Visibility Map for Global Illumination in Point Clouds

R. Goradia[1] A. Kanakanti[1] S. Chandran[1] A. Datta[2]

[1]Indian Institute of Technology, Bombay
Mumbai, India
{rhushabh, kanil, sharat}@cse.iitb.ac.in

[2]University of Western Australia
Perth, Australia
datta@csse.uwa.edu.au

2.12.2007

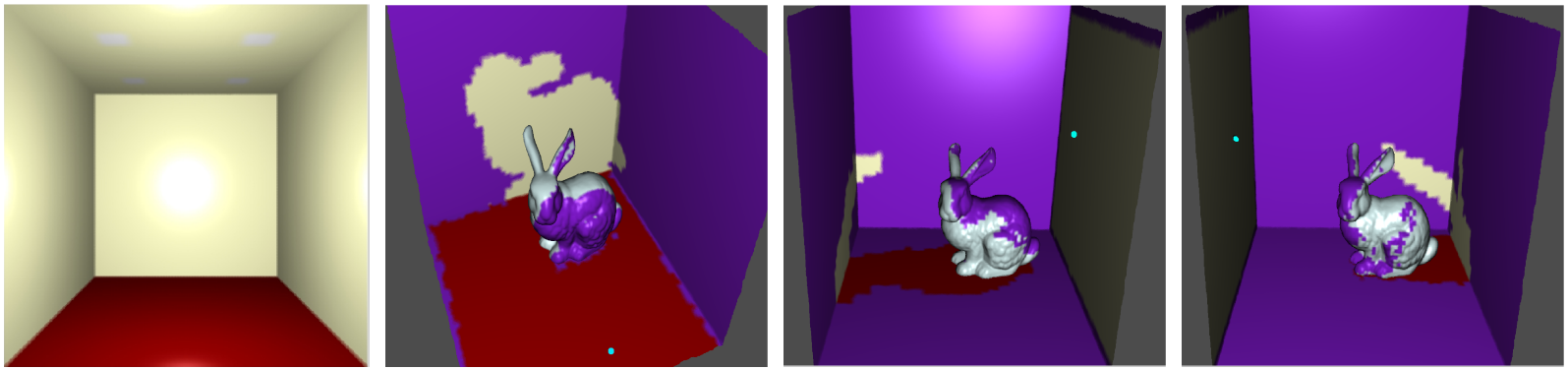
Overview

Overview

Problem Definition

Problem Statement

To compute a **Visibility Map** (*V-map*) for complex scenes represented as **point-models**, for the purpose of **global illumination**.

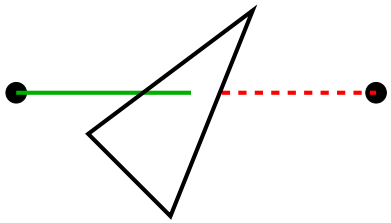


Application Domains

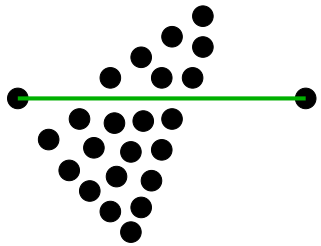
Visibility Between Point Pairs

View Independent Visibility calculation between point pairs is **essential** to give *correct* GI results as a point recieves energy from other point only if it is **visible**

Visibility Between Point Pairs



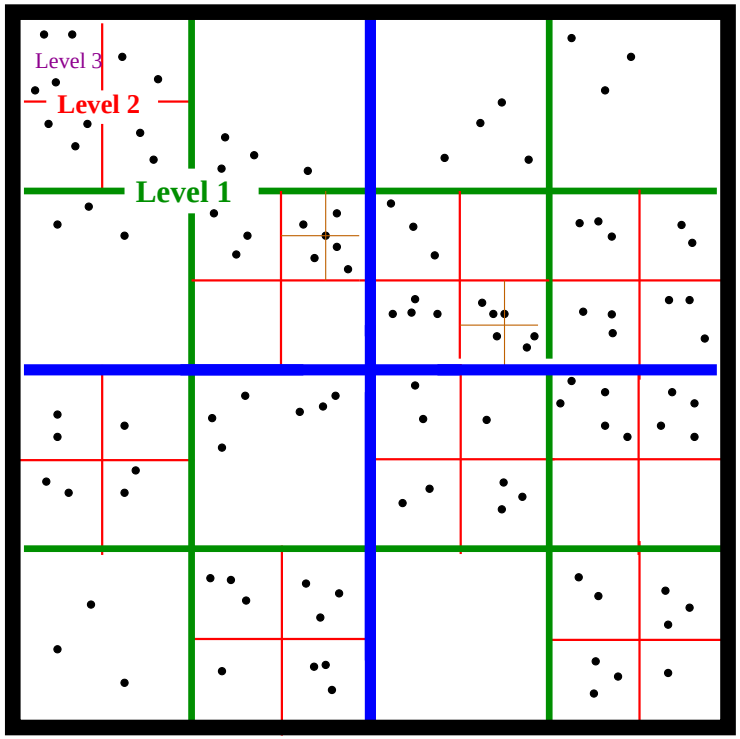
VISIBILITY IN POLYGONAL MODELS



VISIBILITY IN POINT MODELS

Hierarchical Visibility

Hierarchical Visibility approach helps in achieving faster GI solution (eg. hierarchical radiosity).



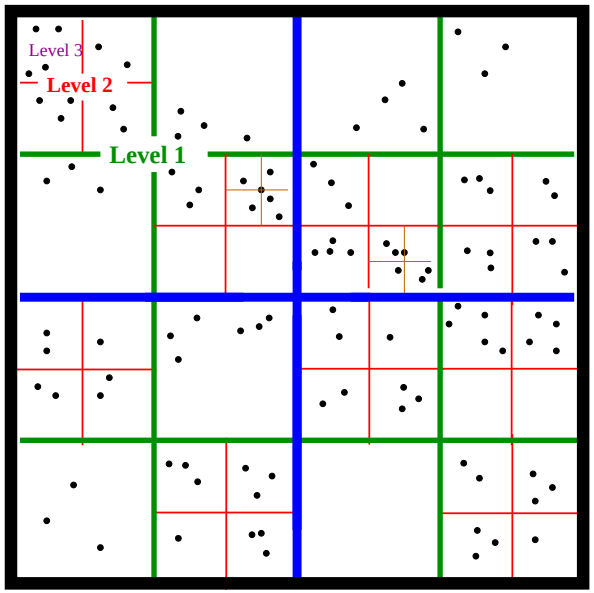
Model	Points (millions)	N^2 links (millions)	V-Map Links (millions)	% Decrease	Memory(MB) N^2 links	Memory(MB) V-Map links	Build V-Map Time(secs)
ECR	0.1	1.4	0.27	79.5%	5.35	1.09	20.6
PCR	0.14	3.85	0.67	82.62%	15.43	2.68	23.8
BUN	0.15	1.53	0.38	74.64%	6.09	1.5	21.7
DRA	0.55	2.75	0.43	84.54%	11.0	1.7	23.5
BUD	0.67	1.58	0.39	74.75%	6.33	1.6	23.9
GAN	0.15	1.56	0.38	75.64%	6.2	1.55	22.0
GOD	0.17	1.62	0.4	75.31%	6.4	1.63	22.9

- ECR – Empty Cornell room
- PCR – Packed Cornell room
- BUN – Bunny in Cornell room
- DRA – Dragon in Cornell room
- BUD – Buddha in Cornell room
- GAN – Indian God Ganesha in a Cornell room
- GOD – Indian Goddess Satya in a Cornell room

Overview

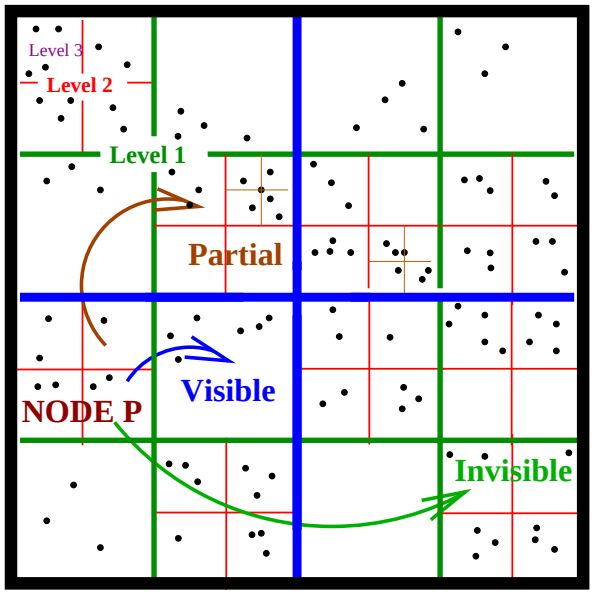
What is a Visibility Map (V-map)?

- The *visibility map* for a tree is a collection of visibility links for every node in the tree
- The *visibility link* for any node p is a list L of nodes
- Every point in any node in L is guaranteed to be visible from every point in p

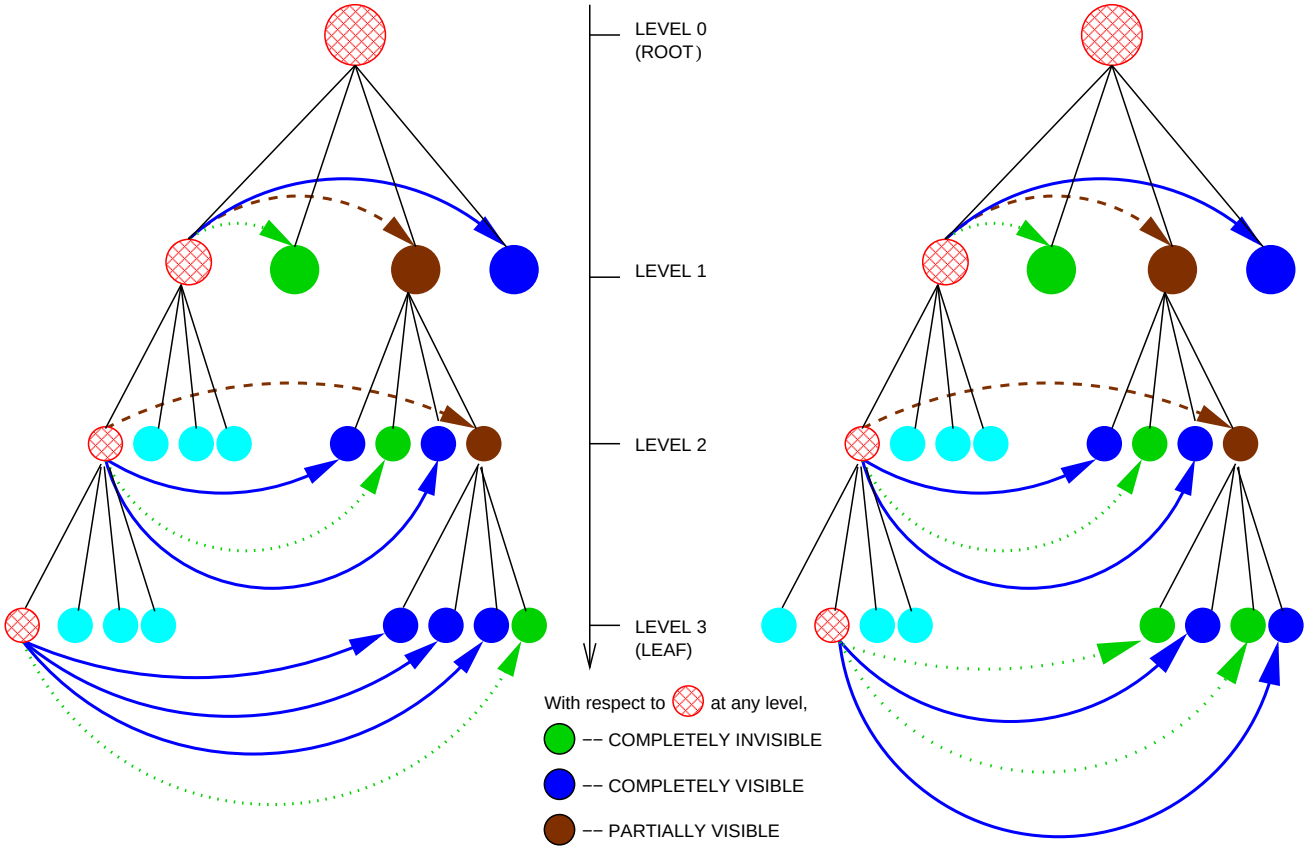


What is a Visibility Map (V-map)?

- The *visibility map* for a tree is a collection of visibility links for every node in the tree
- The *visibility link* for any node p is a list L of nodes
- Every point in any node in L is guaranteed to be visible from every point in p



What is a Visibility Map (V-Map)?



Visibility Map Queries?

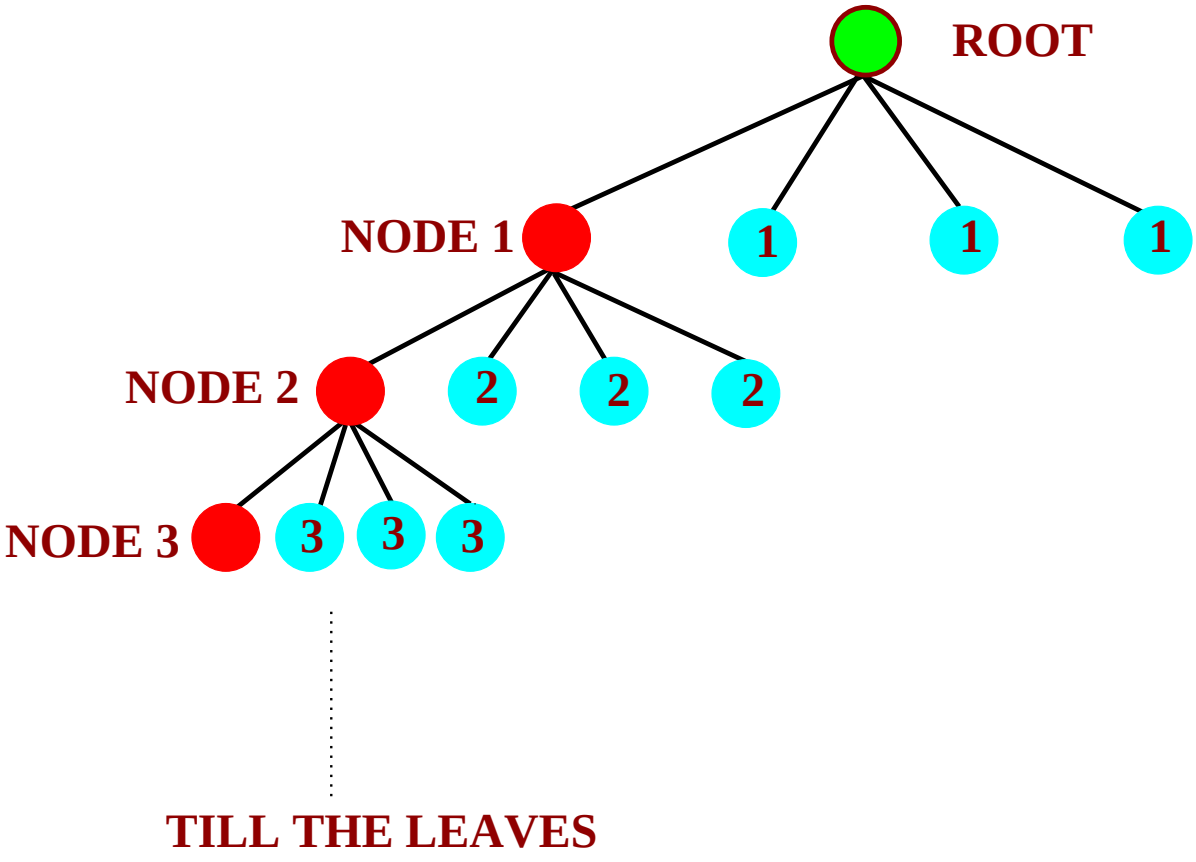
Visibility maps entertain efficient answers to the following queries.

1. Is point x visible to point y?
2. What is the visibility status of u points around x with respect to v points around y?
   - Repeat a "primitive" point-point visibility query uv times
   - V-Map gives the answer with $O(1)$ point-point visibility queries.
3. Given a point x and a ray R, determine the first object of intersection.
4. Is point x in the shadow (umbra) of a light source?

All the above queries are done with a simple traversal of the octree.

V-map Construction Algorithm

- Initialize the o-IL of every node to be its seven siblings

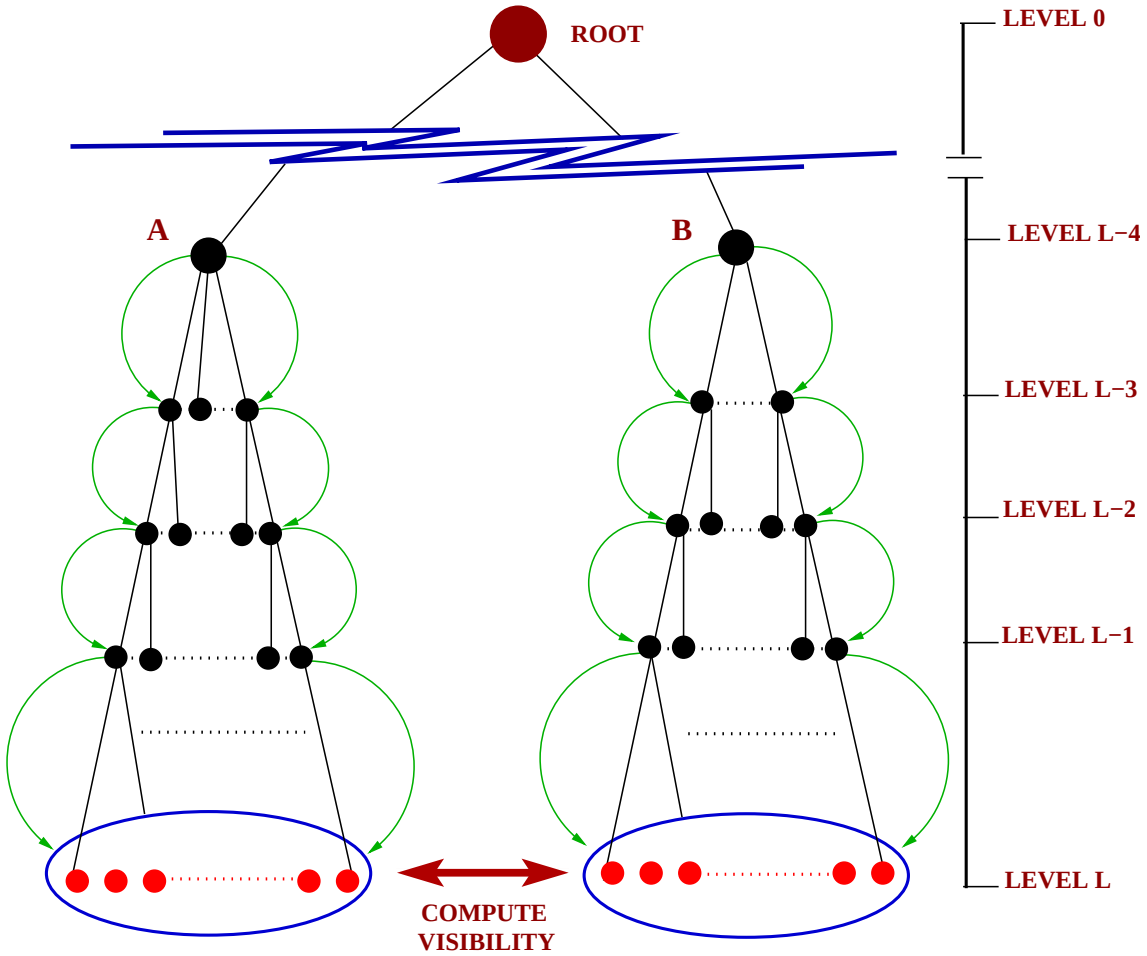


V-map Construction Algorithm

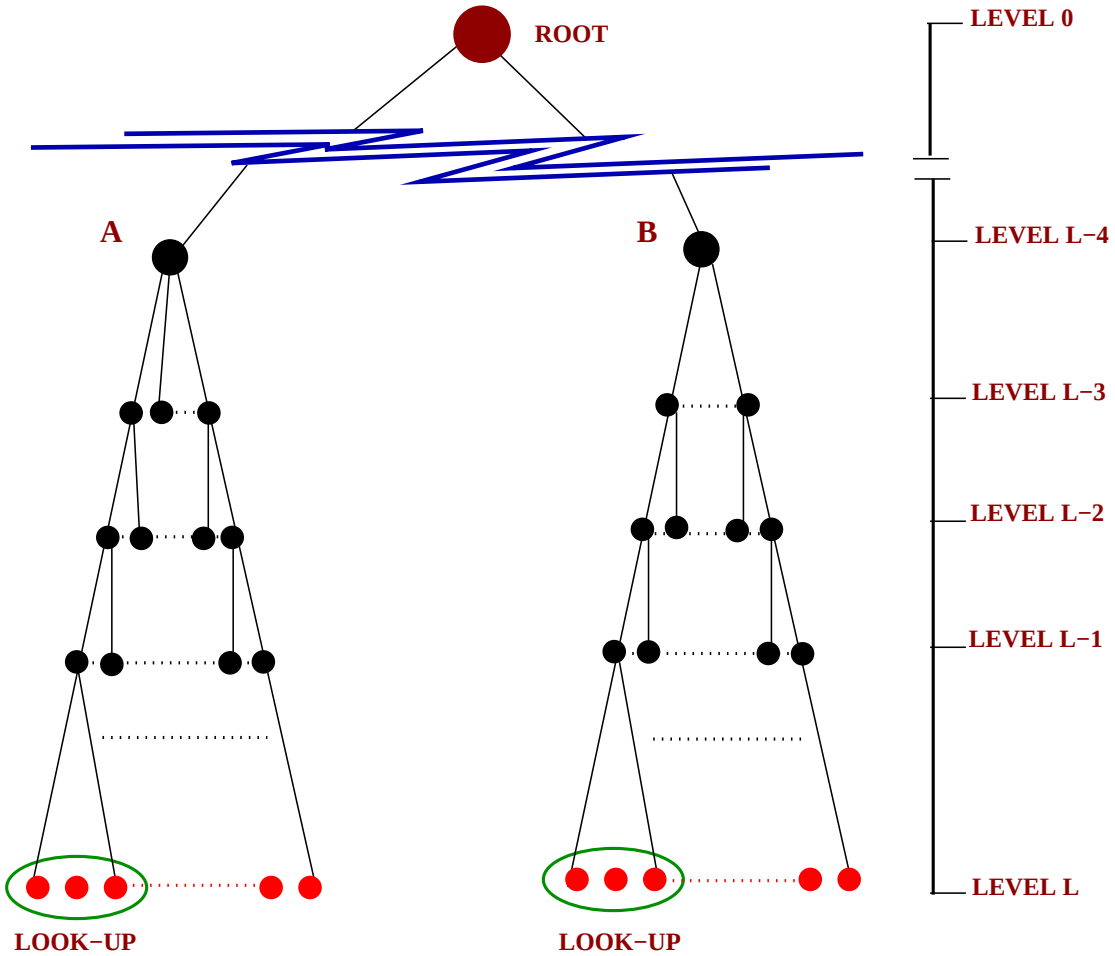
```
procedure OctreeVisibility(Node A)
for each node B in old interaction list (o-IL) of A do
   if NodetoNodeVisibility(A,B) == VISIBLE then
      add B in new interaction list (n-IL) of A
      add A in new interaction list (n-IL) of B
   end if
   remove A from old interaction list (o-IL) of B
end for
for each C in children(A) do
   OctreeVisibility(C)
end for
```

- V-map constructed by calling initially for the root, which sets up the relevant visibility links in n-IL
- NodetoNodeVisibility(A,B) constructs the visibility links for all descendants of A w.r.t all descendants of B (and vice-versa) at the best (i.e. highest) possible level. This ensures an optimal structure for hierarchical radiosity as well as reduces redundant computations

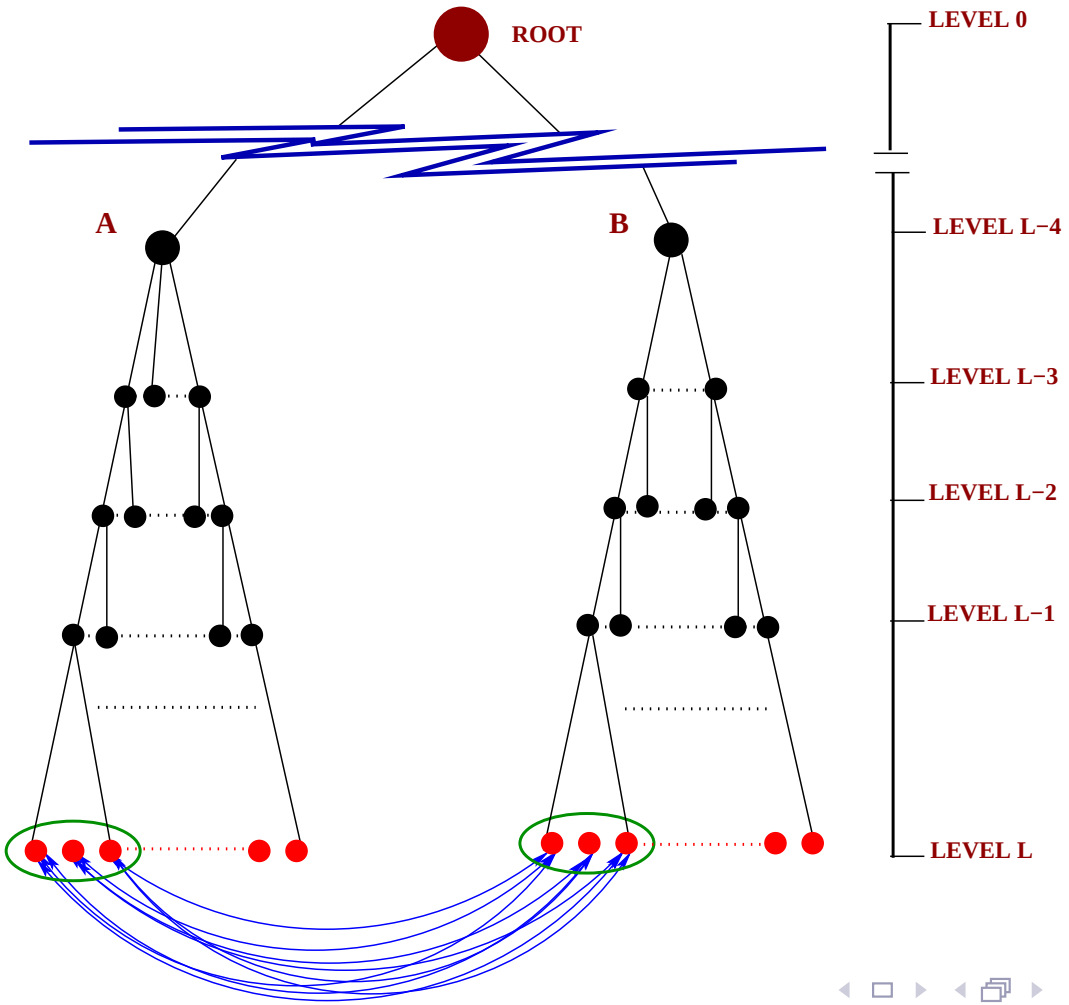
V-map Construction Algorithm



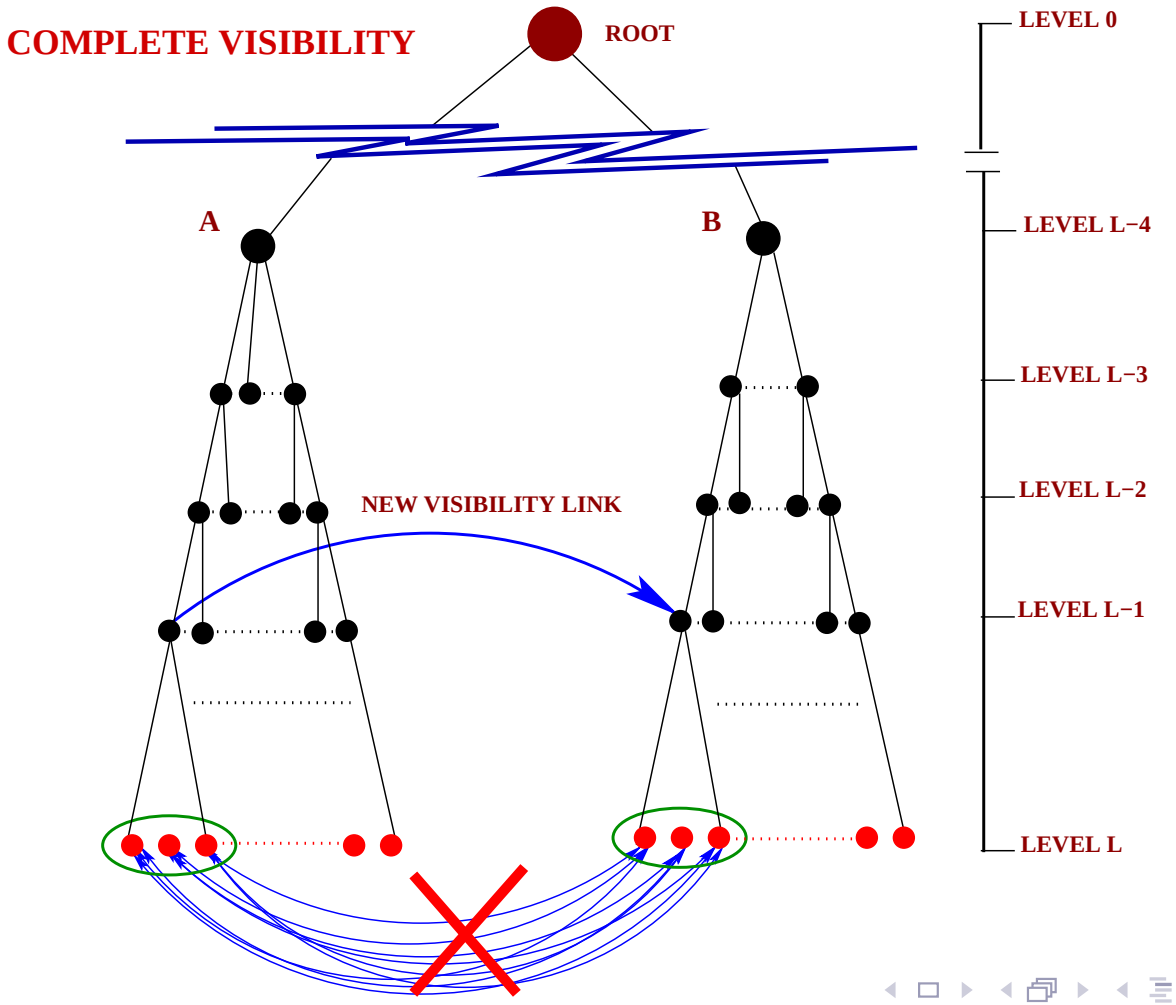
V-map Construction Algorithm



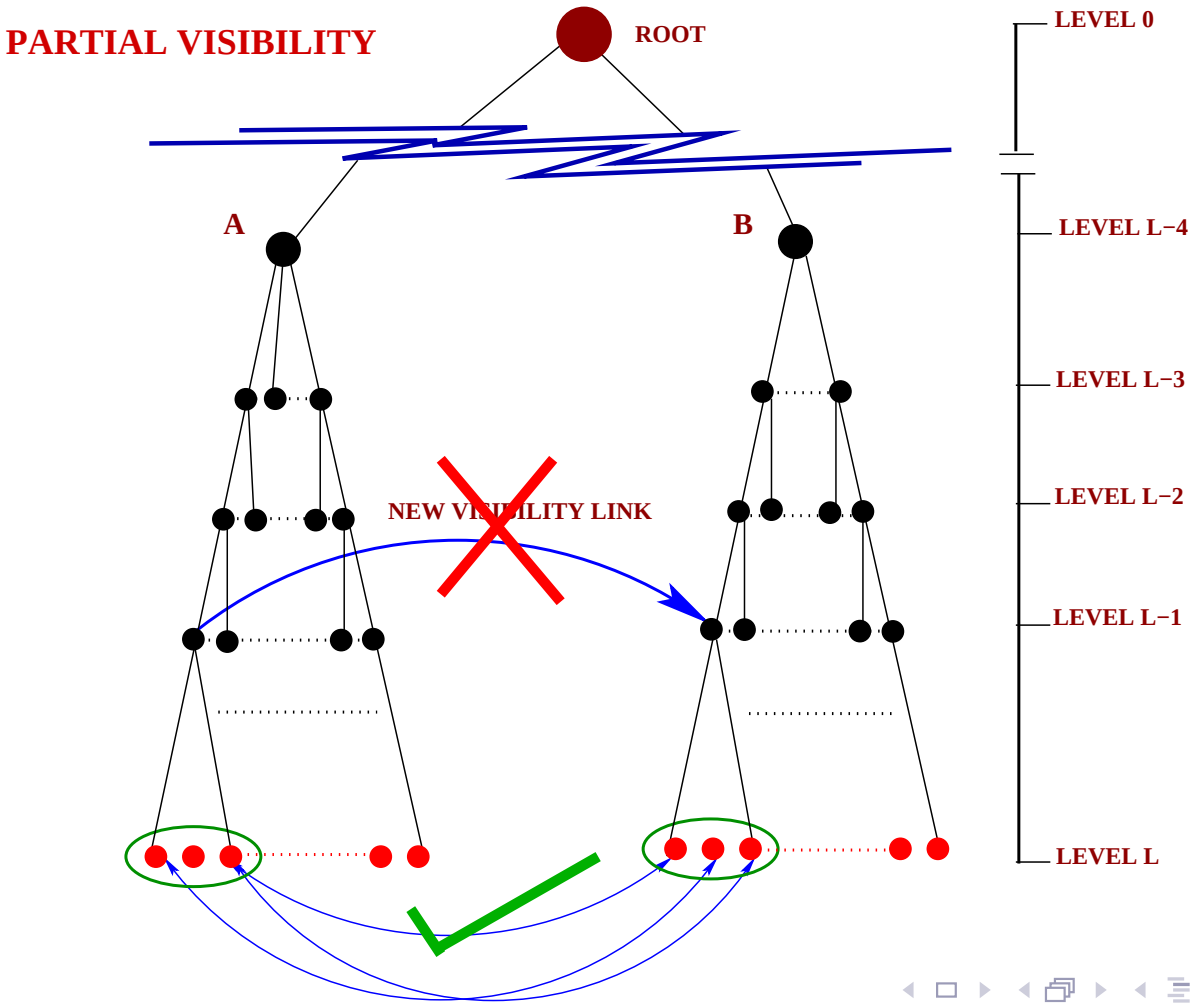
V-map Construction Algorithm



V-map Construction Algorithm

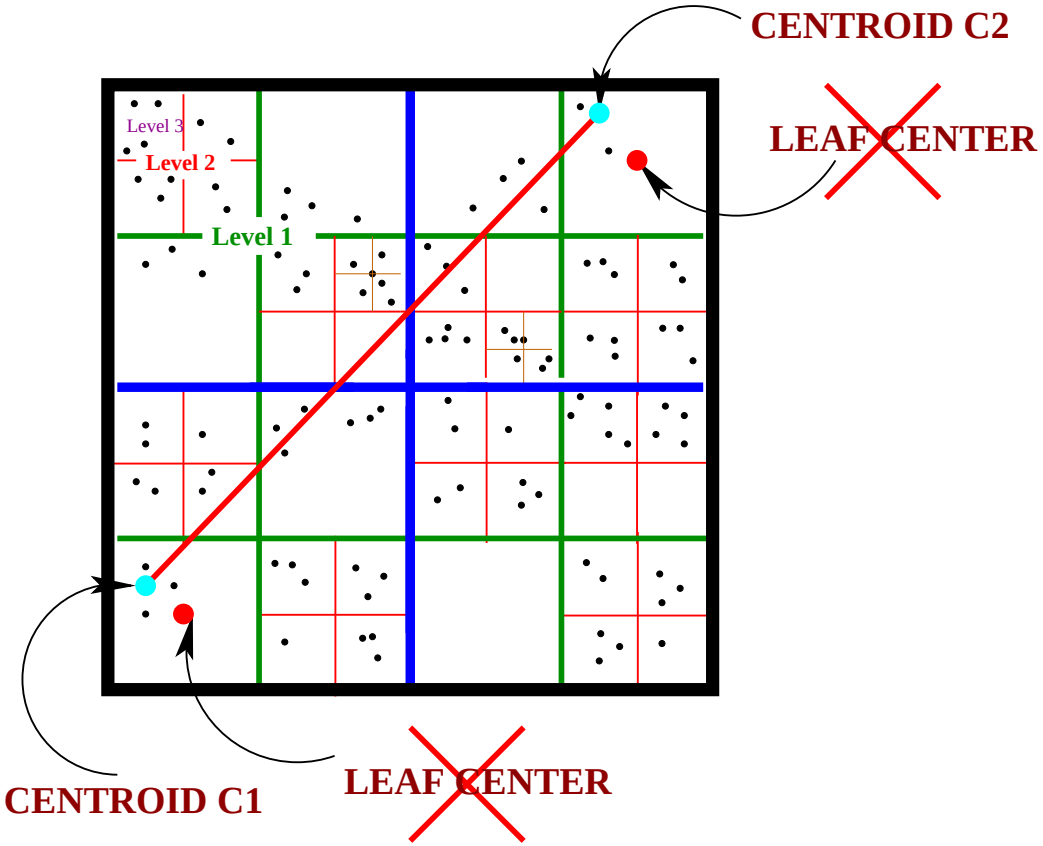


V-map Construction Algorithm

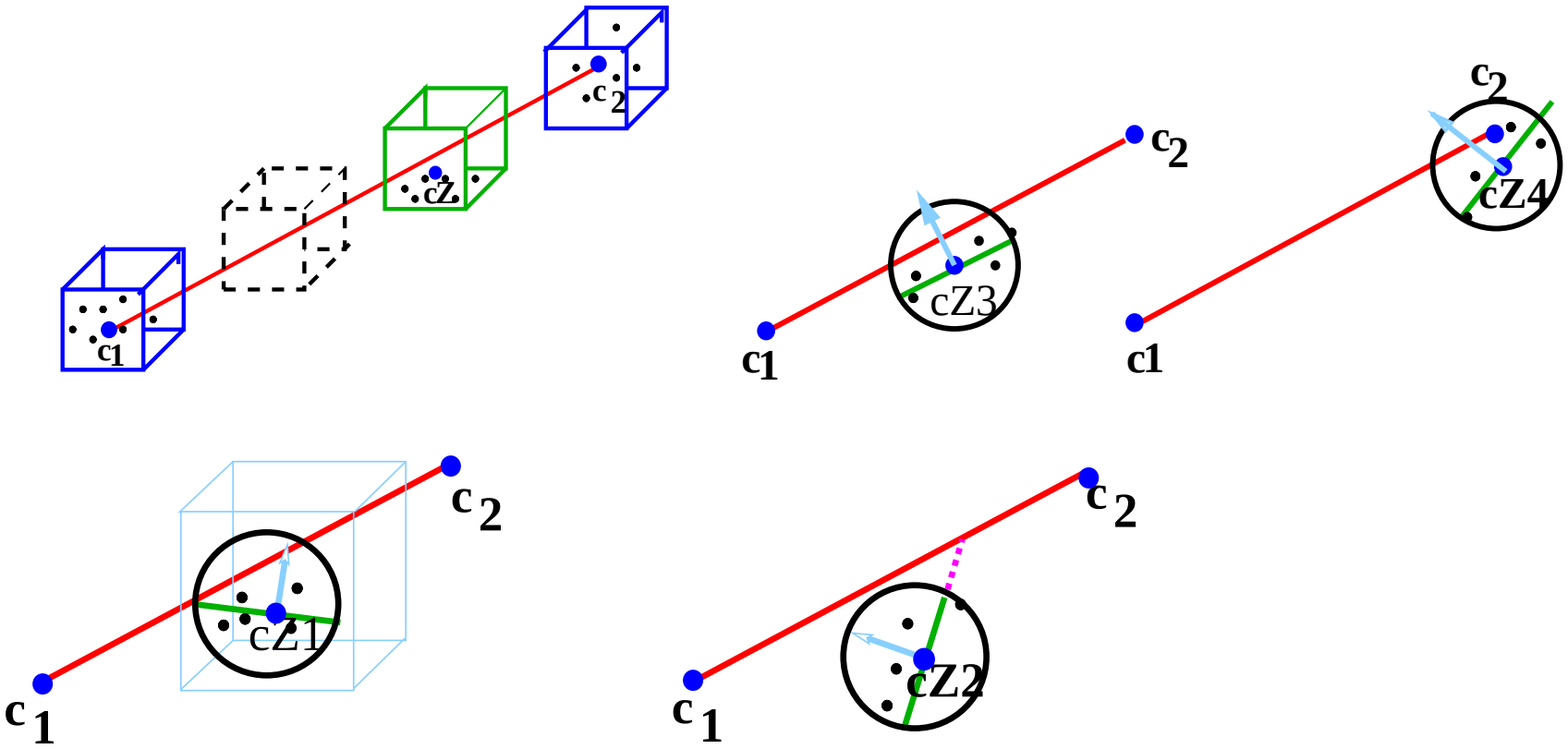


Leaf-Leaf Visibility Algorithm

- Consider centroid and **NOT** leaf center



Leaf-Leaf Visibility Algorithm

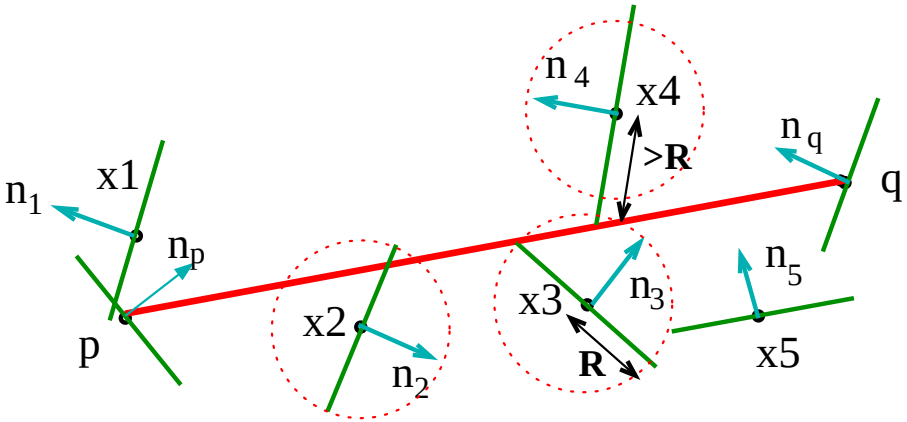


Leaf-Leaf Visibility Algorithm

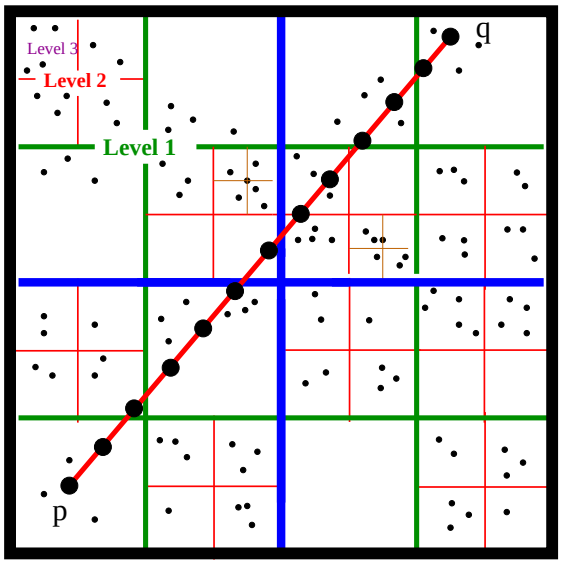
- Distance **R** is unique for each leaf and depends on distribution of points in the leaf (**R** is not a user-input)
- Imposing a **strict** visibility condition balances the leniency introduced
- **Faster**, as we exit on finding the first potential occluder
- Dense point models help in achieving better results

NOTE: We perform this visibility computation (with help of **averaged normals**) only for the leaves. There are no **average normals** defined for internal nodes of the tree.

Point Pair Visibility



Visibility between points p and q



Finding Potential Occluders

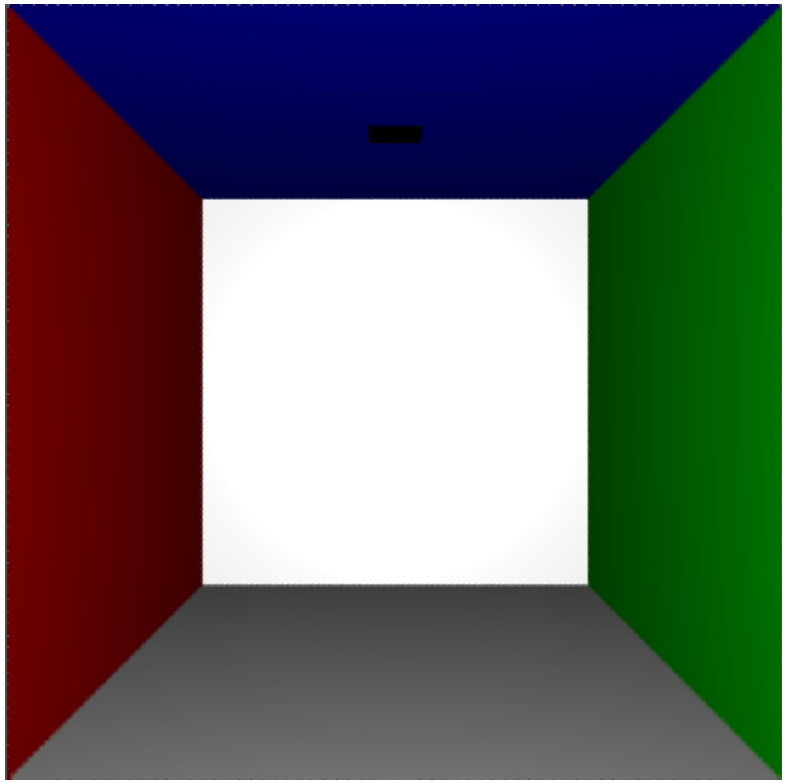
using bresenham line algorithm

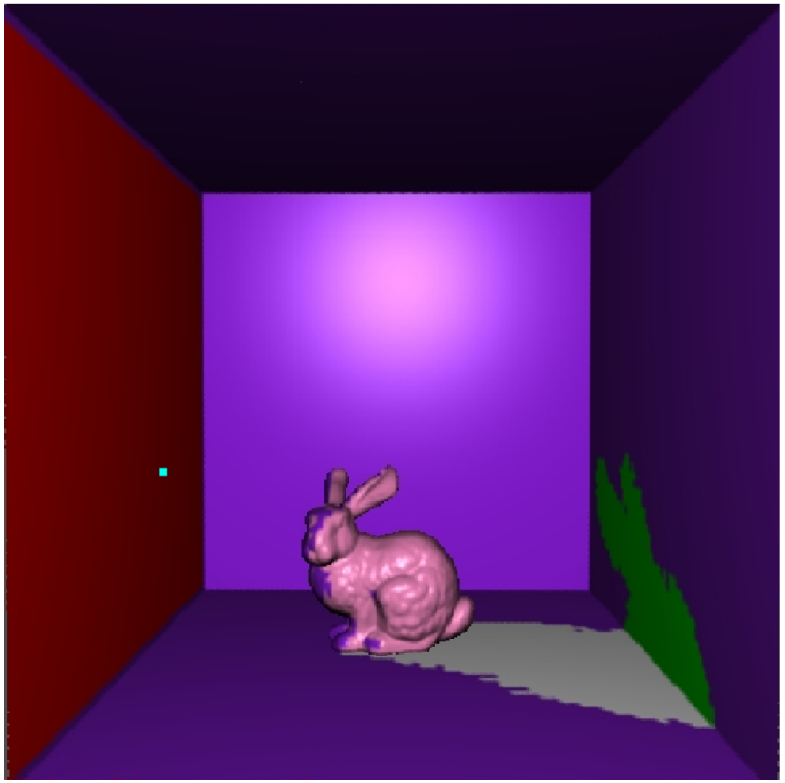
Computational Complexity

- Assume $N = \Theta(n^2)$, n = points in input model.
- Visibility problem provides answer to N pairwise queries. Hence we measure the efficiency w.r.t N
- Octree Visibility has the recurrence: $T(h) = 8T(h-1) + N$ (for a Node A at height h)
- Complexity for *NodetoNodeVisibility(A,B)* is determined by the calls to point-pair visibility algorithm
- Assuming the latter to be $O(1)$, the recurrence relation for the former is $T(h) = 64T(h-1) + O(1)$.
- The overall algorithm consumes a small amount of memory (for storing M) during runtime.

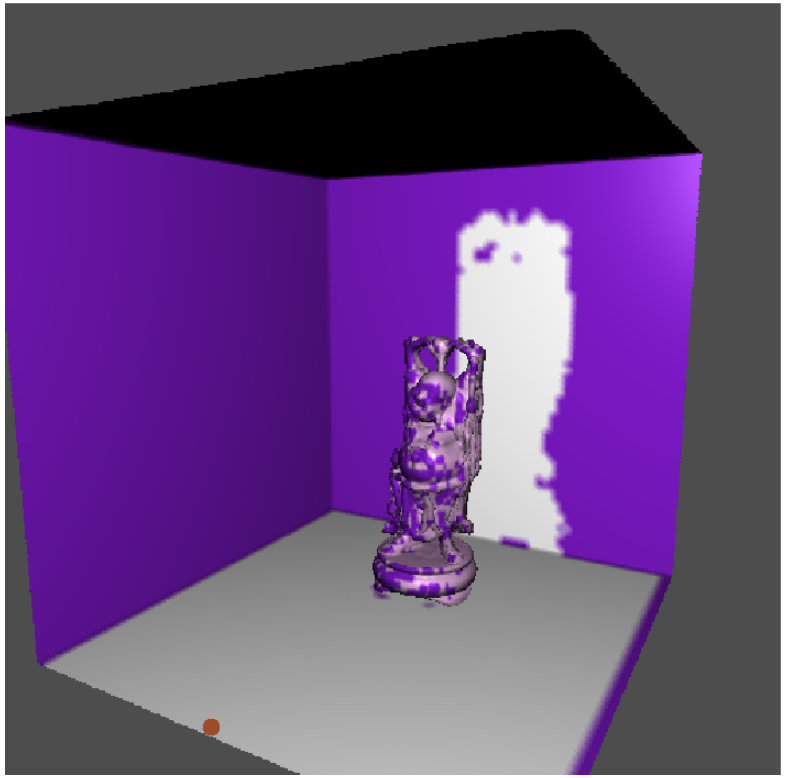
Overview

Qualitative Results: Visibility Correctness

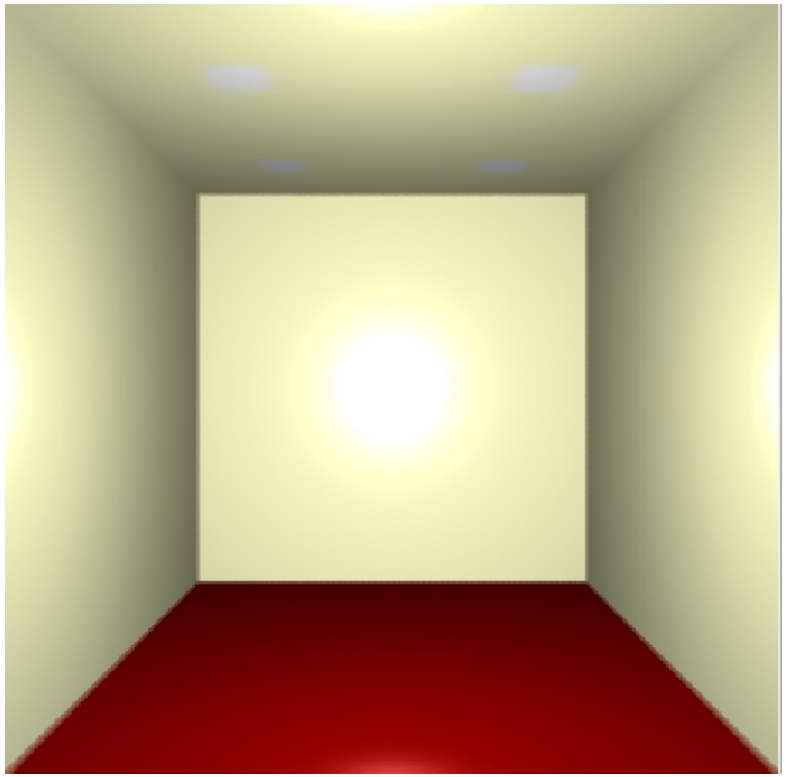


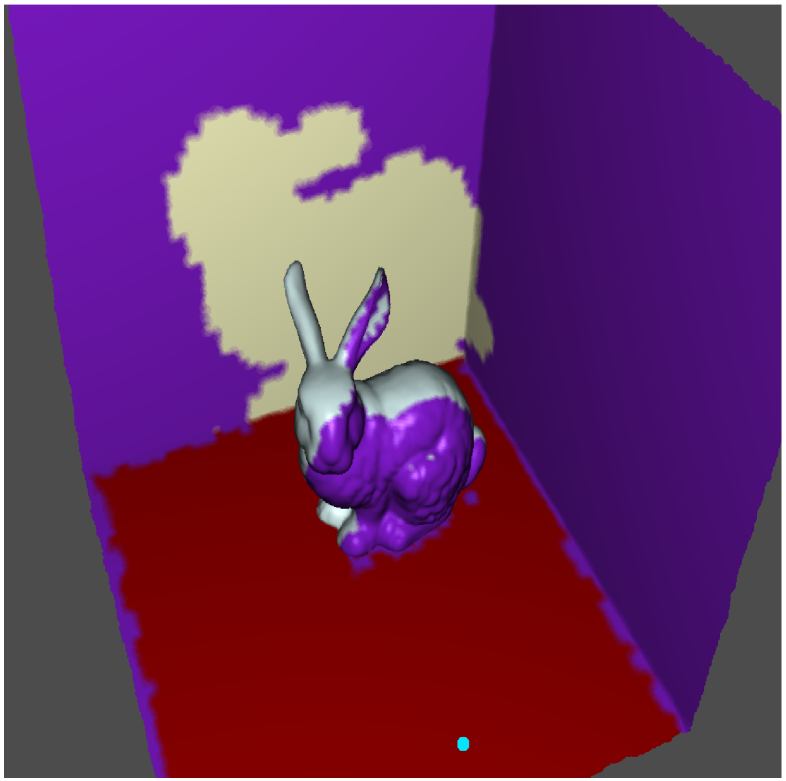


Qualitative Results: Visibility Correctness

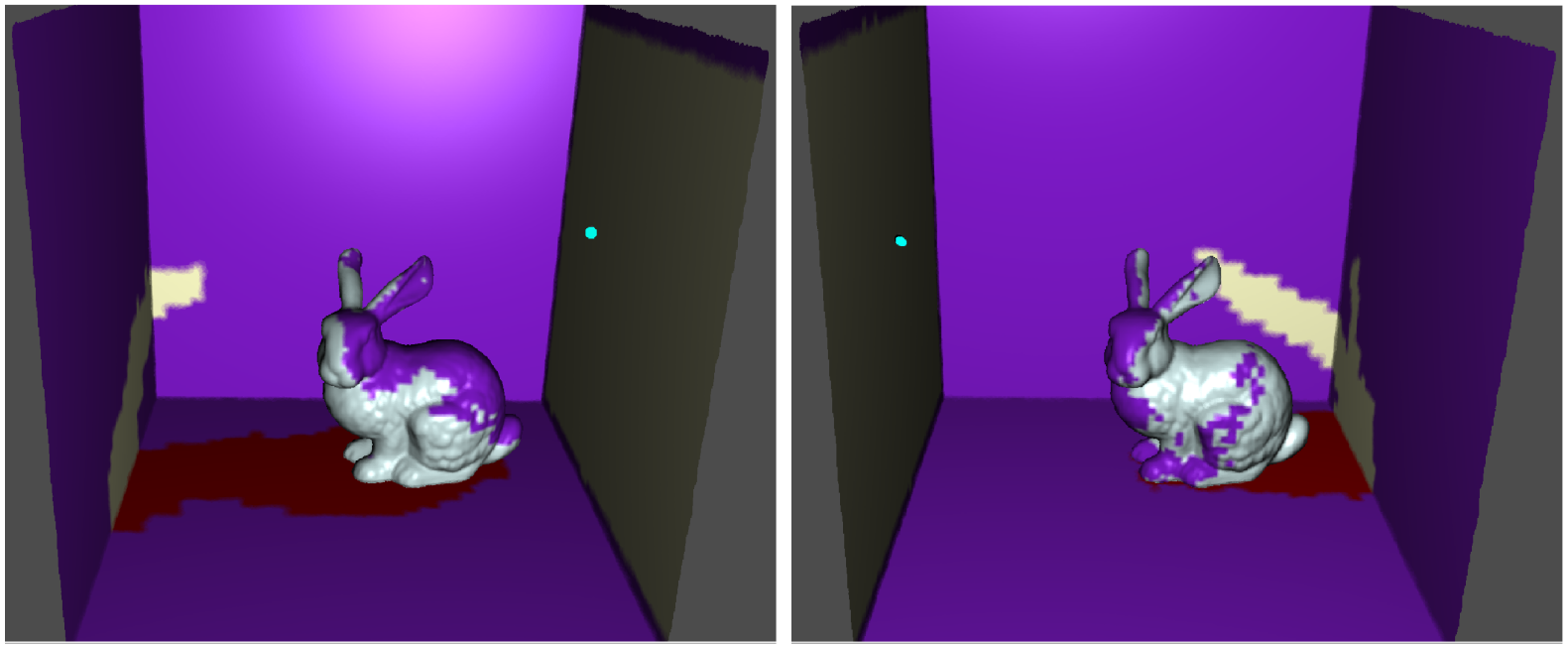


Qualitative Results: Visibility Correctness

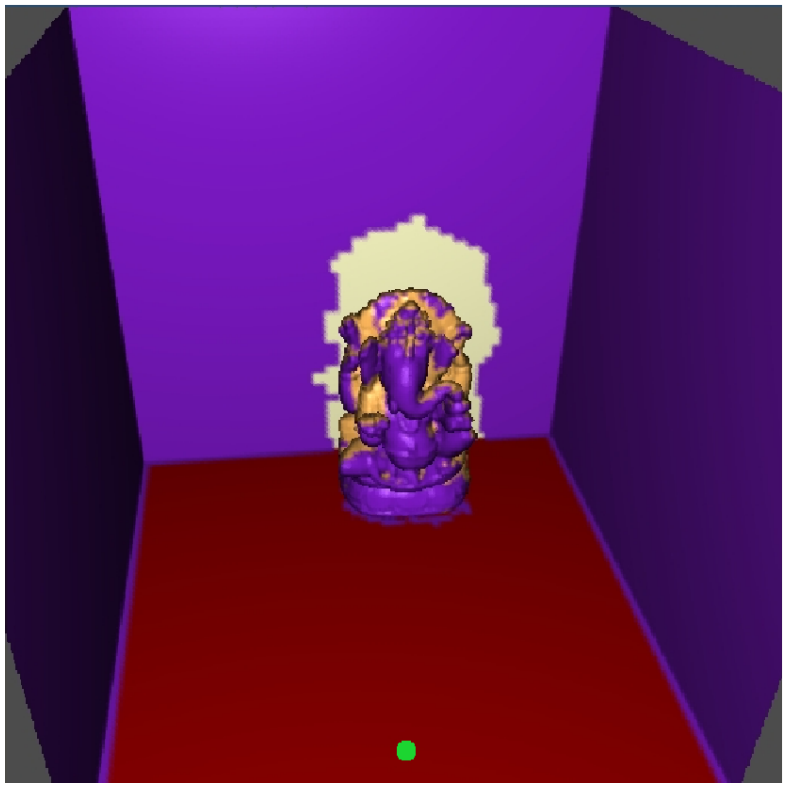


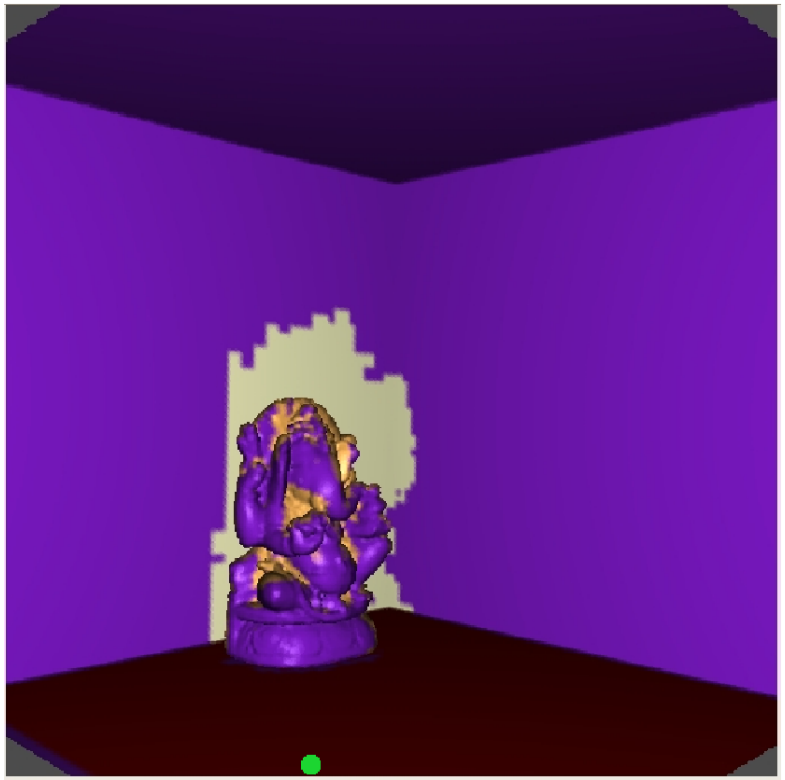


Qualitative Results: Visibility Correctness

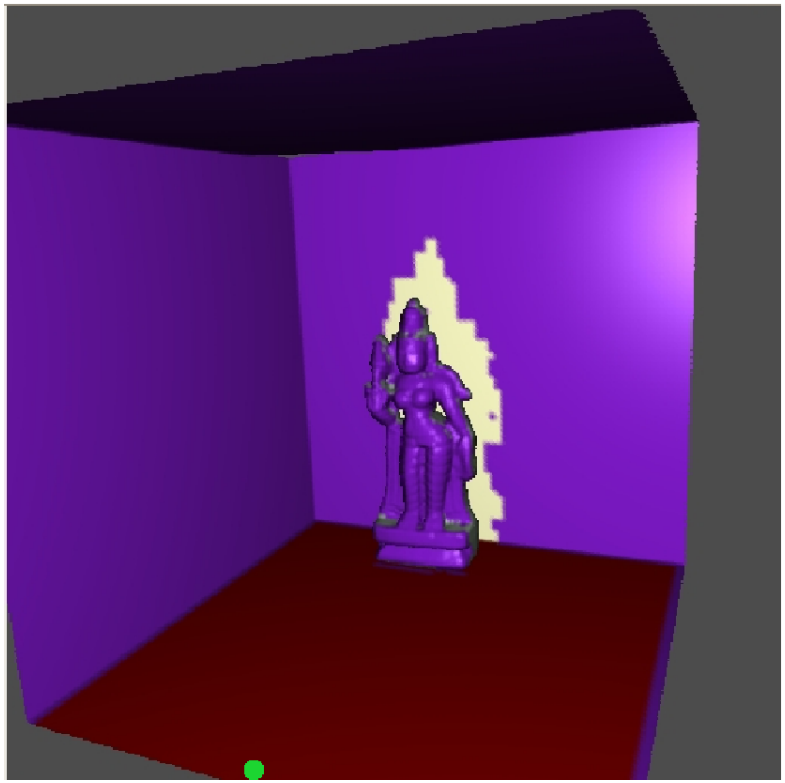


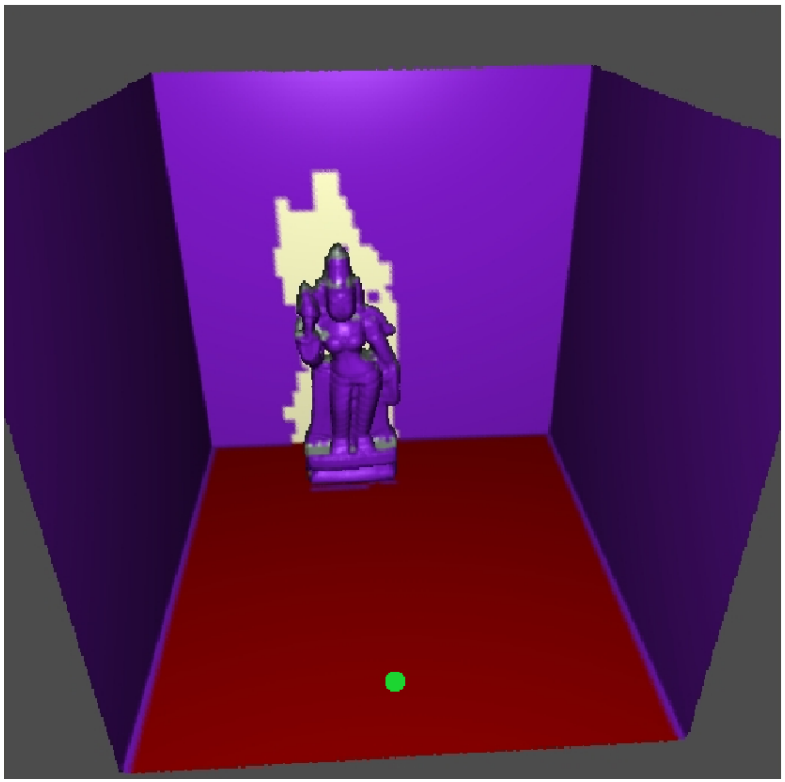
Qualitative Results: Visibility Correctness



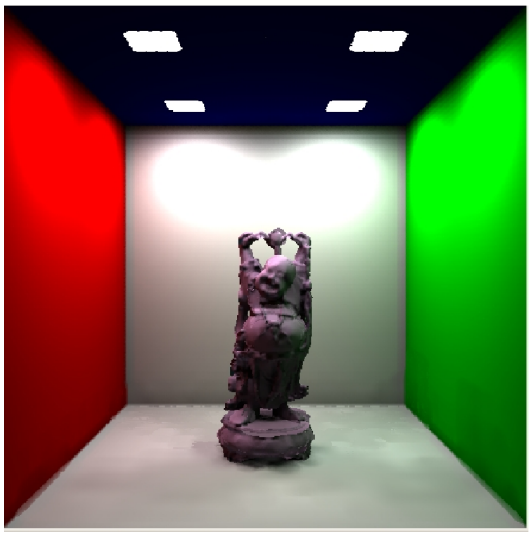


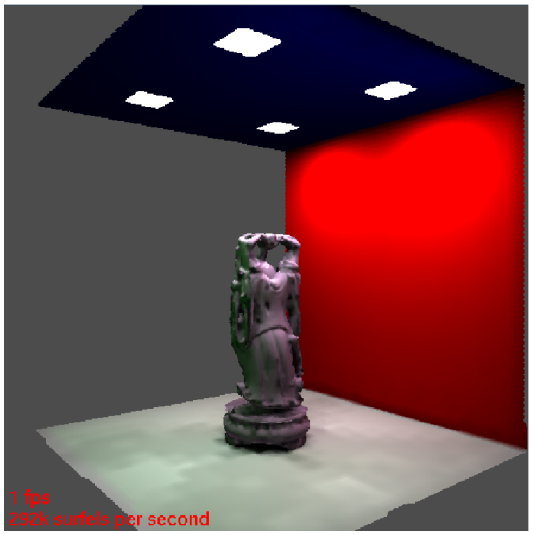
Qualitative Results: Visibility Correctness

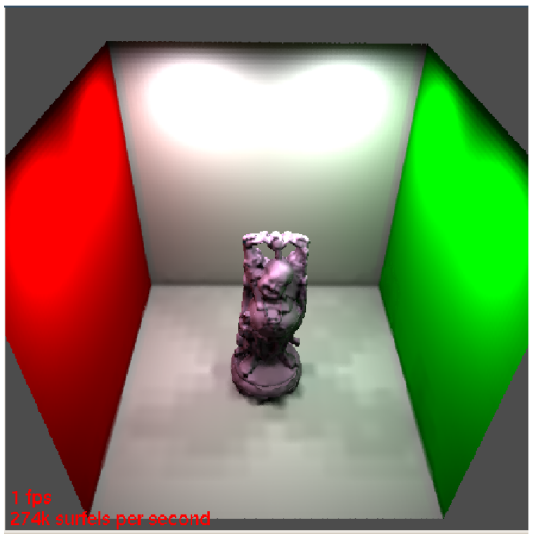




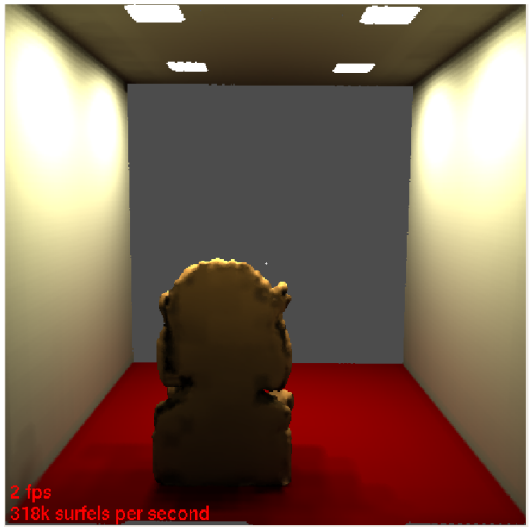
Qualitative Results: Global Illumination



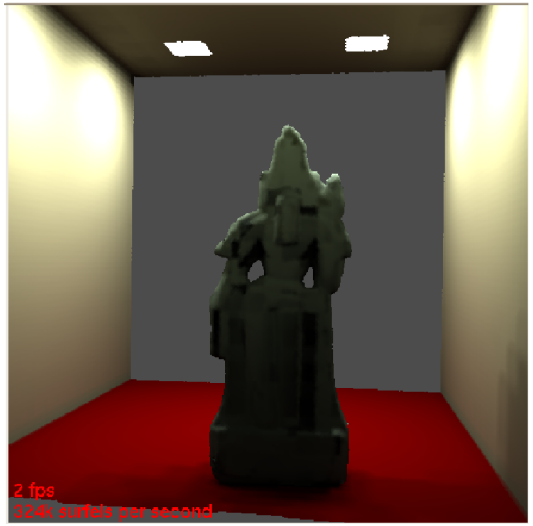




Qualitative Results: Global Illumination



Qualitative Results: Global Illumination



Qualitative Results: Global Illumination

Quantitative Results

Model	Points (millions)	N^2 possible links (millions)	V-Map Links (millions)	% Decrease	Memory(MB) N^2 links	Memory(MB) V-Map links	Build V-Map Time(secs)
ECR	0.1	1.4	0.27	79.5%	5.35	1.09	20.6
PCR	0.14	3.85	0.67	82.62%	15.43	2.68	23.8
BUN	0.15	1.53	0.38	74.64%	6.09	1.5	21.7
DRA	0.55	2.75	0.43	84.54%	11.0	1.7	23.5
BUD	0.67	1.58	0.39	74.75%	6.33	1.6	23.9
GAN	0.15	1.56	0.38	75.64%	6.2	1.55	22.0
GOD	0.17	1.62	0.4	75.31%	6.4	1.63	22.9

Overview

Conclusion

- The lack of surface information in point models creates difficulties in operations like generating global illumination effects and computing point-pair visibility
- Point-to-Point Visibility is arguably one of the most difficult problems in rendering since the interaction between two primitives depends on the rest of the scene
- One way to reduce the difficulty is to consider clustering of regions such that their mutual visibility is resolved at a group level (V-Map)
- Visibility Map data structure we propose enables efficient answer to common rendering queries
- In this paper, we have given a novel, provably efficient, hierarchical, visibility determination scheme for point based models
- By viewing this visibility map as a 'preprocessing' step, photo-realistic global illumination rendering of complex point-based models have been shown
- If analyzed properly, the visibility algorithm is *embarrassingly parallel*


Marc Levoy, Kari Pulli, Brian Curless, Szymon Rusinkiewicz, David Koller, Lucas Pereira, Matt Ginzton, Sean Anderson, James Davis, Jeremy Ginsberg, Jonathan Shade, and Duane Fulk.
The digital michelangelo project: 3D scanning of large statues.
In Kurt Akeley, editor, *Siggraph 2000, Computer Graphics Proceedings*, pages 131–144. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000.


Marc Levoy and Turner Whitted.
The use of points as a display primitive.
Technical Report TR 85-022, University of North Carolina at Chapel Hill, 1985.

Thank you for your time !

Questions ?