

# A Volumetric Method for Building Complex Models from Range Images

**By Brian Curless and Marc Levoy**

**Stanford University**

# Introduction

## **Goal:**

Given a set of aligned, dense range images, we want to reconstruct a manifold that closely approximates the surface of the original model.

# Desirable Properties

- Representation of range uncertainty.
- Utilization of all range data (utilization of redundant data).
- Incremental and order independent updating.
- Time and Space efficiency.
- No restriction on topological type.
- Robustness (should properly deal with outliers and holes)
- Ability to fill holes in the reconstruction.

# Previous work

- From unorganized points
  - Parametric
    - Edelsbrunner92, Boisannat94
  - Implicit
    - Hoppe92, Bajaj95
- From range surfaces
  - Parametric
    - Turk94, Rutishauser94, Soucy95
  - Implicit
    - Grosso88, Succi90, Hilton96

# Volumetric Integration

- This Algorithm employs a continuous implicit function  $D(x)$ , represented by samples.
- This function is a weighted signed distance of each vertex point  $x$  to the nearest range surface along the line of sight to the sensor.

# Volumetric Integration

## Some Notations:

For a point  $x$ ,

1.  $d_k()$  : This represents the signed distance function for the  $k^{th}$  range image.
2.  $w_k()$  : This represents the weight function for the  $k^{th}$  range image.

**NOTE:** Both these functions take a 1D, 2D or 3D vertex point (vector) as the input and return a scalar outputs (i.e. Signed Distances in case of  $d_k()$  and weights in case of  $w_k()$ ).

E.g.

$d_3(x)$  represents the signed distance of point  $x$  in the 3<sup>rd</sup> image.

$w_2(x)$  represents the weight of point  $x$  in the 2<sup>nd</sup> image.

# Volumetric Integration

- Due to various practical errors. we might end up getting different distance values (or depth maps) despite capturing the range image from the same orientation multiple times.
- Hence, we try to find an accurate distance value reading using least squared fitting over the depth values of a point from different range images. This results in getting the least squared fitting as the mean of all the depth values of the corresponding point in all the depth images.

$$D(x) = \frac{\sum_{i=1}^n d_i(x)}{n}$$

# Volumetric Integration

The obtained depth values might not always be reliable, when are these values reliable?

1. The difference in depth values between multiple range images is small if the target object lies within the uncertainty depth range of the sensor. Therefore, this is different for different sensors.
2. The depth values are not reliable if the line of sight is at grazing angles to the surface at the point of interest.
3. The depth values are not reliable at the boundaries of captured surface.



# Volumetric Integration

- To resolve these issues, we introduce **weights** along with the signed distance values to every point.
- Therefore, this results in a simple weighted average of SDF's as,

$$D(x) = \frac{\sum_{i=1}^n w_i(x) d_i(x)}{\sum_{i=1}^n w_i(x)}$$

- This is the required implicit function which was mentioned earlier. The zero crossing of this function describes the best guess for the actual depth corresponding to point  $x$ .
- The locus of points satisfying,  $D(x) = 0$  describes the estimated surface of the target object as an implicit function.

# Volumetric Integration

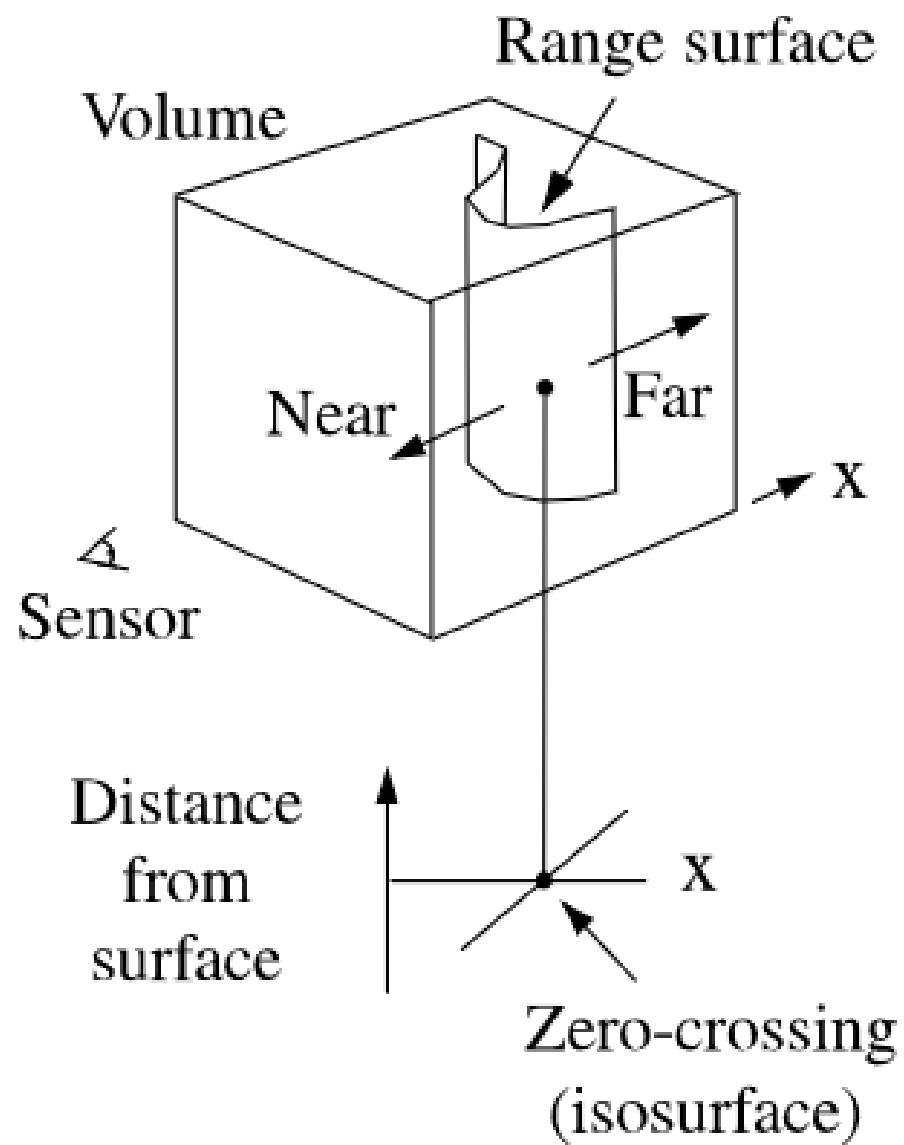
- We also define a Cumulative Weight function as,

$$W(x) = \sum_{i=1}^n w_i(x)$$

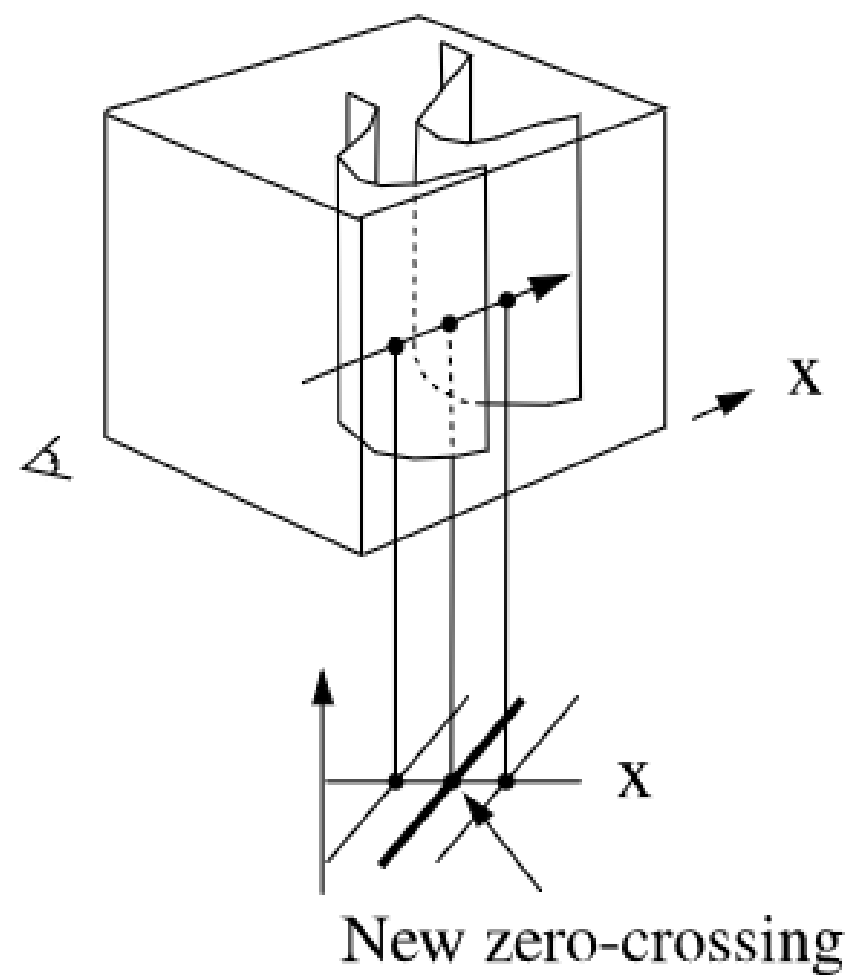
- For incremental calculations, we can define the update rules as follows,

$$D_{i+1}(\mathbf{x}) = \frac{W_i(\mathbf{x})D_i(\mathbf{x}) + w_{i+1}(\mathbf{x})d_{i+1}(\mathbf{x})}{W_i(\mathbf{x}) + w_{i+1}(\mathbf{x})}$$

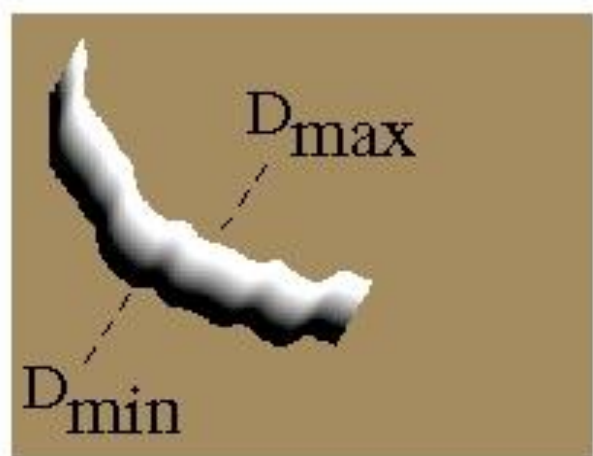
$$W_{i+1}(\mathbf{x}) = W_i(\mathbf{x}) + w_{i+1}(\mathbf{x})$$



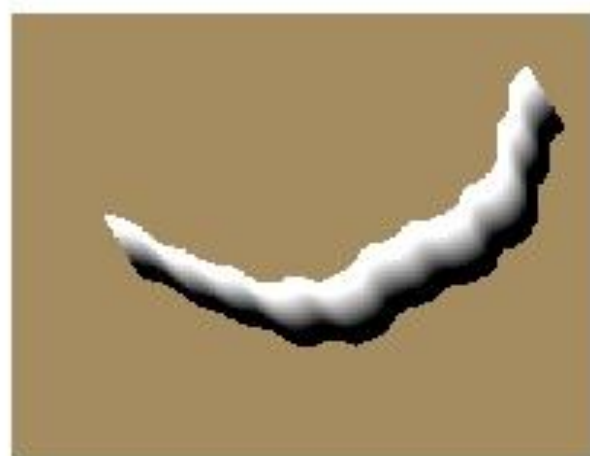
(a)



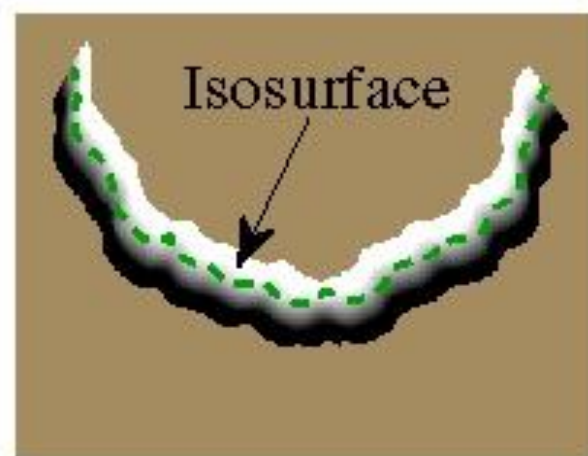
(b)



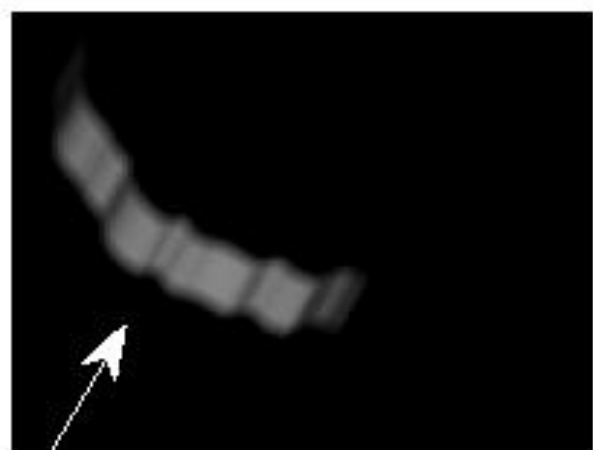
(a)



(b)

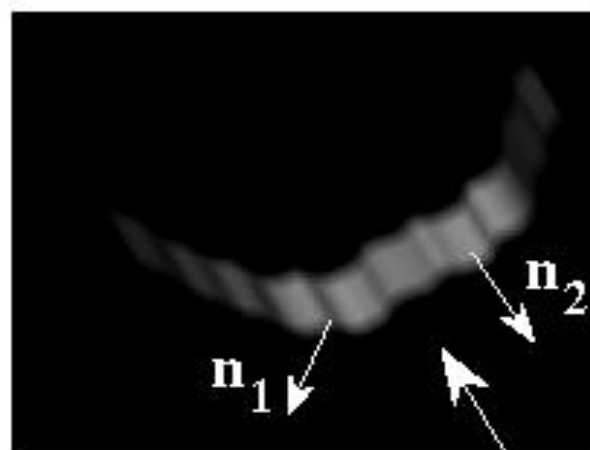


(c)



Sensor

(d)



Sensor

(e)



(f)

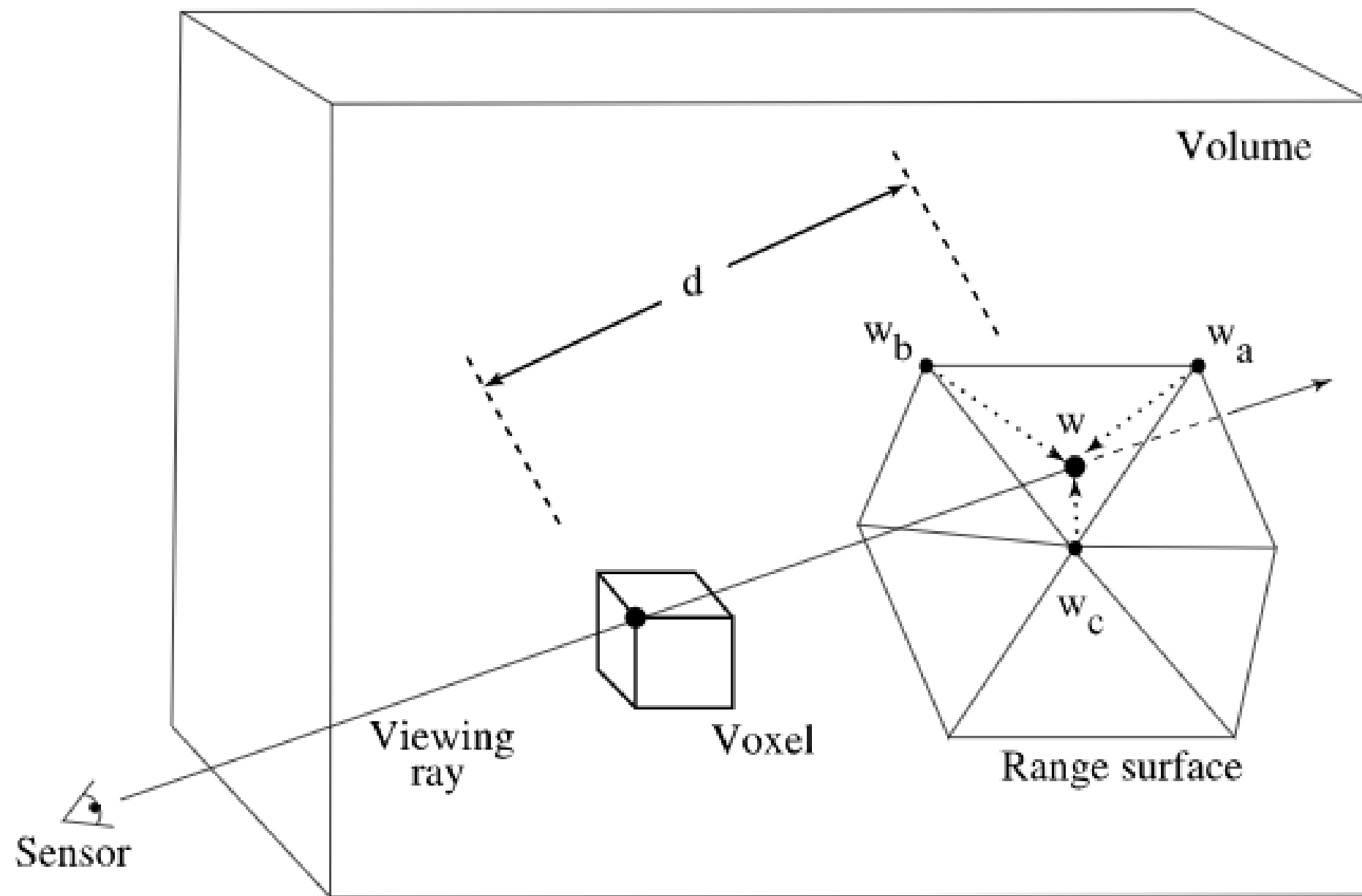
# Volumetric Integration

- Digitally space consists of voxel grids. How are these updates performed in voxel space ?

Solution:

- Tessellation
- Interpolation (for each voxel near the surface)

(Explain...)



# Hole Filling

- We have presented an algorithm that reconstructs the observed surface. Unseen portions appear as holes in the reconstruction.
- A hole-free mesh is useful for:
  - Fitting surfaces to meshes
  - Manufacturing complete models
  - Aesthetic renderings

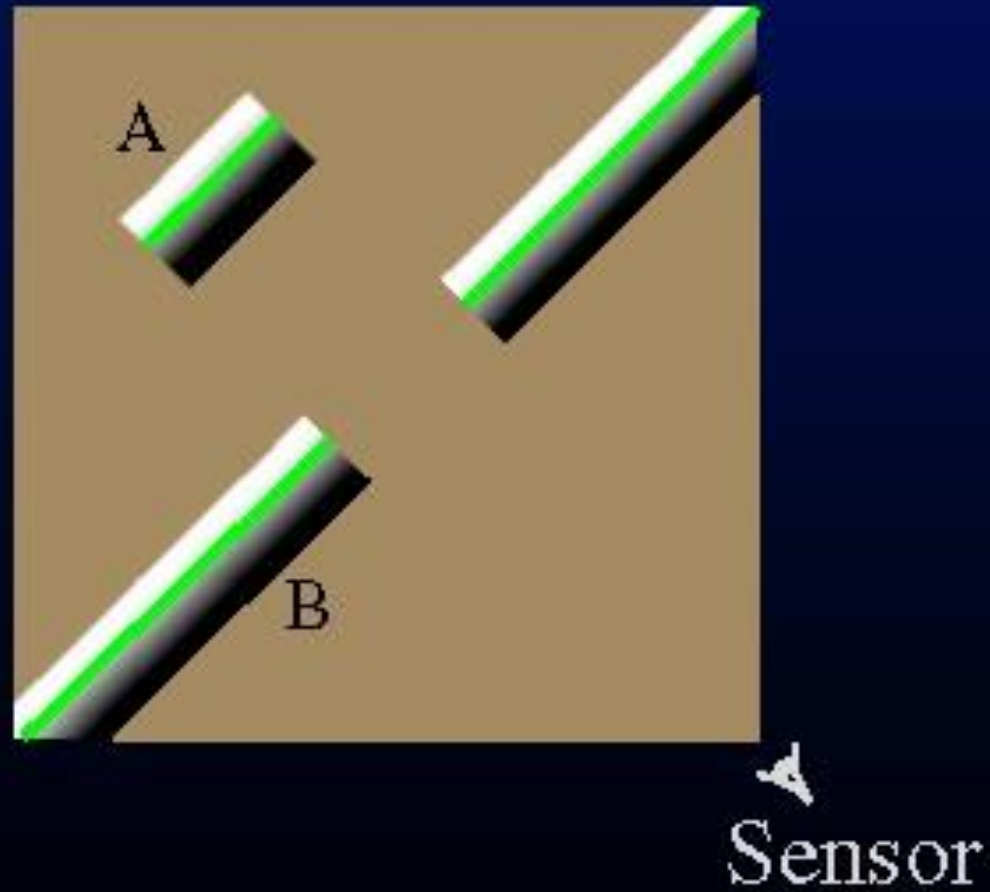
# Hole Filling

- One of the ways to fill the holes is to simply fill the holes with planer surfaces, but this sort of hole filling misses out on preserving the volumetric information of the surface.
- Therefore, it is preferred to follow a hole filling approach which operates on volume entirely. In this method, the Hole Filling is performed using space carving.



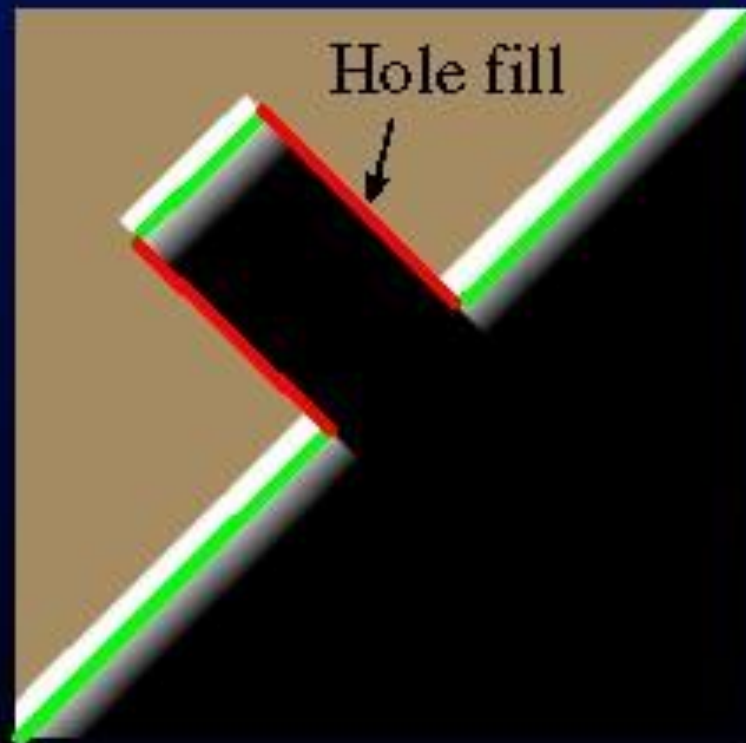
## *Without* space carving




---



## *With space carving*

---



-  Unseen
-  Empty
-  Near surface

  
Sensor

# Hole Filling

## Algorithm:

1. Initialize the voxel space to the “unseen” state.
2. Update the voxels near the surface as described in the previous section. As before, these voxels take on continuous signed distance and weight values
3. Follow the lines of sight back from the observed surface and mark the corresponding voxels as “empty”. We refer to this step as space carving.
4. Perform an iso-surface extraction at the zero-crossing of the signed distance function. Additionally, extract a surface between regions seen to be empty and regions that remain unseen.

# Carving *without* a backdrop

---

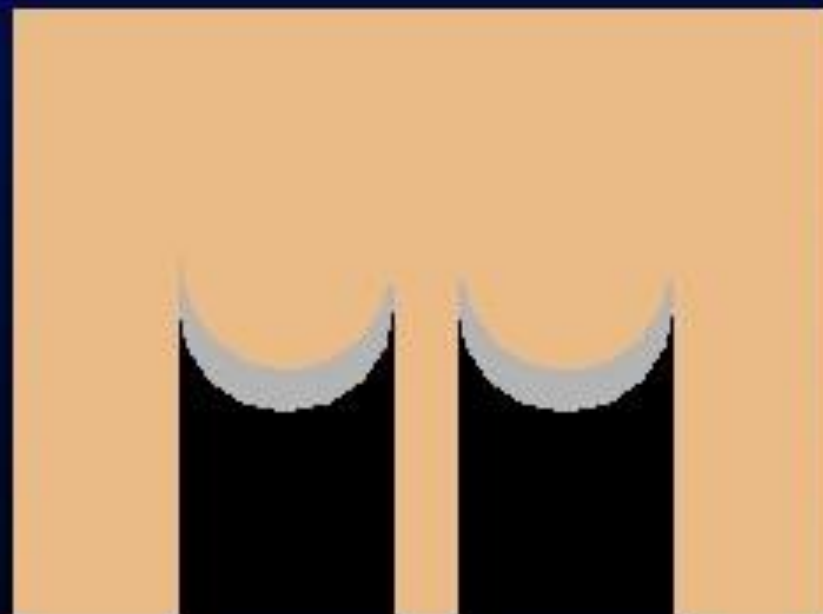
Scanning scenario

Surfaces



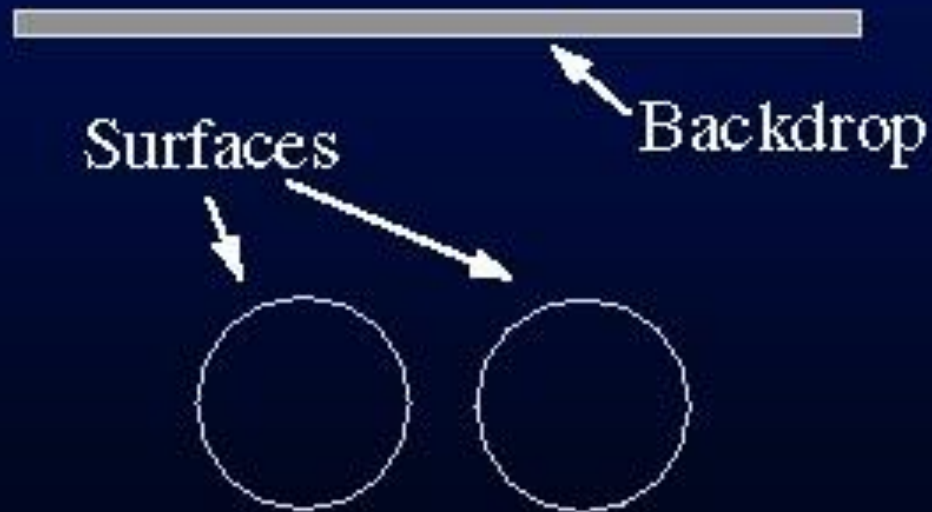
▽ Sensor

Volumetric slice



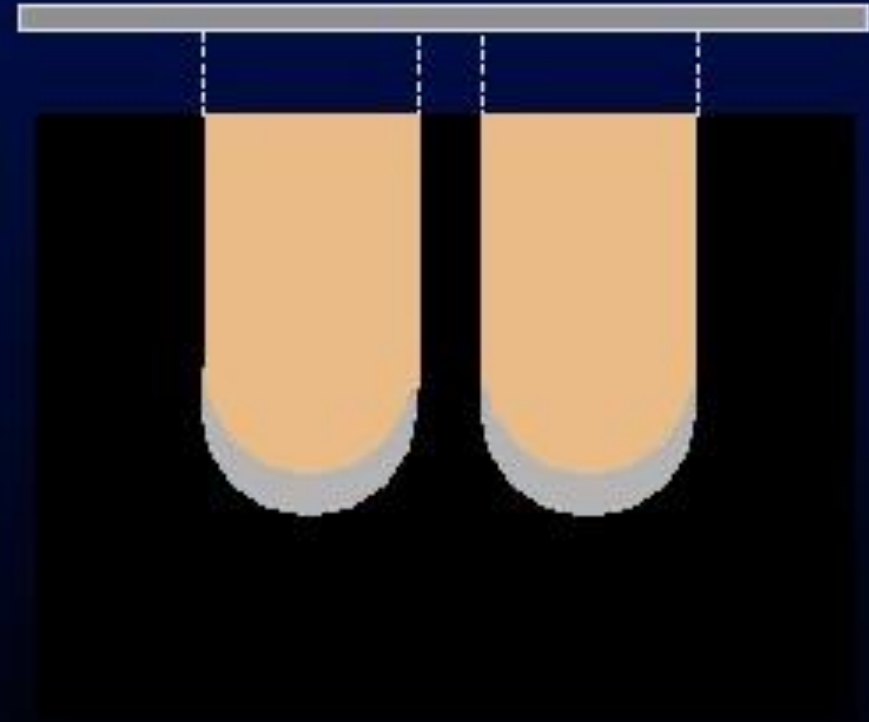
# Carving *with* a backdrop

Scanning scenario

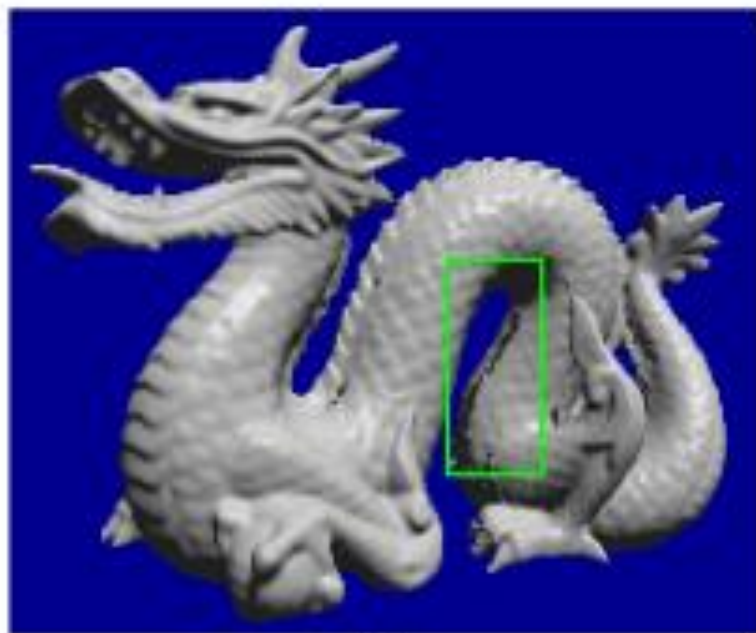


▽ Sensor

Volumetric slice



▽



(a)



(b)

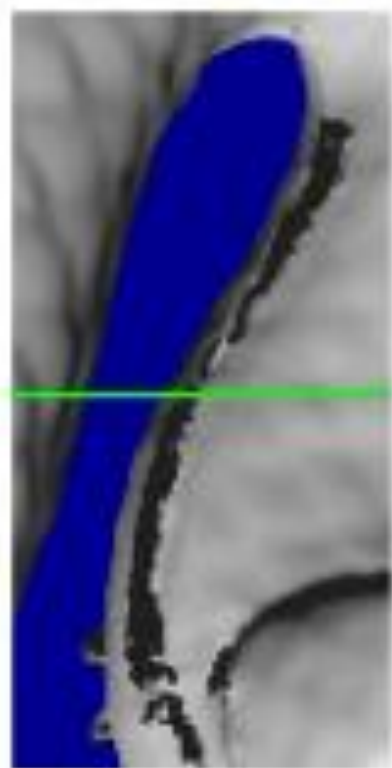


(c)



(d)

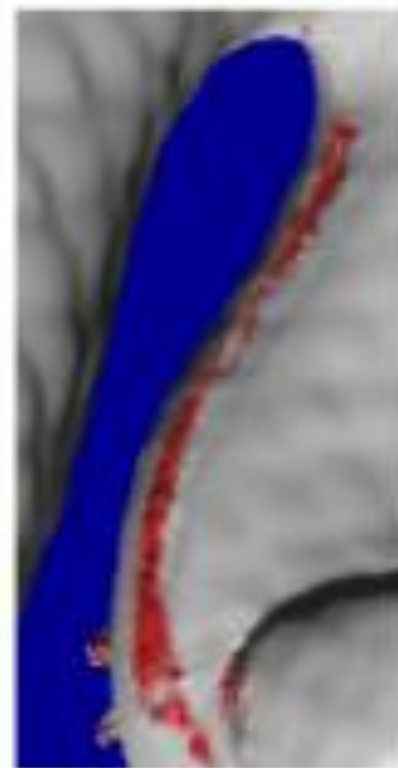




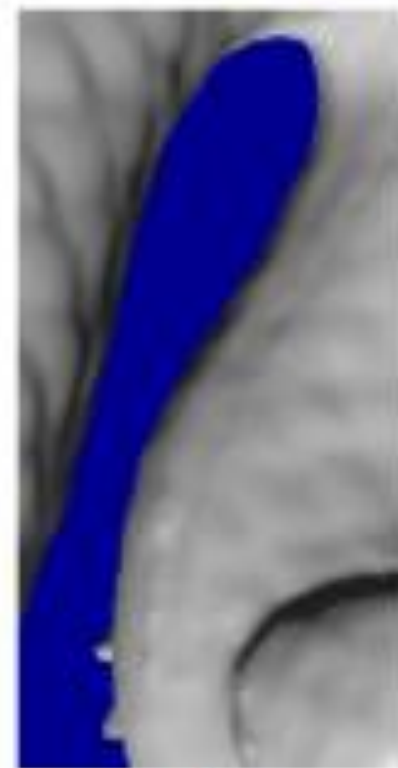
(e)



(f)



(g)



(h)



(i)



(j)



(k)

***THE END...***