Exploration of geo-virtual Environments using 3D Magic Lenses

Timo Ropinski
ropinski@math.uni-muenster.de

Institute for Computer Science
University of Münster, Germany

Aim

- Reveal information, which is hidden in large sets of high-dimensional data

Problem

- Only three dimensions in space (plus one dimension in time) can be used in 3D visualization
- Difficult to visualize a large number of attributes associated with geo-objects

One common Strategy

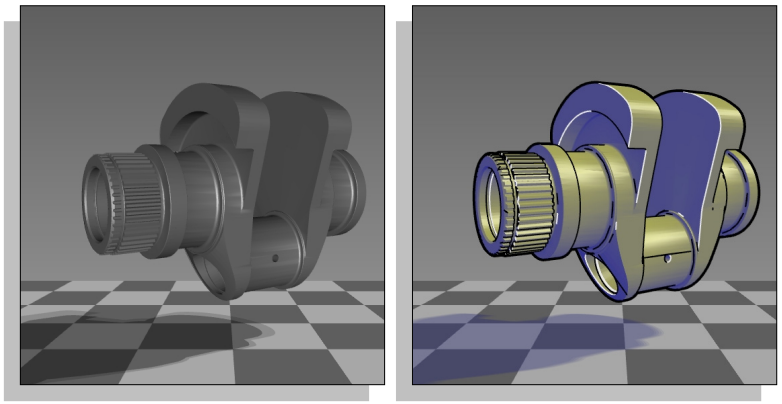
- Visualizing just a subset of attributes
- Same subset of attributes is visualized everywhere in the virtual environment

Why use Magic Lenses?

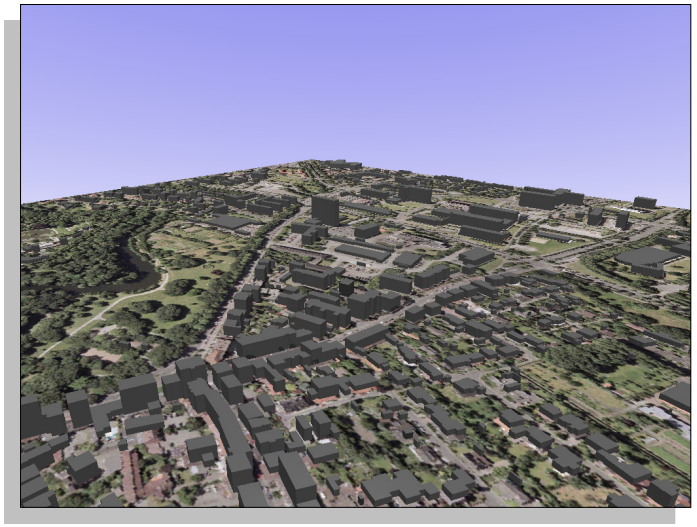
- Magic lenses allow to visualize different sets of attributes in different parts of the virtual environment

City Model Visualization - Rendering Techniques

- **Photorealistic** Rendering:
  Try to match reality as close as possible
- **Non-Photorealistic** Rendering:
  Display only *necessary* information

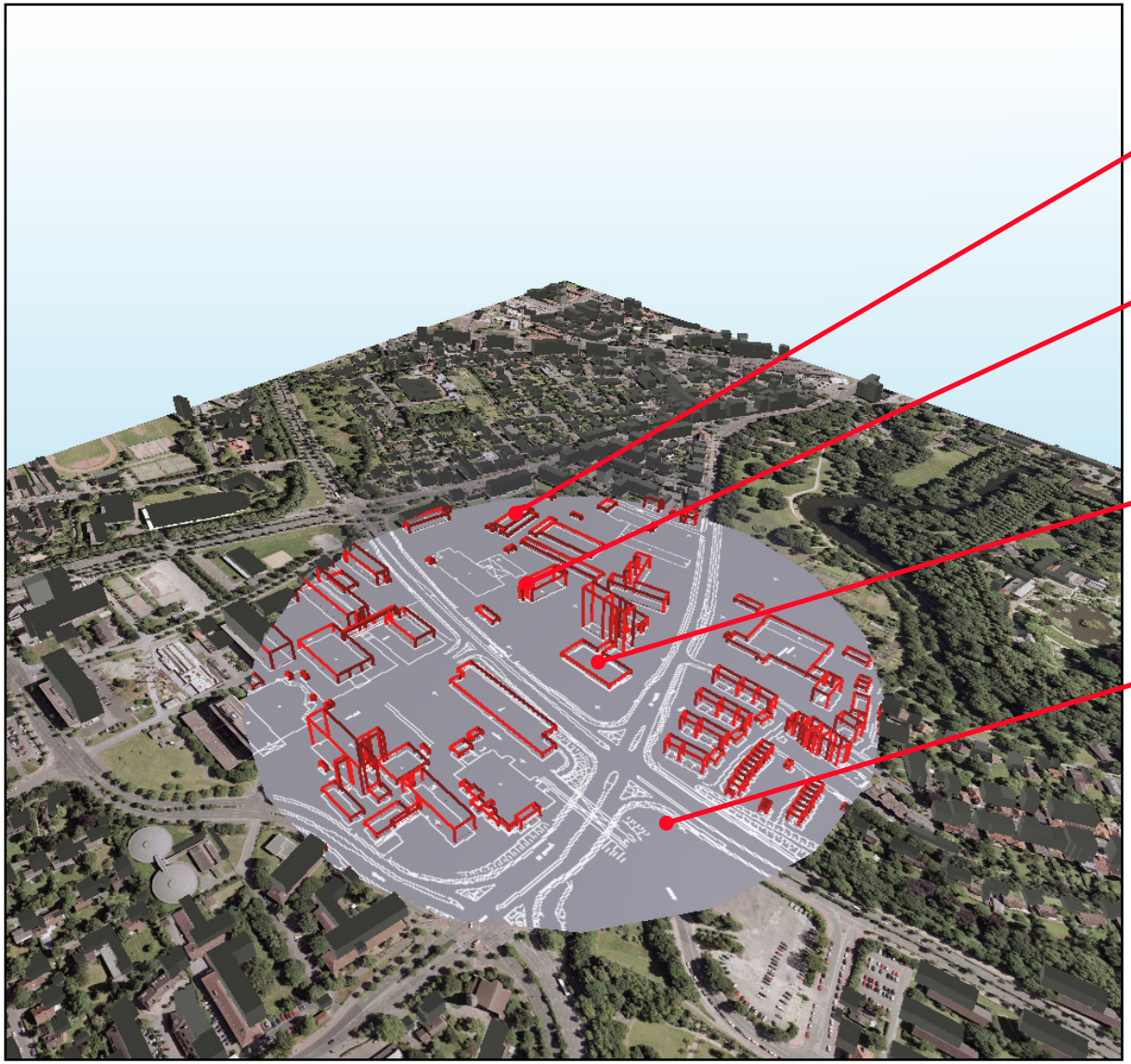


[Gooch et al.: *Interactive Technical Illustration]*



[Döllner et al.: *Real-Time Expressive Rendering of City Models*]

City Model Visualization – Example Lens



Buildings are colored red

Buildings are rendered in wireframe mode

Roofs have been removed

Air photograph has been replaced by sketch

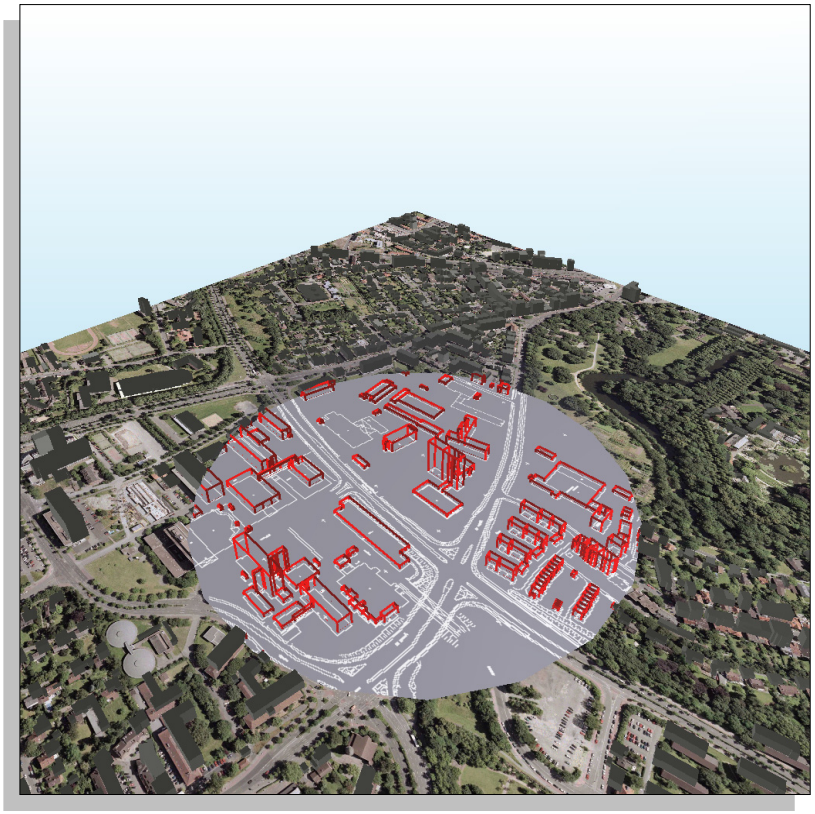
City Model Visualization - Datastructure

- To modify the visual appearance of selected geo-objects they must be identifiable
- Objects can be excluded from rendering

```
excludeList = {'roof'};
```

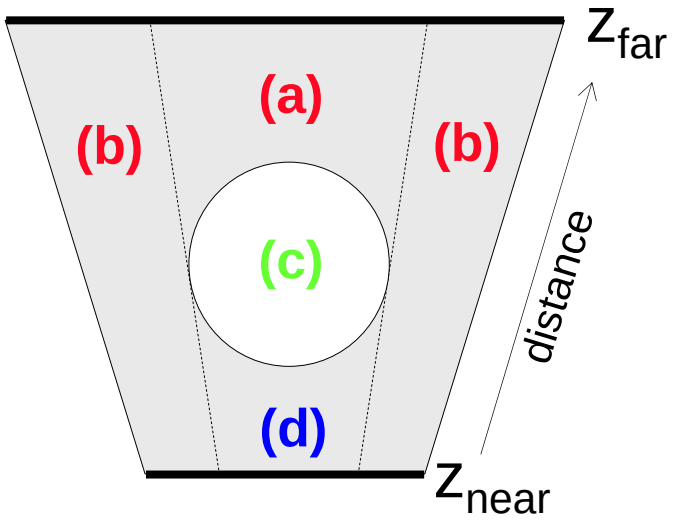
- Graphical representations are modified using an attribute stack

Color **red**	Texture0 **sketch**
FaceStyle **wireframe**	
LineWidth **2.0f**	
building	**ground**



Functionality

- Obtain depth information
- Render scene
  - **Behind** and next to the lens volume (a,b)
  - **Inside** the lens volume (c)
  - **In front of** the lens volume (d)



Characteristics

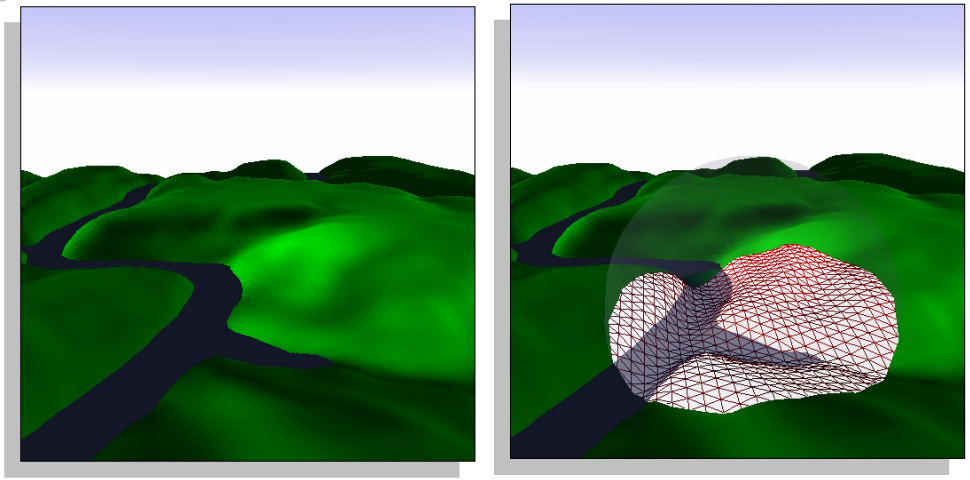
- **Image-based** multi-pass rendering algorithm
- **Hardware-accelerated** by current GPU's
- Implemented using **OpenGL** and **VRS**

Terrain Visualization

- Highlight region of interest

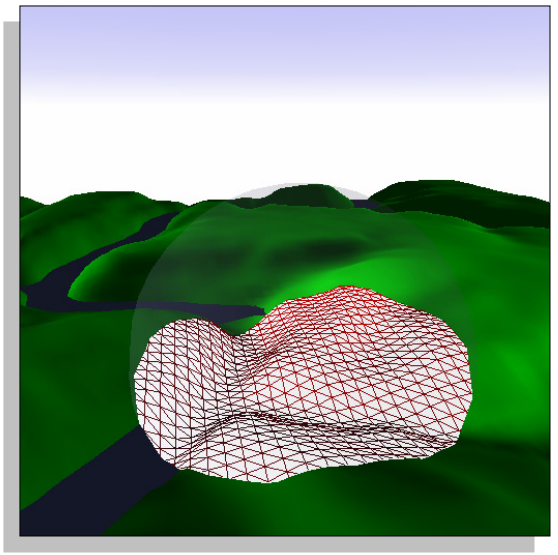
Color **white**	FaceStyle **wireframe**	
terrain		

Opacity **0.4f**	
water	



- Alternatively

```
excludeList = {'water'};
```



Lens Types

Scene Lenses

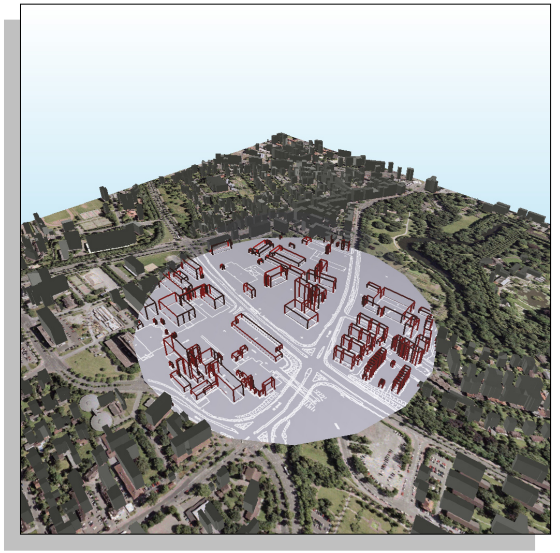
- can be positioned anywhere in the virtual environment
- highlight regions of interest interactively

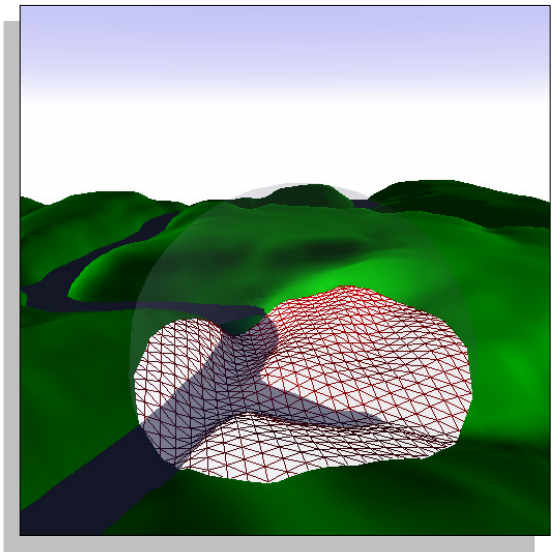
Camera Lenses

- positioned relative to the virtual camera
- assist the user while exploring dense datasets

Example lenses:

- semi-transparent rendering
- removing of partial or complete data
- highlighting data inside the lens volume by applying extra light sources





Conclusion

- Image-based real-time **rendering algorithm**
- Magic lenses in **city model** visualization
- Application to **terrain** visualization
- Classification into two **lens types**

Future Work

- **Multi lens** environments
- Library for **easy use** of magic lenses in existing OpenGL applications

Thank you