Generating Furniture for Augmented Reality Applications using Natural Language

ABSTRACT

In this paper, I present a method for modeling furniture using natural language as a high level description for different furniture parts. Furthermore I show how to use these phrases for the interactive and procedural modeling of 3D furniture objects. Thereafter I will give an introduction outlining how to apply this system to real time augmented reality applications, e.g. in the furniture industry.

CCS Concepts

•**Computing methodologies** → **Mesh geometry models;** *Mixed / augmented reality;*

Keywords

procedural modeling, furniture modeling, natural language procesing, augmented reality

1. INTRODUCTION

With the current development of computer technology, especially augmented reality and 3D graphics, the problem of content generation rises. With a demand of quicker product cycles, better and more interactive ways of product customization, the content generation becomes more and more complex. To aid with augmented reality, real-time content creation needs to be possible and easily usable. This can be achieved in using natural language as an input for the content generation system.

In recent years, procedural modeling as a way of content creation has become a very well established approach, especially in applications which create complex models with many details. Many algorithms exist for the automated generation of landscapes, plants and architecture.

The novelty of this paper lies in the specific generation of 3D furniture objects using natural language, especially for augmented reality applications. The parametrization of the model generator is done using a natural language based approach where a parser parses the language into a top-down structure which is then converted into a tree-like scenegraph. Furthermore I will explain how to use this system in augmented reality applications using cameras with and without depth sensors. The presented framework will be called FURNZ.

DOI:

2. RELATED WORK

As mentioned by [IW13], the field of automated furniture creation is rather unexplored. A more recent study by [STBB14], which seems to be the most current survey on procedural modeling, does not even mention the procedural creation of furniture. In contrast, the automated layout of furniture in rooms, has been researched quite well. For example, [YYT$^+$11] presents a method for placing furniture based on prior knowledge, e.g. the system analyses given examples and builds a spatial relationship, then uses this knowledge to automatically place objects out of a database. [TBSK09] and [TSBK10] on the other hand present methods for a completely procedural indoor arrangement of the given furniture models. The interior design process using augmented reality using depth sensors like the Microsoft Kinect is explained in [TLCT15]. They scan the available space with the depth camera, mark obstacles and allow for the placement of 3D objects in the virtual room. A similar approach with an additional head-mounted display (HMD) and hand tracking can be seen in [PKP16].

For the procedural creation of furniture, the most advanced work available is arguably [KK12]. This paper shows a method for general (and interactive) procedural modeling of interconnected structures with animation that could be seen as a *very* generalized approach to furniture creation, although the special case of furniture models is not mentioned in their paper.

Procedural modeling in general has been researched well in the last decades. For example, the procedural modeling of plants using L-Systems has been researched by [PLH90]. A method for parametric modeling of plants and plant ecosystems has been shown by [DL97] and [DHL$^+$98].

As [Hav05] mentions, the word "‘procedural"’ is very vague, as it is used in too many different contexts. Thus, several forms of procedural modeling have formed. Nowadays, procedural modeling is often a synonym for modeling using production systems using set grammars. For example, L-Systems operate on strings and shape and split grammars operate on shapes. They bothy use a set of rules P to replace an initial symbol Σ with other non-terminal or terminal symbols (N or T). The object is then modeled by recursively applying these rules to the initial symbol Σ. While the procedural creation of plants and architecture - buildings, floor plans, interiour layout, especially using set grammars - are well researched fields, the procedural modeling of furniture is a rather unexplored field.

Procedural modeling of architecture using shape grammars was shown by e.g. [Sti75]. To make them more usable for computer implementation, [WWSR03] simplified shape grammars to set grammars and this was extended by [MWH^{+}06] to CGA shape, a procedural modeling language for architecture. A good introduction and comparison of production systems in general is given by [GS80].

Other forms of procedural modeling use an imperative programming language (e.g. GML, [Hav05]) which is able to create various forms of geometry. Here, the generation of geometry is performed on a very low level, starting with triangles. Although all sorts of geometry can be created, the process is very complex and unsuitable for creating high level objects, especially not with natural language or by novice users. A strong background in computer graphics is needed to use GML in high level applications. To circumvent this, [GK07] proposed a visual language to procedurally model objects using nodes and edges to encapsulate attributes and operations.

One of the first appearances of natural language in a computer graphics context was [Bol80]. He combines speech recognition and position sensing technologies to command simple shapes on a large screen display using a set of simple sentences and commands. [LBW05] introduces the concept of a "Knowledge Representation layer", using entities, attributes and relations to describe the "knowledge" (objects, their visual and non-visual attributes, their relationships, modeling operations etc.) of the system using a semantic representation. [Mil14] uses natural language to specify the parameters of a music synthesizer. Although not computer graphics, the basic concepts of parametrizing an entity with a set of attributes is comparable to my method. He uses a loose phrase structure grammar to be able to describe the sound (filters,...) of a synthesizer using words like "'fat"', "'soft"' etc.

[Bun16] proposed a system to parametrize the reflectance distribution functions of a surface using natural language. The language part is not that different from the methods described here, although the language processing methods found there are used in a completely different domain and attributes for a different semantic structure. The method can be used in combination with the proposed method in this paper to not just generate the furniture object using natural language but also describe the desired surface characteristics to gain a full augmented experience.

3. FURNITURE DESIGN USING NATURAL LANGUAGE

3.1 Introduction

Furniture can be distinguished into a set of four categories (see e.g. [Böt05]), furniture for seating (chairs etc.), furniture for storage (cupboards, wardrobes, etc.), furniture for lying (sofas, beds, etc.) and tables. Within those categories, the furniture have a more or less similar design that follows their function (e.g. one would not find a vertical bed very comfortable) and the furniture parts themselves can be described in a tree-like structure (see [Bei15]).

This concept can be used to describe a basic set of rules to generate a certain furniture object in these four categories - optional features would be doors or a back wall - and can be described procedurally. The word "'procedurally"' has to be taken quite literally here, as the base parts are modeled using classes as known from object orientated programming. These classes represent methods ("'procedures"') which generate certain furniture objects based on given attributes. Using standard OOP-concepts, complex objects can be derived from simpler classes. For example, a base class can describe a common rack. Attributes will be width, height, depth and number of shelves. The individual shelf height is then calculated based on the rack's total height and the shelves distributed equally. One derived class might allow to parametrize the individual shelve height, another derived class could add an even number of doors.

The base parts are highly attributed and linked together in a tree-like structure. For example, a rack consists of at least one shelf (attribute: width, depth, thickness), up to two side walls (height, thickness, depth depends on shelf), zero or one back wall, zero or one stand. A chair on the other hand can be described by zero to several stands, a seating area and zero or two armrests and so on.

3.2 Natural Language

3.2.1 User Studies

To find a proper system how people use natural language to describe furniture, two user studies were conducted. In the first study, the test candidates were shown eleven different pictures of randomly chosen furniture (Bookshelf, Chair, Bed, Armchair, Table, Rack, Bed, Rack, Table, Office chair, Couch, see figure 1). The task was to then describe the shown furniture objects using short notes or sentences.

The average age of the 17 participants was 28.2 years. Three male and 14 female participants with no background in computer science took part in the study. The participants used an average of 3.4 notes per furniture object to describe it (with $\sigma = 1.46$).

One thesis statement which was to investigate here is, that people tend to use a top$\rightarrow$down approach, when describing furniture objects, e.g. first name the furniture category, then describe main features, then details. When carefully checking the filled-out questionnaires, this approach was chosen in 82% of the cases ($\sigma = 0.38$).

After an in-depth analysis, the general structure which was chosen by the participants to describe a furniture object was [furniture category] followed by [material], [main feature] followed by [details]. In 46% of the cases, the material was used after the main feature was described, so these two points should be considered interchangeable. This analysis helps to build a general structure for the grammar parser and information on how the user is supposed to enter the natural language description into the system. The found structure here is actually based on how a majority of people would describe a furniture object, so the user does not have to adjust (much) when using the system.

Another interesting fact is, that users tend to describe the furniture in a more or less constant amount of notes. Although the difference between the participants is quite big (between one and eight notes for the same object) with σ for each individual object ranging between 1.8 and 2.0, the individual changes, how many notes one participant used to

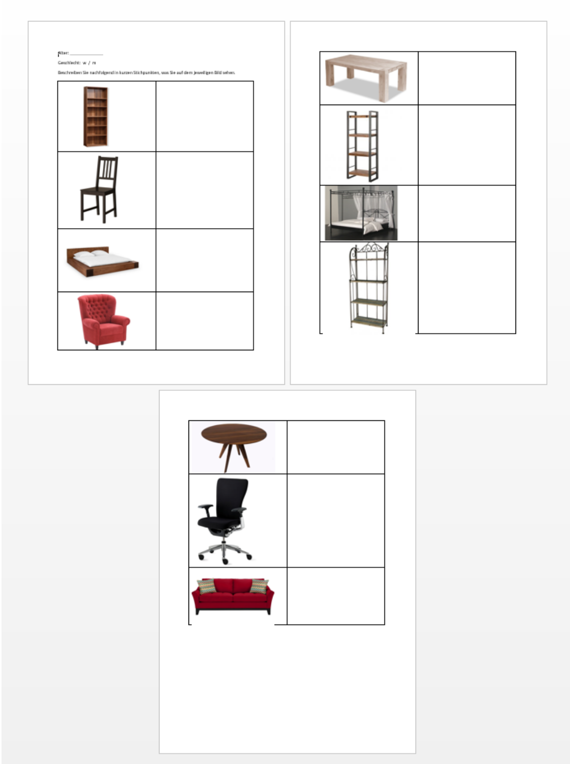

Figure 1: Questionaire

describe different furniture is rather small with σ between 0.3 and 1.1.

The second user study used a completely different method. The group of 18 participants (14 female and 4 male), all different from the participants taking part in the first study, with an average age of 22.1 years, were given the task to pick two items from the following list:

- furniture for sitting
- furniture for lying
- furniture to put things onto
- wardrobes, cupboards, closets, locker

After having chosen their two items, they were given the task to describe a piece of furniture in a way that a designer could draw it. It should be noted, that the study was conducted in the German language and that all the words in list item 4 (wardrobes, cupboards, closets, locker) are described with one German word - ”‘Schränke”’.

Category 1 was chosen 8 times, category 2 was chosen 12 times, category 3 and category 4 were chosen 8 times. In general, the participants used more notes (7.47 with $\sigma = 2.59$, $min = 2$ notes, $max = 12$ notes) to describe the individual furniture objects than in the first study.

In this study, 83% used a top-down approach, meaning the participants first named the subtype of furniture (e.g. a chair, a bed, a wardrobe) and then went further into detail. 77% (14 out of 18) used to name the material and color (for the parsing of the material part, I refer to [Bun16]) immediately after they named the subcategory. The others named the main feature or features first. After this, the structure of the furniture was described (56%) from a coarse to a more detailed view usually (80% of the 56%) starting with the total measurements of the object. Interestingly, none of the participants in the first study provided any kind of measurement, not even adjectives like ”‘big”’, ”‘small”’ etc. After this, the details were described (ornaments, decorations, special designs like painted pictures on the object, etc.)

3.2.2 A linguistic model

The main goal of the user studies was to find a linguistic model which people generally use to describe furniture in day-to-day life and which will found a base for the language parsing. The studies showed that most people (over 80%, 82% in the first study, 83%) in the second study) that people generally use a top-down approach to describe the structure of the object, Top-down in this case describes the way to describe the object from the general structure to finer details and naming the type of furniture (the subcategory, according to [Böt05]) first. To ease the software development, especially the language parser, in the current software version the model presented here has been used (see chapter 7).

As mentioned in the last chapter, the structure found for describing furniture is:

[type/subcategory] [main feature] [material] [measurements] [sub-features] [visual details]

where everything but the type/subcategory is optional. This allows for an efficient parsing of the linguistic input into a top-down structure. An example would be: *”‘A rack with five shelves, made from oak wood, 2 meters tall, 80 centimeters wide, with a back wall”’* which then will be parsed into an object (in the sense of object-oriented programming) representing the given sentence and its attributes.

- Rack
- 5 shelves
- back wall present
- Size: $2.1m \times 0.8m$
- Material: oak wood
- Automatically set attributes
  - board thickness:2cm
  - shelf height 33cm
  - no feet
  - ...

3.3 Language Processing

The basic vocabulary was found by analyzing corresponding literature (e.g. citeboeth05) and business databases gratefully provided by some well known German furniture companies. This gives a good starting point as the vocabulary is based on commonly used wording in the furniture industry. One of the main issues is that the vocabulary used in the furniture industry can be highly complex and different terms with the same meaning in every day language could be used. [Mil14] showed that the number of adjectives to describe (in his case) sound rises quickly and becomes quite diverse when taken from user experience or user interviews. This shows that the unrestricted usage of “every day language” is not feasible and that an approach using controlled natural language ([Kuh14]) which reduces the the vocabulary to a predefined subset should be used. After choosing a basic set of nouns, adjectives and adverbs, several services can be used to extend the vocabulary such as the Wortschatz (wortschatz.uni-leipzig.de). The Wortschatz provides a web interface for finding synonyms (related meaning) and antonyms (opposite meaning) in the German language.

The language processing part of the application is based on a self-developed part-of-speech tagger which uses a lexicon of the German language to annotate the provided words. While annotating the words, the word is transformed into its base form using the Wortschatz-Project (see Chapter 3.2) to allow easier processing.

In the second step, the inflection (especially if working with the German language) of the words needs to be analyzed and the base form has to be found. This is done using a simple baseform lexicon.

The biggest problem with the specification of attributes using natural language is ”‘intent vs. interpretation”’. This means, that the user might intend something different with his input than the framework ”‘interprets”’ by parsing and analyzing it. For example, when asked about the color ”‘blue”’ people might think of colors ranging from ”‘light blue”’ to ”navy blue”’, all being valid shades of ”‘blue”’. Since the earliest speech processing systems in computer graphics ([Bol80]), this remains a more or less unresolvable issue, as different people have a different personal definitions of the meaning behind descriptive words (e.g. adjectives like ”‘blue”’, ”‘tall”’, ”‘slim”’). To resolve this issue, the user can provide further modifiers (*more, less, wider, taller, thinner*; e.g. “more shelves”, “thicker boards”) in several refining steps. This frees the user from specifying discrete values for parameters, as this might contradict the intention of this system. Using the further refinements, the system still has enough potential to find the user’s intented attribute although specifying numerical values is also possible.

When the user now provides a natural language input, the input is parsed into an object representing the furniture. The attributes of the furniture the user does not provide are set to default values. As mentioned, the user then can refine the object by providing further input. For example, the simplest input could be ”‘a rack”’. This would result in a small rack with five shelves using a standard material. The variable attributes for width, height, number of shelves, presence of a back wall, presence of foot stands, size of the stands etc can then be provided in a further refining step. Also, set attributes can be modified using adjectives such as ”‘wider”’, ”‘taller”’, ”‘thicker”’ ”‘thinner”’ etc. This leads to two different input methods. The first one would be to specify the whole object parameters in one sentence (”‘A wide rack made of oak wood with seven shelves on small foot stands without a back wall”’) or in several steps which refine the last given steps (”‘a rack”’ ”‘wider”’ ”‘oak wood”’ ”‘seven shelves”’ ”‘small foot stands”’ ”‘no back wall”’). Between each step, the user is presented with an image of the furniture after each step.

To generate more complex features (”‘A TV bench with two racks on top”’), these have to be assembled from basic parts. Here, the parser checks if further furniture types were named after certain stop-words (”‘and”’, ”‘with”’) and localization phrases (”‘on top”’, ”‘next to”’, ”‘left of”’). Thus, the given example ”‘A TV bench with two racks on top”’ will result in the two racks sitting on top of the bench. Everything between the stop-word in front of the furniture type (”‘rack”’) and the localization phrase will be considered part of the ”‘rack”’, not the TV bench. It might be seen as a limitation to the system to shrink the natural language grammar to such phrases, but otherwise the possible ambiguities can not be properly resolved. On the other hand, these limitations still allow a very natural language ”‘feeling”’.

The parser will generate a tree-like structure of the furniture parts, setting position attributes where necessary (e.g. the two racks on the bench obviously need to have their base on top of the bench to avoid geometry overlaps in the rendering). These position attributes will directly influence the transformation matrix of the resulting geometry node in the furniture’s scene graph.

3.4 Generating Geometry

3.4.1 Commercial Data

The geometry of the object is provided by different sources. The first and most interesting source are the databases used by furniture vendors. These provide the 3D geometry of single furniture parts alongside with business data and, for the scope of this paper important, logic data. These logic data elements describe how certain furniture parts can be assembled by adding variable attributes to certain geometric properties, e.g. a rack could have the attributes corpus-height and corpus-width as float values, the number of shelves as integer values and the presence of a back wall as boolean values (besides having material attributes). This allows for a dynamic (and thus procedural) generation of the rack object with arbitrary parameters. As furniture for the mass market do not come in ”‘any width”’, the attributes can be limited to certain ranges or discrete values. If the user specifies an attribute not matching those limits, a feedback method selects the closest available value and provides a warning message.

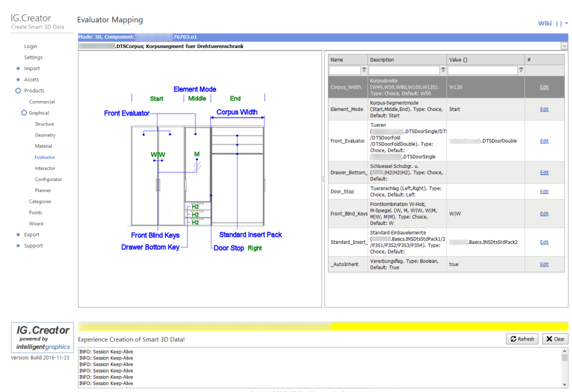

Figure 2: IG.Creator Software

3.4.2 Shape Grammars

Another way of specifying geometric objects is through shape grammars. For furniture, shape grammars like CGA shape (see [MWH$^+$06]) provide a good way to generate the 3D geometry. Shape grammars are a specific class of production systems that generate geometric shapes. The grammar consists of a set of rules which transform non-terminal symbols into other non-terminal or terminal symbols, beginning with a initial (non-terminal) symbol. The rule hereby describes how an existing non-terminal symbol (representing a shape or parts of a shape) can be transformed in the geometric space. A generating engine then applies these rules recursively to the initial symbol. The shape grammar can be extended using parameters so the output of a rule is also dependent on those.

Although shape grammars are a very well researched field, there seem to be no publications or software tools using them for the generation of furniture objects. As mentioned in chapter 2, [IW13] stated that the whole field of procedural furniture creation in general is rather unexplored.

For this project, a very basic shape grammar was created, loosely based on CGA shape to generate furniture parts. For example, most furniture objects can be described using several transformed (translated, rotated and scaled) boxes (representing wooden boards) of different sizes. Other furniture parts can be described using basic volumetric shapes like cylinders, spheres, boxes with round corners etc. Also FFD (free-form deformation, [SP86]) can be applied to these objects to allow more freedoms designing certain shapes.

When used with the correct materials to render (e.g. [Bun16]), photo-realistic images are possible. A production has the form[1]:

```
ruleNo: predecessor [(condition)] => [{spatial location}]
        successors[(parameter)] [:probability];
```

For example, a simple rack can be described with

```
1: Init => outline;
2: outline => Box(5,150,30) center Box(5,150,30);
3: center => REPEAT(Y,5,30){Box(30,5,30)};
```

Instead of the provided numerical values, attributes can be used which then will be parametrized using the natural language parser. The grammar and the parser are implemented in C++ using the Boost::Spirit::Qi library. This allows to predefine basic furniture, furniture parts and design items such as ornaments.

4. PUTTING IT ALL TOGETHER

After extracting the attributes and parameters from the natural language input, the geometry generator needs to generate the specified 3D objects. The language parser parses the natural language input into an oop-style class. This class contains attributes for the specific furniture type. As an example, the class definition for racks are given.

```
class BasicFurniture
{
    float width;
    float height;
    float depth;
                // etc.
};
class Rack: BasicFurniture
{
    int num_shelves;
    bool back_present;
    bool left_present;
    bool right_present;
    bool feet_present;
    MATERIAL material;
                // etc.
};
```

For each furniture type, a base class is defined. This means, there is a class for chairs, beds, tables, armchairs, benches, sofas, etc. Each class provides attributes for the basic features of the desired furniture object, e.g. the Rack-class provides `num_shelves`, whereby a Wardrobe-class would provide attributes for drawers and hangers (`int num_drawers` and `bool hanger_present`). To create a new type of furniture, one just need to derive from either the abstract class `BasicFurniture` or, if the new type is a subtype of an already existent class, derive from this class. At the moment, this is implemented in C++ and needs to be compiled. The extension of the framework using cling (see [VCNR12]), a C++ interpreter built on top of Clang and the LLVM compiler infrastructure is currently in the planning stage.

The class also provides the methods to generate the furniture geometry data based on the set attributes. As mentioned, if an attribute is not specified by the user, a default value has to be used. To apply this on the basic rack example (see figure 5), the generator would look like

```
NODE Rack::Generate()
{
  NODE result;
  float shelv_height = height/num_shelves;
  // left board
  MATRIX m;
  m.SetScale(board_thickness, height, depth);
  NODE n;
  n.Name("Left Board");
  n.SetGeometry(BasicCube);
  n.SetTranslation(m);
  for (int i = 0; i < num_shelves+2; i++)
  {
    Node child_n;
    float y = i * shelv_height;
    Matrix child_m;
    child_m.SetScale(width, board_thickness, depth);
    child_m.SetTranslate(0.0, y, 0.0);
    child_n.Name("Shelve " + i);
    child_n.setGeometry(BasicCube);
    child_n.SetTranslation(m);
    n.Add(child_n);
  }
  // Add right side, back wall,
  // etc.
  result.Add(n);
}
```

As shown, the generator is fully parametrized. Instead of using `BasicCube`, geometric data for the basic furniture parts directly from the manufacturers database could be used as well. This allows to use completely dynamic object descriptions for individual furniture types which are parametrized by the natural language processor.

To allow for more complex furniture (see chapter 3.3), the object will be represented in a screnegraph-like structure, where a node represents a basic furniture object (e.g. a rack) and contains the geometry (vertices, normals, texture coordinates, indices, etc) and the $4x4$ transformation matrix of the object as well as material parameters.

5. AUGMENTED REALITY

The augmented reality part in the application is implemented using C++, OpenGL and ARToolkit. For the camera-only version, two models exist. One uses AR-Markers like the HIRO marker which will be placed on the floor. In the AR Application, this marker will be used to place the furniture .

Another method, which also works with "'just"' a camera uses edge-detection filters (the Sobel kernel for example) to detect the edges of a wall and approximate its size. This will then be used to place the generated furniture.

To use specific measurements of the surrounding geometry in the augmented environment, depth cameras like the Microsoft Kinect can be used to arrange the furniture in an augmented environment. The arrangement of the furniture is shown in [PKP16], so this framework (FURNZ) can be used in conjunction with the application presented in their

[1]condition and probability are optional

Figure 3: ”‘A small Rack with 4 shelves made from dark oak wood’”

paper.

An important fact for photo realistic rendering, especially in augmented reality applications, is the real-time rendering of materials, their BRDF and the lighting environment. As this is a difficult matter in itself, a quick introduction on how the rendering in the framework presented in this paper was achieved will be given. As natural language is used to describe the furniture object, it was necessary to look for an approach to describe the furniture's material using natural language, suited for real time rendering as well. This approach was found in [Bun16] and is heavily used here. Together, they allowed for a powerful way to describe furniture and its associated materials.

For the reconstruction of the lighting environment, I refer to [Deb98] and [Deb02]. Image-based lighting and physically based rendering are the best approaches to achieve photorealistic rendering in this context. The implementation is already in a planning stage.

6. RESULTS

The results show the furniture objects for “A small Rack with 4 shelves made of dark wood” and “A small table made of wood” rendered in an augmented reality application using ARToolKit5 and a HD Webcam (Microsoft Lifecam HD-3000). The rendering is in real time with 50 frames per second (the framerate is limited by the camera). The tracking is done using the Hiro-Pattern. The language processing takes less than a second on standard PC hardware (e.g. Quadcore Intel i7, 3GHz, nVidia Geforce GTX960).

All in all the application is fully suitable for real time rendering and evaluation. Even if the base form transformation is fully used (not cached), the results will be available within a few seconds. First tests on mobile platforms (Samsung Galaxy S5 + S6) using the Android operation system and Unity as a 3D engine with the Vuforia plugin for augmented reality have also shown very good results.

Figure 4: ”‘A small wooden table’”

The whole framework allows for the easy generation of (basic) furniture for augmented reality application, quickly helping users to get a first impression of how certain furniture pieces will look in a specific environment. The user does not need to interact with the system in any ”‘complicated’” way, the furniture is created simply by naming the type and some parameters. Combined with the language processing unit, the whole system is fast enough to be used in a real-time augmented reality environment.

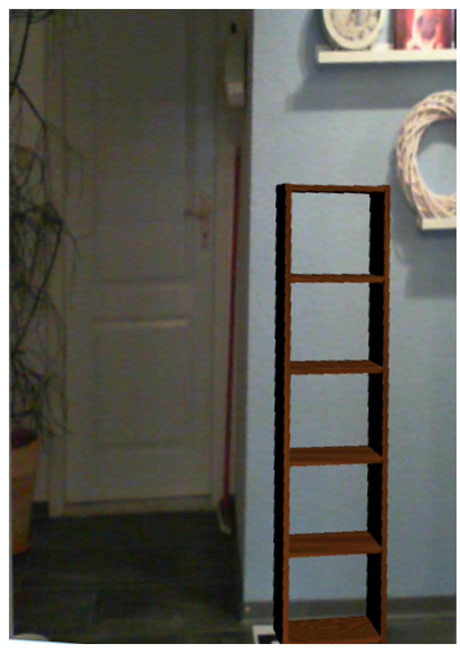

Figure 5: ”‘A small Rack with 4 shelves made from wood’” in an AR Application

7. FURTHER RESEARCH

At the moment, only basic furniture objects can be created. The plan is to link the system to the (already parametrized) databases of some furniture fabricators, a project that is currently implemented with two big German furniture companies. Furthermore, the problem of the limited grammar has to be addressed to allow a completely natural description of the furniture objects. Although this requires more and more detailed user studies on the question on how people would normally describe furniture objects to other people.

8. REFERENCES

[Bei15] Ekkehard Beier. OFML - Standardisiertes Datenbeschreibungsformat der Büromöbelindustrie. *Verband Büro-, Sitz- und Objektmöbel e.V.*, 3(2):1–242, 2015.

[Bol80] Richard a Bolt. Put-that-there. *Proceedings of the 7th annual conference on Computer graphics and interactive techniques - SIGGRAPH '80*, pages 262–270, 1980.

[Böt05] G. Böth. *Möbel: eine Typologie für Museen und Sammlungen*. Materialien aus dem Westfälischen Museumsamt. Deutscher Kunstverlag, 2005.

[Bun16] Sebastian Buntin. Generating Materials for Augmented Reality Applications using Natural Language. In *Proceedings of the 7th Augmented Human International Conference 2016 on - AH '16*, pages 1–7, New York, New York, USA, 2016. ACM Press.

[Deb98] Paul Debevec. Rendering synthetic objects into real scenes. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques - SIGGRAPH '98*, pages 189–198, New York, New York, USA, 1998. ACM Press.

[Deb02] Paul Debevec. Image-based lighting. *IEEE Computer Graphics and Applications*, 22(2):26–34, mar 2002.

[DHL+98] Oliver Deussen, Pat Hanrahan, Bernd Lintermann, Radomír Měch, Matt Pharr, and Przemyslaw Prusinkiewicz. Realistic modeling and rendering of plant ecosystems. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 275–286, New York, NY, USA, 1998. ACM.

[DL97] Oliver Deussen and Bernd Lintermann. A modelling method and user interface for creating plants. In *Proceedings of the Conference on Graphics Interface '97*, pages 189–197, Toronto, Ont., Canada, Canada, 1997. Canadian Information Processing Society.

[GK07] Björn Ganster and Reinhard Klein. An Integrated Framework for Procedural Modeling. *Spring Conference on Computer Graphics 2007 (SCCG 2007)*, pages 150–157, 2007.

[GS80] J Gips and G Stiny. Production Systems and Grammars: a uniform characterization. *Environment and Planning B: Planning and Design*, 7(4):399–408, 1980.

[Hav05] Sven Havemann. Generative mesh modeling. *PhD Thesis, Technische Universitaet Braunschweig, Germany*, 1(1-303):4, 2005.

[IW13] M Ilcík and M Wimmer. Challenges and Ideas in Procedural Modeling of Interiors. *Eurographics Workshop on Urban Data Modelling and Visualization (2013)*, pages 1–2, 2013.

[KK12] Lars Krecklau and Leif Kobbelt. Interactive modeling by procedural high-level primitives. *Computers and Graphics (Pergamon)*, 36(5):376–386, 2012.

[Kuh14] Tobias Kuhn. A survey and classification of controlled natural languages. *Computational Linguistics*, 40(1):121–170, March 2014.

[LBW05] Marc Erich Latoschik, Peter Biermann, and Ipke Wachsmuth. Knowledge in the Loop: Semantics Representation for Multimodal Simulative Environments. *Proceedings of the 5th International Symposium on Smart Graphics 2005*, pages 25–39, 2005.

[Mil14] Jan-torsten Milde. Using natural language to specify sound parameters. *Proceedings of the ICMC-SMC*, pages 1627–1630, September 2014.

[MWH+06] Pascal Müller, Peter Wonka, Simon Haegler, Andreas Ulmer, and Luc Van Gool. Procedural modeling of buildings. *ACM Transactions on Graphics*, 25(3):614, 2006.

[PKP16] Jinwoo Park, Sung Sil Kim, and Hyerim Park. Ş DreamHouse Ť: NUI -based Photo-realistic AR Authoring System for Interior Design. *Proceedings of the 7th Augmented Human International Conference 2016*, 2016.

[PLH90] P. Prusinkiewicz, A. Lindenmayer, and J. Hanan. *The algorithmic beauty of plants*. Virtual laboratory. Springer-Verlag, 1990.

[SP86] Thomas W. Sederberg and Scott R. Parry. Free-form deformation of solid geometric models. *ACM SIGGRAPH Computer Graphics*, 20(4):151–160, aug 1986.

[STBB14] Ruben M. Smelik, Tim Tutenel, Rafael Bidarra, and Bedrich Benes. A survey on procedural modelling for virtual worlds. *Computer Graphics Forum*, 33(6):31–50, 2014.

[Sti75] George Stiny. *Pictorial and formal aspects of shape and shape grammars*. ISR, Interdisciplinary systems research. Birkhäuser, Basel, Stuttgart, 1975. Originally published as the author's thesis, University of California, Los Angeles.

[TBSK09] Tim Tutenel, Rafael Bidarra, Ruben M Smelik, and Klaas Jan De Kraker. Rule-based layout solving and its application to procedural interior generation *. In *3AMIGAS: Proceedings of the CASA 2009 Workshop on 3D Advanced Media in Gaming and Simulation*, pages 15–24, 2009.

[TLCT15] Jeff K T Tang, Wan Man Lau, Kwun Kit Chan, and Kwok Ho To. AR interior designer: Automatic furniture arrangement using spatial and functional relationships. *Proceedings of the 2014 International Conference on Virtual Systems and Multimedia, VSMM 2014*, pages 345–352, 2015.

[TSBK10] Tim Tutenel, Ruben M Smelik, Rafael

Bidarra, and Klaas Jan De Kraker. A Semantic Scene Description Language for Procedural Layout Solving Problems. In *AIIDE '10: Proceedings of the 6th Conference on Artificial Intelligence and Interactive Digital Entertainmen*, pages 95–100, 2010.

[VCNR12] V Vasilev, Ph Canal, A Naumann, and P Russo. Cling - The New Interactive Interpreter for ROOT 6. *Journal of Physics: Conference Series*, 396(5):052071, dec 2012.

[WWSR03] Peter Wonka, Michael Wimmer, François Sillion, and William Ribarsky. Instant Architecture. *Instant Architecture. ACM Transactions on Graphics*, 22(3):669, 2003.

[YYT+11] Lap-Fai Yu, Sai-Kit Yeung, Chi-Keung Tang, Demetri Terzopoulos, Tony F. Chan, and Stanley J. Osher. Make it home. *ACM Transactions on Graphics*, 30(212):1, 2011.