Parametric Voxel Geometry Control for Digital Morphogenesis

Thomas Fischer
Spatial Information Architecture Laboratory
Royal Melbourne Institute of Technology
Melbourne, Australia
sdtom@polyu.edu.hk

Torben Fischer
Mathematische Fakultät
Georg-August-Universität Göttingen
Göttingen, Germany
torben.fischer@stud.uni-goettingen.de

1 Abstract

We discuss the extension of an experimental 3D voxel-automata system for generative (evolutionary developmental) design with means for parametric geometry control on the scale of single voxel units. The objective of this extension is to allow the automata system to support the generation of *free form*. Being the result of a longer-term interdisciplinary software development project, the voxel automata system allows simulations of three-dimensional, computation-universal cellular structures and behaviours based on decentralised, massively parallel, programmable units. It represents a kit of parts in which virtual form, programme and data structure are fused. From this top-down perspective we demonstrate a number of geometric operations we have identified and show some early results. We also give an outlook into more challenging future developments.

2 Zellkalkül

The software is named *Zellkalkül* in reference to Konrad Zuse's *Plankalkül* and his early reflections on spatial calculation. It is designed to facilitate explorations of tempo-spatial mechanisms of morphogenesis in cellular (voxel-based) architectural and design contexts such as theoretical architectural design research [3] as well as in generative design teaching [2]. *Zellkalkül* differs from classic cellular automata systems in terms of its programming logic as well as geometrically. Its programming logic supports non-uniform high level coding. That is, different cells can be equipped with individual code scripts. The scripting language used is an extended version of ECMAScript. Besides its general control structures and data types, which are commonly known from other ECMAScript dialects such as JavaScript, our system also provides purpose-centered functions and objects to support intercellular communication and developmental actions. These are partially inspired by biological epigenetic processes (splitting, differentiating, moving, dying) and partially intended to support generic actions (evaluating fixed identities, exchanging and modifying code scripts etc.). The cellular voxel units are based on rhombo-dodecahedral geometry in close-packing arrangement.

We prefer this topology, derived from face-centered cubic close-packing of spheres, over the square or cubic arrangement of common 2D or 3D cellular automata systems since it allows equal distances and relationships between all neighbouring cells (as discussed and exemplified in [4]). Moreover, it bears a close resemblance to many natural cell tissues as identified and illustrated early by [6] (see left of figure 1). The arrangement is equivalent to the *isotropic vector matrix*, which, used for example to form so-called *octet trusses*, is of special relevance to architecture and structural engineering (see [5], p.138 ff.). At the time of this writing, *Zellkalkül* supports 3D rendering of this data structure in form of "solid" spheres and as rhombic dodecahedra. Vector

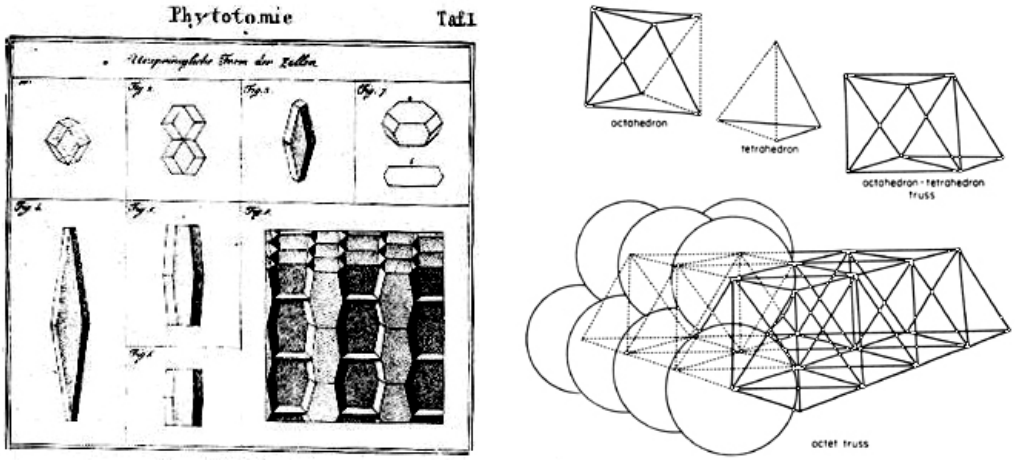

Figure 1: Rhombo-dodecahedral cell geometry by [6] (left) and octet truss by [5] (right)

representation similar to the illustration on the right of figure 1 is planned.

3 Towards Supporting *Free Form*

We believe that *Zellkalkül*'s geometry is more flexible than traditional square or cubic cellular automata systems. However, from an architectural design viewpoint, this current spatial/cellular automata structure still shows a particular shortcoming with respect to its formal expressive capabilities. Geometrically, generated tissues are largely restrained to the system's homogeneously close-packed lattice structure and generate "jagged" forms only. Support of forms with smooth surfaces, straight edges and so forth (or architecturally speaking: *free form*) is being developed at this point and it is the purpose of this paper to explore possible solution strategies. Our interest is to complement customary surface and solid modelling techniques with a different operational mode that embraces developmental morphogenesis in form finding. Approaches that could in principle qualify to allow the generation of free forms in such a way include:

1. Using large numbers of automata at a high resolution to approximate smooth 3D shapes. One disadvantage of this approach is that smooth surfaces are not achieved, only approximated at the cost of exponentially increasing memory consumption during shape generation. Formal expression is moreover achieved primarily by means of additive composition and not by parametric control of geometric relations. The usefulness of both parametric design principles in combination has been discussed by [7].

2. "Skinning" of cell assemblies using curve fitting algorithms.

3. "Skinning" using cell centre points or cell attributes to control skin geometry, e.g. mapping cell location or other cell-related data onto free curve control-points.

4. Mapping of data generated in *Zellkalkül*'s automata structure onto secondary output geometries. This and the above two strategies however contradict the software's intention to provide a single unified geometric and data structure.

5. Parametric position control of cell polygon vertices. We will discuss this approach in most of the remaining part of this paper.

6. Parametric cell sizes control. This appears to a highly interesting future extension. We have however not yet been able to identify suitable operations and constraints for this approach (see section 5.2).

7. Changing the distances between cell centres, i.e substituting the isotropic vector matrix by an "anisotropic vector matrix". This approach is a combination of the above two approaches in which tissues are represented in form of vector trusses instead of as "solid" cells.

Our first approach towards parametric manipulation of the fourteen cell vertices involves the following initial steps. First of all, the rhombic cell faces are triangulated in order to allow vertices to move individually without affecting other vertex positions of the same cell while continuing to allow tissues to remain close-packed without void or overlapping spaces. This triangulation is also useful in facilitating the creation of triangle-based output file formats such as STL, which is useful in producing stereo-lithographic rapid prototype models of generated form. For this purpose, rhombic faces are simply split into two isosceles triangles. As a next step it is necessary to identify the movement ranges of all vertices — the spaces or domains within which each vertex is allowed to move while avoiding vertex eversions (which would describe unwanted *inverse spaces*). This eversion problem would for instance occur if vertices {1} and {9} on the right of figure 4 were to change their positions in such a way that {9} will be located above {1}; the cellular space between both vertices would evert and result in an undefined space. A last step in this preliminary study requires a suitable control mechanism, which will allow users to move vertices in intuitive ways using scripting functions. Before discussing these control mechanisms, we will first proceed to describe the identified vertex ranges.

To allow vertices to move freely without mutual eversion, the ranges must occupy the entire tissue space while not overlapping and hence themselves allow close packing. The left of figure 4 shows a rhombic dodecahedron with faces numbered using Miller indices (a face identification system adopted from the field of mineralogy), vertex numbers as used in *Zellkalkül* and two different vertex types labeled 'a' and 'b'. Vertices between six adjacent cells (labeled 'a') have octahedral ranges while vertices between four adjacent cells (labeled 'b') have tetrahedral ranges. The side length (or in Fuller's terminology: the geodesic vector length) of both geometries is 1 (identical to cell diameter). Figure 2 shows both types of these *platonic shapes* (left, octahedra are only shown half) and (right) their ability to close-pack into cuboctahedral assemblies. The twelve cells neighbouring the central cell are shown as wire frames in both images. By constraining vertices to remain within their ranges, this topology guarantees that self-intersecting cell surfaces are avoided.

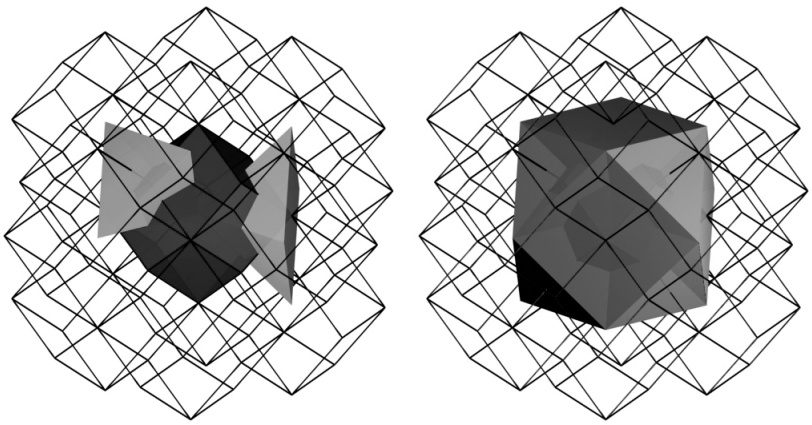

Figure 2: Cuboctahedral vertex range clusters defined by tetrahedral and octahedral (half shown) vector matrices

Though the coboctahedral geometry shown in light grey on the right of figure 2 as such does not allow space-filling close packing, this arrangement still occupies complete tissue spaces due to partial overlapping. The reason for this overlapping is that ranges are associated with multiple cells that have vertices in common.

4 User Interface

The user interface for vertex position control should be as simple, intuitive and yet as flexible as possible. For this reason, from the user's perspective, it does not distinguish between the two different vertex types 'a' and 'b' even though they are internally processed in different ways. We have decided to use a *pressure* model that allows users to control the pressure cells bear in the direction of a given vertex. Figure 3 shows the force vectors with which adjacent cells can manipulate both types of vertices. An advantage of using a pressure model for vertex position control is that positions, as in Nature, result relatively from intercellular "negotiation" rather than by unilaterally controlled, absolute positioning. A problem emerges when trying to move a vertex between four cells (shown on the left of figure 3) by means of four pressure vectors in caltrop arrangement (see inside tetrahedral range) since this does not allow the vertex to reach any point within this range. We have solved this problem by modelling forces as negative pressure (or: "tension") rather than as "pressure".

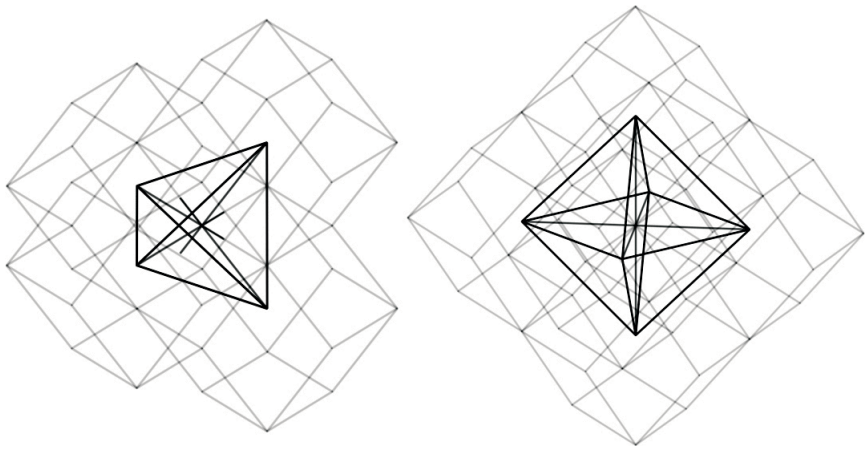

Figure 3: Force vectors within vertex ranges

The user interface is represented by the script interpreter associated with every cell. For this purpose we have extended the scripting language specification with functions that allow the identification of a vertex and a (negatively interpreted) pressure in the format `setTension(v, p)` to set a pressure and in the format `getTension(v)` to acquire a pressure. The pressure parameter is expressed as a byte value which allows relative pressure control at a resolution of 255 steps per cell involved in a vertex positioning operation. The neutral default pressure of each vertex, as well as the "atmospheric" pressure (relevant at tissue edges where vertices have no neighbours), is 127. With this default value assigned to all vertices internally and to vertices of adjacent cells, a cell assumes a normal rhombic dodecahedral shape. Other values result in "morphed" variations as shown in image 5.

5 Outlook

This section discusses further system extensions that, within this ongoing project, are of interest to us but for which we have not yet been able to identify appropriate algorithms and constraints. Firstly, the fixed ranges described in the above section appear to be rather limiting compared to the possibility of *dynamic ranges*. Secondly, *flexible cell sizes* would be of great value for generating geometrically non-uniform tissues and structures.

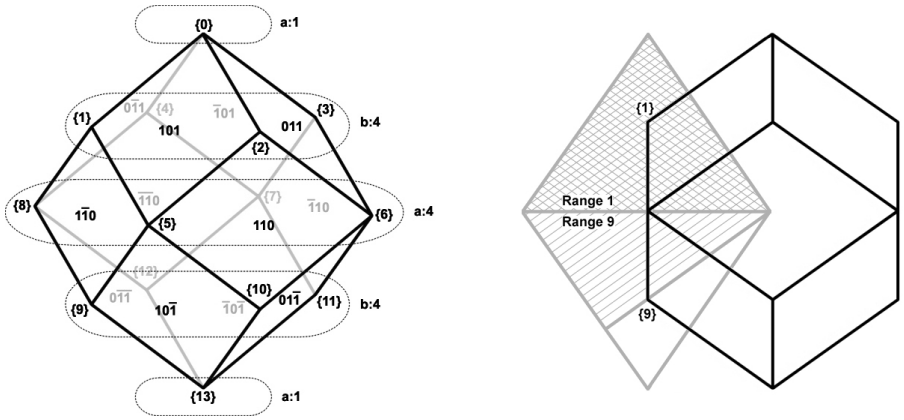

Figure 4: Miller face indices, vertex numbers and vertex layers (left), vertex ranges 1 and 9 in front view (right)

5.1 Dynamic Vertex Ranges

The ranges (vertex domains) discussed above are rigidly defined as *platonic* tetrahedra and octahedra. With these two types of ranges, vertices are not yet given the maximum possible ranges of movement as shown in the illustration on the right of figure 4. Given that vertex {9} was to remain its shown default position, the tetrahedral range of vertex {1} (double hatched) could potentially be extended by as much as half of the range of vertex {9} (single hatched). The three-dimensional interrelatedness of all vertices of a cell and its neighbours suggests a solution strategy that is based on a process of recursive approximation. This approximation should make highly economic use of physical machine resources. In tissues with large numbers of cells (tissues with up to 20,000 cells have been generated and larger ones are likely to be generated in the future) such operations are likely to jeopardize the system's present interactive responsiveness. It was noted by [1] that in the translation from idea to a product, it is in this responsiveness, where one of the key values of parametric design lies.

5.2 Flexible Cell Sizes

Especially from a structural design point of view, a function to generate geometrically non-uniform structures (cell tissues, octet trusses etc.) is enticing. This will require (again virtual pressure-based) geometric control algorithms for variable automata diameters and consequently for variable centre point locations. For such a function, however, a non-uniform 3D close packing system will not be sufficient (an early investigation into this possibility was presented by [8]). In order to maintain *Zellkalkül*'s intercellular communication infrastructure, such a system needs to be constrained in a way that always preserves the number of twelve cell neighbours. An alternative solution could be based on intercellular communication facilities that are designed for variable numbers of neighbours. Since this would however result not only in structurally chaotic forms but also in a highly untidy organisation of the user scripting interface, a solution based on geometric constraints would be preferable.

6 Acknowledgements

We gratefully acknowledge the support and advice from our colleagues and teachers at the School of Design at the Hong Kong Polytechnic University, at the Spatial Information Architecture Laboratory at the Royal Melbourne Institute of Technology and at the Faculty of Mathematics at the University of Göttingen, in particular Prof. John Frazer, Prof. Mark Burry and Timothy Jachna.

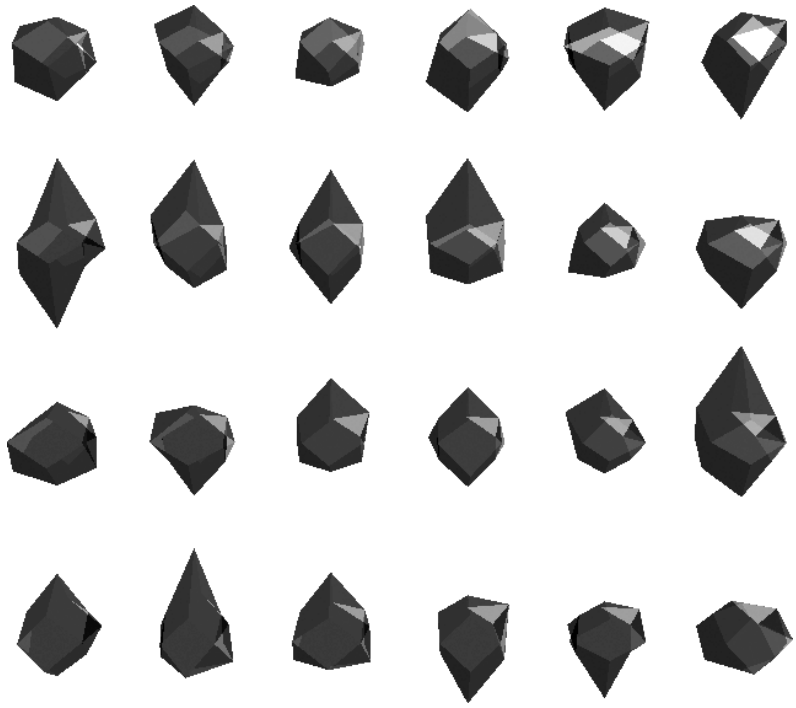

Figure 5: Array of parametrically morphed cells

References

[1] Mark C. Burry and Zolna Murray. Architectural design based on parametric variation and associative geometry. In *Challenges of the Future. Proceedings of the 15th eCAADe Conference*, pages 1–11, Österreich Kunst und Kulturverlag, Vienna, Austria, 1997.

[2] Thomas Fischer. Computation-universal voxel automata as material for generative design education. In Celestino Soddu et al., editor, *The Proceedings of the 5th Conference and Exhibition on Generative Art 2002*, pages 10.1–1.11, Generative Design Lab, DiAP, Politechnico di Milano University, Milan, Italy, 2002.

[3] Thomas Fischer, Torben Fischer, and Cristiano Ceccato. Distributed agents for morphologic and behavioral expression in cellular design systems. In George Proctor, editor, *Thresholds. Proceedings of the 2002 Conference of the Association for Computer Aided Design in Architecture*, pages 113–123, Department of Architecture, College of Environmental Design, California State Polytechnic University, Pomona, Los Angeles, 2002.

[4] John H. Frazer. *An Evolutionary Architecture*. Architectural Association, London, 1995.

[5] R. Buckminster Fuller. *Synergetics. Explorations in the Geometry of Thinking*. Macmillan Publishing, New York, 1975.

[6] D. G. Kieser. *Phytotomie, oder Grundzüge der Anatomie der Pflanzen*. Cröcker, Jena, 1815.

[7] Branco Kolarevic. Digital morphogenesis and computational architectures. In Jose Ripper Kos, editor, *The Proceedings of SIGraDi2000 - Construindo (n)o espacio digital (constructing the digital Space)*, pages 98–103, Facultad de Arquitectura - Universidad Nacional de Mar del Plata, Rio de Janeiro, Brazil, 2000.

[8] Gary Kong. Cellular automatas and "stacking balls". Technical dissertation forming part of the Diploma of the Architectural Association, London, 1994.