Computer-Aided Design 56 (2014) 104–119

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

Layered shape grammars☆,☆☆

Manuela Ruiz-Montiel [a,*], María-Victoria Belmonte [a], Javier Boned [b], Lawrence Mandow [a], Eva Millán [a], Ana Reyes Badillo [a], José-Luis Pérez-de-la-Cruz [a]

[a] Escuela Técnica Superior de Ingeniería Informática, Bulevar Louis Pasteur, N° 35, Campus de Teatinos, Universidad de Málaga, 29071 Málaga, Spain

[b] Escuela Técnica Superior de Arquitectura, Campus de El Ejido, Universidad de Málaga, 29071, Málaga, Spain

HIGHLIGHTS

- We propose a computer-aided conceptual design system to assist modelling in the early phases of design.
- Our system enhances shape grammars with layers and logic predicates.
- Layers improve time performance and structuring of shape grammars, and predicates control the application of shape grammars.
- We have applied these new techniques to examples taken from the architectural and video games domains.

ARTICLE INFO

Article history:
Received 27 November 2012
Accepted 20 June 2014

Keywords:
Computational design
Computer-aided conceptual design
Shape grammars
Architecture
Video games

ABSTRACT

In this article we propose a computer-aided conceptual design system to assist modelling at the early stages of design. More precisely, we address the problem of providing the designer with design alternatives that can be used as starting points of the design process. To guide the generation of such alternatives according to a given set of design requirements, the designer can express both visual knowledge in the form of basic geometric transformation rules, and also logic constraints that guide the modelling process. Our approach is based on the formalism of shape grammars, and supplements the basic algorithms with procedures that integrate logic design constraints and goals. Additionally, we introduce a layered scheme for shape grammars that can greatly reduce the computational cost of shape generation. Shape grammars, constraints, goals and layers can be handled through a graphic environment. We illustrate the functionalities of ShaDe through two use cases taken from the architectural design and video games domains, and also evaluate the performance of the system.

1. Introduction

Computer-aided design tools traditionally help designers in many tasks, such as description, documentation and visualization of their projects. However, these tools can also play a more active role in creative aspects of the design process. Some approaches automatically produce highly polished visualizations of designs to be used in movies, video games or computer graphics applications [1–3]. Others offer support in the early, conceptual stages of design, when the involved shapes and ideas are still malleable. These last assistants are usually called *computer-aided conceptual design* (CACD) tools [4–6]. Assistant systems that evaluate, propose and allow exploration of different conceptual design alternatives can be of great help in the early stages of the design process [7–9].

A number of CACD tools have been developed to date. They share the need for some kind of expert knowledge specification, in the form of design requirements and performance criteria, that can be used to evaluate solutions or to guide generation/exploration processes. Approaches range from visual methods, like shape grammars [10], to more explicit techniques such as semantic models [11] or object-oriented relations [12].

In this paper we present two main contributions. In the first place, we propose a CACD system to help in modelling at the early stages of design. The system provides the designer with design alternatives that aid to overcome the *blank page syndrome*. This condition appears when facing a new design project and the absence of starting points entails a lack of inspiration. The alternatives provided can be used as starting points to be further

☆ This work is partially funded by: grant TIN2009-14179 (Spanish Government, Plan Nacional de I+D+i). Manuela Ruiz-Montiel is funded by the Spanish Ministry of Education through the National F.P.U. Program.

☆☆ This paper has been recommended for acceptance by Graham Jared.

* Corresponding author. Tel.: +34 952132863.
E-mail address: mruiz@lcc.uma.es (M. Ruiz-Montiel).

http://dx.doi.org/10.1016/j.cad.2014.06.012
0010-4485/

completed and polished by the human designer. To guide the generation of such alternatives according to a given set of design requirements, the designer can express both visual knowledge in the form of basic geometric transformation rules, and also by logic requirements. This combination takes advantage of the fact that some requirements are better described using logic predicates (in the form of constraints or goals) that are evaluated during the generation process, in the form or constraints or goals. As far as we know, this combined specification procedure is a novel approach to CACD. Our approach is based on the formalism of shape grammars, supplementing the basic algorithms with procedures that integrate logic predicates.

In the second place, we propose to decompose shape grammars into layers. This reduces the computational cost of algorithms for grammar interpretation, and can greatly speed up grammar execution. Moreover, layers can be used to better organize the process, arranging design elements in groups that can be later visualized separately.

These formalisms and algorithms to represent and process rules, constraints, goals and layers, have been implemented and integrated in ShaDe, a new CACD tool built on the commercial CAD software *SketchUp*. ShaDe provides a general editor and interpreter of 2D shape rules, logic predicates (constraints and goals), and layers. Rules are expressed in 2D, but height information can be associated to each layer, allowing to visualize the final drawings in three dimensions. We illustrate the modelling process with ShaDe through an example taken from the architectural field.

The rest of the paper is structured as follows: in Section 2 we provide the necessary background on shape grammars and discuss briefly some related prior work on CACD tools; then, in Section 3 we present the new concepts and algorithms that allow the representation and handling of layers and constraints in a grammar. In Section 4 we describe ShaDe, the software tool that we have developed based on the aforementioned ideas. To illustrate the capabilities of the method and the tool, in Sections 5 and 6 we develop two examples in the field of architectural design and the area of strategy/simulation video games. Section 7 presents a quantitative evaluation. Finally, in Section 8 some conclusions are drawn and future continuations of this research are outlined.

2. Antecedents

2.1. Shape grammars

A shape grammar is a production system in which shapes are generated by means of replacement rules. The concepts of *shape rule* and *shape grammar* were introduced by Stiny and Gips [10]. In this section we gather some necessary definitions [13], and explain the role that shape grammars have played in computational design over the last decades.

A shape grammar is a 4-tuple $\langle S, L, R, I\rangle$ where:

- S is a finite set of shapes
- L is a finite set of symbols
- R is a finite set of rules $\alpha \rightarrow \beta$, where α is a non-empty labelled shape and β is a labelled shape
- I is a non-empty labelled shape, called *initial shape* or *axiom*.

A *shape* is defined by a finite set of distinct lines that cannot be combined to form another line, that is, they are maximal. The representation of a shape is thus unique. A *labelled shape* consists of two parts: a shape and a set of *labelled points*. A labelled point (p, A) is a point p with a symbol A. A labelled shape σ is an ordered pair $\sigma = \langle s, P\rangle$ where s is a shape and P is a finite set of labelled points. A *segment* or *line* l, $l = \{p_1, p_2\}$ is defined by any pair of distinct points p_1 and p_2, the so-called *end points* of the line.

A rule $\alpha \rightarrow \beta$ applies to a shape γ when there is a transformation τ such that $\tau(\alpha)$ is a sub-shape of γ, that is, $\tau(\alpha) \leq \gamma$ (a labelled shape s_1 is sub-shape of another labelled shape s_2 if and only if every line and every labelled point of s_1 is in s_2). τ can be any general geometric transformation. In particular, we will use translations, rotations and regular scales. The sub-shape recognition process needs at least three distinguished points (that can be labels or intersections between segments) in the left side of every rule, as well as in the current design, in order to properly determine τ.

The labelled shape produced by the application of a rule $\alpha \rightarrow \beta$ to a labelled shape γ under transformation τ is given by the expression $\gamma - \tau(\alpha) + \tau(\beta)$. This labelled shape is obtained by substituting the appearance of $\tau(\alpha)$ inside γ with $\tau(\beta)$. In Fig. 1 we can see a rule and one *derivation*, that is, a sequence of shapes generated by successive applications of the rule.

Shape grammars have been used for numerous recreation and generation tasks related to decorative arts, paintings, architectural plans and engineering design [14]. This wide usage relies on the power of shape grammars to capture and recreate heterogeneous design styles. Indeed, many authors have pointed out the advantages of shape grammars as a visual design framework. For example, Stiny showed that they have the potential of producing any possible shape [15]. They are also a *compact* method because few rules can yield such complex and unexpected shapes [16]. More specifically, from the point of view of generative CAD tools, it has been noticed [17] that grammar-based systems can easily automate design and thus allow a great deal of exploration. Shape grammars have been suggested as a geometric design framework due to some desirable properties [18], such as their parametric (instead of symbolic) nature and their maximal unique shape representation.

2.2. Related CACD tools

In this section we provide an overview of several CACD tools. Some of these tools have provided support from an evaluative perspective. Kraft and Nagl [8] developed prototype software providing a visual knowledge specification language for conceptual design. Graph-based domain ontologies define concepts and relations between them, as well as design rules. The designer can use the concepts and relations of this ontology in order to manually instantiate sketches of conceptual buildings (in the context of this work, a sketch is not a shape, but a graph of semantic objects that formalizes the different features of the design idea). The created sketch can be checked against the rule base specified inside the ontology. Pauwels et al. use Semantic Web technologies in order to formalize rules about building performance [11]. The process of defining a design concept would be similar to the one in the work of Kraft and Nagl, using semantic web technologies instead of graph-based techniques. In both works, design concepts are not geometric objects, but explicit, symbolic descriptions about their features. Grabska et al. [19] have developed prototype software to support architectural conceptual design that extracts symbolic information from graphical sketches drawn by the designer. It provides a form of visual requirement specification by means of arrows that relate rooms of different floors. The extracted symbolic information can be used to define logic rules that will check the validity of the sketches drawn by the designer.

Other CACD tools have focused on generation capabilities. A considerable part of the corpus of generative CACD tools are shape grammar interpreters. Chau et al. [14] compared 21 implementations up to 2004. More recently, McKay et al. [9] gathered some of the most relevant interpreters up to 2011. In this last review, the systems are evaluated according to a set of requirements for shape grammar implementations derived from the works of Gips [20] and Chau et al. [14]. Most of the evaluated tools are generic in the sense that they are not aimed at a particular field of design, and provide