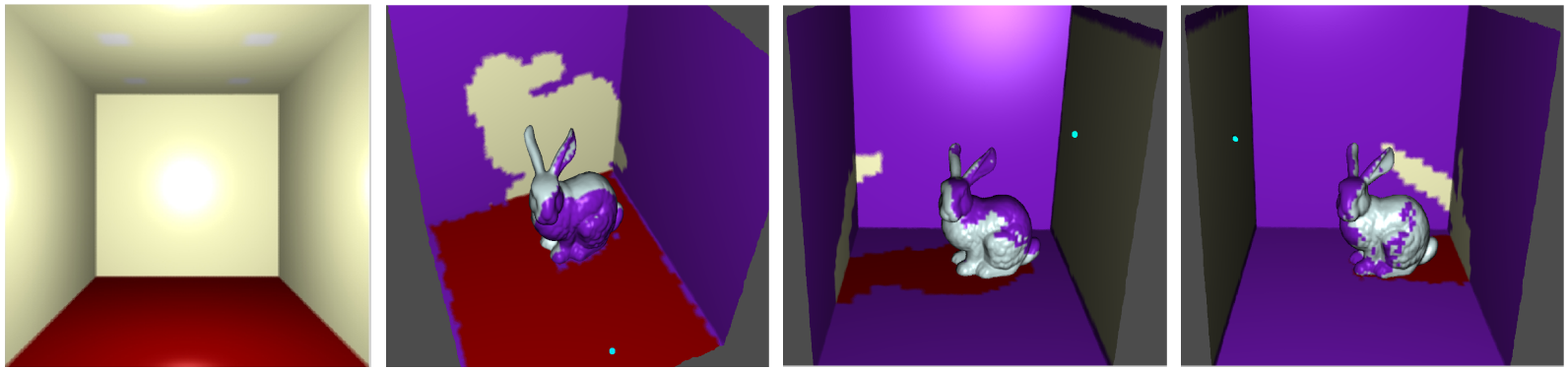
Visibility Map for Global Illumination in Point Clouds



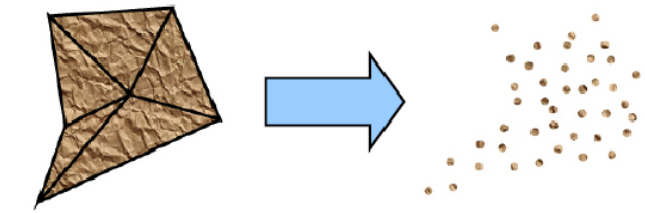
http://www.cse.iitb.ac.in/∼sharat

Acknowledgments:
Joint work with Rhushabh Goradia. Thanks to ViGIL, CSE dept, and IIT Bombay
(Based on ACM Siggraph Graphite (2007) paper)

Overview

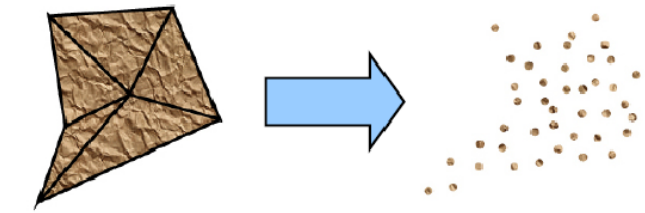
Overview

What are Point Models?



- Model surfaces as points
- Each point has attributes: [coordinates, normal, reflectance, emmisivity]
- Immediate question: Why not triangles, why points? And how do we get these points?

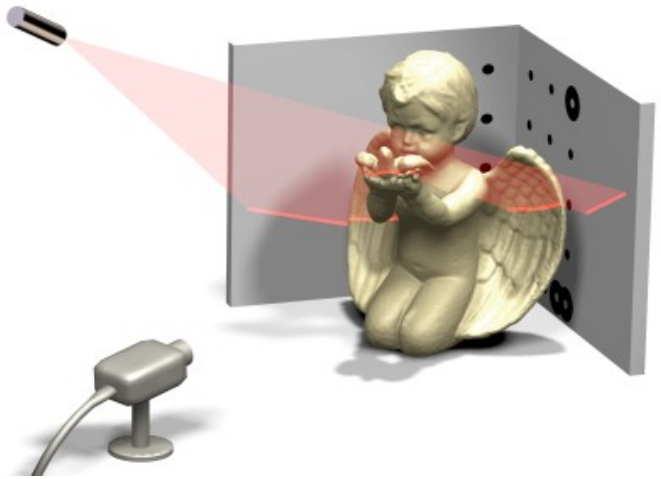
What are Point Models?



- Model surfaces as points
- Each point has attributes: [coordinates, normal, reflectance, emmisivity]
- Immediate question: Why not triangles, why points? And how do we get these points?

Polygons v/s points: The data

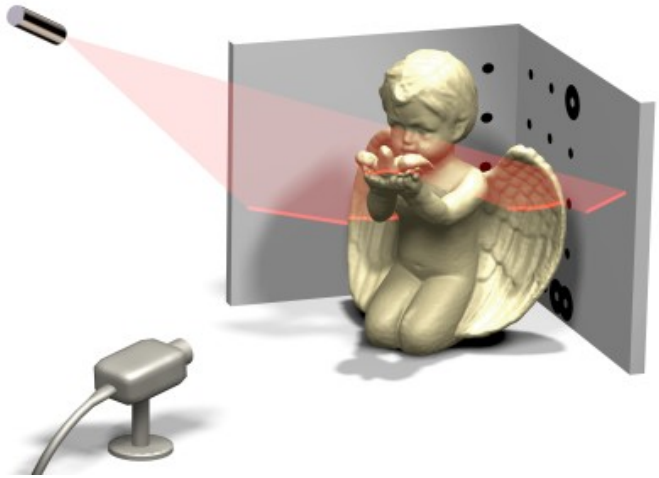
- Laser range scanners produce 3D data



- Creating a consistent polygon mesh is expensive
- Maintaining a consistent polygon mesh is expensive: Dynamic shapes (e.g., fluttering flag)

Polygons v/s points: The data

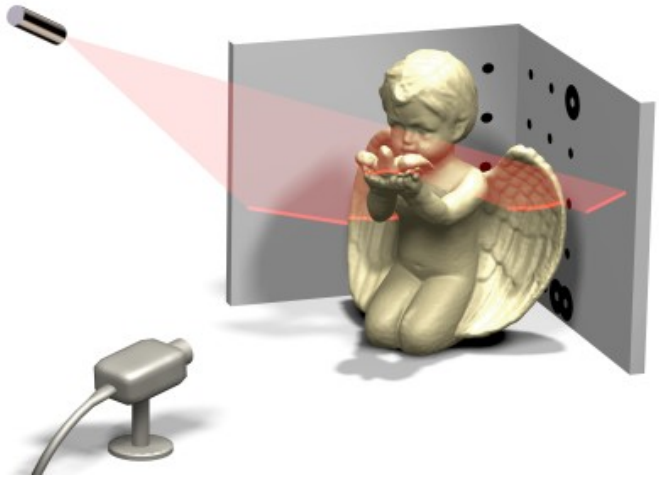
- Laser range scanners produce 3D data



- Creating a consistent polygon mesh is expensive
- Maintaining a consistent polygon mesh is expensive: Dynamic shapes (e.g., fluttering flag)

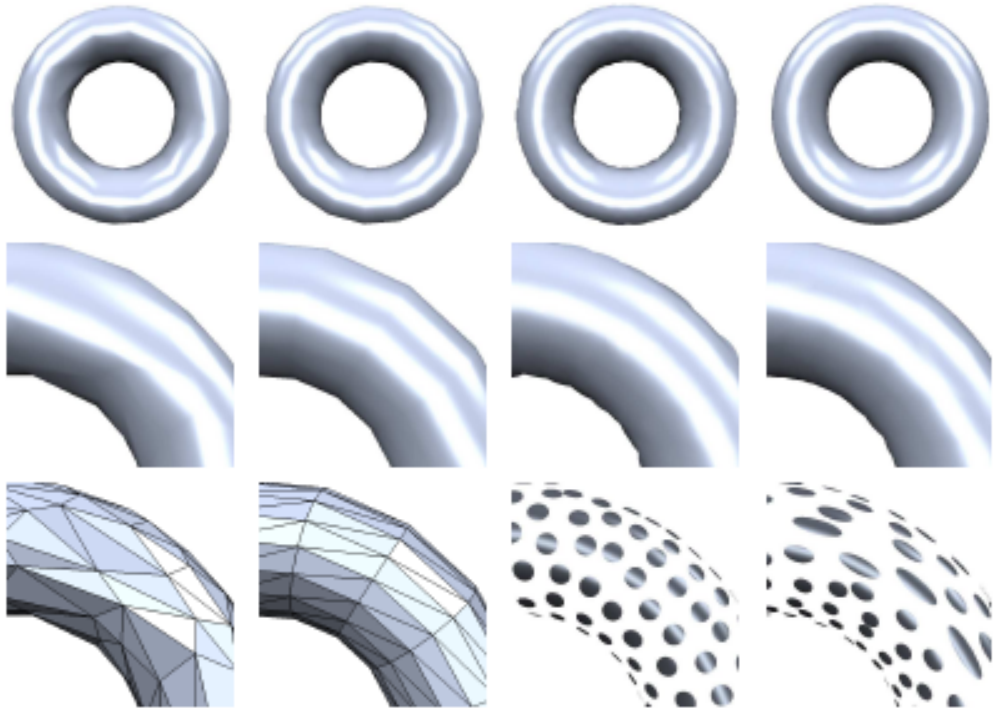
Polygons v/s points: The data

- Laser range scanners produce 3D data



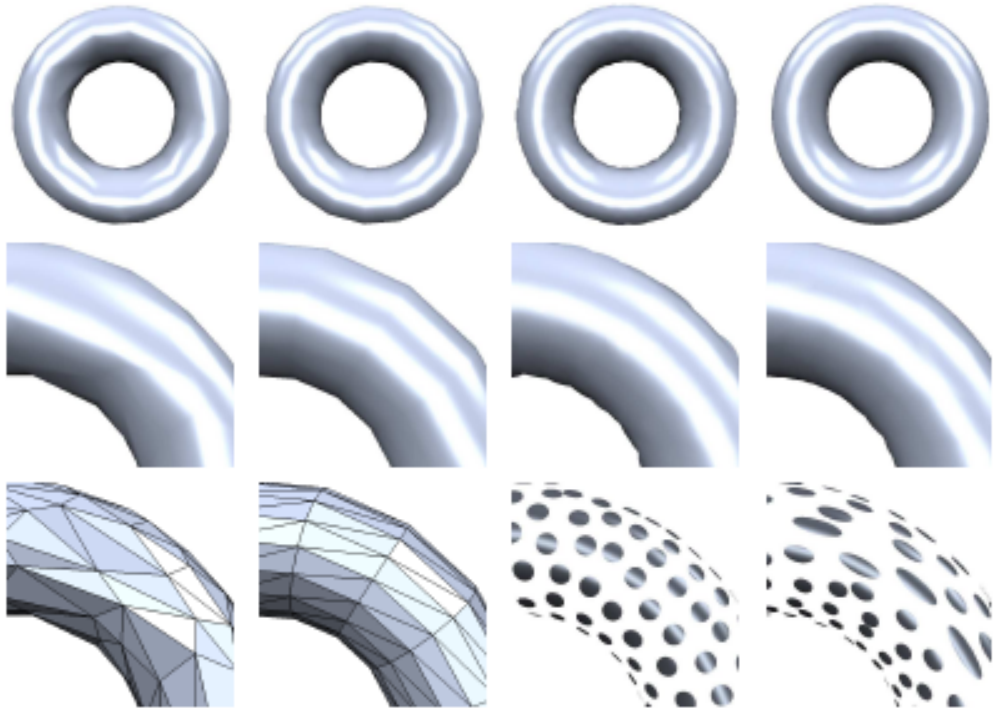
- Creating a consistent polygon mesh is expensive
- Maintaining a consistent polygon mesh is expensive: Dynamic shapes (e.g., fluttering flag)

Polygons v/s points: The Illusion



- Polygons good for large, flat, or subtly curved regions
  Points better for models with details everywhere.

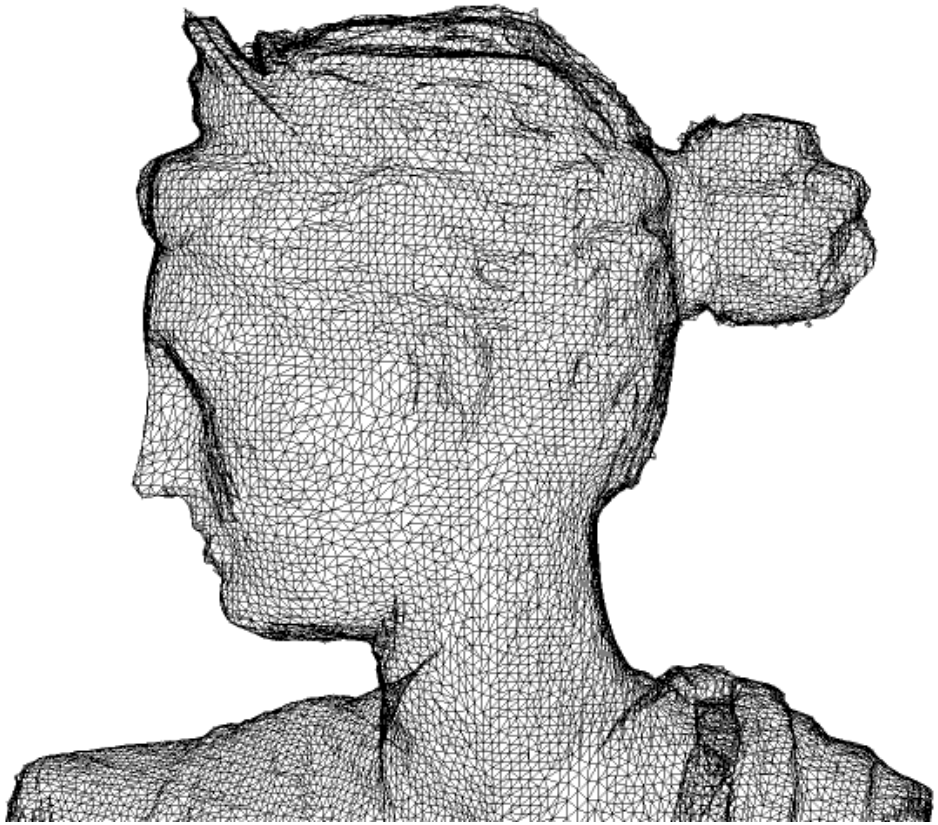
Polygons v/s points: The Illusion



- Polygons good for large, flat, or subtly curved regions
  Points better for models with details everywhere.

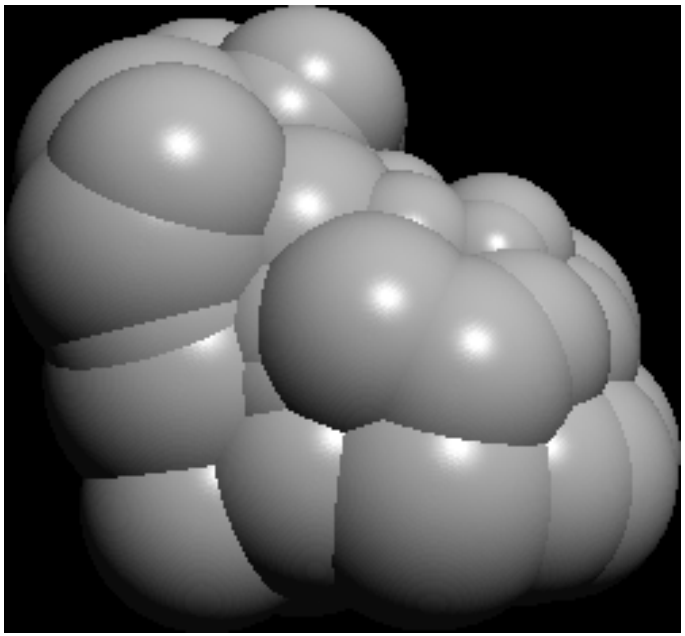
Polygons v/s points: Scan Conversion

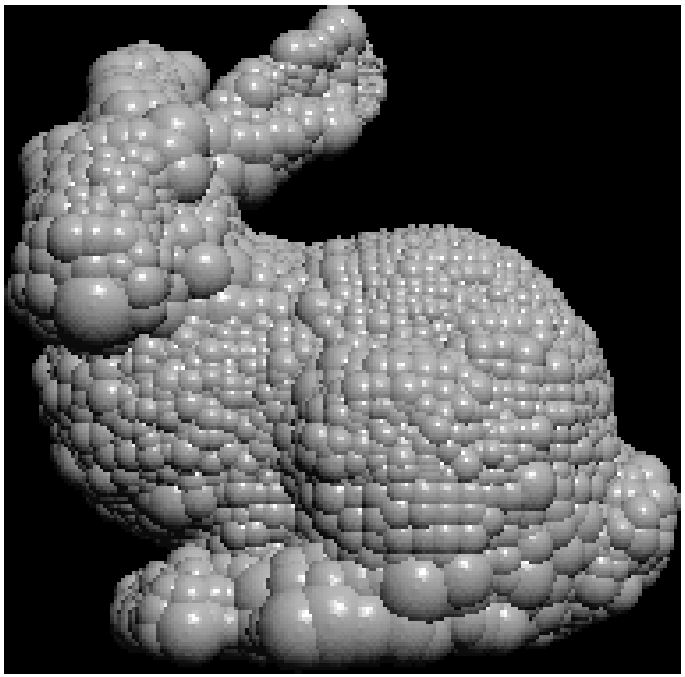
- Smaller polygons lead to higher rasterization costs



Polygons v/s points: LOD

- Level of Detail (LOD) based hierarchy is simpler in point based models





Points Summary

Although the triangle is the defacto standard in representing objects, points are good things to have.

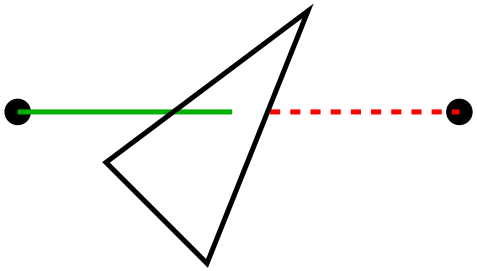
So why have points not been used all this while?
Incidentally, images are now a third representation in computer graphics

Points Summary

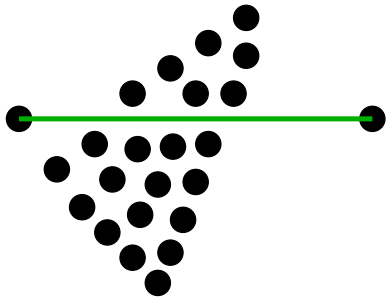
Although the triangle is the defacto standard in representing objects, points are good things to have.

So why have points not been used all this while?
Incidentally, images are now a third representation in computer graphics

Visibility Between Point Pairs



VISIBILITY IN POLYGONAL MODELS



VISIBILITY IN POINT MODELS

Point representation implies holes

What are Global Illumination Algorithms?

Global illumination algorithms are those which, when determining the light falling on a surface, take into account not only the light which has taken a path directly from a light source (direct illumination), but also light which has undergone reflection from other surfaces in the world (indirect illumination).

- GI is "obviously" needed: we see it all the time
- GI is expensive: The appearance at a point depends on receiving illumination from all other points, which in turn depends on other points

What are Global Illumination Algorithms?

Global illumination algorithms are those which, when determining the light falling on a surface, take into account not only the light which has taken a path directly from a light source (direct illumination), but also light which has undergone reflection from other surfaces in the world (indirect illumination).

- GI is “obviously” needed: we see it all the time
- GI is expensive: The appearance at a point depends on receiving illumination from all other points, which in turn depends on other points

What are Global Illumination Algorithms?

Global illumination algorithms are those which, when determining the light falling on a surface, take into account not only the light which has taken a path directly from a light source (direct illumination), but also light which has undergone reflection from other surfaces in the world (indirect illumination).

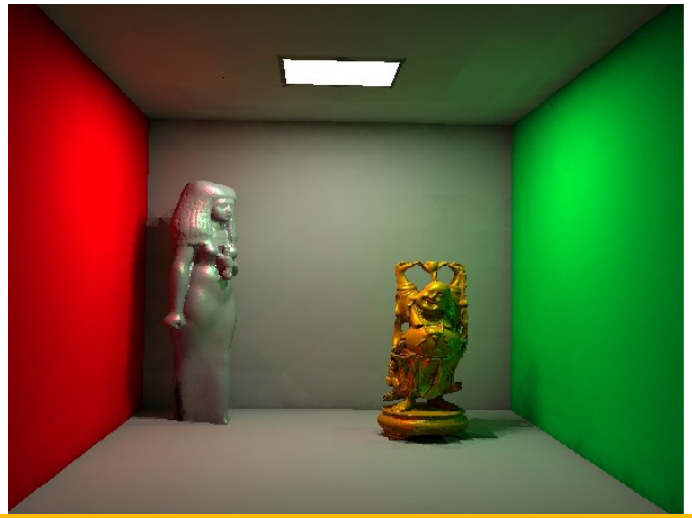
- GI is “obviously” needed: we see it all the time
- GI is expensive: The appearance at a point depends on receiving illumination from all other points, which in turn depends on other points

What are Global Illumination Algorithms?

Global illumination algorithms are those which, when determining the light falling on a surface, take into account not only the light which has taken a path directly from a light source (direct illumination), but also light which has undergone reflection from other surfaces in the world (indirect illumination).

- GI is “obviously” needed: we see it all the time
- GI is expensive: The appearance at a point depends on receiving illumination from all other points, which in turn depends on other points

Examples showing GI Effects

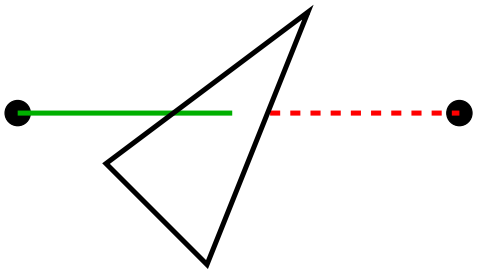


Application Domains

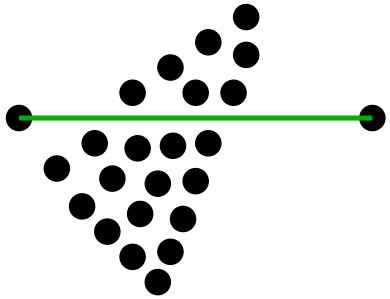
Visibility Between Point Pairs

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

Visibility Between Point Pairs



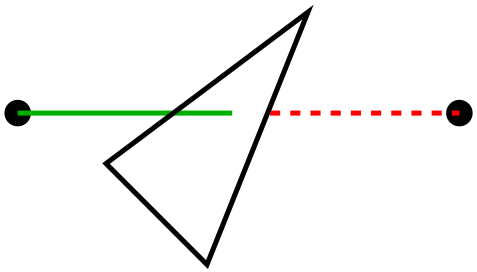
VISIBILITY IN POLYGONAL MODELS



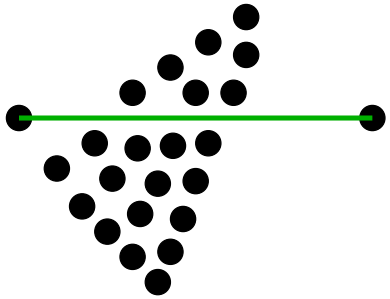
VISIBILITY IN POINT MODELS

- View dependent visibility versus view independent visibility
- Although view dependent visibility based point based rendering solutions exist, we present the first global illumination solution for point models based on the view independent paradigm

Visibility Between Point Pairs



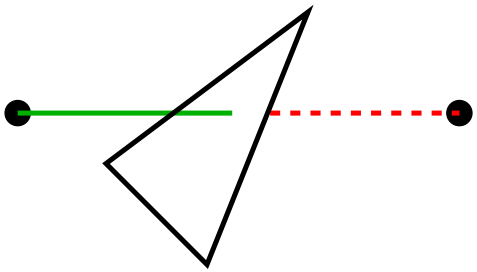
VISIBILITY IN POLYGONAL MODELS



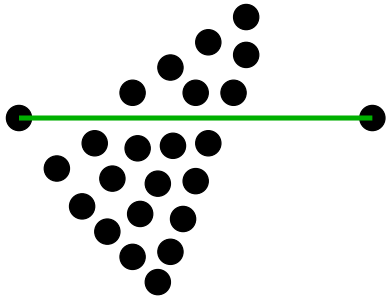
VISIBILITY IN POINT MODELS

- View dependent visibility versus view independent visibility
- Although view dependent visibility based point based rendering solutions exist, we present the first global illumination solution for point models based on the view independent paradigm

Visibility Between Point Pairs



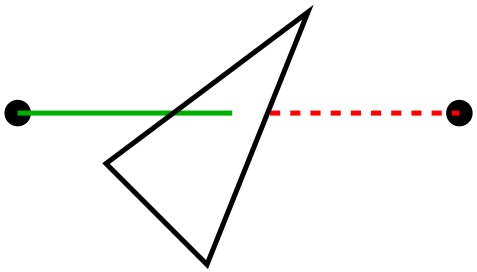
VISIBILITY IN POLYGONAL MODELS



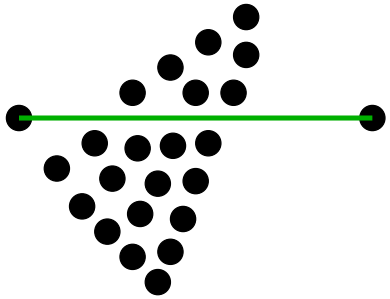
VISIBILITY IN POINT MODELS

- View dependent visibility versus view independent visibility
- Although view dependent visibility based point based rendering solutions exist, we present the first global illumination solution for point models based on the view independent paradigm

Visibility Between Point Pairs



VISIBILITY IN POLYGONAL MODELS

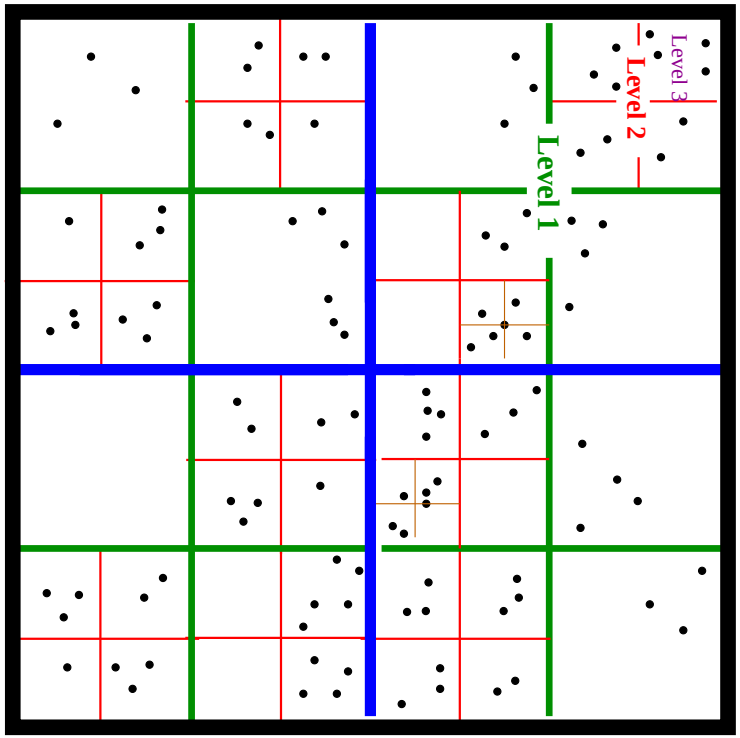


VISIBILITY IN POINT MODELS

- View dependent visibility versus view independent visibility
- Although view dependent visibility based point based rendering solutions exist, we present the first global illumination solution for point models based on the view independent paradigm

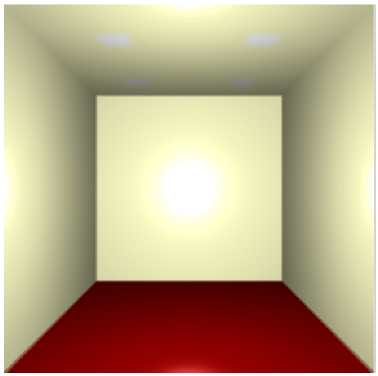
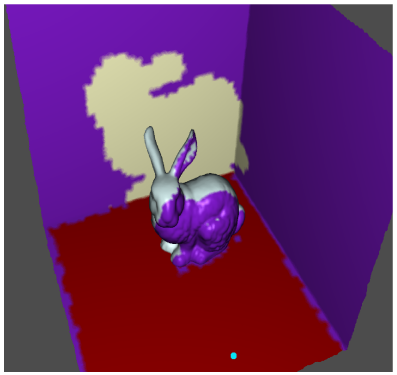
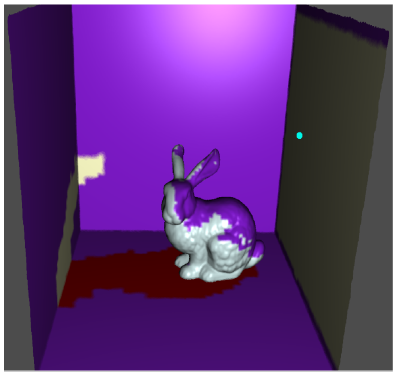
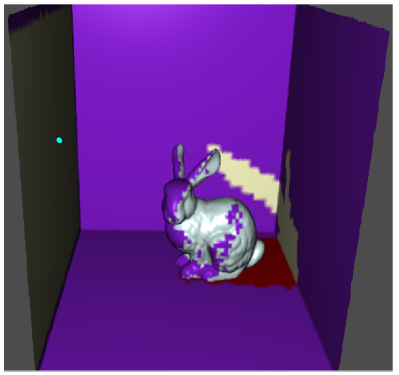
Hierarchical Visibility

Hierarchical Visibility enables *quick* answers to visibility queries, thus enabling a faster GI solution



Hierarchical Visibility

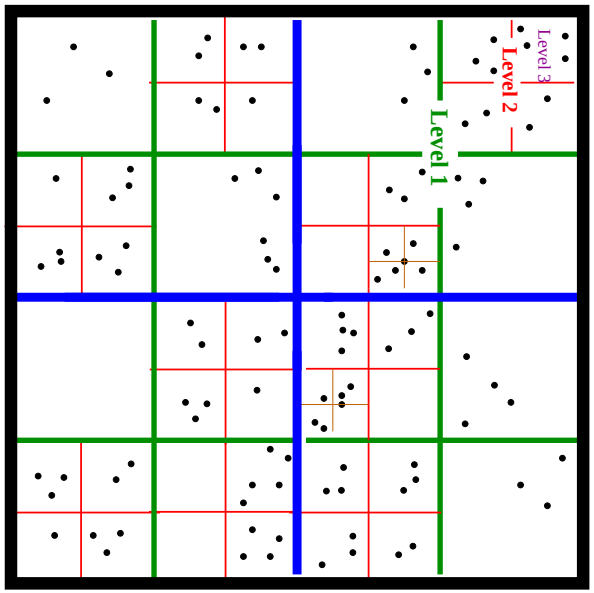
Key Notion: We define a **Visibility Map (V-map)** for the resulting tree to enable *quick* answers to visibility queries.

Overview

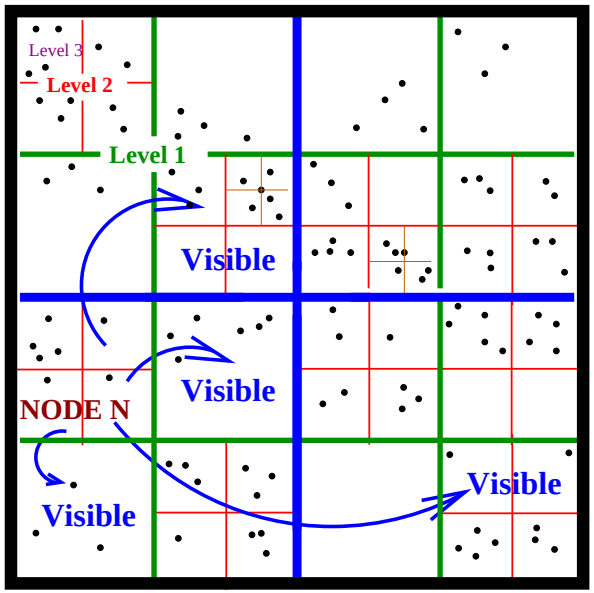
What is a Visibility Map (V-map)?

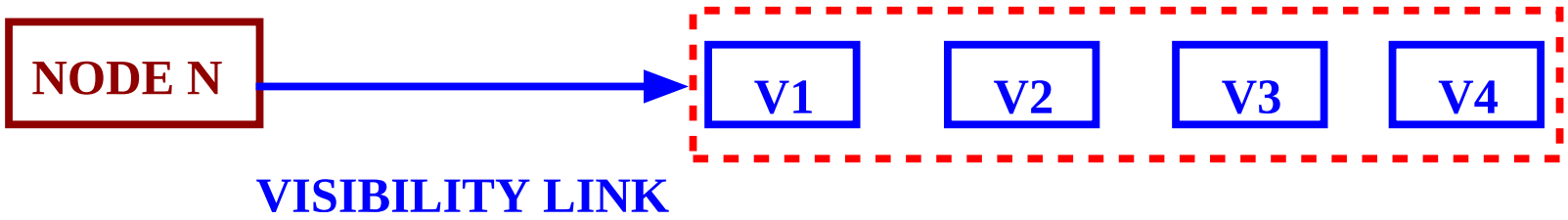
- The *visibility map* for a tree is a collection of visibility links for every node in the tree
- The *visibility link* for any node N is a set L of nodes
- Every point in any node in L is guaranteed to be visible from every point in N



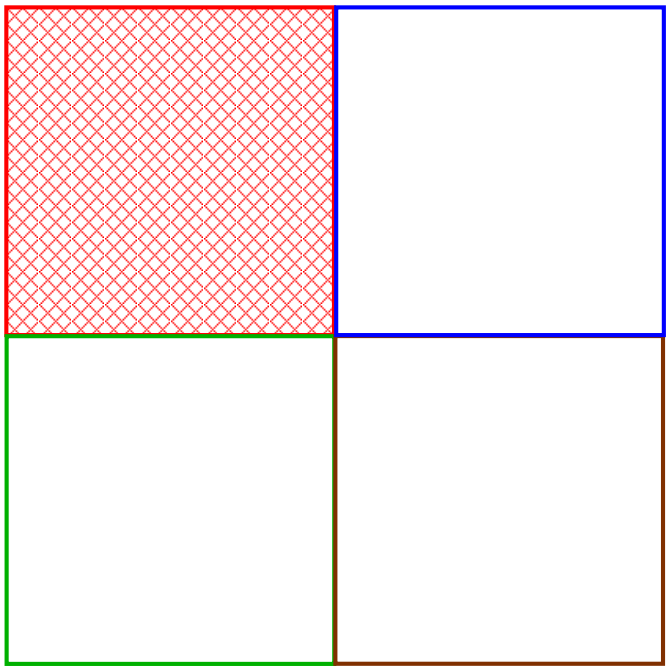
What is a Visibility Map (V-map)?

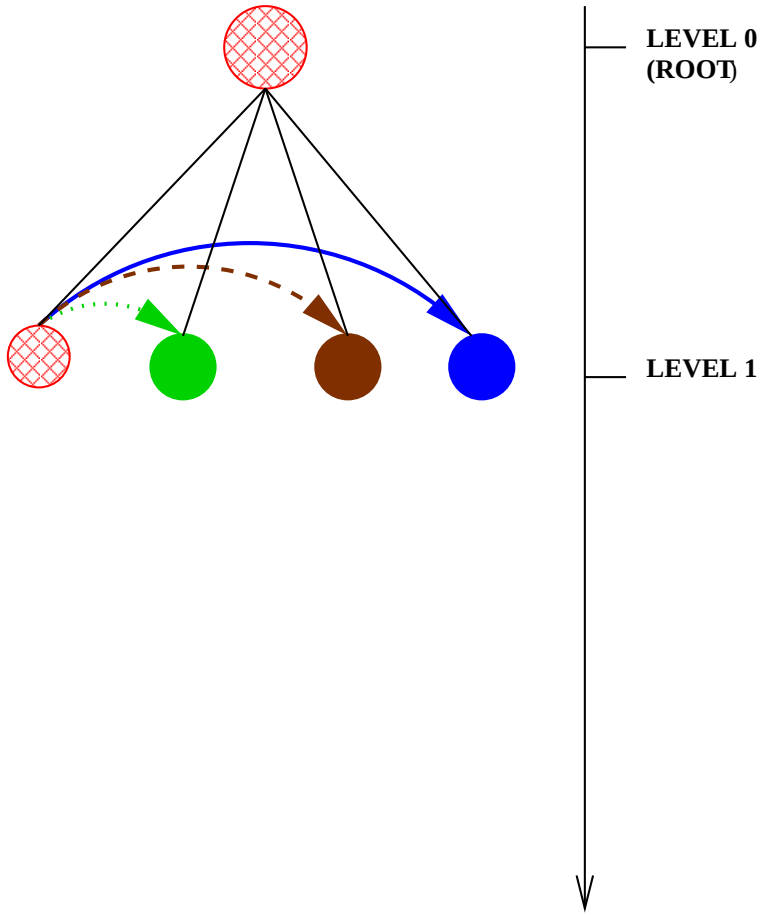
- The *visibility map* for a tree is a collection of visibility links for every node in the tree
- The *visibility link* for any node N is a set L of nodes
- Every point in any node in L is guaranteed to be visible from every point in N





What is a Visibility Map (V-map)?





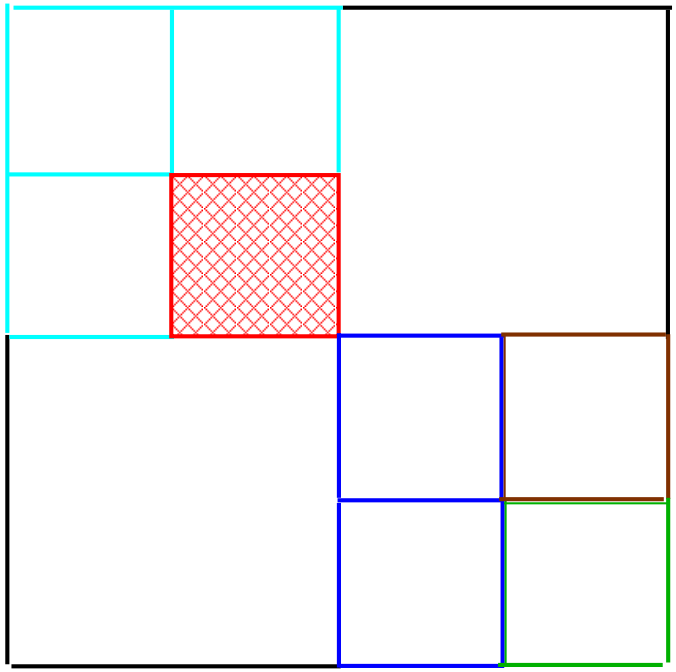
With respect to  at any level,

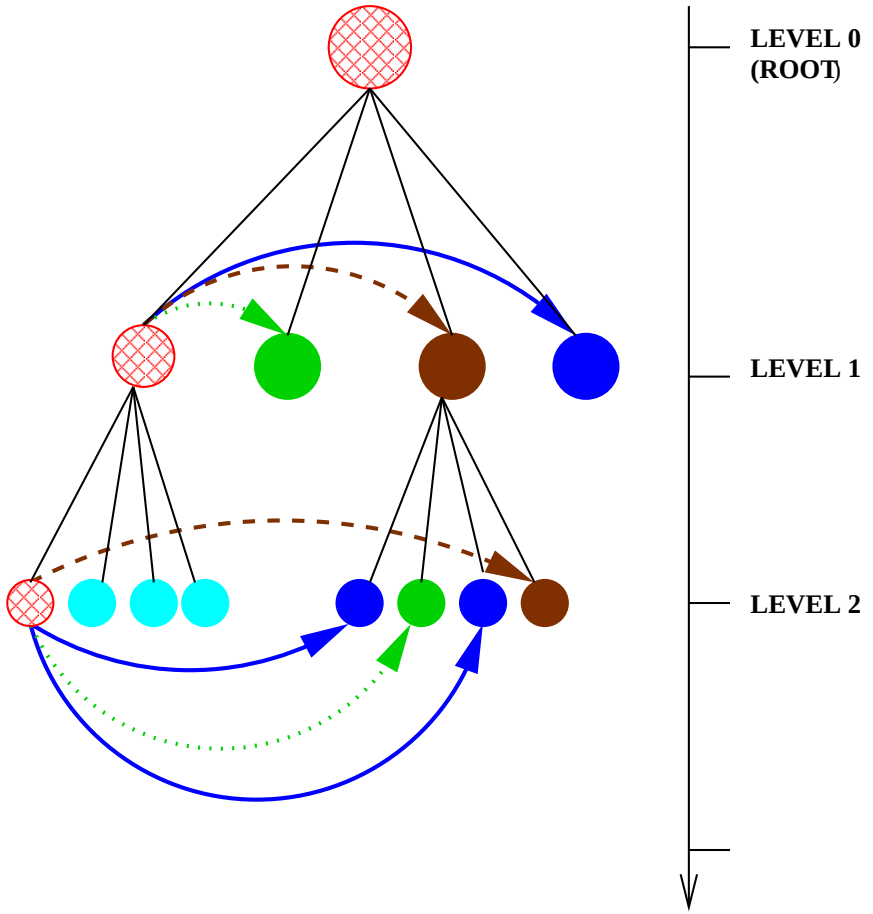
-- COMPLETELY INVISIBLE

-- COMPLETELY VISIBLE

-- PARTIALLY VISIBLE

What is a Visibility Map (V-Map)?





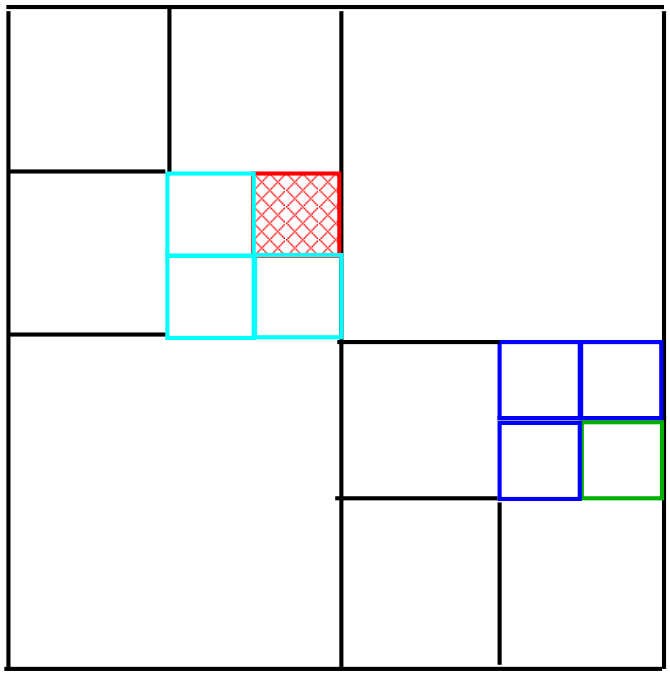
With respect to  at any level,

-- COMPLETELY INVISIBLE

-- COMPLETELY VISIBLE

-- PARTIALLY VISIBLE

What is a Visibility Map (V-Map)?

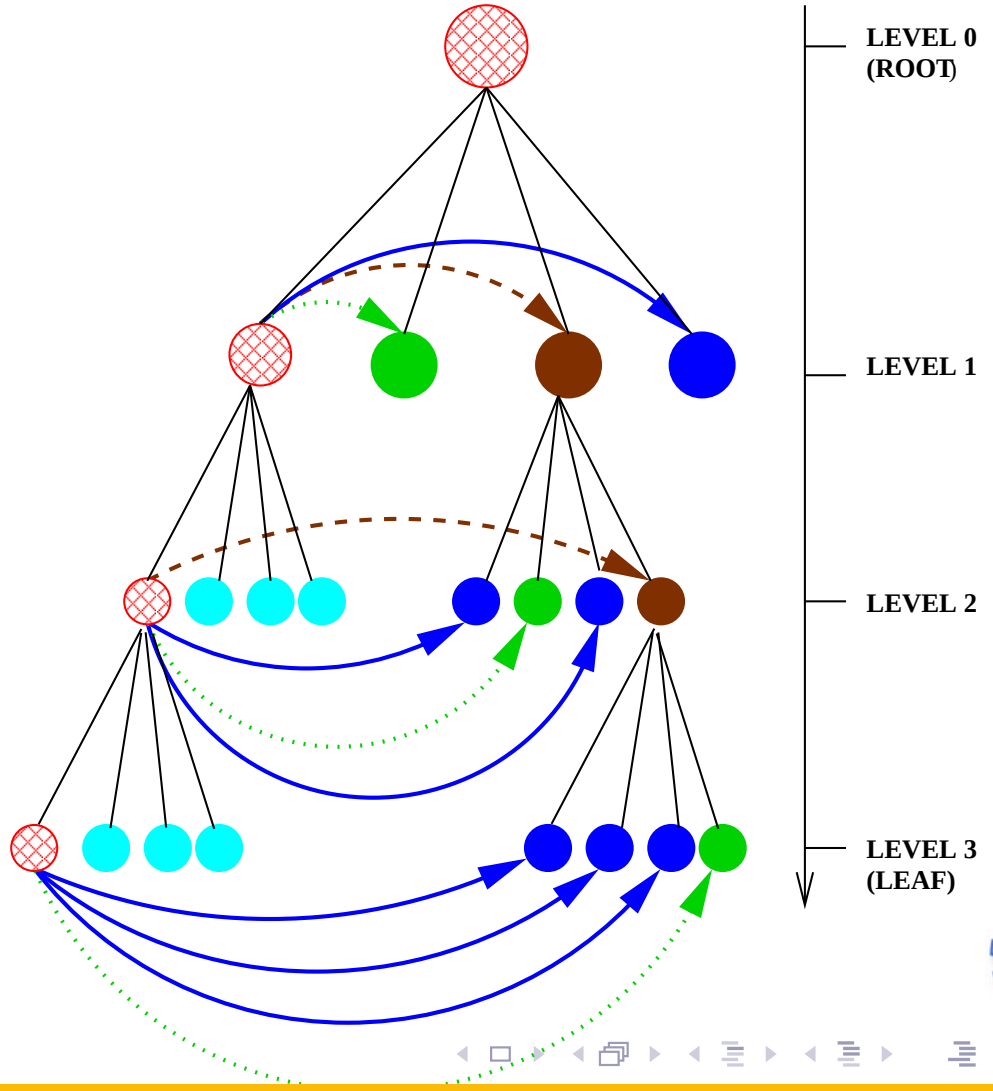


With respect to  at any level,

-- COMPLETELY INVISIBLE

-- COMPLETELY VISIBLE

-- PARTIALLY VISIBLE



Visibility Map Queries?

Visibility map entertain efficient answers:

1. Is point x visible from 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 queries answered with a simple octree traversal

Visibility Map Queries?

Visibility map entertain efficient answers:

1. Is point x visible from 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 queries answered with a simple octree traversal

Visibility Map Queries?

Visibility map entertain efficient answers:

1. Is point x visible from 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 queries answered with a simple octree traversal

Visibility Map Queries?

Visibility map entertain efficient answers:

1. Is point x visible from 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 queries answered with a simple octree traversal

Computational Complexity

- Visibility problem provides answer to pairwise queries, and visibility is not a transitive phenomenon.
- Our algorithm, results in an **overall linear time algorithm w.r.t.** N the number of such pairs,
- This is the best possible for *any* algorithm that builds the V-Map
- The overall algorithm consumes a small amount of extra overhead memory

Computational Complexity

- Visibility problem provides answer to pairwise queries, and visibility is not a transitive phenomenon.
- Our algorithm, results in an **overall linear time algorithm w.r.t.** N the number of such pairs,
- This is the best possible for *any* algorithm that builds the V-Map
- The overall algorithm consumes a small amount of extra overhead memory

Computational Complexity

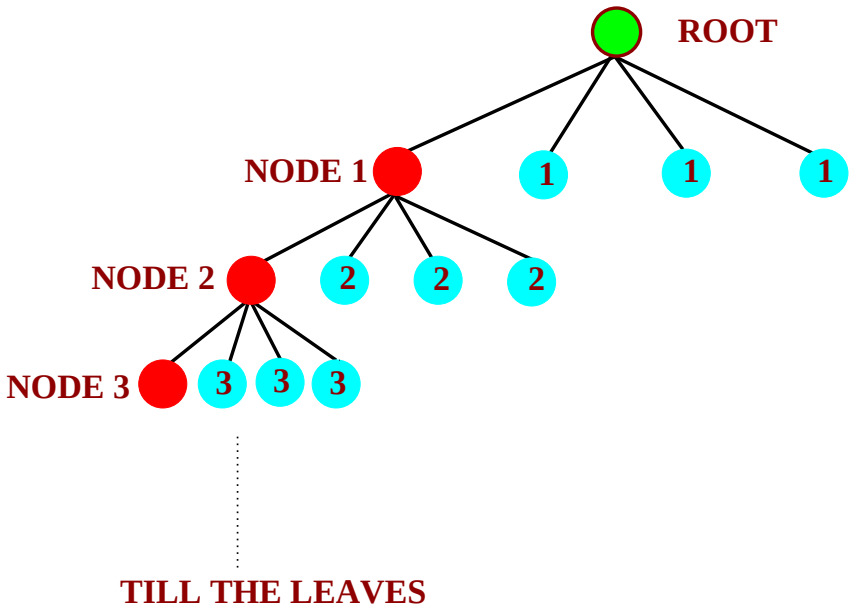
- Visibility problem provides answer to pairwise queries, and visibility is not a transitive phenomenon.
- Our algorithm, results in an **overall linear time algorithm w.r.t.** N the number of such pairs,
- This is the best possible for *any* algorithm that builds the V-Map
- The overall algorithm consumes a small amount of extra overhead memory

Computational Complexity

- Visibility problem provides answer to pairwise queries, and visibility is not a transitive phenomenon.
- Our algorithm, results in an **overall linear time algorithm w.r.t.** N the number of such pairs,
- This is the best possible for *any* algorithm that builds the V-Map
- The overall algorithm consumes a small amount of extra overhead memory

V-map Construction Algorithm

- Initialize the old interaction list (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 traversal algorithms
  - reduces redundant computation

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

```
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
```

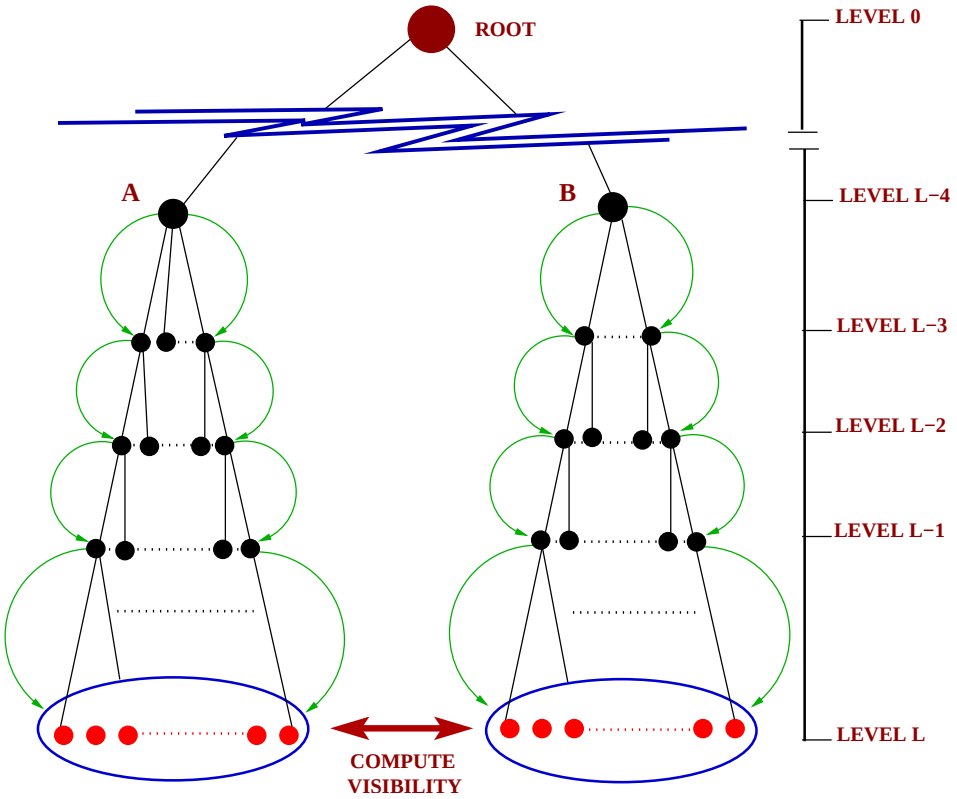
- V-map constructed by calling initially for the root, which sets up the relevant visibility links in n-IL
- NodetoNodeVisibility(A,B)
  - Constructs the visibility links for all descendants of A w.r.t all descendants of B (and vice-versa) at the best (i.e. highest) possible level.
  - This ensures an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

V-map Construction Algorithm

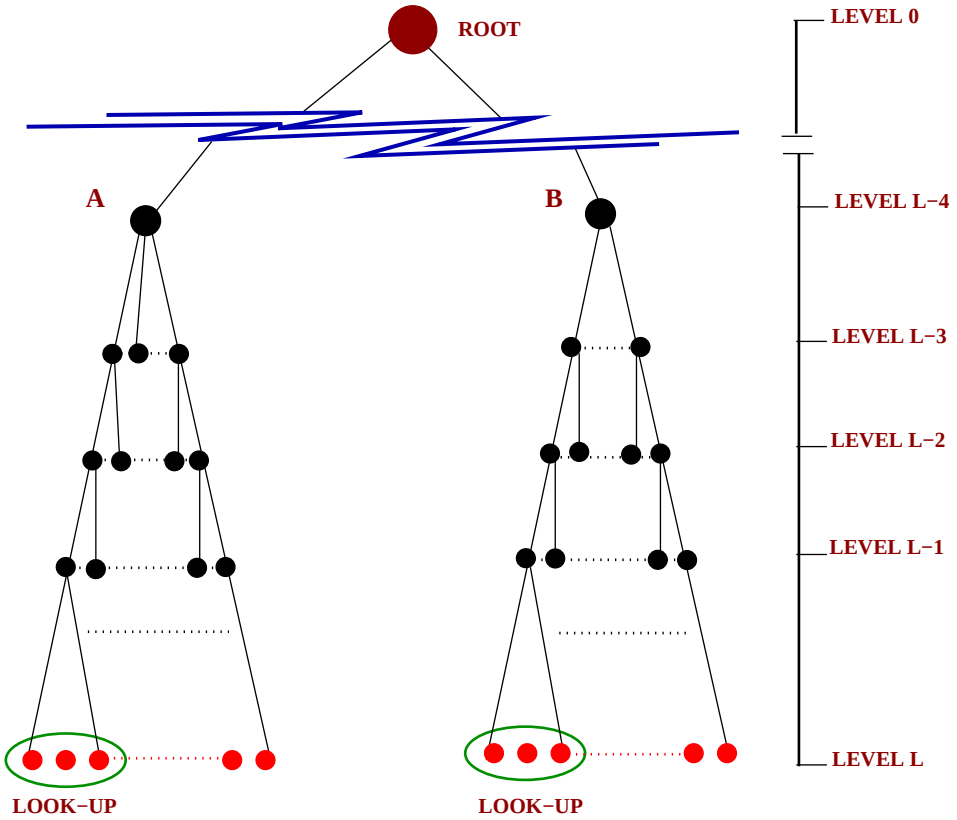
```
procedure OctreeVisibility(Node A)
for each node B in old interaction list (o-IL) of A do
   if NodetoNodeVisibility(A,B) == VISIBLE then
      add B in new interaction list (n-IL) of A
      add A in new interaction list (n-IL) of B
   end if
   remove A from old interaction list (o-IL) of B
end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

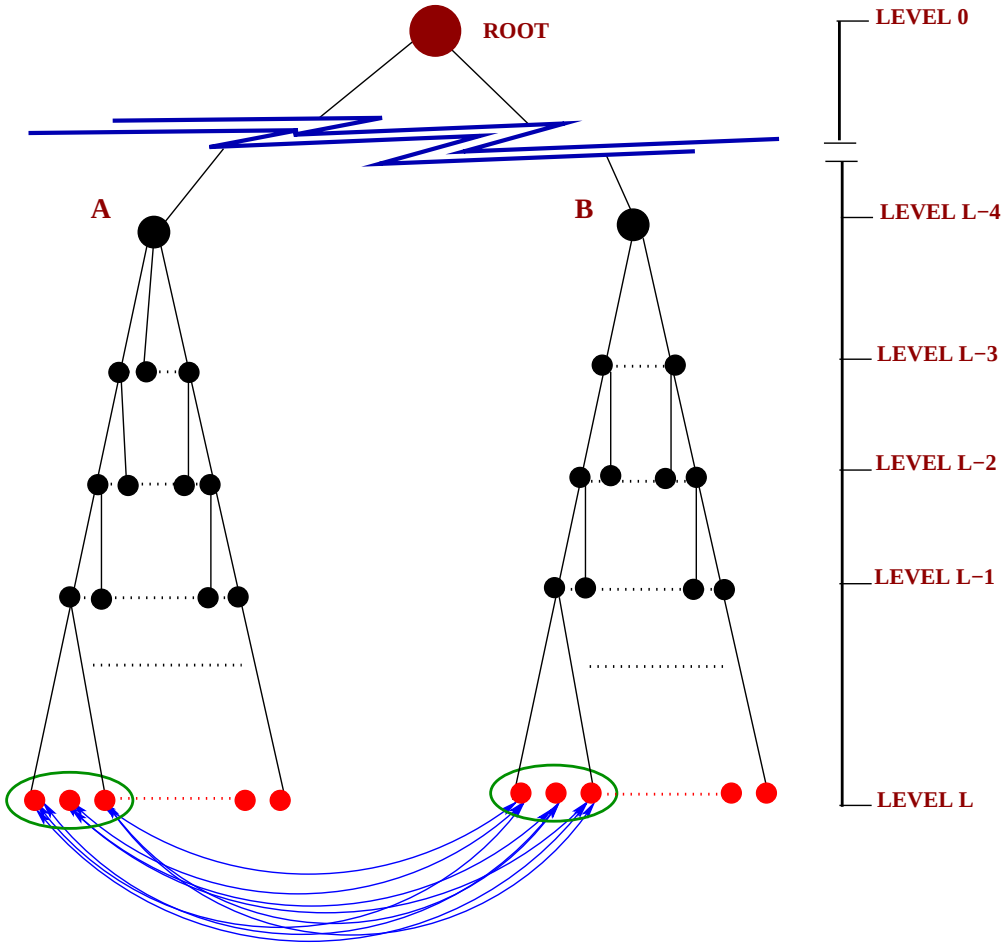
V-map Construction Algorithm



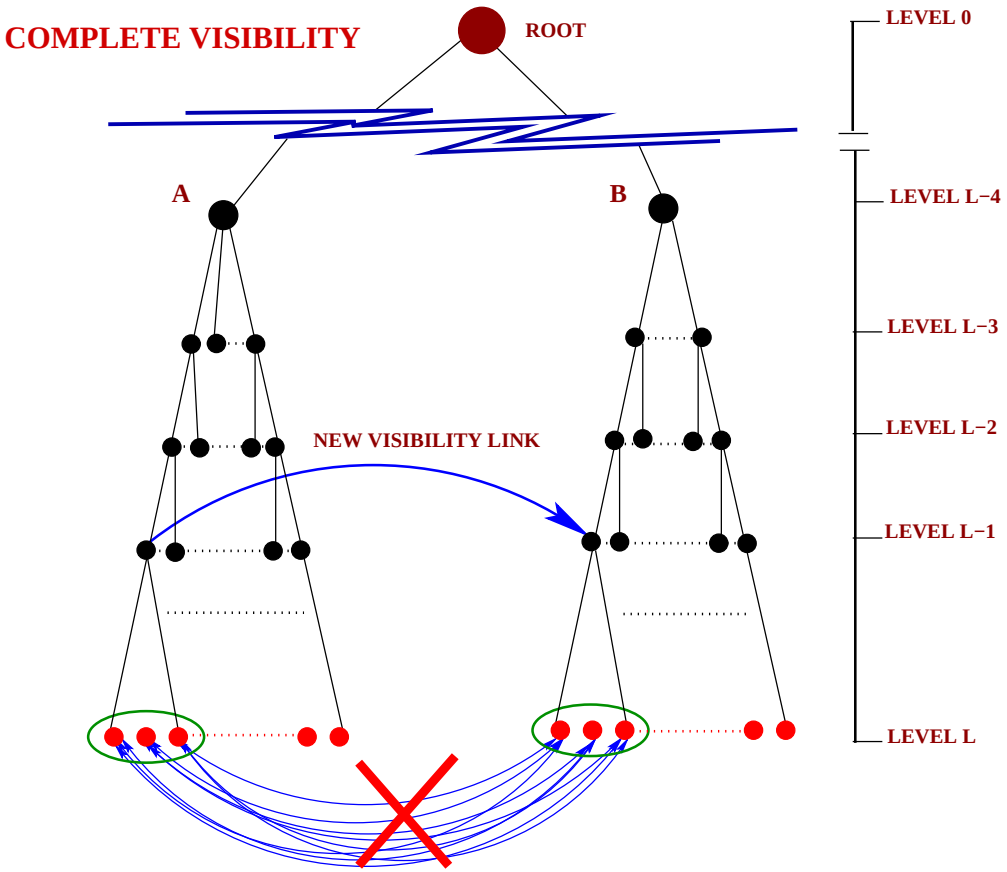
V-map Construction Algorithm



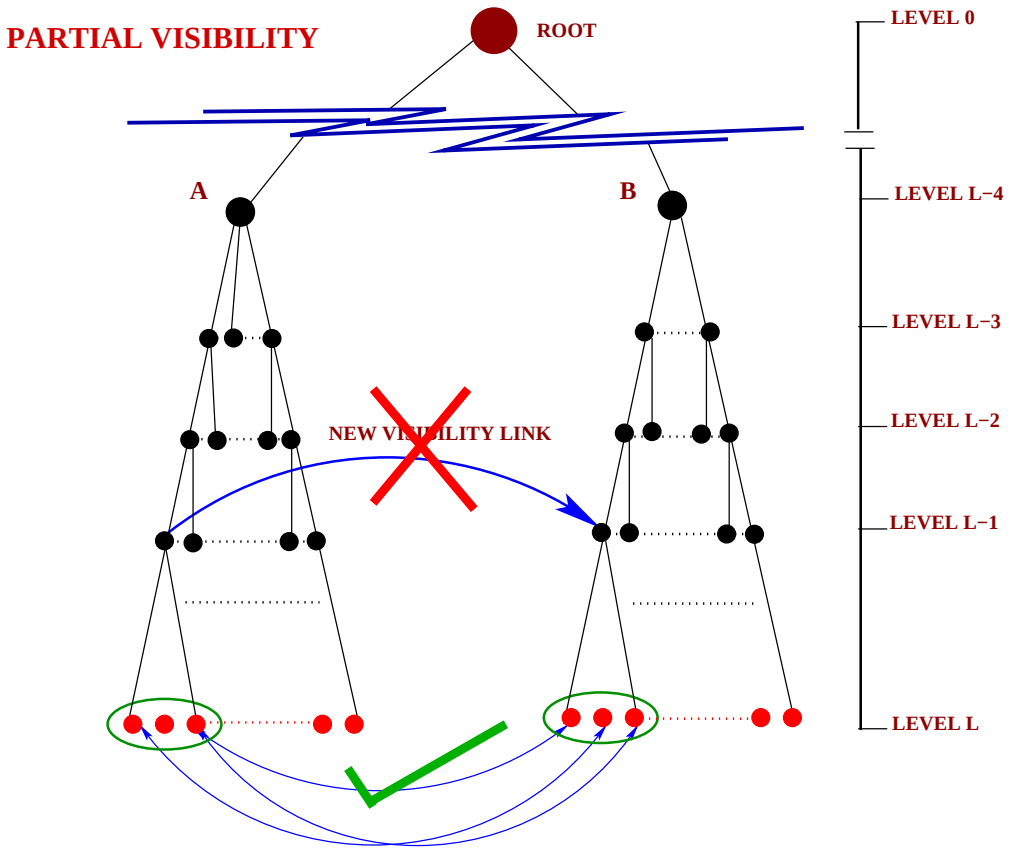
V-map Construction Algorithm



V-map Construction Algorithm

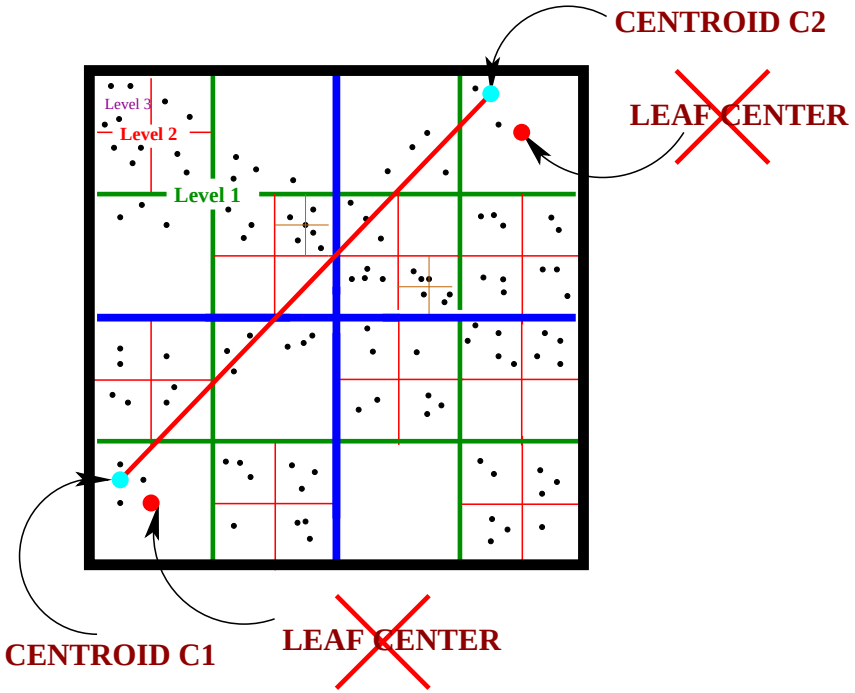


V-map Construction Algorithm

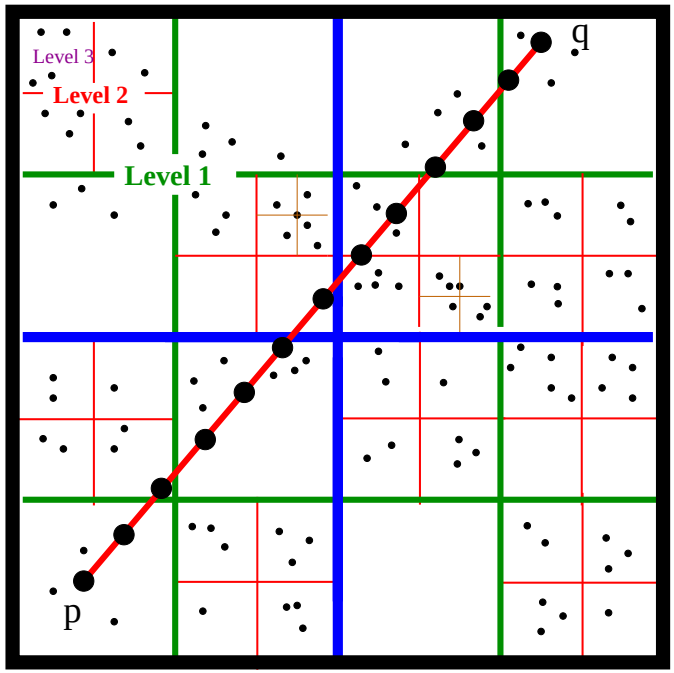


Leaf-Leaf Visibility Algorithm

- Consider centroid and **NOT** leaf center



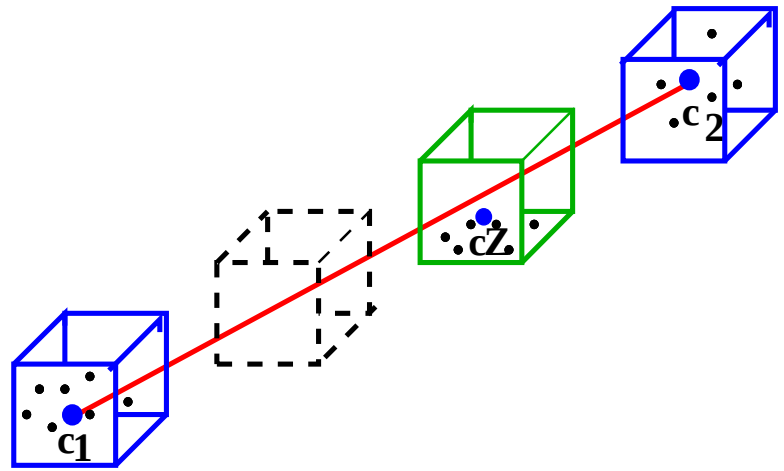
Leaf Pair Visibility

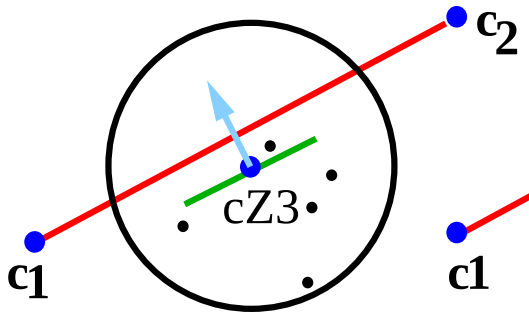


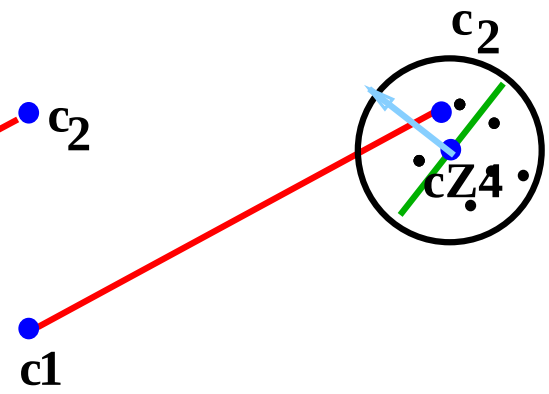
Finding Potential Occluders using the Bresenham algorithm

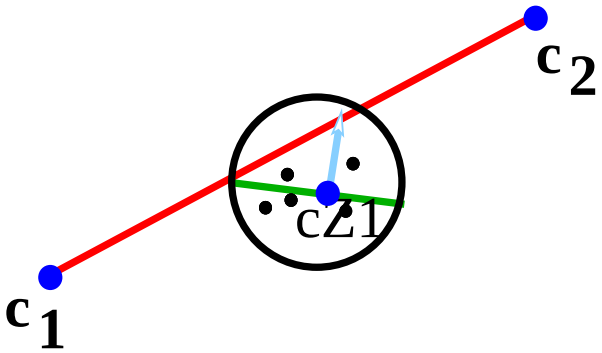
Approximate visibility revisited

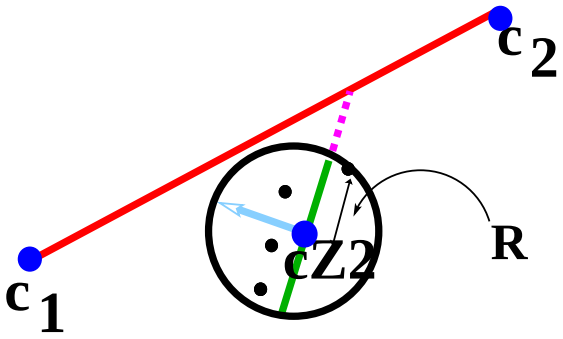
Leaf-Leaf Visibility Algorithm











Leaf-Leaf Visibility Algorithm

- Distance **R** is unique for each leaf and depends on distribution of points and **not** the size of leaf.
- **R** is not a user-input
- The **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.

Leaf-Leaf Visibility Algorithm

- Distance **R** is unique for each leaf and depends on distribution of points and **not** the size of leaf.
- **R** is not a user-input
- The **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.

Leaf-Leaf Visibility Algorithm

- Distance **R** is unique for each leaf and depends on distribution of points and **not** the size of leaf.
- **R** is not a user-input
- The **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.

Leaf-Leaf Visibility Algorithm

- Distance **R** is unique for each leaf and depends on distribution of points and **not** the size of leaf.
- **R** is not a user-input
- The **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.

Leaf-Leaf Visibility Algorithm

- Distance **R** is unique for each leaf and depends on distribution of points and **not** the size of leaf.
- **R** is not a user-input
- The **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

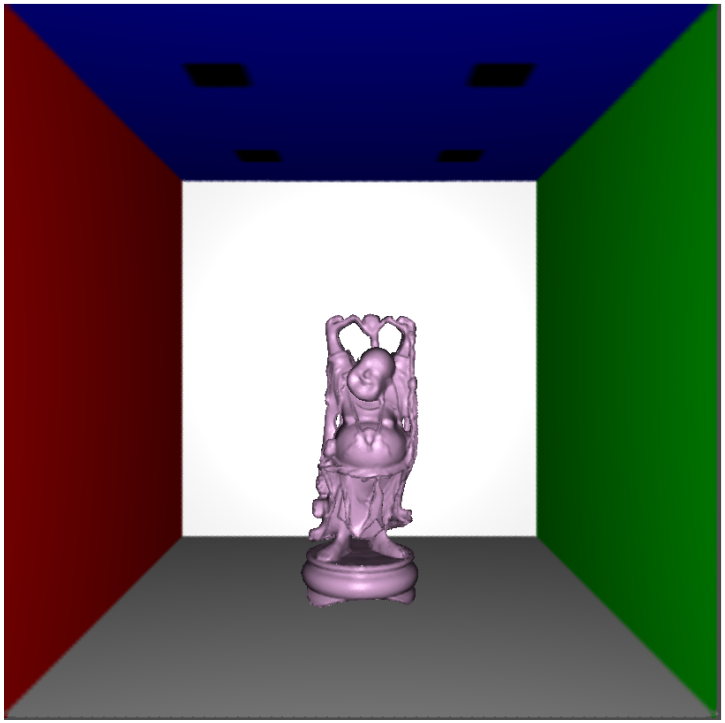
NOTE: We perform this visibility computation (with help of **averaged normals**) only for the leaves. There are no **average normals** defined for internal nodes of the tree.

Leaf-Leaf Visibility Algorithm

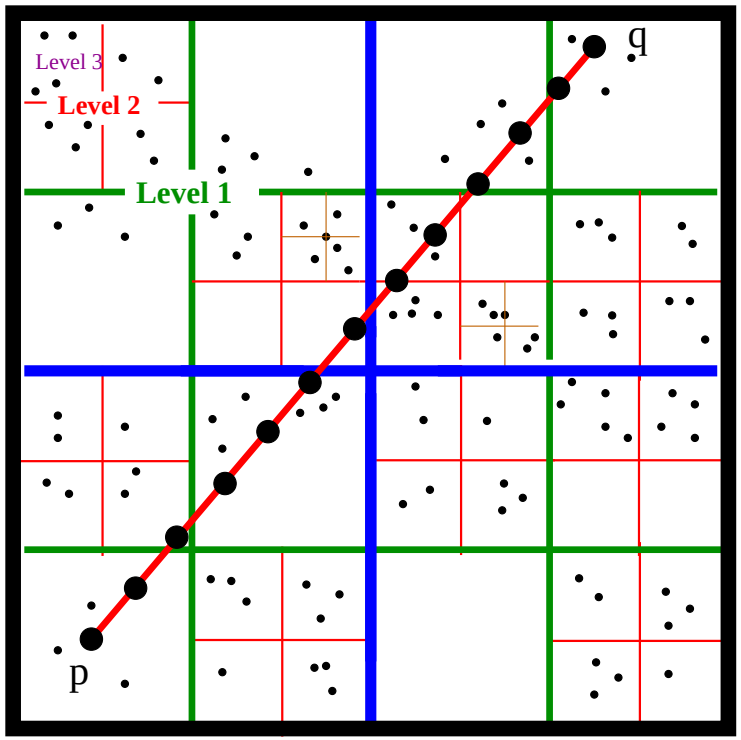
- Distance **R** is unique for each leaf and depends on distribution of points and **not** the size of leaf.
- **R** is not a user-input
- The **strict** visibility condition balances the leniency introduced
- **Faster**, as we exit on finding the first potential occluder
- Dense point models help in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

Extending to Adaptive Octrees



Cornell room (160,000 points) with the Buddha model (534000 points)



Potential Occluders using the Bresenham algorithm

Ray-Sphere Intersection Algorithm

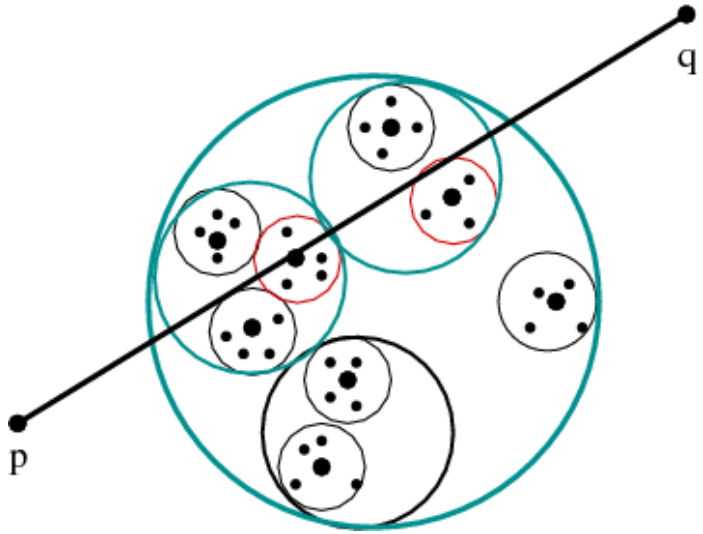
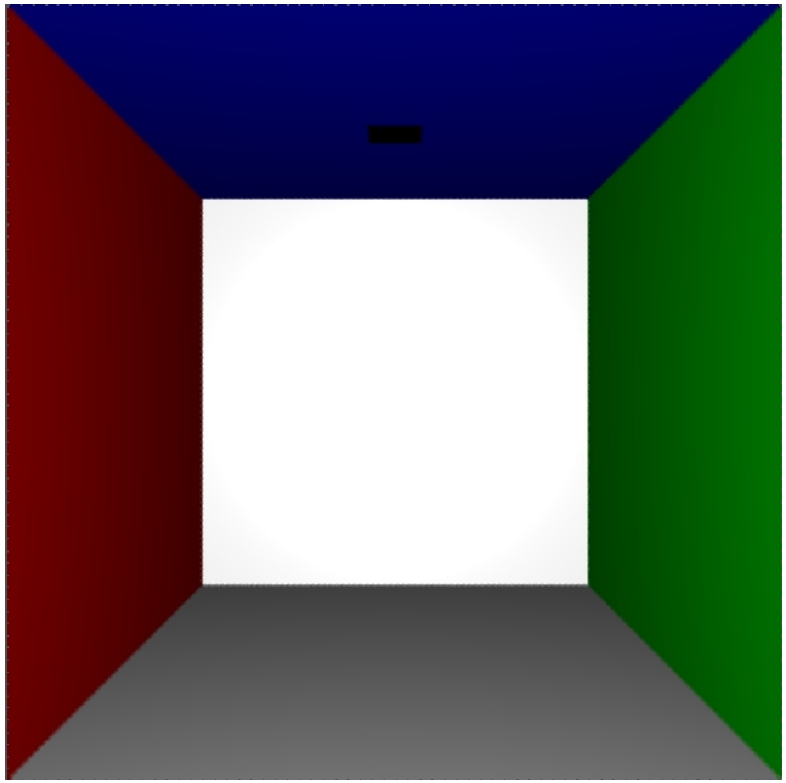


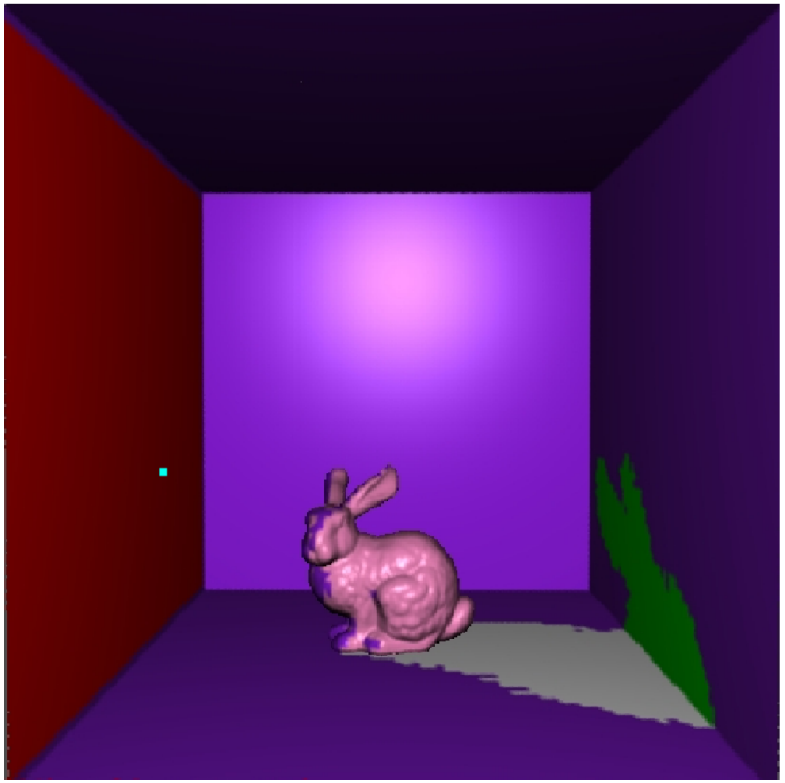
Figure: Ray-Sphere intersection algorithm to determine point-point visibility

- If node is not a leaf and $\overline{pq}$ intersects the node then traverse its children
- If node is a leaf then check whether tangent plane of that node intersects pq within radius R then node p and q are invisible otherwise declare p and q visible

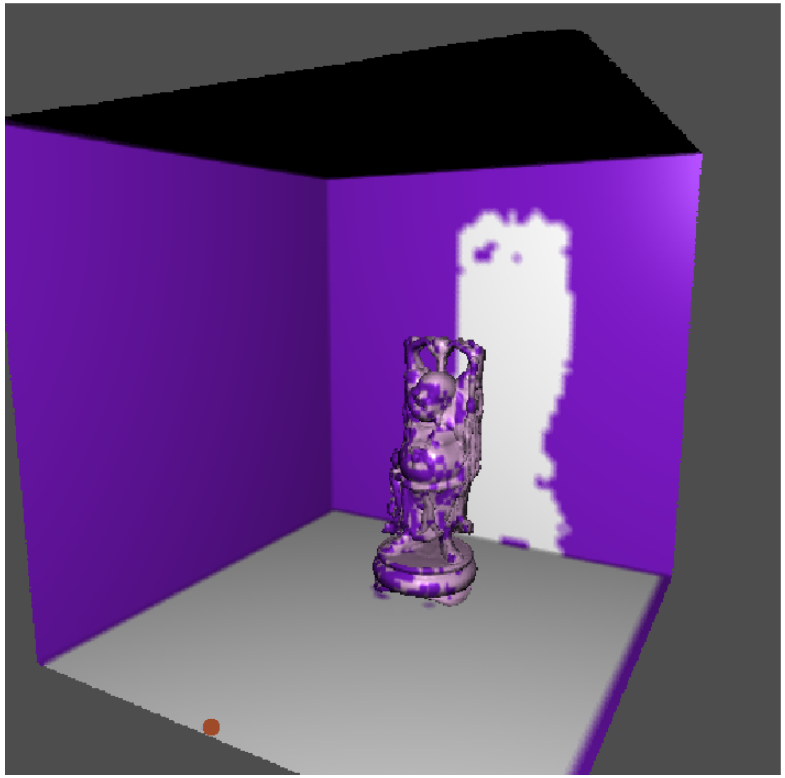
Overview

Qualitative Results: Visibility Correctness

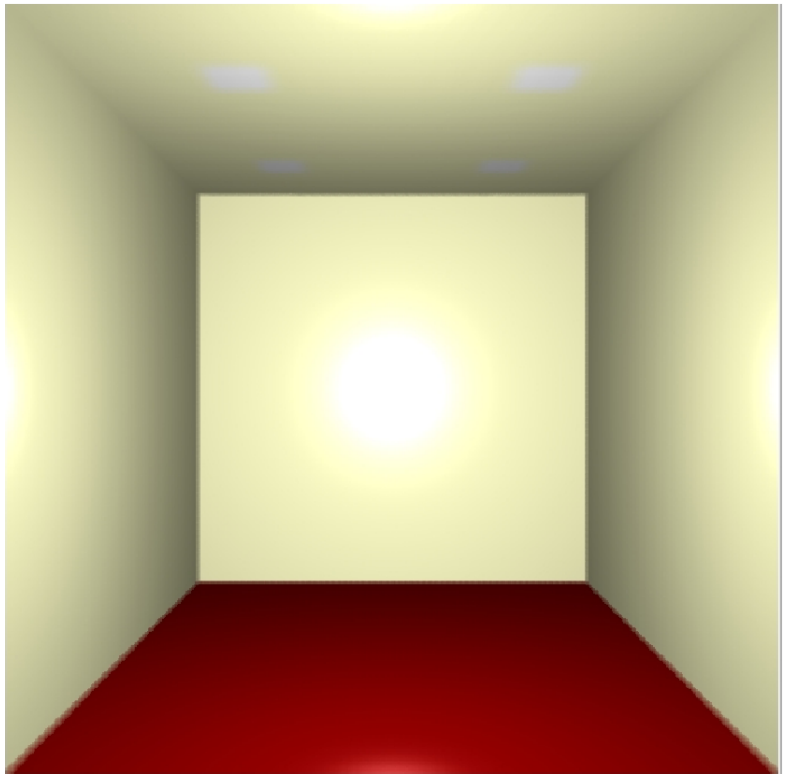


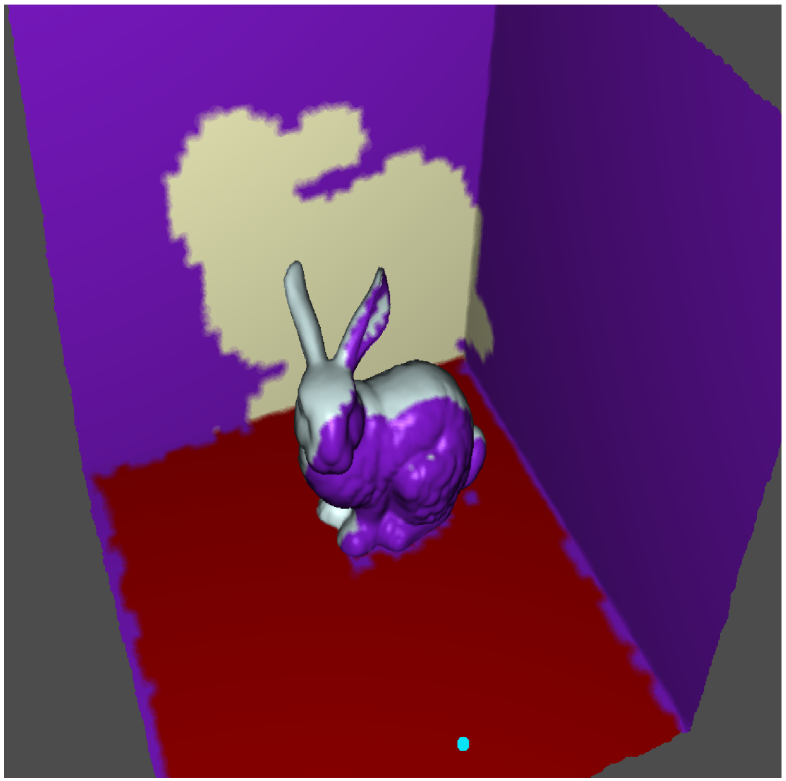


Qualitative Results: Visibility Correctness

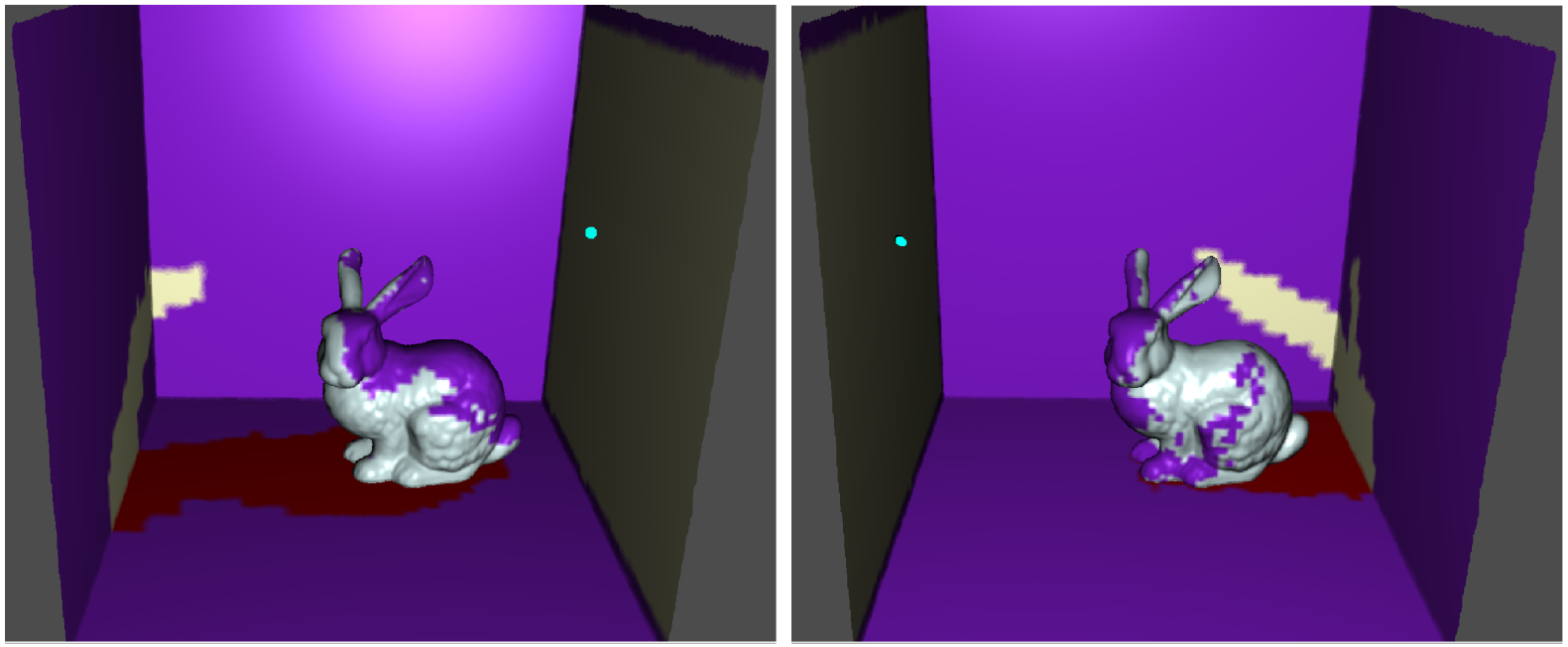


Qualitative Results: Visibility Correctness

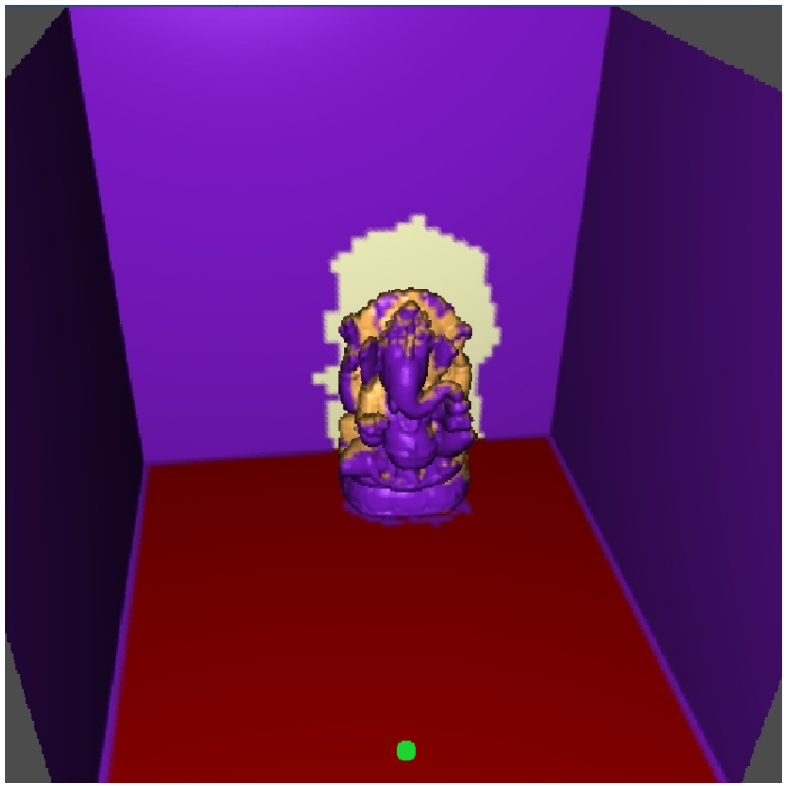


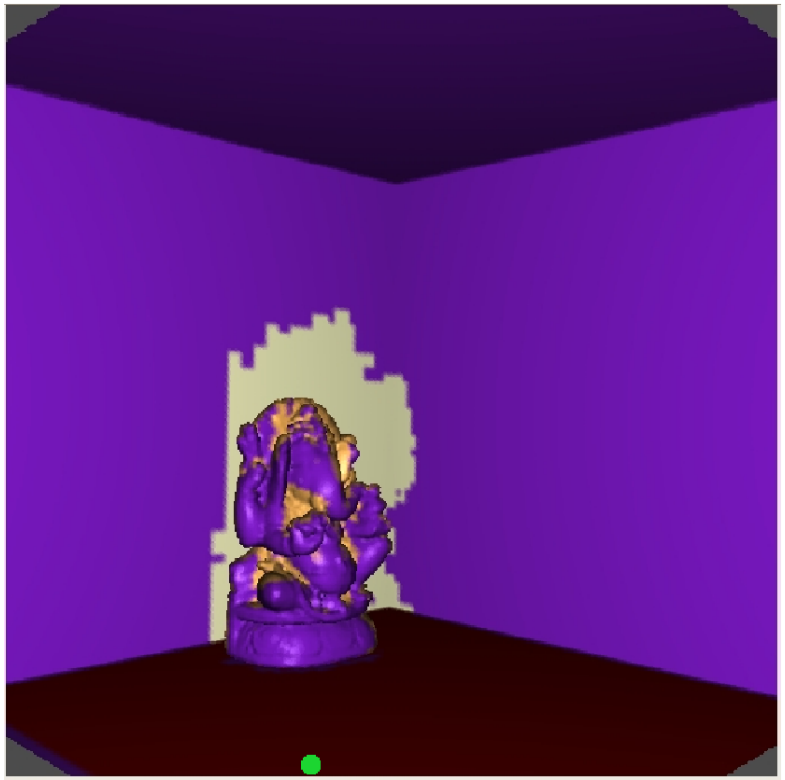


Qualitative Results: Visibility Correctness

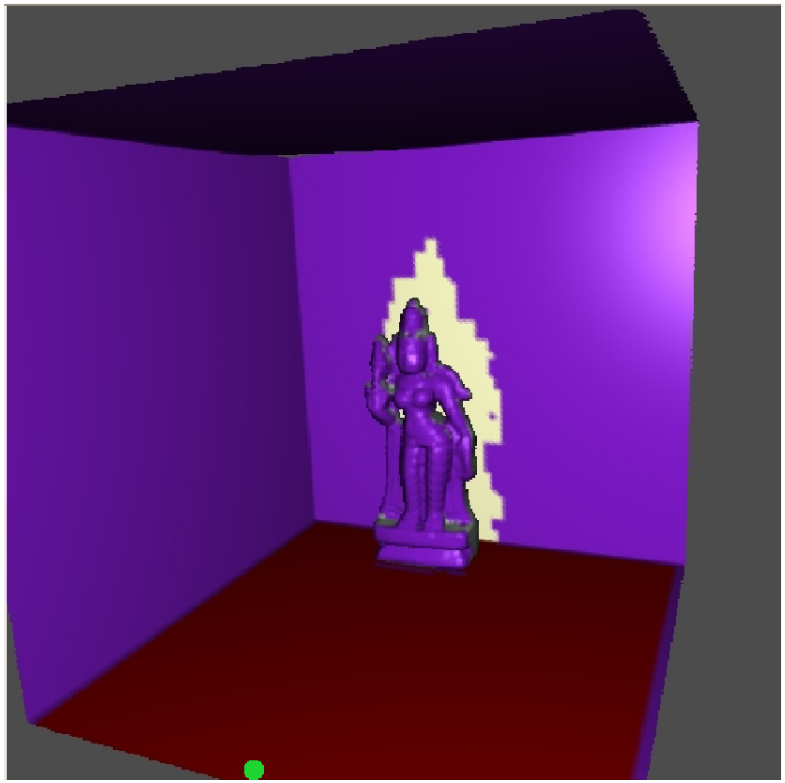


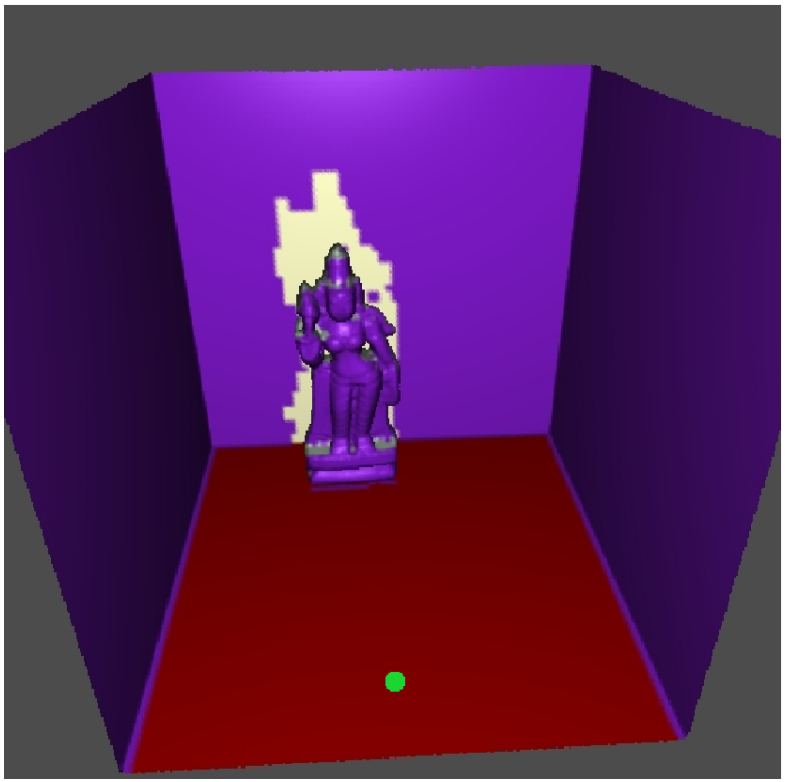
Qualitative Results: Visibility Correctness



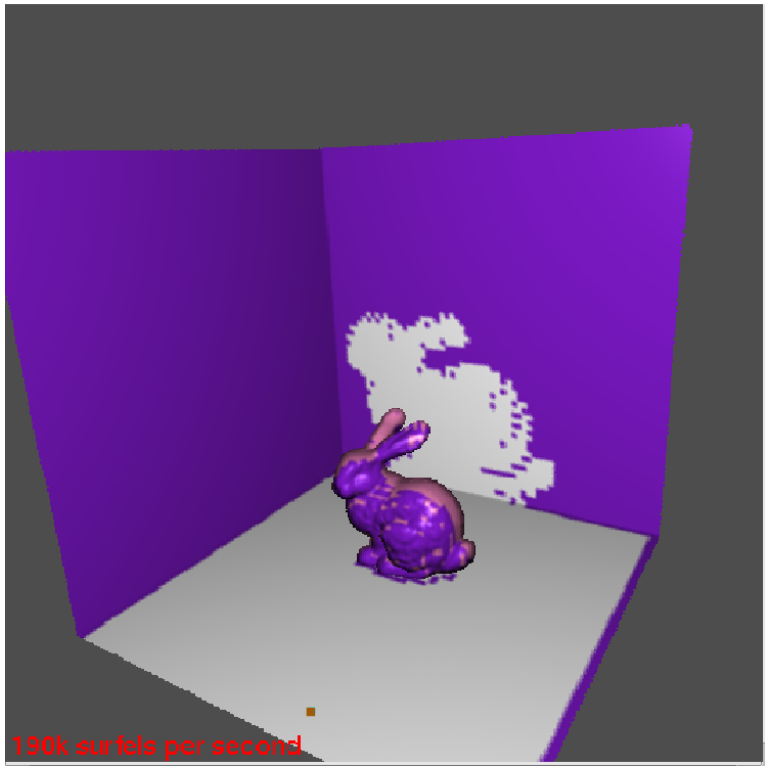


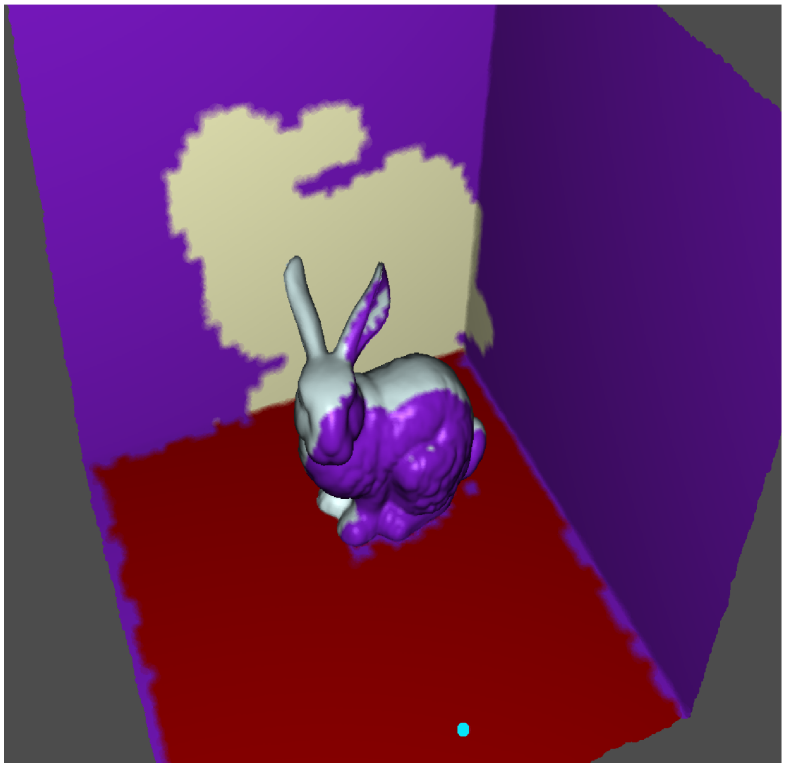
Qualitative Results: Visibility Correctness



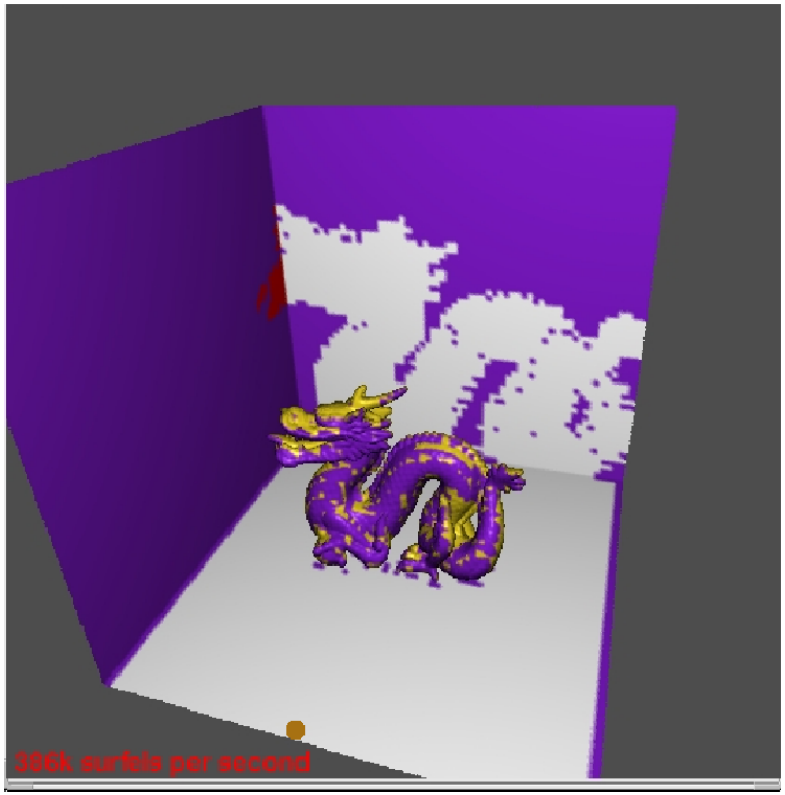


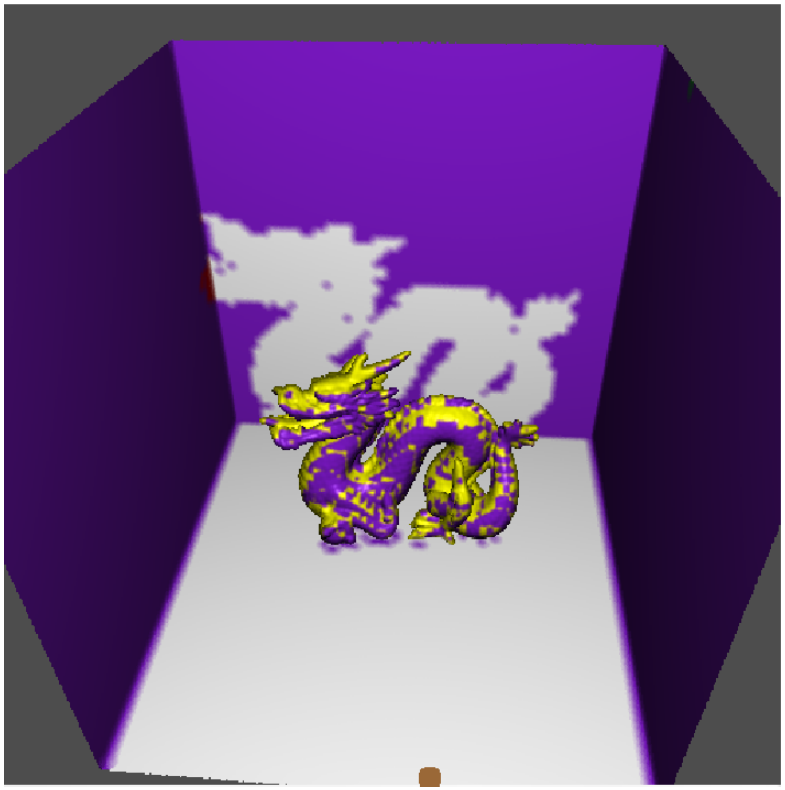
Qualitative Results: Comparisions



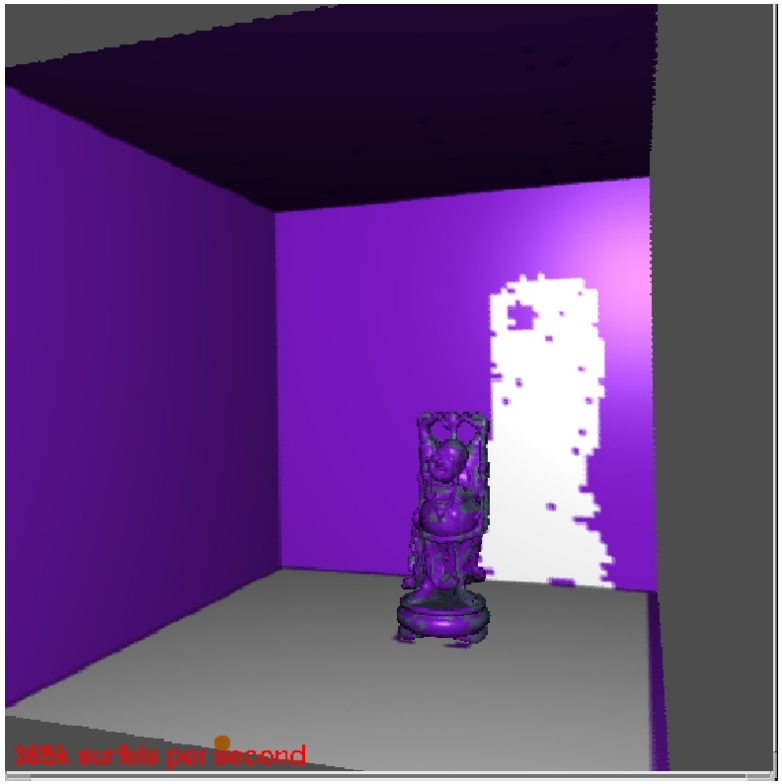


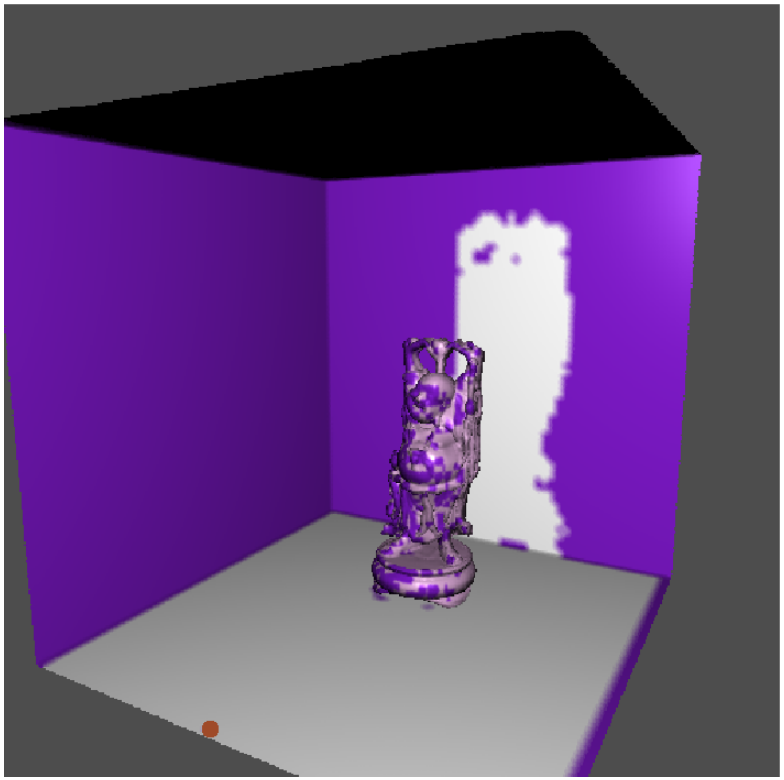
Qualitative Results: Comparisions



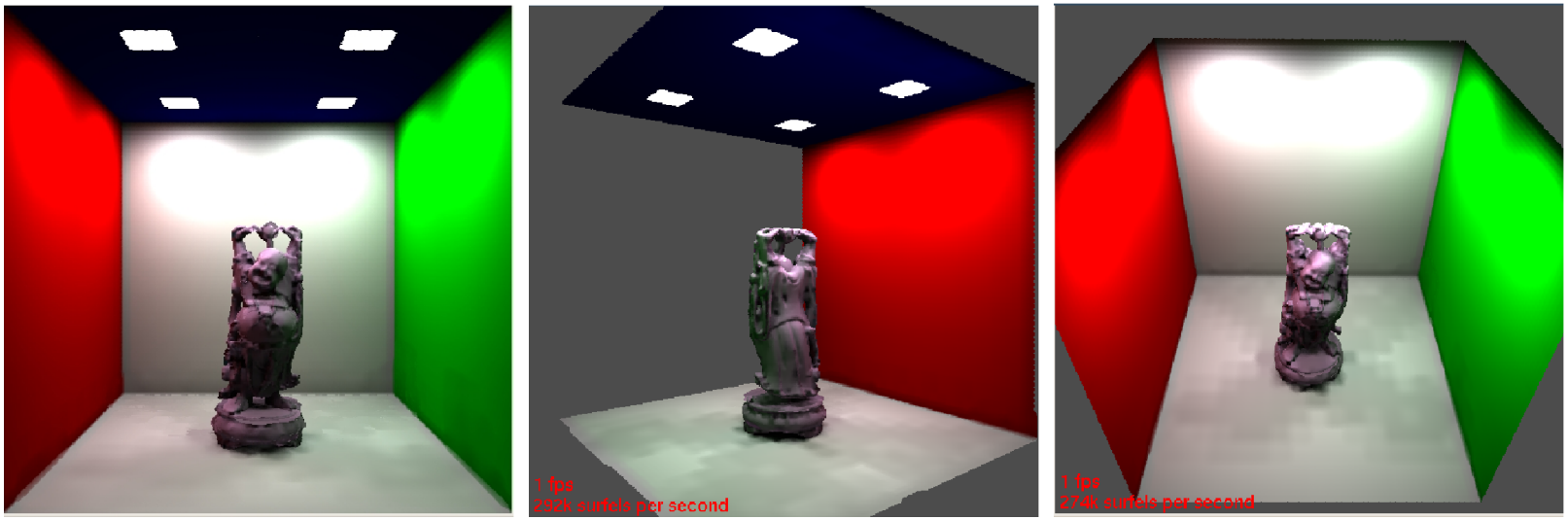


Qualitative Results: Comparisions

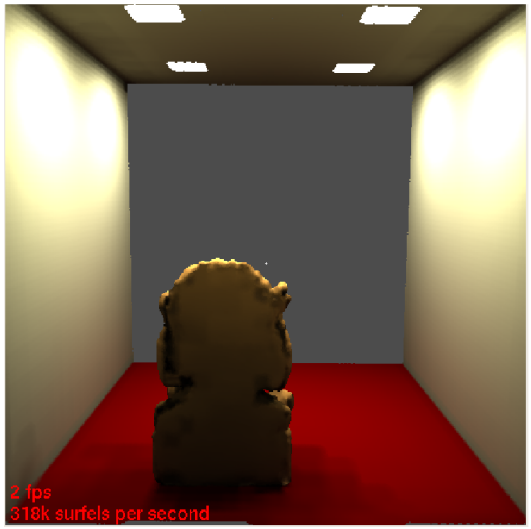




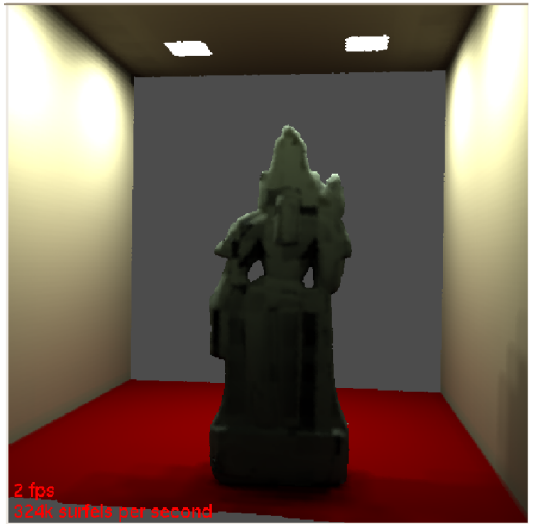
Qualitative Results: Global Illumination



Qualitative Results: Global Illumination



Qualitative Results: Global Illumination



Qualitative Results: Global Illumination

Quantitative Results

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 Satyavati in a Cornell room

Quantitative Results

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

Quantitative Results

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 Satyavati in a Cornell room

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 enables efficient answer to common rendering queries
- We have presented 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

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 enables efficient answer to common rendering queries
- We have presented 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

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 enables efficient answer to common rendering queries
- We have presented 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

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 enables efficient answer to common rendering queries
- We have presented 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

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 enables efficient answer to common rendering queries
- We have presented 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

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 enables efficient answer to common rendering queries
- We have presented 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

Thank You

That's it !