ALGODE TOKYO 2011

Proceedings of the International Symposium on Algorithmic Design for Architecture and Urban Design,
ALGODE TOKYO 2011
March 14-16, 2011, Tokyo, Japan

Procedural Ornament

A Dataflow Approach to Subdivision

Michael Hansmeyer

Chair of Computer Aided Architectural Design
Institute for Technology in Architecture
Swiss Federal Institute of Technology (ETH), Zurich
hansmeyer@arch.ethz.ch

Abstract

This paper expands on the use of subdivision algorithms for the production of three-dimensional ornament. These processes have the potential to generate a broad range of geometric morphologies. Yet these processes can be difficult to control as the link between parameters and output is not always discernable. To address this problem, this paper proposes a dataflow representation to depict and control the subdivision processes. This exercise in abstraction greatly aids transparency, and it allows these processes to be used in a more deliberate, intentional manner.

Keywords: Procedural ornament, subdivision processes, dataflow programming, generative design

1 Introduction

Subdivision processes can be used beyond their initial intention of smoothing forms to generate a broad range of geometric morphologies. Given even the simplest input mesh, the recursive application of these processes can create elaborate geometries that exhibit both a topographical and topological complexity [1]. Subdivision processes thus have the potential to be used as a generative design tool. They can be applied, for instance, to the production of intricate systems of ornament.

To make full use of these processes, however, necessitates the specification of a profusion of parameters. These can vary in both time and space. The major obstacle in using these processes proactively as design tool is the difficulty in establishing causality between the parameters employed and the output produced. This difficulty is exacerbated by the fact the output itself cannot easily be described or classified.

This paper briefly reviews modifications to subdivision processes that broaden the range of possible morphologies. It then examines the issues that hinder an understanding of the linkage between parameters and output. Finally it proposes a solution that entails the use of a dataflow approach to mapping the generative processes.

2 Extended subdivision schemes

The processes described in this paper are largely derivates of two established subdivision schemes: the Catmull-Clark process [2][3], and the Doo-Sabin process [4]. The scope of morphologies that can be generated using these schemes is broadened on the one hand by modifying the schemes themselves, and on the other by expanding the manner in which these schemes are applied.

2.1 Parametrization of the vertex placement rules

Modification of the two schemes occurs primarily on the level of the vertex placement. Thus parameters are introduced – henceforth referred to as *weights* - to specify vertex positions as a variable interpolation of previous-generation vertices. The initial subdivision schemes are amended with further weights to allow an extrusion of vertices along the normal vectors of previous-generation faces, edges and vertices. Finally, weights are introduced to control the rotation of vertices around face midpoints, edges, and corner points. These changes can be summarized as modifications to processes' topographical rules.[1]

2.2 Temporal and spatial differentiation of weights

In the initial subdivision schemes, one implicit weighting value was uniformly applied for multiple iterations to produce rounded forms. The introduction of weighting parameters, however, allows the possibility of *non-stationary* weights. This implies that weights can assume distinct values at each iteration of the subdivision process. Certain characteristics of a form can be generated by a single iteration of the process, while most emerge through the interplay of several iterations with distinct parameters.

The concept of *non-uniform* implies distinct weighting values at different parts of the mesh. These non-uniform weights can be specified according to the position within the mesh, according to pre-determined differentiation of the mesh[2], or by considering the mesh's intrinsic attributes – both in terms of its topography and topology. The latter method is particular in that the initial weights used affect the mesh's topography and topology, which in turn influence the weights of the subsequent iteration, and so forth. Thus the weights used are not all determined in advance, but by the output that is produced at each iteration.

2.3 Process behavior

The range of morphologies that these modified processes can generate is immense. The processes themselves behave mostly in a linear manner: a small change to the value of weight generally leads to a small change output. As such, they are entirely deterministic. This ensures that their output is easy to modify: output is traceable and can be adjusted and optimized over a series of trials.

---

[1] Subdivision processes also employ combinatorial rules that state how the topology of the mesh is refined. These rules can be amended by introducing additional topological masks.

[2] This could be achieved by classifying faces, edges or vertices of the input mesh into groups. Separate weights can then be applied to each group.

Proceedings of the International Symposium on Algorithmic Design for Architecture and Urban Design,
ALGODE TOKYO 2011
March 14-16, 2011, Tokyo, Japan

3 Intransparent causality

Despite the linearity of the processes, it is difficult to intuitively grasp the link between the parameters used and the output produced. Yet defining and understanding this relationship is precisely what is necessary in order to use these processes in a deliberate manner. The problem in establishing this link lies on two levels: first, there is a true profusion of possible parameters, and second, the output itself is difficult to classify.

3.1 Profusion of process parameters

Assuming a process based on the Catmull-Clark scheme that runs for ten iterations with weights to control interpolation, extrusion, and rotation of face points, edge points, and corner points respectively, one arrives at a total of 90 parameters that can be specified. This example, shown in figure 1, is based on uniform weights throughout the mesh. In the case of non-uniform weights, the number of parameters increases manyfold.

			iteration									
point	operation	id	1	2	3	4	5	6	7	8	9	10
face	interpolation	w_{fi}	-	-	-	-	-	-	-	-	-	-
	extrusion	w_{fe}	-	-	-	-	-	-	-	-	-	-
	rotation	w_{fr}	-	-	-	-	-	-	-	-	-	-
edge	interpolation	w_{ei}	-	-	-	-	-	-	-	-	-	-
	extrusion	w_{ee}	-	-	-	-	-	-	-	-	-	-
	rotation	w_{er}	-	-	-	-	-	-	-	-	-	-
corner	interpolation	w_{ci}	-	-	-	-	-	-	-	-	-	-
	extrusion	w_{ce}	-	-	-	-	-	-	-	-	-	-
	rotation	w_{cr}	-	-	-	-	-	-	-	-	-	-

Fig. 1: Subdivision parameters in table form.

While one has the possibility to simply list these parameters or to arrange them in tables, doing so fails to express potential relations between them. Many of the parameters are unlikely to be independent and entirely unrelated values. Rather, a value may evolve over several iterations, or one value may be a multiple of another. Besides failing to depict these relationships, lists and tables also make it difficult to establish which of the parameters, at which iteration, is responsible for generating a certain attribute of the form. It is unavailing to extract relationships and draw conclusions when one is presented with dozens and dozens of values. What one is left with is a highly ineffective, repetitive tweaking of individual parameters on a trial-and-error basis.

3.2 Output defies classification

A further challenge to establishing causality lies in the fact that the forms generated are difficult to describe and do not lend themselves to facile classification. Despite being deterministic, they often cannot be understood through the reductionist process of identifying their constituent parts. These forms arguably have no constituent parts beyond their simple input mesh, as all information is added on a procedural level through the subdivision operations. The precise nature of these operations is equally difficult to derive by examining the output. The recursiveness of the processes, in which output of one iteration servers as input for the next, impedes a reductionist understanding.

Descriptions of generated forms are thus usually limited to inspecting their state, rather than characterizing their formative processes. They tend to be either surveys of a form's geometric attributes and features, or to be purely metaphorical descriptions. Geometric surveys can identify both topographical and topological properties, and can describe singular conditions, repetitive elements, or the overall form. Labels such as protruding, bulging, flaring, shearing, straining, creasing, and branching – just to name a few – come to mind.

Beyond these geometric surveys, descriptions are often metaphorical in comparing the form to natural phenomena; whether biological, ecological, or geological. Frequently motifs are identified that are associated with plants and animals.

3.3 The missing link

On the one hand, one thus has an abundance of input parameters without the possibility to derive meaning from their values, on the other hand one has forms that are difficult to classify. Establishing relationships between lists of parameters and vague descriptions is nearly impossible through a purely empirical process. What is required is a degree of abstraction that describes these processes on a higher level, so as to illustrate sets of relationships between the processes and parameters.

4 Dataflow representation

A solution to this problem lies in depicting and controlling the subdivision processes using a dataflow approach. This approach entails creating a series of graphs by connecting different sets of elements. As such, this approach borrows heavily from ideas initially put forth by William Sutherland [5], and more recently implemented in visual programming languages such as Pure Data and Max.

4.1 Anatomy of the graph

A graph is composed of a series of elements (similar to nodes) that are connected together using arrow that indicate the direction of the flow of information. Elements are generally classified into three categories: source elements, operator elements, and target elements, with information generally flowing from the former through to the latter. Elements can have output port and input ports as well as control ports that specify how they can send, receive, and process information. The three types of elements are described in more detail:

4.1.1 Target elements

An important feature of the subdivision processes is that they remain constant; it is only the values of the parameters – the weights – that are allowed to change. These weights are considered the *target elements* of the graph. Each weight has a default value that is used if a value is not specified using the graph.

4.1.2 Source elements

Source elements take the form of either fixed data, or of measurements. The former consist of fixed scalar or vector values. The latter quantify or classifications of diverse topographical and topological attributes of the mesh. Measurement can take place on the level of either the mesh's vertices, edges, or faces. In addition measurement elements can be used to calculate attributes of the mesh's relationship to environment. A selection of source elements that have proven particularly effective is shown in figure 2.

Topographical	Topological	Environmental
·Planarity (quads) ·Local curvature ·Area ·Perimeter ·Rotation ·Relative position ·Jaggedness (=perimater/area)	·Valence of vertices ·Number of neighbors ·Distance to initial edge ·Distance to initial point ·Identification of motifs	·Distance to point/vector/plane ·Orientation

Fig. 2: Selection of topographical, topological and environmental attributes of mesh

4.1.3 Operator elements

Operator elements are used to link source elements with target elements (i.e. the weights). They can be either mathematical or logical in nature, or they can be more complex combinatorial techniques such as heat maps. Logical elements can incorporate conditional tests.

4.2 Graphs in the context of subdivision processes

There are several graph conventions that can be useful in the context of subdivision processes. First, weights in the graph can be specified with relative iteration indices rather than being tied to a fixed iteration. This allows the graph to be "plugged in" to any step in the subdivision process. Graphs can nevertheless depict weights from multiple (relative) iterations, allowing the possible relations between them to become explicit. Second, while weights usually function as target elements, they can also act as source elements to be connected to further downstream weights. Figure 3 displays a simple graph example that targets three weights. Also shown is its output when given cylindrical mesh.
.

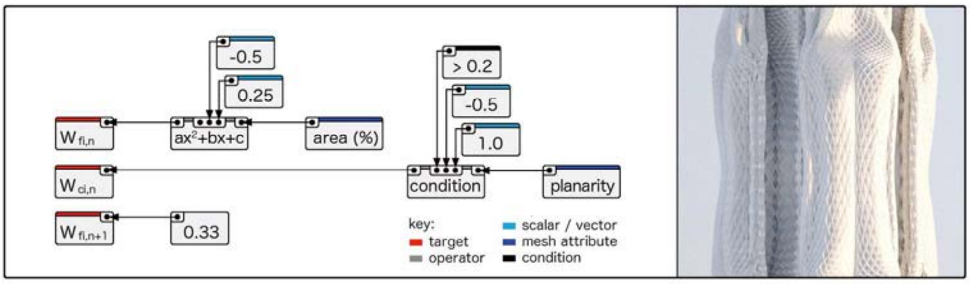


Fig. 3: Graph using the faces' planarity and area percentile to specify weight values.

In the context of subdivision processes, it is essential that certain operator elements can process several sources in parallel. For instance, a spatial interpolation between several points along the mesh can be applied to multiple sets of weights. Another feature of the graphs is that they are entirely modular, and a graph can function as an element within another graph. Both of these concepts are shown in figure 4.

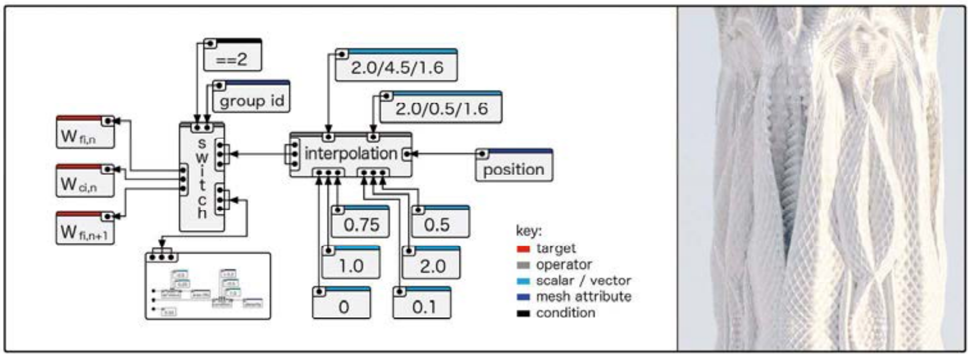


Fig. 4: Graph with interpolation of weights along y-axis, using graph from figure 1 as modular component (see bottom left corner).

Graphs are usually particular to one input mesh, as the process parameters will produce different output depending on the input mesh's topology, topography, and – if specified - the grouping of its faces. While graphs can be applied to any iteration or series of iterations of the subdivision process, many graphs are most effective when calibrated to work at specific steps.

These steps can roughly be divided into three distinct phases of the subdivision iterations. The first two or three iterations are instrumental in generating the basic underlying form, particularly in regards to the broad curvature of its surfaces. The next two or three iterations will generate surface attributes on top of this basic form: they will create features such as folds, protrusions, compression and straining. The final iterations of the process generate a type of micro-structure on top of these surfaces. This concept is shown in figure 5.

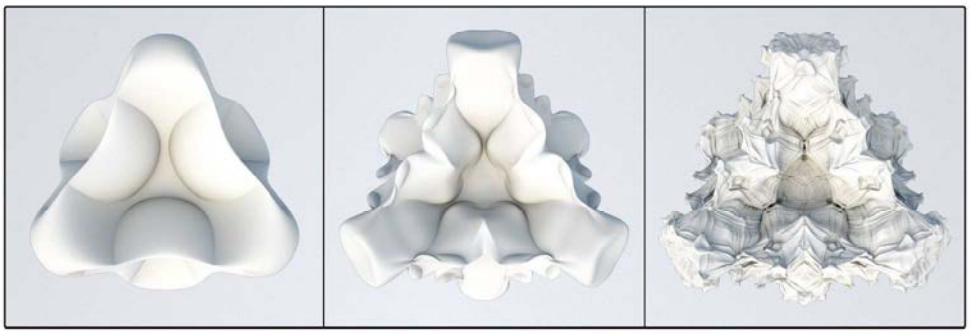


Fig. 5: A cube divided for 2, 5, and 8 iterations, showing the development of a basic form, a distinctive surface, and finally a micro-structure respectively.

This division into three steps is by no means absolute, yet it can provide a general guideline for the design process and indicate how graphs can be used in a modular manner.

4.2.1 Graphing permutations

One way of exploring the potential of subdivision processes is to use them to generate permutations of a form. The dataflow approach is predestined for this task, as any fixed scalar or vector source element can simply be replaced by a random number generator. Unlike the list and tables approach, in which permutations can only be generated by varying the weights themselves, the graph approach allows relationship between elements to stay intact. Further, one can use random numbers not only as inputs to target elements, but to determine how measurements are transformed. This can be done by using them to control operator elements. Going one step further, randomization can be used to trigger switches, thus activating/deactivating parts of the graph, or specifying how different modules of graphs are combined.

4.3 Vocabulary of forms

The graphs can be understood as modular building blocks, which can be copied, combined and reconfigured to create new processes with ever increasing degrees of specificity. Combined with either a very general or a highly differentiated mesh, they can lead to an entire vocabulary of new forms.

While these forms can be tagged with descriptive keywords, perhaps it is fully sufficient to describe them simply in terms of their constituent dataflow graphs. These very simple mechanisms greatly facilitate and understanding of the generative processes: an ornament attribute is simply a function of its visually explicit dataflow graph.

The graphs are easy to read and interpret, they are easy to compare – far more so than comparing individual values of tables – and they are easy to reconfigure. As they can be plugged into different iterations of the process, or into different parts of the mesh, establishing a causality with the output is greatly facilitated.

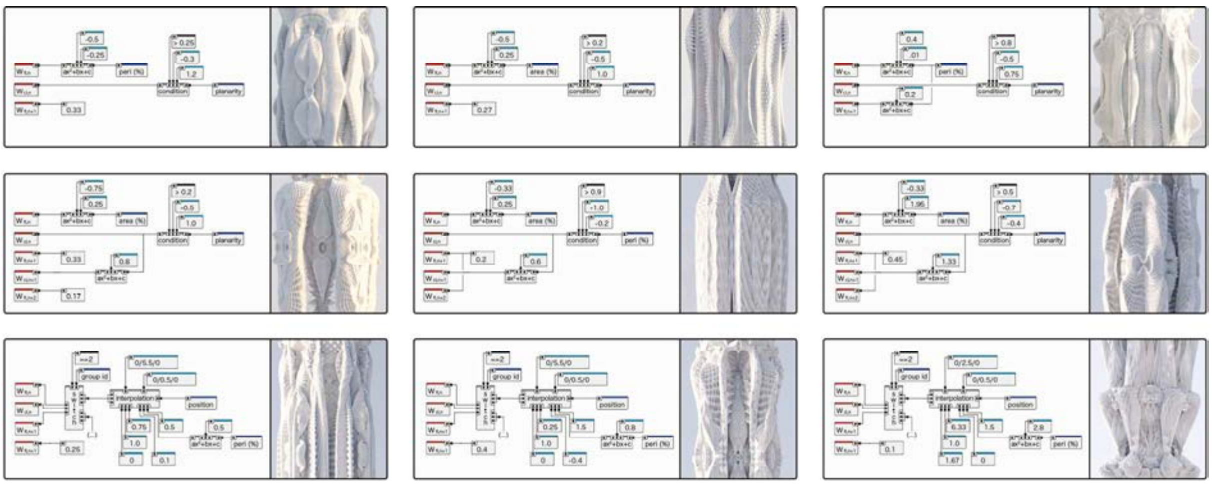


Fig. 5: Families of ornament developed using permutations of graphs.

While resorting to this degree of abstraction may appear at first glance to increase the complexity of the design process, the exact opposite is the case. The abstraction gives meaning to the weighting values by establishing relationships between them, it thus makes the values more concrete. One gains a high degree of control, and one can use this control to seek out the full potential of the subdivision processes.

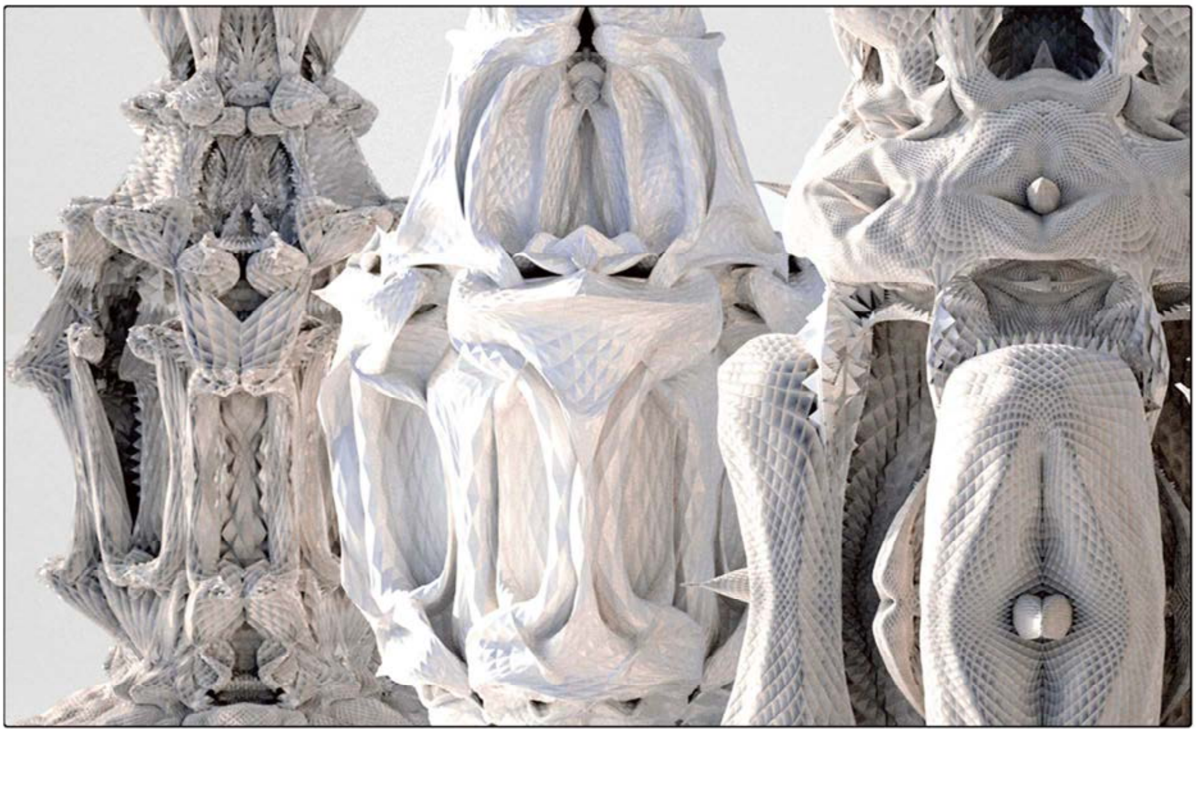
5 Outlook

The dataflow approach could be used to develop an entire vocabulary of subdivided forms. More specifically, it could be used to generate a vocabulary of ornamental morphologies. This vocabulary could serve as points of reference, or as points of departure for a new design. The degrees of freedom in combining elements are almost infinite, and any vocabulary will be far from exhaustive. This raises the question how one would approach the development of such a vocabulary. Should it be through a top-down approach, should it be by permutating the graphs, or perhaps through some completely different technique? Should it be driven by the output, or by the generating process? In the case of the former, the question arises how one should evaluate output.

The prospect of a vocabulary of forms also raises another question: on what type of input mesh should this catalog be based? The processes produce entirely different output depending on the attributes of the mesh used. Should the input mesh be simple or heterogeneous? The former would broaden the applicability of the graphs and would encourage their use in a modular manner. The latter would require far less expansive graphs to achieve a comparable level of complexity in output.

While many questions remain to be answered, a dataflow approach to subdivision puts such a vocabulary well within reach

6 Appendix: Ornamented columns



References

[1] M. Hansmeyer, Subdivision Beyond Smoothness, *Proc. Computational Aesthetics*, London, England, 2010

[2] E. Catmull, *A Subdivision Algorithm for Computer Display of Curved Surfaces,* Dissertation, Department of Computer Science, University of Utah, December 1974

[3] E. Catmull, J. Clark, Recursively Generated B-spline Surfaces on Arbitrary Topological Surfaces, *Computer Aided Design 10*, pp. 350-358, November 1978

[4] D. Doo, A Subdivision Algorithm for Smoothing Down Irregularly Shaped Polyhedrons, *Proc. on Interactive Techniques in Computer Aided Design*, pp157-165, 1987

[5] W. R. Sutherland, *The On-line Graphical Specification of Computer Procedures,* Dissertation, Massachusetts Institute of Technology, January 1966