Exploring Buildings' Surface Patterns

Inês Caetano
INESC-ID/IST
Lisbon, Portugal
Ines.caetano@tecnico.ulisboa.pt

António Leitão
INESC-ID/IST
Lisbon,Portugal
antonio.menezes.leitao@ist.utl.pt

Abstract: Facades characterized by complex shapes and patterns benefit from the use of Generative Design (GD), which promotes design exploration. However, GD requires the development of algorithms, which can be far from trivial for architects. To simplify this process, we propose DrAFT, a computational framework for the generation and exploration of buildings' surface patterns, which is based on a classification of facades that helps architects identify the algorithms that best suit their design intent. After combining the algorithms provided by DrAFT, designers can more easily explore the solution space of their design ideas, allowing for continuous improvement in the design process.

Keywords*: generative design, facade design, algorithms, framework*

1. Introduction

Architectural creativity requires a design process that embraces change (Bukhari, 2011). Unfortunately, traditional design tools require too much time and effort to change models. On the other hand, new technologies allow further design exploration, promoting the emergence of more complex shapes, details and patterns (Moneo, 2001), which were difficult and costly to produce until recently. In fact, architects have not rediscovered complex forms, they rather found new possibilities to generate and construct them (Kolarevic, 2005). These innovations have had a large impact in the exploration of facade designs, since they allow the generation of shapes and patterns that would not be viable to produce manually (Kolarevic, 2003). Nevertheless, there are still some limitations in the architectural practice, mainly in the production of more complex designs.

2. Algorithmic Approaches to Design

As Woodbury stated (2010), new technologies brought "fresh and needed new capabilities" to the human enterprise of design. Generative Design (GD) is a design process where the output is generated with the help of a computer and through the use of algorithms (Terzidis, 2003). GD not only allows architects to quickly generate, compare, and evaluate multiple solutions, but it also enables the manufacturing of complex solutions.

GD requires algorithms implemented in programming languages (Leitão, 2014). Integrating algorithmic thinking and programming techniques into the design process requires an initial investment but it provides considerable returns in later design phases (Woodbury, 2010).

Parametric Architecture is a design approach based on algorithmic thinking, in which a set of parameters and rules encode and define the relationship between the design intent and design response (Jabi, 2013). The constituent geometry is mutually linked (Burry, 1999), where architects design a set of principles encoded as a sequence of parametric equations, which then generates the model's design and also changes it when needed.

3. Background

GD is being used in the production of digital ornament and complex building skins, which are visible in many contemporary buildings. Some authors have already studied a variety of facade designs, creating different classifications to synthetize and organize contemporary facades into different typologies (Moussavi, 2006; Pell, 2010; Velasco, et al., 2015). However, none of the previous classifications intends to help architects with the algorithmic description of new facade designs which might still require a lot of effort to invent, experiment, and produce.

In the past, we proposed a classification of facades based on a computational approach, which identifies algorithms and strategies that address the needs of different designs (Caetano, et al., 2015). We started with an extensive analysis of contemporary building skins, which was followed by an algorithmic description of some of them in order to recreate the corresponding models. This process helped us realize the existence of similar algorithmic structures (algorithmic patterns) within this variety of facade designs, which means architects can reuse the algorithms to generate further designs. This was also recognized by (Su & Chien, 2016).

In this paper, we present an updated classification that better matches the development process used in the implementation of new facade designs.

4. Buildings' Surface Patterns Framework

Architectural practice is highly dependent on the specific circumstances of the design brief and, thus, it is unlikely that the exact same approach can be used in a different project. Nevertheless, modular programming techniques allow the designer to adapt and reuse ideas in different projects which imply, at least partially, that the systematic application of these techniques reduces the initial investment required.

If these techniques were available at the early stages of facade design exploration, architects would spend much less effort in the programming task, since they would not have to rewrite from the scratch all the algorithms every time they started a new design.

4.1 DrAFT Framework

In our previous work (2015), facades were classified into different categorical dimensions that we considered algorithmically relevant, namely *Facade Geometry*, *Elements Geometry*, *Elements Size*, *Elements Distortion*, *Elements Rotation*, *Elements Distribution*, *Facade Articulation* and *Material & Color*. For each dimension, we developed several algorithmic functions and operators, covering a wide variety of design solutions. Draft Algorithmic Facades Tool (DrAFT) is a framework based on this classification, which guides the designer in the identification

of the functions and operators that best suit his design intent, allowing further design exploration and easier adaptation to the ever-changing design process conditions.

4.2 An Improved DrAFT

After having used DrAFT for more than one year in the exploration of various facade designs, we found that an improved classification was needed, one that better reflects the development process used in the implementation of new facade designs. This classification is based on four main categorical dimensions: A) *Facade Geometry*, which defines the facade's surface geometry; B) *Elements,* which generates the facade's elements; C) *Distribution*, to distribute the elements along the facade's surface; and D) *Articulation*, which defines the appearance of the facade. These four main dimensions are then subdivided to deal with more specific details of each design stage.

In the next sections we discuss each of these dimensions.

4.2.1 Facade Geometry

The Facade's Geometry dimension is used to describe the shape of the facade's surface. This shape is specified by a $\mathbb{R}^2 \rightarrow \mathbb{R}^3$ function. For example, a five-by-ten rectangle on the XZ plane can be described by $S(u,v) = XYZ(u \times 5{,}0, v \times 10)$, where XYZ is the Cartesian coordinate function. Other coordinate systems can be used, such as the Cylindrical, represented by function CYL, and the Spherical, represented by function SPH, to which can be applied Euclidean transformations. To simplify the presentation, each parametric function $S(u,v)$ will range over the domain $0 \leq u \leq 1, 0 \leq v \leq 1$.

To make the framework more intuitive, several kinds of surfaces are predefined as higher-order functions (HOFs), i.e., functions that receive other functions as arguments and/or compute other functions as results (Leitão, 2014). We also make liberal use of anonymous functions, which, following the terminology of the λ-calculus, we will represent by the letter λ.

As an example, our five-by-ten rectangle can be described by the function application $Straight(5, 10)$, where $Straight$ is defined by (1):

$$Straight(w,h) = \lambda(u,v).XYZ(u \times w, 0, v \times h) \qquad (1)$$

It is also possible for the designer to define additional functions. To this end, we also provide a set of functional operators that can be arbitrarily combined. One such operator represents a one-dimensional linear variation: $L(a,b) = \lambda(t).a + (b-a)t$. Another, represents a (paradoxical) constant "variation":$C(c) = \lambda(t).c$.

Given that the facade geometry domain is two-dimensional, it is also useful to extend the domain of the above one-dimensional variations into $\mathbb{R}^2$. To this end, we define $D_u(f) = \lambda(u,v).f(u)$ and $D_v(f) = \lambda(u,v).f(v)$. A final but important operator is the generalized composition of functions:

$$\circ(f, g_1, \cdots, g_n) = \lambda(x_1, \cdots, x_m).f\big(g_1(x_1, \cdots, x_m), \cdots, g_n(x_1, \cdots, x_m)\big) \quad (2)$$

In order to simplify the notation, we define $a \otimes b = D_a\big(L(0,b)\big)$, we treat all numbers n that occur in a function context as $C(n)$, and we treat any ordinary first-order function f that is used

with functional arguments $g_1, \cdots, g_n$ as $\circ(f, g_1, \cdots, g_n)$. As an example, the $Straight$ function presented above can be equivalently defined as

$$Straight(w, h) = XYZ(u \otimes w, 0, v \otimes h) \quad (3)$$

For a different example, consider the sinusoidal facades which are common in recent architecture. The sinusoidal HOF is: $sinusoid(a, \omega, \phi) = \lambda(x).\, a \times \sin(2\pi\omega x + \phi)$, where a is the amplitude of the sinusoid, ω is the angular frequency, i.e. the number of cycles per unit length, and ϕ is the phase. A facade that represents a horizontal wave in the XY plane is then defined by function (4):

$$Sb(w, h, a, \omega, \phi) = XYZ(u \otimes w, D_u(sinusoid(a, \omega, \phi)), v \otimes h) \quad (4)$$

On the other hand, there are facades with completely irregular shapes, which are classified in the Facade's Geometry dimension as Free-Form. In this last case, the designer creates the shape manually, and imports it into our framework where it is represented as another parametric function that results from an interpolation process.

4.2.2 Element

Most facade designs or patterns are created by the repetition of a certain unit, which can be kept unchanged along the facade's domain or can be changed in relation to its shape, size, etc. We name these units as *Elements* and our second dimension is in charge of generating them. For this, we merged the dimensions *Element's Geometry, Size, Deformation* and *Rotation* into this one, and we also added some more features. We sub-divide this dimension *Element* into *Shape* and *Transformations* groups. The first group provides a set of functions that generates different shapes. The second group provides a set of operations that deals with different kinds of transformations, including contraction, expansion, dilation, reflection, rotation, shear, twisting, bending, interlacing, and combinations of these.

In practical terms, the facade element is implemented with a HOF that takes other functions as arguments, representing the different characteristics of the element (including shape and shape transformations).

After selecting the facade's geometry and the element's geometry, it is time to combine them to define the complete facade. One of the advantages of the functional representation is that it makes this combination a trivial composition of functions. As an example, a straight facade with spherical elements of radius r is defined by $sphere(Straight(w, h), r))$. The result is a continuous function that generates spheres on a $w \times h$ rectangle on the XZ plane. The next section discusses the actual distribution of spheres.

4.2.3 Grid Distribution

In the previous sections, we described the functional description of the facade geometry and of its elements. Although this description is a continuous function, most facades are discretized, which means that the functional description of the facade is mapped, not on a continuous domain, but on a discrete domain, obtained by a sampling process. This process characterizes the Grid Distribution dimension, which deals with the placement of the elements

along the facade. This is accomplished by a discretization function $sample(f, n, m)$ that, given the f function defined in the domain $[0,1] \times [0,1]$, and the intended number of samples in the two dimensions, computes the bi-dimensional sampling of f. There are many different ways to do the sampling process (regular grid, hexagonal grid, chess-grid, etc.) and each of these will produce a different kind of facade. An additional feature to consider is the influence of the *metric* of the facade's surface on the elements' shape or size, so that these adapt to the distortions of the facade. To this end, we include an additional step that combines sampled values into a virtual quadrangle whose vertices define a *patch* used to adjust each surface element. This adjustment can also be controlled by the presence of one or more attractors.

4.2.4 Facade Articulation

The same surface with the same elements and grid distribution can originate different aesthetic results according to how they relate or the material used. Articulation is the way of jointing the different parts in order to create the final facade. This can be done through the use of materials, different techniques applied on the buildings surfaces (Pell, 2010), or by the means of architectural patterns (Schumacher, 2009). Therefore, we considered as sub-dimensions: i) material used, ii) different techniques (perforations, stacked, applied, etc), and iii) depth of the facade pattern (number of layers, web effect, etc).

These sub-dimensions provide a range of functional operators that, after being selected by the user, are combined in the HOF *articulation* along with the functions from the other main dimensions: *facade-Geometry, element* and *distribution.*

5. Evaluation

In this section, we develop two facade designs using our framework. We explain the whole generation process starting from the classification of facades until the selection and, then, combination of the functional operators and algorithms. As a result, the obtained combination of functions generates the corresponding facade design, on which we can apply several variations.

The first example, visible in Figure 1, is inspired by the Nolan Building facade in Melbourne (by PLUS Architecture). It has a sinusoidal shape and is composed by squared elements distributed in a regular-grid and showing two transformations: a horizontal randomly-controlled rotation, and a vertical rotation that increases with the facade's height. These elements are then applied on the facade's surface. This classification determined the algorithms that we combined to create the facade's model.

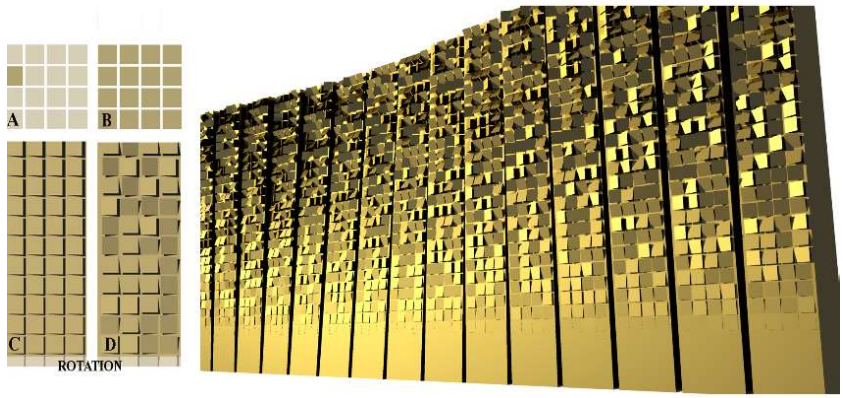

Figure 1 – Nolan Building facade model. On the left: the algorithms combined to produce the model (A- Element; B- Grid Distribution; C/ D – Element Transformation and Applied Articulation); On the right: the generated model.

The second example was inspired by the facade of the Louis Vuitton store in Shenzhen. This facade has a regular shape and is composed by squared elements whose size decreases with the facade's height. Their distribution is in chess-grid and they are applied on the facade's surface.

After combining the provided algorithms and, thereby, generating this facade model, we can experiment some design alternatives. Figure 2 summarizes this process and shows three possible design variations that were produced by changing the geometry of the element or its size variation.

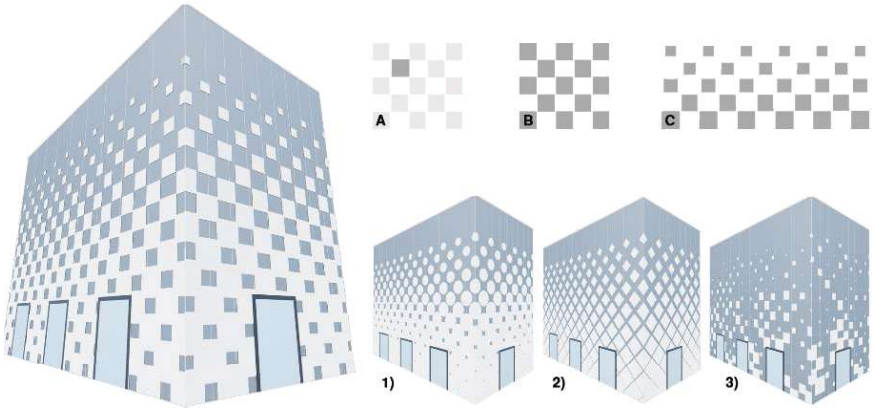

Figure 2 – Left: model based on the Louis Vuitton store facade; Top (A-C): the algorithms combined to generate the model (A- Element; B- Distribution; C–Transformation); below: three design alternatives. In 1) and 2) we changed the element's shape to circular and to diamond-shape. In 3) we used a random size variation.

6. Conclusions and Future Work

GD promotes design exploration but requires that architects spend time and effort in the algorithmic description of their designs. Were architects able to take advantage of modular programming techniques, they could adapt and reuse some pre-defined algorithms in new projects, which would reduce their initial investment.

In this paper we presented an improved framework for facade design - DrAFT. Our framework is based on a simple and flexible classification scheme that addresses the needs of architects designing facades. The classification is divided into four main dimensions, each

representing a stage of a facade design, which are then further subdivided to deal with a wide range of design characteristics in each design stage. For each subdivision, we propose a set of algorithms implemented as functions or functional operators. Depending on the number of algorithms selected (or implemented) from each dimension, the more or less customized the facade design will be.

In practical terms, the composition of the algorithms within DrAFT facilitates the algorithmic description of facade designs and allows the experimentation and variation of the generated models. DrAFT is implemented using the Rosetta IDE (Lopes & Leitão, 2011), allowing its exploration with different programming languages and different CAD and BIM applications.

Our case studies demonstrated the simplicity of using DrAFT not only to produce a facade model, but also to easily change its design.

In the near future, we plan to expand the set of available algorithms and we are particularly interested in conducting a wider field study of its application in real situations, including its extension to manufacturing.

Acknowledgements

This work was partially supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UID/CEC/50021/2013.

References

Bukhari, A. F., 2011. *A Hierarchical Evolutionary Algorithmic Design (HEAD) System for Generating and Evolving Buildings Design,* Queensland University of Technology: Built Environment and Engineering, School of Design.

Burry, M., 1999. Paramorph. In: *Hypersurfaces 2*: Academy Editions , pp.78-83.

Caetano, I., Santos, L. & Leitão, A., 2015. *From Idea to Shape, from Algorithm to Design: A Framework for the Generation of Contemporary facades.* São Paulo, Brazil, The Next City: 16th CAADFutures, CCIS 527, Springer.

Jabi, W., 2013. *Parametric Design for Architecture.* London: Laurence King Publishing.

Kolarevic, B., 2003. *Architecture in the Digital Age: Designing and Manufacturing.* London: Spon Press.

Kolarevic, B., 2005. Towards the Performative in Architecture. In: *Performative Architecture: Beyond Instrumentality.* NY: Spon Press, pp.203-214.

Leitão, A. M., 2014. *Improving Generative Design by Combining Abstract Geometry and Higher-Order Programming.* Kyoto, 19th CAADRIA.

Lopes, J. & Leitão, A., 2011. *Portable Generative Design for CAD Applications.* Calgary, Canada, ACADIA.

Moneo, R., 2001. The Thing Called Architecture. In: *Anything by C. Davidson.* New York: Anyone Corporation, pp.120-123.

Moussavi, F., 2006. *The Function of Ornament.* ACTAR.

Pell, B., 2010. *The Articulate Surface - Ornament and technology in Contemporary Architecture.* Birkhauser.

Schumacher, P., 2009. Patterning Parametrically. *Architectural Design: Patterns of Architecture,* Volume 79, Nº6 (Nov/Dec), pp.28-41.

Su, H.-P. & Chien, S.-F., 2016. *Revealing Patterns: using Parametric Design Patterns in Building Facade Design workflow.* Melbourne, Australia, 21st CAADRIA.

Terzidis, K., 2003. Introduction. In: *Expressive Form: A Conceptual Approach to Computational Design.* NY: Spon Press, pp.1-8.

Velasco, R., Brakke, A. P. & Chavarro, D., 2015. *Dynamic Façades and Computation: Towards an Inclusive Categorization of High Performance Kinetic Façade Systems.* São Paulo, Brazil, CAADFutures, CCIS 527, Springer, pp.172-191.

Woodbury, R., 2010. *Elements of Parametric Design.* NY: Routledge.