Binary Orientation Trees for Volume and Surface Reconstruction from Unoriented Point Clouds

Yi-Ling Chen[1] Bing-Yu Chen[2] Shang-Hong Lai[1] Tomoyuki Nishita[3]

[1]National Tsing Hua University, Taiwan
[2]National Taiwan University, Taiwan
[3]The University of Tokyo, Japan

Outline

- Introduction & motivations
- Related work
- Binary orientation trees
- Volume and surface reconstruction
- Discussions and conclusion

Motivations

- Hierarchical space partitioning structures are extensively exploited in various research fields.
  - Octrees,
  - K-d trees,
  - Binary space partitioning (BSP) trees.
- Partition the space to produce a collection of subsets of the data satisfying a given criterion.
  - Lacking of additional semantic information.

Introduction

- Orientation vs. Visibility
  - Basic idea: when observing a 3D model, the exterior region is visible while the interior region is occluded (invisible).
  - The **in/out** information is very helpful to determine the orientation w.r.t the 3D model.
- Binary orientation tree (BOT)
  - Hierarchical space partitioning structure (Octree-like)
  - Roughly splits the 3D space into inside/outside parts w.r.t. a 3D model. (*visually carve out* the exterior region)

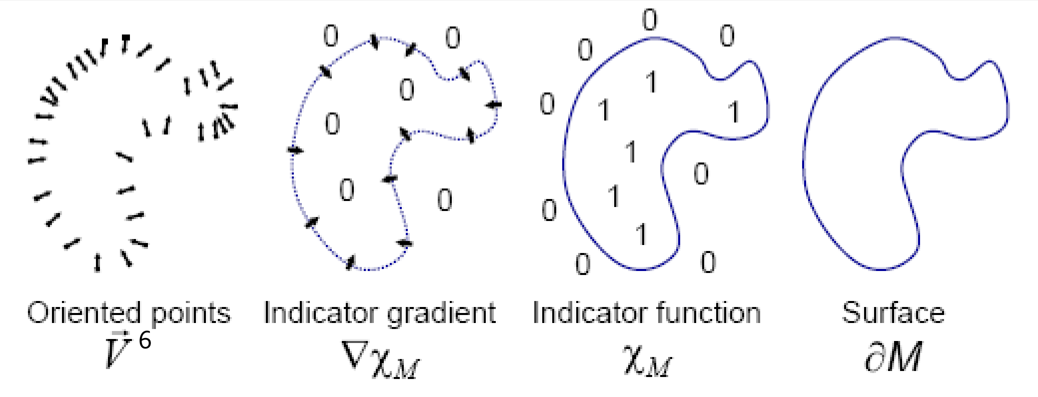
Related Work

- Surface reconstruction
  - Algebraic surface [Taubin‘91][Taubin‘93]
  - Level set methods [Zhao *et al*.‘00][Zhao *et al*.‘01]
  - Radial basis functions [Turk *et al*.‘99][Carr *et al*.‘01][Dinh *et al*.‘02]
  - Moving least-squares [Dey and Sun‘05][Lipman *et al*.’07][Kolluri ‘05]
  - Partition-of-unity based approaches
    - Octree [Ohtake *et al*.‘03][Xie *et al*.‘04][Gois *et al*.’08]
    - BSP tree [Tobor *et al*.‘04]
- And much more!
- Most of them require orientation information.

Related Work

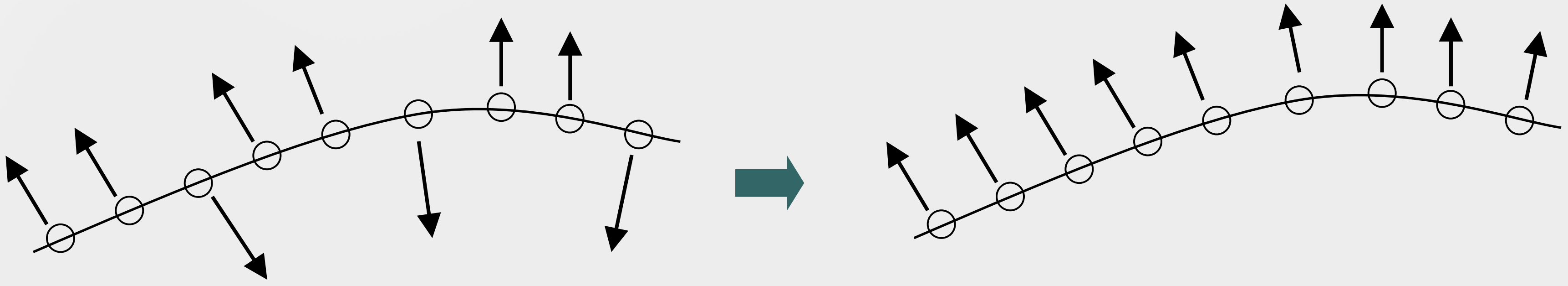
- To construct the *characteristic/indicator* function of a shape defined by the point samples. (one/inside and zero/outside)
  - Poisson equations [Kazhdan *et al.*'06]
  - FFT [Kazhdan '05]
  - Wavelets [Manson *et al.*'08]
  - Generalized eigenvalue problem [Alliez *et al.*'07]

Most of them require surface normals



Related Work

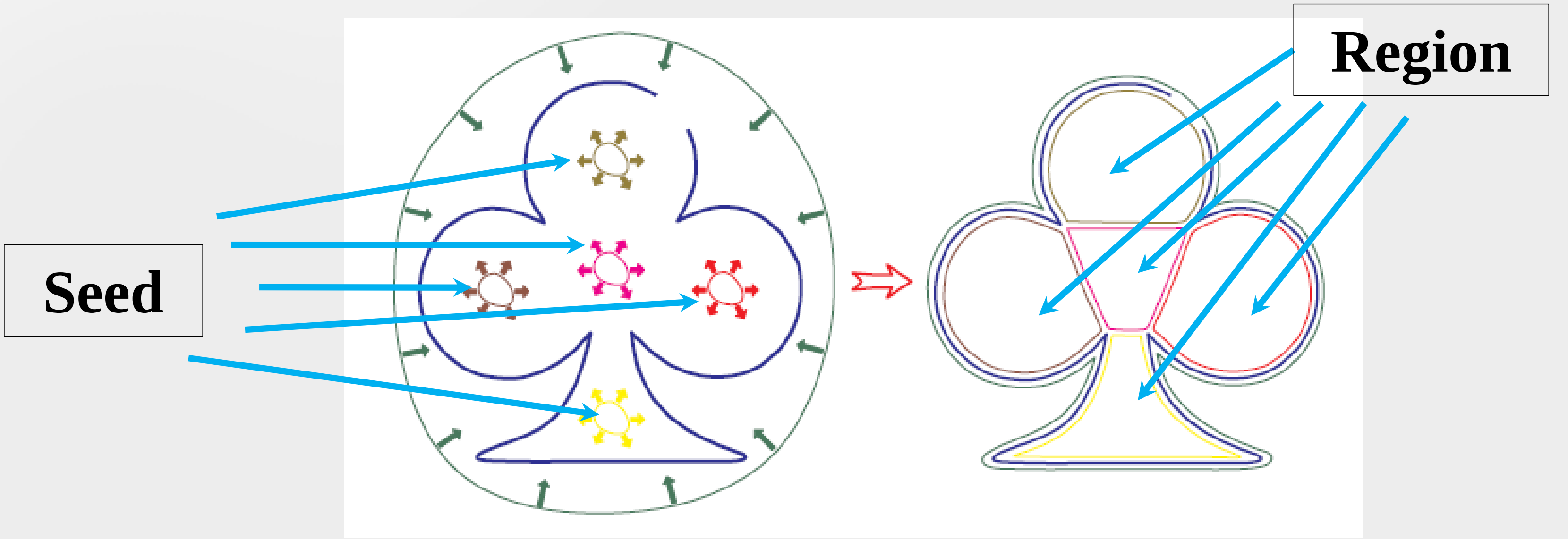
- Orientation propagation
  [Hoppe *et al.*'92][Xie *et al.*'03][Pauly *et al.*'03][Guennebaud *et al.*'07][Huang *et al.*'09]
  - Traversing a *minimal spanning tree* built over a point set.
  - Vulnerable against non-uniform sampling, sharp features or close-by surface patches.



Unoriented normal vectors **Oriented normal vectors**

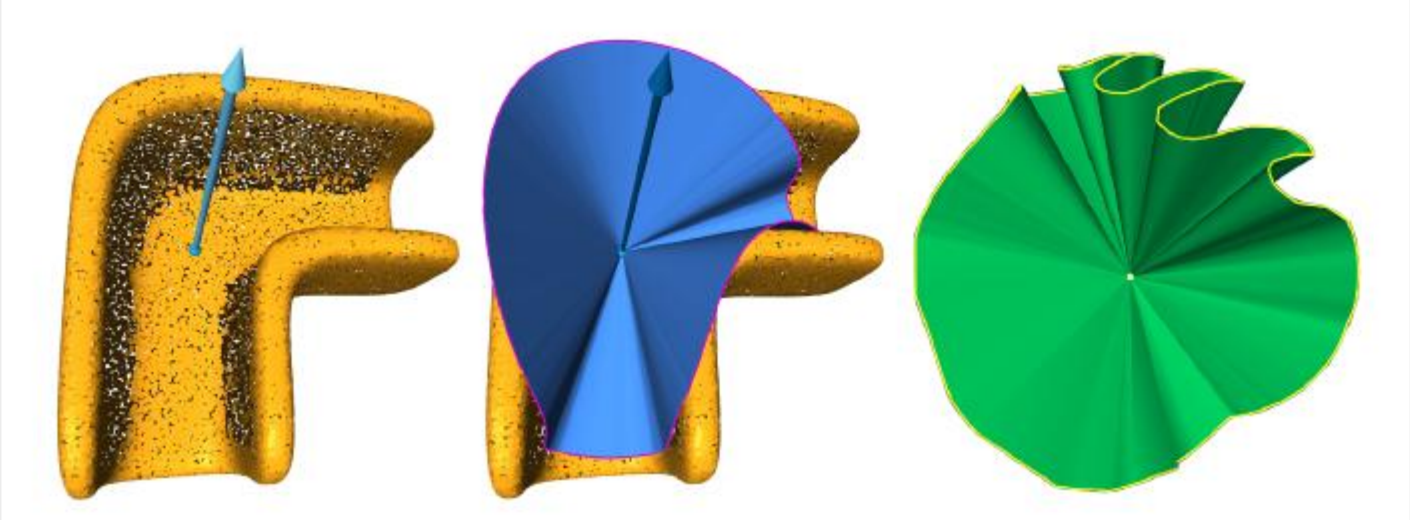
Related Work

- Active contour based method [Xie *et al.*’04]
  - Region-growing (in/out regions).
  - Voting for orientation determination.
  - Computationally expensive.



Related Work

- Cone Carving [Shalom *et al.*'10]
  - Create a visibility cone apexed at a sample point that extends beyond the outward direction to carve out the outside space.
  - Capable of dealing with missing data.
  - Computationally expensive.

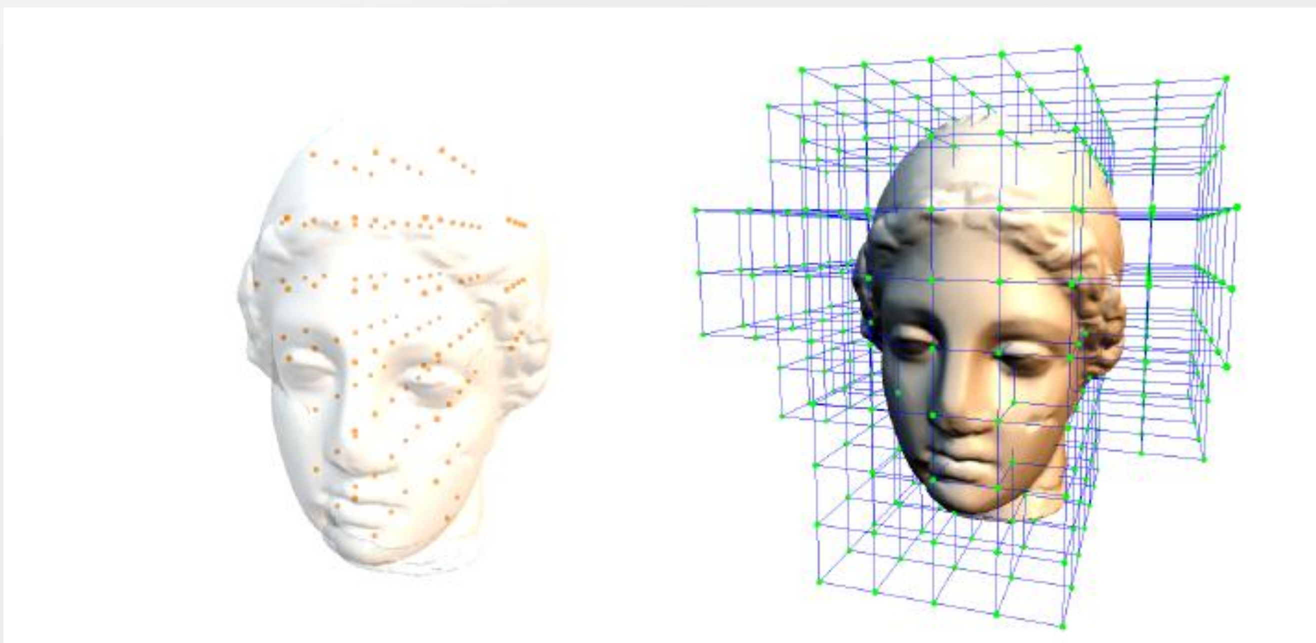


Binary Orientation Tree (BOT)

- A hierarchical data structure
  - Given a complete unoriented point set,
    - Octree-based space partitioning.
  - “Binary Orientation”?

g

ut point



Binary Orientation Tree (BOT)

- Basic idea
  - Points not belonging to the input point set are either "visible (out)" or "invisible (in)" when viewed from outside.
  - Directly obtain the tags without building the surface of the input point cloud by visibility check.

Hidden Point Removal operator.
Katz *et al*. Direct Visibility of Point Sets.
In *Proc. of SIGGRAPH 2007*.

Hidden Point Removal

- Hidden Point Removal (HPR) operator
  - determines the visible points in a point cloud as viewed from a given viewpoint.

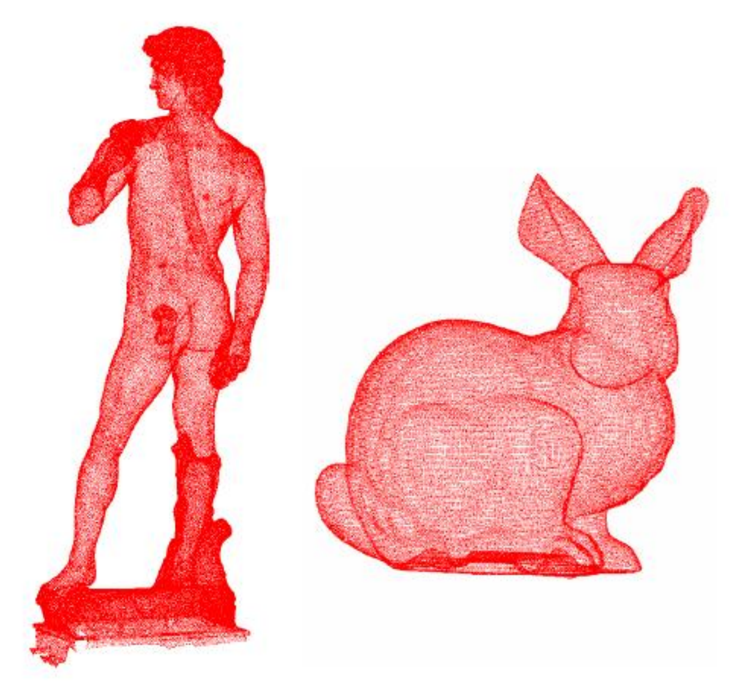


Figure 1: Input to the operator – Are the objects looking forwards or backwards?

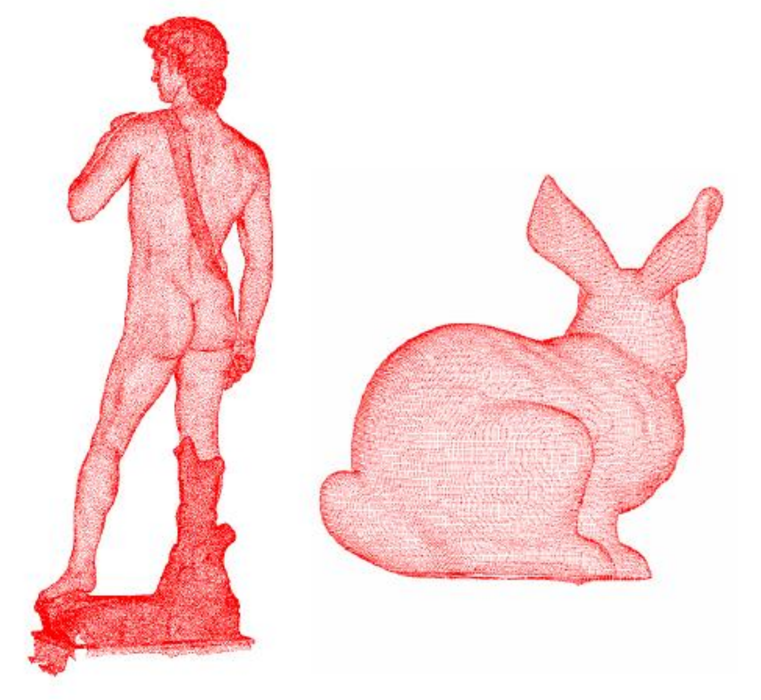


Figure 2: Output of the operator - They are heading backwards!

Hidden Point Removal

$$\widehat{p}_i = f(p_i) = p_i + 2(R - ||p_i||)\frac{p_i}{||p_i||}.$$

- Easy to compute
  - Transform the point cloud P to P’ by spherical flipping.
  - Compute convex hull of P’ and C (viewpoint)

HPR can not deal with *holes*, which disocclude the interior part of the point clouds.

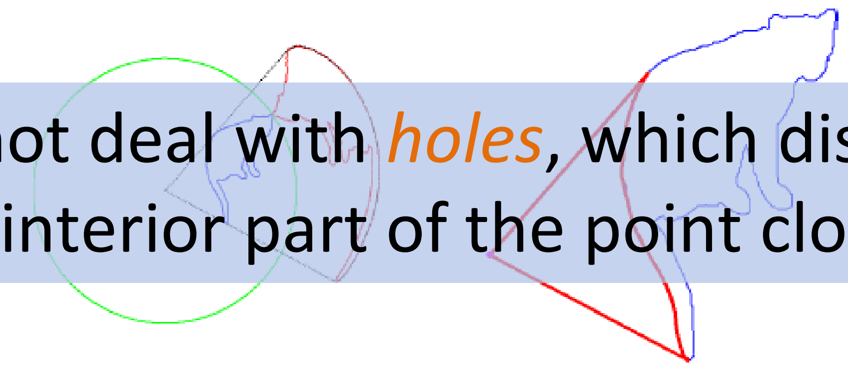


Figure 3: HPR Operator – Left: spherical flipping (in red) of a 2D curve (in blue) using a sphere (in green) centered at the view point (in magenta). Right: back projection of the convex hull. Note that this image is used only for illustration; in practice, R is much larger.

Definition 3.1 *A point $p_i \in P$ is marked visible from C if its inverted point $\widehat{p_i}$ lies on the convex hull of $\widehat{P} \bigcup \{C\}$.*

Building Binary Orientation Tree

- Building Binary Orientation Tree (Partitioning & Tagging)
  - Perform standard octree subdivision on the input point cloud.
  - Tagging of cell corners.
- Tagging (Growing & Carving)
  - Growing of **mono-oriented** region (from outside).
    - Start from the root cell with all corners tagged as **out**.
    - Propagate the tags to the connected empty cells.
  - Carving of **bi-oriented** regions
    - Determine the tags of bi-oriented cells (non-empty cells) by visibility check.

Building Binary Orientation Tree

- Growing
  - Recursive back-tracing

```
tag_tree(C)
begin
C' ← all leaves containing tagged corners of C;
if C is not leaf and not traced then
    for all cells C_leaf in C' do
        call back_tracing(C_leaf);
    end for
end if
if C is not leaf then
    for all subcells C_sub of C do
        call tag_tree(C_sub);
    end for
end if
end
```

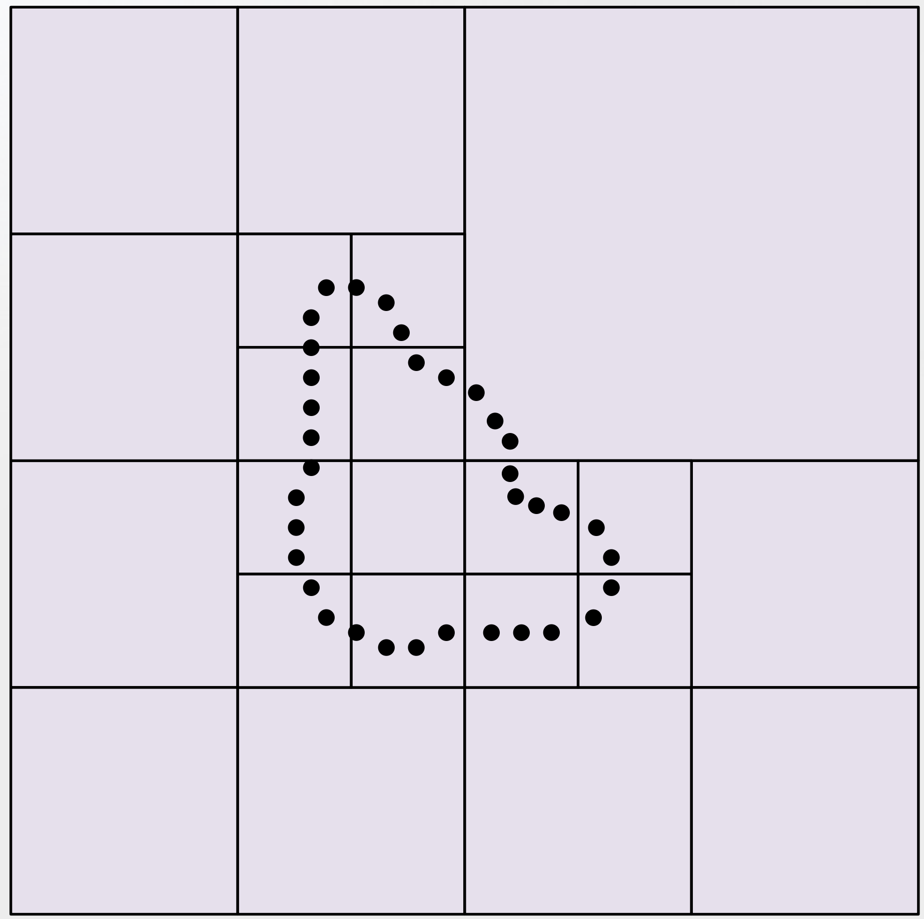
```
back_tracing(C)
begin
call tag_corners_if_empty(C);
if C is not leaf then
    set C as traced;
    for all subcell C_sub of C do
        call tag_corners_if_empty(C_sub);
    end for
end if
if C is not root then
    C_parent ← parent cell of C;
    call back_tracing(C_parent);
end if
end
```

Building Binary Orientation Tree

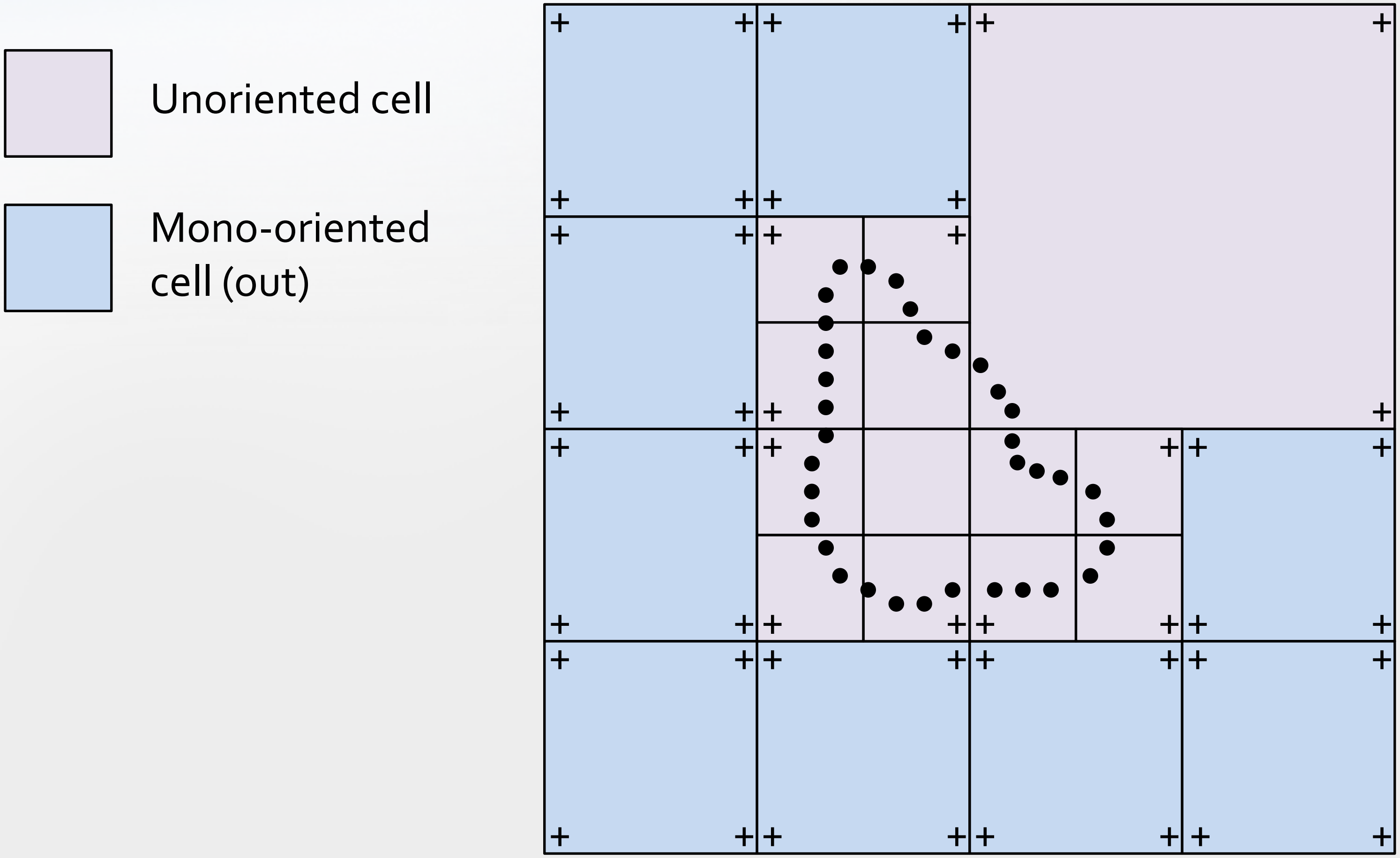
- Carving
  - Collect the input point set P and untagged corners P'.
  - Iteratively view (P and P') by HPR with various viewpoints.
  - "Carve out" the visible points among P' and tag them as **out**.
  - Terminate if no **out** points can be detected.
  - Tag the rest points in P' as **in** (ideally, they are always occluded by P).

Partitioning (Octree Subdivision)

Unoriented cell



Growing

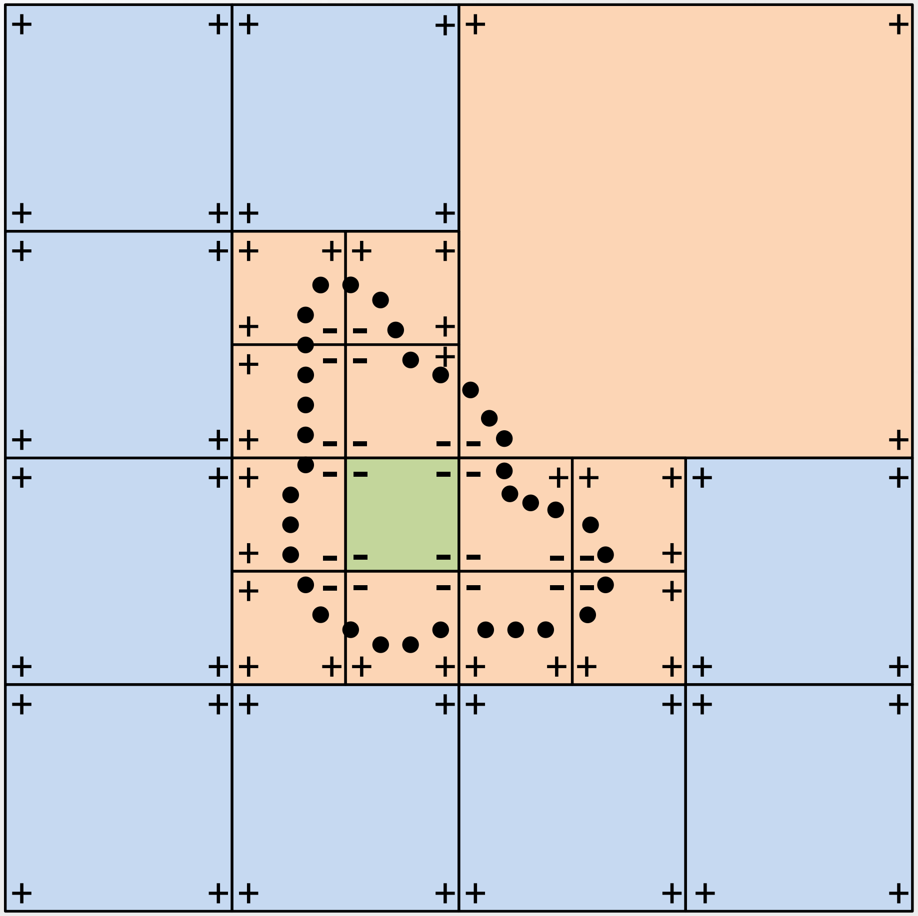


Carving

Bi-oriented cell

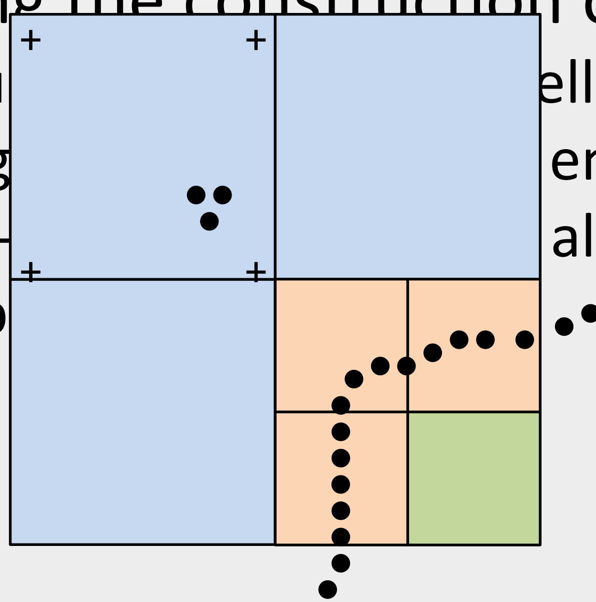
Mono-oriented cell (out)

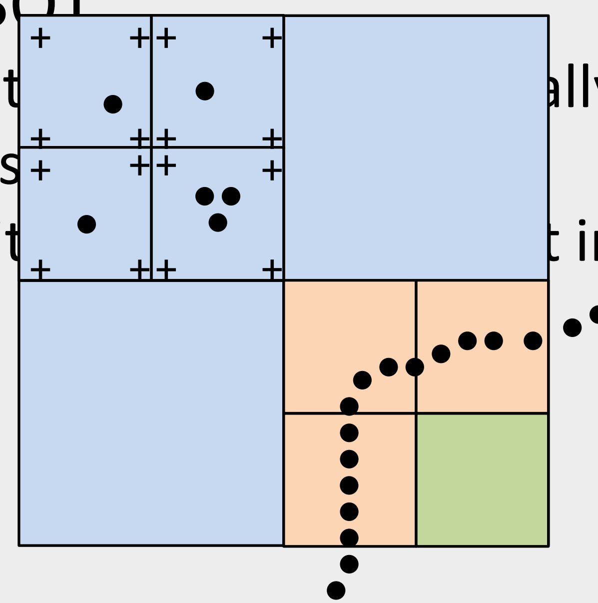
Mono-oriented cell (in)



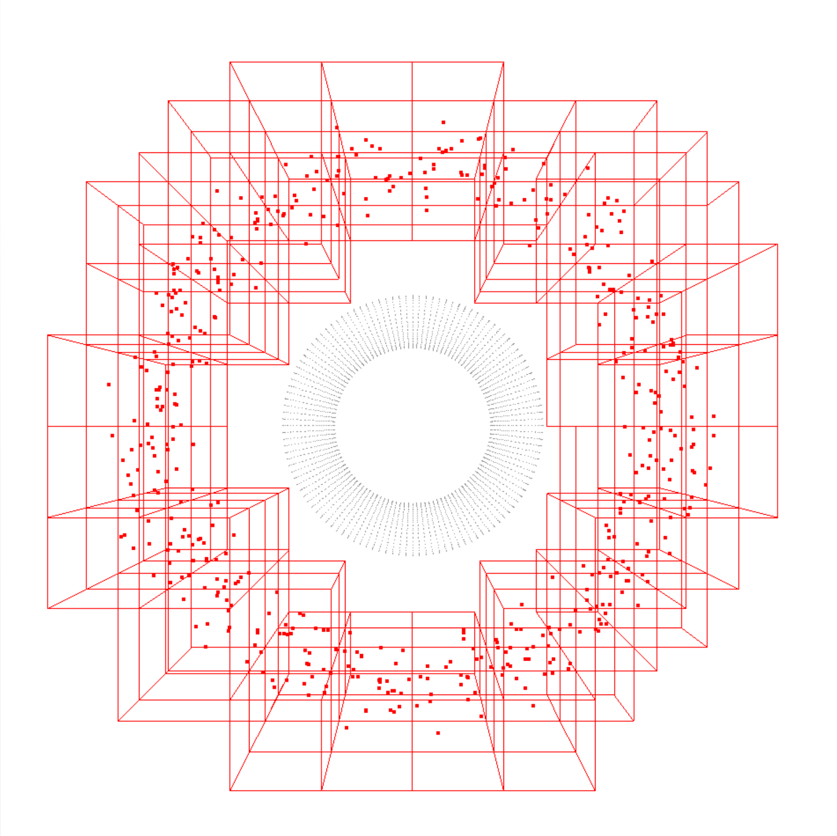
Outlier Detection

- Observation:
  - Outliers are sparse and disorderly distributed, and thus can hardly occlude the nearby corners.
  - The outliers will be "*enveloped*" in cells with all corners identically tagged (out).
- During the construction of BOT
  - Qu ... ells wit ... ally tag ... enclos
  - Re- ... algorit ... t in/out lab

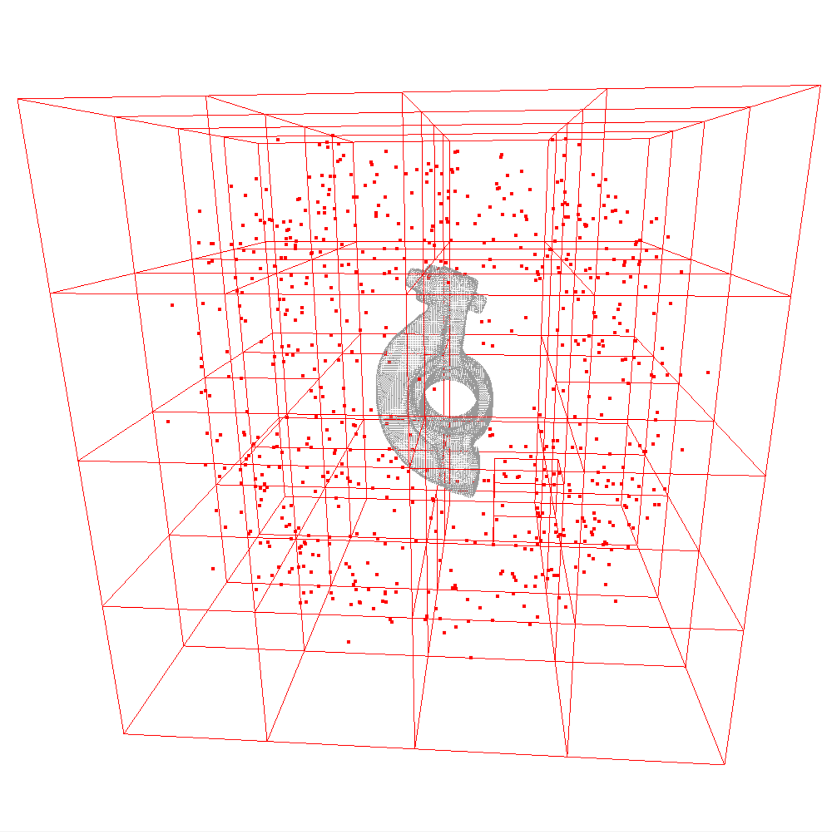




Outlier Detection

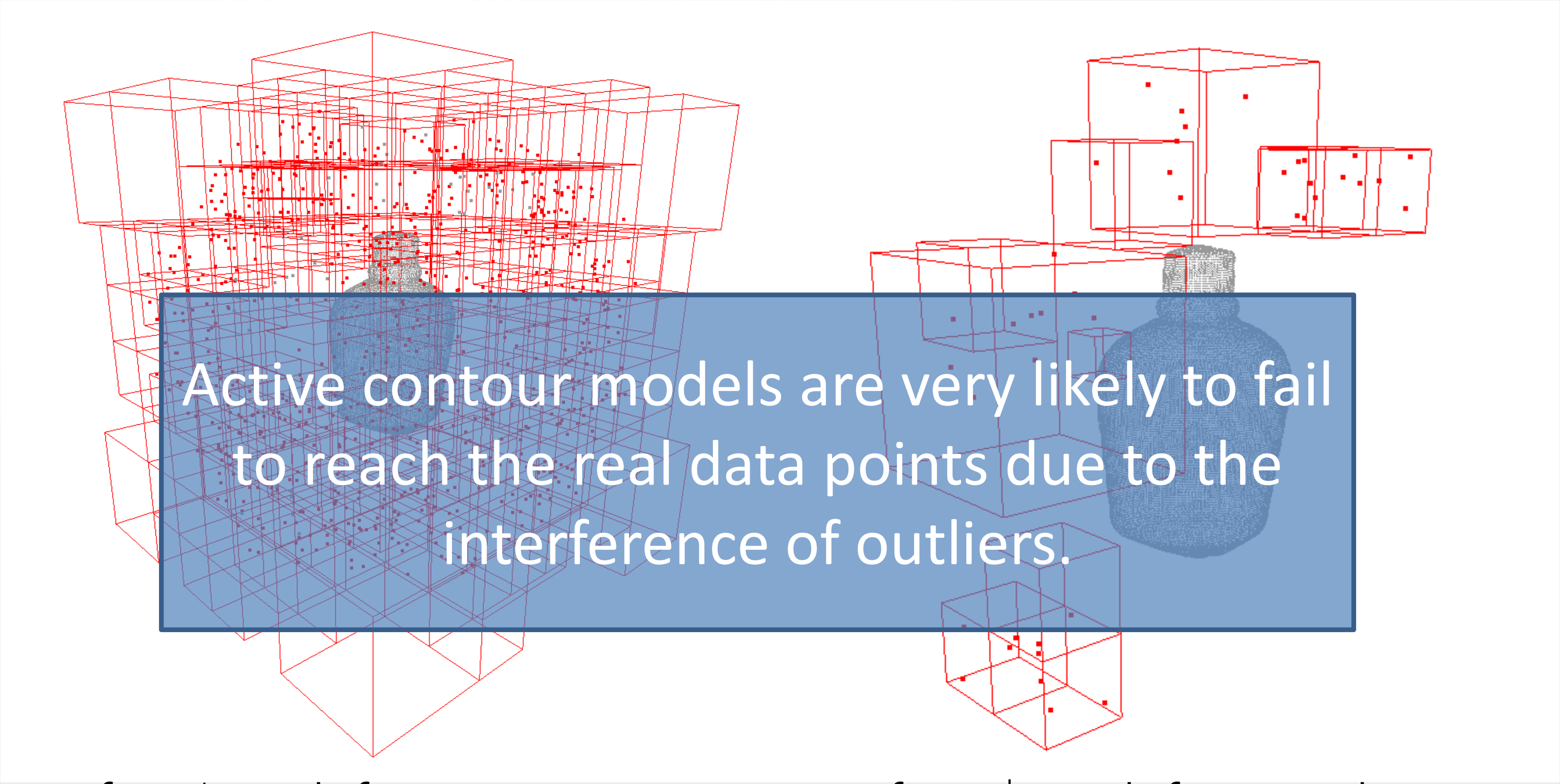


500 outliers



700 outliers

Outlier Detection



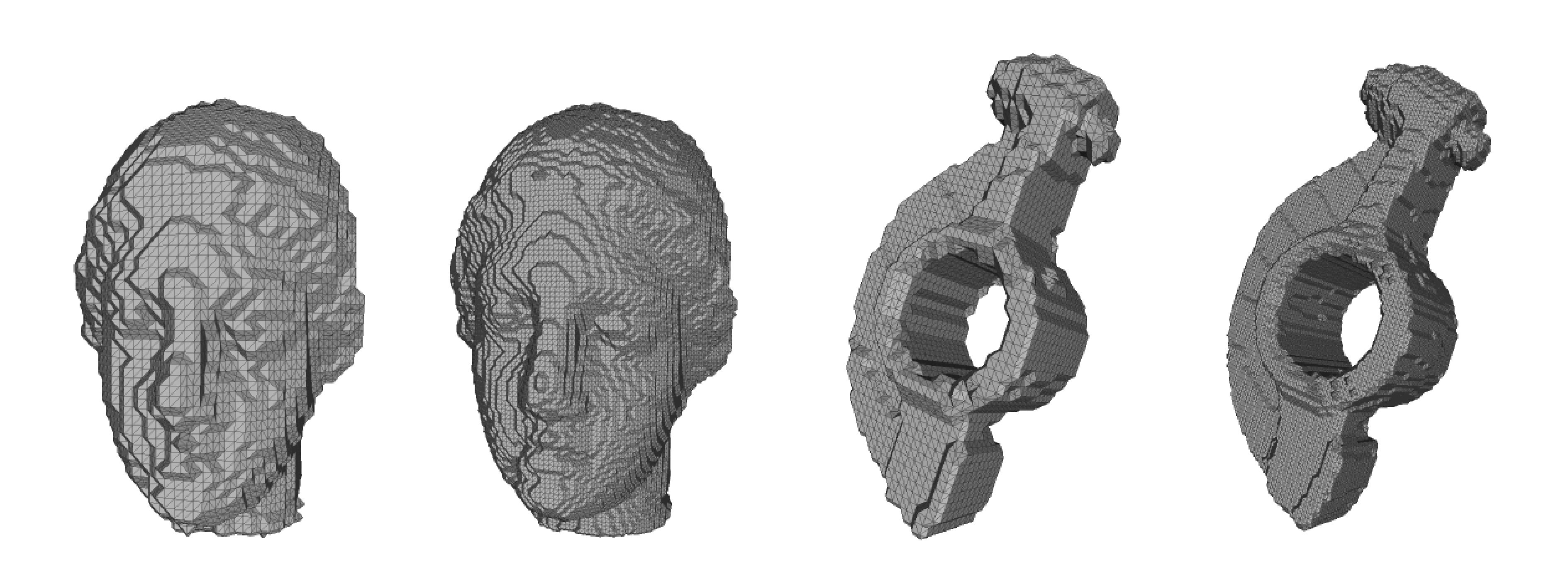
Active contour models are very likely to fail to reach the real data points due to the interference of outliers.

After 1st round of tagging, 759 outliers detected (total 800 outliers)

After 2nd round of tagging, the remaining 41 outliers are detected.

Volume Reconstruction

- Conceptually, the **in/out** information stored in a BOT is the same to the **volume data** also represented by octrees.



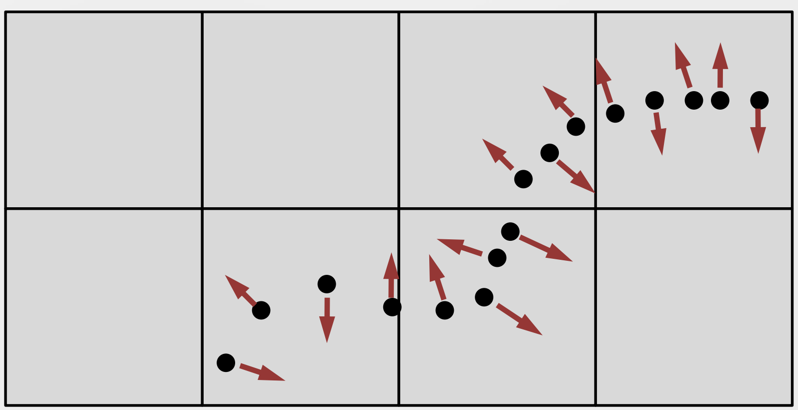
Surface Reconstruction

- Reconstruct MPU implicit surfaces [Ohtake *et al.*'03] from unoriented points.
  - Octree-based adaptive approximation.
  - Take advantage of the tagged BOT corners as auxiliary points to orientate the local implicit surfaces.

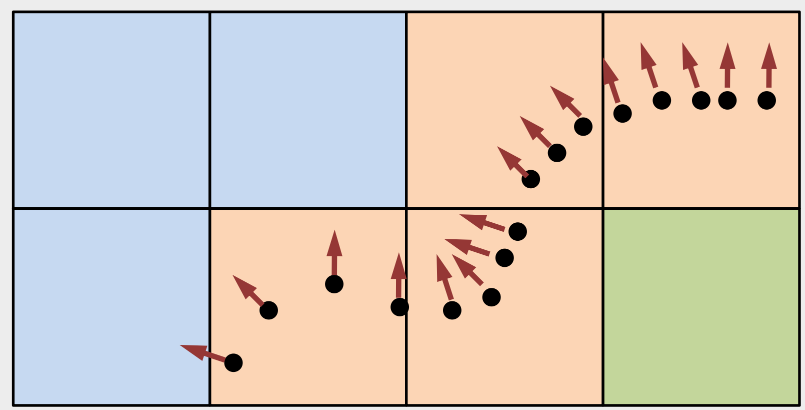
Reconstructed MPU implicit surfaces by BOTs

Orientation Determination

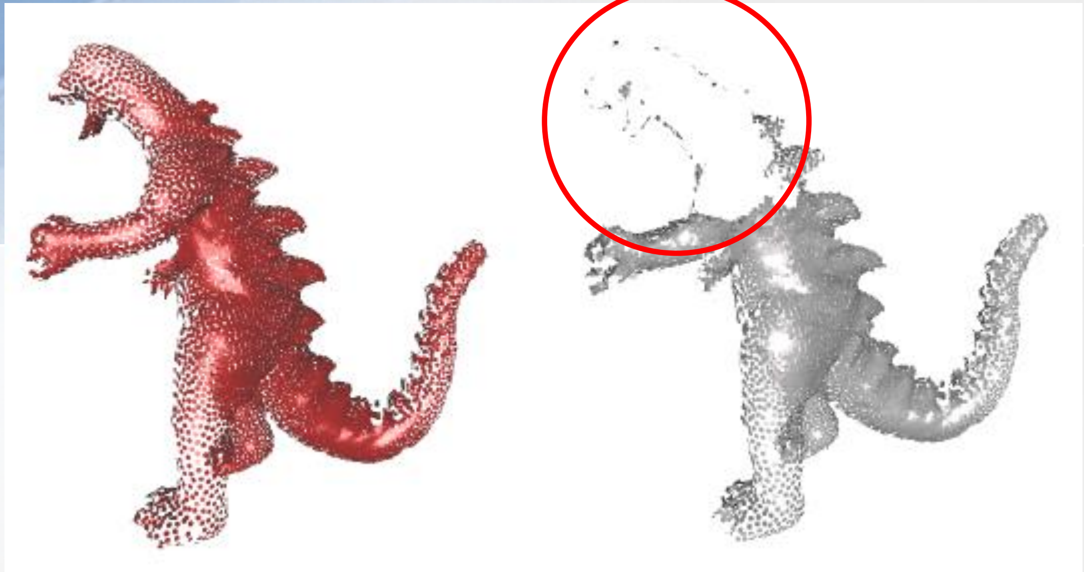
- Globally consistent orientation of normal fields
  - Traditional approaches (Minimal spanning tree)
    [Hoppe et al.'92][Guennebaud and Gross'07][Huang et al.'09]
  - Orientate the unoriented normal vectors by BOT tags.



Unoriented normal field

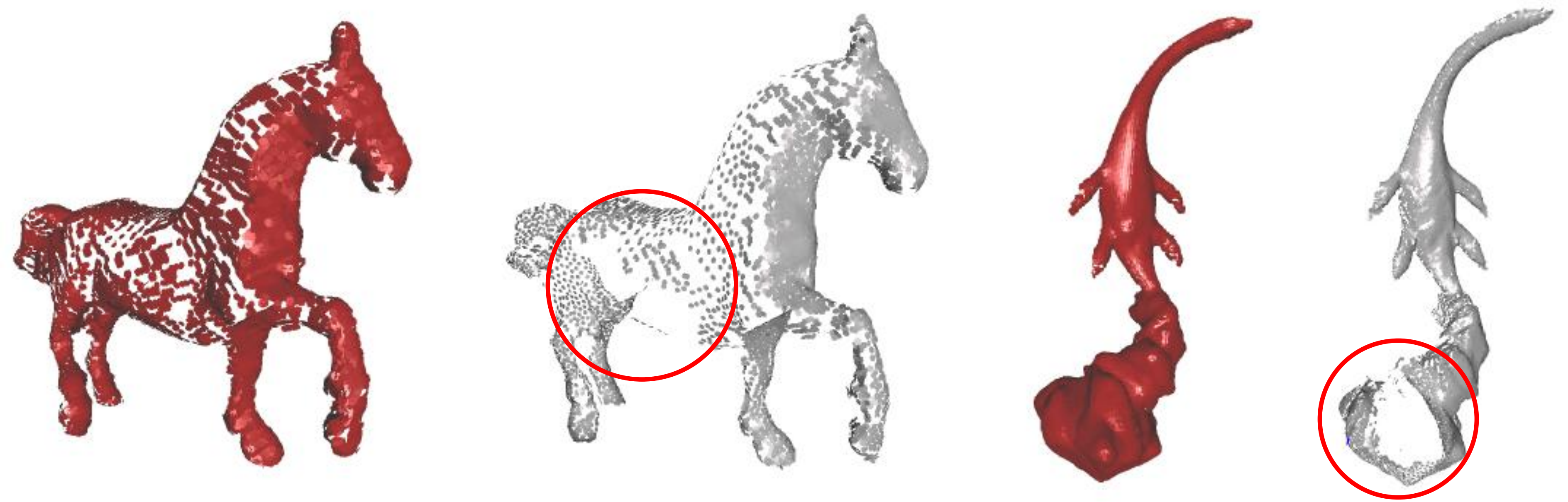


Oriented normal field by BOT



Globally consistent normal estimation

Splating with back-culling enabled.



Compared with [Huang et al. 2009]

Results

- Computation time
  - A subset of a dense point set is sufficient for visibility check.
  - Compute particles for visibility checks.

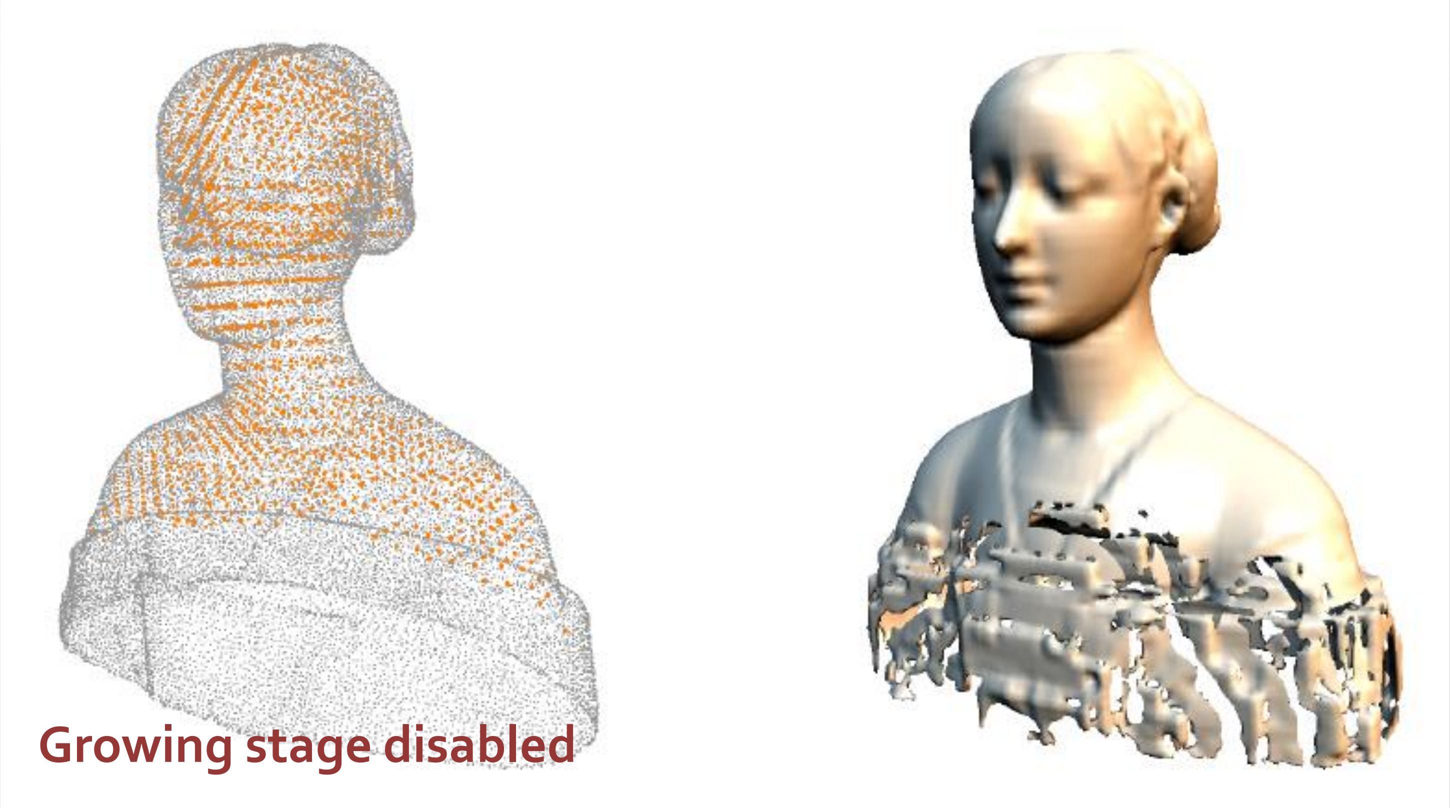
Data Sets	Point #	Particles #	Tagging	MPU
TORUS	4,800	3,591	0.188	1.844
DINOSAUR	36,988	22,301	2.891	1.985
RABBIT	67,038	17,408	1.391	3.406
SANTA	75,781	17,572	1.765	5.266
IGEA	134,345	32,940	2.547	6.828

(Represented in seconds)

Discussions

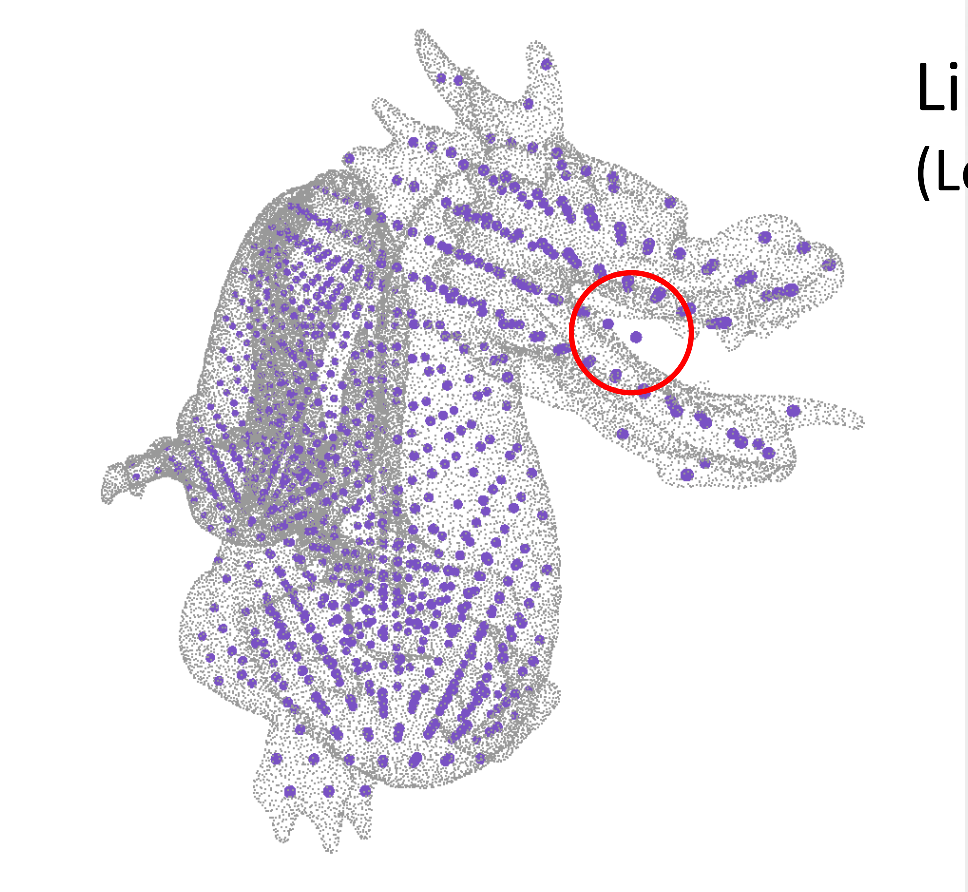
- Visual space carving does not resolve everything.

- Limitation 1: incomplete point sets



Discussions

- Visual space carving does not resolve everything.



Limitation 2: concave region
(Lemma 4.3 [Katz *et al.*'07])

Points on concave regions may not be correctly handled by HPR.
The local curvature must be sufficiently low.

Conclusion

- Binary Orientation Tree
  - Easy to implement
  - Efficient to compute
  - Useful for many geometric modeling and processing problems.
- Future directions
  - Handling of incomplete point sets.
  - More robust space carving (concave regions).
  - Embedding other useful metadata other than in/out tags.

Thank you for your attention!