Subdivisional Growth Logics

Christoph Klemmt1
1University of Cincinnati / Orproject
1christoph@orproject.com

This research explores the combination of two different types of algorithms that have so far been treated separately in architectural computational design: recursive subdivision, and differential or cellular growth. The two algorithms appear to act in opposite directions, the first is a refinement going inwards while the latter is growing outwards. However, both algorithms are based on the refinement of mesh geometries by inserting new vertices and faces and can be used in combination. The resulting subdivisional growth can be used to enhance specific geometric traits of either recursive subdivision or cellular growth at different scales or in different sections of the design object. The resulting geometries have been explored through case studies that utilize those possibilities.

Keywords: architecture, computational design, differential, growth, recursive, subdivision

INTRODUCTION

In architectural design, two types of algorithms have in recent years gained interest: those generating recursive subdivision surfaces, and those creating differential or cellular growth. The two algorithms appear to be opposites: Recursive subdivision refines a geometry on a finer and finer scale proceeding inward, while differential growth grows geometry larger and larger outward.

Both types of algorithm are closely related, and this research explores their use in combination in order to utilize the strengths of both systems. Subdivision algorithms, as well as differential or cellular growth algorithms, are based on point clouds and, more specifically, on mesh geometries. Both algorithms begin with a relatively small set of vertices, and insert additional vertices into the geometry in iterative steps, resulting in a large accumulation.

Recursive subdivision, as the name suggests, operates by subdividing the faces of a mesh. In each iteration of the algorithm, specific faces, or possibly all faces, of a mesh are identified for subdivision. Identified faces are then removed and replaced by three or four new faces. These new faces connect to the existing vertices of the removed face, but require the insertion of one or more new vertices to fully define their geometries. The new vertex might be inserted in the center of the previous face, so that the new faces connect the edges of the previous face with the new vertex. The new vertex can be repositioned according to mathematical rules or attractors in space. The subdivision algorithms result in a geometry that becomes more and more refined in each iteration as the faces are further subdivided and more vertices are inserted into the geometry. The original form is often still recognizable in the final geometry, and distinct tessellations and patterns with sharp edges can occur. (Hansmeyer 2010a; Hansmeyer 2010b; Hansmeyer and Dillenburger 2013; del Campo 2016; Carpo 2016)
Differential growth or cellular growth algorithms are likewise based on point clouds that are often connected to form a mesh geometry. Vertices identified by the algorithm, most often those at the edges of the mesh, divide, form new faces, and reconnect to the existing mesh. The vertices are pushed apart from each other so that the overall geometry extends outwards and is usually smoothed in the process.

Variants of the algorithm can identify interior vertices for division, and mesh edges may be flipped in order to generate a more coherent geometry. The outcomes are often relatively smooth, organic geometries. (Hart 2009; Lomas 2014; Patrick 2015; Klemmt and Bollinger 2015; Klemmt and Sugihara 2017)

The main difference between the two algorithms is that the first one is based on a refinement and inner development toward a smaller and smaller scale, whereas the second is based on an outward, growing development. In the first algorithm the process of division operates on mesh faces, while in the second one the division operates on vertices. Also, the forces that push proximal vertices away from each other in the second algorithm often leads to smooth final geometry, while the first algorithm often embeds distinct patterns with sharp edges and corners in the final geometry.

The algorithm explored for this paper is based on a subdivision of faces. However, forces are introduced to push close vertices apart from each other in order to create a subdivisional growth logic. Specific behaviors of both algorithms, such as a mathematical repositioning of vertices, influences of attractors, or a readjustment of mesh edges, are explored for their potential, often only applied at specific iterations during the development of the form.

**COMBINED ALGORITHM**

The algorithm is based on mesh geometries that are iteratively refined over time. An initial mesh, which may only have a small number of vertices and faces, is given as the starting geometry. The meshes that have been used for this paper are always triangulated. Different types of operation are repetitively applied to the mesh; the operations fall into one of four categories:

**Repositioning of Vertices**

The repositioning of mesh vertices is usually executed several times in small steps between the other operations. Repositioning is determined by forces acting on the vertices. A force that pushes vertices apart from each other that are connected by an edge is necessary in order to generate a growth of the mesh rather than just a refinement. The following forces have also been explored:

- A spring force with a rest length larger than zero between vertices connected by an edge. This force will attempt to keep adjacent vertices at a defined distance from each other. It will cause areas with dense vertices to expand, but it will also cause areas with sparse vertices to contract, resulting in a smoothing of the overall mesh geometry.
- A repulsion force between vertices that are connected by an edge. This force can be constrained to only act on vertices that are within
a specified proximity to one another, resulting in an expansion of the overall mesh geometry.

- A general spring force that can act between neighboring vertices, between vertices that are not connected by an edge but that are in close proximity to each other, or that can act toward attractor points or objects placed in the scene.
- A general repulsion force that can act between neighboring vertices, between vertices that are not connected by an edge but that are in close proximity to each other, or that can act toward attractor points or objects placed in the scene.
- A general attraction force that can act between neighboring vertices, between vertices that are not connected by an edge but that are in a certain position towards each other, or that can act toward attractor points or objects placed in the scene.
- A smoothing force that smooths the mesh by repositioning the vertices. The force works by pulling vertices towards the average position of its neighbors.
- A unary force that pulls vertices in a specified direction.

The forces can be controlled through their strength and in many cases have been set to not act uniformly across the whole mesh. They can be controlled by the distance between vertices, by the position of a vertex in space, or by the distance between a vertex and a control point or control object in the scene. The behaviors of the forces can also change from iteration to iteration during the refinement process.

**Vertex Insertion**

The mesh topology is refined by subdividing its faces. Either all or only certain faces may be subdivided in one step. The meshes for this paper are all triangulated, so the face to be subdivided is triangular, as are the resulting faces after the subdivision.

A face can be subdivided by inserting a new vertex at its center and connecting three faces from the previous face’s edges to the new vertex. Alternately, two adjacent faces can be subdivided by inserting a new vertex in the middle of their common edge, and the four outer edges of the two previous faces reconnected with new faces to the new vertex.

The new vertex can be inserted exactly at the center point of the previous face or edge, or it can be inserted offset from that position. Successful variations used an offset along the normal direction of the previous face, or an offset according to attractor points within the scene.

**Flipping of Mesh Edges**

The flipping of mesh edges forms an alteration to the mesh topology without any change to the vertex positions. This operation has been used especially in order to create a more distributed triangulation across a mesh. If the mesh is subdivided by inserting a new vertex in the middle of a face several times, the original vertices of the mesh can begin to have an excessive number of edges connected to them. This can be countered by flipping those edges.

The flipping was applied to the common edge between two triangular faces that together form a quadrangle, with the common edge between the two faces forming one of the quadrangle’s two possible diagonals. By flipping the edge, the quadrangle is divided into two different triangular faces along its other possible diagonal. If the new diagonal, common edge after the flipping was shorter than the previous one, the flipping was executed.

**Removal of Mesh Faces**

It is possible to remove certain faces from the mesh. This has been applied to faces that were above or below a certain size, aspect ratio, or that were at a certain distance from a control point in the scene. It has also been applied to individual faces across the whole mesh in order to generate patterns of openings.

**EXPLORATIONS**

**Generic Explorations**

Various tests have been performed with the combined subdivisional growth algorithms. Most explorations started with a very basic mesh geometry, such as a simple rectangle or platonic solid. The behaviors of the different rules for repositioning and
Figure 3
Development Sequence.

Figure 4
Development Sequence.

Figure 5
Development Sequence.

Figure 6
Geometry Explorations.

Figure 7
Organic Growth with Subdivisional Refinement.
subdividing were explored, but also the effects that can be achieved by executing certain operations at specific times during the process of the formation. (Figures 1-4)

As expected, it was found that in general, early iterations have a significant influence on the overall geometry of the final form, and often a geometric tendency is already visible after the first few iterations. Most of the refinement of the surface treatment then occurs during the later iterations. However, it was also found that specific geometric traits that had been generated, such as a spikiness of the surface, could be counteracted by introducing a smoothing later on that possibly acted only on certain parts of the overall geometry. (Figure 5)

The findings of the general explorations were applied to three design case studies: A given geometry that was to be refined by the growth; a structure that was to react to spatial constraints and performance; and a geometry that was to grow in a distinctly different manner across its extents.

**Geometry Refinement through Growth**

This example is closest to a standard subdivision algorithm in that it is based on a given geometry that is to be refined, in this case the basic geometry of a single-surface chair. The edge of the geometry is fixed and cannot move, however it can be subdivided and therefore refined. The aim of this example was the exploration of behaviors for subdivision that would usually be associated with differential or cellular growth.

The subdivisional growth in this case resulted in a bulging surface that clearly exhibits properties of both subdivision and differential growth algorithms. The subdivision is present in the tessellation of the surface, especially visible from the front of the chair. The conditional subdivision based on the proximity to attractor points resulted in a patterning with slight similarities to reptile skins.

On the backside of the chair two traits of differential growth algorithms can be seen: the smoothing of the surface and the repulsion of close vertices.

For the design project, the different behaviors have been used to create a smoother and softer surface in the central areas of the chair where the users are most likely to make contact, while the geometry along the edges was given a larger pattern with increased depth for structural stability. (Figure 6)

**Organic Growth with Subdivisional Refinement**

While in the previous project the larger scale geometry was driven by a subdivisional logic and the differential growth logic became visible on the smaller scale, in this project the opposite is the case: The overall geometry follows a differential growth behavior that generates a relatively smooth distribution of interconnected volumes. At a smaller scale, this geometry is then refined by distinct subdivisional treatments that pattern the surface, going from an almost flat tessellation towards a directional, highly distorted triangulation.

In the design project, the control over those geometric behaviours has been utilized in order to create an acoustic pavilion that can limit or enhance sound reflections across its different spaces. (Figure 7)

**Differentiation by Region**

The aim of this project was the creation of morphologic variety across the overall geometry. The shape was developed out of a simple regular tetrahedron with four sides. Apart from using attractors and location parameters in space to create variety in the subdivision and repositioning behaviors, this project made extensive use of the iterative removal of mesh geometry and of flipping as well as limiting the flipping of mesh edges after the subdivision.

The resulting geometry bears little resemblance to its underlying regular tetrahedron. There is a clear differentiation between interconnected more blob-like and more spiky macro-geometries. On the smaller scale, a texture is generated through a combination of subdivisional and differential growth logics.

**CONCLUSIONS**

Although recursive subdivision and differential or cellular growth algorithms appear to work in opposite directions, they can easily be combined into a co-
herent logic for the development of form. The resulting combination can make use of the different specific traits of both algorithms, and as shown in the explorations and case studies, those can be applied to varying degrees, at various scales, and across varying regions within a geometry in order to create desired aesthetic qualities or to suit the functional needs of a design project.

This exploration concentrated on a few basic behaviors and operations that were applied to the subdivisional growth processes. Future work should explore a variety of more specific behaviors that have already been developed for either recursive subdivision or cellular growth. Applications should be tested for more complex architectural projects.

REFERENCES

del Campo, M 2016, 'Moody Objects: Ore Fashion Stores and Blocks', *Architectural Design*, 86(6), pp. 54-57

Carpo, M 2016, 'Excessive Resolution: From Digital Streamlining to Computational Complexity', *Architectural Design*, 86(6), pp. 78-83

Hansmeyer, M 2010a ‘Subdivision beyond smoothness’, *Proceedings of the Sixth international conference on Computational Aesthetics in Graphics, Visualization and Imaging*, pp. 75-81


Hart, G 2009 ‘Growth Forms’, *Proceedings of Bridges*


Klemmt, C and Sugihara, S 2016 ‘Growth Structures’, *Proceedings of TxA Emerging Design and Technology*

Lomas, A 2014 ‘Cellular forms: an artistic exploration of morphogenesis’, *SIGGRAPH Studio*