Enhancing Voxel Carving by Capture Volume Calculations

International Conference on Image Processing 2010
Outline

1. Introduction
2. Basics
   - Clipping Polygons With Half Spaces
3. Contribution
   - Polyhedron, View Frustum and Adjacency Matrix
   - Polyhedron clipping
   - Problems in the real world
   - Volume calculations
4. Evaluation
5. Conclusion
Introduction

- Dense volumetric reconstruction of gymnasts in sports.
- Multi camera based voxel carving approach.
Problem Statement

During voxel based 3d reconstruction one question directly arises:

**Question**: Which size for the reconstruction area?
During voxel based 3D reconstruction one question directly arises:

**Question:** Which size for the reconstruction area?

**Answer:** Bounding box of clipping polyhedron of all camera frusta.
Quick review of half spaces

Hyper plane $E$ defined by three points $\vec{a}, \vec{b}, \vec{c} \in \mathbb{R}^3$.

Normal $\vec{n}$ and distance to origin $d_E$:
$$ \vec{n} = (\vec{b} - \vec{a}) \times (\vec{c} - \vec{a}), \quad d_E = \langle \vec{n}, \vec{a} \rangle$$

Plane $E$:
$$ E : \langle \vec{n}, \vec{x} \rangle = d \quad \text{with} \quad \vec{x} \in \mathbb{R}^3 $$

Hesse normal form:
$$ \vec{n}_0 = \frac{\vec{n}}{\|d\|}, \quad d_0 = \frac{d}{\|d\|} $$

Distance of a point $\vec{p} \in \mathbb{R}^3$:
$$ d_{E,\vec{p}} = \langle \vec{n}_0, \vec{p} \rangle - d_0 $$

- Half space $H_1$ defined by hyperplane $E$ with normal $\vec{n}$.
- Each point $\vec{p}$ with positive or zero distance to $E$ is $\in H_1$. 
**Goal**: Clipping camera view polygons with each other.

**Approach**: Sutherland & Hodgman [1].
Clipping polygons with planes

- **Goal**: Clipping camera view polygons with each other.

- **Approach**: Sutherland & Hodgman [1].
Sutherland & Hodgman Algorithm

- Published 1974 by Ivan E. Sutherland and Gary W. Hodgman in [1].

- Algorithm is able to clip an arbitrary start polygon with an convex clip polygon:
  1. Treat start polygon as sorted set of points.
  2. Split clip polygon into edges.
  3. Check distance of all points to each clipping edge.
  4. Create target polygon from points with positive distance and points of intersection with clipping edges.

- Edge definition implicitly by order of positive and intersection points.
Clipping Example - Step I

- $A$ and $B$ both have a positive distance to $E$.
  $\Rightarrow B$ added to new polygon (green dot).
$B$ has positive distance, $C$ has negative distance to $E$.  
$\Rightarrow$ Intersection point $BC$ added to new polygon (green dot).
C and D both have a negative distance.
⇒ No point added to new polygon.
$D$ has negative distance, $A$ has positive distance to $E$.  
$\Rightarrow$ Intersection point $DA$ and $A$ added to polygon.
Contribution
Let’s go into 3d now!
Polyhedron and View Frustum

- **Polyhedron**: A geometric solid in 3d with flat faces defined by points or straight edges.
- **Frustum**: Subtype of polyhedron. Apex $A$ defines camera center, ground $BCDE$ defines far plane.

List representation of the faces:

- $A \ B \ C$
- $A \ D \ C$
- $A \ E \ D$
- $A \ B \ E$
- $B \ C \ D \ E$
Problem definition

What do we want to do next?

- Sutherland & Hodgman Algorithm has to be extended to 3d polyhedrons:
  - Clipping of a view frustum ABCDE with a plane H in 3d.
Sutherland & Hodgman in 3d

- **Problem:** In 3d no unique direction along the edges to traverse the start polyhedron.

- **Two possible solutions:**
  1. **Treat faces as polygons.**
     - **Drawback:** Similar intersection points might occur due to numerical precision multiple times and merging is error prone.
  2. **Run along the points instead of the faces.**
     - **Drawback:** Edges will get lost.

- Edges can be reconstructed easily with the Quickhull algorithm [2].

⇒ Polyhedron is traversed along the points.
Problem: In 3d no unique direction along the edges to traverse the start polyhedron.

Two possible solutions:

1. Treat faces as polygons.
   Drawback: Similar intersection points might occur due to numerical precision multiple times and merging is error prone.

2. Run along the points instead of the faces.
   Drawback: Edges will get lost.

Edges can be reconstructed easily with the Quickhull algorithm [2].

⇒ Polyhedron is traversed along the points.
Problem: In 3d no unique direction along the edges to traverse the start polyhedron.

Two possible solutions:

1. Treat faces as polygons.
   Drawback: Similar intersection points might occur due to numerical precision multiple times and merging is error prone.

2. Run along the points instead of the faces.
   Drawback: Edges will get lost.

Edges can be reconstructed easily with the Quickhull algorithm [2].
⇒ Polyhedron is traversed along the points.
Point Traversal via Adjacency Matrix

A  B  C
A  D  C
A  E  D
A  B  E
B  C  D  E

List of faces.
Point Traversal via Adjacency Matrix

A  B  C
A  D  C
A  E  D
A  B  E
B  C  D  E

List of faces.

\[
\begin{pmatrix}
0 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 1 & 0 \\
1 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 1 & 0 \\
\end{pmatrix}
\]

Full adjacency matrix (undirected graph).

If edge between points \( i, j \), then 1, else 0 into adjacency matrix.

- Upper triangle contains directed graph for edge traversal.

Tobias Feldmann, Karsten Brand, Annika Wörner – Capture Volume Calculations
Point Traversal via Adjacency Matrix

A B C
A D C
A E D
A B E
B C D E

List of faces.

\[
\begin{pmatrix}
0 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 1 & 0 \\
1 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 1 & 0 \\
\end{pmatrix}
\]

Full adjacency matrix (undirected graph).

If edge between points \( i, j \), then 1, else 0 into adjacency matrix.

- Upper triangle contains directed graph for edge traversal.

Introduction
Basics
Contribution
Evaluation
Conclusion

Tobias Feldmann, Karsten Brand, Annika Wörner – Capture Volume Calculations
February 23, 2011
13/27
Point Traversal via Adjacency Matrix

\[
\begin{pmatrix}
0 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 1 & 0 \\
1 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 1 & 0
\end{pmatrix}
\]

Full adjacency matrix (undirected graph).

\[
\begin{pmatrix}
1 & 1 & 1 & 1 \\
1 & 0 & 1 \\
1 & 0 \\
1
\end{pmatrix}
\]

Reduced triangular adjacency matrix (directed graph).

- If edge between points \( i, j \), then 1, else 0 into adjacency matrix.
- Upper triangle contains directed graph for edge traversal.
Clip all edges defined by points reachable from A.
Clip all edges defined by points reachable from B.
Clip all edges defined by points reachable from C.
Clip all edges defined by points reachable from D.
Two Steps to Finalize the Clipping

- **Clipping with ground plane:**
  - Assumption: Floor is the plane with coordinate $z = 0$.
  - Additional clipping with the ground plane.

- **Finally: Reconstruction of edges:**
  - Clipping algorithm results in point cloud.
  - Each clipping is convex $\Rightarrow$ Convex hull with Quickhull [2].
    - Edges are not necessarily the same after clipping.
    - Cooplanar points get removed.
From theory to praxis

In praxis two problems arise:

1. A scale factor has to be found for the initial frustum.
2. The frusta need to be convex but in reality they are not.
Problem 1: Scale Factor of Initial Frustum

- Corner points of initial polyhedron are scaled and projected into 3d.
- Too small → erroneous final polyhedron.
- Too large → might lead to numerical instabilities.
- Scale factor to be found heuristically, e.g. a priori knowledge about setup.
- Needed for initial polyhedron only (because clipping polyhedra represented by planes).
Problem 2: 
Non-Convex Camera Frusta

Camera lense distortion deforms view frustum.
View frustum not convex any more ⇒ S&H not applicable.

Solution: Use smallest enclosing / largest inner convex frustum as clipping polyhedron.
Creating a convex enclosing frustum

Init frustum: Corner point projections of a 2d camera image for creation of initial convex view frustum.

For each face $PST$ of the frustum do:

1. Find biggest negative difference $Q$ outside frustum; determine projection $Q'$ onto $PST$.
2. Calculate angle $\theta$ between $\vec{PQ}$ and $\vec{PQ'}$ by 
   \[ \theta = \arccos \left( \frac{\langle Q'-P, Q-P \rangle}{||Q'-P|| \cdot ||Q-P||} \right). \]
3. Use $\theta$ and normed rotation axis $\vec{a}$ parallel to $\vec{ST}$ to create rotation matrix $R$.
4. Rotate plane $PST$ and recalculate intersection points $S'$ and $T'$ (necessary, since faces not necessarily perpendicular)
Creating a convex enclosing frustum

**Init frustum:** Corner point projections of a 2d camera image for creation of initial convex view frustum.

For each face $PST$ of the frustum do:

1. Find biggest negative difference $Q$ outside frustum; determine projection $Q'$ onto $PST$.

2. Calculate angle $\theta$ between $\vec{PQ}$ and $\vec{PQ'}$ by
   \[
   \theta = \arccos \left( \frac{\langle Q'-P, Q-P \rangle}{||Q'-P|| \cdot ||Q-P||} \right).
   \]

3. Use $\theta$ and normed rotation axis $\vec{a}$ parallel to $\vec{ST}$ to create rotation matrix $R$.

4. Rotate plane $PST$ and recalculate intersection points $S'$ and $T'$ (necessary, since faces not necessarily perpendicular).
Creating a convex enclosing frustum

**Init frustum:** Corner point projections of a 2d camera image for creation of initial convex view frustum.

For each face $PST$ of the frustum do:

1. Find biggest negative difference $Q$ outside frustum; determine projection $Q'$ onto $PST$.
2. Calculate angle $\theta$ between $\vec{PQ}$ and $\vec{PQ'}$ by $\theta = \arccos \left( \frac{\langle Q' - P, Q - P \rangle}{||Q' - P|| \cdot ||Q - P||} \right)$.
3. Use $\theta$ and normed rotation axis $\vec{a}$ parallel to $\vec{ST}$ to create rotation matrix $R$.
4. Rotate plane $PST$ and recalculate intersection points $S'$ and $T'$ (necessary, since faces not necessarily perpendicular).
Creating a convex enclosing frustum

**Init frustum:** Corner point projections of a 2d camera image for creation of initial convex view frustum.

For each face $PST$ of the frustum do:

1. Find biggest negative difference $Q$ outside frustum; determine projection $Q'$ onto $PST$.
2. Calculate angle $\theta$ between $\vec{PQ}$ and $\vec{PQ'}$ by $\theta = \arccos \left( \frac{\langle Q'-P, Q-P \rangle}{||Q'-P|| \cdot ||Q-P||} \right)$.
3. Use $\theta$ and normed rotation axis $\vec{a}$ parallel to $\vec{ST}$ to create rotation matrix $R$.
4. Rotate plane $PST$ and recalculate intersection points $S'$ and $T'$ (necessary, since faces not necessarily perpendicular).
Creating a convex enclosing frustum

**Init frustum:** Corner point projections of a 2d camera image for creation of initial convex view frustum.

**For each face PST of the frustum do:**

1. Find biggest negative difference $Q$ outside frustum; determine projection $Q'$ onto $PST$.
2. Calculate angle $\theta$ between $\overrightarrow{PQ}$ and $\overrightarrow{PQ'}$ by $\theta = \arccos \left( \frac{\langle Q' - P, Q - P \rangle}{||Q' - P|| \cdot ||Q - P||} \right)$.
3. Use $\theta$ and normed rotation axis $\vec{a}$ parallel to $\overrightarrow{ST}$ to create rotation matrix $R$.
4. Rotate plane $PST$ and recalculate intersection points $S'$ and $T'$ (necessary, since faces not necessarily perpendicular).
Example with 8 cameras

View from top.

View from the side.
View from top.

View from the side.
Example - Camera 2

View from top.

View from the side.
Example - Camera 3

View from top.

View from the side.
Example - Camera 4

View from top.

View from the side.
Example - Camera 5

View from top.

View from the side.
Example - Camera 6

View from top.

View from the side.
Example - Camera 7

View from top.

View from the side.
Example - Camera 8

View from top.

View from the side.
Example - Final Result

Final polyhedron of reconstruction area.

Backprojection.
Calculating the Voxel Size

- Extrema of polyhedron define:
  - bounding box of reconstruction,
  - centered origin,
  - ratio of the three dimensions $d_i$ with $i \in \{1 \ldots 3\}$.

- Under assumption of cubic voxels, $\lambda$ describes scale factor to adopt voxel granularity to given memory constraints:

$$\lambda = \frac{3 \sqrt[n_{\text{voxels}}]{\prod_{i=1}^{3} d_i}}{\text{max. memory \hspace{1em} voxel mem. size}}$$

- After scaling, voxel edge length is $\frac{1}{\lambda}$ the original size.
Reconstruction Evaluation

Evaluations:

- Synthetic examples with known sizes.
- Real life example “Gymnast Sequence”:
  - Camera space: $13, 3 \times 6, 2 \times 4, 4$ m.
  - Polyhedron space: $7, 2 \times 4, 6 \times 2, 7$ m.
  - Voxel granularity increase from 23.7mm to 14.9mm.
Automatically Improved Reconstruction Granularity

Coarse reconstruction based on camera positions.
- \#Voxels: \(300^3 = 27\text{Mio}\)
- Voxel size: 23.7 mm

Fine reconstruction based on polyhedron (proposed method).
- \#Voxels: \(300^3 = 27\text{Mio}\)
- Voxel size: 14.9 mm
Conclusion

- Extension of Sutherland & Hodgman algorithm to 3d polyhedron clipping.
- Clipping of view frusta.
- Handling lense distortions.
- Improving voxel reconstructions by using the calculated clipping polyhedron as voxel space parameterization.

- Additional future applications:
  - Automatic segmentation initialization, e.g. with GrabCut.
  - Interactive camera arrangement.
  - Plausibility checks for depth estimation.
Conclusion

- Extension of Sutherland & Hodgman algorithm to 3d polyhedron clipping.
- Clipping of view frusta.
- Handling lense distortions.
- Improving voxel reconstructions by using the calculated clipping polyhedron as voxel space parameterization.

- Additional future applications:
  - Automatic segmentation initialization, e.g. with GrabCut.
  - Interactive camera arrangement.
  - Plausibility checks for depth estimation.
Any questions?


Performance

- Evaluation performed on sequences of three different setups:
  - HumanEVA-Benchmark [3]: 4 cameras,
  - Gymnast sequence: 7 cameras,
  - Laboratory sequence: 8 cameras.
- Windows Vista 32 Bit, 3,3GB RAM, Intel Core 2 Duo 2,3 GHz

<table>
<thead>
<tr>
<th>Sequence</th>
<th>#Cams</th>
<th>Max. time</th>
<th>Volume</th>
<th>#Joints</th>
<th>#Faces</th>
</tr>
</thead>
<tbody>
<tr>
<td>HumanEVA</td>
<td>4</td>
<td>66 ms</td>
<td>20,619 m³</td>
<td>32</td>
<td>52</td>
</tr>
<tr>
<td>Gymnast</td>
<td>7</td>
<td>101 ms</td>
<td>54,978 m³</td>
<td>32</td>
<td>60</td>
</tr>
<tr>
<td>Laboratory</td>
<td>8</td>
<td>166 ms</td>
<td>13,114 m³</td>
<td>52</td>
<td>76</td>
</tr>
</tbody>
</table>

- Performance mainly depends on the number of used cameras.