

Interactive Data Exploration of Large Scale Urban Scenarios Using Mobile Devices

V. Varduhn, R.-P. Mundani, and E. Rank
Technische Universität München, Germany
varduhn@tum.de

Abstract. In this work, we present an approach how mobile devices such as iPhone and iPad can leverage the interactive handling of constructions and built infrastructure. Today, large sets of digital product model information are available and the amount is still growing rapidly. In order to benefit from the entirety of this information w. r. t. insight, focussing on single entities separately is not sufficient. Our approach is capable of storing, handling, and delivering access to vast amounts of constructions and built infrastructure while user interaction can still happen in real time. As a sample application, a real time visualisation with interaction using mobile devices for navigation, construction assembly, and product model modification is presented.

1. Introduction

In the process of planning new constructions or investigating the performance of existing buildings, highly optimised tools focusing on single objects are well known. In order to take effects into account, which originate from the spatial embedding of constructions into a varying surrounding such as air flow change or shadow generation in an urban region due to newly designed sky-scrapers, large sets of constructions have to be processed simultaneously, compare van Oosterom (2005).

This introduces the need for efficient navigation metaphors and data manipulation interfaces for constructions and built infrastructure, as standard input devices such as a keyboard or mouse are not feasible for those needs.

Therefore, we present an implementation of an iPhone/iPad interface for handling large sets of constructions and built infrastructure, which allows large scale navigation and specialised in-door navigation and furthermore supports the modification of single buildings by adapting the underlying product model and modifying the assembly of multiple buildings.

This paper is organised as follows: Initially, the requirements and realisation of the data basis containing product model from constructions and built infrastructure is given, followed by the application of levels of detail in order to handle large amounts of data. Then, the efficient parallel handling of data and their visualisation is presented. After this, the main part of this work is introduced, which describes the coupling of handheld devices to the interactive, parallel framework in a client server implementation.

2. Pre-processing of IFC and BIM Data

As the presented framework has the aim to handle large sets of constructions and built infrastructure, Industry Foundation Classes (IFC) have been chosen as input format, see Froese (1999), since this is the industry standard for Building Information Models (BIM) and, thus, assures a general interoperability of the framework with almost arbitrary data sources, see Benner (2005).

In Fig.1, an overview and the representation of construction details such as computers and equipment of a model are given, which is directly made available from the 3D construction plans of the building without manual processing.

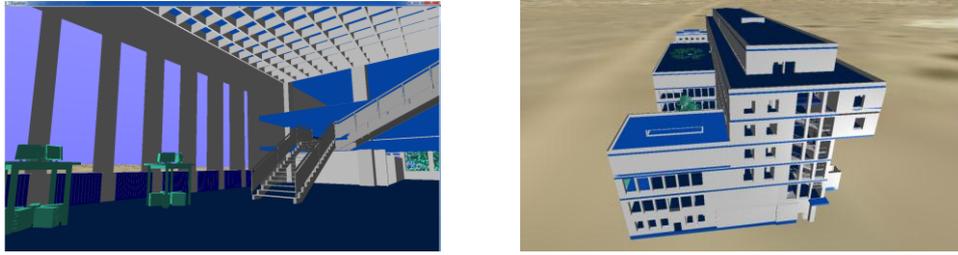


Figure 1: Extracted geometry from IFC input data

From the IFC data, the fully resolved geometry as well as auxiliary details such as the isolation values of glass used for the windows is extracted and stored together with a set of multiple levels of detail (LoD), generated by a step-wise coarsening of the data, compare Lebiedz (2005).

In our framework, two different types of LoDs have been implemented. The first type of LoDs is based on the step-wise coarsening of data by approximating geometric complex entities with their bounding boxes. LoDs are applied during visualisation and selected based on the distance of the user i.e. the camera to the construction and keep the perception without a loss of information, since details such as the edging of a window are negligible at a certain distance.

As the processed data arise from IFC, the mapping between the primitives representing the geometry and their semantic information is available. This information groups the geometry of the construction into the assembly of its forming entities and allows by starting from the fully detailed representation to derive multiples LoDs. In a first step, single entities such as windows or doors are approximated by their bounding boxes. In a next step, groups of elements are reduced to their bounding boxes and as the strongest simplification, a whole building is reduced to a geometric primitive with the matching colouring.

Besides the coarsening of the geometric representation, the distinction between an in-door exploration of a building and the camera being positioned out-door makes a massive reduction of the rendering load possible. If the camera is positioned within one building, the rendering of the terrain and the surrounding buildings can be saved and only the single building in focus has to be processed. If the user explores the data from an out-door position, only the outer hull of the buildings has to be rendered.

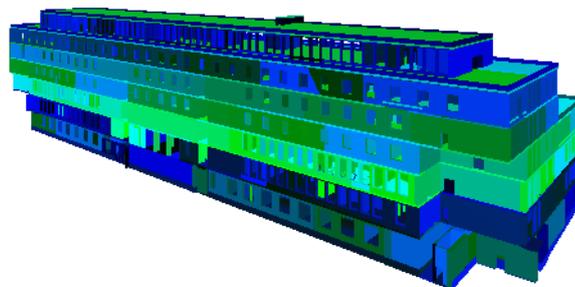


Figure 2: Representation of TUM 1 building with colour encoded identifiers for generating the outer hull of the building

Following this approach, during the pre-processing of the data, the outer hull of a building is computed and stored as an additional level of detail. This level is precomputed based on the fully detailed geometric representation derived from the IFC by exploiting the capabilities of the GPU.

In order to identify all the triangles that form the outer hull of a construction, we encode the identifier of every triangle in the colour by representing its ID to the base of 256. By doing so, with the three digits for red, green and blue we can distinct approximately 16mio triangles, see Foley (1995), whereas large models exceeding this limitation can be handled by splitting up into segments and processing the combined single results. As shown in Fig.2, when rendering the construction with the new colouring, the frame buffer contains the encoded IDs of all triangles visible on the outside. By rotating around the x and y axis, all triangles forming the outer hull are identified.

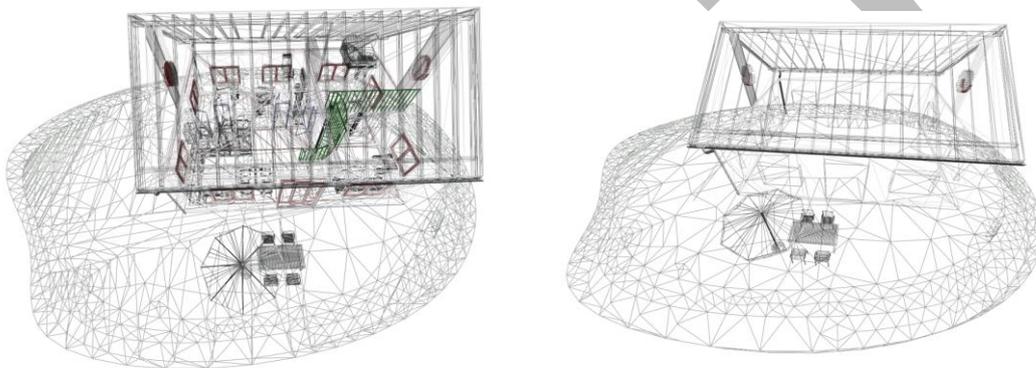


Figure 3: Comparison of the complete model (left) and the remaining primitives forming the outer hull (right)

The reduction of a model to its outer hull results on average in a saving of over 90% and an example is given in Fig.3 for a building with 86k triangles for the original geometry and 7k triangles for the outer hull.

3. Distributed Data Handling Using Hierarchical Structures

In this chapter, we are focusing on the efficient handling of the data, which is inevitable when dealing with large amounts of data, as holding the data for thousands of buildings directly in main memory is not feasible even on these days' modern hardware architectures. Therefore, an distributed memory parallel decomposition of the data basis has been implemented, which orchestrates the holding of the product model on a sufficient large amount of cluster nodes and provides access to the data with the according level of detail applied.

In order to have efficient access to the data handled by the framework, compare Alshawi (1998), a two layer octree structure is implemented, see Fig.4, whereas the first-level octree is kept in main memory over the runtime and delivers access to the spatial assembly of the complete dataset. This first-level octree is built up based on the bounding boxes of all constructions and built infrastructure and keeps as additional data only the storage information for every model.

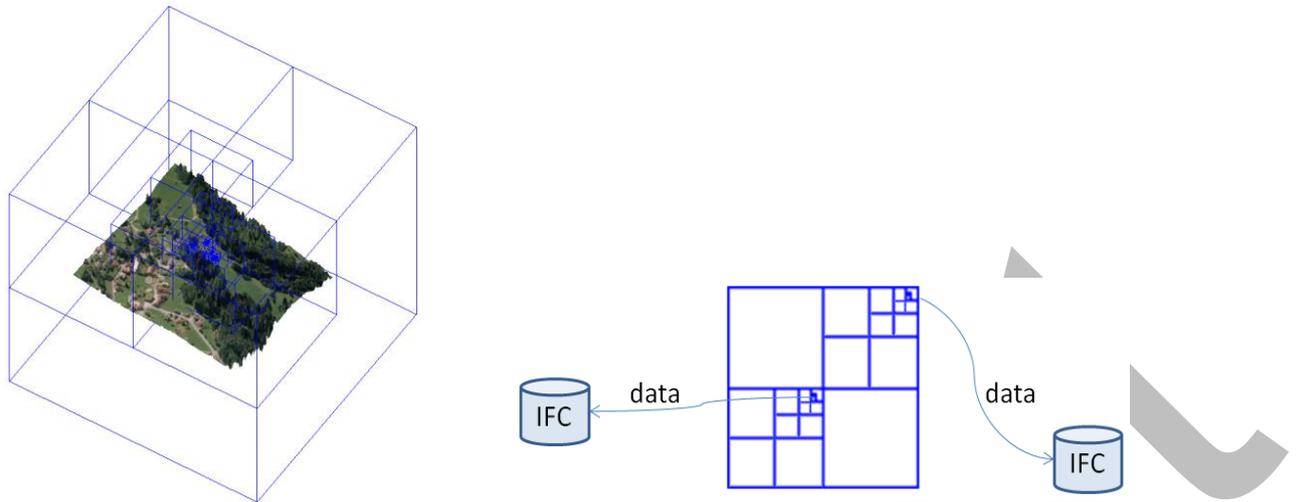


Figure 4: Hierarchical data structures delivering efficient access to the underlying data

A second layer consists of a dynamic set of on-the-fly generated octrees, where each of them operates a single building. In this set, all elements are contained, which are close enough to the point of interest and therefore have to be processed. The point of prefetching, generating, and discarding data is driven by the user interaction and results from the proximity of an object to the point of interest, for a detailed description the reader is referred to Varduhn (2010).

4. Parallel Processing and Visualisation of Construction Data

Based on the presented hierarchical data structure, the processing is distributed to a sufficient large set of parallel processes in a distributed memory fashion. The parallelisation strategy is shown in Fig.5 and incorporates multiple sets of processor types. A single master process holds the first-level octree and orchestrates the whole framework. Every update of the status of the system such as position updates or changes to the data basis is fetched by the master process.

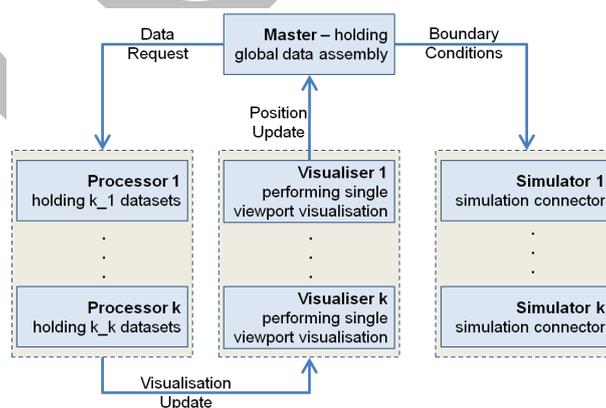


Figure 5: Characterisation of the communication and synchronisation of different process

In the set of processors, every single process holds a set of product model and their corresponding octree representation. The distribution of the models to the processors is done by using a round robin scheme over the list of models ordered by their distance to the point of

interest. By doing so, clustering of models lying close to each other to a few or even a single processor is prevented, as a clustering of load would tremendously increase the response time of the framework.



Figure 6: Parallel visualisation on the NEXcave system running at KAUST.

An update of the visualisation is processed by the master by identifying all buildings and constructions whose level of detail or information have to be adapted. As the first level octree holds the global assembly in a hierarchical data structure, affected models can be identified efficiently. Based on the mapping of processors to the models, all affected processors are identified and triggered for the necessary update. By using an asynchronous communication pattern following MPI, processing times on the master process are kept short in order to avoid this process to become a bottleneck of the framework.

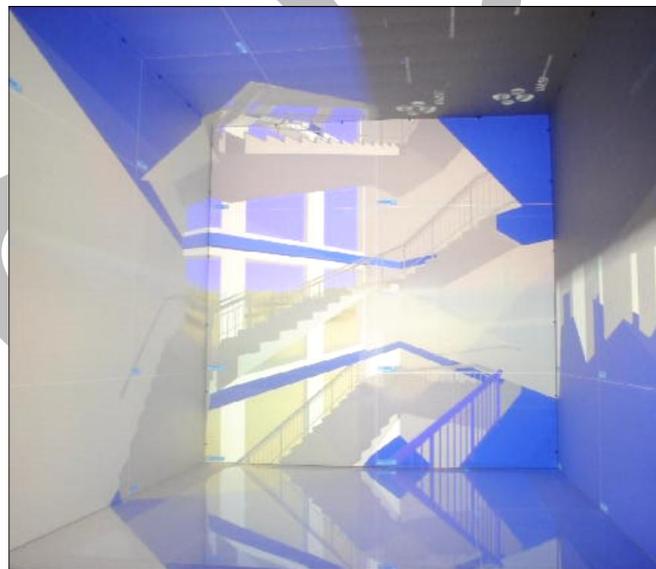


Figure 7: Visualisation of stairway details running on a 200mio pixel CAVE at KAUST.

In order to exploit modern visualisation hardware, to every GPU driving the hardware a single visualisation process is assigned. As the processors are triggered simultaneously but asynchronous, also the update of the visualisation is carried out without slowing down the framework and put into practice by sending the updated geometry to every visualisation process and switching the geometric representation on the target GPU, compare Bishop (2005)

In Fig.6 the running visualisation is shown on the NEXcave installation with 21 passive 3D monitors and 12 GPUs, in Fig.7 for a six-sided Mechdyne CAVE equipped with 96 GPUs, both installed at KAUST VisLab.

5. Communication and Interaction of Handheld Devices

In this section, the implemented interaction metaphors and their technical realisation are introduced. As already described, we have a framework at hand, which is capable of handling large sets of constructions and built infrastructure in an interactive fashion. The described realisation on multi-side immersive visualisation environments gives users a tool at hand, to dive into the exploration of data and perform engineering relevant tasks.

With the presented approach, engineers or architects can explore planned constructions in systems such as a CAVE from the inside or the outside, investigate phenomena such as the influence of shadows to new constructions or plan maintenance tasks for existing buildings. As the data basis contains the fully detailed IFC product model data, the location of specific elevators can be investigated or the installation of smoke detectors can be planned e.g., compare Hammad (2006) and Han (1999).

For all these engineering relevant tasks, users need an intuitive and ergonomic navigation and data manipulation interface when using modern visualisation environments, as standard interfaces such as keyboard or mouse are not applicable in these environments.



Figure 8: Captured motion-based degrees of freedom for navigation using handheld devices

To show the applicability of our approach, we implemented a navigation interface, see Fig.8, and a manager for manipulating the data sets in total, as well as single buildings. In the data set, single buildings can be added or removed and their position and azimuth can be changed. By doing so, the assembly of buildings to each other or whole cities can be investigated or optimised, see Fig.9.

Concerning single buildings, the strength of product models comes into play. As IFC contain the geometric representation of a building indirectly by specifying the location of a window by its alignment in the corresponding wall, e.g., and support of construction specific details, changes to the data can be performed efficiently by manipulating the underlying product model and regenerating the geometric representation.

The communication is put into practice by using the HTTP protocol. The framework is accessible via a socket implementation accepting HTTP connections to which the device sends the request by URL-encoded data and receives the response by extracting the results from an XML stream stored in the message body.

The processing of the incoming commands and requests is performed on the master process and follows the communication paths of the framework, extended by the socket connection. For dispatching and receiving the incoming connections, a standard socket implementation using the BOOST libraries has been made available.

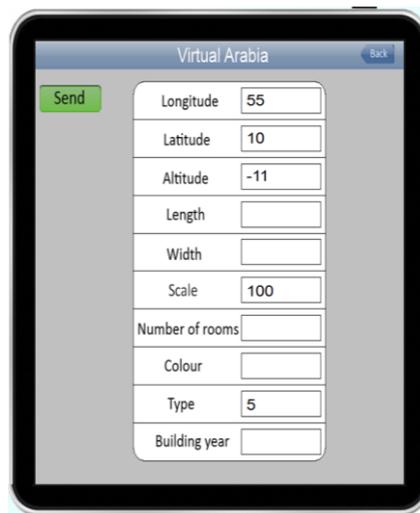


Figure 9: Application interface for manipulating the data basis of the framework and its assembly

Fig. 10 sketches the flow of information i.e. the communication path between the handheld device and the framework. In order to cope with access limitations in modern visualisation facilities, we restricted the framework to the use of standard network interfaces. By allowing the configuration of the listening port for the incoming connections on the framework side, port limitations can be avoided.

To provide basic interaction possibilities with the framework and enable an adaption of the interaction to the need of users, besides the configuration parameters of the network connection such as IP address of the framework and port of the target system, usage specific parameters can be adapted. These contain the sensitivity of the sensors in order to let the user choose, if small changes on the device should be damped or forced, or switching the visualisation to wireframe rendering.

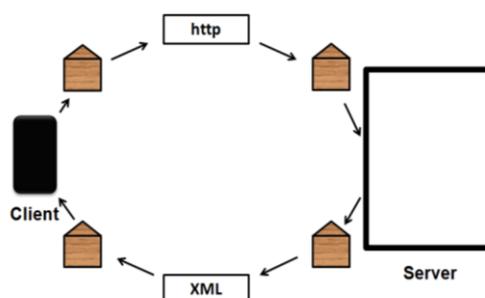


Figure 10: Process cycle for steering and exchanging information between the handheld device and the framework

The navigation is realised with two different metaphors. For the navigation on an urban scale, a “flight-simulator” fashioned implementation uses the gravity sensors of the mobile device and three degrees of freedom for spatial rotations are provided. Besides the adaption of speed which the user travels thru the scene with, also a reverse motion can be activated in order to give the user a maximal freedom for navigating.

Inside buildings the navigation is switched to a behaviour which represents a user walking through the scene. When using this metaphor, gravity sensors are deactivated and motion is based on the input given via cursor keys.

6. Conclusions

In this paper, we presented an approach for extending a framework, which is capable of holding and visualising large amounts of constructions and built infrastructure in order to make it usable in complex visualisation environments.

Since immersive and CAVE like environments let users dive into the data exploration and support engineering tasks by surrounding the user with the information provided, standard input tools such as keyboard and mouse are not applicable and in this work replaced by customer handheld devices.

By introducing a parallel hierarchical processing framework, which is easily adaptable to various topologies of visual environments, real time visualisation and data manipulation is put into practice.

7. Acknowledgements

This publication is based on work supported by Award No. UK-c0020, made by King Abdullah University of Science and Technology (KAUST). Furthermore, the work in this project is supported by the International Graduate School of Science and Engineering (IGSSE) of Technische Universität München.

References

- Bishop, I., & Dave, B. (2001). *Beyond The Moving Camera: Systems Development For Interactive Immersive Exploration of Urban Environments*. Computers in Urban Planning and Urban Management.
- Lebiedz, J., & Mieloszyk, K. (2005). *Real Terrain Visualisation on the Basis of GIS Data*. Intelligent Media Technology for Communicative Intelligence.
- van Oosterom, P., Zlatanova, S., & Fendel, E. M. (2005). *Geoinformation for Disaster Management*. Springer Berlin Heidelberg.
- Varduhn, V., Mundani, R.-P., & Rank, E. (2010). *Real Time Processing of Large Data Sets from Built Infrastructure*. Proc. of the 7th Int. Conf. on Cybernetics and Information Technologies, Systems and Applications.
- Alshawi, M., Aouad, G., Faraj, I., Child, T., Underwood, J. (1998) *The implementation of the industry foundation classes in integrated environments*. CIB W78 Conference, pages 55–66,.
- Benner, J., Geiger, A., Leinemann, K. (2005) *Flexible Generation of Semantic 3D Building Models*. Proc. of the 1st Intern. Workshop on Next Generation 3D City Models.

Faraja, I., Alshawi, M., Aouad, G., Child, T., Underwood, J. (2000) *An industry foundation classes Web-based collaborative construction computer environment: WISPER* . Automation in Construction, 10:79–99.

Foley, J. D., van Dam, A., Feiner, S. K., Hughes, J. F. (1995) *Computer Graphics: Principles and Practice in C. 2nd Edition*. Addison-Wesley Professional.

Froese, T., Yu, K. (1999). *Industry Foundation Class Modeling for Estimating and Scheduling* . 8th International Conference on Durability of Building Materials and Components.

Hammad, A., Zhang, C., Hu, Y. (2006) *Mobile Model-Based Bridge Life Cycle Management*. Proceedings of the 8th International Bridge Management Conference, page I-5/1–13.

Han, C. S., Kunz, J. C., Law, K. H. (1999) *Building Design Services in a Distributed Service Architecture*. Journal of Computing in Civil Engineering, ASCE, 13.