About Virtual Worlds
Our purposes

- Create huge worlds.
- Reduce working time for artists.
- Reduce rendering time and memory consumption.
Our purposes

- Create huge worlds.
- Reduce working time for artists.
- Reduce rendering time and memory consumption.

**Automatic and controlable content creation**
Our purposes

- Create huge worlds.
- Reduce working time for artists.
- Reduce rendering time and memory consumption.

Automatic and controlable content creation

Hardware compliant data
Summary

1. Procedural Modeling
2. Visibility & Occlusion Culling
3. Occlusion Tiling
4. Conclusion
Procedural Modeling : Previous Work

In a general sense : Automatic Data Generation
In a general sense: **Automatic Data Generation**

In a computer graphics sense: **Automatic Data Generation for geometries, texture, ...**

- Wei '00
- Prusinkiewicz '90
- Maréchal '08
- Deussen '98
- Peytavie '09
- Leblanc '11
Peekaboo!

1. Procedural Modeling
2. Visibility & Occlusion Culling
3. Occlusion Tiling
4. Conclusion
Reminder

- **What?** Reduce the number of primitives to increase framerate.
- **How?** By not displaying hidden geometries.
From Point Visibility

Blockers boundaries determine the discontinuities of the visibility function.
From Region Visibility: Vocabulary

A - occluder
T - occludee

T is not occluded by A in regions 1
T is partially occluded by A in regions 2
T is fully occluded by A in region 3
Visible objects from a view cell: potentially visible parts of the entire scene.
Potentially Visible Sets reduce display costs.
Huge **display savings** but also huge **memory costs**.
Ze solution

1. Procedural Modeling
2. Visibility & Occlusion Culling
3. Occlusion Tiling
4. Conclusion
Elaborate a procedural occlusion tiling technique.
Goals : Feng Shui

Elaborate a **procedural** occlusion tiling technique.
Goals: Feng Shui

Elaborate a procedural *occlusion* tiling technique.
Goals: Feng Shui

Elaborate a procedural occlusion technique.
Idea

- A *priori* hidden surface removal.
- Object procedural modeling VS object procedural positioning.
- Replacing *creation, display, and memory* costs by *construction precomputing* costs.
Occlusion Tiling: Definition

2D visibility through a tile:
- How to ensure any line of sight is blocked?
- How to determine visible regions?
Occlusion Tiling

- Square tiling

1 tile = 1 set of blockers
1 tile = 1 independent view cell
Occlusion Tiling

- Square tiling
- 1 tile = 1 set of blockers
Occlusion Tiling

- Square tiling
- 1 tile = 1 set of blockers
- 1 tile = 1 independent view cell

**Goal:**
Ensure that all lines of sight coming from the view cell are blocked beyond the first ring of tiling neighborhood.
Occlusion Tiling

- Square tiling
- 1 tile = 1 set of blockers
- 1 tile = 1 independent view cell

Goal: Ensure that all lines of sight coming from the view cell are blocked beyond the first ring of tiling neighborhood.
\( \bar{u} \)-occlusion Scheme

Emitter segment

Receiving segments
\( \bar{u} \)-occlusion Scheme

4 visibility constraints to ensure on each tile.
\( \bar{u} \)-occlusion Scheme

4 visibility constraints to ensure on each tile.
$\bar{u}$-occlusion Scheme

4 visibility constraints to ensure on each tile.
$\bar{u}$-occlusion Scheme

4 visibility constraints to ensure on each tile.
\( \bar{u} \)-occlusion Scheme

4 visibility constraints to ensure on each tile.
Randomly generate a grid of blockers (uniform or according to user specification).

Test occlusion.

If occluding, add to the tiling set.
Results

16 × 16 occluding tiles, occluding density : 1/5 (20%)
32 × 32 occluding tiles, occluding density : 1/5 (20%)
64 × 64 occluding tiles, occluding density : 1/7(14,3%)
## Results – 90 minutes

<table>
<thead>
<tr>
<th>Dimension</th>
<th>% Occlusion Density</th>
<th># Tested</th>
<th># Positive</th>
</tr>
</thead>
<tbody>
<tr>
<td>16 x 16</td>
<td>1/4 (25%)</td>
<td>260725</td>
<td>2960</td>
</tr>
<tr>
<td></td>
<td>1/5 (20%)</td>
<td>1064670</td>
<td>209</td>
</tr>
<tr>
<td></td>
<td>1/6 (16.6%)</td>
<td>2482720</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>1/7 (14.3%)</td>
<td>4399280</td>
<td>0</td>
</tr>
<tr>
<td>32 x 32</td>
<td>1/4 (25%)</td>
<td>801</td>
<td>329</td>
</tr>
<tr>
<td></td>
<td>1/5 (20%)</td>
<td>3710</td>
<td>112</td>
</tr>
<tr>
<td></td>
<td>1/6 (16.6%)</td>
<td>19608</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>1/7 (14.3%)</td>
<td>62224</td>
<td>0</td>
</tr>
<tr>
<td>64 x 64</td>
<td>1/4 (25%)</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>1/5 (20%)</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>1/6 (16.6%)</td>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>1/7 (14.3%)</td>
<td>59</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>1/8 (12.5%)</td>
<td>244</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1/9 (11.1%)</td>
<td>909</td>
<td></td>
</tr>
</tbody>
</table>
1. Generate a $n \times n$ random tile satisfying the 4 $\bar{u}$-schemes.
2. Double the resolution.
3. Mutate pixels in the $2n \times 2n$ resolution by randomly moving occluders; Test new tiles for occlusion.
4. Repeat steps 2 and 3 with the resulting mutated tiles that passed the occlusion test until the desired resolution is reached.
Occlusion Tiling: Hierarchical Scheme

Occluding tiles and derived occluding tiles
\( \times 10 \) more solutions
\( \div 10 \) less computation time
Occlusion Tiling: Customization

Each tile pixel is generated with a probability linearly derived from the source image.
Aerial and ground views of our extruded 3D city from our 2D blockers.
Height variation in building construction.
Multi-scale Occlusion Tiled City

Aerial and ground views of our extruded 3D city from our 2D blockers.
Zen Attitude

1. Procedural Modeling
2. Visibility & Occlusion Culling
3. Occlusion Tiling
4. Conclusion
Conclusion

- Increasing visual field occlusion.
- *A priori* Hidden Surface Removal.
- Replacing creation, rendering and memory costs by construction of visibility relations between tiles.
Current Work

Visibility propagation.

Univ. of Montreal & Toulouse III
Future Work

- Other types of worlds.
- Other shapes of tiles.
- Non planar worlds.
- 3D visibility computations.
Thank you! email me at dorian.gomez@irit.fr for any question concerning fish cod(e) recipe.
Figure: Occlusion Tiling: instantiation by trees.