Attributed Collage Grammars: A Rule-Based Modeling Framework

Carolina von Totth
University of Bremen, Department of Computer Science
Linzer Strasse 9A, 28359 Bremen, Germany
caro@tzi.de

ABSTRACT

The strength of procedural modeling methods in general and grammar-based modeling in particular lies in their ability to generate visually complex objects and scenes from a comparatively small and simple description. This paper presents attributed collage grammars, a general and versatile modeling method for the generation of scenes. Tree grammars are employed to generate a syntactical description of objects in the form of trees, which are then assigned an actual geometric meaning by an algebra. Complex scenes and objects with a level of detail that would be tedious or impossible to reproduce manually can be constructed from a number of user-defined components which are replicated or transformed by manipulating their attributes through the iterative application of rules.

Keywords

procedural modeling, grammar-based modeling, Greek architecture, tree grammar, algebra, data amplification

1. INTRODUCTION

The construction of any convincing virtual environment, be it photo-realistic or not, is always a difficult task. The reason lies in the complexity of natural phenomena, and although all manner of modeling methods have been devised in order to handle such visual complexity (e.g. particle systems for simulating dynamic processes [RB85], L-systems with an extensive application in plant generation [PL96, PMKL01] as well as many procedural algorithms with their roots in fractal geometry [EMP+98], to name but a few), there is no general solution that is uniquely suitable to all applications.

So far grammar-based methods, L-systems in particular, have been primarily used in the modeling of plants and more recently modern cities [PM01].

This paper presents a new method that is not bound to a specific modeling purpose. Rather, it provides a flexible grammar-based scene generation framework that can be customized to suit multiple modeling goals, from architecture to jewelry design. Attributed collage grammars build upon collage grammars as presented by Drewes and Kreowski in [DK99], using and extending the tree-based version specified in [Dre00] and [Dre04].

In the diploma thesis[1] by Thomas Meyer and the author a restricted implementation has been used to model languages of buildings based on the principles governing ancient Greek architecture (Figure 1).

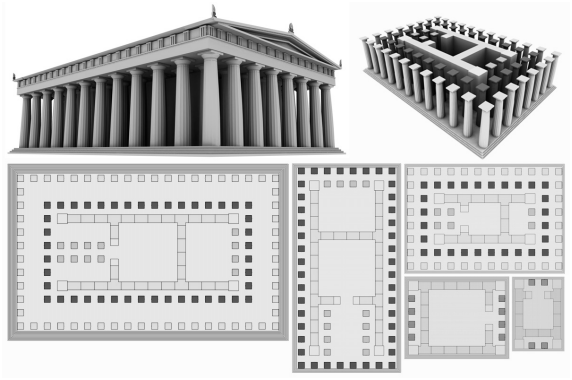
The paper is structured as follows: in Section 2 an overview of attributed collage grammars is given. Section 3 gives solutions to two problems which were previously not solvable using context-free collage grammars; one of these is illustrated by means of an example. Section 4 concludes the paper with an outlook on future work.

SHORT papers proceedings, ISBN 80-903100-9-5
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.

2. ATTRIBUTED COLLAGE GRAMMARS

Most grammar-based methods of image or geometry generation, like chain code grammars, turtle grammars and L-systems, originated from the string-rewriting formalisms of formal languages. Such grammars generate strings that require the additional step of a pictorial interpretation to become images. In their original form [DK99], collage grammars were an attempt to work spatially from the beginning, bypassing the

[1] http://www.tzi.de/~ariel/GreekArchitectureWithCollages.pdf



(a) Sample layouts of temples generated with a single attributed collage grammar

(b) A peripteral and a dipteral temple generated with the same grammar

Figure 1: Models of Greek temples generated with attributed collage grammars

string rewriting mechanism by replacing the nonterminal symbols of formal languages with labelled spatial placeholders, called *hyperedges*, and terminal components with actual point sets in $\mathbb{R}^d$, called *parts*. A clustering of hyperedges and parts is called a *collage* and provides a spatial counterpart to the concatenation operation used to group line drawings.

The rules of a collage grammar specify how hyperedges and parts can be altered through the application of affine transformations, and are applied iteratively starting from an initial collage. For collage grammars in $\mathbb{R}^2$ this already allows generating well-known fractals like the Barnsley fern or the Koch snowflake curve. In fact the image generating power of collage grammars in this form already exceeds that of iterated function systems.

The method presented here builds upon the LEGO™-like idea of collage grammars, using it as a basis for a much more general approach. Attributed collage grammars are described using a tree-based formalism, once again separating the syntax of the generated structures from their semantics; this is mainly done in order to achieve better modularity.

In tree-based scene generation, objects or scenes are represented as trees over scene components (leaf nodes) and operations on these (internal nodes). Such trees are generated as purely syntactic constructs over a signature by a tree grammar, and evaluated by an algebra which provides the nodes of the tree with the proper semantics. This approach is highly modular, since different algebras can be paired with different types of grammars.

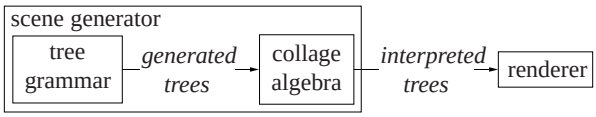


Figure 2: The scene generator principle

Attributed collage grammars are *scene generators* (Figures 2 and 3), consisting of a *tree grammar* and an *attributed collage algebra*. Therefore, in trees produced by an attributed collage grammar, the leaf nodes are the collages, which have retained their meaning as a grouping of parts, while the operations serve as a grouping mechanism for internal transformations.

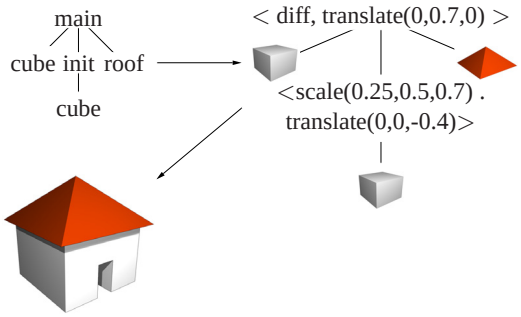


Figure 3: Scene generation: an example

In the small example displayed in Figure 3 a very simple house is built the following way:

- First a syntax tree is generated by some grammar: its root is an operation called `main` which is applied to subtrees `cube`, `init[cube]` and `roof`. At this stage we have no idea what `main` does — or what `init`, `cube` and `roof` are, for that matter.
- In the next step the tree has been interpreted by some attributed collage algebra. The middle cube is distorted by a scale transformation and moved a little, and then it is subtracted from the first cube using `diff`, which is a Boolean difference transformation that takes two arguments. Finally the roof is moved up a little: the result is a small cartoony house.

In attributed collage grammars, the parts are extended to include not only simple point sets, but also any other type of scene component available, among them splines, particle emitters and light sources. All aspects of an object that should be modifiable are encoded into *attributes*, of which each part can have an arbitrary number. Examples for such attributes are

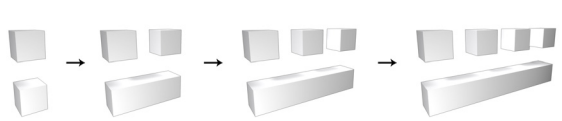


Figure 4: An example of linear growth

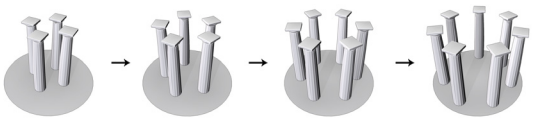


Figure 5: Circular expansion of columns

the object's transformation matrix, its local coordinate system, or its materials. For a spline or a parametric surface the control points are additional attributes; for light sources anything from the light type to parameters like brightness and shadow type are attributes as well.
Transformations operate on attributes only, leaving the underlying part definition unchanged. The transformation concept is no longer restricted to only affine transformations, but has been extended to include every possible form of user-defined action, from random distribution algorithms to Boolean or color-changing operations on components.

3. APPLICATIONS

Attributed collage grammars have a multitude of applications. Since the attribute and transformation system allows practically unlimited extensions to both the set of transformations and the objects that are considered, many modeling problems can be solved by either the addition of new attributes or new operations on existing attributes.

3.1 Linear growth

One problem that needs to be addressed when modeling in a recursive way with affine transformations is the restriction to exponential growth through repeated scaling operations (see [DKL02]). This problem also occurs in the construction of temples, e.g. where the floor and roof have to grow to match the addition of new columns. The occuring linear increase in area size cannot be matched by the recursive application of affine scaling.
We have solved this problem by introducing variables, which are used to alter the transformation parameters as needed during the derivation process.

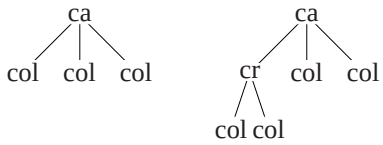
3.2 Recursive circular arrangement

A spatial function with many uses is one that arranges its arguments in a circular fashion around some central point. This can be done with benches in an amphitheatre, columns in a round temple or support beams in a pavillion, the petals of a flower, etc. When such a function is used recursively however, as is the case within the derivation process of a grammar, one must be a little more careful than when the same type of function is used just once on a fixed number of arguments.

Example 1
The following is just the rule section of a *regular tree grammar* [Dre00] where `S` is the start symbol, `C` is another nonterminal, `ca` is an operation that contains exactly one circular arrange transformation, `cr` changes the value of the global radius variable `r`, and `col` is a collage with the shape of a column. A section of the generated language (displayed on a circular base) is shown in Figure 5.

```
S -> ca[C, col, col],
C -> cr[col, C],
C -> col
```

Derivations are very straightforward, with the first two trees generated by the grammar looking like this:



And this is the relevant part of the algebra:

```
r = gl_var(2),

ca           = circularArrange(~r),
changeRadius = assign(~r=~r+1),

ca = < ca >,
cr = < changeRadius, id>
```

The very simple version of a circular arrangement transformation used here has only one parameter: the radius of the desired circle, $r \in \mathbb{R}$. It works the following way:

1. For argument collages $C_1, \ldots, C_k$ and a radius r, *circularArrange* takes the concatenation $\bigcup_{i=1}^{k}(C_i) = p_1 \cdots p_n$ of the parts p_i contained in the given collages C_i. (This way, the sequence of parts is maintained.)

2. Then all parts p_i are translated to the center of the coordinate system, where

3. every part p_i is first translated $-r$ along the z-axis and then rotated $\frac{360}{n} \cdot i$ degrees about the y-axis.

It might be interesting to note that the same circular arrangement of objects can also be achieved using only variables, but with considerably more effort.

△

Figure 6: Model of a Greek tholos generated with attributed collage grammars

4. CONCLUSION

In this paper, attributed collage grammars have been introduced as a framework for the grammatical modeling of scenes that is geared towards extension and customization by the user.

Future work will include specifying and implementing a constraint architecture that allows further influencing the selection and application of rules based on constraints specified by the user, and also automatically postprocessing the generated scene. Examples for such constraints are collision control and handling during the growth of a scene, or synchronization of rule application. Spatial coherence in the generated structures might also be enforced by constraints.

Another topic for future work is a referencing system for scene generators where grammars can access the yield of other scene generators as terminal symbols in their rules. This would allow modularizing the generation of scene components and therefore achieving a greater variation of results. This concept together with a sample implementation and possible applications were tentatively explored in the author's diploma thesis, where a first attempt to model the growth of an ancient settlement has been made using both collision constraints and grammar hierarchies.

Apart from further study of the method itself, attributed collage grammars will be applied in a number of case studies. In this context, the generation of ancient Greek architecture will continue to be studied with an emphasis on creating more detailed buildings as well as complete settlements. Grammar libraries will be developed and made available together with an implementation. Another interesting topic for such a case study is the domestic and religious Japanese architecture starting with the Heian period.

An implementation of tree generators, including a wide range of two-dimensional picture generation methods is provided with the software TREEBAG [DK01]. TREEBAG is written and maintained by Frank Drewes and has already been used as a framework for our first prototypical implementation of attributed collage grammars. In the future it will be extended to include a full implementation of a complex attributed collage algebra; it will also acquire plugin functionality for such commercial modeling packages as *Cinema 4d* and *Maya*.

5. REFERENCES

[DK99] Drewes F. and Kreowski H.-J. Picture generation by collage grammars. In Ehrig H., Engels G., Kreowski H.-J., and Rozenberg G., editors, *Handbook of Graph Grammars and Computing by Graph Transformation*, volume 2, chapter 11, pages 397–457. World Scientific, 1999.

[DK01] Drewes F. and Klempien-Hinrichs R. TREEBAG. In Yu S. and Paun A., editors, *Proc. 5th Intl. Conference on Implementation and Application of Automata (CIAA 2000)*, volume 2088 of *Lecture Notes in Computer Science*, pages 329–330, 2001.

[DKL02] Drewes F., Kreowski H.-J., and Lapoire D. Criteria to disprove context-freeness of collage languages. *Theoretical Computer Science*, pages 1445–1458, 2002.

[Dre00] Drewes F. Tree-based picture generation. *Theoretical Computer Science*, 246:1–51, 2000.

[Dre04] Drewes F. *Grammatical Picture Generation – A Tree-Based Approach*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2004. To appear.

[EMP+98] Ebert D. S., Musgrave F. K., Peachey D., Perlin K., and Worley S. *Texturing and Modeling: A Procedural Approach*. AP Professional, 2nd edition, 1998.

[PL96] Prusinkiewicz P. and Lindenmayer A. *The algorithmic beauty of plants*. Springer-Verlag New York, Inc., 1996.

[PM01] Parish Y. I. H. and Müller P. Procedural modeling of cities. In *Proceedings of the 28th annual conference on computer graphics and interactive techniques*, pages 301–308, 2001.

[PMKL01] Prusinkiewicz P., Mündermann L., Karwowski R., and Lane B. The use of positional information in the modeling of plants. In *Proceedings of the 2001 conference on Computer Graphics*, pages 289–300. ACM Press, 2001.

[RB85] Reeves W. T. and Blau R. Approximate and probabilistic algorithms for shading and rendering structured particle systems. In *Proceedings of the 12th annual conference on computer graphics and interactive techniques*, pages 313–322, 1985.