Shape Recognition in 3D Point-Clouds

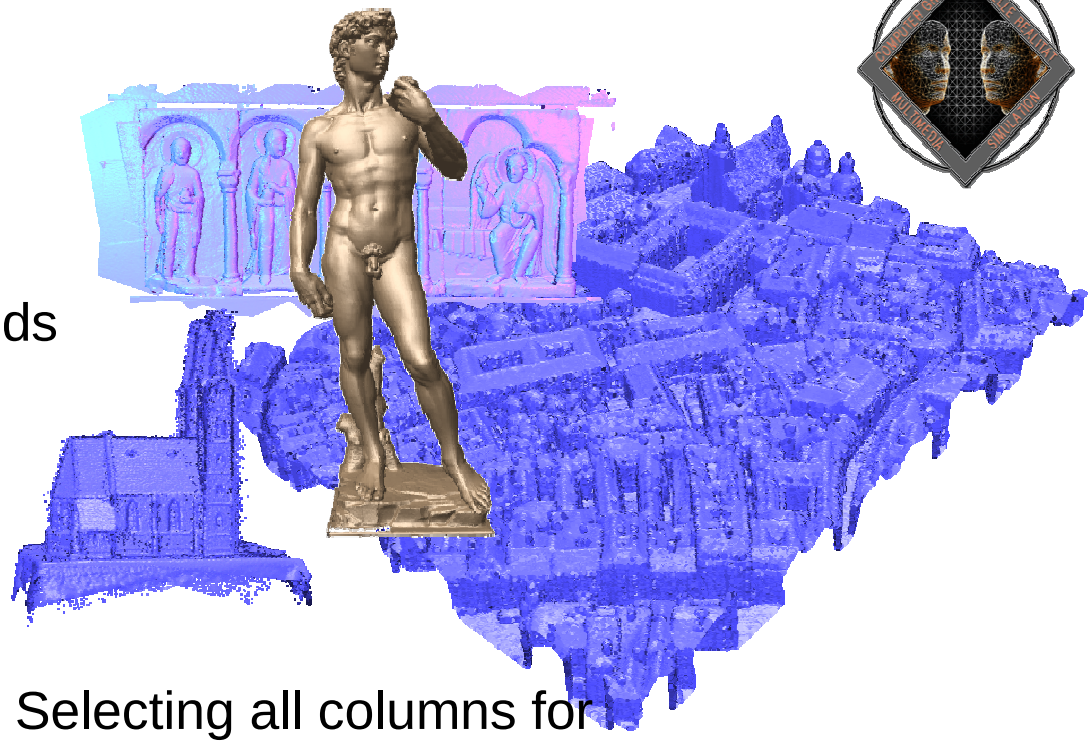
Ruwen Schnabel, Raoul Wessel, Roland Wahl
and Reinhard Klein

University of Bonn

universitätbonn

Introduction

- Size of geometric datasets increases quickly
- In particular: Large point-clouds from 3d scanning

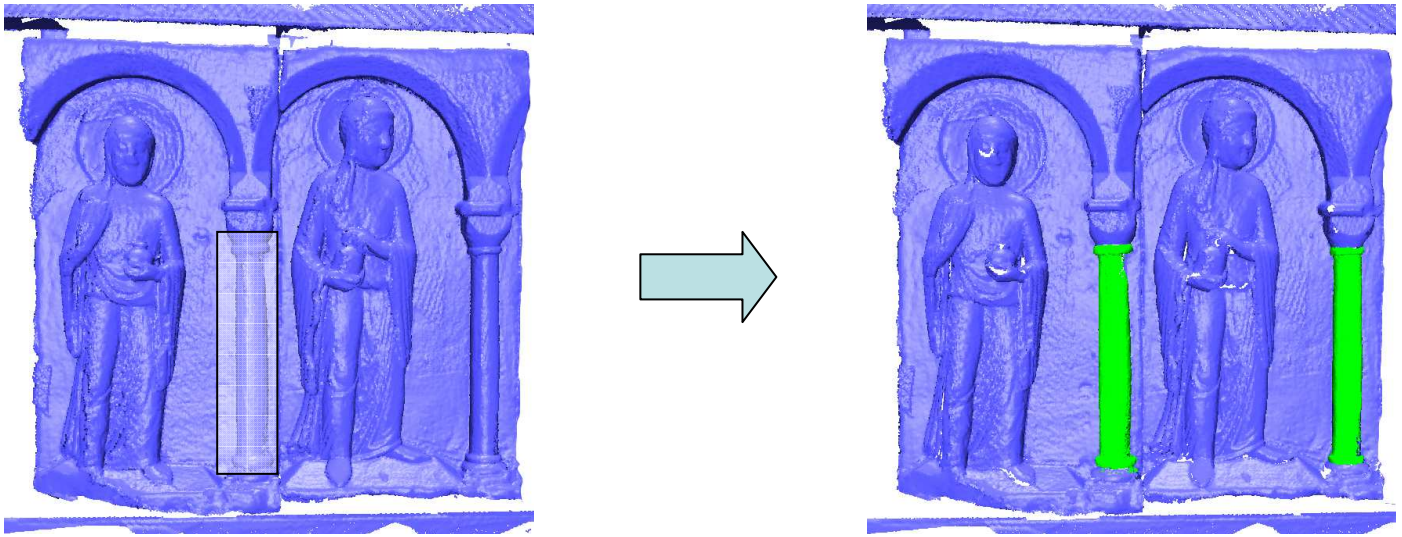


Problem:

- Interaction still limited, e.g.: Selecting all columns for assigning them colors or textures
- Common approach:
  - Selecting columns with mouse
  - Time consuming and imprecise

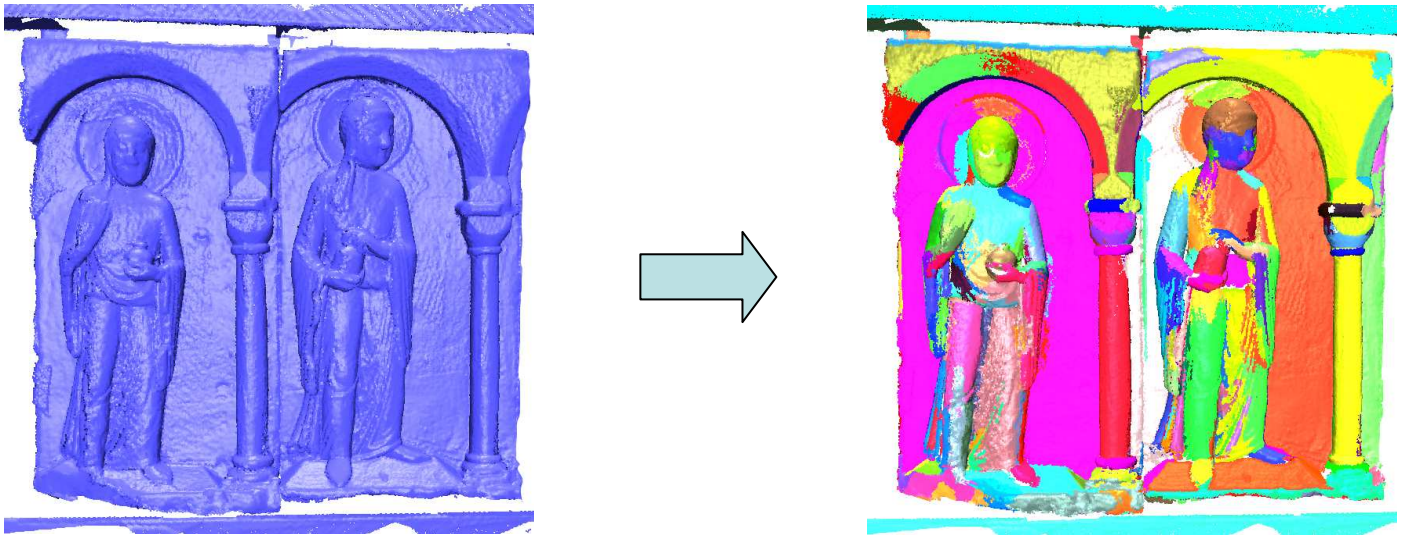
Introduction

- Desirable approach:
  - Select single column
  - find all other columns automatically



Introduction

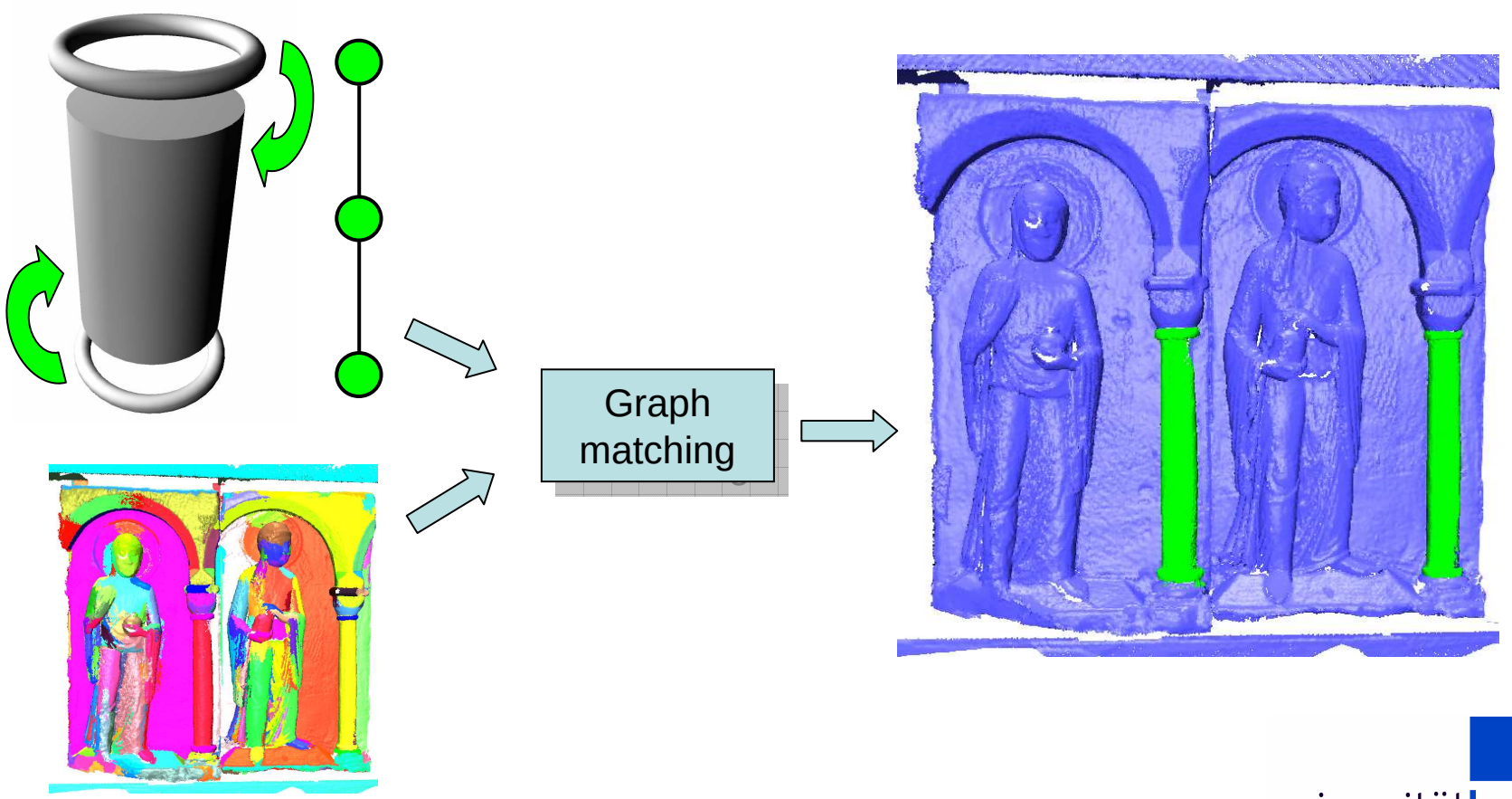
- Our approach:
  - Based on primitive shapes (plane, cylinder, cone,...)
    - Well suited for man-made objects
    - E.g.: column ≈ cylinder
  - Decompose point-cloud into primitive shapes



universitätbonn

Introduction

- User defines „query graph“
  - Desired configuration of primitive shapes
- Search for configuration in decomposed point-cloud
  - Graph matching



universitätbonn

Related Work

- 3D city reconstruction
  - Verma, Kumar, Hsu: *3D Building Detection and Modeling from Aerial Lidar Data*, CVPR 2006
    - detection of roof types
    - configurations of planes (region growing)
    - limitations: only planes, no node and graph constraints

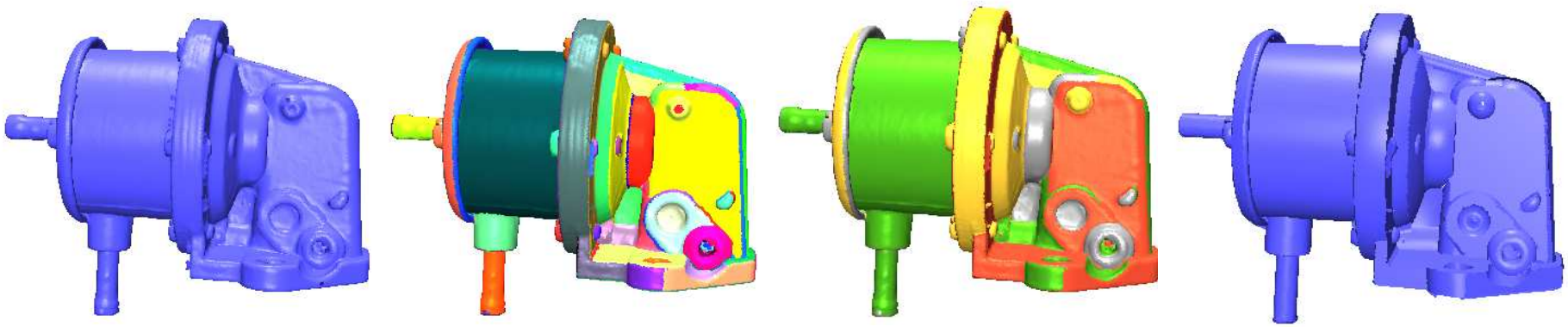
- Graph based shape matching
  - Zuckerberger, Tal, Shlafman: *Polyhedral surface decomposition with applications* Computers and Graphics, 2002
    - segmentation into *meaningful* units
    - unconstrained subgraph matching -> prohibitively time consuming
    - restricted to triangle meshes -> segmentation not stable

Point Cloud Decomposition

- Efficient RANSAC
- Detection of planes, cylinders, spheres, cones and tori
  - Fast, robust

$$P = S_{\phi_1} \cup \ldots \cup S_{\phi_A} \cup R$$

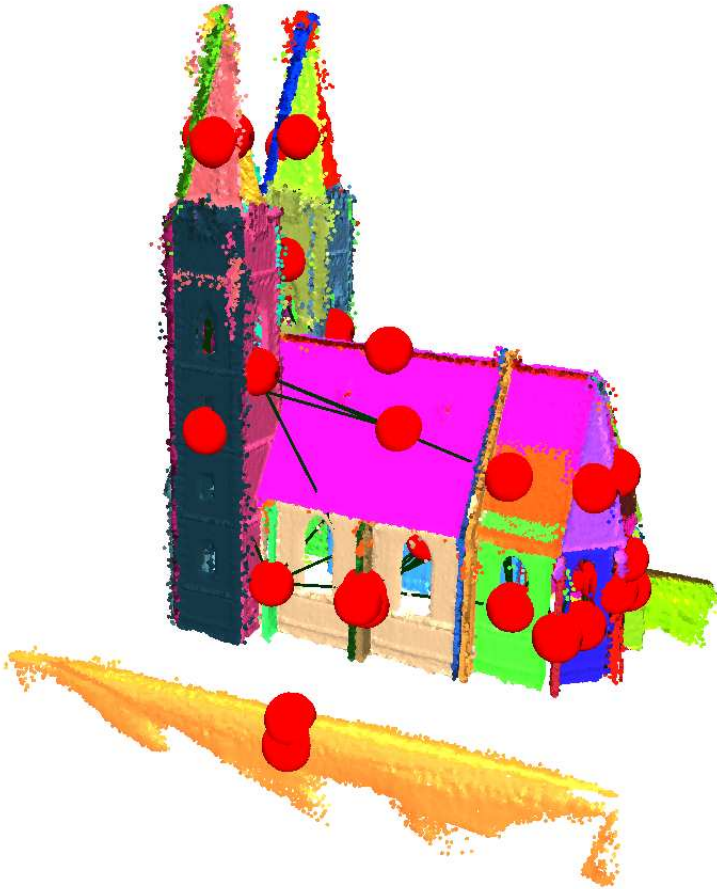
- Set of remaining points will be ignored
  - Outliers, noise, complex geometry

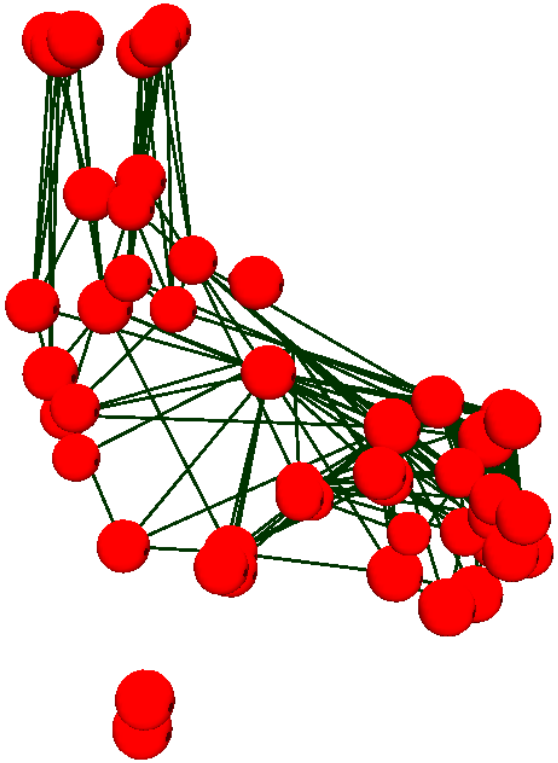


universitätbonn

Topology Graph

- Captures spatial relation between primitives
- Represent primitive shapes as **nodes**
- Neighboring nodes are connected by **edges**
  - controlled by proximitiy threshold t

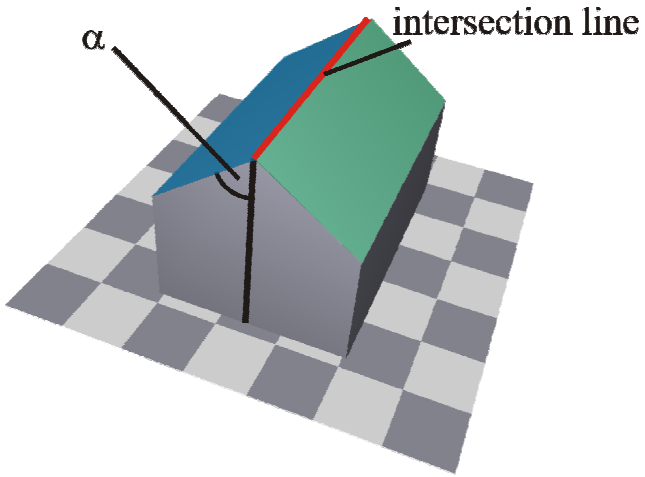




universitätbonn

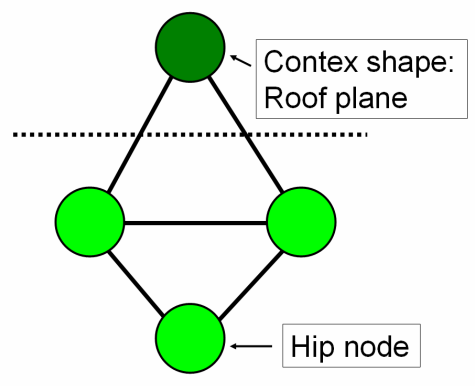
Defining Query Graphs

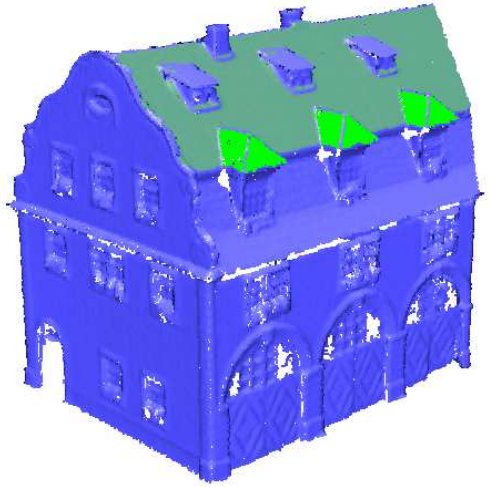
- Node Constraints
  - type of underlying primitive shape (e.g. plane, cylinder,...)
  - assign additional constraints manually (e.g. normals, angles, radius,...)
- Edge Constraints
  - describes spatial configuration between two shapes (e.g. angles, intersections, relative sizes,...)
- Graph Constraints
  - configuration constraints between unconnected nodes



Defining Query Graphs

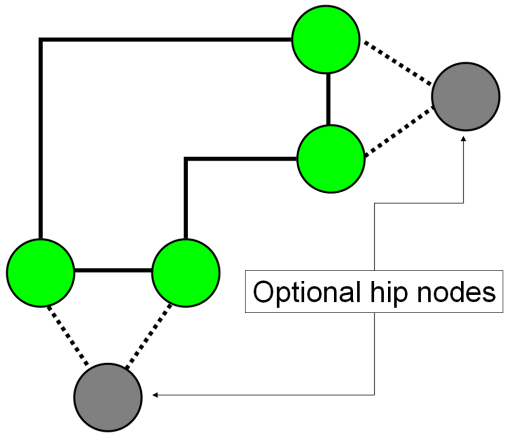
- Context Nodes
  - certain features benefit from *context objects* to distinguish them from other structures
  - model context objects as additional nodes in query graph
  - after matching, remove context nodes from result

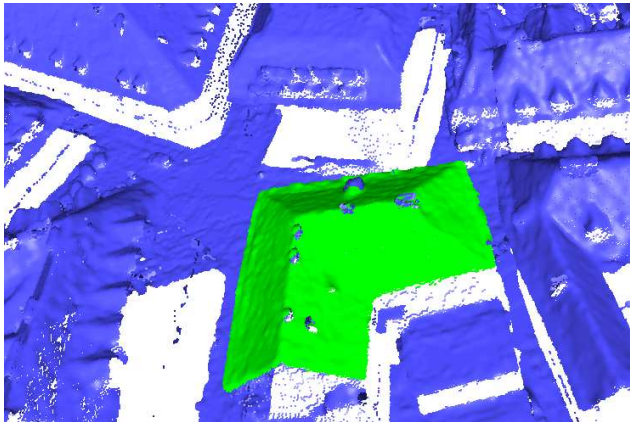




Defining Query Graphs

- Optional Nodes
  - represent variants of one shape
- Matching
  - first match query graph without optional nodes
  - try to maximally extend results by optional nodes

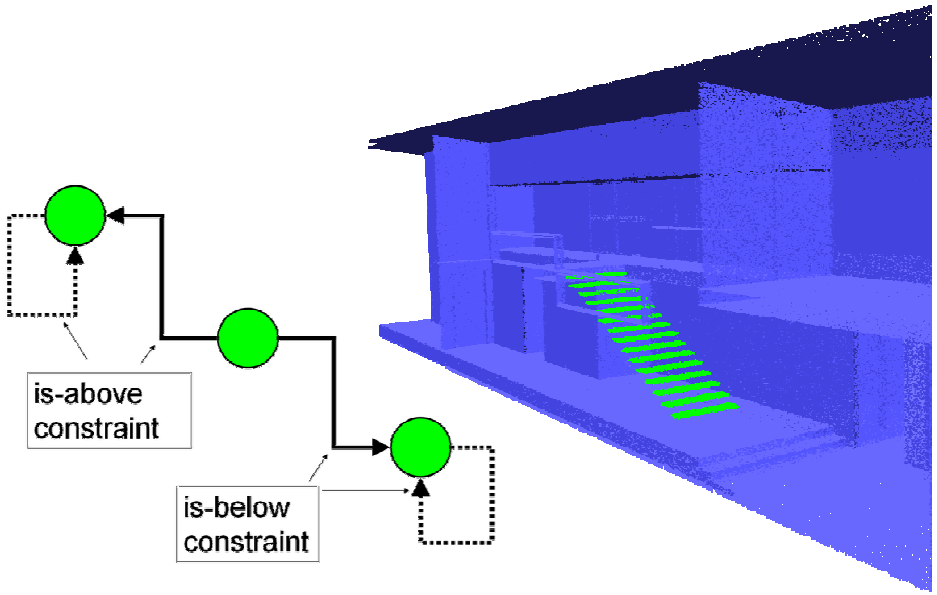




- Optional graph components
  - use graph constraints to assure completeness of component

Defining Query Graphs

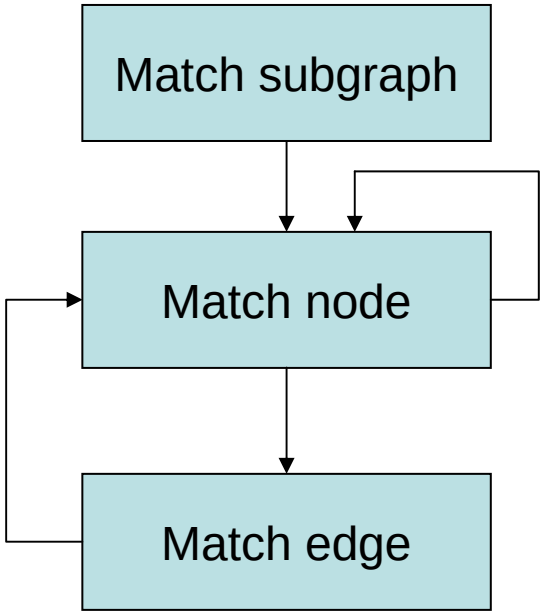
- Multinodes
  - represent repititive patterns in shapes (e.g. stairs)
  - modeled as self loops in the query graph



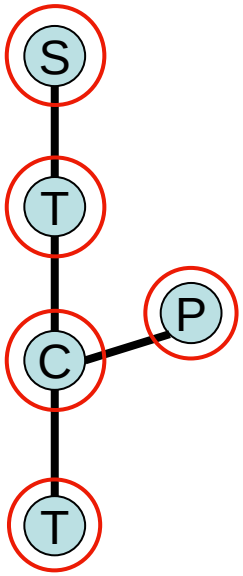
  - use of directed edges
  - restricted to one outgoing edge per multinode

Graph Matching

- Recursive graph matching
  - node and edge constraints are checked early on
  - graph constraints are checked when all nodes and edges have been correctly matched



Topology graph



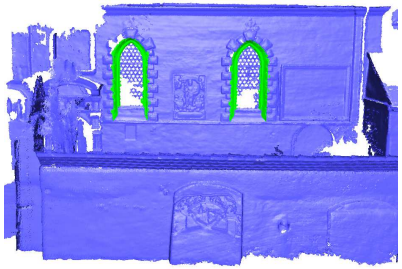
Query graph

start node

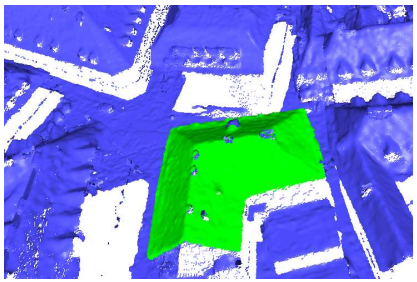
checking edge constraints...
edge constraints match!!

universitätbonn

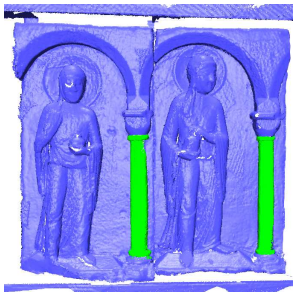
Timings



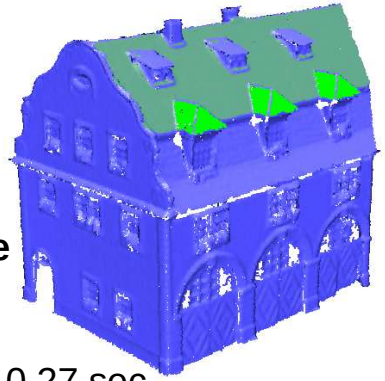
chapel
nodes: 232
edges: 406
graph construction: 1.8 sec
matching: < 10 msec



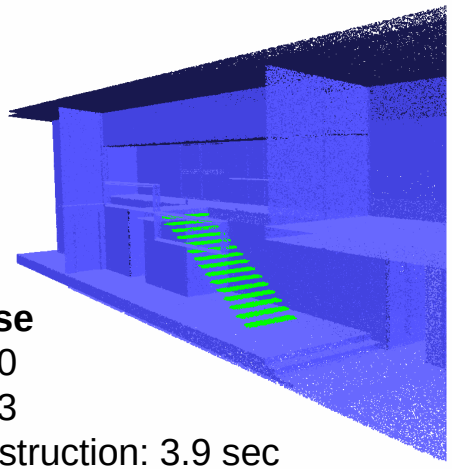
city model
nodes: 431
edges: 351
graph construction: 2.5 sec
matching: < 10 msec



choir screen
nodes: 537
edges: 2731
graph construction: 1.8 sec
matching: < 10 msec



dormer roof house
nodes: 106
edges: 138
graph construction: 0.27 sec
matching: < 10 msec



CAD house
nodes: 160
edges: 513
graph construction: 3.9 sec
matching: < 10 msec

universitätbonn

Conclusion

- Object detection in point clouds based on primitive shape decomposition
  - suited even for large amounts of data
  - applicable to nearly every data source like point-clouds, polygon soups, meshes
  - very fast due to constrained graph matching

- Limitations
  - restricted to objects consisting of primitive shapes
    - Not suited e.g. for natural objects like flowers or animals
  - large graphs might lead to degeneration of retrieval performance

Future Work

- User Interface
  - more comfortable interface for unexperienced users
  - easily define query graphs and constraints
- Compression
  - detect self-similarities
  - replace instances of query graphs by generic representations
- Editing
  - Apply basic point cloud operations like copy and paste of semantic units

universitätbonn

Thanks for your attention !

universitätbonn