The Room Connectivity Graph: Shape Retrieval in the Architectural Domain

Raoul Wessel
Computer Graphics Group
University of Bonn
53117 Bonn, Germany
wesselr@cs.uni-bonn.de

Ina Blümel
German National Library
of Science and Technology
30167 Hannover, Germany
ina.bluemel@tib.uni-hannover.de

Reinhard Klein
Computer Graphics Group
University of Bonn
53117 Bonn, Germany
rk@cs.uni-bonn.de

ABSTRACT

While advances in CAD modeling techniques led to an ever increasing number of available architectural 3D models, reusability of these models as templates or as inspiration sources is still very limited. One reason for this is that there exist basically no shape retrieval methods specialized in the architectural domain. In this work, we therefore present a method to efficiently characterize 3D architectural models according to the underlying arrangement of their rooms by a *room connectivity graph*. In this graph, rooms are represented by attributed nodes. Connections between rooms, i.e. doors or windows, are represented by attributed edges. We show that these attributed graphs can be used for an efficient retrieval of 3D architectural models using fast graph matching techniques.

Keywords: 3D Shape Retrieval, Graph Matching

1 INTRODUCTION

3D modeling plays a more and more important role in the architectural domain. Along with the increasing amount of available 3D models of buildings comes the requirement to search building databases for those models that are similar to a query object, either for inspirational purposes or for reuse.

Common state-of-the-art 3D shape retrieval techniques mostly rely on the extraction of global or local geometric features and shape descriptors. Two 3D shapes are supposed to be similar if the distance between their shape descriptors is small. These methods are not well suited for the domain of architectural 3D models, as they are strongly characterized by topological properties of their underlying 2D floor plans rather than by local geometric 3D features.

While drafting, architects are particularly using floor plans, because those can easily demonstrate geometry and structure of a building. Taking all desirable drafting opportunities into consideration, floor plans are most unambiguous showing the spatial organization (development structure, topology and disposition of rooms, [Sch04]). Therefore they are a major ingredient for architectural drafting.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Copyright UNION Agency – Science Press, Plzen, Czech Republic.

For bottom-up-design of buildings the topology which means the arrangement of space is important, whereas for top-down-design the outer geometry always act as starting point. There exist basic characteristic floor plan geometries (for instance comb type, linear type, central type) that are also used to categorize a given collection of buildings [Neu05]. A floor plan often implies the scale of a building, the style (e.g. gothic, modernism, deconstructivism) and its content (as use, function, appropriateness), see [Pev79].

In this paper, we present a shape retrieval technique for 3D architectural models that is tailored to the specific requirements of architectural data and overcomes the limitations of common 3D shape retrieval algorithms. We propose the use of *room connectivity graphs* to characterize building models according to the topology of their underlying floor plans. In these graphs, rooms are represented by attributed nodes. Attributes might for example be the area and the extent of the room. Connections between rooms are represented by edges. These edges are also assigned attributes characterizing the type of room connection, i.e. door or window. For each floor of the underlying building, one such room connectivity graph is computed.

The extraction of the room connectivity graphs is not trivial as two major requirements need to be fulfilled to make the approach feasible for a large number of arbitrary 3D building models:

- **Format independence** Especially in the domain of 3D CAD modeling there exists an abundance of different file formats. Room connectivity graph extraction should therefore rely on a simple model representation that can easily be created from the underlying file format.

- **Robustness towards modeling errors** 3D meshes often contain T-vertices, intersecting triangles, wrong connectivity, etc. Additionally, we also encountered a lot of 3D building models in which separating walls between two rooms have gaps of several centimeters (see Figure 3a). These gaps were obviously not intended by the modeler and should therefore be eliminated.

To deal with these requirements, our method for the room connectivity graph extraction relies only on simple polygon soups that can easily be derived from almost any 3D representation. For each building story we compute several cuts between the polygon soup and a plane parallel to the ground. A 2D halfedge structure is built from the resulting line segments in which faces that correspond to rooms in the 3D building model can be determined. Unintended gaps between walls are closed using a technique that is inspired by topological simplification [SZL92]. Given one graph representation for each building story, determining those models in a database that are similar to a query object results in determining subgraph isomorphisms of the associated graphs with respect to node and edge attributes.

The contribution of our work is threefold. First, we introduce the room connectivity graph as a basic structure for 3D object retrieval in the architectural domain. Second, we describe a robust method to extract room connectivity graphs from polygon soup. This way we support most of the available 3D formats. Third, we show that the room connectivity graph concept is feasible for efficient and fast retrieval of architectural 3D models even in large databases.

2 RELATED WORK

In contrast to our approach, most methods for retrieval, classification and matching of shapes concentrate on arbitrary 3D objects instead of a special domain like architecture. In this section we focus on graph-based methods as they are most related to our approach. A more detailed introduction to 3D matching and shape retrieval can be found in [TV04].

Graph-based methods Graph-based shape retrieval methods rely on the extraction of certain geometric components and use a graph to show how these components are linked together. In contrast to our approach, these methods rely on low-level geometric 3D components rather than high-level semantic features like rooms, doors and windows.

Model graph-based approaches are mostly used for solid models like those in CAD applications. They describe solid objects in terms of connectivity of freeform surfaces (*Boundary Representation*) or as a set of geometric primitives that are connected by Boolean operations *(Constructive Solid Geometry)*. Local clique matching [EM03], [EMM03] or comparison of graph spectra [MPSR01] are used to determine the similarity of the graphs. In [ZTS02], VRML objects are segmented according to different decomposition techniques. The resulting patches are assigned basic shapes like planes and spheres. An attributed decomposition graph is built containing the determined shapes as nodes. Neighboring shapes are connected by edges. The similarity between two objects is computed by matching the associated decomposition graphs using error-correcting subgraph isomorphism.

Reeb graph-based methods rely on a function that is computed on the model surface which is supposed to be a compact manifold. The surface is divided into segments corresponding to intervals of this function. A skeleton graph in which the resulting segments are represented as nodes is built. Reeb graph-based methods are mainly used for matching of articulated objects [HSKK01], [TL07]. In [PSBM07], a robust method for fast Reeb graph computation is proposed that even allows for the use of non-manifold meshes.

Skeleton graphs of 3D shapes can be computed using topological skinning of voxel representations [BNdB99], medial axis transform, ridge point tracking [CSM05] or deformable model-based reconstruction [SLSK07]. Matching of two shapes is done by comparing the associated skeleton graphs using greedy bipartite graph matching [SSGD03] or by detecting subgraph isomorphisms using decision trees [LJI$^+$03].

View-based methods The idea behind view-based methods is similar to our approach in a way that the 3D shape retrieval problem is transformed to several problems involving data of lower dimension. First, a set of view-dependent shape descriptors is computed. The similarity between two 3D objects is determined by the comparison of the associated descriptor sets. Descriptors based on pure 2D information like silhouettes [DYCO03], [MD06] are used as well as descriptors based on 2.5D depth-buffer information [Vra04]. Unfortunately view-based methods in general concentrate on shape instead of topology and are therefore not suited for retrieval in the architectural domain.

3 ROOM CONNECTIVITY GRAPH EXTRACTION

The extraction of room connectivity graphs consists of three steps, the detection of the building stories, the determination of story rooms and finally the determination of doors and windows connecting these rooms.

The prerequisite for our method is a polygon soup representation of the building. The units of measurement in the model as well as the plane corresponding to the ground must be known. Note that this does not imply a severe restriction, as architectural models in general provide measurement information. The X-Y plane is the one corresponding to the ground in almost all cases.

3.1 Automatic Story Detection

The idea for the automatic detection of stories is given by the fact that in general all building stories are bordered by a flooring and a ceiling both of which result in patches that are parallel to the ground. The detection of floorings and ceilings according to these patches faces several problems. First, architectural models contain furniture and staircases that include patches which are also parallel to the ground. Second, the *floor construction*, i.e. the volume between a ceiling and the flooring of the room above, often contains additional large patches that are parallel to the ground, see e.g. the green planes in Figure 1. In the following we will show how to overcome these difficulties and identify those patches that correspond to floorings and ceilings.

First, we determine the set of polygons that are parallel to the ground. For each of these polygons we compute the distance to the ground. Those polygons that have approximately the same distance to the ground (we use a maximum deviation of 1mm) are put into a common bin. For each resulting bin we compute the sum of the included polygon areas. Patches caused by furniture or staircases are much smaller than those caused by floorings, ceilings and floor constructions. We therefore discard all bins with an area sum below a certain threshold t.

To get rid of those patches caused by the flooring construction, we use the fact that the thickness of flooring constructions is limited. It is generally determined by the span, the load and the construction material, see e.g. in [Hol07]. In buildings up to 4 floors the distance between a ceiling and the flooring of the room above usually is less than 0.4 meters, whereas multi-story buildings, industrial buildings or hall structures may have floor profiles up to 2 meters and more. As we are mainly dealing with few-floors residential buildings, we apply another binning scheme and collect all patches within the distance of 0.4 meters in one bin. Within one bin, that patch with the maximum distance to the ground represents a flooring (see e.g. the blue planes in Figure 1) and that with the minimum distance represents a ceiling (see e.g. the red planes in Figure 1). All patches in-between belong to the flooring construction and are therefore discarded. In case there is only one patch contained, it represents both, the flooring of the upper story and the ceiling of the lower story.

We encountered some building models in which the flooring of the lowest story is not modeled. We therefore check whether the lowest detected ceiling patch is further apart from the ground than the minimum common story height (i.e. about 2.40m). In case the building has no flat roof, we test if the distance of the highest flooring to the ground is smaller than the bounding box distance to the ground, which means that an attic is located above the highest flooring.

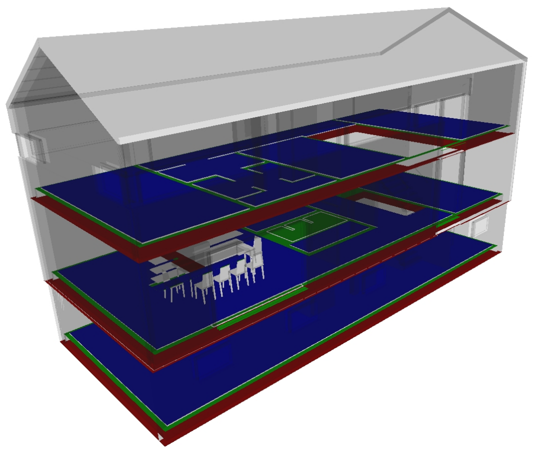


Figure 1: Building containing three stories. Red planes represent ceilings, blue planes represent floorings. The floor constructions to which the green planes belong are located between these planes.

3.2 Floor plan generation

As in architecture 2D floor plans are used to unambiguously show the spatial organization of a story, we restrict ourselves to the use of horizontal cuts instead of the 3D representation to efficiently extract rooms, windows and doors from building models. A horizontal cut of a 3D polygon soup results in a set of two-dimensional line segments that do not provide any connectivity. We transfer this set into a two-dimensional halfedge representation in which the line segments are connected with respect to their spatial relationship. Computing three cuts for each story at different height levels, we use the resulting halfedge structures to derive the room connectivity graph.

We represent each line segment that was created by the horizontal cut by two vertices v_i and v_j, a halfedge e_{ij} going from v_i to v_j and another halfedge e_{ji} going from v_j to v_i. We denote the set of vertices by V and the set of halfedges by E.

In the first preprocessing step, we determine the connectivity between the vertices. Two vertices v_i and v_j with $i \neq j$ are merged if the distance $d(v_i, v_j)$ between them is below a certain threshold ε (see Figure 2a). In our experiments we often encountered buildings containing double-sided modeled polygons, resulting in two line segments that are very close to each other. After the vertex merging it therefore might happen that the connection from vertex v_i to vertex v_j is represented by more than one halfedge. In this case we eliminate all halfedges from v_i to v_j but one.

In the second step, we determine the connectivity between vertices and halfedges. For each halfedge e_{ij} we determine the distance $d(e_{ij}, v)$ to every vertex $v \in V \setminus \{v_i, v_j\}$. If $d(e_{ij}, v)$ is below the above introduced threshold ε, e_{ij} and its opposite halfedge e_{ji} are split into two halfedges. The resulting halfedges are connected to the vertex (see Figure 2b). In all our experiments we used $\varepsilon = 1mm$.

In the next preprocessing step we eliminate intersecting halfedges. For all halfedges we check if it intersects with another halfedge. Intersecting halfedges e_{ij} and their opposite halfedges e_{ji} are split and a new vertex is inserted at the point of intersection (see Figure 2c).

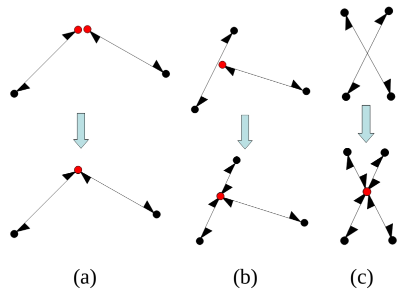


Figure 2: Line segment processing steps. a) Merging of close vertices. b) Halfedge split due to close vertex. c) Halfedge split and vertex insertion due to halfedge intersection.

Note that the resulting graph still contains unintended gaps due to modeling errors like that in Figure 3a. In the beginning of the room detection described in the next paragraph we do not take care of these inconsistencies but resolve them only after a first version of the room connectivity graph was constructed.

3.3 Room Detection

All rooms of one story are contained as faces in a halfedge graph constructed from a horizontal cut of the polygon soup that is located marginally below the story ceiling. At this height, no line segments corresponding to doors and windows will be contained in the cut. Given the halfedge graph representation of this cut, we determine the graph faces by walking along the halfedges, see Algorithm 1.

Considering the resulting faces, two cases must be distinguished. A face was either traversed inside (*inside face*) or outside (*outside face*). Inside faces correspond to either rooms or walls of the story building. Outside faces either represent rooms or walls lying completely inside another face without being connected to it or they correspond to the facade. To distinguish between inside and outside faces, we compute the angle between the incoming halfedge and the exiting halfedge with respect to clockwise direction for each of the n vertices during the face traversal. If all angles sum up to $(n-2)\pi$ (i.e.

```
Input Halfedge graph $G = (V, E, F)$

Function DetermineFaces(Graph G)
for all $e_{ij} \in E$ do
    if $e_{ij}$ was not visited yet then
        Create new face $f$
        Assign $f$ a pointer to $e_{ij}$
        TraverseHalfedge($e_{ij}$,$f$,$e_{ij}$)
        Add $f$ to $F$

Function TraverseHalfedge(halfedge $e$, face $f$,
                          starting halfedge $e_{start}$)
Mark $e$ as visited
Assign $e$ a pointer to $f$
$v_{target} \leftarrow$ target vertex of $e$
$e_{next} \leftarrow$ next outgoing halfedge of $v_{target}$ in clockwise
        direction with respect to $e$
if $e_{next} \neq e_{start}$ then
    TraverseHalfedge($e_{next}$, $f$, $e_{start}$)
```

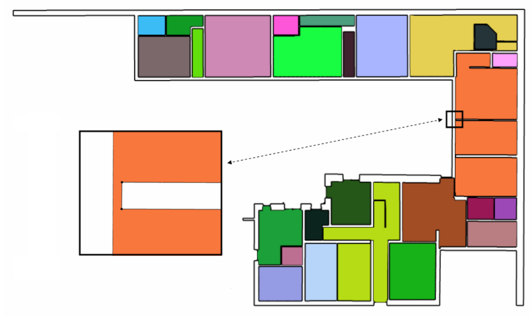
Algorithm 1: Face Determination

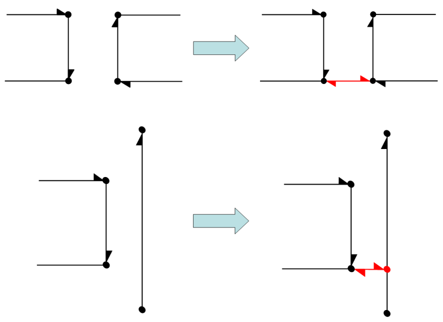
the sum of interior angles in a polygon), an inside face was determined, if the angles sum up to $(n+2)\pi$ an outside face was determined. After all faces have been extracted, one arbitrary vertex of each outside face is selected and it is tested if it lies inside one of the inside faces. By that we can determine faces corresponding to facades. Note that in case the building model consists of several tracts (see e.g. Figure 5), there also exist several facade faces for each of which we store a node in the room connectivity graph and assign it the attribute `facade`. All other outside faces are discarded.

The inside faces represent either rooms or walls of the story building. In the next step we identify those faces that belong to rooms. First, all faces with an area of less than $1m^2$ are discarded, as according to DIN guidelines (see [Neu05]) rooms are larger than this size in general. Although this eliminates much faces, there still remain faces representing walls. To deal with those faces, we additionally consider the room extent. In general, walls have an elongated shape while rooms correspond to rather quadratic-shaped structures. Considering a wall of width w with an area A and a quadratic room of the same area A, then the extents of the wall E_w and of the room E_r are given by

$$\begin{aligned} E_w &= 2\left(\frac{A}{w} + w\right), \qquad (1) \\ E_r &= 4\sqrt{A}. \end{aligned}$$

Common wall widths range from 6cm to 40cm. Using $w = 40$cm, the wall extent evaluates to $E_w = 5A + 0.8m$ which is much more than E_r for the remaining face areas with $A \geq 1m^2$. This implies that in general the ratio α between extent and area is larger for walls than for


(a) Unintended gaps between walls result in rooms that are not separated


(b) Gap-closing operations. Insertion of two halfedges between existing nodes and insertion of a new node with subsequent insertion of two halfedges.

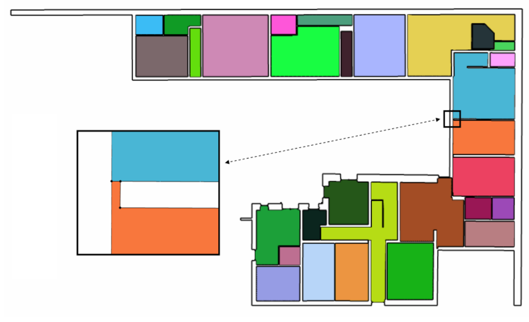

(c) The same as a) after the gap closing.

Figure 3: Gap-closing

rooms. To exactly determine up to which α value faces should considered to be rooms, we divided all faces of 20 halfedge graphs created from different stories by hand into rooms and walls and computed their α values. Using Bayesian decision theory [DHS01] we compute the a-posteriori probabilities for rooms and walls, given the ratio observation α:

$$\begin{aligned} p(room|\alpha) &= \frac{p(\alpha|room)p(room)}{p(\alpha)}, \\ p(wall|\alpha) &= 1 - p(room|\alpha), \end{aligned} \tag{2}$$

where $p(room)$ is given by the ratio between the total number of rooms and the total number of faces. To estimate the prior $p(\alpha)$ and the conditional probability $p(\alpha|room)$ we use the non-parametric Parzen window method [DHS01].

Given unclassified faces, we use the estimated a-posteriori probabilities to eliminate faces corresponding to walls. For each detected room a node is stored in the room connectivity graph and is assigned `area` and `extent` as additional attributes.

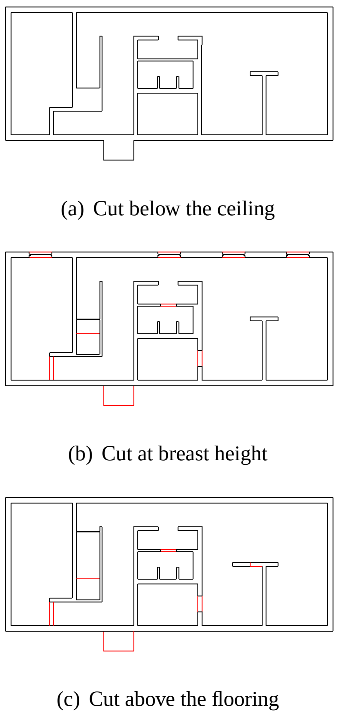
Figure 3a shows the problem of gaps between walls creating connections between rooms that actually were intended to be separated. In the data we found gaps ranging to a size of about 10cm. A naive approach to deal with this problem would be to increase the threshold ε that is used for the construction of the halfedge graph. By that, gaps up to a size of ε would be closed. The drawback of this method is that the larger ε is chosen, the more vertices are moved and the floor plan shape changes. Walls will disappear due to this simplification and structures like windows and doors will become more and more unrecognizable.

We follow a different approach that is inspired by topological simplification techniques which were invented for mesh simplification, see [SZL92]. While this method concentrates on determining simplification operations that preserve the topology, our algorithm identifies operations causing significant topology changes, i.e. closing gaps that create unintended room connections. A gap-closing operation can either be the insertion of two halfedges between two vertices or the insertion of a new vertex on a halfedge including the split of this halfedge and a subsequent insertion of two halfedges, see Figure 3b. We denote a gap-closing operation to be *valid* if it splits the face into two faces that still satisfy the room conditions (i.e. minimum area of $1m^2$ and $p(room|\alpha) > 0.5$). For each face we perform a sequence of virtual gap-closing operations, until a valid one is found and conducted. The resulting faces are tested for valid gap-closing operations recursively. We consider gaps up to a size of 375mm (minimum window width according to DIN, see [Neu05]), which is large enough to close all unintended gaps we found in our data and is still small enough to not close window openings unintentionally. By that, our method preserves the floor plan shape while handling all unintended gaps, see Figure 3c.

3.4 Door and Window Detection

For the detection of building elements that connect rooms, we compute two more horizontal cuts with the polygon soup. The first cut is located marginally above the story flooring, such that it will cut through the doors but not through the windows. The second cut is located at breast height (i.e. 1.40m above the flooring), such that it will cut through both, the windows and the doors. For both cuts, rooms are detected in the above described way. Windows and doors produce geometric inconsistencies in a wall. By determining the inconsistencies between the three cuts, these elements can be identified. Inconsistencies between the cut at breast height and the cut below the ceiling indicate the presence of either windows or doors. If a corresponding inconsistency is also found in the cut above the flooring, it indicates the presence of a door. In Figure 4 we show three cuts of one story at the different height levels and the inconsistent line segments that are contained in the high cut but not in the lower cuts.

Once the inconsistent line segments have been identified, we must determine whether they really correspond to doors or windows as there are also inconsistencies caused by other structural differences. Figure 4 shows that each door and window causes one pair of inconsistent segments in the two lower cuts. According to DIN standards (see [Neu05]), there exist minimum widths for windows (375mm) and doors (55cm), such that inconsistent segments caused by these elements must provide according minimum lengths. All pairs of inconsistent line segments are tested whether they provide these characteristics. Once a pair of segments is identified to be a door or a window, we determine to which faces the according halfedges in the cut below the ceiling belong to (note that this might also be the face representing the facade). We finally add an edge in the room connectivity graph connecting the room nodes representing the determined faces. As an attribute, the edge is either assigned `door` or `window`.


(a) Cut below the ceiling

(b) Cut at breast height

(c) Cut above the flooring

Figure 4: Cuts through the story of a building. In b) and c) those line segments that are only contained a) are shown in red. Inconsistent segments caused by doors are contained in both lower cuts whereas inconsistent segments caused by windows can only be found b).

4 RETRIEVAL WITH ROOM CONNECTIVITY GRAPHS

The room connectivity graph represents a basic structure for retrieval of building models in databases. With this representation at hand, similarity between an attributed query graph and a building model in a database can be determined by checking whether the query graph is isomorphic to a subgraph of the room connectivity graph of the building model, with respect to the attributes. Subgraph-isomorphism in general is known to be NP-complete [Coo71]. However, there are two aspects that make this approach feasible to this particular retrieval problem. First, the graphs we consider are in general not very large. Second, the attributes that are assigned to the nodes and the edges efficiently accelerate the graph matching process.

While determining a subgraph isomorphism between an attributed query graph and a room connectivity graph, we only match nodes and edges of the two graphs if their attributes are similar. We therefore use global constraints describing the amount of similarity the nodes or edges must provide such that they can be matched. Considering the edge attributes we introduced, they result in a constraint involving the edge type: Two edges can only match if they represent the same structure, that is either window or door. Considering the node attributes, they result in three constraints, that is: First, two nodes can only match if they represent the same structure, that is either a room or the facade, second, two nodes can only match if the area of their associated rooms differs only by a certain amount d_A of square meters and third, two nodes can only match if the extent of their associated rooms differs only by a certain amount d_E of meters. Although we restricted our experiments to these four constraints, the attributes introduced so far allow for the construction of further constraints involving for example the ratio between area and extent. Note that all constraints are global, i.e. they count for all the nodes and edges, respectively.

In case the graphs to be matched both provide a facade node, these nodes should be matched first to further accelerate the retrieval. If no such node exists in the query graph, an arbitrary node can be chosen.

5 RESULTS

For our experiments we used a database consisting of 100 residential building models that were created by students of architecture. For the room connectivity graph extraction all thresholds were chosen as described above. The threshold t for the automatic story detection was set to $10m^2$. The total processing of the whole database took about 7 minutes (see Table 1) and created 289 room connectivity graphs, one for each story. In Figures 5 to 9 examples of these graphs are shown. Each colored face represents a room node in the graph. Edges between room nodes are shown including the type connection (either `door` or `window`). Figure 9 shows a problem we discovered in some building models. The arrow points to a structure representing a window. In contrast to the other windows in this story, the window pane and the facade are exactly in one line. Therefore, the wall structure provides no inconsistency and the window was not detected. Timings for the extraction of the depicted room connectivity graphs can be found in Table 1.

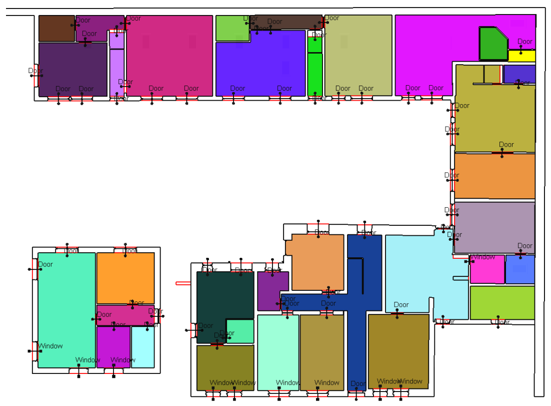


Figure 5: Ground floor of a complex building containing two tracts. The resulting room connectivity graphs consists of two connected components.

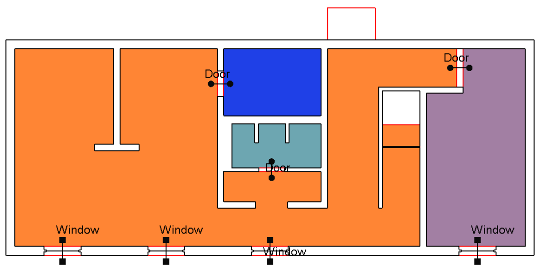


Figure 6: Cellar of a residential building.

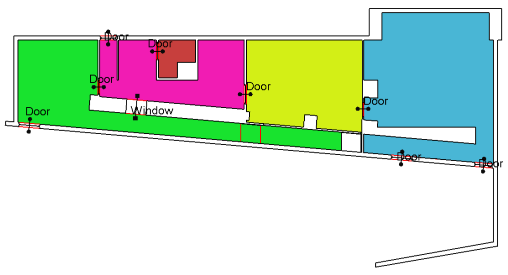


Figure 7: Ground floor of a residential building.

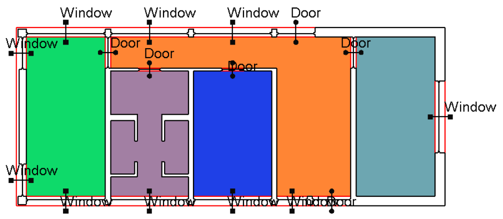


Figure 8: Ground floor of a residential building.

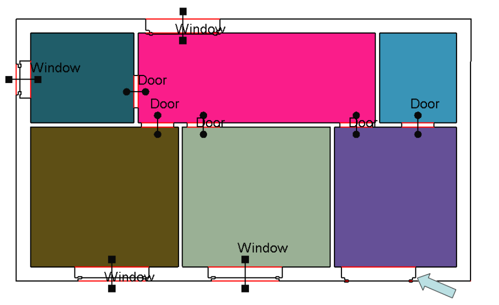
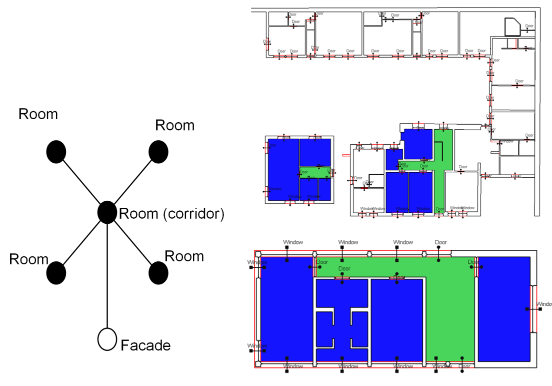


Figure 9: First floor of a residential building. The arrow points to a window that could not be detected by our method as the window pane is positioned exactly in one line with the facade.

Object	#Graphs	Extraction	Retrieval
Figure 5	2	10.451s	< 10ms
Figure 6	3	0.967s	< 10ms
Figure 7	4	4.828s	< 10ms
Figure 8	3	1.077s	< 10ms
Figure 9	2	0.421s	< 10ms
Database	289	436.512s	0.173s

Table 1: Statistics on room connectivity graph extraction. *#Graphs* describes the number of room connectivity graphs that were extracted from the underlying 3D model. The time that was spent for the extraction is given by *Extraction*. *Retrieval* states how long it took to search the extracted room connectivity graph for the query graph shown in 10a. All experiments were run on an AMD Athlon^{TM} 64 with 2 GHz and 2 GB RAM.

Figure 10 shows results of our retrieval experiments. As query we used a graph representing the structure of a typical apartment including a corridor that leads to four rooms (Figure 10a). The results shown in Figure 10b were generated using the constrained subgraph matching as described in Section 4 without considering the area and extent attributes. In other experiments we also included these attributes and could thereby further refine the retrieval result. The time consumption for searching all 289 room connectivity graphs for the query graph in Figure 10a was 0.173s (see Table 1 for more details).



(a) (b)

Figure 10: a) Graph representing a typical apartment. All edges represent doors. b) Amongst the floor plans in Figure 5 to 9, the query graph was found twice in the floor plans of Figure 5 and once in that of Figure 8. The corridor room is shown in green, the other rooms are shown in blue.

6 CONCLUSION

In our work we presented a new method to efficiently characterize architectural data by room connectivity graphs. The method only relies on polygon soup representations and is therefore feasible for a large number of 3D models. Our approach is robust towards modeling errors like unintended gaps between rooms. In our experiments we showed that the extracted room connectivity graphs can be used for fast and efficient shape retrieval in architectural databases.

Additionally to the room area and extent it would also be interesting to include attributes that describe the shape of the room more precisely. 2D descriptors like Zernike moments or centroid distance-based ones could therefore be used to further enhance retrieval results. Future work should also consider building units that connect different stories. By that, the room connectivity graphs of each story will be interlinked by edges representing staircases and elevator shafts.

ACKNOWLEDGEMENTS

This work was partially funded by the German Research Foundation (DFG) under grant GZ 554975(1) Oldenburg BIB 48 OLof 01-02 *Probado*.

REFERENCES

[BNdB99] Gunilla Borgefors, Ingela Nyström, and Gabriella Sanniti di Baja. Computing skeletons in three dimensions. *Pattern Recognition*, 32(7):1225–1236, 1999.

[Coo71] Stephen A. Cook. The complexity of theorem-proving procedures. In *STOC '71: Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158, New York, NY, USA, 1971. ACM Press.

[CSM05] Nicu D. Cornea, Deborah Silver, and Patrick Min. Curve-skeleton applications. In *IEEE Visualization*, page 13. IEEE Computer Society, 2005.

[DHS01] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. Wiley, 2001.

[DYCO03] M. Ouhyoung X.-P. Tian Y.-T. Shen D.-Y. Chen and M. Ouhyoung. On visual similarity based 3d model retrieval. In *Eurographics*, pages 223–232, Granada, Spain, 2003.

[EM03] M. El-Mehalawi. A database system of mechanical components based on geometric and topological similarity. part 2: indexing, retrieval, matching, and similarity assessment. *Computer-Aided Design*, 35(1):95–105, January 2003.

[EMM03] Mohamed El-Mehalawi and Allen Miller. A database system of mechanical components based on geometric and topological similarity. part 1: representation. *Computer-Aided Design*, 35(1):83–94, January 2003.

[Hol07] Klaus Holschemacher. *Entwurfs- und Berechnungstafeln*. Bauwerk Verlag GmbH, Berlin, 2007.

[HSKK01] M. Hilaga, Y. Shinagawa, T. Kohmura, and T. L. Kunii. Topology matching for fully automatic similarity estimation of 3D shapes. In *Proceedings of ACM SIGGRAPH*, 2001.

[LJI$^+$03] K. Lou, S. Janyanti, N. Iyer, Y. Kalyanaraman, S. Prabhakar, and K. Ramani. A reconfigurable 3d engineering shape search system part ii: database indexing, retrieval and clusturing. In *DETC*, 2003.

[MD06] Ameesh Makadia and Kostas Daniilidis. Light field similarity for model retrieval. In *SHREC2006, 3d Shape Retrieval Contest*, June 2006.

[MPSR01] David McWherter, Mitchell Peabody, Ali C. Shokoufandeh, and William Regli. Database techniques for archival of solid models. In *SMA '01: Proceedings of the sixth ACM symposium on Solid modeling and applications*, pages 78–87, New York, NY, USA, 2001. ACM Press.

[Neu05] Ernst Neufert. *Bauentwurfslehre*. Vieweg, 38th edition, September 2005.

[Pev79] Nikolaus Pevsners. *A History of Building Types*. Princeton University Press, December 1979.

[PSBM07] Valerio Pascucci, Giorgio Scorzelli, Peer-Timo Bremer, and Ajith Mascarenhas. Robust on-line computation of reeb graphs: simplicity and speed. *ACM Trans. Graph.*, 26(3):58, 2007.

[Sch04] Friederike Schneider. *Floor Plan Manual Housing*. Birkhäuser, third edition, April 2004.

[SLSK07] Andrei Sharf, Thomas Lewiner, Ariel Shamir, and Leif Kobbelt. On–the–fly curve-skeleton computation for 3d shapes. In *Eurographics*, pages 323–328, Prague, september 2007.

[SSGD03] H. Sundar, D. Silver, N. Gagvani, and S. Dickinson. Skeleton based shape matching and retrieval, 2003.

[SZL92] William J. Schroeder, Jonathan A. Zarge, and William E. Lorensen. Decimation of triangle meshes. *Computer Graphics*, 26(2):65–70, 1992.

[TL07] Gary K. L. Tam and Rynson W. H. Lau. Deformable model retrieval based on topological and geometric signatures. *IEEE TVCG*, 13(3):470–482, 2007.

[TV04] J. W. Tangelder and R. C. Veltkamp. A survey of content based 3d shape retrieval methods. In *SMI '04: Proceedings of the Shape Modeling International 2004 (SMI'04)*, pages 145–156, Washington, DC, USA, 2004. IEEE Computer Society.

[Vra04] Dejan V. Vranić. *3D Model Retrieval*. PhD thesis, University of Leipzig, Germany, 2004.

[ZTS02] Emanuel Zuckerberger, Ayellet Tal, and Shymon Shlafman:. Polyhedral surface decomposition with applications. *Computers and Graphics*, 26(5):733–743, October 2002.