Eurographics Conference on Visualization (EuroVis) (2013)
M. Hlawitschka and T. Weinkauf (Editors)

Short Papers

B:3D - Visualize Land-use Plans Interactively

Max Langbein, Inga Scheler, Achim Ebert, Hans Hagen

TU Kaiserslautern,Germany

Abstract

In Europe, urban land-use plans only give a very abstract representation of possible building forms. Combined with a written statement describing all constraints, the so-called building code, architects have to start planning. This can create many problems by violating the building codes. To overcome these problems we developed a system visualizing building code violations in real-time and giving a spatial impression of the planned area by an additional output on 3D printers and 3D displays.

Categories and Subject Descriptors (according to ACM CCS): J.1.4 [Computer Applications]: Administrative Data Processing—Government H.5.m [Information Systems]: Information Interfaces and Presentation—Miscellaneous I.3.8 [Computing Methodologies]: Computer Graphics—Applications

1. Introduction

The goal of this project is two-fold: one goal is to help architects to avoid building code violations, the other goal is to help town planners to get a better idea of the practical effect of the restrictions they predefine and to filter out building code violations before the building project starts. Traditionally, town-planners and architects used two-dimensional maps of a land-use plans or properties. These, however, give almost no spacial impression of the plan with constructed houses. Also building code violations of a constructed house can not be recognized in an interactive manner. In discussions with a group of architects the idea to visualize the building constraints in correlation to the three dimensional house and terrain models came up. As result we provide a tool to create, change and visualize a set of standard houses and the specifically planned houses on the land-use plan together with building code violations in 3D. Here our workflow for the visualization: The architects convert an existing land-use plan giving all terrain information to Autocad's dxf-format because of its widespread use in architectural environments. To use it in our system, it has to adhere to our naming conventions. Then the we converted written statements of the building code and its violations (see figure 1) into the format needed for our system. The building plan together with a terrain model is the loaded into the system, and example houses are placed on every property and optimized semi-automatically (see figures 9,10). Having done this, the spatial relationships, the spatial impression, the violations and the shadow casts for the whole plan are visible and the user can "play" interactively with the houses and the plan to define multiple building variants. For the three-dimensional visualization we use a framework for stereo displays so different three-dimensional monitors can be used as output. Additionally, we implemented the interface for the use of a "physical" visualization using a three-dimensional printer. The checking of the building code is implemented by a fuzzy logic function including all given constraints, allowing to browse the condition tree (see figure 8), and finally do an automatic optimization. Previous work that has been done in this area includes [För02], a diploma thesis that tried to construct "maximal" houses that obey the rules. For the Fuzzy-Logic optimization procedures in section 4, a similar approach together with some nice references can be found in [GDS10]. The first version of this project has been presented from the architect's perspective by [Ker08]. A central part of our approach is automatic parameter optimization. A visualization with a good database concept and user study can be found in [SHR06].

2. Visualization

To visualize violations of the building laws, metaphors have been introduced for the different rules (see figure 3): The parts of the distance surfaces which are outside of the property are displayed in red. The part of the houses' base area outside the allowed area is shown as a red polygon outline above the house with its vertices connected downwards with red lines. Roof inclination angles outside the allowed range

DOI: 10.2312/PE.EuroVisShort.EuroVisShort2013.037-041

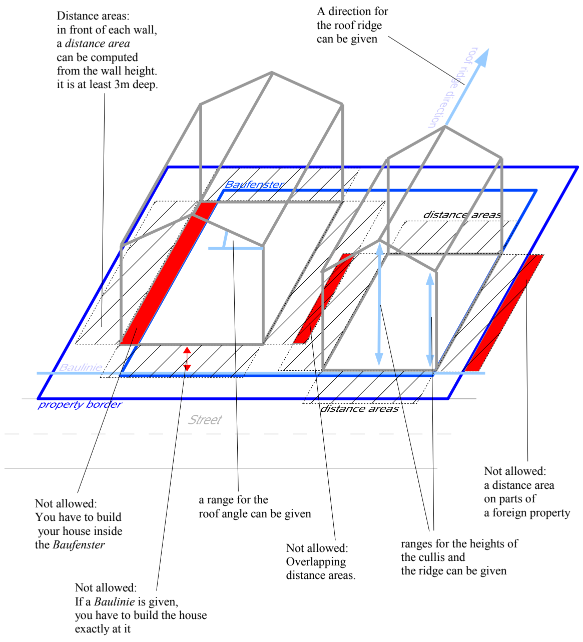


Figure 1: *Some of the restrictions which are or can be given in a german land-use plan*

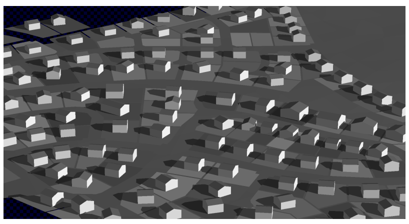


Figure 2: *a newly-planned area with example house stubs*

are visualized by two triangles with one leg's angle corresponding to the angle limit that is exceeded and the other legs angle corresponding to the actual angle of the roof. Ridge angle violations are visualized by two triangles attached to the ridge whose one leg corresponds to the prescribed orientation and the other leg to the ridge. Finally, ridge and cullis height violations are shown by red rectangular surfaces at the limit height. Apart from the violations, also shadow casts are visualized using OpenSceneGraph's Volume Shadows, see e.g. [EK03]. For this, the sun's position during a day is simulated, and you have a slider to choose the sun's seasonal height in a range from midwinter to midsummer, and another slider to choose the time scaling factor in units of hours per second.

In tables 1 and 2 you see our assessment of different output devices (green=good, yellow=ok, red=bad). We tried an autosteroscopic 3D display to show the output also to the public without additional effort managing glasses. However,

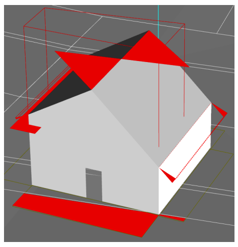


Figure 3: *violations shown in our software*

Criterion \ Type		Auto-stereo-scopic	Polari-zing glasses	Shutter glasses	Color glasses	Color glasses low band-width	Head-mounted displays	2D Display
	Separation Method	direction	polari-zation	time	color	wave-length	position	none
Quality	Resolution	🟥	🟩	🟩	🟩	🟩	🟨	🟩
	Color	🟩	🟩	🟩	🟥	🟩	🟩	🟩
	3D-Impression	🟨	🟨	🟨	🟨	🟨	🟩	🟥
Usability	Input	🟥	🟨	🟨	🟨	🟨	🟨	🟩
	Viewing	🟩	🟨	🟨	🟨	🟨	🟥	🟩
Cost		🟥	🟥	🟨	🟩	🟥	🟥	🟩

Table 1: *assessment of 3D displays*

because of its usually bad resolution you also need a small 2D steering screen. It turns out to be not the best choice for actually working with the system, as 1 shows, but the framework we use also supports multiple other 3D screens. To have a model which you can look at and touch it, we also use a 3D printer as output device. Because of its low costs and good usability we used a printer which uses cellulose powder and one that uses gypsum as material. The rather bad quality is acceptable, as the houses should not represent real buildings but just possibilities. It should be like an indentikit picture, which also should not be too good quality to leave room for interpretation.

3. Create 3D Prints

To print the 3D model, some additional constraints have to be met: The model has to be robust enough, it should not use too much material because of cost and weight, it has to fit into the 3D printer, and if the model is not colored, some parts have to be visualized as relief. In our case the prop-

Criterion \ Method		Computer Navigated carving	Raster Methods			
			Photo-induced Polyme-rization	Fused Deposition Modeling	Powder Gluing	
					Gypsum	Cellulose
Quality	Resolution		🟨	🟥	🟥	🟥
	Robustness	🟩	🟨	🟨	🟥	🟥
	Surface	🟩	🟨	🟥	🟥	🟥
	Weight		🟨	🟨	🟥	🟩
Usability	Generating Input	🟥	🟩	🟩	🟩	🟩
	Operating	🟨	🟩	🟩	🟨	🟨
Cost	Aquisition	🟥	🟨	🟨	🟩	🟩
	Material		🟥	🟨	🟩	🟨

Table 2: *assessment of 3D printing methods*

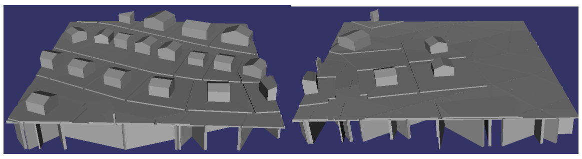


Figure 4: *Two adjacent tiles of the model made printable in 3D*

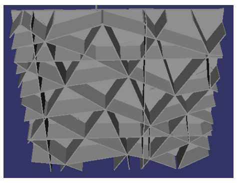


Figure 5: *one tile seen from below showing the supporting structure.*

erties boundaries are modeled as relief (the elevated polygons around the houses in figure 4). To make the model light and save material only the surface of the model is printed and thickened so it does not break. As the thickening should not be visible, it is done in the direction of the inside of the objects. Then to keep the model robust, the following supporting structure is added below the model: The edges of the terrain model's triangulation are elongated downwards and thickened. Additionally, planes in direction of the coordinates are added equally spaced to help stability if the triangulation is too raw.(see figure 5). Afterwards, the model is split into tiles which fit into the printer. To make the tiles composable well while not having too high support structures, the lower bound z_l of the supporting structure is rounded down to the next multiple of a certain height step : $z_l = \lfloor z_{min}/h - 1.5 \rfloor h$, with z_{min} being the lowest point of the terrain of this tile and h the height stepping (see figure 6).

4. Construction and Optimization of the Houses

To construct arbitrary houses, we have a a library of housetypes with parameters $\vec{x}$. Every house type implements an abstract class containing the following query methods: Volume, base area, floor area, ridge orientation(s), roofs, walls, basis, distance surfaces, roof type(s). Additionally the parameter names and the sanity ("has a reasonable shape") can

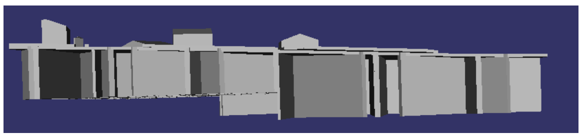
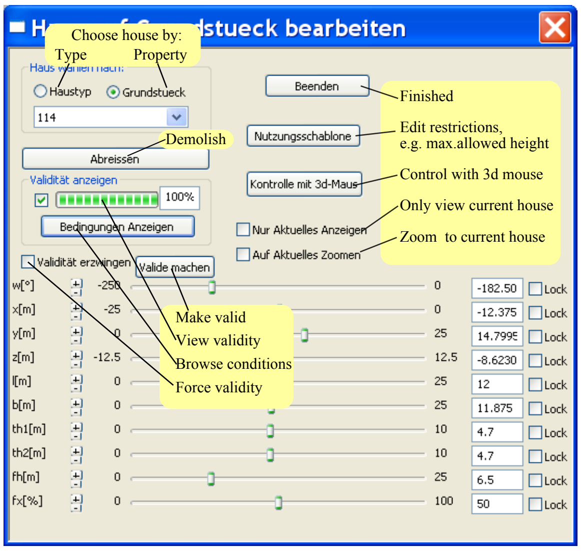


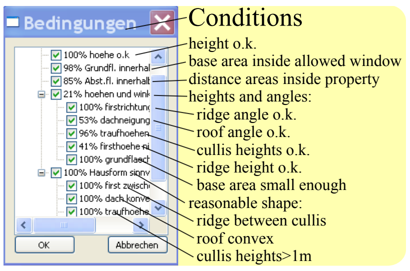
Figure 6: *A tile seen from the side showing two height levels for the supporting structure*



w[°]	-250	0	-182.50	Lock
x[m]	-25	0	-12.375	Lock
y[m]	0	25	14.7995	Lock
z[m]	-12.5	12.5	-8.6230	Lock
l[m]	0	25	12	Lock
b[m]	0	25	11.875	Lock
th1[m]	0	10	4.7	Lock
th2[m]	0	10	4.7	Lock
fh[m]	0	25	6.5	Lock
fx[%]	0	100	50	Lock

Figure 7: *edit a house by changing its parameters*

be queried. The parameters e.g. for a house got from a 3D construction program are the scaling factors in the three dimensions (`sx,sy,sz`), the parameters for a generic house with saddle roof are length(`l`), breadth(`b`), cullis heights (`th1,th2`), the ridge's height (`fh`) and position (`fx`). When being placed on a property, additional parameters for a house are position and angle relative to street (`x,y,z,w`). To actually do the check, and to find a legal and reasonable shape and placement for a given house, we construct a fuzzy-logic function $L : \mathbb{R}^n \rightarrow [0,1]$ from all the conditions, which returns 1 if all conditions are fulfilled (details see below). Fuzzy logic operations are the same as the probability computation for an event depending on other events with boolean operations (a and b e.g. transforms to $a \cdot b$). Fuzzy logic has to be used so you have a direction to go to get "more legal": To come to a legal configuration, we run an optimizer which maximizes the fuzzy logic function (see below). If the found maximum is smaller than 1, there is either no legal possibility for this house to be built on this property, or a better starting parameter set has to be provided to reach a global maximum. To keep the parameters from going astray during optimization, the housetypes additionally give a maximum parameter stepwidth for every of their parameters. As the houses are described by meaningful parameters, we used sliders as user interface for editing a house. For every parameter, there is also the possibility to lock it. That means that it will stay unchanged if the user invokes automatic optimization of the house (see figure 7). To give more details for the legality function: L is composed as product ("and") of the fuzzy logic-functions for every of the following conditions: height below limit, base inside allowed window and distance surfaces inside property. Additional conditions for saddle-roof houses are: direction of ridge o.k., roof angle in-



Bedingungen

- ✓ 100% hoehe o.k
- ✓ 98% Grundfl. innerhal
- ✓ 85% Abst.fl. innerhalt
- ✓ 21% hoehen und wink
  - ✓ 100% firstrichtung
  - ✓ 53% dachneigung
  - ✓ 96% traufhoehen
  - ✓ 41% firsthoehe ni
  - ✓ 100% grundflaech
- ✓ 100% Hausform sinnv
  - ✓ 100% first zwisch
  - ✓ 100% dach konve
  - ✓ 100% traufhoehe

OK Abbrechen

Figure 8: *browse the conditions in the application*

side allowed range, height of cullis below limit, and base area small enough. To avoid a house being optimized to zero width, height or breadth, the conditions that house width and breadth should be above 2 meters and the roof convex are added. For the condition "distance areas inside the property" we use the formula $\frac{1-|D\backslash P|\ /\ |D|}{1+|D\backslash P|\ /\ |B|)}$, where $D \subset \mathbb{R}^2$ are the distance surfaces, B the house's base, P the property and $|X|$ the area of a polygon; for the condition "house's base inside allowed area" the formula $1 - |B\backslash A|\ /\ |B|$ where A is the allowed area. For the conditions containing inequalities, we created a special formulation in `C++` to get smooth fuzzy-logic-functions out of them: we convert the equation $x < a$ into the probability $P(x - d\varepsilon < a)$ by writing `x-d*eps<a`, with `eps` being the instance of a special class representing a value ε that has a certain cumulative probability distribution g. With $g = \Theta(x)(1-(x+1)^{-3})$ and d=1 an equation $x < 1$ is then transformed into the fuzzy-logic-function $f(x) = \Theta(1-x)(1-(1-x+1)^{-3})$. To have it C2-smooth at the transition from 1 to < 1 (100% true to not 100% true) it is smoothened by writing `f(x)||f(x)` which is converted into $1-(1-f(x))^2$. To also get the tree of conditions for L together with the values, the conditions are wrapped in a special class `boolobj` that allows to give pointer to a tree, and overloads all boolean operations in fuzzy-logic manner. Together with the macro definition `#define B(x) boolobj(x,#x,tree)`, and the previously described formulation for fuzzy equations, the conditions can be formulated in a natural way. The tree of conditions the additionally allows us to switch off parts of the conditions (see figure 8). To get the error visualizations described in section 2, every house type has a storage for error graphic objects whose filling during evaluation of the legality function L can be switched on. To optimize L if not yet legal (= 1), the following iteration is done:

1. In every iteration, a paraboloid $\vec{x}^T M\vec{x} + b^T\vec{x} + \vec{c}$ is fitted locally to $L(\vec{x})$ by evaluating it at $n(n+1)/2$ points in the parameter space. The points are generated by adding a certain fraction of the parameter stepwidth to the parameters: either two parameters ($\binom{n}{2}$ points), one parameter (n points), or no parameter is changed.
2. if its extremum is higher than the starting point:

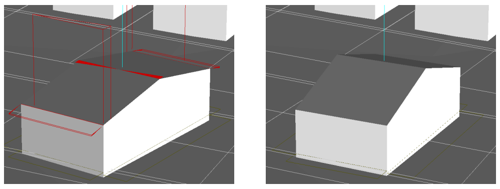


Figure 9: *To the left: a non-legal house. To the right: the house after automatic optimization*

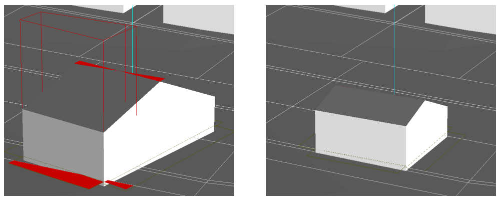


Figure 10: *To the left: a non-legal house. To the right: the house after automatic optimization,with y (the y-coordinate of it's center) locked*

   - if its distance to the old x is lower than a maximum parameter step width,that is the new $\vec{x}$.
   - otherwise, a step is done into that direction limited to the maximum width.
3. Otherwise, a step into the direction of steepest descent is taken.
4. if $L(\vec{x}) >= 1 - \varepsilon$ stop.
5. otherwise, goto 1.

5. Summary

To summarize, we have created an extendible system to visualize land-use-plans and explore possible building forms with the given constraints . An urban planner wrote us: *"B3D is a meaningful tool to support urban planners in their daily work. Not only that it integrates the possibility to represent plans and results in 3D (3D display / 3D printer). It can also support planners during the expensive process of developing a land use plan, whether through better presentations within the participation required by law (citizens and stakeholders) or through showing limitations and alternatives on the fly during the concept phase. Therefore B3D could be considered as a reasonable extension of the CAD-based toolbox for urban planners."* To further improve the system, the building type library can be extended and more constraints could be supported. Moreover, the support of CityGML [Cit12] and an attachment of the system to geographical databases would be desirable. We like to thank Prof. Dirk Bayer for initiating the project, Katrin Kern and Marten Ulpts for a good cooperation and Dr. Sebastian Petsch for his feedback, and the "Stiftung Rheinland-Pfalz für Innovation" for funding.

References

[Cit12] Open Geospatial Consortium: *OGC City Geography Markup Language (CityGML) Encoding Standard*, 2012. URL: `www.opengis.net/spec/citygml/2.0`. 4

[EK03] Everitt C. W., Kilgard M. J.: Practical and robust stenciled shadow volumes for hardware-accelerated rendering. *CoRR cs.GR/0301002* (2003). 2

[För02] Försterling S.: *Plan N*. Master's thesis, Architecture Department TU Kaiserslautern, 2002. URL: `http://www.sachenmacher.de/plan_n/`. 1

[GDS10] Gray A. W., Daniels A. S., Singer D. J.: *Impacts of Fuzzy Logic Modeling for Constraints Optimization*. Tech. rep., American Society of Naval Engineers, 2010. `doi:10.1111/j.1559-3584.2010.00273.x`. 1

[Ker08] Kern K.: Visualizing large sets of nonlinear constraints on highly parametrized 3d objects. In *13th INTERNATIONAL CONFERENCE ON GEOMETRY AND GRAPHICS* (2008). 1

[SHR06] Steinicke F., Hinrichs K. H., Ropinski T.: A hybrid decision support system for 3d city planning. In *Proceedings of International Commission II Symposium (ISPRS)* (Vienna, 2006), pp. 103–108. URL: `http://viscg.uni-muenster.de/publications/2006/SHR06`. 1