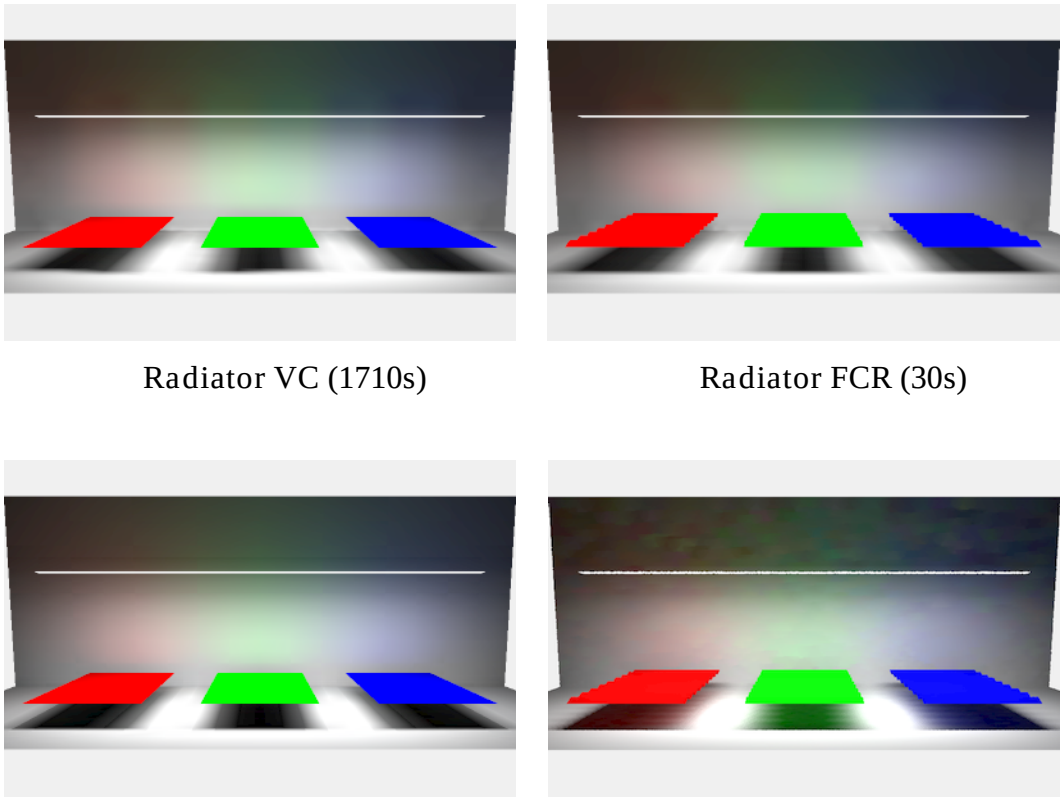


Figure 87: The sawtooth surface used in the "bumpy" test scene.

The most interesting result here is that the two volume clustering methods are very slow on this seemingly innocuous scene. As we shall see in the next test, their poor times are not due to the polygon count. The reason is the geometry of the bumpy patch. Because the volume clustering methods use the total projected



Radiator VC (1710s) Radiator FCR (30s)

RenderPark (4823s) Radiance (284s)

Figure 88: The bumpy patch scene. The volume clustering methods are very slow on this scene. Radiance is fast, but the rear-wall reflection is somewhat splotchy. Face cluster radiosity is even faster.

area (NUPA) in a particular direction to estimate radiosity transfer, rather than the total *visible* projected area (VPA), they can heavily overestimate the transfer parallel to the patch. (In effect they count each bump in a cluster as contributing energy to adjacent clusters, rather than just the bumps at either end.)

This results in over subdivision of the patch in an attempt to find the amount of radiosity it reflects to itself, and the consequent poor rendering time performance. On the other hand, the bounded visible projected area estimates introduced in **Section 4.3** prevents face cluster radiosity from suffering from this

kind of problem; they correctly estimate the side area of the bumpy patch clusters, requiring fewer interreflection links.

For these images, each program was run such that approximately equal quality resulted. If we had normalised on speed instead, HRVC and Radiance would have looked much worse.

We conclude from this test that the higher quality approximations being made by the face cluster radiosity algorithm permit it to represent light transport using a shallower tree than HRVC, saving significant time and memory.

7.1.3. Whale Test Patches

As a final stress test, the medium-resolution displacement-mapped patches were replaced with high-resolution whale models, each containing 100,000 polygons. The whales provide more realistic objects, being very much the kind of model used in a high quality scene, while the test setup still allows us to check that colour bleeding is still occurring properly, and for consistent results between renderers. **Figure 89** shows the results for both a simplified, 1,000 polygon of the original model, and the full, detailed model. **Figure 90** shows a close-up of the whales of the face cluster radiosity solution.

While the simulation times for face cluster radiosity and the two hierarchical radiosity with volume clustering methods are similar for the simplified, low-detail whales, the times diverge dramatically for the more complex scene, even though both were run with the same refinement epsilon as for the simpler scene, and both produce similar results. Face cluster radiosity takes hardly any more time at all to deal with the complex model, while RenderPark in particular suffers from a twenty times increase in solution time. (Most of the increase in the rendering time for face cluster radiosity can be attributed to the greater expense of visibility queries for such a large scene.

Radiance does well with the full resolution whales compared both to the simplified whale scene and the previous bumpy-patch scene; its rendering time increases by about half. Its under-fin shadows are the best of the methods, although the softer shadows start to get a little noisy. The linear basis functions of RenderPark have some problems with the shadows in this scene; inter-patch discontinuities are obvious in several places.

The statistics for the renderers are shown below in **Table 10**, including the preprocessing time for the Radiance and FCR methods. (With Radiance you must build an on-disk octree of the scene before running the actual renderer, using the program *oconv*.) Interestingly, this preprocessing time is roughly the same for both

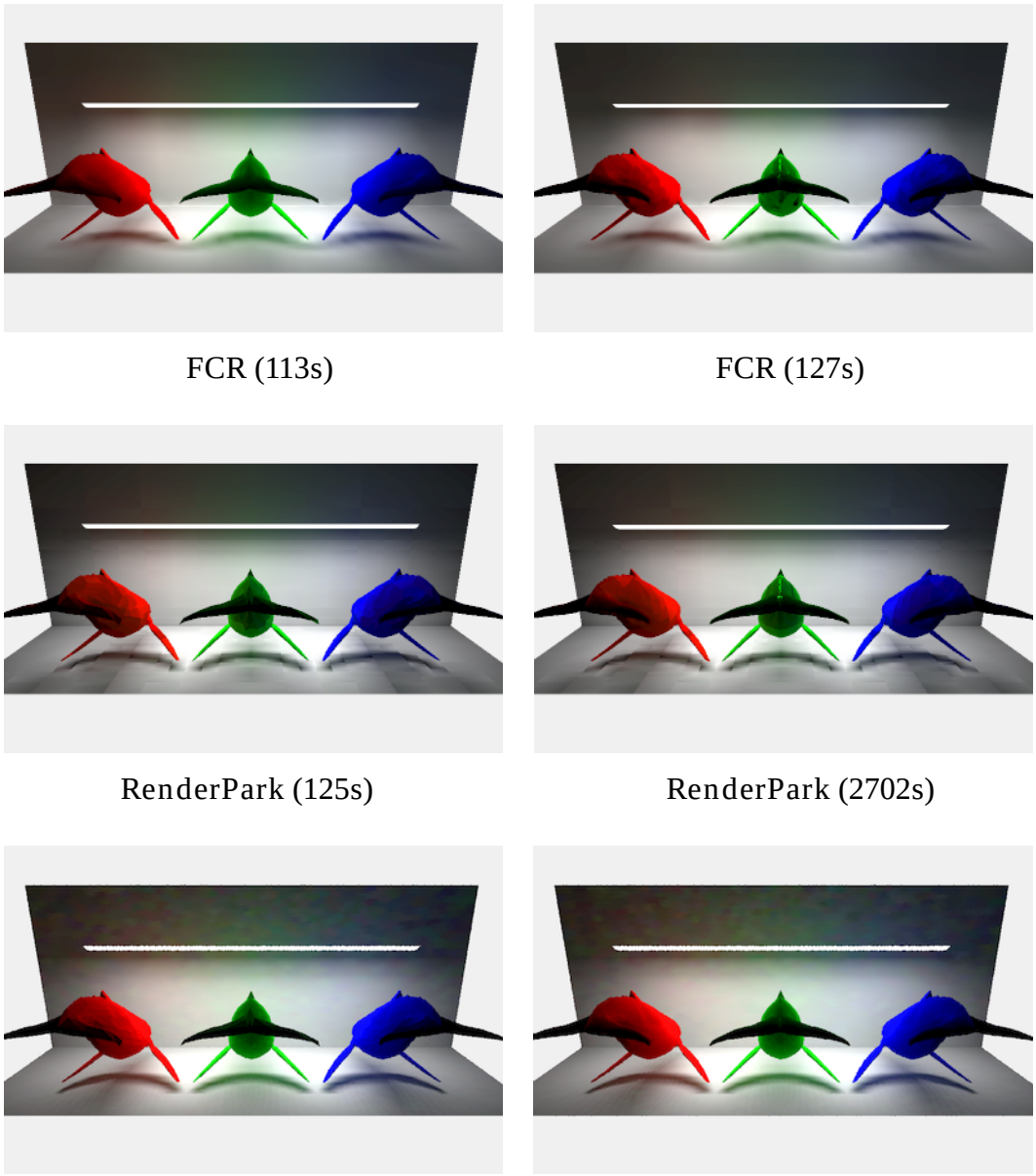


Figure 89: The test patch whales. These magnificent creatures of the deep gracefully reflect light onto both the rear wall and the floor. Left: Results using a simplified 1,000 polygon whale model. Right: Results for the full scene; each whale contains 100,000 polygons.

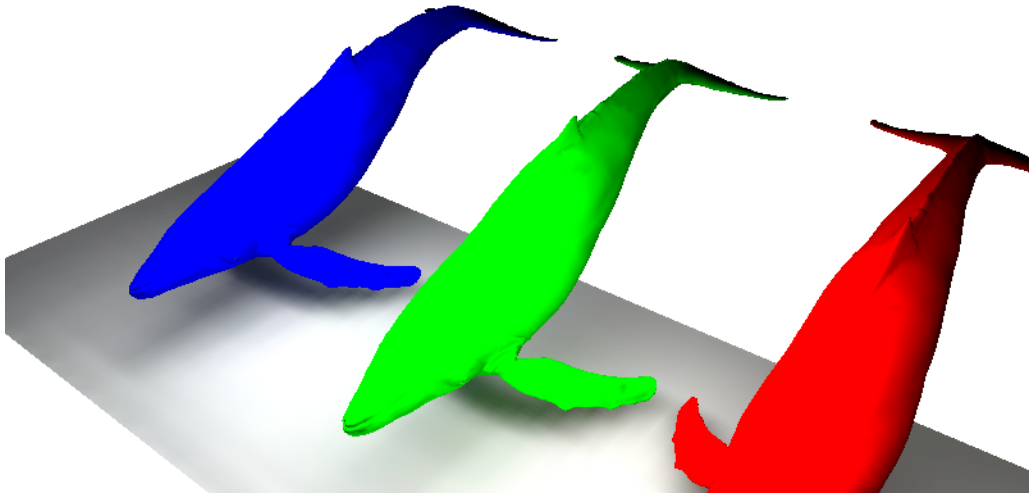


Figure 90: A whale close-up from the FCR solution.

methods; though the face-clustering time is shown as one third of the octree-building time, this is because the whale's cluster hierarchy is reused across the three instances of the whale in the scene.

	Simple whale		Complex whale	
Renderer	**Rendering**	**Preproc.**	**Rendering**	**Preproc.**
Radiator Vol. Clustering	111s		953s	
RenderPark	125s		2702s	
Radiance	248s	1.5s	378s	76s
Face Cluster Radiosity	113s	0.2s	127s	25s

Table 10: Rendering times for the Test Whale scene.

7.2. The Museum Scene

For a more general test of renderer performance, I designed an indoor scene more typical of those seen in the radiosity literature. The scene contains a number of high resolution scanned models, a polygonized implicit surface (the podium), and a displacement-mapped surface (the floor). These models range in complex-

ity from 4,140 polygons to 100,000 polygons. For the lighting design, there are three overhead spotlights illuminating in turn, the Isis statue, the Buddha statue, and the table. The Venus head is illuminated by a floor-mounted red spotlight.

The scene is available in the MGF format [Wardb] in a number of different resolutions from the web page http://www.cs.cmu.edu/~ajw/thesis/museum-scene/.

7.3. Empirical Complexity

Using the quadric-based surface simplification method [Garl97], I applied all renderers to versions of the museum scene with varying polygon counts, to demonstrate the effect of using ever more detailed models on these algorithms[1]. Because the Radiator algorithms share the same code base, we can be reasonably sure that any differences in results are not due to any variance in code optimization or approaches to, for example, visibility testing or geometrical representation. I also performed the same tests using Radiance and Renderpark, to verify my results against well-known external code bases.

While previous experiments have investigated the effect of increasing the lighting or geometric complexity of a scene on radiosity algorithms [Will97b], this experiment shows the effect of increasing model complexity in a scene with fixed geometric layout. While exact error comparisons between the solutions are not available due to the difficulty of generating a reference solution, I took care to use similar parameter settings for all three methods. The most complex version of the scene in this experiment was generated by simplifying all large models to 100,000 polygons, and creating multiresolution models from the results. Ten "complexities" of the scene were then generated by simplifying each model to have s^2k polygons, where k was the number of polygons in the highest resolution of the model, and scene n had $s = n/10$. This produced ten scenes with polygon counts ranging from 7,400 polygons to 546,000 polygons. **Table 11** shows the time it took to generate the associated face cluster hierarchies for the highest complexity level, and the size of each file.

Figure 91 shows the results of running the two volume clustering methods on this scene. Interestingly, they both have very poor, super-linear performance in time; this is much worse than the $k\log k$ performance we would hope to see. This occurs for similar reasons as their poor performance in the "bumpy" test patch scene; the flat but bumpy stone floor in the original scene leads to over refinement

1. All results were collected on a Dell Precision Workstation 610 with a 450MHz PIII processor and 512MB of main memory, running RedHat Linux 6.1.

Model	Polygons	Clustering Time
Buddha	100,000	25.3s
Isis	100,000	25.5s
Dragon	100,000	25s
Venus	100,000	25s
Pedestal (under Venus)	24,000	5.7s
Candlestick x 2	4,150	1s
Floor	100,000	18s
Picture x 2	7,400	1.63s
Total	**547,100**	**127.13**

Table 11: Face Cluster Hierarchy Generation

due to overestimates of the side areas of floor clusters, especially with the red spot-light spilling directly onto it. **Figure 92** shows images of the resulting scenes.

To give in some senses a fairer comparison of these methods with face cluster radiosity and Radiance, I reran them on a version of the scene without the detailed stone floor. It was instead replaced with a flat mesh, having the same number of polygons as the original floor in each scene. This gave performance results much closer to those expected for hierarchical radiosity with volume clustering, but it should be noted that in the following results both face cluster radiosity and Radiance are solving a more difficult scene.

To give an idea of the image quality produced by the various methods, **Figure 93** and **Figure 94** show images for the lowest and highest resolution scenes respectively.

Figure 95 shows a graph of solution time for Radiance, volume clustering, and face clustering. (RenderPark was omitted because its solution times were so much larger; on the order of hours.) Notably, the time cost of the face cluster radiosity algorithm stays approximately constant, while the other methods have costs that are linear (volume clustering) and sub-linear (Radiance) in the number of input polygons. As suggested in **Figure 96**, this is because after a certain point, fine levels of detail of the face cluster hierarchy are never accessed, in spite of the increasing polygon count. .

The most important conclusions we can draw from this test are

- unlike previous finite-element methods, face cluster radiosity is sublinear in the number of input polygons;

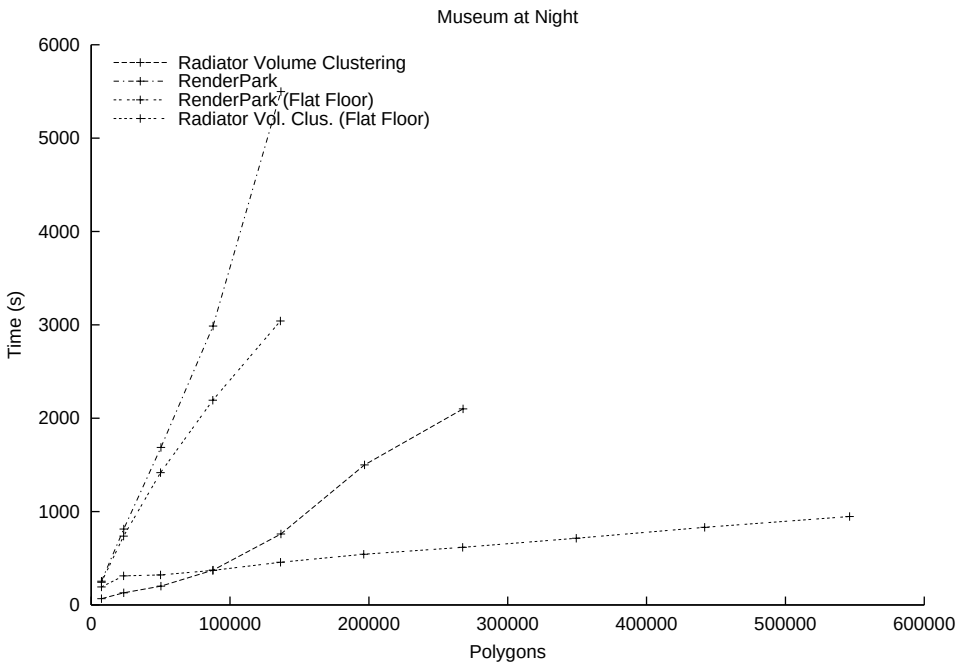


Figure 91: Volume clustering on the museum scene. The bad performance of volume clustering methods on the stone floor is illustrated by the much better performance of the methods when the floor is replaced by a single, flat polygon. In general, RenderPark takes much longer to render the scene than Radiator's Haar-based volume clustering, even though its results are not as good. (See **Figure 92**.)

- beyond a certain point (200,000 polygons in this case) its solution time is independent of the number of input polygons;
- volume clustering methods can do poorly on flat but bumpy surfaces, due to over-refinement;
- Radiance is also sublinear in the number of input polygons, although it does not level off so quickly, and has a higher constant than face cluster radiosity.

Radiator, Haar (760s)

RenderPark, M2 basis (5500s)

Radiator, Haar (207s)

RenderPark, M2 basis (3042s)

Figure 92: Images for the volume clustering methods, medium scene complexity (140,000 polygons). Rendering times are drastically reduced by replacing the detailed stone floor (top) with a flat one (bottom).

Note that overall Radiator gives better results than RenderPark for this scene. It is possible to get higher quality output from RenderPark, but doing would raise the already large rendering time.

Radiator, volume clustering

RenderPark

Radiator, face clustering

Radiance

Figure 93: Images for the lowest scene complexity level for the four methods. This scene contains 7,400 polygons.

Radiator, volume clustering

RenderPark (medium)

Radiator, face clustering

Radiance

Figure 94: Images for the highest scene complexity level. This scene contains 546,000 polygons. Face clustering and Radiance both produce the best results.

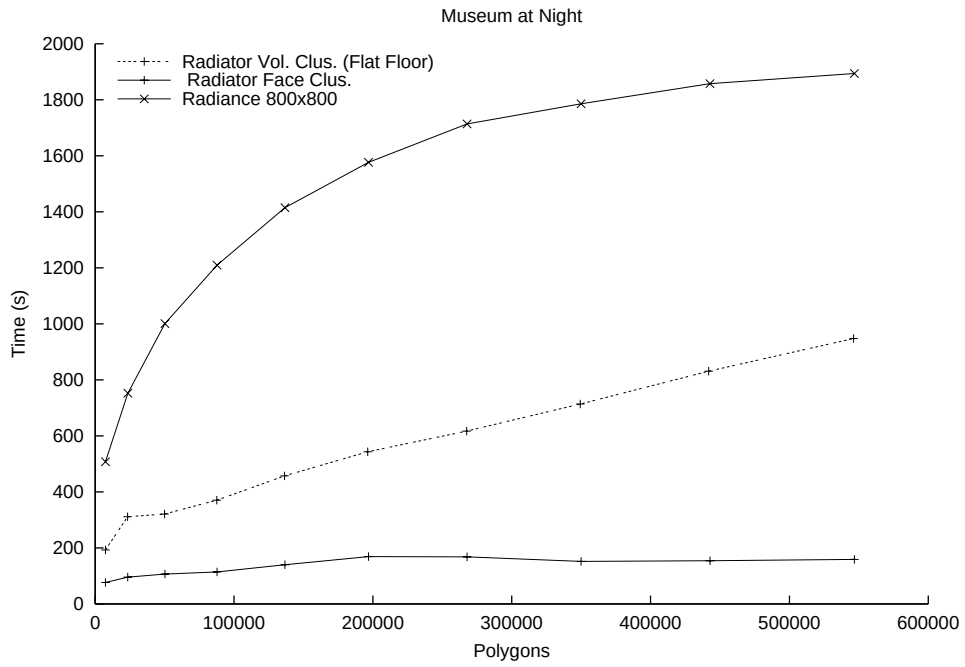


Figure 95: Rendering times for Radiance, volume clustering, and face clustering. While Radiance is initially expensive, its sub-linear performance means it can handle large scenes as well as smaller ones. Volume clustering is rendering a simpler scene than the other two methods, because of its quadratic behaviour with a non-flat floor. Its performance is linear, and thus eventually it will be more expensive than radiance. Face clustering, on the other hand, has close to the ideal flat performance after 200,000 polygons, and is much faster than the other two methods. It produces better results than volume clustering, and comparable results to Radiance.

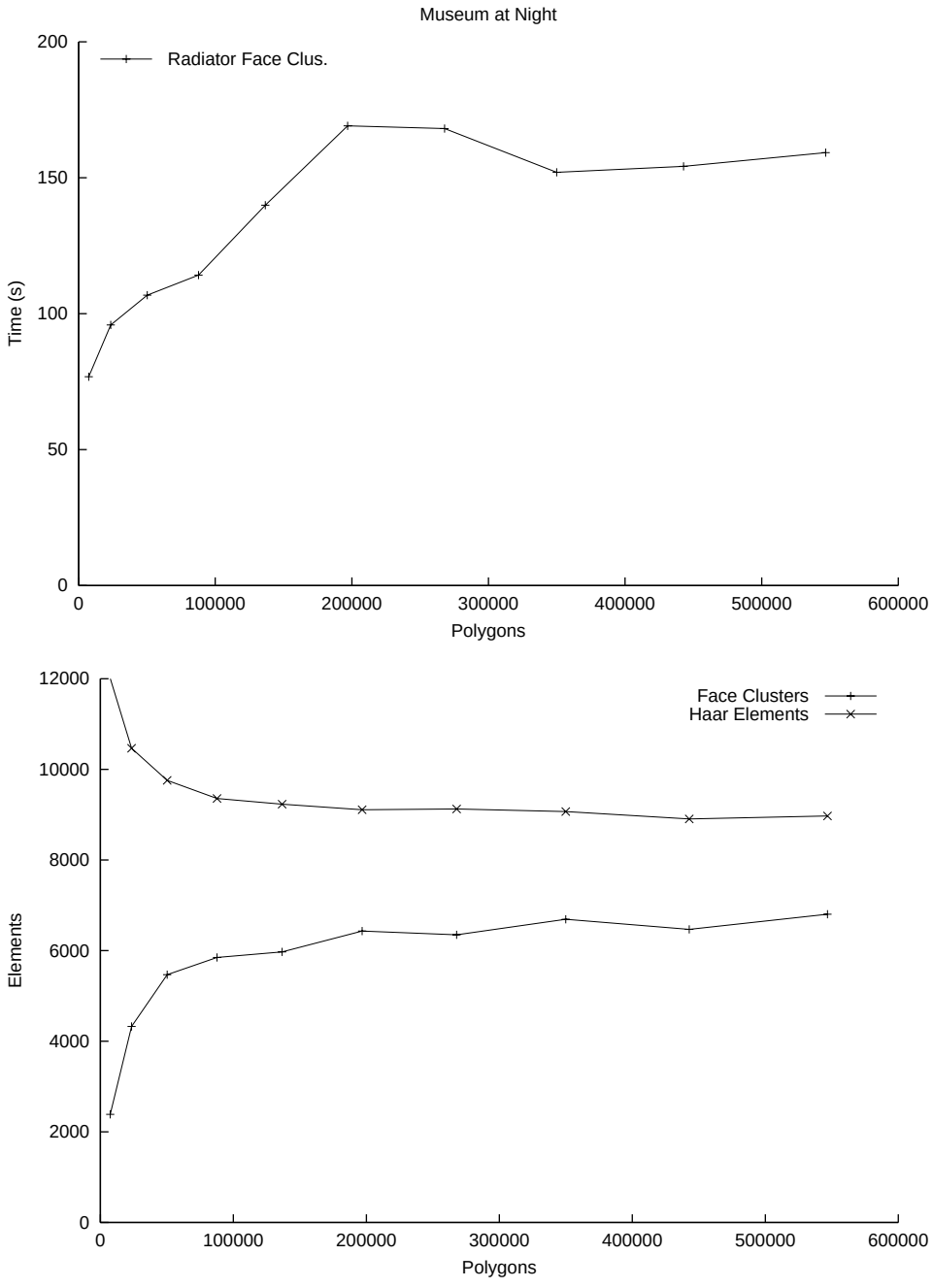


Figure 96: Rendering time for face cluster radiosity. Top: The rendering time increases until around 200,000 polygons, and then levels out. Bottom: initially Haar elements predominate in the final solution. These are gradually replaced by face clusters until a steady state is reached around 200,000 polygons.

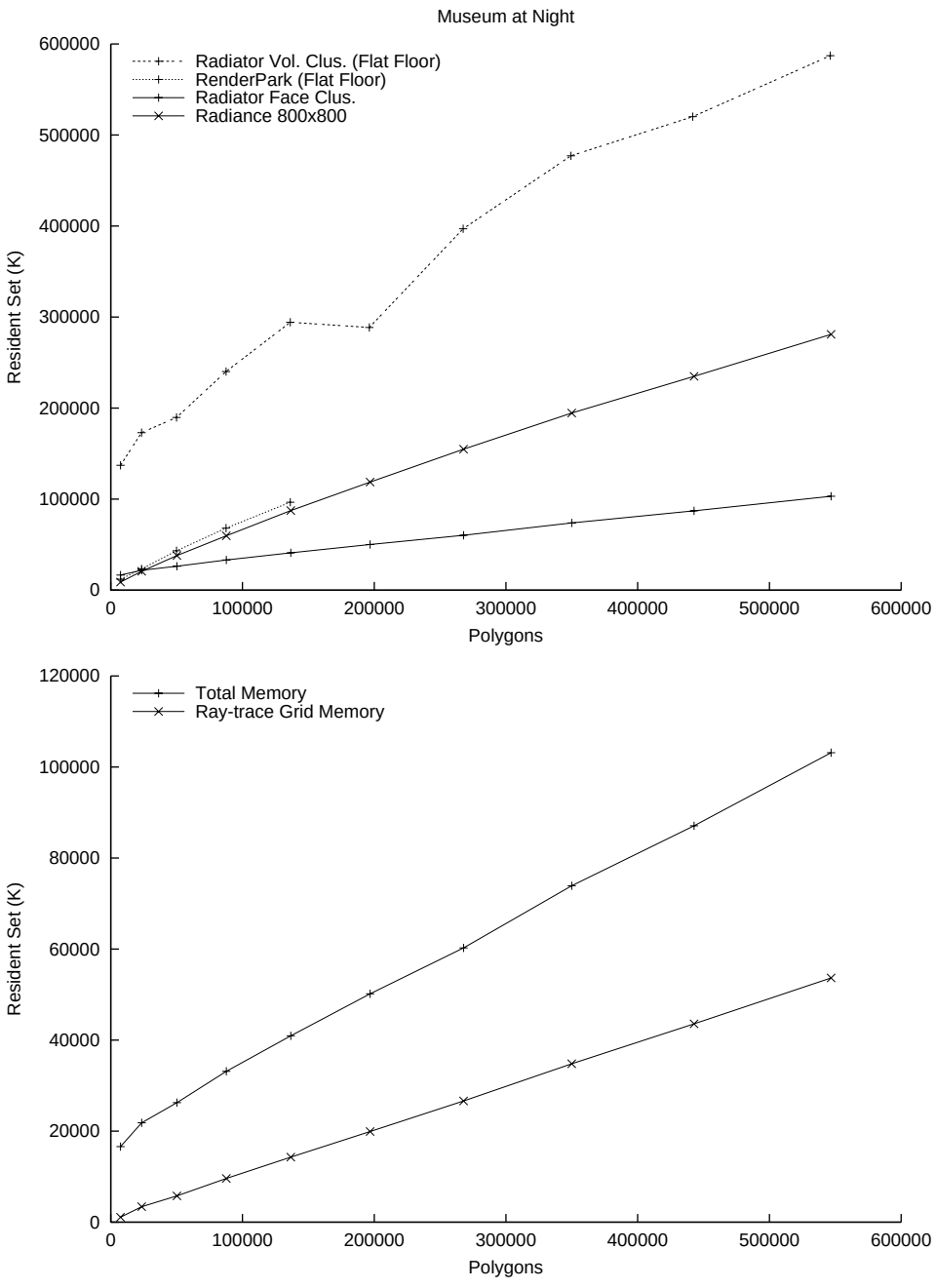


Figure 97: Memory results. All methods demonstrate sub-linear memory complexity. The volume clustering methods and radiance have very similar memory consumption. Bottom: Much of the memory use in face cluster radiosity is attributable to the data-structures necessary for occlusion testing via ray-tracing

7.4. A Highly Complex Museum Scene

As a final stress test of the radiosity, I ran face cluster radiosity on the museum with the full resolution version of each model. The full scene contains over 3.3 million polygons. The resulting illuminated mesh is shown in **Figure 98**, and some statistics for the corresponding radiosity run given in **Table 12**.

For this scene, the approximate visibility strategy outlined in **Section 6.3.3** was used, primarily as a memory-saving device. Without its use, the ray-tracing data structures alone took up some 330 MB of memory. By using the face cluster hierarchy to provide approximate visibility, this was reduced to under 40 MB[1].

In some senses this is an unrealistic scene; the models it contains are far more detailed than needed for most viewpoints, except for extreme close-ups. However, it does illustrate the freedom face cluster radiosity gives us with respect to model size. No other finite-element radiosity method is capable of rendering this scene without requiring an unrealistically large amount of CPU time and especially memory. Previously, only parallelised radiosity methods have been able to deal with scenes this large.

The extra detail in this scene can be important. **Figure 99** shows two close-ups of the Buddha statue in the scene—one from the complexity scene discussed in the last section, and one from the full-size scene. The full-size statue clearly shows extra detail over the simplified one, which looks like a smoothed version of the original. In some cases, and for some camera angles, this detail can be important.

Figure 100 shows a close-up of the Dragon model in the full-size scene. This picture is useful because it illustrates some of the limitations in the method; the model is small enough that its lighting is relatively coarse compared to the larger models. In particular, there are obvious discontinuities at shadow boundaries, most particularly on the hump of the dragon. The worst artifact is the outer claw on the hind leg, where the cluster has been incorrectly treated as entirely in shadow. In general, incorrect shadow resolution is most often responsible for any artifacts in output from the face cluster radiosity algorithm.

These artifacts can be eliminated by lowering the refinement epsilon to a suitable level, at the expense of increased rendering time. However, this will also increase the accuracy of all other parts of the solution scene, even if this accuracy is unnecessary—this is a limitation of having one global epsilon value controlling the accuracy throughout the scene. For a particular animation containing both longshots and extreme close-ups, using the camera path to set epsilon differently

1. The complexity level used was 0.1.

Figure 98: The highly complex museum scene. This scene contains 3.3 million polygons. The solution took 216 seconds. A final pass with vector irradiance interpolation took a further 105 seconds. (See **Table 12**.)

After	Time	Polygons	Face Clusters	Transport links	Memory
Initial linking	0 s	142	92	1	32 MB
Iteration 1	120 s	11658	8478	74510	39.2 MB
Solution	216 s	11,670	8,666	120,345	41 MB
Mesh Colouring, including VRI	321 s	11,670	8,666	120,345	41 MB

Table 12: Statistics for the big scene. The memory use includes ray-tracing data structures (a constant 32MB) and radiosity data structures only, ignoring geometry overhead. There were 33 volume clusters in this scene.

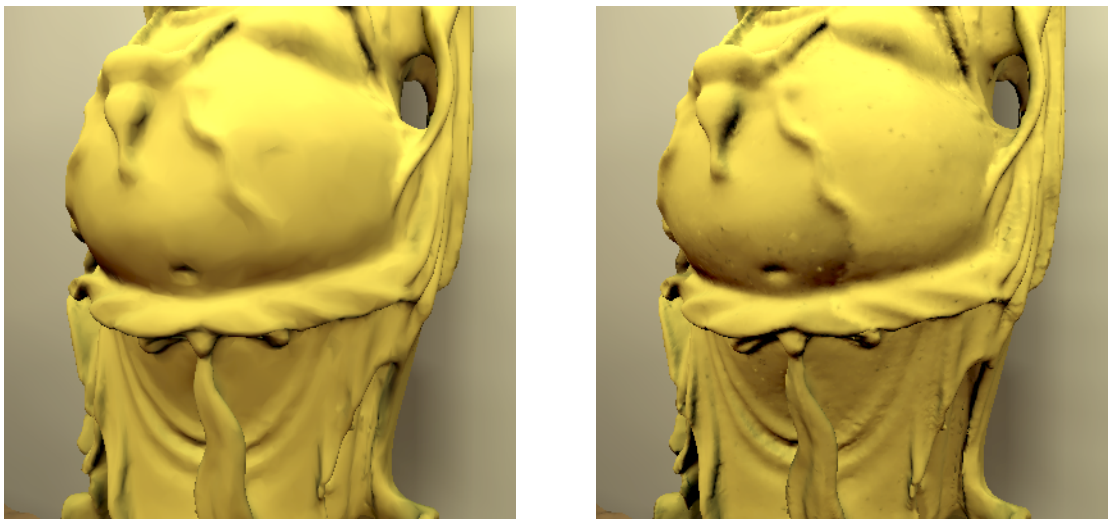


Figure 99: Close-ups of the Buddha for a 100,000 polygon model (left), and the full 1,000,000 polygon model (right).

in different parts of the scene, or the application of importance-based radiosity concepts [Smit92], may help reduce solution times significantly.

The results shown here use a final pass in which the irradiance vector to each cluster is resampled at four points, and then interpolated between those points (**Section 6.3.6**). While this process typically adds 50% to the total rendering time, it can be worthwhile where illumination is changing rapidly, especially on a

Figure 100: A close-up of the Dragon model from the museum scene.

smooth surface. **Figure 101** shows the effects of this process on the front of the Isis model, where it makes a noticeable improvement.

The polygon counts and face clustering times for each of these models are shown in **Table 13**. The preprocess of building face cluster hierarchies for each model in the Museum scene took a total of 436 seconds, and no model took more than 150MB of memory to process (and that only for the million-polygon model.) As pointed out previously, this cost can be heavily amortized; the same models can be reused over multiple scenes and radiosity simulations.

Finally, **Figure 102** illustrates the difference global illumination makes to this scene. The floor from the full museum scene radiosity solution is shown, together with another solution with all objects removed except for the table and the lights, for comparison purposes. The latter essentially shows the floor with direct illumination only. Notice how in the full solution the reflected light from the walls “fills in” the unlit areas of the floor, and under the table. Also, there is a yellowish area in the upper left of the full solution where light spills off the buddha statue onto the floor.

Figure 101: The effect of vector irradiance interpolation on the museum scene. Shown is a close-up of the Isis statue both with (left) and without (right) interpolation.

Model	Polygons	Clustering Time	Disk Use	Disk Use (Truncated)
Buddha	1,085,634	148s	158 MB	30 MB
Isis	375,736	47s	55 MB	11 MB
Dragon	869,928	109s	127 MB	24 MB
Venus	268,714	34s	39 MB	8.2 MB
Pedestal	24,000	5.7s	3.4 MB	744 kB
Candlestick x 2	4,150	1s	621 kB	132 kB
Floor	717,602	87s	104 MB	22 MB
Picture x 2	7,400	1.6s	1.1 MB	246 kB
Total	**3,357,310**	**436s**	**488.2 MB**	**96.7 MB**

Table 13: Face cluster generation for the full scene

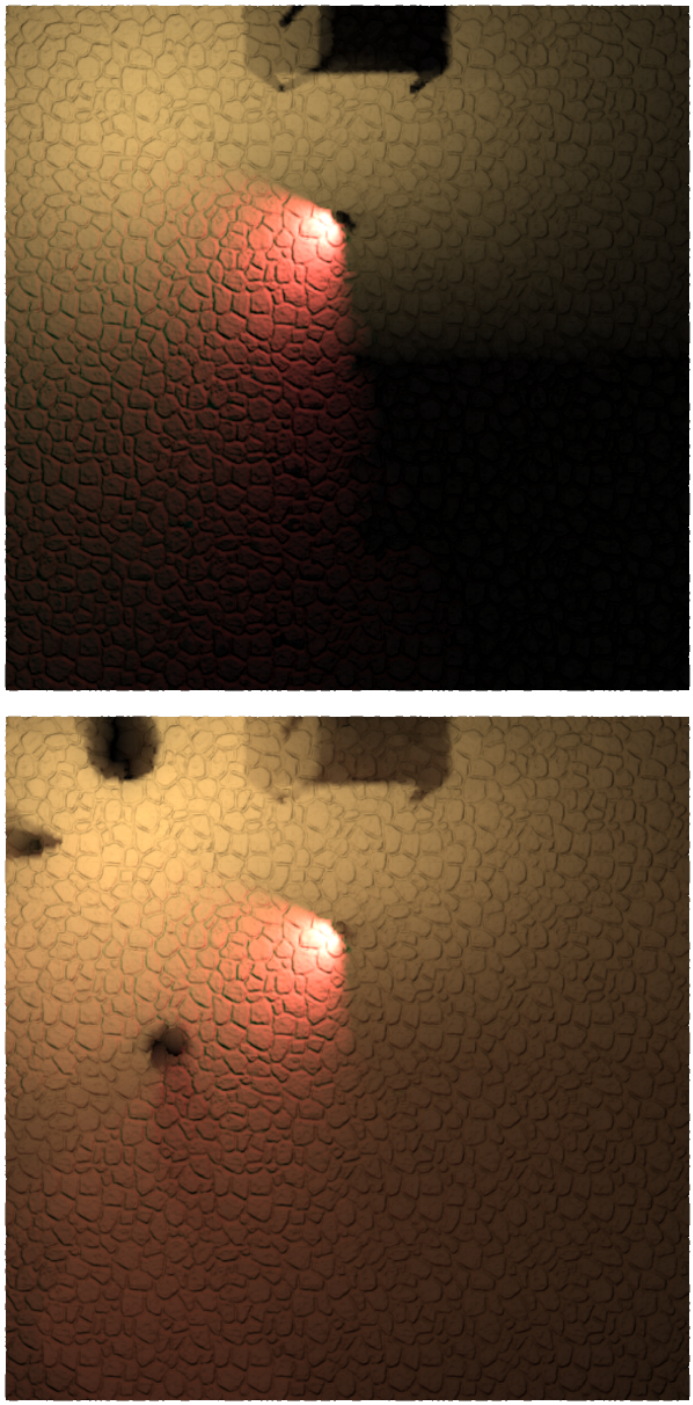


Figure 102: The effects of including diffuse global illumination. Shown are the stone floor from the museum scene with direct lighting only, top, and the complete radiosity solution for the floor, bottom.