# Progressions in Defining the Digital Ground for Component Making

Onur Yüce Gün
*KPF NY, Computational Geometry Specialist*
*onuryucegun@alum.mit.edu*

Jonas Coersmeier
*Büro NY, Partner / Design Critic, Pratt Institute*
*jonas@burony.com*

## Abstract

Terms *digital* and *computation*, once accepted as emergent understandings in design, became commonly known and used in recent years. Transformation of techniques from analog to digital created a shift in the understandings as well as products of design. Digital design exploration enabled the designers' exposure to variety and richness. Increasing number of digital tools became easily-accessible. Thus design thinking in both practice and academia was transformed.

Computation, via increasing power and speed of processing, offers mass information execution. Once this power was utilized to inform the discrete pieces of design, *"component making"* quickly became one of the trends in architectural design. Idea of components transformed the enclosing forms of architecture into subdivision surfaces which act as fields for components to aggregate on. While there has been a great interest in creating variety via manipulation of components as individual members, the characteristics of the surfaces became overlooked via common use of parametric (UV) subdivision.

This paper, with a critical look at the current component field generation techniques, focuses on alternative methods of transforming a surface into a digital ground for component aggregation. Series of studies address and deal with various pitfalls of component design and application on software-dictated UV subdivision surfaces. Studies aim to release the component design logic from being software-specific by creation and use of customized digital tools and scripts.

## 1. Variety

Variation plays a crucial role in both conceptual and construction phases of architectural design. In nature, variation is an outcome of a certain condition in nature, thus individual members of a group may differ in some characteristics like topology and functioning[1]. Similar ideas apply in architecture: discrete pieces of a building are designed, manufactured and built depending on their function and their structural behavior.

Computation brings ease in generation and control of variety. Emergent design tools feature easy-to-use component compiling techniques. Once created, components can be aggregated on parametrically subdivided surfaces. Components embody certain intelligence within. They could be manipulated by internal parameters (like numerical values) as well as being driven by behavioral commands, thus they respond to external entities (forces, neighboring elements, proximities and like). While such approach exposes the designers to richness of

possibilities it also promotes geometrical repetition and monotony. Most UV based subdivision engines fail to suffice in going beyond repetitive polygonal compositions as they imply regular two dimensional subdivisions. Besides, over-crowdedness and unperceivable deviation of members entail definition of certain criterion for filtration. Variety stands as neither something that could directly be dismissed from design nor something that could be pushed to the utmost limit without control. Following studies, having a critical look at the foundation of current component-making techniques, offers systems in which variety could be better understood, designed and generated.

## 2. Progressions in component logic

### 2.1. Poromorph: Basic understanding in component design

Poromorph is a five day research study done in alpha version of Bentley's Generative Components as a test user back in 2004. While testing a new parametric design environment, the study aims creation of a component via use of solid modeling techniques and Boolean operations to test behavioral changes of components via regular parametric (UV) distribution on surface.

Generative Components, by default, offers a workflow for transforming surfaces into polygonal grids. A point placed on the surface can easily be transformed into a point cloud, in which the locations of points on the surface are defined by fractional parameters. A series of decimal numbers between zero and one enables regular distribution of the points. This point cloud implies a quadrilateral polygonal grid. The individual polygons in the resulting grid are used as cellular entities, and the components are distributed on top of them.

Poromorph takes this workflow as a precept as the base component is built on a quadrilateral polygon. Five individual solids, whose thicknesses are defined by global parameters feature parametrically adjustable pores on themselves. Variation is achieved mostly via use of the UV parameters, with an additional parameter derived from the steepness values along the surface. Poromorph represents slight variation and behavioral changes driven by parameters and fulfills the very basic ideas about components, however fails in creation and control of variation (Figure 1).

*Figure 1. Poromorph: Variation in solid forms.*

The default surface subdivision workflow of Generative Components, which plays a great role in understanding algorithmic system design, may quickly become a restrictive dictating technique: While the design ideas, forms and surfaces change, the logic of subdivision never changes and the perceivable variety of members ends up being virtual: the stretching of a surface affects the components on the very periphery of the surface, but this doesn't necessarily imply a functional or topological change.

### 2.2. Alternative Subdivision of Surfaces

The default workflow in Generative Components also dictates a one-to-one relationship between the underlying polygonal grid and the component. System does not easily allow the user to break the flawless continuity of series of components distributed along the surface.

A later study focuses on dealing with limiting aspects of direct application of components onto UV subdivision surfaces. It aims to define ways for uneven and discontinuous application of components. While skipping some polygons during component distribution process overcomes the flawless repetition, further subdivision in selected members generates more deviation in the scalar change.

A selective recursive subdivision approach (Figure 2) helps in realization of both ideas. Not all the members are divided in the recursion depth, and not all the polygons are used as component base. A drastic scale deviation occurs in between different recursive depth levels. Components with scale driven properties start showing a wider range of possibilities including deviations in topological characteristics. While the approach produces a valid solution for the initial problems, the products end up being mechanical rather than natural. With no intermediate conditions regarding the scale, the system fails to create fluid translations; thus harmony and continuity is interrupted.

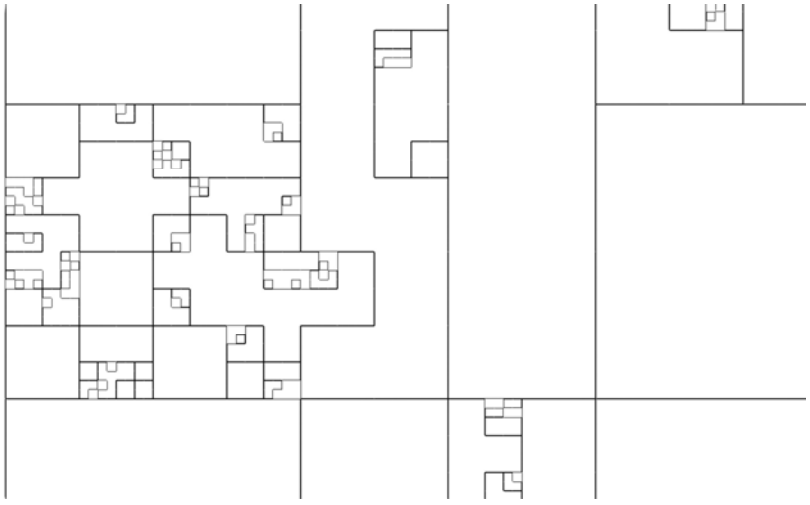

*Figure 2. Selective recursive subdivision of quadrilateral polygons*

### 2.3. Reversing the Component Logic: Nanjing train station

Another preconception about components is accepting the field populations as the only way of aggregating them. The conventional understanding dictates a workflow in which a

compiled component is aggregated on a surface via certain placement rules. Thus surface is always accepted as a mandatory input for any kind of component population. Nanjing Train Station Competition entry of Kohn Pedersen Fox New York rolls over this understanding.

The preliminary massing model of the train station reveals the design intentions: Five hundred meter long platforms lying between the fifteen train tracks are sheltered with individual surfaces of continuous fluid forms. The canopies raise and tear apart in the middle to accentuate the main concourse. The entrances on the north and south are highlighted by projections. While the train station represents a global harmony, the individual surfaces feature specific identity, as they are generated with smaller bits of information: Various configurations, deformations and transformations of simple “S” curves, driven by global rule sets and internal parameters for local adaptations define the characteristics of the surfaces (Figure 3). Generated as a derivative of a series of discrete invisible components, the “S” curves, this time the surface becomes the product of series of components.

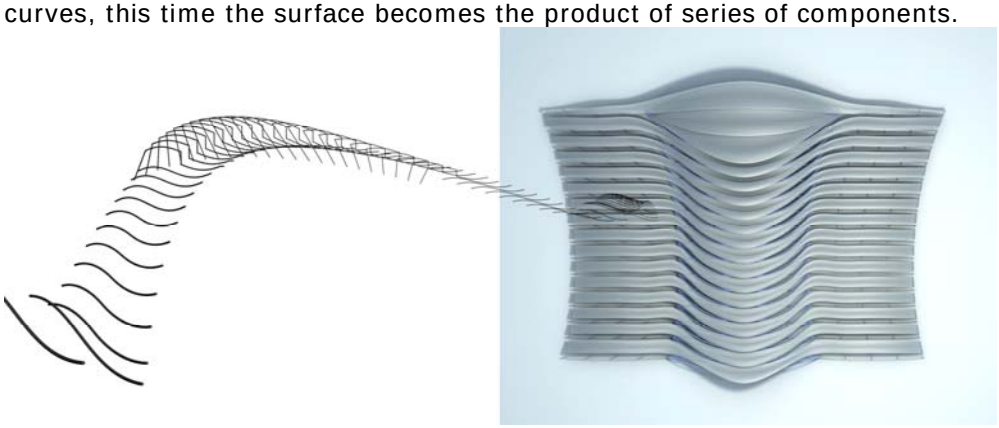

*Figure 3. Invisible components influencing the form*

## 3. User defined subdivision techniques

### 3.1. Inspiration: Studio at Pratt Institute

In the light of concerns above, in the second year undergraduate studio we ran at Pratt Institute in spring 2007, we aimed to propose an alternative way of thinking in creation of digital grounds for designing with components. While supporting the studio with necessary theoretical background for student’s development in the conceptual understanding of design computation we also helped students develop their technical skills in use of parametric modeling environment. From a range of possibilities, regarding student’s experience and strength of the platforms, we used Rhino and of Generative Components throughout the semester.

The studio ran through stages of studies which encouraged students to develop analytical systems to generate architectural envelopes and enclosures by deriving data from the project site. Students were expected to develop systems which are driven by their design concepts and solidify their ideas by creating geometrical aggregation models.

The activities taking place around the given site in Bowery, New York became the medium for students to start with as they recorded and mapped these activities. While mapping, students used line drawings and point clouds to express and re-construct the connections

between activities and to relate them to their design concepts. This very geometrical composition, derived from the site was used as an input for compiling polygonal meshes, which played the role of the digital ground and created the back bone for their further studies in the studio.

### 3.2. Mesh as a Digital Design Ground

A polygonal mesh is a composition of three different geometrical entities: First the vertices, which is a collection of points in space, second, the edges that connect these vertices and the third, the faces that fill in between the edges[2]. In computer graphics, meshes are generated via use of different algorithms, by taking different geometrical entities as inputs. A group of points as well as a surface could be the inputs for mesh algorithms. The characteristics of the output is also defined by the algorithm: A mesh could be composed of only triangular polygons, or only quadrilateral polygons, or could be a product of various concave and convex polygonal compositions.

Perceived as a digital ground for component making, we attribute a new character to polygonal mesh. Mesh is no longer accepted only as a digital representation of an enclosing geometry, but rather considered as a space constructor that is composed of interconnected responsive parts. Elements of mesh, with this logic become “cells”, vertices, edges and surfaces become the part of design idea. Instead of representing a global form, they become placeholders for various architectural components.

A polygonal mesh, depending on the algorithm of origin, may create a polygonal composition in which the polygonal members change tremendously in scale. Polygons in various sizes address resolution and density. When controlled sensibly, this capability becomes a design understanding: While the placement of points in a Delaunay triangulation[3] algorithm may define the resolution by activity, a mesh extracted from a bent surface may indicate curvature. Similar approaches and understandings may be used together or independently (Figure 4).

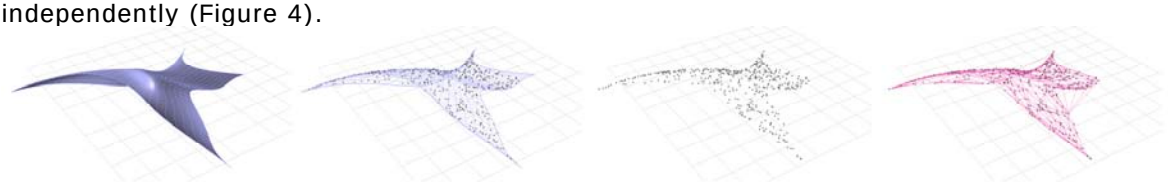

*Figure 4. Point cloud projected on a surface is then triangulated with Delaunay algorithm*

A mesh delivers the availabilities related to variety, scale and topology and breaks the limitations that are dictated by UV subdivision techniques. However, custom digital tools are needed to realize this conceptual understanding, which are not yet available through the software packages.

### 3.3. Essence: The Tool

Generative Components, while bringing ease in application of repetitive actions, is not an open platform since it doesn’t accept geometric information to be imported in by use of conventional import and export commands. Since it is a dynamic parametric platform, all the geometry has to be constructed and manipulated in GC itself. However it accepts data sets, which makes regeneration of models generated in different software platforms via use of numerical references in GC.

McNeel's Rhino, by default and by open source materials features many different meshing algorithms, which enables the designer to design and configure the meshes in desired way. But to use the meshes as digital grounds for component population, certain level of scripting knowledge is a prerequisite. Regarding the different strengths of different platforms, and the interest in re-interpretation of a digital ground for component making, the motivation to have a set of custom tools to generate a smooth work flow between cross platforms emerges.

To enable transformation of a Rhino mesh into a digital generative ground in Generative Components, a series of scripts are prepared. The meshes modeled in rhino are transformed into two numerical data sets. While the first one stores the coordinate values of the mesh vertices, second set stores the information of the way the vertices are grouped to generate the mesh polygons. The data sets are then exported into spreadsheets.

The two data sets are read into GC via customized scripts. While vertices are represented as individual points each mesh polygon becomes a responsive generative base on which components systems could be applied. The digital ground in GC, now represents vast amount of variety in scale, size, density, curvatures compared to basic UV surface subdivision systems existent in GC (Figure 5).

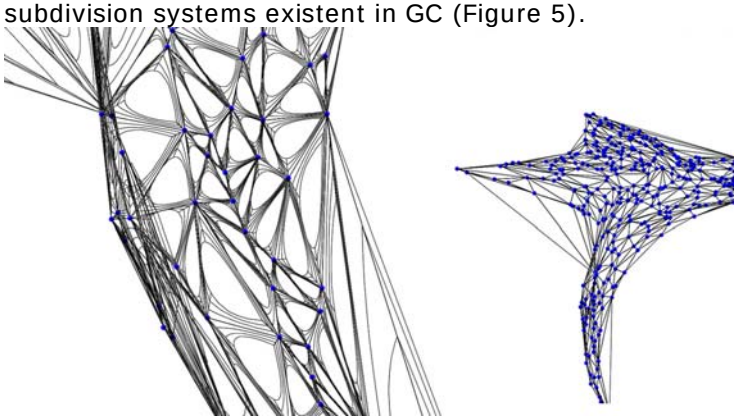

*Figure 5. Variety in scale: Digital ground in GC*

### 3.4. Design Cycle

A proposed idea of compiling of components that typologically change depending on scalar deviations is tested during the studio sessions at Pratt. Once the mesh is imported into GC, designer creates components that are driven by internal and external parameters, but mainly are dependant to the scale: While the components become surface elements in high curvature values, or become less sophisticated in detail as they get smaller, they become articulated spatial entities as they get larger (Figure 6).

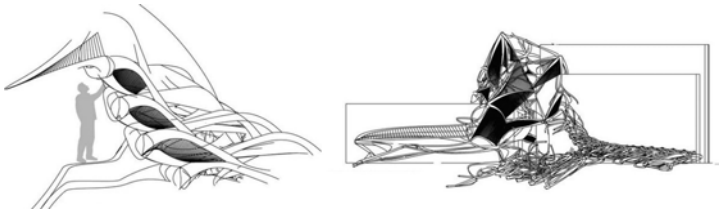

*Figure 6. Architectural interpretation: Bradley Rothenberg (Pratt Institute)*

The test grounds once populated by components can be evaluated and then manipulated by use of initial mesh algorithm. While an evaluation and re-generation cycle develops between cross-platforms, the design becomes concept-driven rather than falling into the trap of being software-specific.

## 4. Epilogue: Further Investigation

The sequence of studies concluding with the final mesh creation approach proposes an open system for creation of digital grounds for designing with components. The richness of the approach lies in this openness: A mesh could be generated in many different ways by use of many different algorithms. Such availability releases the designer from the limitations of software-dictated design understandings. In the specific case of mesh generation, the design concept defines which meshing algorithm to chose and the selected method affects the characteristic of the polygonal mesh in organizational means, thus the surface and spatial compositions and subdivisions become products of design *intentions*[4].

The work in progress includes development of several scripts to classify and group certain mesh polygons depending on design intentions. Orientation, proximity to attraction points, scale or material properties could most definitely be considered as some of the possible influencing drivers for design. While some other studies focus on different techniques of point cloud creation as an input for meshes, some parallel studies aim to develop custom tools to increase the ease of manipulation of meshes as digital grounds in GC.

**Sources:**

[1] See Hensel M., Menges A. and Weinstock M. *Techniques and Technologies in Morphogenetic Design,* Great Britain: Wiley-Academy, 2006.
[2] See Portman H., Asperl A., Hofer M. and Kilian A. *Architectural Geometry*, United States of America: Bentlye Institute Press, 2007, pp. 381-396.
[3] See de Berg M., van Kreveld M., Overmars M. and Schwarzkopf O. *Computational Geometry: Algorithms and Applications,* Germany: Springer, 2000.
[4] See Kostas, T. *Algorithmic Architecture*, Great Britain: Architectural Press, 2006, pp. 25.