Generative Sculpture by Evolutionary Design

Leo S. Bleicher
San Diego, CA : l_bleicher@pacbell.net

Abstract

Complex and varied shapes can be generated by applying sequences of coordinate transformations to a cube. Previous work has demonstrated the ability of the polar coordinate transformation used in conjunction with standard symmetry operations to create unusual and interesting forms. The appearance of these forms depends on the operations performed and their order. In this paper, the use of three-dimensional transformation operations, various options for visualization, and an evolutionary algorithm for exploring these possibilities are described.

Introduction

Previous communications [1, 2] outlined some of the forms attainable by repeated geometric transformation of simple shapes. Use of the standard symmetry operations of rotation, translation, scaling, and reflection in the creation of patterns and tilings is one of the most ancient decorative methods. While repeated application of symmetry operations leave a shape unchanged in terms of angles and boundaries, interleaving these operations with symmetry-breaking transformations can rapidly lead to visually complex forms. In the previous studies two-dimensional coordinate transforms such as polar, inverse polar, and circle inversion were used exclusively as the symmetry breaking operations. In this paper these ideas are extended into three dimensions yielding virtual sculptures as the output.

As transformation sequences become longer and the outputs more complex, two artistic considerations need to be handled: sequence selection and visualization. Due to the combinatorial explosion of possible sequences, exhaustive exploration quickly becomes impossible. While random generation of transformation sequences may generate interesting results, a better method is to use a directed search of the sequence space based on the aesthetic or novelty value of related sequences. One of the best methods for directing this type of search is an evolutionary algorithm [3, 4]. Additionally, working in three dimensions requires viewing the resulting objects from multiple points of view to yield a full understanding.

The Coordinate Transformations

As described by Greenfield [5], the coordinate transforms used in this work map point in a region to new locations in that region into itself via a function. For example, the polar transformation is formulated as

$P : (\theta, r) \rightarrow ((1 + r \sin(2\pi\theta))/2, (1 + r \cos(2\pi\theta)/2))$, transforming vertical lines into lines through the origin and horizontal lines into concentric circles. Likewise, the spherical transformation is formulated as

$S : (\theta, \varphi, r) \rightarrow ((1 + r \sin(2\pi\theta) \cos(\pi\varphi))/2, (1 + r \sin(2\pi\theta) \sin(\pi\varphi))/2, (1 + r \cos(2\pi\theta)/2))$.

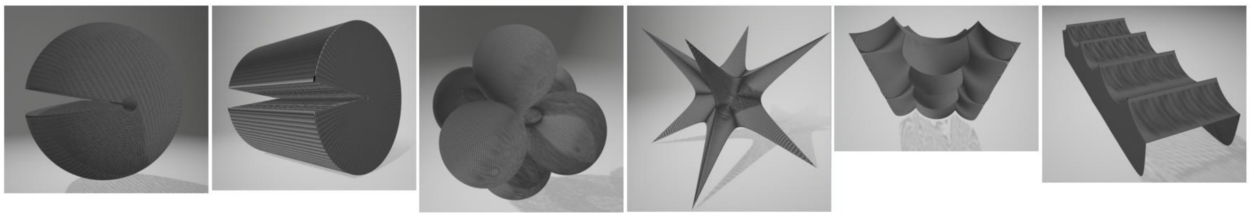


Figure 1: *Spherical, cylindrical, sphere and cylinder inversions, inverse spherical and cylindrical operations on a cube. Cube initialized with side length of 0.95 to highlight mapping of outermost region.*

Additional transformations used here include mirroring, inverse spherical, cylindrical, inverse cylindrical, and sphere inversion. These transformations alter the Euclidean distances between points, the orientation of lines, and the curvature of surfaces. Figure 1 shows results from selected transformations of a cube.

Generation and Rendering of Virtual Sculptures

The process used for realizing the outcome of a sequence of transformations of a cube is implemented using scripting and Blender as follows. A set of evenly spaced points on the surface of a cube (typically 60,000) is generated and triangular faces are constructed to cover the surface. Using PilotScript within the scientific visual programming system Pipeline Pilot, the appropriate transformation is carried out, mapping the points to new positions in space, and the resulting shape is rescaled to the maximum size still fitting in the cube bounded by -1 and 1 in all dimensions. Resulting faces with area exceeding 0.2 square units are replaced with three smaller faces by adding a new vertex at the centroid of the original face. The result is fed into the next transformation. Once all transformations complete, the data from this process is written to a .ply file describing each triangular face in the final shape. This file is used by Blender to render high quality visualizations with a wide array of lighting, and textures as well as animations [6, 7]. Figure 2 shows Blender generated renderings of such data.

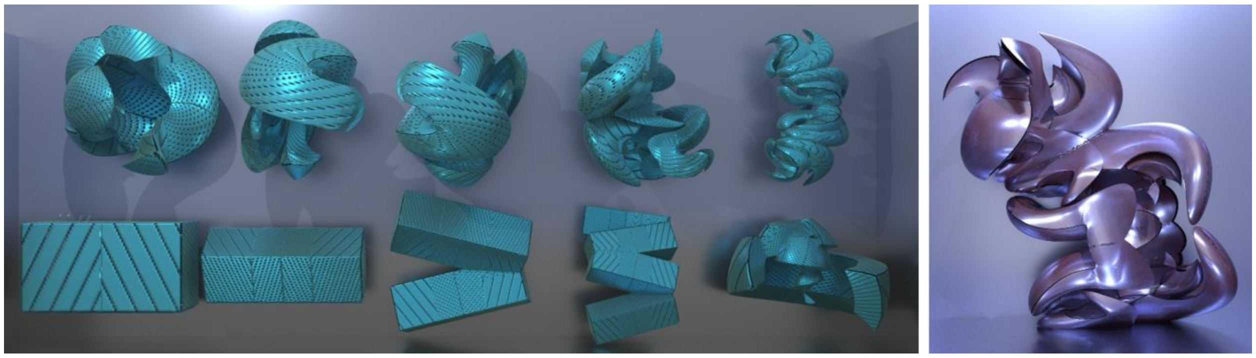


Figure 2: *Left: rendering of a 10 step sequence. Selected faces are omitted (darker regions) to illustrate how the original surface became stretched and warped. Sequence runs bottom left to top right. Right: final view with positioning, surface texture and lighting.*

Genetic Algorithm for Exploration of Transformation Sequences

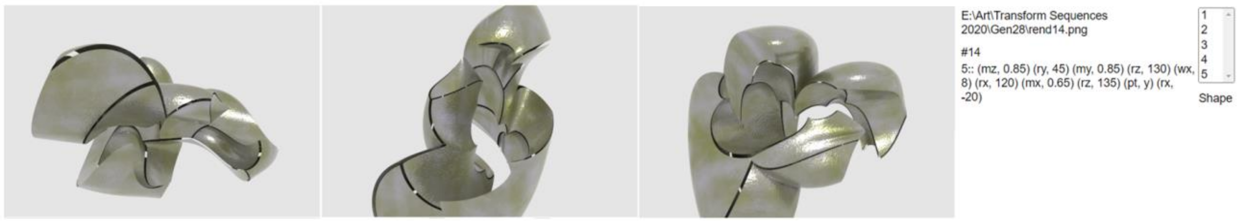
The number of possible short transformation sequences is staggering. In this work, I have explored sequences in the range of 6 to 20 steps. Only even numbers of steps are used and the sequences are constrained to alternate between operations that preserve neighborhood characteristics (such as rotation) with operations that alter them (such as spherical). Further constraints on the rotation steps to multiples of 15º yield 420 options per pair of steps. Obviously, even the shortest sequences are far too numerous to exhaustively generate and review (420^3 > 74 million), and longer sequences are exponentially worse.

Table 1: *Example transformation sequence coding showing operation codes and operation parameters.*

Step	1	2	3	4	5	6	7	8	9	10	11	12
Operation	s2	rx	pt	rz	s2	rx	to	rx	mz	rz	wy	rz
Operation Parameter	none	0	z	105	none	10	x	60	0.65	150	0	75

To quickly identify sequences of interest, a genetic algorithm has been employed. Each transformation in the collection is represented by a short code allowing the sequence for generating a particular sculpture to be captured by an array of such codes. Table 1 shows an example of a 12 step sequence. The initialization of the process uses random, maximally dissimilar, short sequences. Sets of sequence codes can easily be recombined, mutated or drifted by the genetic algorithm. Over time the sequences are allowed to lengthen.

In addition to a coding scheme, a genetic algorithm requires a fitness function that can influence the level of participation of each sequence in the population of the next generation of sequences. To assign the reproductive fitness to each individual sequence I have simply looked at each image from a generation of sequences and assigned a score based on my evaluation of its aesthetic quality or novelty. These scores are used in the recombination process to bias the population of parent sequences. Scoring is done in batches of three to five hundred sculptures with values of 1-5 assigned to each as shown in Figure 3. To strengthen the effect of higher scores, the cube of the value becomes the number of chances a sequence has (out of the sum of the cubes of all members of that generation) to be one of the two parents of each sequence produced in the subsequent generation.



```
E:\Art\Transform Sequences
2020\Gen28\rend14.png

#14
5:: (mz, 0.85) (ry, 45) (my, 0.85) (rz, 130) (wx,
8) (rx, 120) (mx, 0.65) (rz, 135) (pt, y) (rx,
-20)
```

Figure 3: *Browser-based interface for scoring sculptures. Three orthogonal views are presented along with a scoring element and the code of the generating sequence. Surface textures are scored separately.*

Over several generations, certain operations emerge as dominant in specific parts of an appealing sequence. For example, in the 14th generation of a recent run the spherical operation is extremely common as the first step, but extremely rare as the final step. Conversely, cylindrical shows up mainly in the later sections of sequences, being the final transformation in nearly 1/3 of the sequences of this generation as shown is Table 2. Further, occasional motifs emerge, in the form of frequently observed subsequences, which yield interesting building blocks or finishing flourishes. A future consideration would be to reduce the probability that the recombination crossover point falls within one of these preferred motifs, thus allowing it to spread more quickly through the future generations of sculptures.

Table 2: *Prevalence of selected operations at each step in every sequence of the 14th generation of a recent run. The total count declines in later steps as short sequences are completed.*

Operator / Sequence #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Spherical	373		58		48		31		36		14		4		4		2		2	
Mirror Z-axis	29		43		33		25		29		10		11		5		5		2	
Mirror X-axis	11		135		74		74		37		36		15		12		11		8	
Torroidal	5		29		29		33		31		33		21		18		8		5	
Polar	3		9		10		11		13		13		12		4		2			
Mirror Y-axis	1		20		47		33		32		31		12		12		5		3	
Rotate X-axis		312		193		143		127		119		85		50		29		21		12
Rotate Z-axis		98		186		192		189		160		130		106		67		38		28
Rotate Y-axis		23		54		98		96		85		69		57		35		25		15
Sphere Inversion			57		30		29		14		13		6		4		4		1	

Observations, Summary, and Conclusions

Beyond the aesthetic qualities of the virtual sculptures generated by this approach, there are several elements that I find compelling. Least surprising is the production of families of related sculptures. These come about through high rating of sculptures differing only in rotation parameters and having otherwise identical transformation sequences where several members of the family have a pleasing balance of complexity, flow, and symmetry. Interpolation between the members of such families, however, yields surprisingly few forms that have different but pleasing arrangements. Nevertheless, animations of parameter space traversals are interesting [6, 7].

Generative methods for producing art typically use randomness or unrelated sources to make choices, constraining the work in the details of the algorithm. The work described here instead makes use of a process with a wide scope and, in every place where randomness is used, including initial population generation, parent selection, and crossover point selection, it is filtered, biased, and constrained as much as possible to allow the influence of the artist to be exerted. Thus, different people using the same algorithms and processes would find different designs and motifs. Some examples are presented in Figure 4.

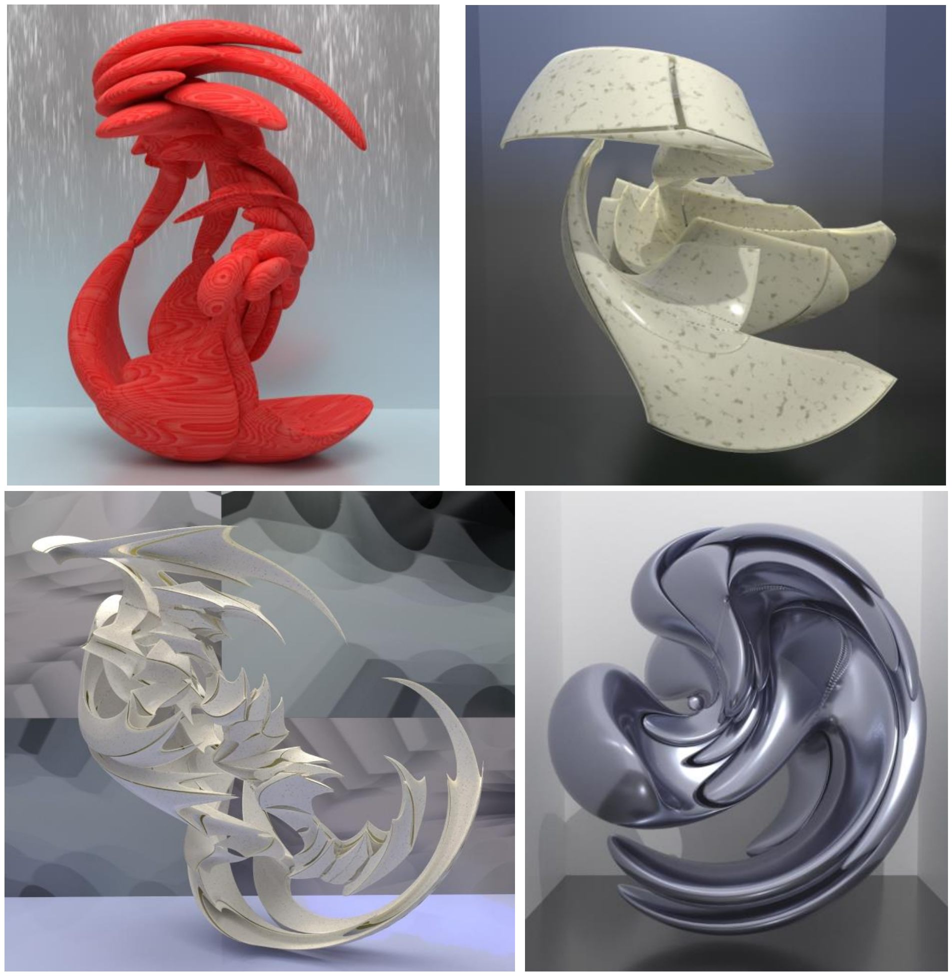


Figure 4: *Some examples of sculptures generated by the sequential transformation method.*

References

[1] L. S. Bleicher. "Serial Polar Transformations Of Simple Geometries". ISAMA–CTI 2004 Proceedings (ed. S. Luecking), 2004, 65–68. http://www.mi.sanu.ac.rs/vismath/bleeicher/index.html

[2] L. S. Bleicher. "Serial Polar Transformations Of Simple Geometries II". *Hyperseeing* Fall 2011, 25–30. 11c.pdf (isama.org)

[3] K. Sims. "Artificial Evolution for Computer Graphics." Proceedings SIGGRAPH '91, vol. 25, pp. 319–328.

[4] S. Todd and W. Latham. Evolutionary Art and Computers. Academic Press, 1992.

[5] G. R. Greenfield. "Serial Polar Transformation Motifs Revisited". *Bridges 2005 Proceedings*, 443–448. The Bridges Archive: 2005 Paper (bridgesmathart.org)

[6] A growing collection of generated sculptures is at https://www.instagram.com/bleicherleo/

[7] For animations in 2D and 3D see http://www.youtube.com/user/solporter?feature=mhum