FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Interactive Exploration of Architecture Using Exploded Views

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Medieninformatik

eingereicht von

Felix Fleiß
Matrikelnummer 0527038

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: Associate Prof. Dipl.-Ing. Dipl.-Ing. Dr.techn. Michael Wimmer
Mitwirkung: Projektass.(FWF) Dr.techn. Dipl.-Mediensys.wiss. Przemyslaw Musialski

Wien, 19. April 2015

(Unterschrift Verfasser) (Unterschrift Betreuung)

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Interactive Exploration of Architecture Using Exploded Views

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Media Informatics

by

Felix Fleiß
Registration Number 0527038

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Associate Prof. Dipl.-Ing. Dipl.-Ing. Dr.techn. Michael Wimmer
Assistance: Projektass.(FWF) Dr.techn. Dipl.-Mediensys.wiss. Przemyslaw Musialski

Vienna, 19th April 2015

(Signature of Author) (Signature of Advisor)

Erklärung zur Verfassung der Arbeit

Felix Fleiß
Linzerstraße 94, 3003 Gablitz

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum) (Unterschrift Verfasser)

Acknowledgements

I want to thank my family for supporting me. Furthermore, I want to thank all test users for their time, patience, and feedback. In memory of my beloved brother David.

Abstract

The master thesis at hand addresses automatically generated exploded views as a tool of visualization and how they can be applied to illustrate architecture. Therefore, I investigate existing solutions of the generation and the related theoretical foundation of explosion views, assembling processes, architecture and visualization. Then I extract useful concepts from the foundation research to deduce design principles. The major part of this work is the designing and implementation of a visualization system that empowers the user to interactively explore and understand an architectural building. Hereby, I examine to what extent explosion views can be applied to do so.

The work at hand involves many different scientific domains. It combines architecture, visualization, and user interface design. The main challenge of this work is how to present information about a building. This does not only concern the geometrical representation, but also meta information, like relations and memberships of building parts. All this information shall be communicated mostly in an implicit way.

Kurzfassung

Die vorliegende Diplomarbeit befasst sich mit automatisch generierten Explosionszeichnungen als Mittel der Visualisierung und wie diese verwendet werden können, um Architektur darzustellen. Dabei untersuche ich sowohl existierende Lösungen für deren Generierung, als auch die damit in Zusammenhang stehende theoretische Grundlage für Explosionszeichnungen, Montagevorgänge, Architektur und Visualisierung. Aus dem Resultat der Grundlagenrecherche werden brauchbare Erkenntnisse extrahiert, um daraus Design Prinzipien abzuleiten. Ein großer Teil der Arbeit ist das Entwerfen und Implementieren eines Visualisierungssystems, das den User dazu bemächtigen soll interaktiv ein architektonisches Gebäude zu erforschen und zu verstehen. Dabei wird untersucht, inwiefern Explosionszeichnungen dafür geeignet sind.

Die vorliegende Arbeit umfasst viele verschiedene wissenschaftliche Bereiche. Sie kombiniert Architektur, Visualisierung und User-Interface Design. Der Hauptaspekt der Arbeit ist es, Information über ein Gebäude darzustellen. Dies betrifft nicht nur eine geometrische Darstellung, sondern auch Metainformation, wie Beziehungen und Zugehörigkeiten von Gebäudeteilen. All diese Information soll größtenteils implizit kommuniziert werden.

Contents

CHAPTER 1

Introduction

1.1 Motivation

Architecture is an essential aspect of our society. It shapes the environment where the lives of humans take place and, therefore, has a great impact on them. First of all, the term architecture describes the building itself as its physical structure. These are the concrete physical objects which surround us, the occupants and users. The building thereby serves as a refuge and offers protection from the environment. This may be wildlife, the weather, or acoustic and visual disruption. Secondly, the term architecture refers to the art or science of designing a building. This point of view describes and determines the form and the function of buildings and building elements. Finally, the term architecture defines the process of planning and constructing a building. The realization of a building is a time-consuming and costly development. It consists of many dependent steps in which many people are involved. All three concepts require a proper visualization of the building to facilitate a non-ambiguous and valuable communication. The appliance of architecture visualization can communicate an idea and concepts of a building, transport structure information, reveal errors of the constructed building and, therefore, avoid problems and reduce costs.

There exist many different techniques to visualize architecture. Some are very common, like floor plans, side views, cuts, or renderings. Others are quite unconventional, like exploded views. In particular, exploded views have a couple of advantages compared to conventional visualization techniques. First of all, they are a valuable tool to show the interior of complex objects. This is important for a viewer of architecture to understand the contained structure. On the other hand, no information is left out in exploded views. The locations of parts of the considered object are just rearranged to reveal parts of interest. This way, the viewer can mentally reconstruct the rearranged parts to the complete object, while the single parts are displayed in their original forms. Spatial relationships are preserved via the visualization of the path of a rearranged part to its original position.

The potential user of an exploded view of a building is everybody who engages in architecture. This may be a person who is unfamiliar with an existing building and wants to find a particular room. This also may be a future occupant of a building. She/he wants to get an impression of the complete building, the arrangement of rooms, furniture, doors and windows. However, the group of potential users also contains all other persons who are involved in the process of planning and constructing a building. In this domain, exploded views could be used to reveal details like the composition of multi-layered elements, the statical system of a building, or water pipe and electric wire installations.

These are just a few examples of potential use cases. As one can recognize, the scope of application is wide-ranging. Despite this fact, there is just a moderate amount of literature addressing exploded views in architecture. Although some illustrators experiment with exploded views to visualize a building, there is no proper formalism for this technique in the domain of architecture. In this work, I try to find a remedy for this lack by an investigation of existing illustrations and approaches, and by designing and defining a formalism.

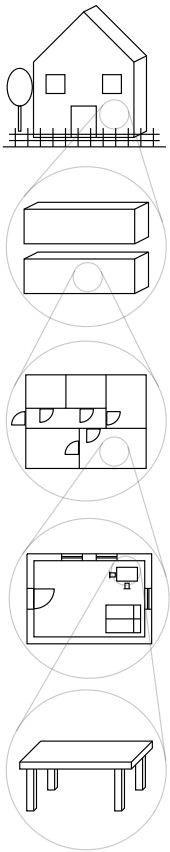
1.2 Problem Definition

Architecture is a challenging domain for visualization because a building contains structures and elements at widely diverged scale. This means that the building includes very small elements like furniture (chairs, tables, etc.), and also very big parts like the façade of the building.

Not only the different scale of the building elements, but also the great amount of them has to be faced in architectural visualization. A building consists of many elements, e.g., furniture, rooms, floors and coverings, like walls and slabs. Each of these elements may occlude others. This is crucial for the general problem of occlusion in visualization, because if there is an increased number of objects in 3D to display on a 2D-screen, also the number of occlusions rises. These occlusions have to be handled to reveal occluded elements of interest.

However, an architecture not only consists of many building elements, it also contains meta information about the function and the state of its elements. For example, the function of an elevator is to transport persons from floor A to floor B. On the other hand, its state includes the current location of the elevator, e.g., floor B. Furthermore, there exist relationships and dependencies between the building elements. For instance, a wall separates two rooms, or a floor contains multiple rooms. In the following, I will refer to these relationships as the context of building elements. In order to visualize a building, many relevant details should be highlighted, but also structural and relationship information should be transported by the visualization.

An architectural building is a complex structure. Accordingly, one has to take a look at the different levels of detail. This is handled by a top-down approach.



- At the topmost level, there is the building as a whole. One is interested in the outer appearance, i.e., the façade, entrances, roofs, etc., and its relation to the environment.
- Next are the building sections. At this level, the functionality and independent parts of the structure of a multifunctional building are divided. For example, a building consists of a restaurant and an apartment-building section.
- At the next level, the floor plan of a building is inspected. One is interested in the accessibilities of the rooms and correlations among them, e.g., their sizes, locations, and arrangements.
- The next level of detail relates to the room. One is interested in the relationships between fixtures, furniture, doors and windows, and the space between. This approach is called interior design.
- At the lowest level, there is the single building element or furniture element.

Figure 1.1: Levels of detail

A single static illustration cannot satisfy the requirements of all these different levels of detail. My approach is to develop an interactive visualization system that enables the user to view the architecture at all these levels and, therefore, allows her/him to explore the interior of the building. Therefore, smooth transitions from one level of detail to another, but also from one viewpoint to another, are needed to preserve context while exploring the architectural domain.

In order to convey the internal structure of such complex objects consisting of many parts, "illustrators often create exploded views in which parts are separated (or "exploded") away from one another to reveal parts of interest" [LACS08]. An exploded view must balance compactness and separation [ALB11] by moving occluding parts away into a proper direction with a suitable offset. The viewer shall be empowered to recognise how the exploded parts belong together. Nevertheless, exploded parts shall be separated from the part of interest to offer an unhindered sight onto it. An exploded view is also a valuable tool to show the context of elements. The advantage of exploded views compared to alternative visualization methods is that the related context elements are not masked or made transparent. They are just moved out of the way to reveal the part of interest. This way, the viewer can mentally reconstruct the exploded object while perceiving every detail of the part of interest and its context.

Figure 1.2: Here is an example of an exploded view of a building. The image is taken from the article [Tho06] by courtesy of the author and the illustrator.

1.3 Aim of the Work

The aim of the work at hand is to implement a visualization system that uses exploded views to handle occlusions while preserving the context of elements. The resulting visualization system shall automatically generate interactive visualizations that communicate an overall picture of an architectural building and simultaneously provide a mechanism to gain a detailed view of parts of interest. Furthermore, the system shall enable the user to interactively explore the architecture to gain insight into the building and its structure. The user shall be empowered to explore a building seamlessly at any level of detail mentioned above. Therefore, smooth and adequate animations of the camera and of the motion of exploding parts are needed. The system shall provide a meaningful exploration experience for a building. A meaningful visualization has to omit or de-emphasize unnecessary details [ALB11], decrease visual clutter [LACS08] and reveal information of interest. This way, the user can understand the space, functionality, relations of

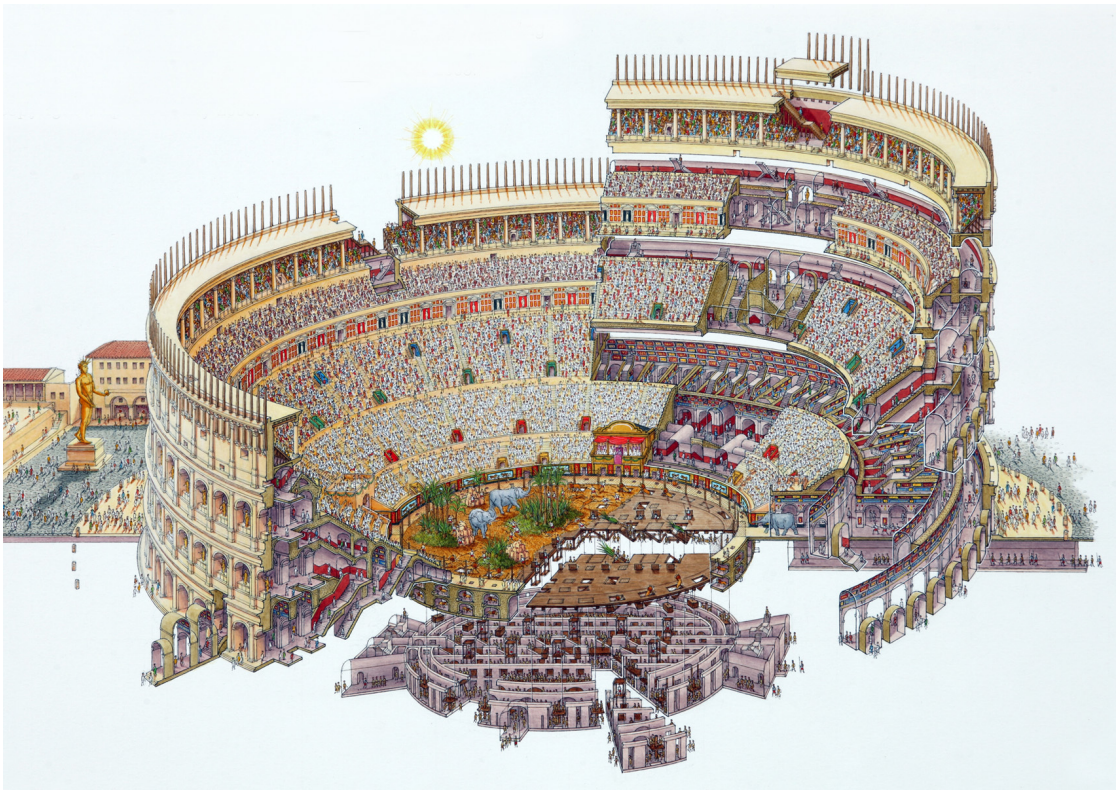


Figure 1.3: The figure shows an illustration by Stephen Biesty. He uses the technique of exploded view and cuts to reveal the interior of the Colosseum. Illustration copyright 2003 Stephen Biesty.

parts and finally the building itself.

In order to support the requirements of the visualisation and of the exploration of the building, I propose a hierarchical architectural model of the building. For this, a building is subdivided into the following parts: A building consists of building sections that are divided into floors. Floors contain rooms and these contain furniture.

Furthermore, proper design principles shall be extracted from existing literature and sample illustrations. Design principles "explain how visual techniques can be used to either emphasize important information or de-emphasize irrelevant details", they are "guidelines that help improve viewers' comprehension of visually encoded information" [ALB11]. They consist of hard constraints (for instance, only use orthogonal explosion directions)and evaluation criteria (for example, balance part separation and compactness). They are implemented by a set of formal design rules that can be applied to an automated system.

The hierarchical segmentation of the architecture and the use of customized local design rules at each level of the hierarchy shall satisfy the particular requirements of the different levels of detail mentioned above. These customized local design rules implement design principles

which ensure a valuable visualization. Local design rules define rules at a particular level of the hierarchical model, for instance, to separate floors by exploding them vertically.

At a later stage of this work, it shall be attempted to generalize the developed local design rules and transform them to global design rules. A global design rule is valid at each level of the hierarchical model. The following example represents a global design rule: a part of interest must not be occluded by any other part. The resulting design rules shall provide concepts to generate valuable visualizations of a building. This includes design rules to determine explosion directions (i.e., direction to move away a part from the part of interest), explosion offset (i.e., distance to move) and explosion order of architectural elements. Moreover, one can also find rules to illustrate explosion paths and the shape of spaces of the building among them.

Furthermore, it shall be investigated whether exploded views are a convenient tool for architectural visualization and what adaptations to state-of-the-art methods for exploded views have to be applied. The created design principles shall be evaluated. For this, the implemented system shall be tested via an informal user test, and the resulting feedback shall be documented.

It is neither an aim of the work to provide a detailed visualization of the technical construction of a building, nor to show every material component of a building element. There are conventional architectural visualization techniques that are already doing this well, like floor plans, detailed views or section views. In fact, the result of this work shall be considered as an alternative visualization technique for serving potential occupants or users of a building. It shall visualize a building while reducing the weaknesses of other existing methods, like masking, transparency or a Walk-Through. These weaknesses are the loss of context, visual clutter and not gaining an overall picture.

1.4 Methodological Approach

In order to gain insight into the complex of problems of exploded views, I investigated the literature of this domain. For the purpose of the fulfilment of architectural requirements, I adapted and extended existing approaches and principles for exploded views to provide methods to generate valuable visualizations. For this, I implemented the approach of Li et al. [LACS08], which operates on technical assemblies. I will demonstrate why additional concepts are needed for the application of exploded views to architecture.

For the purpose of understanding the central issues of architecture, I considered two architectural theories, the “theory of space” and “space syntax”. Furthermore, I informally discussed and extended the extracted information together with two architects and a student of architecture. I exploited the concept of “space and hull” to develop a hierarchical arrangement of spatial structures as a foundation for the method of the work at hand. The method takes into account previous approaches of exploded views and architectural theoretical aspects to generate context-sensitive exploded views that shall preserve the relationships of building parts.

The gathered information was used to deduce design principles for the generation of exploded views of buildings. I developed and generalized design rules that implement these design principles. A visualization system was realized that complies with these design principles and

was used to evaluate them. For this, I conducted an informal qualitative usability test. It was tested whether the system provides an adequate tool for the user to explore and understand a building, its interior structure, and its spaces.

CHAPTER 2

Foundations and Related Work

In this chapter, I will present concepts of the found literature that are related to the topic of this work. First of all, I will handle assembly planning because many approaches of automated exploded view generation are based on it. Then, I consider automated exploded view generation for technical assemblies due to the lack of the use for buildings in literature. Afterwards, I search for a proper architectural theory to be able to deduce design principles and a semantic building model. In the end, I deal with papers that address visualization and architectural illustrations.

2.1 Exploded Views

Exploded Views are an often used visualization technique to illustrate assemblies. This technique reveals the hidden and occluded information of a complex 3D structure by decomposing it to smaller parts and transforming them via translation, rotation, and so on. Thereby, a part of interest is exposed, but at the same time, no information about the context has to be omitted.

> Well designed exploded views not only expose internal parts, they also convey the global structure of the depicted object and the local spatial relationships between parts. Furthermore, unlike other illustration techniques that reveal internal parts in situ by removing or de-emphasizing occluding geometry, such as cutaways and transparency, exploded views show the details of individual parts [LACS08].

An exploded view can be considered as the disassembly of a product and, therefore, as the reciprocal process of assembling [YXBW14]. Thus, most approaches of automated generation of exploded views use the concepts and methods of automated assembly planning. Assembly planning determines how and in which order parts have to be mounted to a product [Wil92]. This supports the process of designing because of the immediate feedback of assembling steps without the need of a prototype [RGGR95].

2.1.1 Assembly Planning

Wilson [Wil92] intensively engaged in the theory and concepts of assembly planning. He developed a method that automatically plans the process of assembling. It only takes into account the geometry and the structure of an assembly without considering the influence of the environment on the assembly process itself. He uses an inverse approach to remove unblocked parts from the product and, therefore, his concepts are the foundation for many automated exploded view implementations of assemblies. In order to do so, he defines the terms "local freedom" and "global freedom". Local freedom determines the directions into which a part is not blocked by other parts in contact. A connection graph is used to represent the contact relation information of all parts of the assembly to each other. Global freedom, on the other hand, defines the directions into which a part can be translated infinitely without intersecting any other part of the assembly. Sweeping a part into a given direction can be used to check global freedom of the part in this direction. The set of directions of the global freedom is a subset of the directions of the local freedom (see Figure 2.3). Wilson [Wil92] uses spheres to represent these possible directions. In doing so, a cone determines the directions of freedom. The author also considers other operations for removing parts than translation, for example, rotation. He constructs an AND/OR-graph that represents the set of all possible assembly sequences for a product. For this, the parts of the assembly are iteratively decomposed into two sub-groups that can be separated in a feasible assembly step. A node of the graph represents a sub-assembly of the product. The root is the final assembly, whereas the leaves are all possible sub-assemblies with only one part remaining. An edge of the graph represents a single decomposition step. Wilson also demonstrates examples of assemblies that fail with this approach as shown in Figure 2.1.

Romney et al. [RGGR95] extend the work of Wilson [Wil92] and present a system (STAAT) that automatically determines how to assemble the parts of a product, given only a geometric description of the assembly. The system uses a non-directional blocking graph to represent the combined blocking information of all parts of the assembly. At the date of the publishing of their paper, STAAT was limited to assembly sequences that are binary, monotone, and composed of single-step translations. Monotonicity describes the fact that a part can be removed from the assembly without the need of traversing intermediate positions. Binary means that the result of one disassembling step does not contain more then two sub-assemblies. Consequently, the system can not cope with the cases shown in Figure 2.1. The system also supports the analysis of the ease of part removal. For this, it finds the shortest disassembly sequence to extract a given part. Local translational freedom (LTF) cones are generated by combining the contact information of parts (see Figure 2.2). These cones represent the possible disassembly directions for each part and are combined to the non-directional blocking graph representing the blocking information for the whole assembly. In order to obtain the cones, a sphere representing all directions is subdivided into different regions deduced from the local freedom. Candidate directions of disassembling a part are intersections of boundary arcs of these regions. The global freedom of a considered part is tested by sweeping or projecting a part into a given candidate direction and checking for collisions.

Agrawala et al. [APH+03] also exploit the work of Wilson [Wil92] and use his concepts for a visualization system. They present design principles and a system that automatically generates

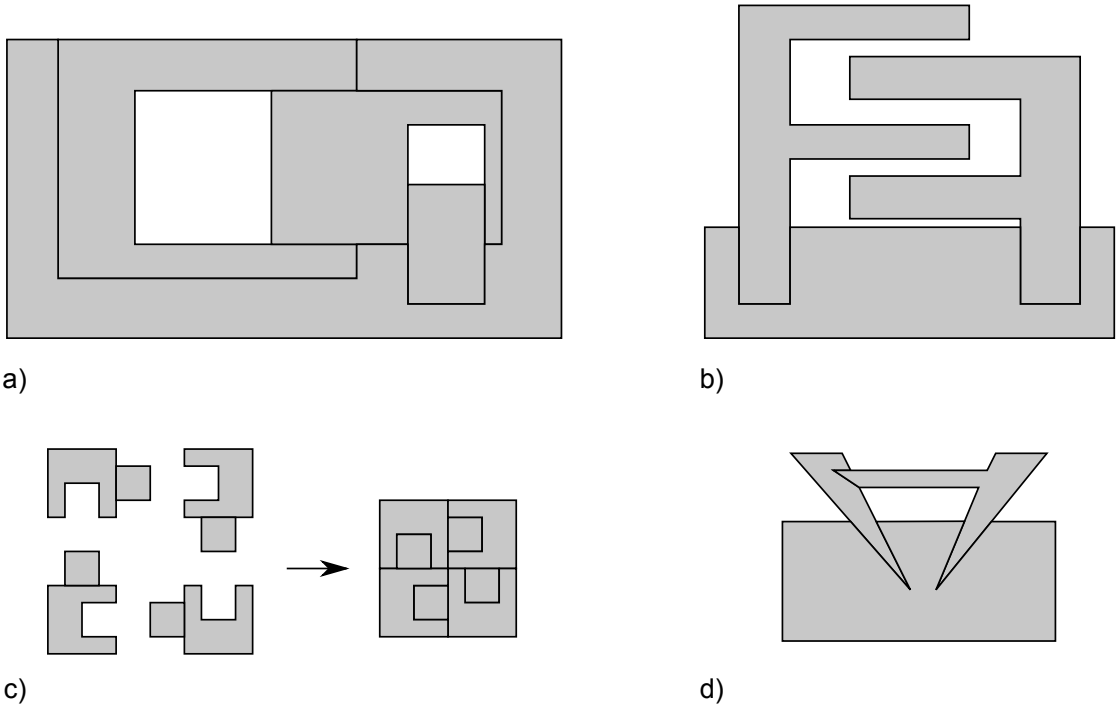


Figure 2.1: These examples demonstrate cases in which no single part can be assembled via one single operation. In a), multiple steps are needed to assemble the most inner part. In b), c), and d), multiple parts have to be assembled simultaneously. The figures are just reproductions drawn by me. The original images can be found in Wilson's "On Geometric Assembly Planning" [Wil92].

static step-by-step assembly instructions of everyday objects, such as furniture, appliances, and toys. The system is divided into a planner and a presenter. The planner computes the assembly steps while the presenter renders the sequence of assembly steps as series of structural diagrams or action diagrams. In a structural diagram, all already assembled parts are rendered at their final position. On the other hand, in an action diagram new added parts are rendered at the position from where they have to be inserted. The insertion direction is displayed via an arrow pointing towards the final position. The main task of the planner is to evaluate the constraints of blocking, visibility, etc. for each subset of not yet added parts. Afterwards, it chooses the next parts to assemble based on the evaluation of these constraints. Moreover, the planner also handles special cases like omitting repetitive operations, reinserting fasteners, or the reorientation of the camera to improve the visibility of parts. The planner uses blocking constraints and grouping information of parts of the assembly to determine the order of assembling parts. Only local translational freedom is applied to do so. Furthermore, the planner ensures visibility of new added parts. The authors state that visibility is a very important design principle. In order to ensure the visibility of a new added part P, their system renders P without and then together with the already assembled parts and counts the pixels of the frame-buffer covered by P for both

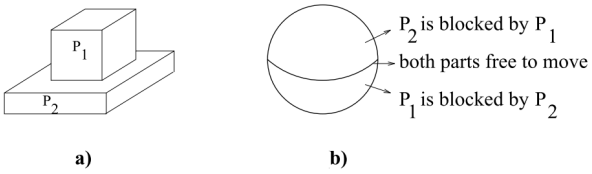


Figure 2.2: a) shows two parts in contact. b) demonstrates the LTF cones for these two parts. The image is taken from [RGGR95] by courtesy of the author.

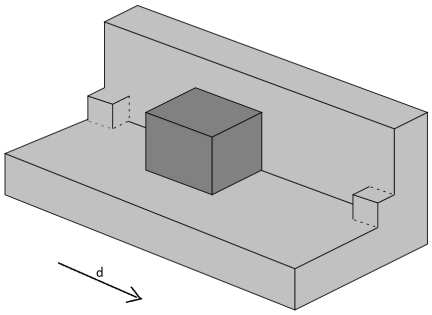


Figure 2.3: The figure shows a locally free but globally blocked relationship between the two displayed parts along the direction d. The image is taken from [RGGR95] by courtesy of the author.

steps. The relation between these two counts is calculated and used to determine the quality of the visibility. However, also the already assembled parts shall be visible, therefore, the authors also handle this requirement. On the other hand, the presenter renders the assembly steps. For a structural diagram, as described above, the presenter has to choose the separation direction of new added parts from the set of possible directions that are locally free. It yields the one separation direction into which the new added part moves the smallest distance from its assembled position to the outside of the bounding box of the already assembled parts. A fixed value is used for the separation distance. However, for adding multiple parts during one step, a stack of these parts is built to additionally separate them from each other.

In order to determine the local translational freedom of a part, Agrawala et al. [APH+03] use an algorithm presented by Lin et al. [LC91]. The algorithm finds the nearest points between two convex polyhedrons in roughly constant time. If the distance between the nearest points is zero, the polyhedrons are in contact. By comparing the neighboured features (vertices, edges and faces) of a polyhedron with respect to the distance to the second polyhedron, the method "walks" around the geometry from one nearer feature to another.

2.1.2 Automated Generation of Exploded Views

Many approaches of automated generation of exploded views rely on assembly planning. Consequently, these approaches need to take into account blocking relations among the parts of an assembly. Blocking relations are needed to determine if a part P_i can be removed from the assembly without being blocked by any other part and to determine the disassembly direction. In the following, the disassembly direction is also referred to as the explosion direction. The calculations that yield the blocking relations of parts of the assembly are very time-consuming and are needed multiple times. Therefore, it is common practice to pre-calculate and store this information. One method to store it is the use of interference matrices. The interference matrix

is a squared matrix. One interference matrix M_D only holds the interference information for one given disassembly direction D, which is usually one of the coordinate directions. Every element E_{ij} of a matrix M_D represents the interference relation of two parts of the assembly P_i and P_j. If E_{ij} is zero, there is no interference between the part P_i and P_j in the direction D, else it is one. Each column E_i describes the status of interference for one particular part P_i to each other part in the assembly. If all elements of a column E_i of a matrix M_D are equal to zero, the part P_i can be removed from the assembly into the direction D without being blocked by any other part. When one adds all the information of all interference matrices M_{D_k} together, the result is the local/global freedom of all parts of the assembly for a set of given disassembly directions D_k.

Yu et al. [YXBW14] complement the interference matrices to Extended Interference Matrices (EIM) and apply them to automatically generate exploded views of assemblies. The authors overcome the limitation of previous concepts of generating exploded views, which are restricted by the use of orthogonal axis-aligned explosion directions. With their approach, also oblique or inclined directions are possible. Whereas conventional interference matrices only use the global coordinate system axis as disassembly directions, the EIM also takes into account the local coordinate system (x_i, y_i, z_i) of a considered part P_i of the assembly to determine the disassembly direction and to deal with inclined assembly interferences. Accordingly, the set of possible disassembly directions for a part P_i is extended by the directions D_{x_i}, D_{y_i}, and D_{z_i}. The elements of an EIM are called friction factors. The friction factor F_{ij} qualifies the kind of contact of two parts P_i and P_j by moving P_i along the direction D. Parts in contact are called assembly mates (i.e.: “surface coincidence”, “axial coincidence”, and “screw mate”). The friction factor represents these relations and can be used to favour a particular explosion direction. The authors extend the design principles visibility and compactness by the restriction of the consistence with sequence:

> An ideal exploded view should be a visual carrier of the disassembly sequence. Therefore, an exploded view should have each component placed according to their assembly/disassembly sequence and directions [YXBW14].

In order to reduce the complexity of the computation of the disassembly sequence, Xing [Xin12] applies the ant colony optimization (ACO) [DBS06]. He presents a different technique for exploded view generation also based on disassembly sequence planning. The idea of his technique is that only the parts of the assembly without any interference can be disassembled by the ants during the searching process. The ACO approach calculates the probability that an ant chooses a particular disassembly step based on the quantity of pheromone corresponding to the positive feedback-loop and a heuristic value. The explosion distance is determined by the axis aligned bounding box of the assembly. The currently exploded part has to be moved to the outside of the bounding box of the remaining parts. Furthermore, a disassembly matrix represents the interference relationship of all parts to each other. It is derived from the interference matrix described above. For this, the three interference matrices for the coordinate directions D_x, D_y, and D_z are combined to a single matrix. An element E_{ij} of the disassembly matrix is imple-

mented by three digits representing the three directions. Consequently, the possible explosion directions of the method of Xing are also limited to these directions.

Another remarkable approach is the work of Li et al. [LACS08] who extend the work of Agrawala et al. [APH+03]. They introduce a method of automated generation of interactive exploded views (Figure 2.5) of 3D models. Their method uses of an acyclic explosion graph (Figure 2.6) that determines how parts of the model are exploded with respect to each other. Every part of the assembly is a node of the explosion graph. The explosion animation is realized by expanding descendants of a part in the explosion graph before the part itself to ensure not violating blocking constrains during the animation. This means that a part can only be exploded when all of its descendants were previously exploded. The system computes a nested collection of explosion graphs for models which are hierarchically organized. The authors use following conventions distilled from example illustrations to ease the mental reconstruction of the exploded diagram for the viewer:

Blocking constraints: Parts can only be explode into unblocked directions.

Visibility: The offsets between parts are chosen so that parts of interest are visible.

Compactness: The method minimizes the distances parts are moved.

Canonical explosion directions: The method uses a restricted number of explosion directions.

Part hierarchy: In technical assemblies parts are often grouped. In order to take this fact into account, the visualization should be generated with respect to the hierarchy of the assembly.

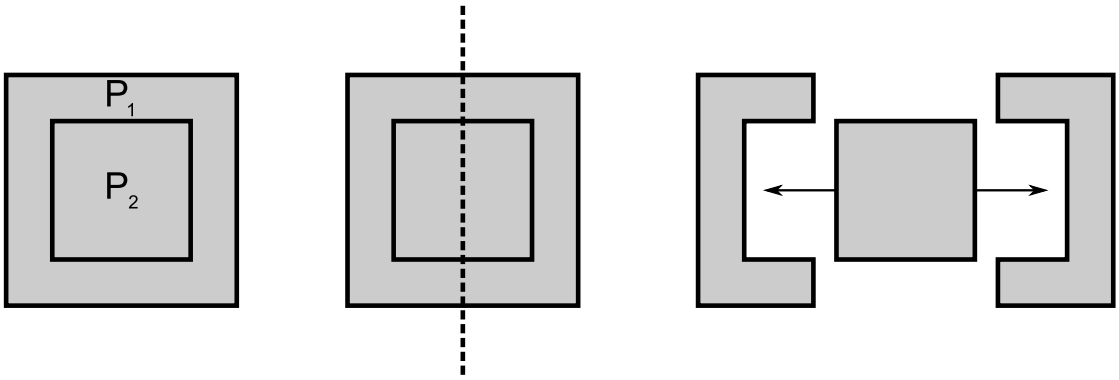


Figure 2.4: Part P_1 encloses part P_2. The method of Li et al. [LACS08] applies a cutting plane to split part P_1. The sub-parts of P_1 are separated to reveal part P_2.

The authors implemented their concepts in a system that also supports splitting containers and cutaways. In many assemblies, parts are completely enclosed by container parts. The system splits them via a cutting plane and explodes the segments apart to reveal the nested part (see

Figure 2.4). Their system also supports direct manipulation of parts by dragging them via the mouse. Furthermore, it offers a riffling function. This means that, when the user hovers the mouse cursor over parts of the assembly, they are exploded away from adjacent portions of the model. The system also provides a high-level interface where the user can search for parts in a list. The user clicks on the name of a part from the list or clicks on a part itself during riffling mode to select it. As a result, the system explodes the necessary parts to reveal the selected one. Furthermore, labelling of parts of the assembly is supported. Long pre-computation is needed (up to half an hour) to calculate the blocking-relations and the explosion graph.

The system of Li et al. [LACS08] is limited to local freedom but could be extended to global freedom, for example, by applying a technique by Canny [Can86]. Canny developed a method for efficient collision detection for moving polyhedrons. In order to do so, he derives three types of constraints between a polyhedral object and polyhedral obstacles and uses an unconventional representation of rotation, a projection of the 3-sphere of unit quaternions onto a hyperplane.

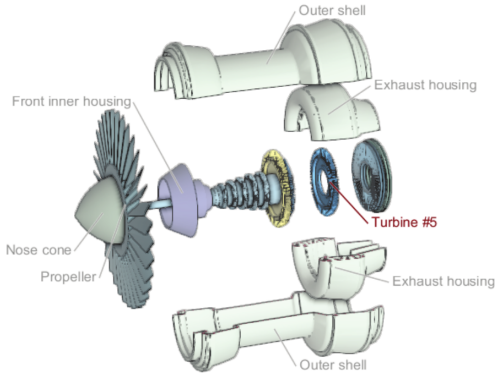


Figure 2.5: Here, one can see the resulting exploded view of a water turbine by the method of Li et al. The image is taken from [LACS08] by courtesy of the author.

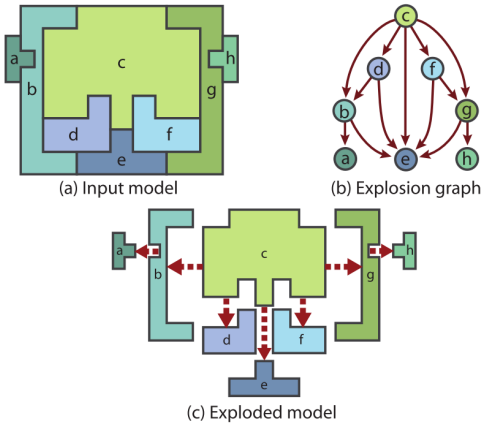


Figure 2.6: The figure demonstrates the concept of the explosion graph of Li et al. The image is taken from [LACS08] by courtesy of the author.

A further system for generating exploded views based on assembly planning has been proposed by Tatzgern et al. [TKS10]. The authors implemented a system that automatically generates compact explosion diagrams inspired by handmade illustrations. In order to generate compact exploded views, the system only explodes a subset of the assembly parts. This eases the inference by the viewer from the exploded view of the assembly to the entire 3D-model (see Figure 2.7). Their concept is to identify recurring, similar groups of parts. Afterwards, their system chooses a representative group with respect to the quality of the exploded view of the group. It only explodes the representative group and leaves the other groups untouched. This approach reduces the complexity of an explosion diagram and increases the readability. The system uses the blocking constraints and disassembling method of Wilson [Wil92]. The method is extended by the idea that smaller parts have to be removed first. This results in sticking these

parts to bigger ones during the explosion. Furthermore, their system removes similar parts or groups in a sequence. The outcome of this approach are more similar exploded views of similar sub-assemblies because of almost identical conditions. In order to realize their approach, the system has to find similarity between groups of parts. It determines such sets of similar sub-assemblies by performing a frequent sub-graph (FSG) search on the graph that represents the assembly. The selection of a representative group for exploding is determined by a quality measurement. The measurement takes into account the footprint (size in screen space) of the exploded group candidate, the footprint of all other similar groups and the visibility of the parts of the exploded group candidate. A weight sum score is calculated for each group candidate and the one with the best score is taken as a representative and, therefore, is actually exploded.

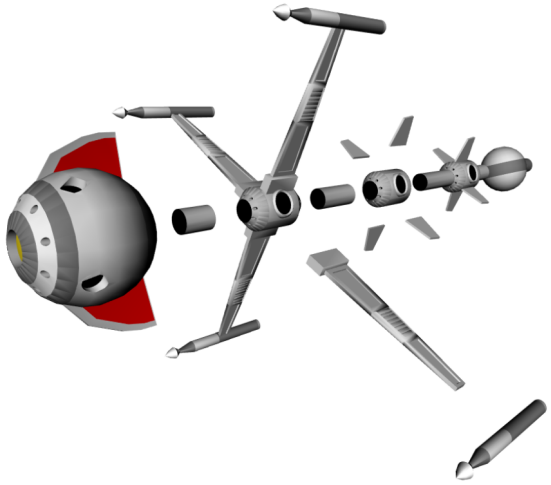


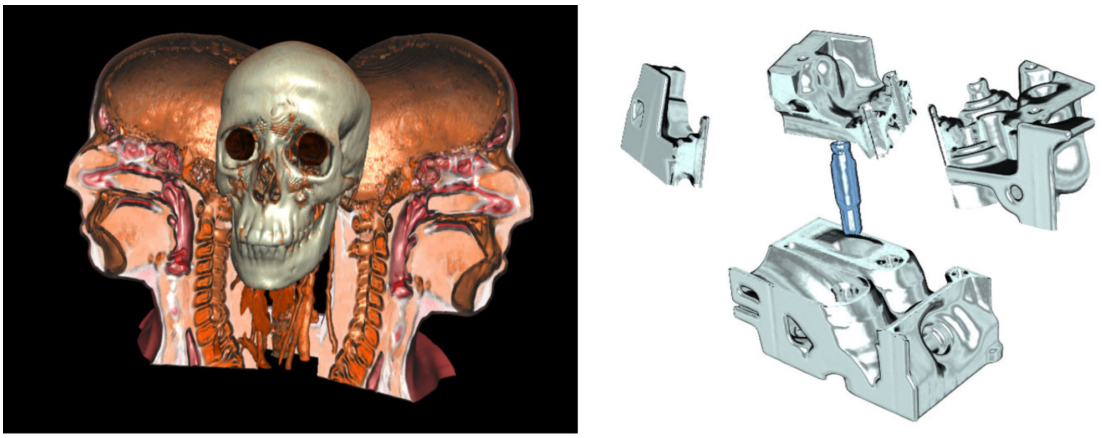
Figure 2.7: The figure displays an exploded view generated by the method of Tatzgern et al. Reoccurring structures are not exploded to reduce complexity. The image is taken from [TKS10] by courtesy of the author.

A completely different approach to generate exploded views is proposed by Bruckner et al. [BG06]. Their method is not based on assembly planning, it is rather based on physical forces. They implemented an interactive system that generates exploded views for volume data (e.g., scanned medical data 2.8). The system does so by using physical forces to move parts away from each other. The used forces are explosion forces, viewing forces and spacing forces. Force-directed layout techniques for graphs provide various aesthetic standards, like space filling or symmetry. The work of the authors focuses on segmenting the volume data and applying the forces to the segments. For this, they distinguish between selection and background of the volume data. The selection defines a degree-of-interest function that provides each data sample with a value ranging from zero to one, where one means most and zero least interesting. Every sample that is not selected composes the background. The background is sectioned and the resulting parts are exploded to reveal the selection. Their system provides user-control over the selection, the degree of explosion, an interface for defining cuts to split the background, and real-time calculation and animation of the exploded view. Furthermore, it empowers the user

to define the importance of background parts by modifying their mass. This affects the layout because of the force-based approach. Additionally, the user can define various joints between parts. This results in a restriction of freedom of movement of those parts during the explosion. For example, a hinge joint can result in folding the background apart when the forces are applied to it. The authors consider the occlusion of parts of interest to be a general and important problem of scientific visualisation. They state that a solution could be to reduce the opacity or to remove occluding parts.

> However, the drawback of these approaches is that parts of the context information are still removed or suppressed [BG06].

Exploded views can handle occlusions without removing context information.



a) b)

Figure 2.8: The figure shows the volume based technique of Bruckner et al. While it works quite well for medical scanned data (a), it is quite hard to mentally reconstruct the single parts of a technical assembly (b). The images are taken from [BG06] by courtesy of the author.

Conventional static exploded views suffer from ambiguous spatial relationships (it might be unclear, how parts fit together) and visual clutter as stated by Li et al. [LAS04]. Static illustrations are designed to include as much information as useful to be presented at the same time. In order to overcome these two drawbacks, the ambiguous spatial relationships and the visual clutter, interactive diagrams can be used. Interactive diagrams enable the user to interact with the model and de-emphasize unfocused information. Li et al. [LAS04] present a system that takes a static 2D image of an exploded view of a complex mechanical assembly as an input and supports the user with semi-automatic authoring tools to generate an interactive exploded diagram. The system empowers the user to segment the individual parts via image processing and enables her/him to form stacks that define how parts move relative to each other while exploding. It also includes occlusion handling and labelling of parts. The resulting interactive diagram supports a smooth and entire expanding and collapsing animation, direct manipulation

of the exploded view by dragging parts via the mouse, and searching and selecting the part names in a list. Selecting a part results in expanding it in the exploded diagram.

2.2 Architectural Theory

In order to understand the requirements of architectural visualization, one has to take into account architectural theories. Most theoretical concepts of architecture consider space as an important part of a building. However, some of these approaches use the conception of space as the main and base element of their theories. I picked two of these theories that are mentionable in this context: the "theory of space" on the one hand, and "space syntax" on the other.

2.2.1 Theory of Space

Sommer [Som10] discusses theoretical architectural viewpoints under the aspect of the theory of space ("Raumtheorie"). For this, the various concepts of space of different considerable architects and architecture theorists are analysed. Henri Lefebvres [Lef91] considers space as a social product, thus, the reproduction of the structure of the society. He differentiates between a material production, a production of knowledge and a production of meaning of a space. These three dimensions result in his three concepts of space: the perceived space, the conceived space, and the lived space. Peter Beherens [Beh09], on the other hand, defines space as a result of its relation to its sensible boundaries. He postulates that the built architecture has to enclose this space. This way, the hull of a space gets an essential significance because it produces the space. He considers the hull, which is shaped by the space boundaries, as a main design concept of architecture. His approach results in structuring the space into compartments of the interior of a building. Walter Gropius [Gro26] postulates the possession of space with respect to a democratic basis. In order to satisfy the requirements of an individual, he suggests the concept of big building blocks that can be exchanged and organized in custom arrangements.

2.2.2 Space Syntax

Another theoretical approach to analyse spatial configurations, like buildings or cities, is the space syntax. The main entity of the theory is the space and its logical, geometrical, and social aspects. Space can be broken down into components and be represented as graphs that describe the relative connectivity and integration of those spaces (see Figure 2.9). Bill Hillier [Hil07] is one of the representatives of theoretical concepts of the space syntax. In his book "Space is the Machine", he states:

> At the most elementary level, a building is a construction of physical elements or materials into a more or less stable form, as a result of which a space is created which is distinct from the ambient space [Hil07].

Such a space is enclosed by physical boundaries, which separate the space from the environment. Thus, the boundaries provide a hull for the space, which protects it, and creates a physical and logical distinction between inside and outside.

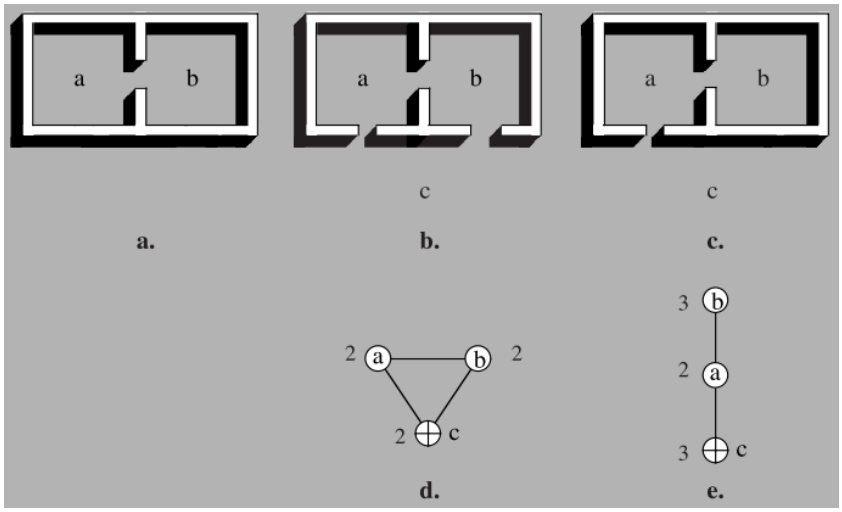


Figure 2.9: The figure shows different configurations of spaces and the analysis of their spatial relations using concepts of the architectural theory of space syntax. The image is taken from [Hil07] by courtesy of the author.

2.3 Architectural Model

A concrete conceptual model of a building that reflects the previously mentioned theoretical concepts is needed. Otherwise, it is not possible to define formalized methods and algorithms that handle operations on a building. I make use of the fact that any computer-integrated construction needs a data model and, therefore, various standardisations of the schema of the building model exist. The schema structures the building and its components, not the format of the document representing the building data. The implementing data structure contains entities, relationships, attributes, and so on. A coherent set of these contained elements forms a data model [Bjö92]. There are some informative papers dealing with conceptual building models [BGL+05, Bjö92].

Björk [Bjö92] conducted a research and an analysis of a collection of different building product models. He states that such a research is essential for future CAD applications. For this, relevant parts of conceptual models of the chosen theoretical works and prototype projects were redefined in a compatible format and analysed. The author compared four picked models with respect to similarities and differences. His research indicates that

> ...the conceptual modelling of spaces, the surfaces bounding them and the structures enclosing them is at the kernel of most of the perceivable aspect product model... [Bjö92]

The four considered models contain information about topological relations (“bounds”, “fills”, “consists of”, etc.) between building components and building spaces. This information is directly modelled or can be indirectly deduced from the positioning and the geometric shape of the building components. Many CAD systems include explicit information of topological

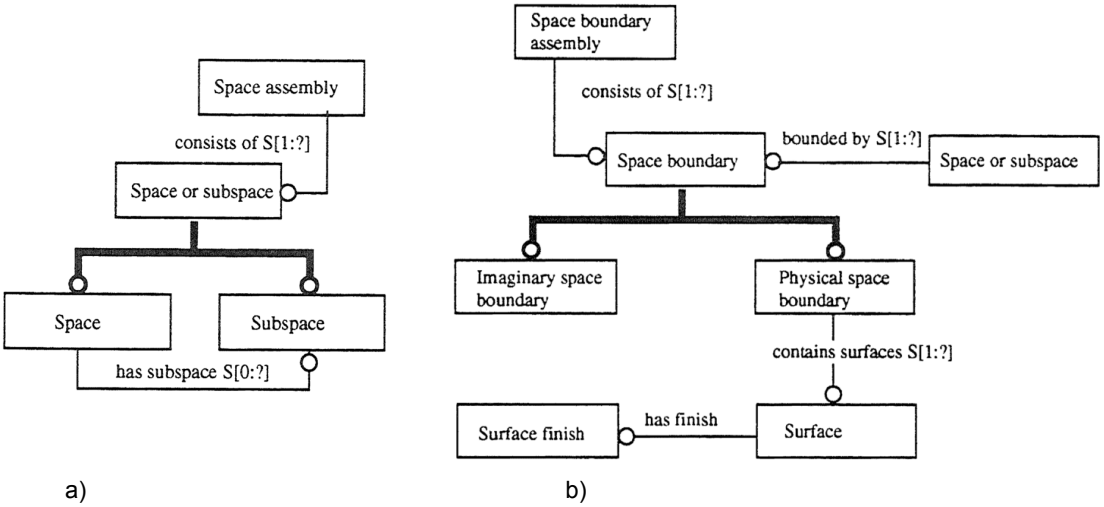


Figure 2.10: The diagrams represent two parts of the schema of the model of Björk et al. The images are taken from [Bjö92] by courtesy of the author.

relationships, therefore, Björk also concentrates on models doing so. He uses the EXPRESS modelling language to model the schemas of the different building models (Figure 2.10). The analysed projects are: RATAS, GSD, House Model, and COMBINE IDM.

RATAS: The RATAS model includes an abstraction hierarchy into the building model that consist of the concepts building, system, subsystem, part, and detail. It contains two main relationships: "part-of" and "connected-to". The model was developed as a guideline in the year 1987.

Groupe Structuration de Donnees (GSD): The GSD group analysed different conceptual models of buildings developed during the years 1985-1990. The group advocates the use of STEP (ISO standard) for geometric data structures.

De Waard's House Model: This data model of residential buildings was developed for the purpose of studying methods for computer-aided conformance of buildings pertaining to building regulations. De Waard defines different types of spatial entities: building block space, building floor space, house space, house floor space, and private elementary space. They are decomposable into each other. The model is derived from a typical organisation of a multi-storey apartment house. It also defines a spatial hierarchy, which is closely related to the space boundary hierarchy. The boundaries serve as separation structures that are categorized into inner and outer separation structure and vertical, horizontal, and sliding separation structures.

Integrated Data Model (IDM) of the Combine Project: The Combine Project is a multi-national project that intends to prove that it is feasible to use the product model approach for the integration of energy-conscious building design tools into various CAD tools.

Björk deduces the concepts that the four models have in common: space, space boundaries, synthesis, and enclosing entities. The main aspect in this context is the approach that a building consists of a network of spaces separated by building elements. This way the space is ranked as the central entity of these concepts. There are two ways of defining space. On the one hand, space is defined by the complete physical separation via physical objects that provide visual, acoustic, and inner climate shelter. Each space is enclosed in a "shell" of boundaries (walls, ceilings, floors, openings filled with doors and windows). This is stated to be the viewpoint of the building user onto the building. On the other hand, the space is defined by its function, i.e., how the space is used.

Another promising semantic 3D building model is presented by Benner et al. [BGL+05]. They criticise that available 3D city models lack of semantic information and suggests to add this information (e.g., floors, passages, opening objects, rooms, walls, outer walls, etc.) to the geometry of the 3D building models, the city model consists of. The authors exploit the Industry Foundation Classes (IFC) standard to derive their own semantic building model. The authors extract the semantic information from the IFC definition of a building and transform it to the own building model by simply mapping some information and deducing additional information like the geometry of the outer shell of the building.

2.4 Visualization

Visualization of architecture is an essential step of the planning phase of a building. It can reveal mistakes of the planning phase like errors of the statical construction, but also transport an idea of the unfinished building to the future occupant. This is crucial because a building is a very expensive acquirement. For this reason, there exists an amount of standardised illustration techniques for buildings like floor plans, cuts, side views, detail views, and so on. But there are two drawbacks of these illustrations: An untrained viewer may not understand conventional symbols that are used. Furthermore, the illustrations are two-dimensional, thus, the viewer has to mentally complement the third dimension. 3D-illustrations of the complete building, like 3D-renderings or an interactive "ArcBall" visualization, lack of conveying the internal structure of the building and the contained spaces. A virtual "Walkthrough" of the building, on the other hand, can reveal its internal structure. However, it cannot communicate an overall picture of the building [HN04].

For this reason Niederauer et al. [NHAH03] developed an interactive system that generates exploded views of any OpenGL based application rendering architectural structures. For this, the system intercepts the graphic pipelines of these applications and explodes the rendered geometry. The system applies a statistical analysis of the geometry to find floors and slabs due to the assumption that the floors are horizontally oriented and positioned with at least a minimum distance. Afterwards, it explodes floors vertically with a user-definable offset. The rendering is done by clipping floors of the multi-storey building and rendering them separately in a multi-pass rendering step (see Figure 2.11 for an example result). However, the authors stated that it would be easier with more information about the meta model of the building because a conventional system to generate exploded views requires extensive knowledge about the part/sub-part decomposition.

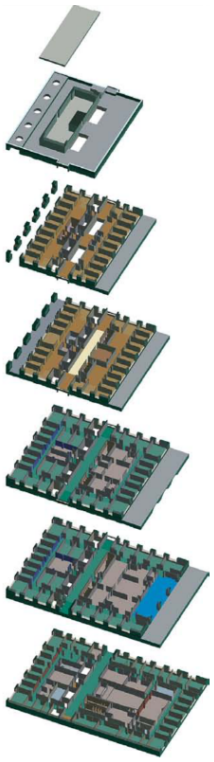


Figure 2.11: The figure displays an example of the result of the work of Houston et al. It is recognizable that the exploded view lacks in compactness. The image is taken from [HN04] by courtesy of the author.

2.4.1 General Approaches of Illustrations

General approaches of illustrations are helpful to get a better understanding for the development of visualizations. Seligmann et al. [SF91] claim that an illustration is a picture that is designed to fulfil a communicative intent. Communication involves both, intention and interpretation. Illustrators have to ensure that those do not differ. The paper of the authors presents the first steps in developing an automated intent-based illustration system (IBIS) that generates such

illustrations and also evaluates its stylistic choices. Communicative goals are accomplished by the proper selection of design rules. The system input are the communicative goals (i.e., location, relative location, property, state). The output is the illustration fulfilling these goals by applying predefined design methods. It can occur that goals cannot be fulfilled because of contradicting other goals. Design evaluators are used to determine how well communication goals have been accomplished. All these concepts are formalized in the intent-specification language of the IBIS. An example of a design rule, holding the goal of showing the location of an object in a context, is as follows:

> The object must be included in the illustration. The achievement threshold "highest" indicates that this style strategy must be fully satisfied. The object must be recognizable. The context object must be included. The object must be visible. The object must be highlighted. The context object must also be recognizable, but with a lower threshold. The context object must also be visible, but with a lower threshold [SF91].

Such design rules are the implementation of design principles, a concept introduced by Agrawala et al. [ALB11]. The authors argue that, due to the mass of information produced nowadays, there is a need of automatically generated visual communication illustrations. They consider design principles not only to be strict rules (design rules) but also rules of thumb (evaluation criteria), that might even contradict one another. Design principles are rather guidelines that help to improve the viewer's comprehension by explaining how to use visual techniques to emphasize important information and de-emphasize irrelevant details. An example of an evaluation criteria are the contradicting goals of separation and compactness of the parts in exploded views. Separation is used to make the parts visible, and compactness is used to make best use of the screen space. The authors suggest a three-stage approach for creating visualization design systems:

Identify design principles: At this stage, they analyse hand-designed domain-specific visualizations and search the literature for prior work in perception and cognition, or conduct own studies on it to deduce design principles.

Instantiate design principles: They use procedural techniques to build a visualization design system out of a set of design rules and evaluation criteria.

Evaluate design principles: Then, they evaluate the system by performing qualitative interviews as user feedback. Furthermore, they analyse quantitative usage statistics of user studies.

The authors state that the advantage of this approach is that skilled designers do not explicitly formulate the applied design principles. Accordingly, there often just exist example illustrations rather than concrete design principles.

2.4.2 Towards Architectural Visualization

Hagedorn and Döllner [HD07] identify two tasks of the visualisation of building information: On the one hand, the visualization of internals of composite structures, on the other hand, the visualization of intangible information like the usage of a room or the grouping of rooms to floors. They present two visualisation techniques: The first one highlights relevant information and unimportant information is reduced in perceptibility or even skipped. The second technique deforms the given building structure. For this, they identify the floors of a building and transform them. Two transformation methods are realized by the authors: tilting the floors up, and exploding the floors vertically. Tilting the floors up provides less information but also needs less screen space (see Figure 2.12). The authors implemented a web-based visualization system, that uses and combines web services of BIM (Building Information Model) and GIS (Geographic Information System) data. It provides configurable 3D-views and rendering styles and an automated calculation of the camera position. The authors describe an approach to visualize and analyse BIM models within 3D virtual city models (GIS data) to embed the information about structure, usage, and other properties of a building into its spatial context. The Industry Foundation Classes (IFC) standard is an implementation of the BIM standard. It provides detailed information about buildings (e.g., about building parts, floors, rooms, openings like windows and doors), but also includes the plumbing fire protection domain. For example, one can use this information to develop and analyse fire rescue scenarios. Furthermore, IFC is extendible with definable properties for elements.

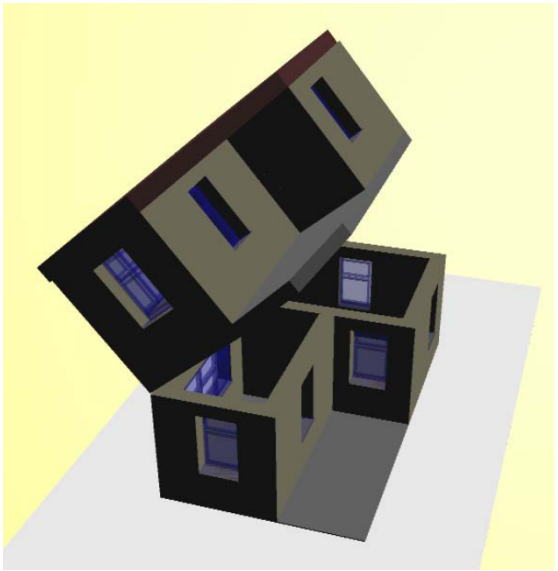


Figure 2.12: The figure demonstrates that tilting up a floor provides less information but also uses less screen space then conventional exploded views, which use translations. The image is taken from [HD07] by courtesy of the author.

Jianping et al. [JFD10] also use the IFC standard as a data model basis. The objective of their

research is to develop a lightweight and low-cost virtual construction (VC) platform integrating IFC and OpenGL.

> IFC has been known as a common product and process model to provide interoperability among various IT systems for AEC/FM [architecture, engineering, construction, and facility management] industry [JFD10].

The IFC standard represents physical components, like walls or doors, by the class `IfcBuildingElement` and its subclasses. Detailed information about the components is linked via `IfcPropertyDefinition`, but also shape and material information can be appended. The paper of the authors focuses on mapping and converting the geometrical and material information (e.g., colour, diffuse reflectance, and texture) from the IFC standard to the own VC platform to generate realistic renderings of 3D-models. Therefore, they analyse the IFC standard regarding its color model (`SurfaceColour`, `DiffuseColour`, `SpecularColour`, and `SpecularHighlight`, implemented in `IfcSurfaceStyleRendering`) and the applied texture mapping.

Furthermore, the IFC standard provides topological information about a building. This information can be used to implement indoor route visualization, as demonstrated by Hagedorn et al. [HTGD09]. The authors discuss the classification of indoor objects and structures with respect to their semantics. Based on the classified semantics, they deduce a level-of-detail (LOD) model for the visualization of a route in a building. For this, the authors define four levels of detail, which differ in their thematic, geometric, topological, and visual complexity. The level-of-detail (LOD) approach is used to cope with the massive amount of data of a building.

CHAPTER 3

Context-Sensitive Exploded Views for Architecture

In this chapter, I will present the applied theoretical foundation of my method to generate context-sensitive exploded views for architecture. In order to do so, I will refer to approaches of related work in this domain and develop own concepts.

In the previous chapter, I have cited techniques to generate exploded views. Most of the approaches are based on assembly planning and, therefore, do not take into account rooms or spaces. However, the concept of space is an important part in architectural theory. Thus, I looked for hand-made sample illustrations of exploded views of buildings. Unfortunately, most of the examples either just separated the floors from each other or exploded too many building elements at the same time (see Figure 1.2). The latter may be a proper method to generate static illustrations that have to include as much information as possible, but in an interactive visualization system, one should exploit the possibility to reduce visual clutter and clarify membership and spatial relations [LAS04]. For this reason, I have to identify the inappropriateness of previous approaches of generating exploded views when applying them in the domain of architecture and define own design principles. For this, I modify the design principles of Li et al. [LACS08] to achieve a correlation to the mentioned architectural theories.

First of all, I will give some definitions of used terms. Afterwards, I will explain goals and deduce design principles. I will extract concrete local design rules and transform them to global design rules. At the end of this chapter, I will describe the designed algorithm that implements these concepts.

3.1 Definitions

The presented concepts of this chapter may be complicated and, therefore, it is worthwhile to use clear definitions of reoccurring terms and ideas. In the following I will give definitions to

clarify such terms to increase the readability of the work. Furthermore, I will introduce symbols to facilitate the usage of these terms in a formal context. The definitions are related to parts of a building and meta information of these parts.

Space: A space is no real physical object. It rather marks a volume usually containing other elements.

Hull: The hull of a space encloses the space thus shaping the geometrical representation of the space. This is concretised by the following: Rooms are shaped by walls and slabs, floors are shaped by façade parts and slabs, and the building is shaped by the façade and the roof of the building.

Front-face: The front-face of a hull are all surfaces which are visible from a given camera view point.

Back-face: The back-face are all surfaces of a hull which are hidden because their surface normal vector is pointing backwards relating to a given camera view point.

Building element: Building elements are all parts P a building consists of. These are spatial structures S, boundaries B, and remaining building elements R.

Spatial Structures: A spatial structure S is a building element that is no real physical object. It is rather the space between boundaries. The boundaries B_S shape the geometric representation of the spatial structure S. While spatial structures implement the space-part, the boundaries represent the hull-part of the concepts of space and hull. Spatial structures can contain other spatial structures, boundaries of those and remaining building elements. The surrounding site of a building, the building itself, building sections, floors, and rooms are ranked among spatial structures.

Boundaries: A boundary B is located between spatial structures to separate them from other spatial structures or from the environment. Walls, slabs and roofs, and also doors and windows are part of the boundary set. But also spatial structures itself can be boundaries of other spatial structures if there is no physical boundary between them.

Remaining building elements: The set of remaining building elements contains all elements R of the building that are neither a boundary nor a spatial structure. One can find furniture, stairs, columns, handrails and many more among them.

Bounding relation: If a spatial structure S is bound by a boundary B, then there exists a relation between S and B. This relation is called bounding relation $Br(S, B)$.

Containment relation: A spatial structure S can contain any building element P. Containment means that the volume of the spatial structure S completely encloses the element P. The symbol $Cr(S, P)$ refers to this containment relation.

Context of building elements: The context of a given building element P are all other elements P_i that have a relationship to P. This relationship may be a bounding relation $Br(P, P_i)$ or a containment relation $Cr(P, P_i)$.

Context-sensitive explosion direction: This term describes the approach of using different explosion directions depending on the considered context. For this reason, a single building element P may have multiple possible explosion directions, which are dynamically chosen related to its relationships to other building elements P_i.

3.2 Design Decisions

In this section, I will informally describe concepts of design decisions. I will also refer to existing approaches of the considered literature. Furthermore, I will enumerate the created design principles and explain them.

3.2.1 Previous Approaches

In order to get an intuition of the concept, I implemented parts of the algorithm of Li and Agrawala et al. [LACS08, APH+03] as a starting point of the implementation work. Unfortunately, the results were not very satisfying in the domain of architecture. Especially the explosion direction, but also the explosion order causes problems.

The approach of Li et al. [LACS08] only applies local freedom of parts for the determination of the explosion direction of a part. This works well for compactly built technical assemblies, but causes problems for architectural buildings, where one has much space between single parts. For this reason, I modelled rooms as standard assembly parts to overcome the drawback of the use of local freedom. However, this approach handles rooms as physical objects and, therefore, neglects the fact that they are invisible and just are shaped by their bounding walls and slabs. By moving the parts away from each other, the shape of the room is destroyed. Some kind of relation is needed to bind walls and slabs to rooms and, therefore, shape the form of a room. The approach of Li et al. includes grouping of parts, which suggest itself to solve this. However, the problem of their concept of grouping is that a part cannot be a member of two different groups. But exactly this would be necessary to model the relationships of a wall that bounds two different rooms.

Furthermore, the explosion direction is pre-calculated in their method and there exists just a single explosion direction for a particular part. This explosion direction is only determined by the surrounding parts in contact. Accordingly, when applying the method of Li et al. to a building, different wall intersections may result in different explosion directions for these walls. Another drawback of their method is that the order of removing parts influences the explosion direction. For these reasons, it is not possible to implement different context-sensitive explosion directions for a single part depending on multiple different considered building elements with the given method of Li et al. [LACS08] and Agrawala et al. [APH+03]. A more dynamic approach is needed.

Most of the considered approaches exploit assembly planning as a base concept (see Section 2.1.1 and 2.1.2). Therefore, they focus on order and blocking constraints derived from the disassembling sequence and suffer from the same drawbacks mentioned above. However, I want to

concentrate on spatial structures, on bounding and containment relations. Thus, I need a modified approach because there is no plausible disassembling sequence for the parts of a building. For example, furniture elements would have to be carried the whole route to the entrance door to be disassembled. However, this is not an expedient procedure for an exploded view.

3.2.2 Goals of Visualization

In order to use an adequate foundation for the resulting visualisation system of this work, I want to apply the concepts of the space theory [Som10] and the space syntax [Hil07]. Therefore, I use the approach of Peter Beherens [Beh09], who segments an architectural building into space and hull, where the hull determines the shape of the space. Both, the hull and the space, are important architectural elements.

Moreover, the space is not visible without the hull. Hence, it is essential to preserve the spatial relation between space and boundaries during the explosion process. On the other hand, the boundaries of a space occlude the interior of the space. For this reason, the occluding boundaries need to be exploded. In order to balance these two contradicting principles, I propose the following approach: Only the front-face boundaries should be exploded to reveal the interior space, whereas the back-face boundaries should remain untouched to form the shape of the space.

The explosion direction of the front-face boundaries shall be intuitive. This means that the explosion direction shall be predictable and traceable by the viewer. Furthermore, the boundary shall be well visible and the spatial relationship to the space shall be obvious. An explosion direction which simulates a real explosion fulfils this. Therefore, the explosion direction of a boundary shall point away from the covered space. One can recognize that there have to exist multiple explosion directions for a boundary that bounds multiple spaces, depending on which space is focused. Consequently, context-sensitive explosion directions are needed (see Figure 3.1).

Many concepts of building models include a hierarchical structuring of the contained space [Bjö92, BGL+05, HD07] (building, floors, rooms, etc.). This approach is very suitable for my purpose because it correlates with the levels of detail approach (see Section 1.2). Therefore, I want to integrate the hierarchical structuring into the visualization and navigation system. The idea of the semantic concept of the hierarchy is that every room is part of a floor, every floor is part of a building section, and every building section is part of the building itself. A tree graph is used to represent these relationships. The generation of the exploded view shall take into account the semantic hierarchy of the building to implicitly transport the information of the structure. Therefore, the order of exploding parts shall be determined by the hierarchy rather than by blocking constraints.

3.2.3 Design Principles

Some of the considered papers introduce design principles to subsume concrete design rules (see Section 2.1.2). I will exploit some of them, adapt others and extend them. The design principle "visibility" is used by all approaches. Also the concept of "compactness" of the exploding parts

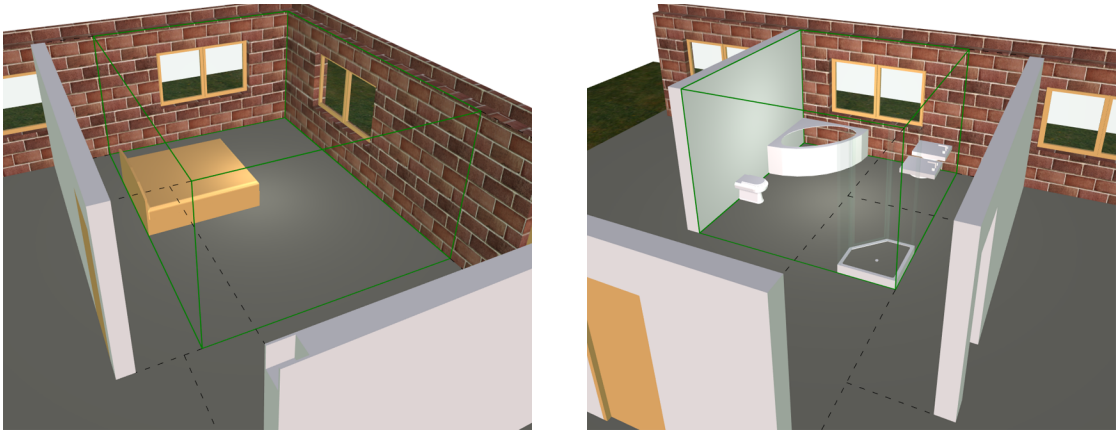


Figure 3.1: The two figures demonstrate the effect of a context-sensitive explosion direction. The left wall in the left figure and the right wall in the right figure are the same. The wall moves into opposite directions while exploding relating to the focused room (bedroom or bathroom) and the view direction of the camera (note: the camera pans to the left).

is mostly applied. I modify the commonly used "blocking constraints" to satisfy the requirements of the above-described goal of visualization. Furthermore, I introduce two new design principles: "context preserving" and "traceability".

Visibility: The focused space shall be revealed by moving occluding parts out of the line of sight. The hull of the space shall also be visible if possible.

Separation: The parts shall be separated from the focused one. The focused part shall be recognizable and distinguishable from the other parts of the building.

Compactness: The resulting exploded view shall be compact to utilize the given screen-space. Therefore, parts shall stick to their containing exploded spatial structures while exploding. Furthermore, the parts shall be moved the minimum distance to completely reveal the focused part plus a separation offset.

Blocking Minimization: This constraint is weakened compared to the blocking constraint of Li et al. [LACS08]. Their method only allows unblocked explosion directions, i.e., directions into which a part can be removed without interfering with other parts of the assembly. Blocking minimization means that a proper explosion direction is chosen to fulfil the other design principles and to minimize the intersection of parts, but intersection is allowed.

Context Preserving: The explosion direction and distance of the hull shall empower the viewer to deduce the shape of the covered space and the spatial relationship to it. This implies the concept of context-sensitive explosion directions. Furthermore, other exploded parts shall be placed at a proper position to ease the mental reconstruction of the exploded view and to preserve the part hierarchy.

Traceability: Not every single part shall be moved separately because the animation would take too much time. This is inappropriate for an interactive system. Therefore, every part is moved at the same time and is bound to the other parts that move into the same direction. Furthermore, the use of a smooth animation of the explosion is recommendable. This is done by accelerating the animation at the beginning and slowing it down at the end of the animation.

3.3 Hierarchical Model

Referring to my approach, the main difference between technical assemblies and buildings is that a building does not only contain much more empty space, but also that this space is a very important aspect of architecture. If one considers a room, then most of the space is not filled with furniture or walls. Otherwise, a person could not use the room at all because this person can only move in the space between objects. However, a room is shaped by its walls and its function is determined by its furniture. So meaningful exploded views of architecture have to take into account the type of a building element and its relations. Therefore, I classify the elements to support an appropriate visualization. These classes are "spatial structures", "boundaries", and "remaining building elements" like furniture or stairs. One can find a detailed definition in the previous Section 3.1.

3.3.1 Hierarchy

In order to come up with an adequate building model, I propose a model in which all building elements are organized hierarchically within a tree graph. A building element P is represented by a node of the tree. In the following, I will not distinguish between building parts and tree nodes. The level L_P of a node P is the maximal path length of the tree reduced by the distance from P to the root node (see Figure 3.2). An ancestor of a node P_i is any node P_j that is located at the path from node P_i to the root node. Then node P_j has a higher level than node P_i. Furthermore, node P_i is a descendant of node P_j. If node P_i additionally is adjacent to node P_j, then node P_i is a child of node P_j and P_j is the parent of the node P_i. The following constraints are added to the model to facilitate the development of algorithms:

- The building site is at the topmost level of the hierarchy. Thus it is the root node. The building site contains every other building element.
- The building and the surrounding environment of the building are children of the site.
- Every internal node and the root node are spatial structures.
- A leaf node is either an empty spatial structure, a boundary, or a remaining building element.
- If a building element P_i is a descendant of a spatial structure S_j, then the boundaries B_{S_j} of the spatial structure S_j completely enclose the building element P_i.

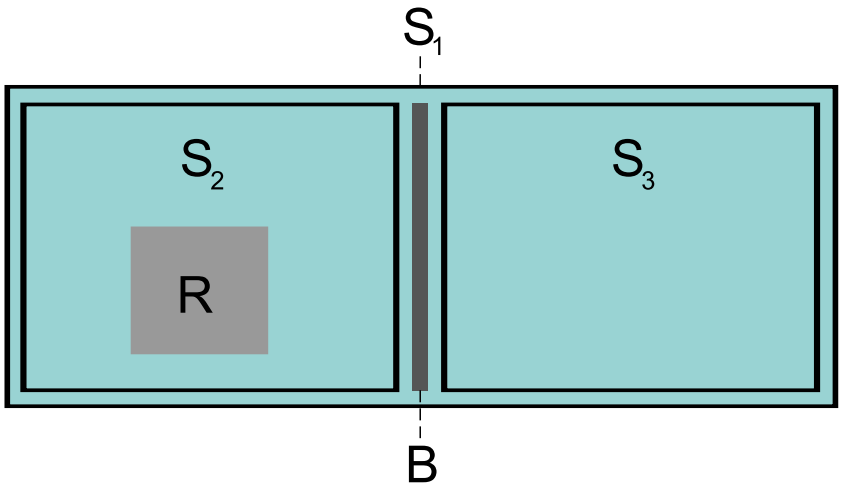


Figure 3.2: The spatial structure S_1 contains all building elements. The boundary B is a wall that separates spatial structure S_2 and S_3. The remaining building element R is contained in spatial structure S_2.

- Assume S_i, S_j, S_k and S_l are spatial structures. If S_i is an ancestor of S_j and S_k is an ancestor of S_l and S_j and S_l share a boundary $B_{S_jS_l}$, then S_i and S_k also share a boundary $B_{S_iS_k}$ containing the former boundary $B_{S_jS_l}$. Figure 3.4 demonstrates this associative relation.
- The hierarchy level L_B of a boundary B equals the hierarchy level L_{S_h} of the spatial structure S_h with the highest level of all spatial structure which have a bounding relation $Br(B, S_i)$.

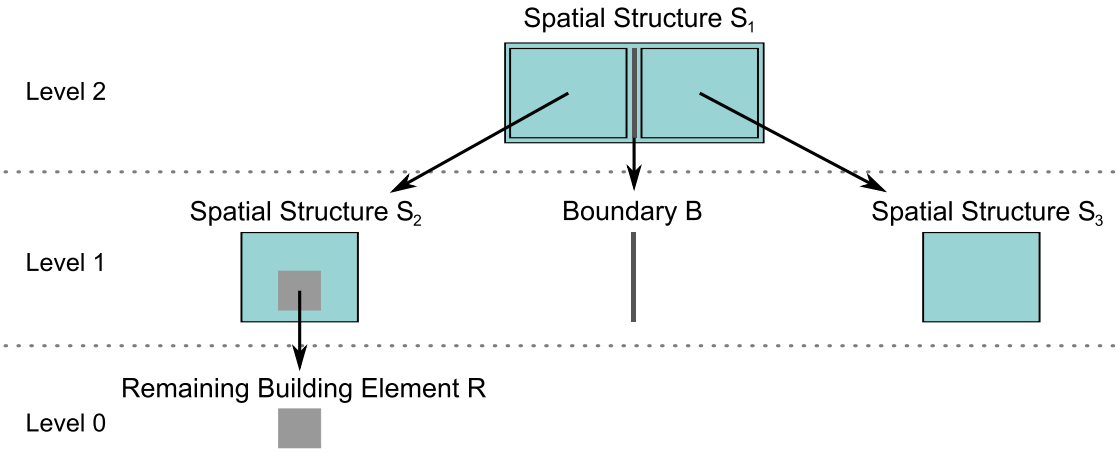


Figure 3.3: The figure demonstrates the structure of the hierarchical model for the constellation of building elements of Figure 3.2.

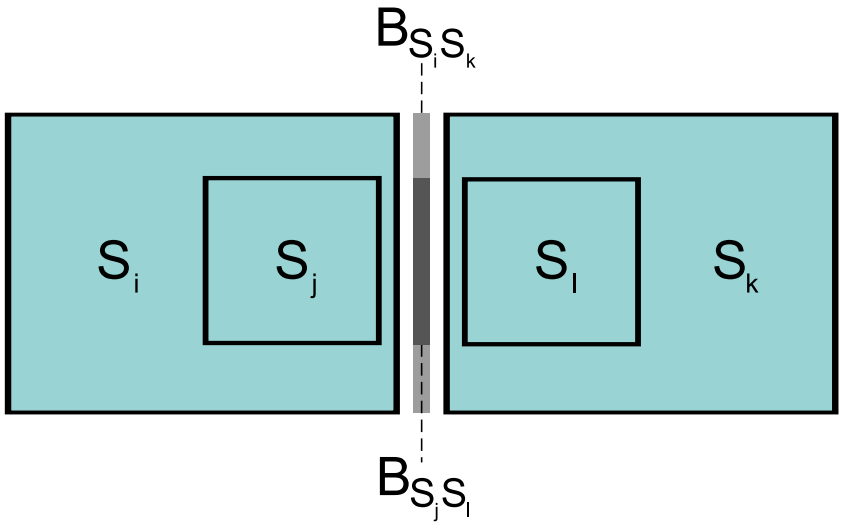


Figure 3.4: This figure demonstrates the associative relation of spatial structures and boundaries.

Relations. There are additional relations between the nodes extending the hierarchy relations. First of all, there is the parent and child relation mentioned above. This represents the containment relation Cr. Every node has a parent node except the root node. Every internal node, i.e., every spatial structure node, can have multiple children. In addition, spatial structure nodes have a relation to their covering boundary nodes and and vice versa. These are the bounding relations Br. Furthermore, a relation between a boundary and its void fillings, like doors and windows, is added. This enables the system to treat a boundary and its void fillings as a single part, but also empowers it to alternatively divide them into multiple parts. These additions to the hierarchical model extend the tree graph by including further edge connections between nodes.

Building Element Node. A building element node P can be a spatial structure S, a boundary B or a remaining building element R. It encapsulates its relations to other nodes. Furthermore, it contains the geometric representation of the related building element. Additionally, it holds the explosion information of the element like the explosion direction and the explosion offset.

3.4 Design Principles to Design Rules

Design principles are the the sets of design rules which lead to a valuable visualization. The implementation of these rules results in a system that automatically generates such visualizations [ALB11]. The main concern of my drafted design rules is to expose a selected spatial structure and reveal its interior by using the concepts of exploded views. This way one can see the internal parts and their relations because occluding parts are moved away. On the other hand, the context of the spatial structure shall be preserved (e.g., the façade of a floor) so that the viewer can mentally reconstruct the moved parts. I start by defining local design rules and transform them to global ones afterwards.

3.4.1 Definitions

In the following I use the phrase “to explode an element”. This means to move a building element P along its explosion direction D_P. The distance to move is determined by its explosion offset O_P. An element P has to be exploded to reveal the part of interest P_{sel} (i.e., the “selected element”, also referred to as the “focused element”) occluded by the element P. If the element P is a spatial structure and if it explodes, all its descendants explode into the same direction with the same offset as P. The current arrangement of all parts, the exploded and unexploded ones, is referred by the term “explosion status”. Now I will introduce the concept of the “outline frustum”.

Outline Frustum. The outline frustum $Of(P_{sel})$ is used to calculate the explosion offset O_P of any building element P relative to a focused building element P_{sel} (see Figure 3.6). The concept is used to find the minimal distance an occluding element P has to be moved into a given direction to completely reveal the occluded building element P_{sel} for a given camera viewpoint. The outline frustum is constructed for a considered building element P_{sel} and the current camera position. It consists of multiple triangles. A triangle is represented by the view point of the camera and an edge of the building element P_{sel}. However, only outline edges of P_{sel} according to the current camera position are taken to construct the outline frustum (Figure 3.5). If any building element P occludes the considered building element P_{sel}, at least a part of P is located inside the outline frustum $Of(P_{sel})$. Therefore, P has to be moved as far into its explosion direction D_P to escape $Of(P_{sel})$ as a whole.

Furthermore, if the focused building element is a spatial structure S_{sel}, the method not only applies the outline frustum of S_{sel}, but also combines it with the outline frustums of the back-face boundaries of S_{sel} to ensure a better part separation. This way, the exploded parts are not only separated from the selected spatial structure S_{sel}, but also from its shaping back-face boundaries.

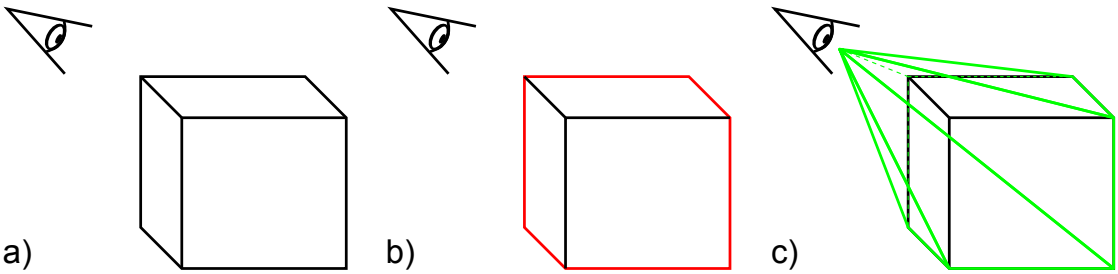


Figure 3.5: a) shows the considered object and the camera. The outline edges of the object relative to the camera have to be found as shown in b). c) demonstrates the resulting outline frustum.

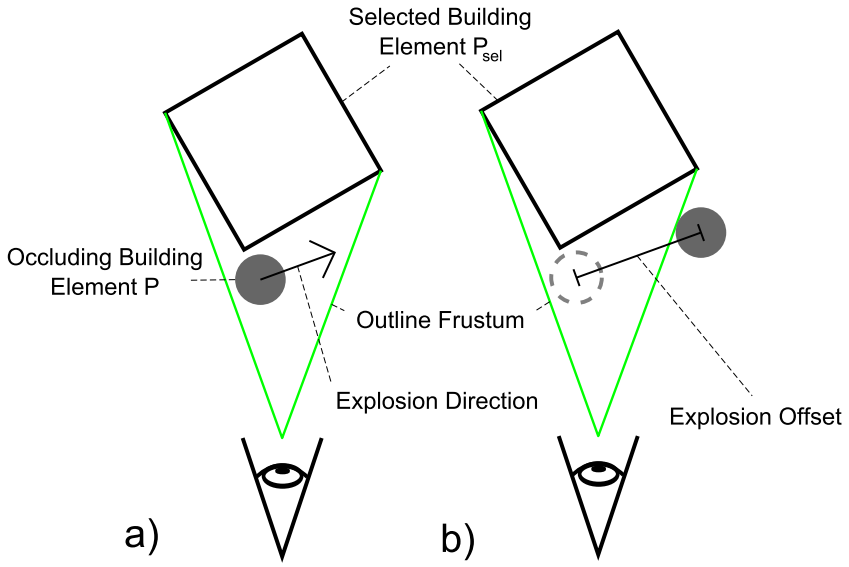


Figure 3.6: The building element P_{sel} is occluded by the building element P. Thus the outline frustum completely or partially contains P as shown in a). Therefore, P has to be moved as far into its explosion direction D_P to escape the outline frustum $Of(P_{sel})$. This is displayed in figure b).

3.4.2 Local Design Rules

Design rules implement design principles. I use the concept of local design rules and global design rules. Local design rules are just valid at a particular level of the hierarchical model. They can only be used with respect to the current hierarchy level and, therefore, are limited to a conventional architectural model, where a building consists of vertically separated floors and floors consist of horizontally separated rooms. The following design rules are applied for the different levels. The general design rules are always in effect. The remaining ones are only valid when focusing an element at the particular level of the hierarchy.

General:

LR1: Explode every building element P as far to escape the outline frustum $Of(P_{sel})$ of the selected element P_{sel} plus a screen offset. This is to ensure separation and compactness.

LR2: Explode all boundaries of the selected spatial structure away from it. This shall result in intuitive explosion directions.

LR3: Explode all parts which use the same explosion direction with the same explosion offset. This is to preserve the structure of the building.

LR4: Use the back-face boundaries of a selected spatial structure S_{sel} to shape S_{sel} and, therefore, do not explode them. They shall stick to the related, focused spatial structure S_{sel} to visualize it. This rule overrules the other contradicting local design rules.

Building:

LR5: Explode the parts of the façade of a building side horizontally as a whole with the same offset to reveal the floors.

LR6: Explode the roof and the surrounding environment vertically, if they occlude the building interior.

Floor:

LR7: Separate the selected building floor from the rest of the building vertically. Only do so if another floor occludes the focused one.

LR8: Explode the floor façade of the selected floor horizontally.

LR9: Explode the occluding slabs of the floor vertically.

Room:

LR10: Separate the floor containing the selected room from the rest of the building vertically if it occludes the floor.

LR11: Explode all other contained rooms and the façade of the floor vertically.

LR12: Explode the occluding slabs vertically.

LR13: Horizontally explode the walls of the room away from the selected room.

Remaining Building Element:

LR14: Consider the containing spatial structure and apply the corresponding design rules of its hierarchical level.

3.4.3 Transformation of Local to Global Design Rules

In order to test the viability of these local design rules, I have implemented them and they worked quite well. For the purpose of generalizing these local rules, I transform them to global design rules. As one can see, some of the local rules are very similar or even identical. For example, LR7 and LR10 could be easily combined. The global rules are created by abstracting, factoring and combining the local ones. The resulting global design rules are independent of the conventional architectural model. This means that it does not matter if the selected spatial structure is a building, a floor or a room. In all cases the same design rules are used. The transformation is done by exploiting the boundary and containment relationship information. The building is split along its boundaries. Therefore, I will introduce the concepts of the principal plane and the major boundary.

Principal Plane. The principal plane is used to implement the concept of context-sensitive explosion directions. A boundary, e.g., a wall or a slab, is assumed to be quite thin and flat. In order to facilitate computations and simplify the algorithm, a boundary B is reduced to a plane, the principal plane $Pp(B)$ of B. The principal plane $Pp(B, S)$ is used to determine the

explosion direction D_B of a boundary B relative to a spatial structure S. For this purpose, the surface normal vector of the principal plane is simply applied as the explosion direction for B. If the related spatial structure S is located at the front of the plane $Pp(B, S)$, the normal of the principal plane has to be flipped to obtain a proper explosion direction for B.

My first intent was to use a plane as principal plane that is obtained by applying the method of least squares to the vertices of the boundary. However, this plane is not a proper candidate to serve as a principal plane. The reason is that the so-created plane may not be parallel to any face or edge of the boundary (see Figure 3.7). Furthermore, the local rules restrict the explosion directions of the walls of the building to horizontal ones (see LR5, LR8, and LR13) and the explosion directions of slabs and roofs to vertical ones (see LR6, LR9, and LR12). These requirements are not guaranteed by the method of least squares. For example, most roofs are inclined and, therefore, also their principal planes calculated via this method would be inclined. For this reason, the explosion directions would be inclined too, not vertical.

In order to overcome this inappropriateness of the method of least squares, I apply a completely different approach to calculate a proper principal plane. It is easy to find suitable principal planes for slabs and roofs that ensure a vertical explosion direction (see LR6, LR9, and LR12). A horizontal plane that intersects the center of the boundary fulfils this requirement.

However, it is a little bit more difficult to obtain the principal plane for walls. A plane is spanned by two directional vectors. One vector direction of the principal plane is yielded by the constraint that the wall shall explode horizontally (see LR5, LR8, and LR13). Accordingly, the first vector has to be aligned vertically to ensure the constraint. In order to find the second vector of the plane, the maximal edge direction $Med(B)$ (see 3.1), as I refer to, is calculated. For this purpose, sets K_i of parallel edges E_b of a boundary B are formed. Then the lengths of the edges of each set are summed up to obtain a rating value per set. The maximal rating value of all sets of the edges E_B of a boundary B yields the second directional vector. It simply equals the corresponding direction of the edges of the set with the maximal rating value. The result is a vertically aligned principal plane. One has to note, that all vertical edges have to be excluded from the sets K_i to ensure that the second directional vector and the first directional vector are not parallel.

One can find examples for principal planes in Figure 3.8. The advantage of the maximal edge direction is that only actual edges are taken into account and the result is always in parallel with at least one edge. This method necessitates the elimination of plane edges described in Section 4.6.2.

$$Med(B) = \mathrm{dir}(\max_{i=1\ldots}\{\sum_{E_B \in K_i} |E_B|\}) \tag{3.1}$$

Major Boundary. By applying the concept of the principal plane, I have determined the explosion direction of all boundaries. However, the explosion directions of the spatial structures are not yet determined (specified by LR7, LR10, and LR11). In order to do so, I derive the

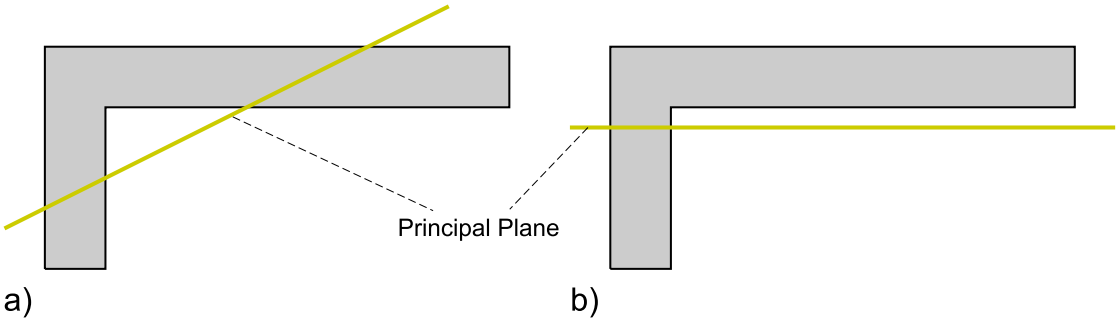


Figure 3.7: The figure shows a wall in the top view. a) demonstrates that the method of least squares does not provide a proper principal plane. One can see the maximal edge direction on the right-hand side in b).

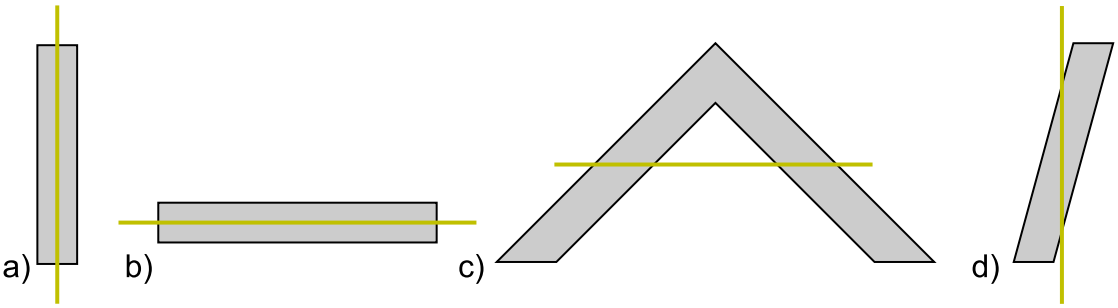


Figure 3.8: Here one can find different examples of principle planes. The considered boundaries are displayed in the side view. a) is a common wall. b) is a common slab. c) is a roof and d) is an inclined wall.

explosion direction D_S of a spatial structure S from its related boundaries B_S with a bounding relation $Br(S, B)$.

One of these boundaries of the spatial structure S is chosen to be the major boundary $Mb(S)$ of S. This is done dynamically for a particular explosion status. The major boundary $Mb(S)$ of a spatial structure S is used to determine the explosion direction of S. The method just applies the explosion direction of the major boundary to the spatial structure. The major boundary $Mb(S)$ of spatial structure S has to satisfy following constraints:

- It has to be exploded itself.
- It must have a higher or equal hierarchy level then the spatial structure S.
- It shall be the nearest boundary to the selected spatial structure S_{sel}. The bounding relations of the building are used to calculate the distance. The nearest boundary B is the one with the fewest amount of boundaries and spatial structures on the shortest bounding relation path between B and the selected spatial structure S_{sel}.

- Its hierarchy level shall be the highest among all nearest boundaries.
- The above-mentioned constraints may result in multiple major boundary candidates. As a result, the one is taken with the greatest explosion offset (the explosion offsets are highly unlikely to be equal when they differ from zero). This is to emphasize the part separation.
- If the explosion offset of all found major boundary candidates of the highest level is zero, then take the one with the next lower hierarchy level.

The constraints shall ensure that spatial structures prefer to stick to other spatial structures of higher levels. This is to preserve the context and obtain a meaningful membership while exploding.

3.4.4 Global Design Rules

Global design rules are rules that have to be valid at each level of the hierarchical model. They completely replace the set of local design rules. In Section 3.4.5, one can find a detailed enumeration of the concrete replacements.

In the following, I will list the global design rules. As an initial condition, it is presumed that the user selects a building element which she/he is interested in.

Focus the selected building element:

GR1: The system shall reveal the selection by exploding all occluding parts.

GR2: Turn the camera towards the selected building element.

GR3: If the selected building element is not a spatial structure, then the selected spatial structure is its parent spatial structure.

Visualize the selected spatial structure:

GR4: Do not explode boundaries of the selected spatial structure if they do not occlude the selected spatial structure. This way the back-face boundaries offer a visual representation of the selected spatial structure.

GR5: Draw the edges of the selected spatial structure color-rimmed to emphasize the selection.

GR6: Explode all boundaries of the selected spatial structure away from it.

GR7: Explode all boundaries orthogonal to the principal plane of the boundary.

Sticking boundary relation:

GR8: Stick very small boundaries to bigger ones while exploding. This is to reduce part-fragmentation and explosion path crossing (see Figure 3.9).

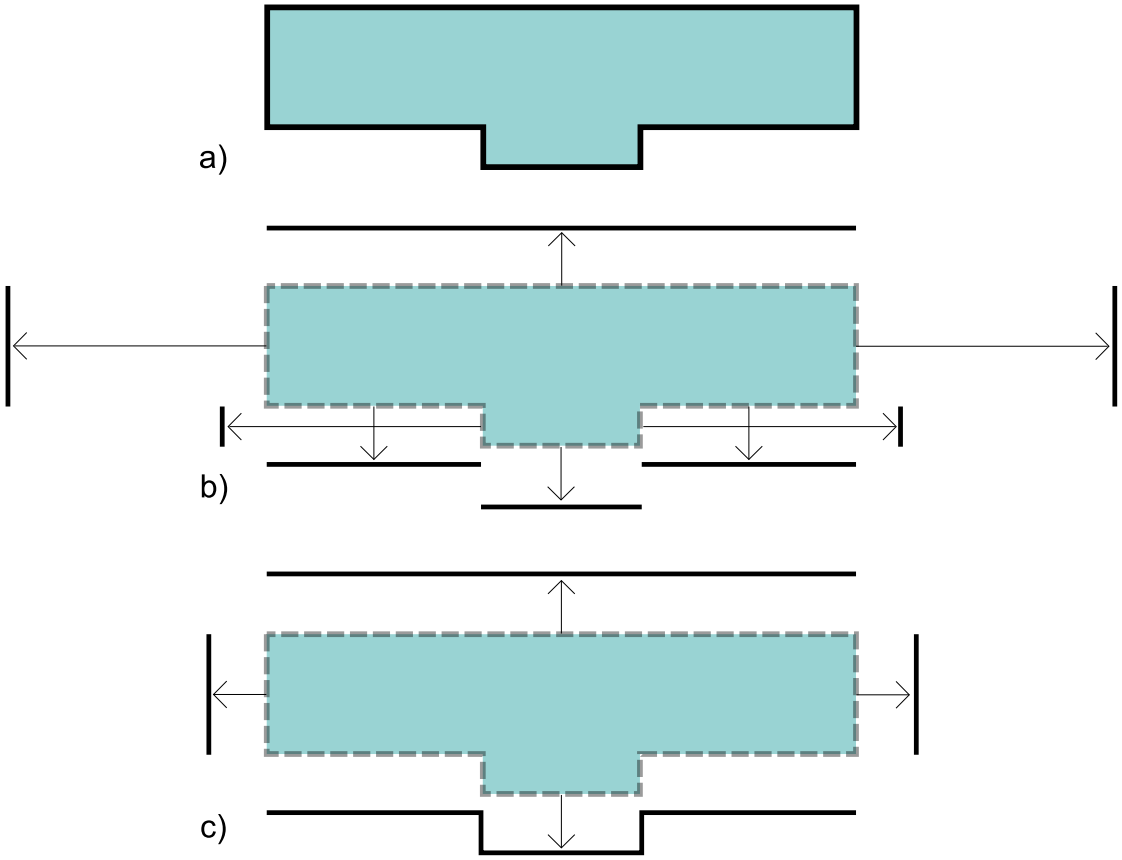


Figure 3.9: a) shows the considered spatial structure and its boundaries. b) demonstrates the problem of fragmenting the exploded structure. c) shows the result of sticking small boundaries to big ones.

Preserve the exploded structure:

GR9: Explode spatial structures into the same direction as their major boundaries.

GR10: Explode everything that explodes into the same direction with the same offset. Choose the maximal offset out of all exploding elements that explode into this direction.

GR11: Only explode a spatial structure if no ancestor of it explodes. This results in only exploding the elements at the highest possible level of the hierarchy. Therefore, the building structure is preserved.

GR12: A building element P is just exploded if it is neither a descendant nor an ancestor of the selected spatial structure S_{sel}.

Balance separation and compactness:

GR13: Explode a building element so far that it escapes the outline frustum of the selected spatial structure.

GR14: Add a screen offset to the explosion offset to emphasize the separation of exploded parts and unexploded parts.

Explosion Lines:

GR15: Use few but meaningful lines to visualize explosion paths. The explosion lines shall provide an aid to enable the viewer to mentally reconstruct the exploded parts. There are two aspects that facilitate that. On the one hand, there should not be too many of them because else they result in visual clutter. In this case the explosion lines may detract from the actual parts of the exploded view.

On the other hand, the lines shall be placed at a proper position. This is necessary to empower the user to relate an explosion line to its corresponding part. An edge or a vertex of the geometry of the exploded part is a good candidate to provide a traceable path for an explosion line.

Avoid visual clutter while animating the explosion:

GR16: Chose proper explosion directions and offsets, to minimize explosion animations and explosion-path crossing during selection change. This is implicitly realized by assuming that the principal planes of boundaries of two adjacent hierarchy levels are orthogonal (e.g., slab to wall).

GR17: Do not mix user-camera interaction and explosion animation. Do animation afterwards.

3.4.5 Replacement of Local with Global Design Rules

In the following, one can find the list of replacements of local via global design rules. In some cases, a set of multiple global rules replace a single local rule, or a single global replaces multiple local rules. Furthermore, newly added global design rules are separately mentioned.

- The local design rule LR1 is distributed to the global design rules GR1, GR13, and GR14.
- LR2 is replaced by GR6.
- LR3 is replaced by GR10.
- LR4 is replaced by GR4.
- LR14 is replaced by GR3.
- LR5, LR6, LR8, LR9, LR12, and LR13 are implemented by the single global design rule GR7. For this, I have introduced the concept of the principal plane.
- LR7, LR10, and LR11 are substituted by the set of global rules GR9, GR11, and GR12. In order to do so, the concept of the major boundary is used.
- The remaining global design rules extend the set of local design rules and improve the visualization (GR2, GR5, GR8, GR15, GR16, and GR17).

3.5 Algorithm

In order to implement the above-mentioned design rules, I apply a greedy algorithm. That means, the algorithm is executed step by step and the system takes the best result of the state of one step to proceed to the next step. My approach is to use the boundaries to determine the explosion offset and the explosion direction of bounded spatial structures. If one considers the bounding relations of the boundaries to their bounded spatial structure as a connected graph, then the algorithm performs a breadth-first search in this graph starting with the selected element P_{sel}. Accordingly, building elements which are nearer to P_{sel} are exploded first. This propagation of the explosion reflects the process of a real explosion.

The algorithm starts by exploding the boundaries of the selected spatial structure away from it. All already exploded boundaries are collected in a boundary list. Every boundary in this list is rated with a priority level that represents the order at which the items of the list are processed. The boundaries of the list are sorted according to their priority level. A detailed description of the priority level can be found below. Afterwards, the boundaries of the list are iterated, starting with the one with the highest priority: First of all, the currently considered boundary B is removed from the boundary list. Then, each spatial structure S_B with a bounding relation to B that are not already processed is exploded. The currently iterated boundary B is the major boundary of these spatial structures. Therefore, the explosion direction of each spatial structure S_B is determined by the explosion direction of B. The explosion offset of S_B is calculated by using the outline frustum of the selected element P_{sel}. Furthermore, each boundary B_{S_B} of each recently exploded spatial structure S_B is moved into the same direction if B_{S_B} is not already processed. Each of these boundaries B_{S_B} is collected in a new boundary list for the next step of the algorithm. After the old boundary list is empty, i.e., every boundary B in the list was processed, the algorithm starts again with the new boundary list.

This procedure continues until no new boundaries are added to the boundary list at a step. After that, every spatial structure and every boundary was processed if the data corresponds to the hierarchical model defined above. At the end, the remaining building elements R that are no boundaries and no spatial structures have to be handled. The major boundary of the spatial structure that contains R is taken to determine the explosion direction of R.

The last step of the algorithm determines the final explosion offset of the exploding parts. For this, all parts that share the same explosion direction are collected in sets. The greatest explosion offset of the parts of a set is applied for the explosion of all parts of the set. Accordingly, parts that explode into the same direction also explode with the same offset and, therefore, the structure is preserved (see global rule GR10).

Priority Level. The priority level determines the order in which the boundaries are processed to explode their adjacent spatial structures, i.e., it impacts the choice of major boundaries. It is a result of the combination of the hierarchy level and the explosion length. The boundaries are rated and sorted according to their priority level. A higher hierarchy level results in a higher rating. Accordingly, spatial structures favour boundaries with a higher hierarchy level to stick to while exploding. If two boundaries have the same hierarchy level, their explosion offsets are

compared. The one with the greater explosion offset gets the higher rating. A boundary with an explosion offset of zero represents an exception. It is put to the end of the list. The influence of the explosion offset is used to emphasize part separation.

3.5.1 Preprocessing

The algorithm of the following sections is performed every time the user selects a building element or moves the camera and, therefore, changes the explosion status. But there are also pre-calculations, that just have to be conducted once at the loading of the building data.

First of all, the hierarchical model has to be deduced from the building meta and geometry data. Then, for every boundary, there has to be calculated the principal plane and the explosion direction relative to each covered spatial structure. Small boundaries that should stick to greater adjacent ones are determined. They shall use the same principal plane and explosion directions as the greater boundaries they stick to. The implemented method simply compares the diagonals of every two adjacent boundaries. If one is shorter than a particular percentage of the longer one, it has to stick to it during the explosion process. It is mentionable that the sticking boundary relation of two considered boundaries is calculated multiple times relating to different spatial structures. Accordingly, the relation can vary for the same two boundaries according to different spatial structures as one can see in Figure 3.10.

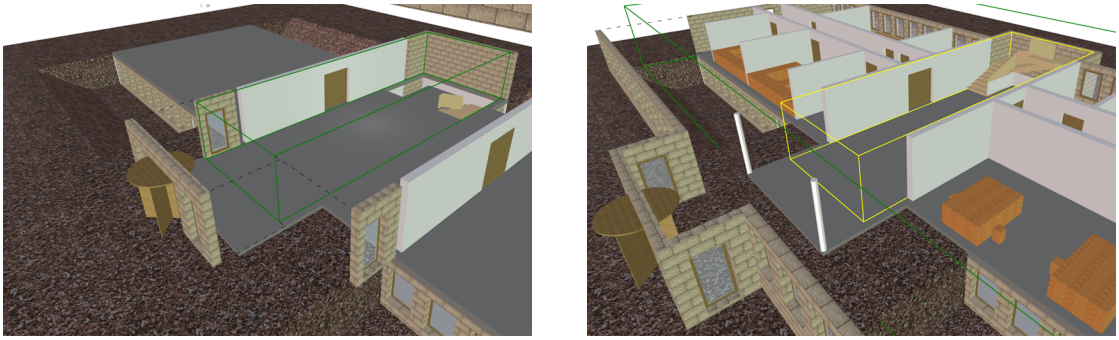


Figure 3.10: The two figures display the sticking boundary relations of a wall relating to two different spatial structures. The behaviour of the small wall with the window at the right-hand side of the entrance should be noted. In the left figure, the explosion direction of the considered wall corresponds to the normal vector of its principal plane. In the right figure, the wall sticks to the big façade wall and, therefore, applies the principal plane of the latter. This difference is produced by the fact that two diverse spatial structures are selected in the two figures (marked with the green bounding box) and so there are diverging wall adjacency relations. These result in different explosion directions.

3.5.2 Verbose Description

In the following, one can find a more detailed description of the algorithm separated into its different steps. This enumeration does not take into account any pre-calculations. Starting with

the selected spatial structure, the explosion is propagated to the other building elements. This simulates the process of a real explosion.

1. The user selects a building element P_{sel} or interacts with the camera.
2. If the selected element is not a spatial structure, then select its parent spatial structure (i.e., the spatial structure containing the element).
3. Explode all boundaries $B_{P_{sel}}$ of the selected spatial structure P_{sel} into the explosion direction according to their principal planes. Only do so if they are front-faces of P_{sel}. Use the outline frustum of the spatial structure P_{sel} to calculate the explosion offset of the boundaries. Collect the boundaries in the boundary list Bl.
4. Calculate the priority levels for every boundary in Bl and sort Bl according to these values.
5. For every boundary B of the boundary list Bl perform following steps:
   - Explode each not yet processed and by B bounded spatial structure S_B into the same direction as B. Thus calculate the explosion offset for each S_B via the outline frustum of P_{sel}. Only explode a spatial structure S_B that is not a descendant of any other already exploded spatial structure (GR11).
   - Explode each boundary B_{S_B} of each previously handled spatial structure S_B the same way as described in the previous step if B_{S_B} is not already exploded. Collect each recently exploded B_{S_B} in a new boundary list Bl_{new}.
6. Replace the boundary list Bl with the new boundary list Bl_{new}.
7. If the boundary list is not empty go to 4. Else proceed with 8.
8. Explode each remaining building elements R of the building into the direction determined by the major boundary of its parent spatial structure.
9. Iterate all exploded building elements. Form sets of elements that explode into the same direction. Find the maximal explosion offset of the elements per set. Explode the building elements of a set with the maximal explosion offset of the set.

3.5.3 Pseudocode

One can find the pseudo code of this algorithm in Algorithm 3.1. The function `SortBl` sorts and returns the given boundaries by their priority levels. The function `CalcDViaPp` returns the explosion direction D for a boundary B relative to the a spatial structure calculated via the principal plane of B. The function `GetAllR` returns all remaining building elements of the building. The function `HasAncestorSharingB` returns true if any ancestor of the considered spatial structure shares the considered boundary with it. This function is needed to prevent a spatial structure to be exploded if any ancestor should rather be exploded (see global rule

GR11). The data structure `sameDSets` is a hash map. The keys of the hash map are the normalized explosion directions and the values of the hash map are lists of all building elements to be exploded into these directions. The function `IsExploded` returns true if a particular element has been already processed.

The function `CalcO` calculates the explosion offset of a building element. It does this by moving the element as far into its explosion direction to escape the outline frustum of the selected spatial structure. However, the back-face boundaries of a selected spatial structure shall not be moved (see global rule GR4). This is to shape the form of the selected spatial structure. I use the following approach to ensure this: if the explosion direction of one of these boundaries points towards the camera, the outline frustum is used to calculate the explosion offset. If the explosion direction points away from the camera, the function just returns zero for the explosion offset. In this case, the boundary is assumed to be a back-face boundary.

It may occur that a spatial structure has no boundaries. The building site is an example for this. If such a spatial structure is selected, then just no boundaries are exploded and so there explodes nothing at all due to the character of the algorithmic approach. So if the user selects the building site, the building is displayed completely unexploded. However, if such a spatial structure with no boundaries contains remaining building elements that should be exploded, there are no boundaries to determine the explosion directions of the remaining building element. Default explosion directions are applied in those cases (e.g up, down,...).

3.6 Relation to Formal Grammar

The idea of context-sensitive rules is not a new one in the field of informatics. Also in the domain of formal grammars, there exists the term context-sensitive. A formal grammar is a rule set to generate a formal language. It consists of non-terminal symbols N, terminal symbols Σ and production rules Φ. The production rules are replacement rules for non-terminal symbols. In Formula 3.2, one can see a formal definition of production rules.

$$(\Sigma \bigcup N)^* N (\Sigma \bigcup N)^* \rightarrow (\Sigma \bigcup N)^* \tag{3.2}$$

A context-sensitive formal grammar, also called type-1 grammar of the Chomsky hierarchy, may add a surrounding context to the left-hand side and right-hand side of any production rule. Therefore, they are less general than unrestricted grammar. All production rules of a context-sensitive grammar are of the form $\alpha A \beta \rightarrow \alpha \gamma \beta$, where $\alpha, \beta \in (\Sigma \bigcup N)^*$, $A \in N$ and $\gamma (\Sigma \bigcup N)^+$. α and β are the context of A and remain during the applying of production rules. It is remarkable that A gets replaced by γ and γ is not empty.

The grammar can be used to express the principles of the design rules in a formal way. The context of a non-terminal symbol can be exploited to represent the context of building elements according to the work at hand. In the following, I will reduce the problem of exploding a building to the one-dimensional space. This means that a building element P can only explode into the left or into the right direction or be unexploded, referred to as the symbols $P^{\leftarrow}$, $P^{\rightarrow}$, and P^{-}. The non-terminal symbols P, S, B, and R relate to the differ-

input : the hierarchical model, the geometry, all building elements, the selection S_{sel} and the camera
output: the exploded positions of the building elements

```
1  foreach B in S_sel.Br do // explodes all boundaries of the selected S
2      B.D ← CalcExdViaPp(B, S_sel);
3      B.O ← CalcO(B, S_sel, camera);
4      sameDSets [B.D ].Add(B);
5      Bl.Add(B);
6  end
7  while Bl.Size > 0 do // iterates all new Mb until no Mb are added to Bl
8      SortBl(Bl);
9      Bl_new ← new List;
10     foreach B in Bl do
11         foreach S in B.Br do // explodes all S of B according to their Mb
12             if HasAncestorSharingB(S, B) or IsExploded(S) then
13                 continue;// does not process if S or any ancestor is exploded
14             end
15             S.D ← B.D;
16             S.O ← CalcO(S, S_sel, camera);
17             sameDSets [B.D ].Add(S);
18             foreach B_S in S.Br do /* explodes all B_S according to the Mb of S
                   and adds them to Bl if not yet exploded                    */
19                 if IsExploded(B_S) then
20                     continue;
21                 end
22                 B_S.D ← B.D;
23                 B_S.O ← CalcO(B_S, S_sel, camera);
24                 sameDSets [B.D ].Add(B_S);
25                 Bl.Add(B_S);
26             end
27         end
28     end
29     Bl ← Bl_new;
30 end
31 Rl ← GetAllR();
32 foreach R in Rl do // explodes all R according to the Mb of the parent S
33     B ← SortBl(R.ParentS.Br);
34     R.D ← B.First().D;
35     R.O ← CalcO(R, S_sel, camera);
36     sameDSets [R.D ].Add(R);
37 end
38 foreach D in sameDSets.Keys do // sets the O of every P with same D to maxO
39     maxO ← 0;
40     foreach element in sameDSets [D ] do
41         if element.O > maxO then
42             maxO ← element.O;
43         end
44     end
45     foreach element in sameDSets [D ] do
46         Explode(element, D, maxO);
47     end
48 end
```

Algorithm 3.1: Algorithm

ent types of building elements defined in Section 3.1. The term $S[P]$ refers to the containment relation $Cr(S, P)$. The terms $S[P]B$ and $BS[P]$ corresponds to the bounding relation $Br(S, B)$. In the following, one can find the grammar to generate the hierarchical model defined in Section 3.3. The grammar is defined by $(Start, P, S, B, R, P^{\leftarrow}, P^{\rightarrow}, P^{-},) \in N$, $([,], S_{sel}, S^{\leftarrow}, S^{\rightarrow}, S^{-}, B^{\leftarrow}, B^{\rightarrow}, B^{-}, R^{\leftarrow}, R^{\rightarrow}, R^{-}) \in \Sigma$ and (3.3, 3.4, 3.5, 3.6, 3.7, 3.8) $\in \Phi$.

$$Start \rightarrow P \tag{3.3}$$
$$P \rightarrow R \tag{3.4}$$
$$P \rightarrow BS[P]B \tag{3.5}$$
$$P \rightarrow PS[P]B \tag{3.6}$$
$$P \rightarrow BS[P]P \tag{3.7}$$
$$P \rightarrow PS[P]P \tag{3.8}$$

However, not only the generation of the hierarchical model but also the explosion itself could be described by a context-sensitive grammar. For this, one has to extend the production rules above. I will not enumerate all necessary production rules, since there are so many of them and their formulation is quite complex and would exceed the scope of this work. Furthermore, this is just to demonstrate the relation of context-sensitive explosion directions (see Section 3.1 and 3.2) and context-sensitive grammars. The rules 3.9, 3.10, 3.11, 3.12, and 3.13 represent an excerpt of necessary production rules. One may realize that the explosion direction of P in 3.11 and 3.12 depends on the surrounding terminals $B^{\leftarrow}$ and $B^{\rightarrow}$. This is the context of P which results in different symbols to replace P.

$$S[P] \rightarrow S_{sel}[P] \tag{3.9}$$
$$BS_{sel}[P] \rightarrow B^{\leftarrow}S_{sel}[P] \tag{3.10}$$
$$PB^{\leftarrow} \rightarrow P^{\leftarrow}B^{\leftarrow} \tag{3.11}$$
$$B^{\rightarrow}P \rightarrow B^{\rightarrow}P^{\rightarrow} \tag{3.12}$$
$$BS[P^{\leftarrow}] \rightarrow B^{\leftarrow}S[P^{\leftarrow}] \tag{3.13}$$

One has to note that, if the problem is extended to the three-dimensional space, the set of symbols representing exploded parts ($P^{\leftarrow}$ and $P^{\rightarrow}$) must also be extended to all occurring explosion directions. Moreover, an alternative term for the bounding relation $Cr(S, P)$ must be defined, since there can be arbitrary many bounding relations, not only a left and a right one.

CHAPTER 4

Implementation

In this chapter, I will go into detail about the implementation of the presented concepts of the previous chapter. I have implemented the ExVAr-system (Exploded Views for Architecture) to realize the above-described approach to generate exploded views for buildings. In the following I will describe details about the implementation, the used data format, the framework, and the concrete realization of the user interface of the ExVAr-system.

4.1 Data Format

The heart of the data format are the Industry Foundation Classes (IFC, www.buildingsmart.org). IFC is in the process to become an international standard (ISO/IS 16739). The IFC standard data schema is developed and maintained by BuildingSMART, an international not-for-profit organization. The aim of their work is to create a standard for exchanging BIM (Building Information Model) data between different applications of various vendors. The BIM standard adds semantic and meta information to the geometrical representation of a building. This contains information about rooms, floors, building parts, etc., and also about the building process. Each building element owns numerous properties that qualify it, however, the standard also allows defining additional properties to support extensibility. Furthermore, it implements relations between building elements, e.g., containment relations between rooms and floors, between doors or windows and walls, but also bounding relations between spaces and walls, and so on. So it is a proper format to represent the architectural theoretical approach of space and hull but also allows to realize the hierarchical model (see Section 3.3.1).

The current version of IFC is IFC4, however, this version is not yet implemented by many applications and SDKs. As a result, I decided to use the foregoing version IFC3x2. In order to conform to the hierarchical model of Section 3.3.1, the extracted data from the IFC file has to be rearranged. All the needed meta information can be derived from the IFC format. For example, the ExVAr-system modifies the hierarchy level of boundaries, doors and windows, but also adds bounding relations to floors and façades.

The IFC standard is also intended to provide material information and, therefore, defines texture surfaces and texture coordinates. Unfortunately, none of the importers or exporters of the IFC file format I have tested includes this texture information. However, to support textured geometric models, I use the Wavefront OBJ file format to complement this information. All the geometric information is taken from the OBJ file, all meta and relation information is read from the IFC file. Nevertheless, the IFC format also includes the geometric information (see Figure 4.1 for geometrical representations in the IFC standard), so the ExVAr-system also permits omitting the OBJ file.

The OBJ fomat was developed by Wavefront Technologies. It supports vertex-based data, but also free-form curves and surfaces. Furthermore, normals and texture coordinates can be defined and material information can be qualified in a .mtl-file that also supports image textures.

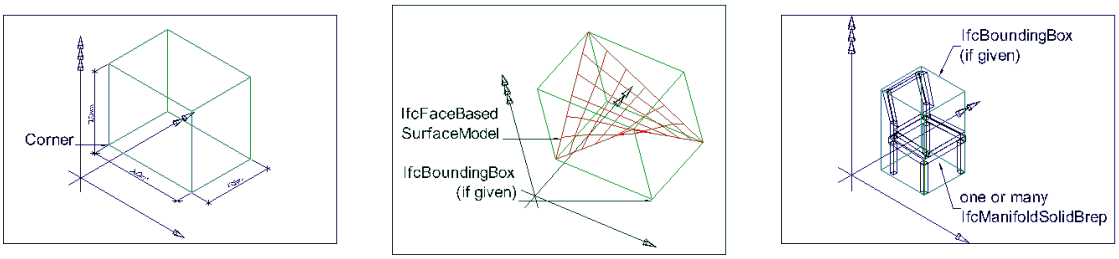


Figure 4.1: The three figures show different geometry representations of the IFC standard. This can be a box, face-based or combined brep representation. The images are taken from `www.buildingsmart-tech.org`

4.2 Data Samples

In the following, one can find a list of all five used sample building models (Figure 4.2, 4.3, 4.4, 4.5 and 4.6). Three of them are downloaded from IFC-libraries. Additionally, I modelled one by myself and the last one was created by my brother, Georg Fleiß. I have chosen different kinds of buildings to see the performance of the ExVAr-system applied to different types of architecture. One can find two one-family houses, an apartment building, an office building and a university institute among the sample models. The set of models contains buildings with diverse amount of floors, and greatly varying in size. This is to test the ExVAr-system on models with different scale of complexity and dimension.

Unfortunately, I had to rework most of the names of the building elements of the models because they must be unique for the purpose of matching them to the corresponding OBJ file geometry, otherwise no textures could be used. In a few cases, it was also necessary to add some missing boundary-relation information.

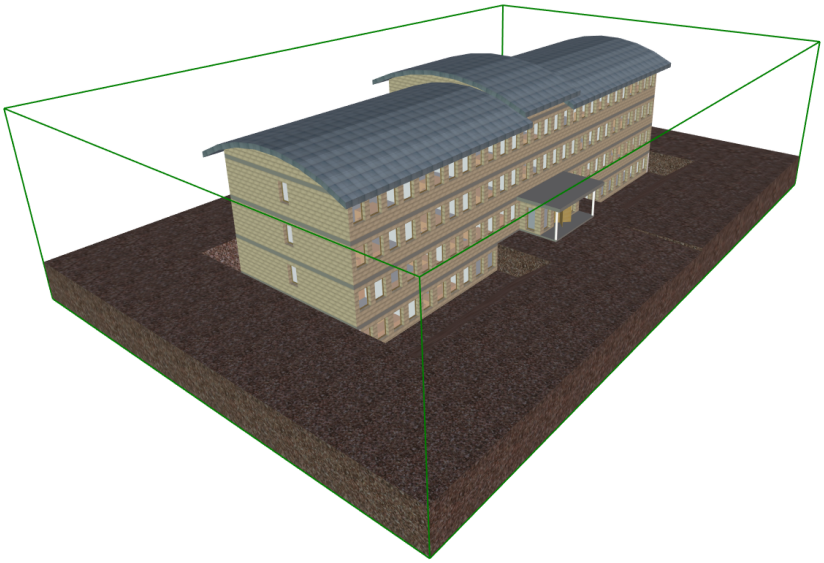


Figure 4.2: Model *Institute*, http://www.iai.fzk.de/www-extern-kit/fileadmin/download/download-vrsys/AC11-Institute-Var-2-IFC.zip

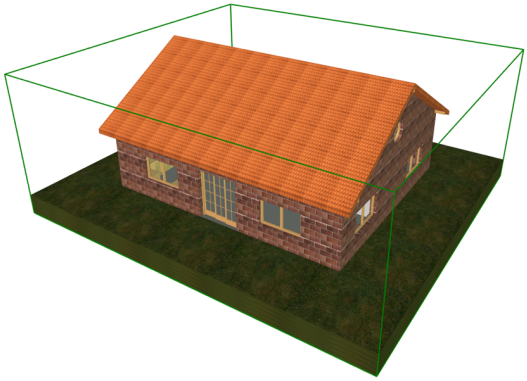


Figure 4.3: Model *House*, http://iai-typo3.iai.fzk.de/www-extern-kit/fileadmin/download/download-vrsys/AC14-FZK-Haus.zip

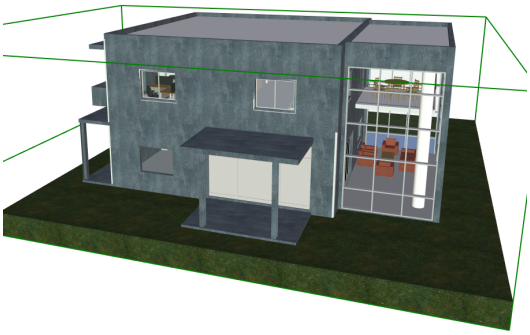


Figure 4.4: Model *Tofu*, http://www.suplugins.com/download/Sample_IFC_files.zip

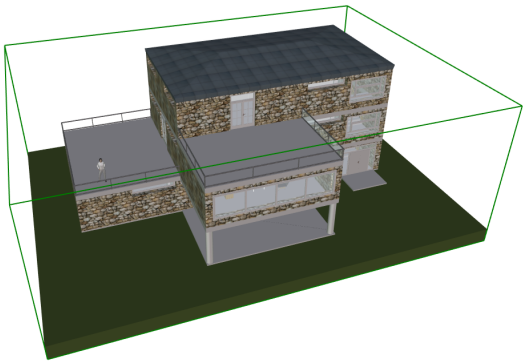


Figure 4.5: Model Own, created by myself

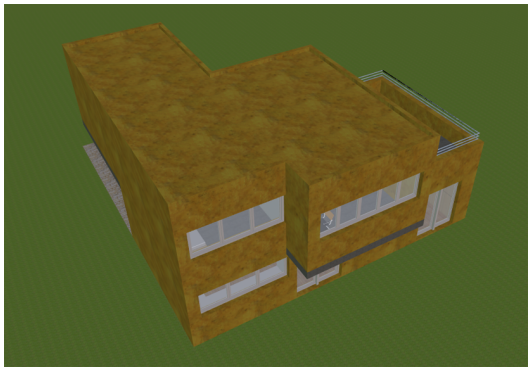


Figure 4.6: Model $Pichlgasse$, by courtesy of Georg Fleiß

4.3 Framework

The base of the used framework for the implementation of the ExVAr-system is the SharpDX version of the Helix 3D Toolkit (`github.com/helix-toolkit`). It is implemented in C# and uses DirectX 11. It provides a scene graph and different types of renderable objects, like meshes or lines. Furthermore, it supports lighting, texturing, and camera navigation. A perspective view and an orthographic view is supported. I apply a perspective view because it intensifies the concept of getting an insight into the interior structure. The toolkit enables the developer to easily create user interfaces and facilitates controls for 3D operations, since it is embedded in WPF (Windows Presentation Foundation). An OBJ-reader is also available in the toolkit. The building element class of the ExVAr-system (see Section 4.4) inherits from the textured mesh class of the Helix 3D Toolkit. Furthermore, I use the open source development toolkit xBIM (eXtensible Building Information Modelling, `www.openbim.org`) to load the data from the IFC file into the ExVAr-system. It is a .NET toolkit that supports the BuildingSMART data model and enables the developer to easily read, write and update IFC files. It offers full support for geometric and topological operations. The toolkit is implemented in C# and, therefore, extension methods can be used to extend the functionality of the xBIM IFC-classes.

4.4 ExVAr Classes

In order to facilitate the development process and to support a trial-and-error approach, I designed the ExVAr classes with the purpose of preparing the system to be extensible and single classes to be easily replaceable. For the purpose of offering the possibility to reimplement parts of the algorithm, I extensively use interfaces. This allows the user to replace modules of the implementation via the user interface by combo boxes.

Furthermore, I applied the three-tier architecture model to separate the presentation, the logic and the data of the ExVAr-system. This approach is supported by the use of WPF that implements the MVVM (Model-View-ViewModel) design pattern. The structure of the ExVAr-system

is shown in Figure 4.7. The presentation part is done by the `MainView`, the `MainViewModel` and its renderer.

The `ExplosionGraph` class performs the actual explosion of the building and, therefore, calculates the new positions of the exploded parts. It uses the `ExplosionLengthCheck` to check if a part shall be exploded and how far the part shall be moved from its origin position if it is exploded.

The `BlockingCheck` class is used by the `ExplosionGraph` to calculate blocking relations of a part to another into the given directions.

The `DataLoader` class handles the loading of the IFC and OBJ files and the conversion of the IFC data and the integration of the OBJ data. It creates the `ExplosionElement` instances and builds the `ExplosionGroups`.

The `ExplosionElement` class holds the geometrical representation of the building elements and the relations between them. Furthermore, it qualifies the building elements by including type information and other properties of it.

The `ExplosionSpatialStructure` class inherits from the `ExplosionElement` class. It represents the above-defined spatial structures.

The `ExplosionGroups` class determines the grouping information of the building elements and, therefore, can hold the hierarchy relations of the above-defined hierarchical model (see Section 3.3.1). The grouping information and the `ExplosionElement` class is used as a data input for the `ExplosionGraph`. The data classes also hold the state of the explosion.

The `ExplosionGraph`, `ExplosionLengthCheck`, `ExplosionGroups` and `BlockingCheck` are interfaces and thus can be implemented by various classes using different approaches and methods. For example, I implemented parts of the algorithm of Li and Agrawala et al. [LACS08] in the class `AgrawalaExplosionGraph`. The `ExVArExplosionGraph` class holds the implementation of the my own approach presented in the previous chapter.

The user can switch between these implementations via a combo-box. But she/he can also replace a fraction of the algorithm, e.g., the used calculation for the explosion offset of an exploding part. In this regard, I implemented a fixed explosion offset in the class `FixExplosionLength` and a method that applies the above-defined outline frustum of the Section 3.4.1 in the class `FrustumExplosionLengthCheck`.

4.5 User Interface

The user interface is designed to support an interactive exploration of a building (see Figure 4.8). The navigation mechanics apply the hierarchical model (see Section 3.3.1: a building consists of floors, floors consist of rooms, etc.). Consequently, this model must be communicated to the user to provide a deeper understanding of the mechanics. To achieve this, the user interface supports two navigation modes.

The first one is the hierarchical view. It contains a simple expandable structural tree that represents the hierarchical model of the building. A node of the tree stands for a particular

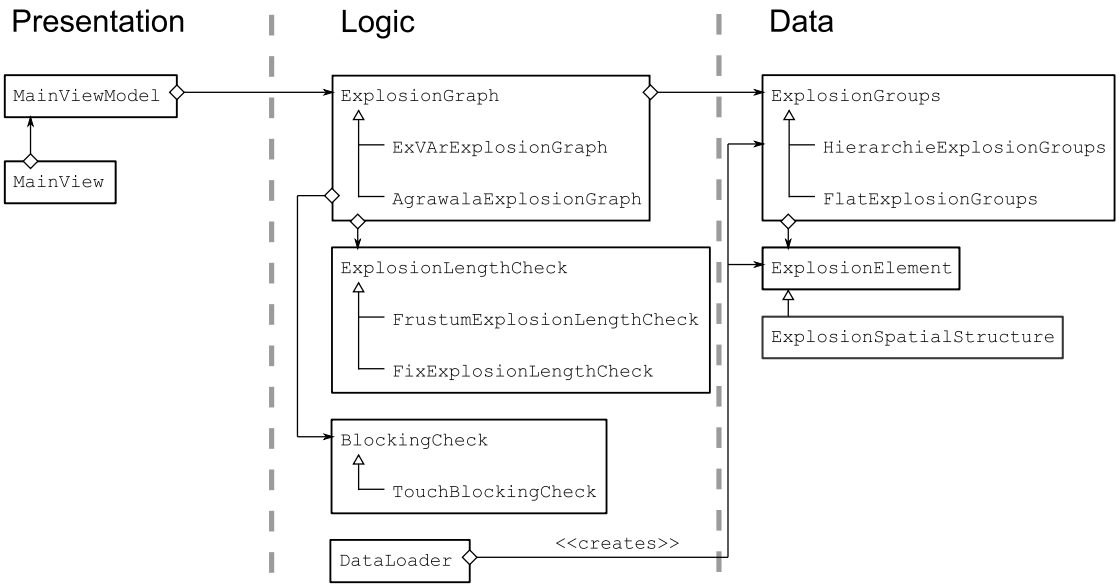


Figure 4.7: The figure shows the main classes of the ExVAr-system and the inheritance and assignment relations. The ExVAr-system applies a three-tier model for the architecture of the system. The `ExplosionGraph` class implements the algorithm explained in the previous chapter. The `ExplosionGroups` class holds the above-defined hierarchical model.

building element. The root of the tree is the building site, its child is the building and so on. But not every building element has a corresponding tree node. Some types of building elements are omitted to reduce the size and complexity of the tree of the hierarchical view. For example, walls, slabs, and roofs cannot be selected via the hierarchical view. The focus of this approach are spatial structures. The assumption is that the omitted elements always belong to one of these spatial structures. The user can mouse-click on a node of the hierarchical view to select a particular building element. This results in an update of the main view.

The main view renders the actual exploded view of the building and, therefore, reveals the selected building element. Thereby it applies the design rules described in the previous chapter to satisfy the defined design principles (see Section 3.2.3 and following). The building elements are rendered at the positions determined by the current explosion status. The translation paths of exploded elements are visualized by thin, dotted explosion lines.

The main view provides the second navigation mode. This is done by an implicit approach. The user directly interacts with the building and its exploded view via mouse actions. In order to select particular building elements, the user has to left click on them. In the main view, the user can only select building elements that are direct children of the current selection. As a result, the amount of possible selections is reduced and ambiguity and complexity of the selection process is reduced. The left-click operation can obviously only select a building part of a lower hierarchy level. Hence, I also implemented a right-click operation that selects the parent of the current selection. In this way, every building element of the hierarchical model can be reached via the

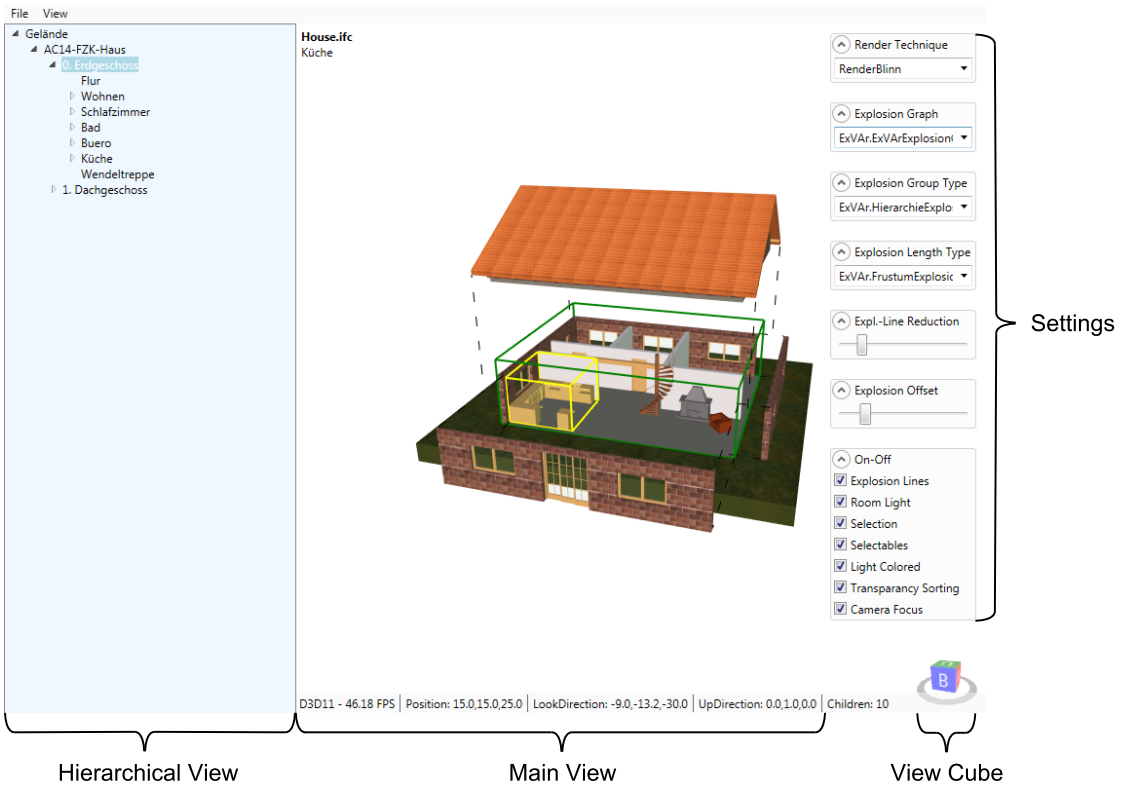


Figure 4.8: The figure displays the interface of the ExVAr-system. At the left-hand side there is the hierarchical view, at the right-hand side one can see the render and algorithm settings. The main view is placed in the middle. The camera view cube located at the lower right corner enables the user to rotate the camera to axis-aligned camera viewing directions.

navigation mode of the main view. When the user selects a building element, the exploded view rearranges to reveal the selected part (see global rule GR1).

The above considerations correspond to the hierarchical model and may communicate it to the user. Of course, the user has to know at which level of hierarchy the current selection is located to successfully make use of this navigation. Therefore, the corresponding node of the hierarchical view is also selected when the user navigates via the main view. Finally, the current selection is emphasized by rendering the edges of the selected building element in green (see global rule GR5).

On the other hand, a possible selection is rendered yellow-rimmed. It occurs when the mouse cursor hovers over a building element that can be selected via the left click operation at that time (i.e., depending on the current selection). Additionally, the name of the possible selection is displayed at the left upper corner of the main view.

The main view also provides camera navigation via the mouse. All camera navigation is handled via the middle mouse button. The user can rotate the camera around the selected build-

ing element by pressing the middle mouse button and simultaneously dragging the mouse into a direction. When the middle mouse button is released the explosion view is rearranged relative to the current camera position.

Camera zooming is done by mouse-wheel scrolling or the SHIFT modifier key and the middle mouse button. Furthermore, the camera can be translated in parallel to the camera plane by pressing the CRTL modifier key and the middle mouse button. Besides, a camera view cube located at the lower right corner enables the user to rotate the camera to axis-aligned camera viewing directions.

There are also additional parameters the user can change during runtime. She/he is empowered to replace parts of the explosion algorithm via combo-boxes as mentioned above. But it is also possible to change other parameters like explosion line density or the explosion offset via the user interface. Furthermore, the user can turn off or on other functionality like the directional light color.

4.6 Concrete Implementation

In the following, I pick the concrete implementations of some of the concepts discussed in the previous chapter and explain them. These implementations just represent possibilities of realizing these concepts and could be easily exchanged because of the software architecture of the ExVAr-system.

4.6.1 Explosion Lines

The explosion lines visualize the explosion path of an exploded part. For a proper traceability, only few but meaningful explosion lines shall be generated corresponding to the actual geometry of parts of the building (see global rule GR15). My approach is to find vertices of the geometry of exploded parts which can be used to generate such lines. All parts P_{D_i} exploding into the same direction D_i are collected in sets. Afterwards, the explosion lines for these different sets are created. The idea is that the parts P_{D_i} of a set stick together while exploding and so also shall share explosion lines.

For this, all vertices V_{D_i} of the building elements P_{D_i} of one set are projected onto a plane $Elp(D_i)$ that is orthogonal to the explosion direction D_i. Then, the convex hull of the projected vertices V'_{D_i} is calculated. All other vertices that are not part of the convex hull are dropped. Afterwards, the method iterates the convex hull clockwise. If three sequenced convex hull vertices (V'_{j-1}, V'_j, V'_{j+1}) are approximately aligned, it drops the one in the middle V'_j. In order to determine the grade of the reduction of these vertices, I use an adjustable threshold. This threshold can be configured via the user interface.

Each vertex V_{D_i} that corresponds to a remaining vertex V'_{D_i} of the previous steps is used for the construction of one explosion line. For that reason, the method simply connects the exploded vertex V_{D_i} with its original unexploded position V by a dotted line. If multiple vertices V_{D_i} are projected to the same point V'_{D_i} on the plane $Elp(D_i)$, the one that is nearest to the unexploded

position of the exploded parts P_{D_i} is taken. Figure 4.9 demonstrates the construction of the explosion lines.

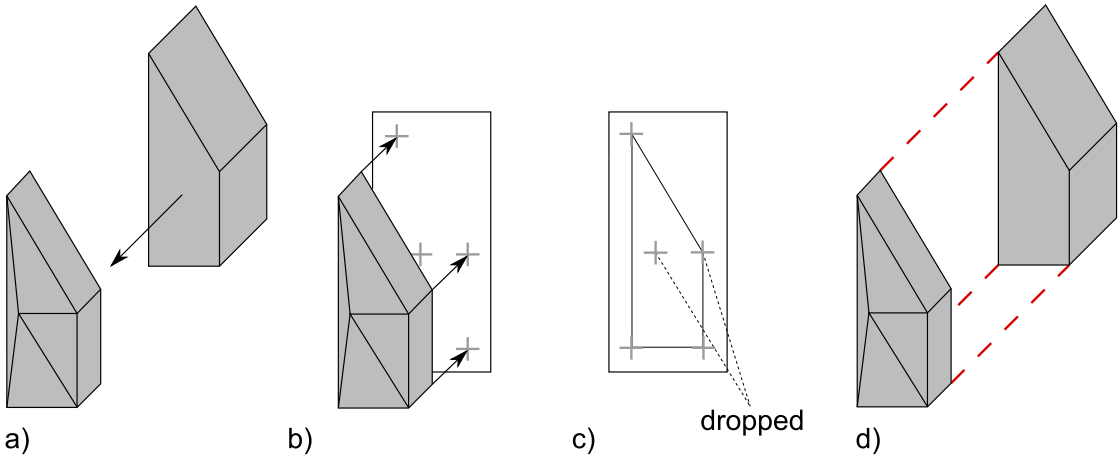


Figure 4.9: In a) one can see the exploded part P at the left-hand side and the remaining building at the right-hand side. In the next step b) the vertices V_{D_i} of P are projected onto a plane $Elp(D_i)$ that is orthogonal to the explosion direction D_P of P. c) demonstrates the elimination of two projected vertices V'_{D_i}. The first one is dropped because it is not part of the convex hull of the projected vertices V'_{D_i}, the second one because it is nearly aligned with its previous and following vertex while traversing the convex hull clockwise. d) displays the resulting explosion lines in red. The original remaining vertices V_{D_i} of the previous steps are connected with their unexploded positions V.

4.6.2 Plane-Edge Elimination

Many of the presented concepts make it necessary to eliminate plane edges. Plane edges are all those edges E where the left and right adjacent faces F_{left} and F_{right} are parallel. They are no real edges the viewer can see, but rather are used by the rendering system, which requires the geometry to consist of triangle surfaces.

However, these plane edges shall not be taken into account for the calculation of the outline, which is used for the construction of the outline frustum (see Section 3.4.1). Also, for the current selection visualization described in the user interface section (4.5), the plane edges appear to be distracting. Furthermore, the construction of the principal plane via the maximal edge direction has to ignore these plane edges (see Section 3.4.3).

In order to calculate the elimination of plane edges, I just use the above-presented definition of them. First of all, the face normal vectors N_F of all triangles $F$of the geometry of the building are computed. Neighboured triangles of the same building element are connected by an edge E. For every edge E, the two normals $N_{F_{left}}$ and $N_{F_{right}}$ of the two adjacent triangles F_{left} and F_{right} are compared by calculating their dot product. If the result is one, the normals and, therefore, the triangles are parallel to each other. In this case, the edge between the two triangles

is eliminated (see Formula 4.1). I use a very small margin α to tolerate inexact calculations and imprecisely modelled geometry data.

$$N_{F_{left}} \cdot N_{F_{right}} < \alpha \tag{4.1}$$

4.6.3 Outline Frustum and Explosion Offset

For the construction of the outline frustum $Of(P)$ of a part P, the outline edges of P are needed. The outline of a part P are the collection of all edges that form the boundaries of P projected onto a viewing plane relating to a particular camera configuration. I use the following method to find these outline edges: If the normal $N_{F_{left}}$ of the left face F_{left} of a particular edge E points away from the camera and the normal $N_{F_{right}}$ of the right face F_{right} points towards the camera, the considered edge E is an outline edge, else it is not. The outline edges of the part are applied to construct the outline frustum as described in the Section 3.4.1. The method uses the dot product of the camera view direction D_{camera} and the face normals N_F. Accordingly, if an edge E is an outline edge, it has to fulfil the Equation 4.2.

$$\text{sign}(N_{F_{left}} \cdot D_{camera}) \neq \text{sign}(N_{F_{right}} \cdot D_{camera}) \tag{4.2}$$

The outline frustum is used to calculate the explosion offset O. An exploding part P has be moved so far so that it escapes the outline frustum $Of(P_{sel})$ of the focused object P_{sel} as a whole (see global rule GR13). The minimal distance to do so into a given explosion direction D determines the explosion offset O. In order to calculate this distance, the features (i.e., vertices, edges and triangles) of the geometry of the exploding part P are swept into the explosion direction D. Then, the method intersects all the swept features with the triangles of the outline frustum $Of(P_{sel})$. The maximum of all distances between the original positions of features of P and their sweeping intersections with the outline frustum $Of(P_{sel})$ is the minimal distance to translate the part to escape the outline frustum. This is the minimum explosion offset a part P has to be translated into its explosion direction D to reveal the focused object P_{sel}.

However, the value is too minor to guarantee a proper part separation on the screen (see global rule GR14). There would be no empty screen space between the focused object and exploded parts. Thus I add an extra screen offset to the explosion offset. For this, the normalized explosion direction D is project into screen-space coordinates. Then, the projection is scaled so that its length corresponds to the desired screen offset (adjustable via the user interface). Afterwards, the scaled projected vector is re-projected back into model-space coordinates. Its resulting length is added to the minimum explosion offset. The result is the actual explosion offset O of P. These computations are necessary to ensure an equal part separation on screen into all screen directions regardless of the explosion directions of the exploded parts.

4.6.4 Lighting

The lighting of a scene is a big deal in visualization, because a proper lighting enhances the viewer's ability to distinguish and identify objects and details. However, in case of the ExVAr-system, the lighting system must have a runtime that allows a suitable frame-rate for an interac-

tive application. This circumstance is complicated by the fact that the used scene graph consist of many objects.

My first approach was to use a fixed, directional light. This works great for a particular viewing direction, but the back-faces relative to the light source are always insufficiently illuminated. As a next step, I tested ambient lightning. However, the use of an ambient lighting causes the edges of objects to disappear. I was encouraged to use a directional light mounted on the camera pointing into the view direction (I_{camera}). This works great for most of the cases. But two problems arise. The more inclined a viewing ray hits a face, the darker it gets. In fact it gets too dark. The second problem is as follows: at a particular view direction, edges disappear, because both connected faces are lighted the same way (see Figure 4.10). I came up with the use of additional multiple directional lights as one can see in Figure 4.11 (I_{+x}, I_{-x}, I_{+y}, I_{-y}, I_{+z}, and I_{-z}). By default, the ExVAr-system uses slightly different colors for these directional lights. They light the scene from all six axis-aligned directions. The method for the scene lighting can be found in Equation 4.3. The use of differently coloured lights emphasizes edges of the geometry without the need of additional lines. Additional lines may result in visual clutter. Furthermore, the applied method brightens up the above-mentioned dark faces.

In addition to the six axis-aligned directional lights and the camera-mounted light, the system uses a point light I_{point} to illuminate the focused room if it or any descendant is selected. This emphasizes the focused spatial structure and, therefore, helps to distinguish between the room and the rest of the building.

$$I = I_{camera} + I_{+x} + I_{-x} + I_{+y} + I_{-y} + I_{+z} + I_{-z} + I_{point} \tag{4.3}$$

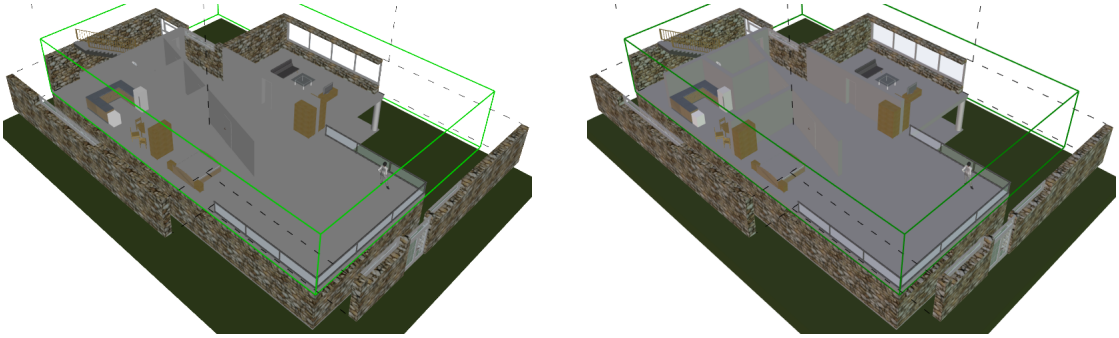


Figure 4.10: The figure demonstrates the advantage of the use of coloured directional light. At a particular viewing angle, some edges disappear as one can see in the left figure (Note: the ExVAr-system uses a camera-mounted directional light source to illuminate the scene). This can be avoided by adding extra, slightly differently coloured, directional light sources.

4.6.5 Others

Furthermore, the implementation for the sticking boundary relation (see global rule GR8) and the maximal edge direction (used for the determination of the principal plane), the concepts I

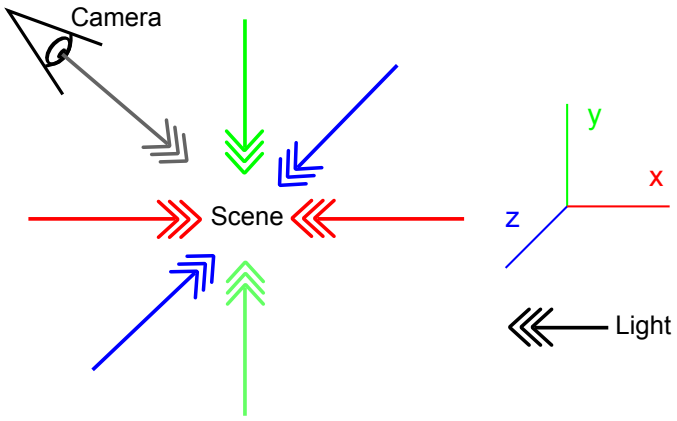


Figure 4.11: The figure displays the concept of differently coloured directional lights. Furthermore, the camera-mounted directional light is displayed.

have introduced and defined in the previous chapter, are mentionable at this point. I will not go into detail about them, because I already explained the implementations of these methods for the ExVAr-system in Section 3.5.1 and 3.4.3. However, it is conceivable also to exchange these implementations due to the modular architecture of the ExVAr-system.

4.7 Configuration of Parameters

In order to facilitate the adaptation and configuration of the ExVAr-system, I use an xml-file to encapsulate all configuration parameters of the system. This includes parameters for geometrical calculations, but also render, color, lighting, and camera parameters. Some of these parameters can also be controlled via the user interface to change the explosion view during runtime. The parameters are set to default values which result in a distinguishable visualization of the building, the explosion and the selection. I have fiddled with the parameters to gain a valuable visualization. This is evaluated via the user test (see Section 5.2 for a detailed description of the user evaluation). One can find the configuration parameters for the ExVAr-system in Table 4.1.

Table 4.1: Configuration Parameters

Name	Value	Range	Description
Geometrical Computation			
PositionRoundMarginForElements	4	0, 1, 2, ...	This parameter determines to which position after decimal point the vertices coordinates of an element has to be rounded. This is to merge imprecisely modelled vertices to connect neighboured faces of an element.
PlanarToleranceForElements	0.0001	0 - 1	The dot product is used to calculate whether two planes are parallel. The planar tolerance determines how great the angle between to planes can be so that they are still being considered to be parallel. If they are considered to be parallel and are neighboured, the edge between them is dropped and will not be used for the outline calculation.
BoundariesStickTogetherThreshold	0.25	0 - 1	The parameter specifies a relation threshold. If the relation of the diagonals of two adjacent boundaries is less than the threshold, the smaller one is stuck to the bigger one.
PrecisionForRoundingPositionsForExplosionLines	3	0, 1, 2, ...	This determines to what position after decimal point the coordinates of the vertices of an element has to be rounded for the calculation of the explosion lines. This is to merge imprecisely modelled vertices to facilitate the calculation of the convex hull of the projected vertices.

ToleranceForExplosion-LinesReduction	0.2	0 - 1	An explosion line is dropped if a projected vertex is almost aligned with its previous and its followed vertex. A line reduction value of 0 drops no vertices of the convex hull of the projected vertices. The dot product is used to determine the angle.
RemainingBuilding-ElementsExplosion-DirectionThreshold	0	-1 to +1	A remaining building element can only explode into a predefined direction if this direction points away from the selected element. If the parameter equals 1 there is no tolerance, 0 means 90 degrees tolerance and -1 results into no restriction of explosion directions.
DirectionRound-PrecisionForExplosionGraph	6	0, 1, 2, ...	The parameter determines to what position after decimal point the coordinates of the explosion direction of an element has to be rounded. This is to form sets of elements exploding into the same direction despite foregoing imprecise computations.
EdgeDropThreshold-ForFrustumExplosion-LengthCheck	0.01	0 - 1	This is used during the calculation of the explosion offset for the element relative to an outline frustum. It determines which edges shall not be used for the explosion sweeping of the edge of an element. An edge that is located almost parallel to the translation direction should be dropped because it does not enrich the result but can lead to an undefined output of the used calculations.

Graphic Display			
ExplosionScreenOffset	10	0, 1, 2, ...	The parameter represents the additional offset to the explosion offset to emphasize the separation of elements in screen space.
DrawSelection	true	true or false	This is a switch to turn on and off the visualization of the current selection. The visualisation is done by rendering the edges of the selected element green-rimmed.
DrawPossibleSelections	true	true or false	This is a switch to turn on and off the visualization of the possible selection. The visualisation is done by rendering the edges of the considered element yellow-rimmed. It is rendered if the mouse cursor hovers over a selectable element.
DrawExplosionLines	true	true or false	The parameter determines whether or not the explosion lines of exploded elements are rendered to visualize their explosion paths.
TransparancySorting	true	true or false	This determines whether or not a transparency sorting for rendering is perform.
ExplosionLinesDot-Distance	15	0 to $+\infty$	The spacing length of the dotted lines of the explosion lines in screen space is represented by this parameter.
ExplosionLinesDot-Length	10	0 to $+\infty$	The dot length of the dotted lines of the explosion lines in screen space is determined by this parameter.
ExplosionLines-Thickness	0.5	0 to $+\infty$	This defines the line thickness of the dotted lines of the explosion lines in screen space.

SelectionLineThickness	1	0 to $+\infty$	The parameter determines the line thickness of edges of the selection in screen space.
Light and Colour			
DrawRoomLight	true	true or false	This represents a switch to turn on and off the room spot light while the room is selected.
RoomSpotLightColor	[5, 5, 5, 1]	0 to $+\infty$	The parameter determines the color of the room spot light in RGBA.
SelectionColor	[0, 128, 0, 255]	0 - 255	This one determines the color of the selection edges in RGBA.
PossibleSelectionColor	[255, 255, 0, 255]	0 - 255	This defines the color of the possible selection edges in RGBA.
UseDirectionLight-Color	true	true or false	The parameter is a switch that determines whether or not different light colors for the directed light sources pointing into each orthogonal direction are used.
PrimaryDirection-LightColorValue	0.22	0 - 1	The primary light color component value to realize the different colors for different light directions is defined by it.
SecondaryDirection-LightColorValue	0.18	0 - 1	This parameter defines the secondary light color component value to realize the different colors for different light directions.
CameraLightValue	1.2	0 to $+\infty$	The intense of the directional light pointing into the viewing direction of the camera is determined by this one.
Others			
CameraStartPosition	[15, 15, 25]	$-\infty$ to $+\infty$	The start position for the camera is defined by this parameter.

CameraFocus	true	true or false	A switch to turn on/off the rotation of the camera towards the selected element during the selection is represented by this one.
ExplosionAnimation-Duration	2000	0, 1, 2, ...	This parameter specifies the duration of the explosion animation.
MouseScrollUpdate-Delay	500	0, 1, 2, ...	The parameter defines the delay to perform the explosion update and animation after the mouse scroll wheel was used. If during this delay the mouse wheel is used again the previous update is cancelled and just the new update is performed.
DefaultIfcFile	House.ifc	path string	This represents the default model file path.
IfcModelScale	0.001	0 to $+\infty$	This is a scale factor to fit the IFC model into the render scene.
OBJModelScale	1	0 to $+\infty$	The parameter is scale factor to fit the OBJ model into the render scene.

CHAPTER 5

Results

5.1 Visual Output

In this section I will demonstrate the visual output of the ExVAr-System and discuss aspects and problems of the resulting images. One has to take into account that these static images are just screenshots of an interactive real-time application and so can hardly reflect the actual user experience. Furthermore, I had to adapt some configurations so that the resulting images are more appropriate for static figures. For example, I disabled the automatic camera rotation during the selection process. That way, the unexploded building parts are located at the same position in the figure at the different stages of the explosion view. I also disabled the selection and mouse-over visualization for some screenshots to reduce visual clutter. All these adjustments are possible via the user interface.

5.1.1 Step-wise Explosion

In the following, I will go through the explosion view handling the different stages of the explosion step by step. I start by selecting the topmost spatial structure and proceed by selecting one of its child spatial structures and so on. I will apply this to the model $House$ and the model $Institute$.

In the first step (Figure 5.1 and 5.5), the building is rendered unexploded as a whole. This status of the exploded view is produced when nothing is selected or the complete building site is selected. When the user selects the building, then the façade and the roof of the building is translated to remove the occluding coverings from the building and to reveal the interior (see Figure 5.2 and 5.6). The user can consequently see the floors of the building. After selecting a particular one, all other occluding floors are translated (Figure 5.3 and 5.7). The façade is split and the parts related to the translated floors stick to them while exploding. This enforces the compactness of the exploded view and clarifies the context of the building elements and, therefore, the relations among the elements. In the last step, a room is selected (Figure 5.4 and

5.8). All other rooms of the same floor are translated. In order to reveal the selected one, they stick to the top slab representing their major boundary.

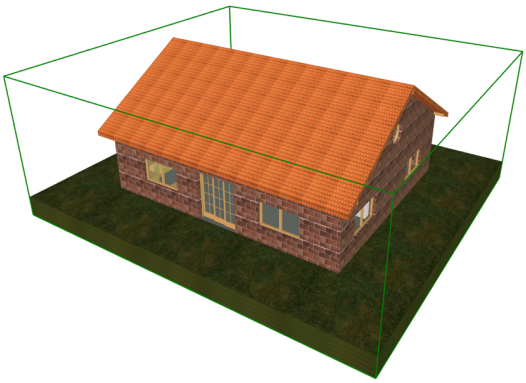


Figure 5.1: This figure shows the building model $House$ as a whole. The complete building site is selected.

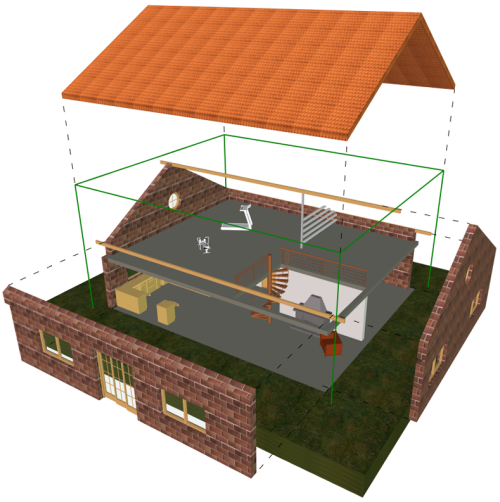


Figure 5.2: When the user selects the building, then its covering boundaries, i.e., the façade and the roof, are exploded to reveal the interior.

5.1.2 Side View

The side view is a common visualization tool in architecture and for displaying technical assemblies. In the ExVAr-system, only a perspective projection and, therefore, only a perspective side view is available. The camera view cube of the main view of the ExVAr-system enables the user to select axis-aligned camera view directions. In this way, a side view from the left, right, back or front can be easily chosen. The underlying algorithm of the system implicitly generates appropriate output images. In the following, one can find examples of generated side views (Figure 5.9 and 5.10) and also a step-by-step explosion of a side view of the model Own (Figure 5.11 to 5.14). In the examples, the covering walls are translated to a position behind the camera because the explosion directions of the walls are parallel to the camera view direction, but reversed. Otherwise, the covering walls could not reveal the interior related to their explosion direction. The horizontal boundaries of the selected spatial structures are not exploded because they do not occlude it. Therefore, the floors are not separated by selecting a floor of interest, as one can see in Figure 5.13. On the other hand, rooms of the same floor may be exploded by selecting a room of interest because they follow their exploded wall boundaries (see Figure 5.14).

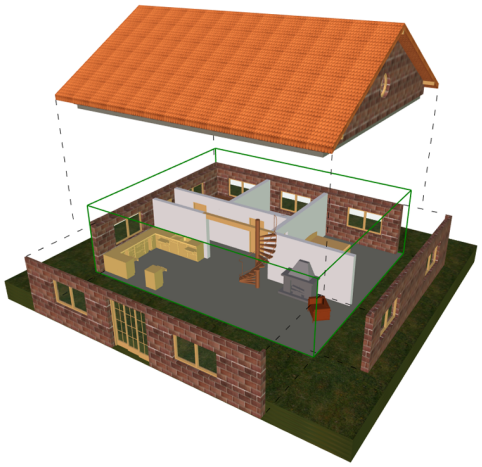


Figure 5.3: In the next step, the ground floor is selected. This results in translating the top floor up. The façade is split up, to enforce compactness and clarify the context of the selection.

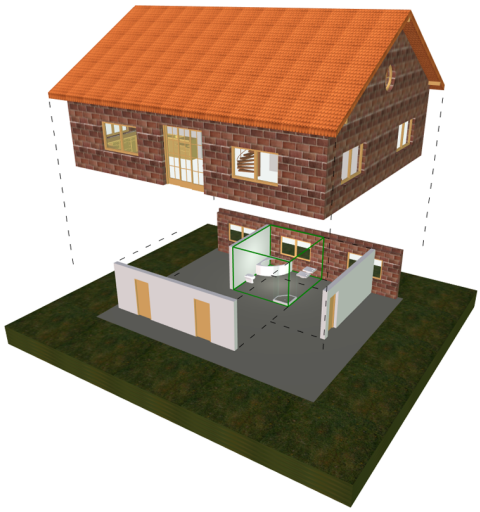


Figure 5.4: Here, one can see what happens if a room is selected. The bathroom is focused and so all other rooms follow the translation of their major boundary, the covering slab, to expose its interior.

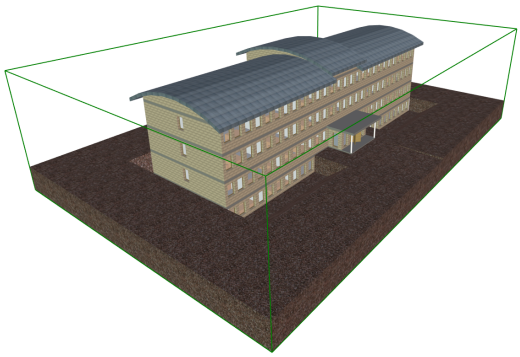


Figure 5.5: This figure shows the model *Institute* as a whole.

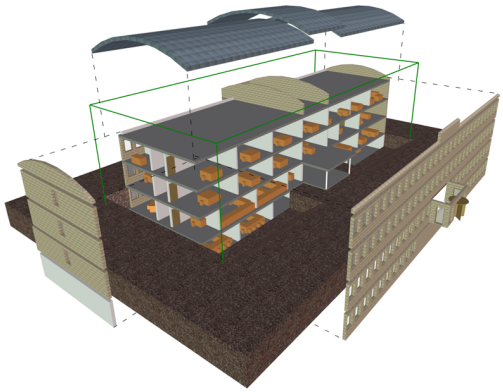


Figure 5.6: The façade and the roof is exploded by selecting the building.

5.1.3 Perspective Floor Plan

The floor plan of a building is a very well-established visualization technique in architecture. A floor plan clarifies the location and size of rooms and the connections and relations between them. Furthermore, it illustrates the arrangement of furniture and other building elements in the horizontal dimension. The ExVAr-system for the most part implicitly implements these requirements by the top explosion view of a floor or a room. The camera has to be positioned

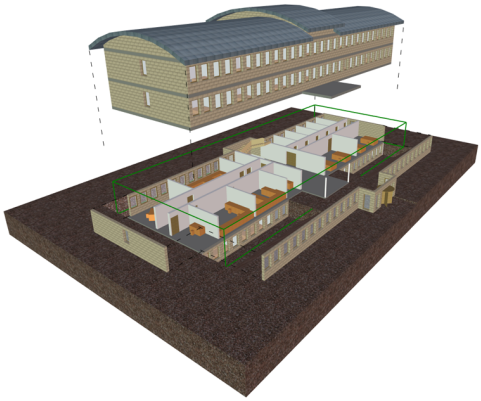


Figure 5.7: The other floors are removed by selecting the floor of interest.

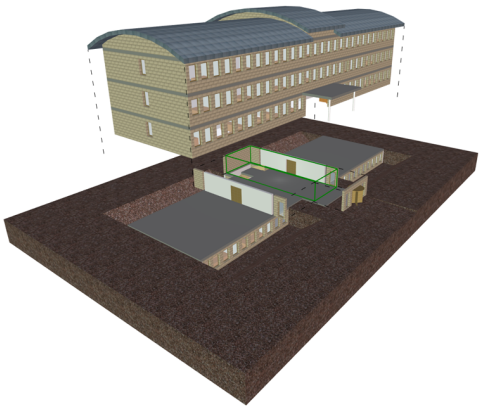


Figure 5.8: Selecting a room reveals its inside.

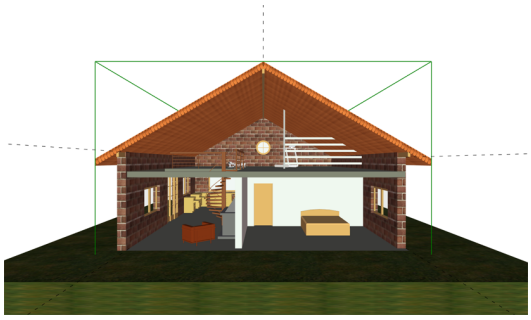


Figure 5.9: The figure shows a side view of the model *House*. The top floor, the bedroom, the living room and even the kitchen can be seen.

Figure 5.10: Side views can reveal the relation of rooms on different floors. Here, one can see the staircase of the model *Institute*.

above the focused spatial structure at the desired distance pointing down onto it. The camera view cube of the main view of the ExVAr-system facilitates an exactly vertically aligned view direction. All floors atop the focused one are translated to a position above the camera and, therefore, are not visible. On the one hand, the perspective view slightly distorts the geometry, but on the other hand, it grants additional insight into the focused spatial structure. Not only the vertical arrangement of building elements becomes perceivable, but also most of the doors and windows are visible without adding an additional overlay visualization or cuts. This works good for small floors or rooms as one can see in Figure 5.15. But the larger the horizontal dimension of the considered spatial structure is, the more similar the view becomes to an orthographic projection (see Figure 5.16). In a top view with an orthographic projection, doors and windows would be occluded by their containing walls.

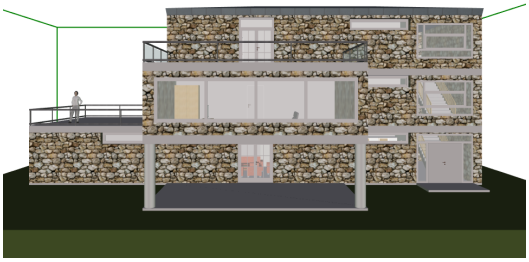


Figure 5.11: This and the following three figures demonstrate the step-by-step explosion of a side view of the model Own. In the first figure, one can see the building as a whole.

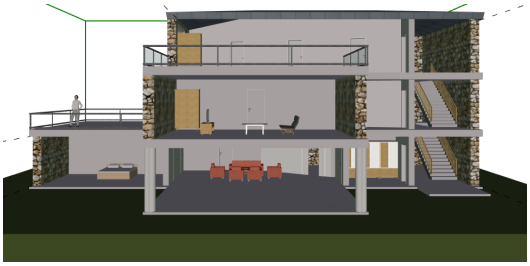


Figure 5.12: In the second step, the façade is translated to a position behind the camera after selecting the building.

Figure 5.13: By selecting the second floor, the façade of the first and third floor returns to them. As one can see, fewer parts are exploded compared to Figure 5.12, the previous selection step. This is because the floors are not separated from the selected one since they do not occlude it.

Figure 5.14: When a room is selected in the side view, all other rooms of the floor may be exploded and, therefore, be translated to a position behind the camera. They just follow their exploded major boundary walls.

5.1.4 Special Cases

The underlying algorithm of the ExVAr-system is based on general approaches. Therefore, it can be used for different configurations of the camera and various arrangements of model data unless they fit the requirements. In the following I address two special cases which are mentionable.

The first one is the bottom view as one can see it in Figure 5.17 and 5.18. When the user chooses a camera view direction that points upwards, i.e., she/he looks at the architecture from the bottom side, then the occluding parts tend to be translated downwards. In this configuration, the occluding slab is not the ceiling of a room or floor, but it is the floor. So the lower floors follow the occluding floor to reveal the interior of the selected floor.

The second kind of special case I want to mention are not axis-aligned explosion directions. It may occur that explosion directions of walls are not parallel to any coordinate axis (see

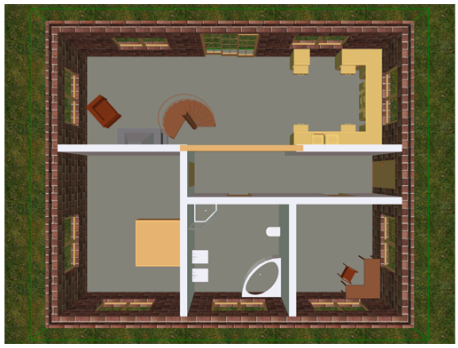


Figure 5.15: This figure demonstrates the perspective floor plan of the model $House$ generated by the ExVAr-system. One can identify the doors and windows of the floor and can perceive the arrangement of rooms and furniture.

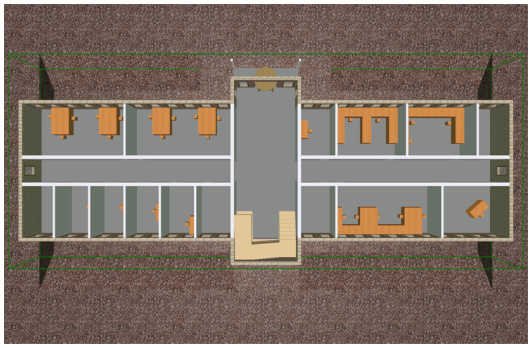


Figure 5.16: The larger the scale of the horizontal dimension of a considered spatial structure is, the poorer the visualization of doors and windows becomes.

Figure 5.19). This is a result of the considered wall geometry which is placed oblique and its related principal plane. The ExVAr-system implicitly handles such cases and generates oblique explosion directions as one can see in Figure 5.20.

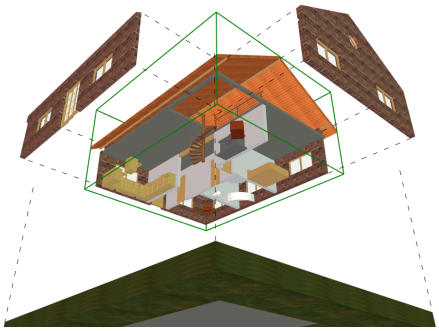


Figure 5.17: This figure shows the bottom view of the model $House$.

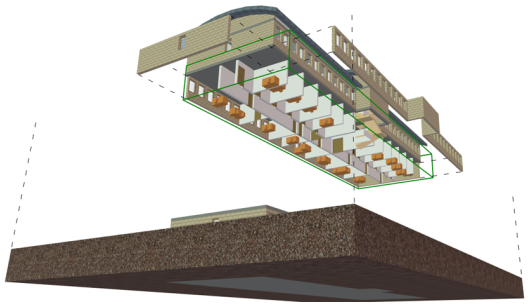


Figure 5.18: Here one can see the bottom view of the model $Institute$. The second floor is selected and, therefore, its occluding floor is translated downwards.

5.1.5 Problems

The screenshots of the ExVAr-system reveal problems of some used concepts, data, and implementations of the underlying algorithm. Most problems arise due to the fact that not all building data is modelled cleanly. In such a case, it may occur that building data does not exactly fit the hierarchical model defined in Section 3.3. For example, if a multilayer building element like a slab or a wall is modelled as multiple building elements, the algorithm may fail to generate

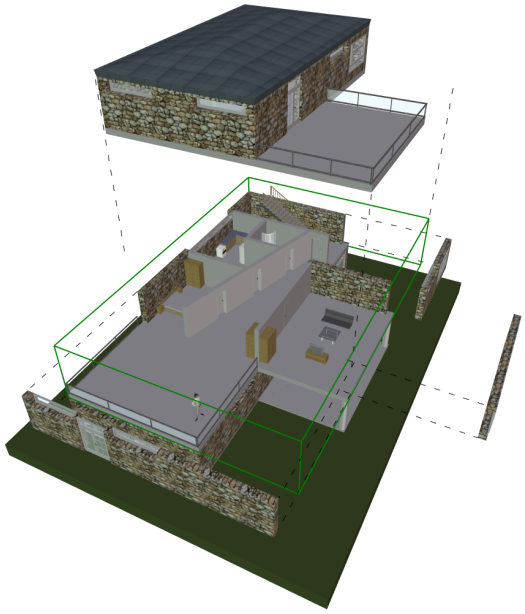


Figure 5.19: This shows the explosion view of the model Own. It contains an oblique wall.

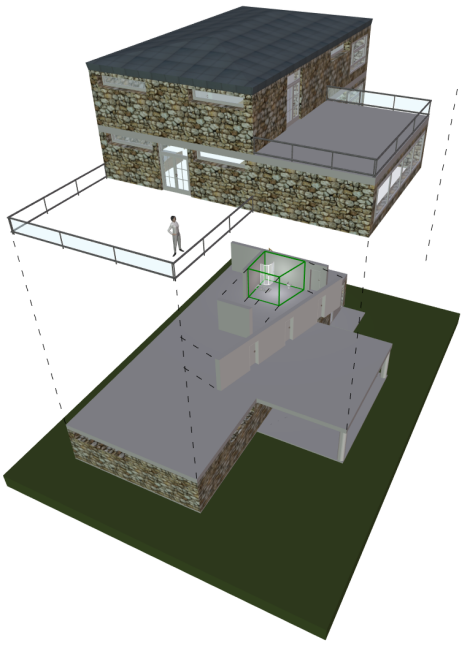


Figure 5.20: When a spatial structure is selected that is occluded by an oblique wall, the explosion direction of the wall is oblique too.

a correct exploded view according to the presented concepts. In Figure 5.22, this problem is demonstrated. The grey slab, which occludes parts of the right room, belongs to the second floor. It represents the insulation of the overhanging floor, but is modelled as a separate building element. Therefore, it is not in contact with any spatial structure. Thus the algorithm fails and the insulation is not exploded at all although it occludes the selected floor because it is not even processed.

Another problem arises from a simplified solution approach of the sticking boundary relation as described in Section 3.5.1. In Figure 5.21, one can see that the façade becomes very fragmented while exploding. This may reduce the compactness of the explosion view and makes it more difficult to mentally reconstruct the façade. This occurs because the difference in size of the considered walls is too minor. Consequently, the used sticking boundary relation implementation fails and the façade is split up.

Another simplified approach causes the generation of wrong results. The back-face sticking procedure may fail for concave spatial structures. In this case, a boundary may occlude a spatial structure although it is a back-face surface. Consequently, it is not exploded and remains occluding the spatial structure. One can see a detailed illustration of the problem in Figure 5.23.

For some types of spatial structures, the bounding box is used to generate the explosion lines unless there is no other geometrical information available (e.g., the floor of a building). This is done for performance purpose. Unfortunately, this may result in explosion lines that cannot be

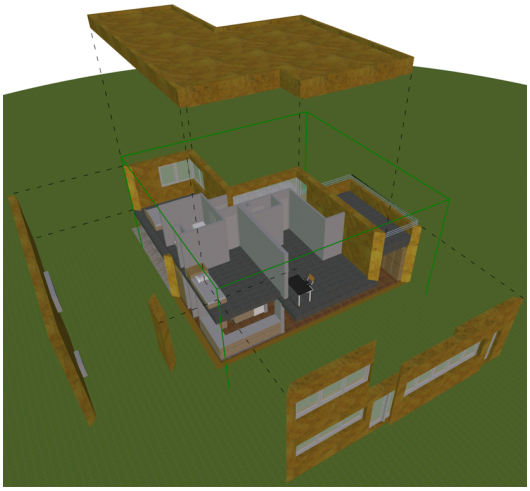


Figure 5.21: In this example, the façade becomes fragmented due to the current implementation of the sticking boundary relation. The result is difficult to mentally reconstruct.

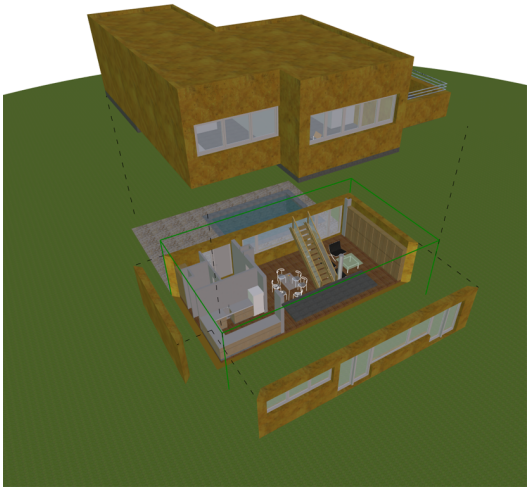


Figure 5.22: This figure demonstrates the problem resulting from inappropriately modelled data. The grey slab is not in contact with any spatial structure and, therefore, is not exploded.

easily related to any real geometry by the viewer. Figure 5.19 demonstrates an example of such an explosion line. The explosion line at the right-hand side of the figure belongs to the top floor of the building.

5.2 User Evaluation

This section describes the user evaluation of the ExVAr-system. I will handle this by starting with a theoretical foundation and proceed to a description of the performed user test itself and important aspects of it. Afterwards I will analyse the performance of users who conducted the test and extract problems and possible improvements of the ExVAr-system.

5.2.1 Methodological Approach

Usability engineering and user evaluation are already well-investigated domains of software development. There are many different methods for usability engineering, but their domain can basically be divided into inspection methods and test methods. Andreas Holzinger concerns himself with different representatives of these two approaches [Hol05]. Inspection methods do not involve test users but usability specialists who check the software interface design against established standards. They have the advantage that the considered interface can be analysed even if it is unfinished and there is no stable prototype [NP93].

On the other hand, test methods must involve end users.

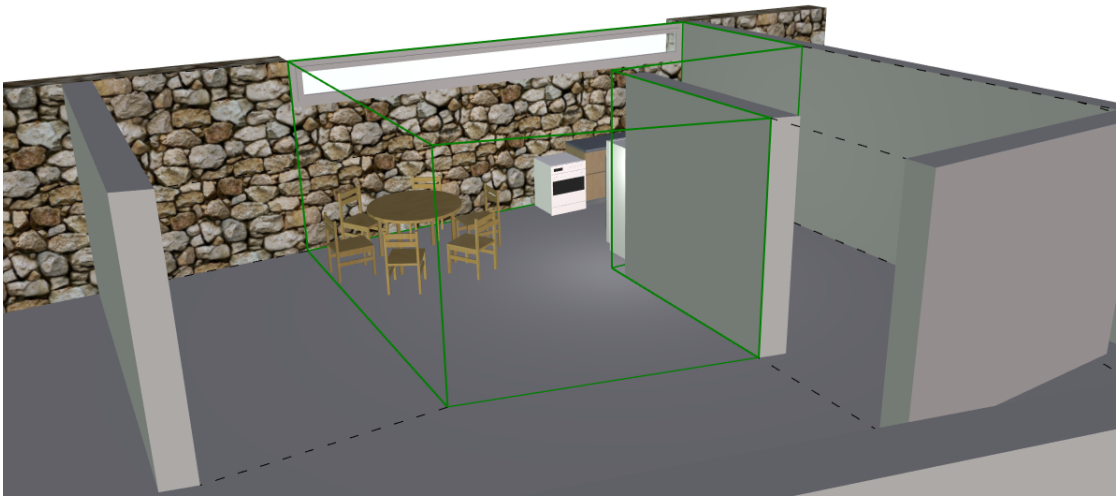


Figure 5.23: The unsophisticated procedure of the back-face sticking causes undesired occlusions.

> Testing with end users is the most fundamental usability method and is in some sense indispensable [Hol05].

Joseph S. Dumas and Janice C. Redish wrote the book "A Practical Guide to Usability Testing" [DR99] about this topic. They assert that the primary goal of usability testing is the improvement of the usability of a product. The participants of the testing represent real users and they have to do real tasks. Their performance is observed and recorded for a latter analysis of the data. As a result, the usability tester can diagnose real problems of the tested software and recommend changes to fix those. Furthermore, the authors also state that

> A usability test includes both, times when participants are doing tasks and times when they are filling out questionnaires about the product [DR99].

There are five generally accepted essential usability characteristics: learnability, efficiency, memorability, low error rate and satisfaction. The importance of these characteristics vary relating to the considered software and its field of application. For example, if end users are expected to use a software only a few times, learnability is very important. But if only expert users will work with it, efficiency becomes much more important [NP93]. There are several methods that can be used for usability testing. The most common ones are thinking aloud, field observation and questionnaires. Thinking aloud may be the most valuable one. It works as follows: the user interacts with the system and permanently verbalizes her/his thoughts related to the interaction. This way, the evaluator directly gets information about what parts of the system cause problems and why users do what they do. This is a direct method and, therefore, does not rely on the user's memory. Nevertheless, it is important to supplement direct with indirect usability tests, for example questionnaires. These are to collect the opinion of the user about the interface but also to understand her/his cultural background and her/his capabilities [Hol05].

5.2.2 Test Set-up

In order to conduct a valuable user evaluation, I created a test set-up according to the cited literature of the previous section. The ExVAr-system is designed for future or current occupants or users of the architecture. Therefore, the expected end user of the system may not use the software regularly. In case of the ExVAr-system, the learnability of the interface is a very important characteristic for good usability. I apply a direct test method supplemented by an indirect one. The test consists of an introduction, seven tasks the user has to perform, and a questionnaire consisting of six questions. The test can be found in Figure 5.28 and 5.29. The test user is asked to speak out loud her/his thoughts (thinking aloud method) and verbally comment each step while handling the tasks. I film the user during the test via a webcam and also record the screen of the computer. Furthermore, I function as a test supervisor who sits next to the user to lend her/him a hand if she/he insistently asks for it or gets hopelessly stuck. These interventions are also recorded and counted. The models $Institute$, $Tofu$ and $House$ are used for the test. It is worth mentioning that the camera location does not change after loading a new model, so it depends on the previously adjusted location.

The different tasks of the test concern different aspects of the interface and the interaction with it. Therefore, I designed potentially real tasks covering most of these aspects. The first two tasks are kind of a tutorial of the basic handling of the system. The test user has to find and select rooms via the structural tree of the hierarchical view and via the main view of the interface and answer questions of detail about them. She/he shall get an understanding of the hierarchical model of the building and learn the reaction of the system during the selection process. Afterwards, the test user shall be able to use both navigation methods on her/his own.

For the third task, the test user has to examine two offices and has to decide in which one she/he would prefer to work in. For this, it is necessary to use one of the two navigation methods and, additionally, to apply the camera navigation to get a suitable view. It shall be tested how the user learns to use the camera navigation and if the visualization offers enough information to answer the asked question. This addresses the feeling of space communicated by the system.

The fourth task handles the perception of the spatial relation of building elements. The test user has to answer if a particular shelf occludes a window (Figure 5.24). Now she/he has to adopt the learned camera navigation to adjust the angle and zoom of the view to reveal this information.

The fifth task shall investigate how well path finding works in the ExVar-system. The test user has to find the path from one room to another and enumerate all passed rooms as one can see in Figure 5.25.

In the sixth task it shall be tested if the user can estimate quantitative attributes of building elements. Therefore, she/he has to compare the size of two rooms.

The last task implies most of the aspects of the previous tasks. The test user shall find two different building elements and shall examine if there is a line of sight between them. The two building elements are located in different rooms and on different floors.

After the test user has performed all tasks, she/he has to answer a questionnaire (see Figure 5.29 for details). The user is asked to asses her/his computer skills and her/his expertise of

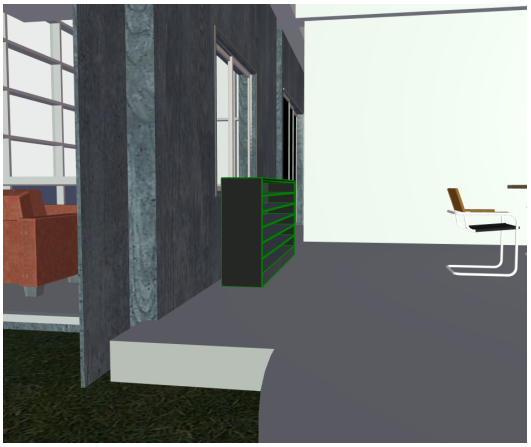


Figure 5.24: This figure shows a possible solution of a camera and selection adjustment for the fourth task of the user test. The shelf partly occludes the window.

Figure 5.25: Here one can see the yellow-rimmed bedroom and the kitchen. The test user has to assess the size of the rooms and name the other rooms passed on a route from the bedroom to the kitchen.

architecture. This is to get an idea of her/his capabilities concerning the main domains of the ExVAr-system. Then the test user has to disclose how difficult it was to solve the previous tasks and what problems occurred. Of course, this may be redundant to the observation and analyses of the user performance during the tasks, but may reveal additional findings because the test user can rethink the task situation. Furthermore, the user is asked to tell which navigation method she/he prefers for which actions. This is to optimize the interactions for particular situational requirements. At the end, the test user shall give feedback about how to improve the system. That way, new ideas may be generated and user wishes can be taken into account.

5.2.3 Analysis

In this section, I will quantitatively evaluate the performance of the test users by analysing the recorded videos. For this, I measure the amount of actions taken by the users, the time needed to solve the task, and the unavoidable interventions by the test supervisor. Furthermore, I note the problems that occur. I will also quantitatively evaluate the questionnaire and investigate dependencies of the user background to the user performance. I will apply the Pearson product-moment correlation coefficient to find correlations among the measured data. The coefficient is used as a base for the interpretation of the analysis. The advantage of this correlation coefficient is that it is normalized. A coefficient of 0 means no linear correlation, 1 is a total positive linear correlation, and -1 is a total negative linear correlation between two sets.

Test Users. I acquired five persons who have never used the ExVAr-system before. Since learnability is the main characteristic, this is an important requirement for the test users. The abilities of these test users widely diverge. Their self-assessments of the expertise of architecture

Test User	Computer Skills	Expertise of Architecture	Difficulty
test user 1	4	2	5
test user 2	5	2	4
test user 3	4	4	3
test user 4	3	1	3
test user 5	5	4	4

Table 5.1: The table represents the self-assessment of the test users. One can also find the difficulty rating of the users for the tasks here.

Test User	Task Time in s	Actions	Interventions
test user 1	376	88	0
test user 2	423	97	1
test user 3	732	91	2
test user 4	1167	67	7
test user 5	636	139	1

Table 5.2: The attributes of the performance of the test users are listed in this table. The values are the result of a count of the recordings.

range from $very\ bad$ to $rather\ good$ an a 5-item rating scale ($very\ bad$, $rather\ bad$, $moderate$, $rather\ good$, $very\ good$). The self-assessment of the computer-skills, however, reach from $moderate$ to $very\ good$. One can find a detailed enumeration of the answers of the first three questions of the questionnaire in Table 5.1. The ordinal values of the answers are mapped to numerical values ranging from 1 to 5 for later calculation purpose where 1 represents $very\ bad$ (respectively $very\ hard$ for the difficulty assessment). The analysis yields that there exists a moderate positive linear correlation between the computer skills of the test users and their assessment for the difficulty of the task (correlation coefficient of 0.428). This signifies that the better the computer skills, the easier the test users found it to solve the tasks. On the other hand, there is no significant correlation between the self-assessment of the expertise of architecture of the user and the perceived difficulty of the tasks. The correlation coefficient is -0.089. Accordingly, the computer skills have a much greater impact on the assessed difficulty then the expertise of architecture.

Task Time. The task time is the total time a user needs to solve all tasks. This excludes the time a test user reads the instruction of a task. The recorded task times of the test users greatly vary. The fastest one solved all tasks in 6 minutes and 16 seconds. The slowest one needed 19 minutes and 27 seconds. This is more than three times as much. In order to look into the reason for that, I investigated the correlation between the user background and the task time. Again, there is hardly any linear correlation between the expertise of architecture and the task time (correlation coefficient of $-0,254$). However, one can recognize a significant negative linear correlation between the task time and computer skills (coefficient of $-0,732$). In other words, the better the computer skills are, the less time the users needed to solve the tasks. A more

detailed overview can be found in Table 5.2.

Actions. Actions sum up all important and countable activities performed by the user to solve a task. One can find right-click, left-click, camera rotation and camera zoom among them. I counted them for all users and all tasks in the recordings. The amount of actions by far does not vary as much as the task time (see Table 5.2). Moreover, the correlation to the computer skills is a positive one (coefficient is $0,822$). This means that better computer skills result in more actions taken. I interpret this as follows: an experienced computer user feels comfortable in a new environment and fiddles about the functionality of a system by a try and error approach. A less experienced user longer considers what she/he is going to do next and how. This also reflects the increased task time. But the less experienced user is also less engaged to actually interact with the system. Additionally, the analysis shows that the expertise of architecture has a significant impact on the amount of actions. A correlation coefficient of $0,733$ indicates that a greater expertise in architecture results in more actions taken. My interpretation of this value is that users with greater expertise also are more interested in the viewed buildings and are more advanced in operating on building models. Therefore, they perform more actions.

Interventions. In addition to the parameters above, I count all interventions of the test supervisor. The supervisor only intervenes if the test user insistently asks for help or gets hopelessly stuck. Only one test user could solve all seven tasks without any intervention (see Table 5.2 for details). Two test users needed one and one needed two interventions. Test user 4 needed seven interventions. This mostly relates to the computer skill level. A correlation coefficient of $-0,775$ between the computer skills and the amount of interventions indicates this. There is also a moderate negative linear correlation between the expertise of architecture and the amount of interventions of $-0,510$. It is also remarkable that there were no interventions needed for the first task concerning the navigation via the structural tree of the hierarchical view. This may be because of the conventional nature of this navigation method. The problems that made it necessary to intervene mostly concern the camera interactions and the upwards navigation in the main view (i.e., selecting the parent of the current selection via a right-click). Therefore, only two out of five test users were able to solve the second task without any intervention because the upwards navigation is absolutely necessary for this task.

5.2.4 Qualitative Results

In the following, I will investigate the problems that arose during the user tests in a more detailed way. Furthermore, I will discuss the suggestions for improvement for the ExVAr-system claimed by the test users. As mentioned before, the upwards navigation in the main view causes serious problems resulting in interventions. Especially in the second task, the upwards navigation is necessary because the test user shall navigate from a selected room to another room located on another floor. Therefore, the user must navigate to a different branch of the hierarchy and, consequently, use the upwards navigation to select an ancestor of the target room (as shown in Figure 5.26 a)). The primary idea of this selection restriction to adjacent hierarchy relations was to support the perception of the hierarchical structure of the model by the user via the main view

navigation. Unfortunately, the users wanted to directly select the target room and, therefore, got stuck unless the attempt failed. As a result of this finding, I will weaken this selection restriction and introduce the cross-hierarchy selection in Section 5.2.5.

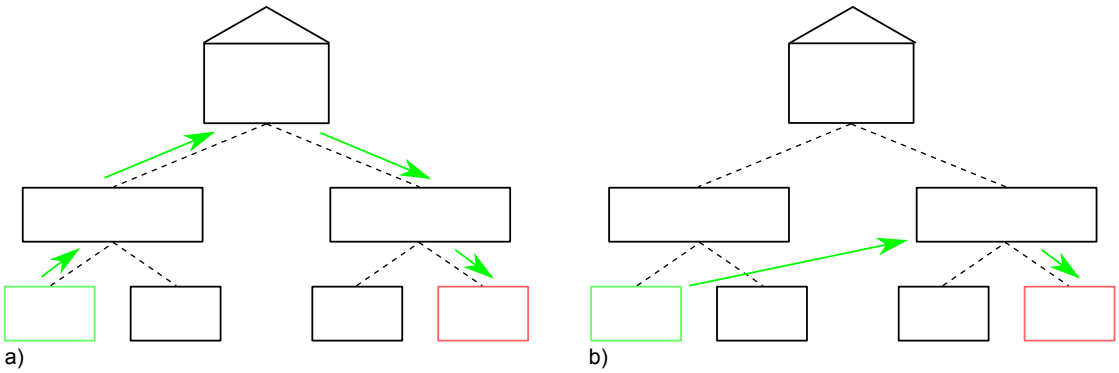


Figure 5.26: Figure a) demonstrates the necessary navigation steps without the cross-hierarchy selection. Two upwards navigation steps and two downward navigation steps are needed in this example. By using the cross-hierarchy selection, the amount of steps is reduced to two as shown in Figure b).

Also, the camera navigation caused different difficulties. First of all, test users who were not familiar with camera navigation needed exercise to master the interaction with the camera. On the other hand, advanced computer users were irritated by the fact that the button for the camera navigation is the middle mouse button, and not, as often used in other 3D-programs, the right mouse button. However, both classes of users got used to it and overcame their initial difficulties.

A more serious problem is caused by the delay of the explosion adaptation after a camera rotation. The explosion adaptation just takes place after the middle mouse button is released and the user has finished the camera rotation interaction. The user may set the position of the camera to a location where there is no line of sight to the selected building element until she/he releases the middle mouse button and the explosion adapts to the new camera position. Till the user releases the mouse button, the line of sight is temporarily blocked by another building element like a boundary. Users who are new to the system avoid these camera positions. Therefore, they unnecessarily restrain the possible camera positions. In order to reduce this occlusion problem, transparency could be integrated at that point. As an alternative to transparency, the explosion adaptation could take place not after but during the camera rotation. However, both approaches exceed the scope of this work.

Another problem is caused by an inadequate information visualization. The name of a building element is rendered at the top left corner of the main view when the mouse cursor hovers over the element. During the user tests, it became apparent that this is not a good location to do so. Some users were not able to detect the notification of the element name on their own.

The questionnaire revealed many interesting suggestions for improvement of the ExVAr-system. Nearly all test users proposed to include a tool to measure the room size. This may be

a result of the sixth task, but is a valuable input. Some advanced users asked for conventional cursor icons and conventional camera navigation support. They also suggested to additionally include a walk-through camera to overcome the limits of the implemented arc ball camera. One test user proposed the insertion of a textual search for building element names. Another valuable suggestion was to visualize the cardinal directions and the sun position. Both factors, the cardinal directions and the sun position, have great impact on the design of a building concerning weather resistance and sunlight integration. All these suggestions of improvement may enhance the ExVAr-system. However, some may interfere with the exploded-view approach or other aspects of the ExVAr-system. For example, the walk-through camera contradicts the looking-inside approach of an exploded view.

5.2.5 ExVAr Changes

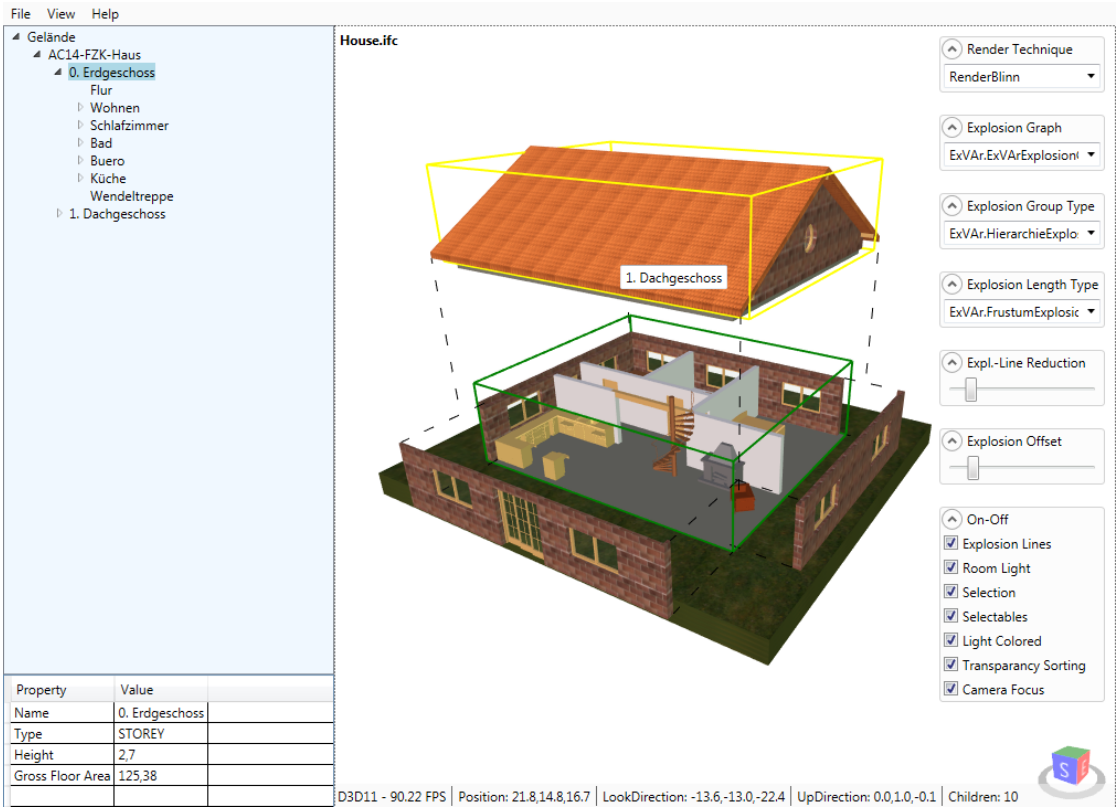


Figure 5.27: This figure displays the interface after the integration of the improvements resulting from the user evaluation.

In this section, I will enumerate the improvements implemented to the ExVAr-system resulting from the user evaluation. First of all, the cross-hierarchy selection is a great benefit to the system (see Figure 5.26). I weaken the restriction that only the parent or a direct child of the current selection can be selected via the main view. In the new version of the ExVAr-system,

all direct children of any ancestor of the current selection can be selected unless the child is no ancestor or descendant of the current selection. Although this increases the amount of selection candidates and, therefore, the complexity of the selection, it is reasonable after all. It is not only more intuitive (see user test), it also reduces the amount of necessary selection steps in several cases.

Furthermore, I also changed some information visualization parts of the system. The cardinal directions were integrated into the view cube of the main view. In addition, I added a detail view to the interface that displays detailed information about the current selection like the name, the type and the size of a building element. This can also display the demanded room size. Additionally, I transformed the display name of a building element where the mouse cursor hovers over to a tool-tip-wise display (see Figure 5.27).

Of course, this newly added and modified features should be user evaluated again, but this exceeds the scope of this work.

Evaluation of the Usability of the ExVAr-System

Introduction:

This user test conduces the evaluation of the usability of the ExVAr-system. I ask you to conduct the following tasks using the ExVAr-system. Therefore, you shall apply the "Thinking Aloud"-method, i.e. vocalize your thoughts and verbally comment each step. Furthermore, I want you to answer the questions of the attachment afterwards.

Quick Guide of the ExVAr-System:

The ExVAr-system serves the visualization of buildings using explosion views and provides tools to the user to empower him to explore architectural buildings and get an idea of it. Therefore, the user-interface is split into two sections: on the left-hand side you can find the structural tree that represents the membership of building elements. On the right-hand side there is the main view where the actual geometry of the architecture is rendered. In both sections a navigation through the building is possible by left-clicking the desired building element to select it. As a result the selected element is exposed.

The main view not just renders the building, but I also enables the user to interact with the architecture via camera navigation and selection of elements.

When the user hovers the mouse cursor over a selectable building element in the main view, this element becomes yellow-rimmed and its name is displayed at he top left corner. It is actually selected by left-clicking it. The current selection is rendered green-rimmed. In the main view it is only possible to select elements that are contained in the current selection related to the hierarchy of the structural tree (e.g. the rooms of a selected storey). In addition, the user can select the parent element of the current selection by right-clicking (e.g. the storey of the currently selected room).

The camera can be rotated around the selected building element. Therefore, the user has to drag the mouse into the desired direction while pressing the middle mouse button. The exploded parts of the explosion view rearrange not before the middle mouse button is released. Furthermore, it is possible to zoom in and zoom out by scrolling the mouse wheel. A view cube placed at the bottom right corner of the main view facilitate to set up axis-aligned camera viewing directions by left-clicking on the related side of the cube.

1/2

Figure 5.28: This is the introduction sheet of the user test.

Tasks:

1. Find the room *Buero Obermueller* and select it via the structural tree on the left-hand side. How many chairs are placed inside the room? (Model *Institut*)
2. Select the rightmost room of the cellar (*Kellers)*, as seen from the main entrance, via the navigation of the main view. What is the name of the room? (Model *Institut*)
3. Consider the two *Office* rooms in the second floor (*2F*). Which one do you prefer to work in and why? (Model *Tofu*)
4. Navigate to the *Meeting Room* on the ground floor. A shelf is placed inside the room. Does the shelf cover the window? (Model *Tofu*)
5. Which rooms have to be passed to get from the bedroom (*Schlafzimmer*) to the kitchen (*Küche*)? (Model *House*)
6. Which one is larger, the bedroom (*Schlafzimmer*) or the kitchen (*Küche*)? (Model *House*)
7. Are you able to see a person sitting in the chair (*Sessel*) on the ground floor (*Erdgeschoß*) when you train on the treadmill (*Laufband*) on the top floor (*Dachgeschoß*)? (Model *House*)

Questions:

1. How do you asses your computer skills? (very bad – rather bad – moderate – rather good – very good)
2. How do you asses your expertise of architecture? (very bad – rather bad – moderate – rather good – very good)
3. How do you asses the difficulty of solving the previous tasks? (very hard – rather hard – moderate – rather easy – very easy)
4. Which tasks do you think to be the hardest and why?
5. Which navigation method do you prefer for which actions, structural tree or main view?
6. How can the ExVAr-system be improved?

2/2

Figure 5.29: Here one can see the tasks and questions sheet of the user test.

CHAPTER 6

Conclusion

6.1 Summary

This thesis described a system that automatically generates exploded views and offers interactive exploration mechanics in the domain of architecture with respect to the context of elements and the levels of detail of a building. In order to realize the implementation, I searched the literature to find suitable ideas and references for exploded views in architecture. However, all considered approaches of exploded views are limited to densely built assemblies. As a result of this circumstance, I extended the search to more fundamental concepts. The work at hand cites various architects and architectural theorists who apply the concept of space and hull to describe architecture. This was integrated into existing approaches of exploded views for assemblies to formulate design principles and the theoretical and algorithmic foundation of the ExVAr-system. The concepts of this foundation were implemented into the system, trying to obtain a user friendly interface.

Afterwards, the system was evaluated via user tests. Test users had to perform potentially real tasks while using the system. Most tasks were solvable by most of the test users without any interventions of the test supervisor. This indicates that the designed formalism has the potential to function as a starting point of an adequate visualisation technique in architecture using exploded views. However, the results also demonstrate that not all applied concepts are successful. There are some that could be easily fixed like the cross-hierarchy selection, but also others that would need further investigations, like the sticking boundary relation.

6.2 Contributions

The contribution of this thesis over previous work is the extension of methods for the automated generation of exploded views. The concepts of previous approaches, which are mainly restricted to technical assemblies, are modified, extended, and transferred to the domain of architecture.

For this, the concept of “space and hull”, which is derived from architectural theories, is exploited to obtain a theoretic foundation for the formulation of design principles and design rules.

Design principles of previous approaches to generate exploded views are extended by the principles “context preserving” and “traceability”. In order to realize the principle “context preserving”, the idea of context-sensitive explosion directions is introduced. Based on the concept of “space and hull”, the principle “context preserving”, and conceptual building models of the literature, a hierarchical model is developed and local design rules are formulated. The local design rules implement the design principles and operate on the hierarchical model. Furthermore, the local design rules are transformed to global design rules by a generalization. This generalization makes it possible to apply the design rules independently of the structure of the concrete building model.

6.3 Prospect

In this section, I will discuss possible refinements and extensions to the ExVAr-system. Not only the implementation, but also the underlying algorithm is modular to support adaptations. Parts can easily be integrated or exchanged.

6.3.1 Improvements

Improvements sum up all existing parts of the system that should be reworked to achieve better results or handle special cases. First of all, the current sticking boundary relation is a very rudimentary approach to solve the problem of the fragmentation of the façade (see Figure 5.21). A more sophisticated approach should not only take into account the size of boundaries, but also other spatial relations between the boundaries like the angle.

Furthermore, the unsophisticated back-face sticking method may cause occlusions for concave spatial structures (see Figure 5.23). A more elaborate approach has to recognise these special cases and exclude the occluding boundaries from the set of back-face boundaries.

Also the method to generate explosion lines could be improved. For some types of spatial structures, only their bounding boxes are used for the generation (see Figure 5.19). A better procedure should construct the explosion lines of such a spatial structure from the geometry of the children or the boundaries of the spatial structure. One should note that this would also increase the calculation time.

Another possible improvement addresses the limitations for multi-layered building elements. If a multi-layered building element is modelled as multiple building elements, the system fails to generate coherent explosion views (see Figure 5.22). This could be avoided by detecting these elements and unite them to a single building element.

6.3.2 Extensions

This section presents possible extensions to the ExVAr-system. These extensions extend the functionality or improve the quality of the visualization and may be a benefit to different use

cases. Mostly, other illustration techniques can be integrated to complement the explosion view. For example, cuts could be used to gain more control over the explosion process. Cuts could be realized as follows: The building is split along a cutting plane. This plane could be modelled as an invisible boundary with the highest hierarchy level. This way it could be integrated into the system with no additional adaptation of the underlying algorithm.

Furthermore, transparency could be used to de-emphasize unimportant information. This might also reduce the problem of temporarily occluding boundaries during the camera rotation discussed in the previous chapter (see Section 5.1.5). These boundaries could be simply made transparent if they are in the line of sight to occlude the focused spatial structure.

On the other hand, additional visualisation techniques could also be applied to emphasize relevant information. For example, ghosts of already exploded building elements may clarify the spatial relation of elements. This may be an advantage for exploded windows and doors unless their exact position has great relevance for the bounded room and its furniture.

It might also be a benefit to the perspective floor plan of the ExVAr-system if the doors are visualized above the containing walls. This way, the user could perceive the connections between rooms although the connecting openings might be occluded by the separating wall (see Figure 5.16 for an illustration of the occlusion problem of doors).

Another possible extension could be a more sophisticated method for detecting sharing explosion directions. At the moment, a threshold for the angle of the explosion directions is used to combine the explosion directions of different building elements. It is also conceivable to include blocking constraints or visibility calculations into the method. This way, the context-forming building elements could be exploded with respect to their visibility and their blocking integrity.

The above-mentioned extensions may improve the quality of the visualization. Other conceivable extensions also extend the functionality of the ExVAr-system. A valuable extension would be to replace the single selection by a multi-selection approach. To achieve this, the system has to expose not a single building element but multiple elements at the same time. However, this would increase the complexity of the arrangement of the context elements and focused ones. It could be integrated into the ExVAr-system, but the underlying algorithm would have to be redesigned to satisfy the requirements of a multi-selection. Such a multi-selection could be used to select multiple rooms, for example, for path-finding from one room to another. Of course, one could then easily integrate an automated path-finding tool with a starting and a target room as input, and the multi-selection of the passed rooms as output. However, the multi-selection extension requires quite elaborate implementations.

On the other hand, there are also other extensions that would be less extensive to integrate. A textual search for the structural tree is quite simple to add but would be a great benefit for special use cases, for example, if the user looks for a particular room in a huge building.

There are many conceivable extensions which could be integrated into the ExVAr-system. Some may optimize the visualization of the building, others actually extend the functionality to support special use cases. However, it might also be possible to apply the concepts of the underlying theoretic foundation to other geometry than buildings. This is possible because of performed abstraction of the design rules and the general formulation of concepts. Conceivable

areas of application are vehicles, tunnel systems, natural structures, like caves and animal dens, or other objects which correlate with the concept of space and hull.

6.4 Concluding Statements

Exploded views in architecture are a remarkable alternative to conventional visualization techniques in this domain. They are not only a valuable tool to show the interior of complex structures, but also offer the advantage of displaying the context of elements and how they belong together. This is very important for the visualization of buildings because a building consists of many relating elements that may occlude each other. These concepts were successfully applied to a system for automated generation of interactive exploded views. The design, implementation and evaluation of the ExVAr-system was the main part of this work. The resulting system demonstrates that it is possible to solve potentially real tasks by end users who are not familiar with the system. The difficulty to solve these tasks rather depends on the computer skills of an end user, but hardly on her/his expertise of architecture. Furthermore, I enumerated possible suggestions of improvement and conceivable extensions. The presented concepts may also be applied to other domains then architecture. It was very interesting to investigate existing approaches and experiment with new concepts during the work.

Bibliography

[ALB11] Maneesh Agrawala, Wilmot Li, and Floraine Berthouzoz. Design principles for visual communication. *Communications of the ACM*, 54(4):60–69, April 2011. 3, 4, 5, 23, 34

[APH+03] Maneesh Agrawala, Doantam Phan, Julie Heiser, John Haymaker, Jeff Klingner, Pat Hanrahan, and Barbara Tversky. Designing effective step-by-step assembly instructions. *ACM Transactions on Graphics*, (July):828–837, 2003. 10, 12, 14, 29

[Beh09] Peter Behrens. Kunst und technik. *Der Zeitgeist. Beiblatt zum Berliner Tageblatt*, (4):1–2, 1909. 18, 30

[BG06] Stefan Bruckner and M Eduard Gröller. Exploded views for volume data. *IEEE transactions on visualization and computer graphics*, 12(5):1077–84, January 2006. 16, 17

[BGL+05] J Benner, A Geiger, K Leinemann, Forschungszentrum Karlsruhe, Angewandte Informatik, Joachim Benner, Andreas Geiger, and Klaus Leinemann. Flexible Generation of Semantic 3D Building Models. In *Gröger/Kolbe (Eds.), Proc of the 1st Intern. Workshop on Next Generation 3D City Models*, pages 17–22, 2005. 19, 21, 30

[Bjö92] Bo-Christer Björk. A Conceptual Model of Spaces, Space Boundaries and Enclosing Structures. *Automation in Construction*, 1(3):193–214, 1992. 19, 20, 30

[Can86] John Canny. Collision Detection for Moving Polyhedra. *IEEE transactions on pattern analysis and machine intelligence*, PAMI-8(2):200–209, 1986. 15

[DBS06] Marco Dorigo, Mauro Birattari, and Thomas St. Ant Colony Optimization. *IEEE COMPUTATIONAL INTELLIGENCE MAGAZINE*, (November), 2006. 13

[DR99] Joseph Dumas and Janice Redish. *A Practical Guide to Usability Testing*. 1999. 75

[Gro26] Walter Gropius. Der große baukasten. *Das neue Frankfurt*, (1(1926/1927)):25–30, 1926. 18

[HD07] Benjamin Hagedorn and Jürgen Döllner. High-level web service for 3D building information visualization and analysis. In *Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems - GIS '07*, New York, New York, USA, 2007. ACM Press. 24, 30

[Hil07] Bill Hillier. *Space is the machine*. 2007. 18, 19, 30

[HN04] By Mike Houston and Chris Niederauer. Visualizing Dynamic Architectural Environments. *Communications of the ACM*, 47(8):55–59, 2004. 22

[Hol05] Andreas Holzinger. Usability Engeneering Methods for Software Developers. *Communications of the ACM*, 48(1):71–74, 2005. 74, 75

[HTGD09] Benjamin Hagedorn, Matthias Trapp, Tassilo Glander, and Jürgen Döllner. Towards an Indoor Level-of-Detail Model for Route Visualization. In *2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*, pages 692–697. Ieee, 2009. 25

[JFD10] Zhang Jianping, Yu Fangqiang, and Wu Dapeng. IFC and OpenGL-based representation and development of 3D realistic model in virtual construction The IFC-based 3D realistic model. In *Nottinham University Press, Proceedings of the International Conference on Computing in Civil and Building Engineering*, 2010. 24, 25

[LACS08] Wilmot Li, Maneesh Agrawala, Brian Curless, and David Salesin. Automated generation of interactive 3D exploded view diagrams. *ACM Transactions on Graphics*, 27(3), August 2008. 3, 4, 6, 9, 14, 15, 27, 29, 31, 53

[LAS04] Wilmot Li, Maneesh Agrawala, and David Salesin. Interactive image-based exploded view diagrams. In *GI '04 Proceedings of the 2004 Graphics Interface Conference*, pages 203–212. Canadian Human-Computer Communications Society, May 2004. 17, 27

[LC91] Ming C Lin and John F Canny. A Fast Algorithm for Incremental Distance Calculation. *IEEE Intemational Conference m Robotics and Automation*, (April):1008–1014, 1991. 12

[Lef91] Henri Lefebvre. *The production of space*, volume 142. Oxford Blackwell, 1991. 18

[NHAH03] Christopher Niederauer, Mike Houston, Maneesh Agrawala, and Greg Humphreys. Non-invasive interactive visualization of dynamic architectural environments. In *Proceedings of the 2003 symposium on Interactive 3D graphics - SI3D '03*, pages 55–58, New York, New York, USA, April 2003. ACM Press. 22

[NP93] Jakob Nielson and Victoria Phillips. Estimating the Relative Usability of Two Interfaces: Heuristic, Formal and Empirical Methods Compared. In *CHI '93 Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*, pages 214–221, 1993. 74, 75

[RGGR95] Bruce Romney, Cyprien Godard, Michael Goldwasser, and G. Ramkumar. An Efficient System for Geometric Assembly Sequence Generation and Evaluation. In *ASME. Intl Computers in EngineeringConf.*, pages 699–712, 1995. 9, 10, 12

[SF91] Dore Duncan Seligmann and Steven Feiner. Automated Generation of Intent-Based 3D Illustrations Design Rules : Mapping Intent to Stylistic. *Computer Graphics, Volume 25, Number 4*, 25(4):123–132, 1991. 22, 23

[Som10] Katrin Sommer. *Raumproduktion im frühen 20. Jahrhundert*. PhD thesis, 2010. 18, 30

[Tho06] Caroline Thorpe. Made to measure. *Inside Housing*, (December):21, 2006. www.insidehousing.co.uk. 4

[TKS10] Markus Tatzgern, Denis Kalkofen, and Dieter Schmalstieg. Compact explosion diagrams. In *Proceedings of the 8th International Symposium on Non-Photorealistic Animation and Rendering - NPAR '10*, pages 17–26, New York, New York, USA, 2010. ACM Press. 15, 16

[Wil92] Randall H . Wilson. *On Geometric Assembly Planning*. PhD thesis, 1992. 9, 10, 11, 15

[Xin12] Yu-fei Xing. A method of partial exploded view generated automatic based on disassembly sequence planning. In *2012 24th Chinese Control and Decision Conference (CCDC)*, pages 4150–4153. Ieee, May 2012. 13

[YXBW14] Jiapeng Yu, Li Da Xu, Zhuming Bi, and Chengen Wang. Extended Interference Matrices for Exploded View of Assembly Planning. *IEEE Transactions on Automation Science and Engineering*, 11(1):279–286, January 2014. 9, 13