EXTRACTION OF FAÇADES USING RJMCMC AND CONSTRAINT EQUATIONS

Claus Brenner and Nora Ripperda

Institute of Cartography and Geoinformatics, University of Hannover, Germany
{Claus.Brenner, Nora.Ripperda}@ikg.uni-hannover.de

KEY WORDS: Markov Chain, constraint equations, façade modelling, building extraction, least squares adjustment.

ABSTRACT:

Today's processes to extract man-made objects from measurement data are quite traditional. Often, they are still point based, with the exception of a few systems which allow to automatically fit simple primitives to measurement data. At the same time, demands on the data are steadily growing. The need to be able to automatically transform object representations, for example, in order to generalize their geometry, enforces a structurally rich object description. Likewise, the trend towards more and more detailed representations requires to exploit structurally repetitive and symmetric patterns present in man-made objects, in order to make extraction cost-effective. In this paper, we address the extraction of building façades in terms of a structural description. As has been described previously by other authors, we use a formal grammar to derive a structural façade description in the form of a derivation tree. We introduce two new concepts. First, we use a process based on reversible jump Markov Chain Monte Carlo (rjMCMC) to guide the application of derivation steps during the construction of the tree. Second, we attach variables and constraint equations to the symbols of the grammar, so that the derivation tree automatically leads to a constraint equation system. This equation system can then be used to optimally fit the entire façade description to given measurement data.

1 INTRODUCTION

The extraction of man-made objects from sensor data has a long history in research (Baltsavias, 2004). Especially for the modelling of 3D buildings, numerous approaches have been reported, based on monoscopic, stereoscopic, multi-image, and laser scan techniques (Brenner, 2005). While most of the effort has gone into sensor-specific extraction procedures, very little work has been done on the structural description of objects.

Nowadays, in extraction systems, one can choose between boundary representation (BRep) and constructive solid geometry (CSG) modelling. BRep modelling is inspired by traditional photogrammetric point measurement, with subsequent topology definition to obtain lines, surfaces, and volumes. CSG, on the other hand, models objects by combining predefined volumetric primitives using Boolean operations. Thus, it has the intrinsic potential to attach meaning to the primitives and to obtain a structural description in terms of a CSG modelling tree. However, primitives usually reduce to simple geometric shapes such as planar patches, cylinders and spheres, and the CSG tree is often derived according to the modelling process and the desired 3D shape rather than with a semantic modelling of the building structure in mind.

Modelling structure though is very important for downstream usability of the data, especially for the automatic derivation of coarser levels of detail (LoD) from detailed models (a process called generalization). Being able to deliver different LoDs tailored to different customers needs, to context-adapted visualizations, such as on mobile displays, or simply to cut down rendering time of large models is essential for 3D models to enter the market. The Sig3D group has defined five levels of detail for building models (Kolbe et al., 2005). However, the definition of discrete LoDs alone does not imply any path to derive one level from the other in an automated way. Experience from 2D map generalization in cartography shows that generalization purely based on geometric information is indeed a hard problem, which becomes even worse in 3D.

Representing structure is not only important for the later usability of the derived data, but also as a means to support the extraction process itself. A fixed set of structural patterns allows to span a certain subspace of all possible object patterns, thus forms the model required to interpret the scene. Patterns can also guide the measurement process (taking place after the interpretation). Especially for man-made structures such as building façades, a large number of regularity conditions hold, which can be introduced into the measurement process as constraints. In interactive measurement processes, introducing structural descriptions can cut down acquisition time, since repeated or mirrored parts can be introduced in one step.

This paper elaborates on the grammar-based extraction of façade descriptions. The grammar is used in two places. First, it guides the generation of possible façade layouts using a reversible jump Markov Chain Monte Carlo (rjMCMC) process to explore solution space. Second, the obtained derivation tree is used for the automatic setup of constraint equation systems during the fine matching of the generated façade layout to measurement data.

2 RELATED WORK

2.1 Extraction of objects using constraints

The extraction of objects from measurement data is different from computer aided design (CAD) construction. In CAD, the general problem is to derive an instance (a geometrical instantiation) of an object, given a sketch (or just an idea), annotated with dimensional information. Algebraically, sketch annotations are constraints and the sketch defines a constraint graph, out of which a constraint equation system

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0} \tag{1}$$

results, where $\boldsymbol{x}$ is the parameter vector describing the (geometry of the) solution. Finding $\boldsymbol{x}$, given (1), is termed *geometric constraint solving*. In order to obtain a finite set of solutions, $\boldsymbol{f}$ must be well constrained or consistently overconstrained. In contrast, when objects are reconstructed using measurements, the task can typically be formulated as

$$\begin{aligned} \|\boldsymbol{g}(\boldsymbol{b}, \boldsymbol{x})\| &\overset{!}{=} \min, \quad \text{subject to} \\ \boldsymbol{f}(\boldsymbol{x}) &= \mathbf{0}, \end{aligned} \tag{2}$$

where $\boldsymbol{g}$ subsumes the (possibly contradictory) constraints imposed by some measurement data $\boldsymbol{b}$, whereas $\boldsymbol{f}$ represents the "hard" constraints imposed by the model. As opposed to the case in CAD, $\boldsymbol{f}$ will be normally underconstrained (as else the measurements will have no effect on the solution), whereas $\boldsymbol{g}$ will be typically overconstrained (since redundant measurement data is used), which leads to a system which is both locally overconstrained and globally well- or underconstrained.

There are no extraction tools which implement (2) rigorously. For example, modelling of objects from close range scan data is usually carried out using CAD-based systems which combine CAD modelling functionality with the ability to fit CAD objects to point clouds (e.g., (Leica Geosystems, 2006)). In this case, the first part of (2) is implemented, but not the second one. For a practical example, assume that four best-fit planar patches have been extracted from laser scan data. Then, it is not possible to make them meet in a single point except by manual modification (usually, a "snap" operation) of one of the planes – which destroys the initial best-fit property.

The need to introduce constraints into the reconstruction process of man-made objects has been recognized early. For example, Weidner extracts roof faces using a DSM segmentation and proposes to automatically derive mutual relationships between the extracted faces, such as 'same slope', 'symmetry', and 'antisymmetry', in order to insert them as constraints into a global robust adjustment (Weidner, 1997). Although this has been proposed by several authors, constraint-based extraction does not play a role nowadays, except for research systems (Ermes, 2000).

The major problems with constraint-based modelling are *(i)* to insert the constraints in a meaningful manner, *(ii)* to manage, introspect, and debug large constraint equation systems, and *(iii)* to solve constraint equation systems. As opposed to the classical geometric constraint solving problem, which attempts to build a solution "from scratch", in reconstruction, initial values are usually available, so that linearization and iterative estimation can be used for solving the equation system. Thus, the main task lies in the structured insertion and management of constraints. To facilitate this in interactive environments, "weak primitives" have been proposed in (Brenner, 2004). The concept has been extended later to include hierarchical structures using containers (Brenner and Sester, 2005).

2.2 Generalization and incremental modelling

Automation of (manual) map generalization procedures has been a topic in cartography for several decades. There are now first operational systems available, which usually start from a scene description in form of 2D primitives like polygons or polylines. From this, implicit relationships are discovered, such as adjacency, parallel and rectangular structures, distances, protrusions, etc., which are to be modified or preserved during generalization. The final outcome is again a description of the objects in terms of their geometry only. Since the discovered structures are not being made explicit, they cannot be modified, which frequently leads to the need to check and correct the outcome of the automatic generalization step manually.

Recently, in cartography methods are being investigated and developed which aim at the recognition of important structures that are needed as a basis for generalization, e.g. parallelism, linear arrangement, clusters (Christophe and Ruas, 2002, Anders and Sester, 2000). Furthermore, there are approaches which try to separate generalization processes related to different objects in different hierarchical levels, e.g. when defining generalization modules that can be handled independently (Kilpeläinen and Sarjakoski, 1995). A first attempt to explicitly model these structures has been done in the AGENT project, where different hierarchical levels of objects have been specified that can act independently with a specific dedicated behavior (Lamy et al., 1999).

In (Brenner and Sester, 2005), the previously mentioned approach of primitives and containers has been extended to include discrete behavior. Primitives are defined as the combination of geometric description (e.g., polygons), sets of constraints (e.g., all line segments aligned horizontally or vertically), and discrete behavior (e.g., boundary simplification rules). Containers provide the ability to spatially layout primitives, with dedicated interface slots which allow to connect primitives to containers. This leads to a simple hierarchical description scheme, which is extended in this paper to a grammar-based description.

2.3 Modelling of architectural patterns

Grammars have been extensively used to model structures. For modelling plants, Lindenmayer systems were developed by the biologist Aristid Lindenmayer (Prusinkiewicz and Lindenmayer, 1990). They have also been used for modelling streets and buildings (Parish and Müller, 2001, Marvie et al., 2005). However, Lindenmayer systems are not necessarily appropriate for modelling façades. Façades differ in structure from plants and streets, since they don't grow in free space and modelling is more a partition of space than a growth-like process.

For this reason, other types of grammars have been proposed for architectural objects. Stiny introduced shape grammars which operate on shapes directly (Stiny and Gips, 1972). The rules replace patterns at a point marked by a special symbol. Mitchell describes how grammars are used in architecture (Mitchell, 1990). The derivation is usually done manually, which is why the grammars are not readily applicable for automatic modelling tools.

Wonka et al. developed a method for automatic modelling which allows to reconstruct different kinds of buildings using one rule set (Wonka et al., 2003). The approach is composed of a split grammar, a large set of rules which divide the building in parts, and a control grammar which guides the propagation and distribution of attributes. During construction, a stochastic process selects among all applicable rules.

Dick et al. introduce a method which generates building models from measured data, i.e. several images (Dick et al., 2004). This approach is also based on the rjMCMC method. In a stochastic process, 3D models with semantic information are built.

3 GRAMMAR-BASED FAÇADE RECONSTRUCTION

In this section, the basic concept of our method is described. As in the approaches outlined in the previous section, we use a grammar to define façade layout. However, we do not want to generate artificial façade descriptions, but rather derivation trees which correspond to measurement data. Two major tasks can be identified:

1. the recognition of the façade structure, i.e., building of a structural description in the form of a derivation tree, together with a first instantiation of all (geometric) parameters, and

2. measurement, i.e., fine-matching the geometry of this initial structure to the measurement data.

The first task is the interpretation step, for which we describe an approach that uses rjMCMC to explore different derivation trees. As for the second task, we propose to attach constraint equation systems to the derivation rules such that a complete derivation tree not only defines the structure and initial layout, but also a set of constraints which allow to precisely match the structure to the measurement data.

For our experiments, we use terrestrial laser scan data and images. For the moment, we concentrate on façades, i.e., the measurement data consists of point clouds and orthorectified images of single façades.

3.1 Façade grammar

The façade model is described in terms of a recursive partition of space. Each part is represented by one of the symbols listed in table 1 and 2. There are two kinds of symbols, the first one being nonterminals (table 1). Geometrically, nonterminals do not represent façade geometry directly but serve as containers which hold other objects, represented in the derivation tree by nonterminal or terminal children. The second group contains the terminal symbols, which represent façade geometry and cannot be subdivided further (table 2).

AboveDoor	IdenticalFaçadeArray
AboveWindow	PartFaçade
Façade	StaircaseColumn
FaçadeArray	SymmetricPartFaçade
FaçadeColumn	SymmetricPartFaçadeMiddle
FaçadeElement	SymmetricPartFaçadeSide
FaçadeRow	SymmetricFaçade
Gable	SymmetricFaçadeMiddle
GroundFloor	SymmetricFaçadeSide

Table 1: Nonterminal symbols corresponding to containers.

Door	Wall
DoorArch	Window
StaircaseWindow	WindowArch

Table 2: Terminal symbols corresponding to façade geometry.

The start symbol is the symbol Façade. Starting from it, the model can be expressed as a derivation tree with Façade as root. The subdivision is made by rules similar to the ones introduced by (Wonka et al., 2003). Figure 1 shows an example façade. The Façade can be partitioned into GroundFloor and upper parts of the building, modelled as PartFaçade. PartFaçade shows symmetry and therefore only one side is modelled as SymmetricPartFaçadeSide. In this part the windows are arranged in a regular grid modelled by an IdenticalFaçadeArray. This array can be instantiated with a single Window which is placed at each grid position. The GroundFloor doesn't show any regularities which is why it is subdivided into FaçadeElements which can contain Windows or Doors. Each rule has a left side which consists of one symbol and a right side which may comprise several symbols in a certain spatial layout. The result of the method is a derivation tree which describes the model of the façade.

3.2 Exploration of the derivation tree using rjMCMC

We use rjMCMC for the construction of the derivation tree. The tree is encoded in a vector θ, which holds all parameters which are present in the derivation tree, e.g. positions and sizes of terminal symbols. The task is to find the optimum value for θ, given measurement data. In terms of a distribution, we are therefore looking for the maximum (mode) of the distribution $P(\theta|D_S D_I)$, i.e., the conditional distribution of θ, given scan data D_S and image data D_I. Finding this maximum by an exhaustive search is not feasible, due to the dimension of θ. Therefore, we use a stochastic method to instantiate the value of θ randomly. The overall approach is thus of the type hypothesize-and-test, where the hypotheses are generated randomly and tested afterwards, using measurement (scan and image) data. In order to be feasible, the samples θ are drawn from the distribution $P(\theta|D_S D_I)$, so that more samples are in the vicinity of high distribution values (i.e., close to probable façade layouts). The problem with this is that first, $P(\theta|D_S D_I)$ usually has a highly complex shape, far from a standard distribution, so drawing samples is nontrivial. Second, $P(\theta|D_S D_I)$ is not analytically available. The first problem is solved using Markov Chain Monte Carlo (MCMC, see e.g. (Gilks et al., 1996)). Basically, using the algorithm of Metropolis-Hastings, a Markov chain is obtained which converges to the desired distribution. Thus, after an initial phase, the algorithm delivers samples drawn from the distribution $P(\theta|D_S D_I)$. As for the second problem, using Bayes' law, $P(\theta|D_S D_I) \propto P(\theta)P(D_S D_I|\theta)$. The first term (prior) is evaluated using plausibility functions, which are set up manually. For example, one part of $P(\theta)$ describes assumptions about window sizes (by assuming a distribution). The second term (likelihood function) is evaluated by a score function based on the model (defined by θ) and scan and image data. The realization of both terms is described in more detail below. Thus, to summarize, the method explores the solution space by drawing samples from a (posterior) distribution, without the need to know this distribution analytically. Since the derivation tree changes during the process, the dimension of θ changes as well, and MCMC is not directly applicable. To resolve this, rjMCMC is used, which allows jumps between spaces of different dimension (Green, 1995). Our approach is described in more detail in (Ripperda and Brenner, 2006).

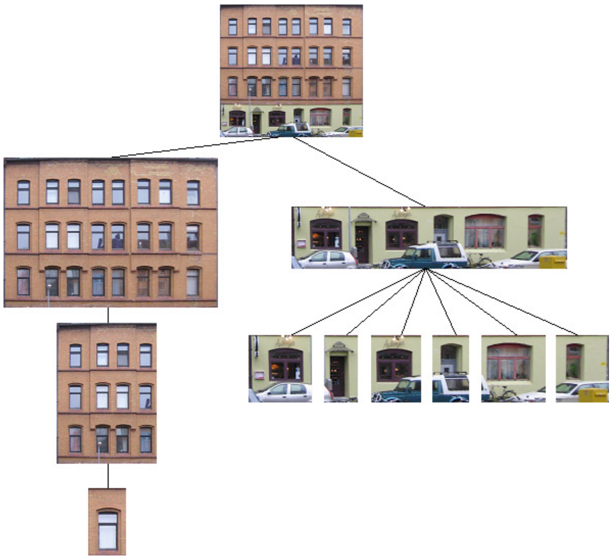

Figure 1: Example partition of a façade.

During the exploration of the derivation tree, any state change can be assigned to one of the following categories:

- Application of a split rule from the grammar. Façade elements are divided horizontally, vertically or in both directions and each part becomes a new symbol (see Fig. 2). In fact, one grammar rule comprises a set of changes to the parameter vector θ, since the associated attributes have to be chosen, such as the number and size of children. Figure 3 shows an example where one rule splits the symbol Façade into FaçadeColumns. The number of columns and their width is determined randomly. If a Façade can be divided into several FaçadeColumns the general rule stands for all rules of this kind with different number of columns and different positions.

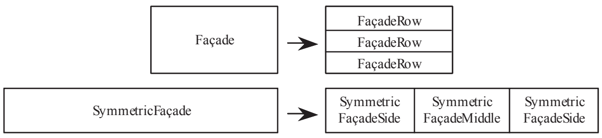

Figure 2: Split rules.

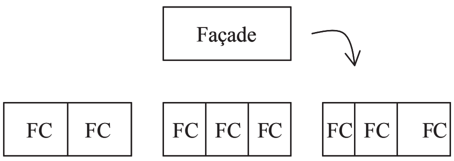

Figure 3: Different applications of a split rule.

- Changes in structure. Even after derivation of new containers according to the previous step, a second set of state changes allows to modify parameters, e.g. the number of columns or the position of the parting lines between columns (see Fig. 4). The same can be done starting from a child symbol. In this case, the neighbor symbols which are involved in the change have to be changed as well.

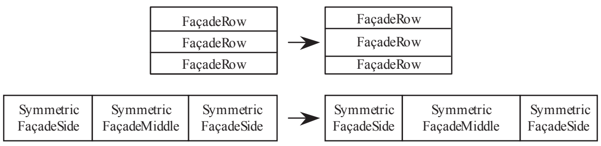

Figure 4: Changes which modify splits.

- Replacement of symbols. This allows to interchange one symbol in the derivation tree by another symbol. In this case, the geometry stays the same, but the denotation changes. This is especially used in the case of the symbols AboveDoor and AboveWindow. For example, the space above a window is modelled by the symbol AboveWindow. The rules

  AboveWindow → WindowArch
  AboveWindow → Wall

  allow to replace this symbol.

The control is done by the rjMCMC method. To ensure the reversibility, each change can be applied from left to right and vice versa. This is a difference to the way split grammars are used, but is a requirement for the rjMCMC approach. A change is proposed depending on the jumping distribution $J_t(\theta_t|\theta_{t-1})$ which expresses the likelihood for each change.

For the evaluation of changes, we use different methods which can be divided into two groups. The first group contains methods which test the general plausibility of the model of the façade. In the group there are methods which test how good the model fits the data. This group subdivides in methods working with range data and methods working with image data. In any case, the evaluation functions return a probability which is used to decide if the change is accepted or rejected.

The general plausibility depends on the alignment, the extent and the position of the façade elements. Windows are usually arranged in rows and columns. Therefore, such layouts are assigned a high acceptance probability. We consider the size and the aspect ratio of façade elements to rate their probability. We also use the size for the rating of the subdivision into rows, columns or arrays. A row which is five meters high is not very likely and thus has a low acceptance probability. The last general criterion is the position of the elements. A door in the third floor is not very likely, so only doors in the ground floor are assigned a high probability.

To evaluate the match of the data to the model, scan and image data are used. In the first case, the fact that window points typically lie behind the façade is exploited. In the second case, color difference has been used since windows typically appear darker than the surrounding façade. In both cases, the information is used for the subdivision into rows, columns, and arrays as well. For example, upon division into rows, the resulting row strips are correlated to obtain an acceptance probability. Additionally, in image data a color change may indicate a changeover of ground floor and first floor.

Fig. 5 and 6 show the partition of a symmetric façade and the corresponding derivation tree. The symbol Façade is replaced by SymmetricFaçade. SymmetricFaçade is split into SymmetricFaçadeSide and SymmetricFaçadeMiddle. Each one is further subdivided into IdenticalFaçadeArray and FaçadeElements, respectively. Window and Door are on the leaf level.

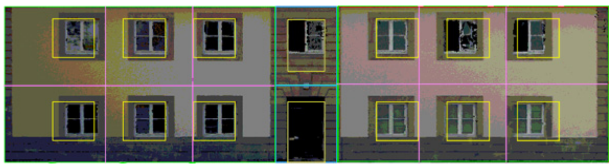

Figure 5: Resulting partition of a façade.

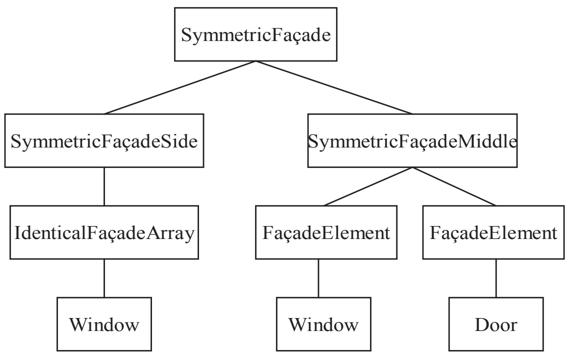

Figure 6: Derivation tree of the façade shown in figure 5.

3.3 Introduction of constraints

In 2D, with points represented by $\boldsymbol{p} = (x_1, y_1)^{\mathbf{T}}, \boldsymbol{q} = (x_2, y_2)^{\mathbf{T}} \in \mathbb{R}^2$ and lines by $\boldsymbol{l} = (a_1, b_1, c_1)^{\mathbf{T}}, \boldsymbol{m} = (a_2, b_2, c_2)^{\mathbf{T}}$ (in Hesse normal form $ax + by + c = 0$), typical logic constraint equations are $a_1^2 + b_1^2 - 1 = 0$ ($\boldsymbol{l}$ having a unit length normal vector), $a_1 x_1 + b_1 y_1 + c_1 = 0$ ($\boldsymbol{p}$ incident $\boldsymbol{l}$), $a_1 a_2 + b_1 b_2 = 0$ ($\boldsymbol{l}$ perpendicular $\boldsymbol{m}$), $a_1 b_2 - a_2 b_1 = 0$

($\boldsymbol{l}$ parallel $\boldsymbol{m}$), whereas dimensional equations include $a_1x_1 + b_1y_1 + c_1 - d = 0$ ($\boldsymbol{p}$ having (signed) distance d from $\boldsymbol{l}$), $(x_1 - x_2)^2 + (y_1 - y_2)^2 - d = 0$ ($\boldsymbol{p}$ having Euclidean distance d from $\boldsymbol{q}$), $a_1a_2 + b_1b_2 - \cos\varrho = 0$ and $a_1b_2 - a_2b_1 - \sin\varrho = 0$ (two oriented lines $\boldsymbol{l}$ and $\boldsymbol{m}$ enclosing the fixed angle ϱ). Thus, constraints between objects often result in bilinear equations. For solving those constraints, linearization and least squares estimation can be used. As noted earlier, the main problem is to introduce constraints in a sensible way so that they are manageable and constraint dependencies are minimized.

We use the derivation tree to define the set of constraints automatically. Two types of constraints can be generated from this tree. Terminal symbols represent geometry, which is fitted to measurement data. Thus, terminal symbols can generate fitting constraints, depending on the measurement data type, e.g. least squares fitting of surfaces to laser scanner data, or fitting of edges to the orthorectified image. Nonterminal symbols, on the other hand, can introduce constraints between their children, such as alignment, size, or orientation.

As an example, Fig. 7 shows a derivation tree (as obtained by the grammar), the corresponding geometric representation, and the generated unknowns and constraints. IDENTICALFAÇADEARRAY, as seen by the grammar, subdivides space into a regular array (depicted here as 2x3 array). From a unknowns/ constraints viewpoint, IDENTICALFAÇADEARRAY introduces column alignment lines at x_1, x_2, x_3 and row alignment lines at y_1, y_2. As IDENTICALFAÇADEARRAY enforces a regular column spacing, a constant distance Δx together with constraint equations $x_{i+1} - x_i = \Delta x$ is introduced. Since IDENTICALFAÇADEARRAY enforces identical sizes as well, width w and height h variables are introduced. All variables are inherited, i.e., the FAÇADEELEMENT shown in the figure receives the relevant alignment variables x_3 and y_1 as well as w and h. WINDOW is a weak primitive p and thus consists of geometry and internal constraints. To the outside, it offers variables $p.c_x$, $p.c_y$ (the center), $p.w$ (width), $p.h$ (height) in the form of fields (slots). Those fields are connected to the inherited variables x_3, y_1, w, h by the addition of four constraints. Being a terminal symbol, WINDOW represents a "real" geometry. Thus, additional constraints are added which match the geometry of WINDOW to the measurement data.

In contrast to the approach in (Wonka et al., 2003), the distinctive feature of our approach is that we do not "copy" attribute values down the derivation tree, but rather distribute (symbolic) variables. These variables can be used by children in arbitrary complex ways by introducing constraint equations. By the distribution of variables and the link by constraints, the geometric representation of the tree is "alive" in the sense that changes in one place can propagate across the entire tree. Finally, mapping the tree to a constraint equation system and subsequent solution of that system in the least squares sense allows a mathematically thorough, well-defined solution, which seamlessly integrates observations and constraints. To experiment with constraint equation systems in 2D, we have developed an environment which allows the interactive modification of geometric items while geometric constraints are enforced using least squares estimation (Fig. 8).

4 CONCLUSIONS AND OUTLOOK

In this paper, we have proposed to use grammars for the extraction of façade descriptions from measurement data. We introduced two major concepts. First, the use of rjMCMC to guide

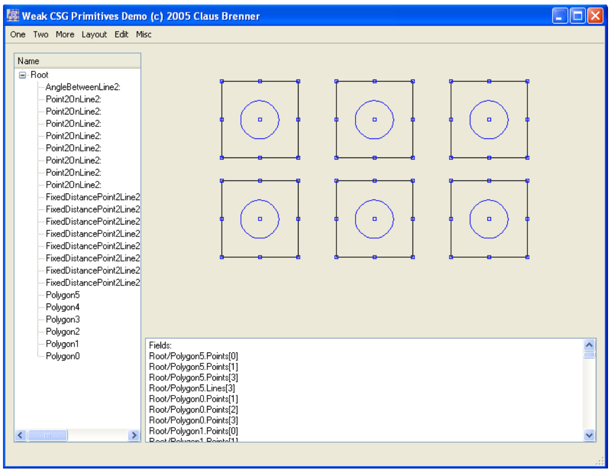

Figure 8: Snapshot of the interactive tool for evaluation of constraint equations.

the construction of the derivation tree, in conjunction with evaluation functions which rate possible changes based on measurement data. Second, the use of the hierarchic derivation tree structure as a means to automatically establish constraint equations for a subsequent least-squares fitting of the façade description to the measurement data. For the future, we plan to enlarge our set of derivation rules as well as to improve our evaluation functions.

REFERENCES

Anders, K.-H. and Sester, M., 2000. Parameter-free cluster detection in spatial databases and its application to typification. In: IAPRS Vol. 33 Part A4, Amsterdam.

Baltsavias, E. P., 2004. Object extraction and revision by image analysis using existing geodata and knowledge: current status and steps towards operational systems. ISPRS Journal of Photogrammetry and Remote Sensing 58, pp. 129–151.

Brenner, C., 2004. Modelling 3D objects using weak primitives. In: International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. XXXV, Istanbul, 2004.

Brenner, C., 2005. Building reconstruction from images and laser scanning. Int. Journal of Applied Earth Observation and Geoinformation 6(3-4), pp. 187–198.

Brenner, C. and Sester, M., 2005. Cartographic generalization using primives and constraints. In: Proc. 22nd International Cartographic Conference, July 9-16, La Coruna, Spain.

Christophe, S. and Ruas, A., 2002. Detecting building alignments for generalisation purposes. In: International Symposium on Spatial Data Handling SDH 2002, Ottawa, Canada, pp. 419–432.

Dick, A., Torr, P., Cipolla, R. and Ribarsky, W., 2004. Modelling and interpretation of architecture from several images. International Journal of Computer Vision 60(2), pp. 111–134.

Ermes, P., 2000. Constraints in CAD models for reverse engineering using photogrammetry. In: IAPRS Vol. 33 Part B5, Amsterdam.

Gilks, W. R., Richardson, S. and Spiegelhalter, D. J. (eds), 1996. Markov Chain Monte Carlo in Practice. Chapman & Hall.

Green, P. J., 1995. Reversible jump markov chain monte carlo computation and bayesian model determination. Biometrika 82(4), pp. 711–732.

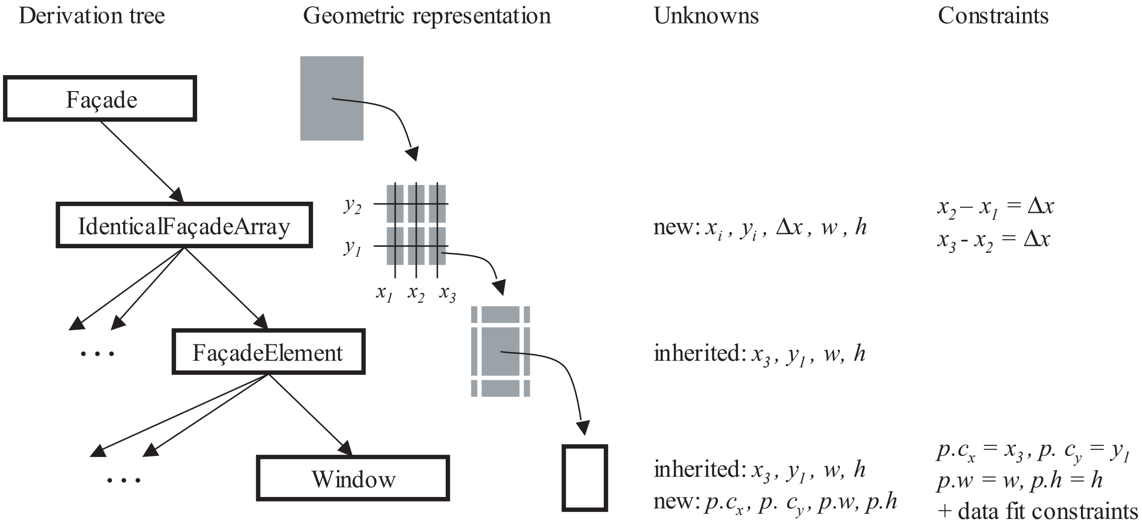

Figure 7: Derivation tree, corresponding geometry, and generated unknowns and constraints.

Kilpeläinen, T. and Sarjakoski, T., 1995. Incremental generalization for multiple representations of geographic objects. In: J. Muller, J. P. Lagrange and R. Weibel (eds), GIS and Generalization: Methodology and Practise, Taylor & Francis, London, pp. 209–218.

Kolbe, T. H., Gröger, G. and Plümer, L., 2005. CityGML – interoperable access to 3D city models. In: P. van Oosterom, S. Zlatanova and E. Fendel (eds), Proc. of the Int. Symposium on Geo-information for Disaster Management, Delft, March 21.-23., Springer.

Lamy, S., Ruas, A., Demazeau, Y., Jackson, M., Mackaness, W. and Weibel, R., 1999. The application of agents in automated map generalization. In: Proceedings of the 19th International Cartographic Conference of the ICA, Ottawa, Canada.

Leica Geosystems, 2006. Cyclone software. http://www.leica-geosystems.com/hds, last accessed March 2006.

Marvie, J.-E., Perret, J. and Bouatouch, K., 2005. The FL-system: a functional L-system for procedural geometric modeling. The Visual Computer 21(5), pp. 329 – 339.

Mitchell, W. J., 1990. The Logic of Architecture : Design, Computation, and Cognition. Cambridge, Mass.: The MIT Press.

Parish, Y. and Müller, P., 2001. Procedural modeling of cities. In: E. Fiume (ed.), ACM SIGGRAPH, ACM Press.

Prusinkiewicz, P. and Lindenmayer, A., 1990. The algorithmic beauty of plants. New York, NY: Springer.

Ripperda, N. and Brenner, C., 2006. Reconstruction of facade structures using a formal grammar and rjMCMC. In: 28th Annual Symposium of the German Association for Pattern Recognition (accepted).

Stiny, G. and Gips, J., 1972. Shape Grammars and the Generative Specification of Painting and Sculpture. Auerbach, Philadelphia, pp. 125–135.

Weidner, U., 1997. Digital surface models for building extraction. In: A. Grün, E. Baltsavias and O. Henricsson (eds), Automatic Extraction of Man-Made Objects from Aerial and Space Images (II), Birkhäuser, Basel, pp. 193–202.

Wonka, P., Wimmer, M., Sillion, F. and Ribarsky, W., 2003. Instant architecture. ACM Transaction on Graphics 22(3), pp. 669–677.

ACKNOWLEDGEMENT

This work has been carried out within the scope of the junior research group "Automatic methods for the fusion, reduction and consistent combination of complex, heterogeneous geoinformation", funded by the VolkswagenStiftung, Germany.