Distance-based iterated function systems on surfaces

Filippo Maggioli
maggioli.filippo@gmail.com
Pegaso University
Naples, Italy

Valerio De Luca
valerio.deluca@unipegaso.it
Pegaso University
Naples, Italy

Andrea Generosi
andrea.generosi@unipegaso.it
Pegaso University
Naples, Italy

Leonardo Galteri
leonardo.galteri@unipegaso.it
Pegaso University
Naples, Italy

Luigi Gallo
luigi.gallo@unipegaso.it
Pegaso University
Naples, Italy

Abstract

Iterated Function Systems (IFSs) are a classical tool for procedural fractal generation, but their formulation is strongly bind to the underlying coordinate system, and thus limited to flat Euclidean domains. We introduce an alternative formulation for an entire family of IFSs, exclusively based on the notion of distances and geodesic paths, that allows defining systems directly on Riemannian domains, enabling the procedural generation of fractal patterns that are aware of the underlying geometry.

CCS Concepts

• **Computing methodologies** → **Texturing**; **Agent / discrete models**; *Massively parallel algorithms*; • **Theory of computation** → *Computational geometry*.

ACM Reference Format:

Filippo Maggioli, Valerio De Luca, Andrea Generosi, Leonardo Galteri, and Luigi Gallo. 2026. Distance-based iterated function systems on surfaces. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Posters (SIGGRAPH Posters '26), July 19–23, 2026, Los Angeles, CA, USA*. ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3799825.3818722

1 Introduction and related work

Procedural texturing is a key topic in Computer Graphics, and is of fundamental importance for creating detailed and realistic materials [Ebert et al. 2002]. Among the various approaches, solid texturing [Hart 2001; Maggioli et al. 2022a; Worley 1996] allows defining functions in the embedding space and restricting their evaluation to the surface, easily generalizing to arbitrary shapes and representations. On the opposite side of the spectrum, textures can also be defined as patterns emerging directly onto the surface [Maggioli et al. 2022b; Nazzaro et al. 2021] or as vector graphics in the geodesic metric [Mancinelli et al. 2022; Marin et al. 2024; Riso et al. 2022], with shape and structure strongly related to their intrinsic geometry. In a similar fashion, decals provide a versatile tool to decorate the surface with arbitrary images and patterns [Rustamov 2010]. We introduce a novel approach for generating fractals directly on surfaces through iterated function systems. Unlike decals, our method is strongly bound to the surface metric, resulting in a better preservation of the recursive structure of the fractal shape.

SIGGRAPH Posters '26, Los Angeles, CA, USA

ACM ISBN 979-8-4007-2548-7/2026/07
https://doi.org/10.1145/3799825.3818722

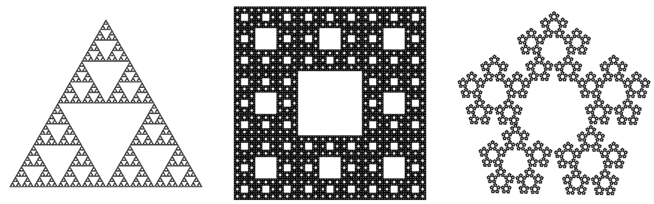

Figure 1: From left to right: the Sierpinski triangle; the Sierpinski carpet; the optimally packed regular pentagon.

2 Method

An iterated function system (IFS) is a finite collection of contraction mappings $\{f_i : \mathbb{F} \to \mathbb{F}\}_{i=1}^{k}$ on a metric space $\mathbb{F}$. IFSs have been extensively used in the context of fractal generation, due to their property of converging to a fixed compact set [Barnsley 2014]. A notorious IFS is the generator of the Sierpinski triangle (fig. 1), namely $f_i(\boldsymbol{x}) = \frac{(\boldsymbol{x}+\boldsymbol{p}_i)}{2}$, where $\boldsymbol{p}_i$ are the vertices of an equilateral triangle. This can be generalized to a wider family of IFSs, including the Sierpinski triangle, where the IFS is defined through a collection of attraction points $\boldsymbol{p}_i$ and attraction factors s_i.

$$f_i(\boldsymbol{x}) = (1 - s_i)\boldsymbol{x} + s_i\,\boldsymbol{p}_i, \quad i = 1, \cdots, k,\ s_i \in (0, 1)\,. \tag{1}$$

This definition allows generating a wider class of fractal shapes (see fig. 1), but it is still limited to the Euclidean domain. However, linear interpolation in the Euclidean domain corresponds to the motion along the shortest path between two points. On a Riemannian surface $\mathcal{M}$, this translates to the motion along the shortest geodesic curve between the two points. Let $d_{\mathcal{M}} : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$ the geodesic distance function over $\mathcal{M}$; for each point $p_i \in \mathcal{M}$, we define a function $\delta_i : \mathcal{M} \to \mathbb{R}$ that maps $x \mapsto d_{\mathcal{M}}(p_i, x)$. In the domain of surface $\mathcal{M}$, eq. (1) becomes

$$f_i(x) = \exp_x(s_i\,\delta_i(x), -\nabla\delta_i(x))\,, \tag{2}$$

where $\nabla\delta_i(x)$ is the gradient of δ_i at x, and $\exp_x$ is the exponential map at x. At each point $x \in \mathcal{M}$, the function f_i represents a movement towards point p_i by a fraction s_i of the distance from it. This behavior is depicted in fig. 2, which also highlights the analogy between the planar case and the sphere. Given a starting point, we

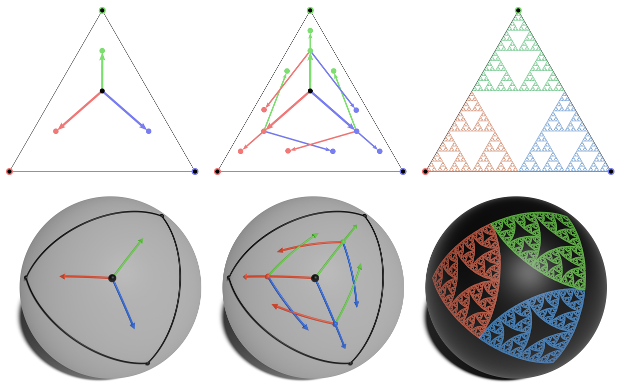

Figure 2: Generating recursion of the Sierpinski triangle on the plane (top) and on a sphere (bottom). Given a starting point, the first generation gets moved halfway through toward the attraction points (left). The second generation is subject to the same attraction, but uses the previous points as starting points (center). The entire process converges to the Sierpinski triangle (right).

perform a motion that brings toward the attraction points by some fraction of the distance, producing the output of the first generation. For each following generation, we iterate the same process, starting from each output of the previous generation. Eventually, the process converges to the fractal pattern associated with the IFS.

On discrete surfaces, the motion described in eq. (2) can be easily implemented as a discrete exponential map [Sharp 2021]. Since we are only interested in moving towards the attraction points p_i, we can precompute the global geodesic distances from these points, and store them as scalar functions over the surface. From these, we can easily derive and store the associated gradient fields at the triangles [De Goes et al. 2016].

The recursive nature of the IFSs results in an exponential complexity: for a system with k attraction points, at the n-th generation we must handle k^n points. To overcome this limitation, we adopt the "chaos game" approach [Barnsley and Vince 2011]. With this method, we initialize a set of q particles to the initial point and fix a maximum number n of generations. At each generation, we select a random attraction point for each particle and perform a single motion toward that attraction point. Given enough particles and generations, the process converges to a result that is visually indistinguishable to the exact solution. Other than reducing the complexity to $O(q\,n)$, this method is embarassingly parallel, enabling straightforward GPU implementations [Lawlor 2012]. As a positive side effect, this formulation makes it easier to encode additional rules and constraints (*e.g.*, no particle can chose the same attraction point two consecutive times) to generate a wider family of fractals [Barnsley et al. 1989], such as the examples in the bottom left and bottom center of fig. 3.

3 Results

With our approach, we are able to replicate several 2D fractals over arbitrary surfaces. In fig. 3 we show different examples of IFS fractals emerging on surfaces from eq. (2), compared with their planar

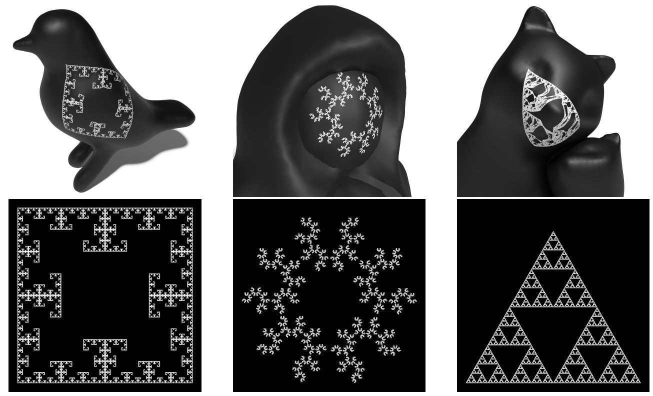

Figure 3: Top row: different IFS fractals computed directly onto surfaces with our method. Bottom row: the corresponding fractals in the plane.

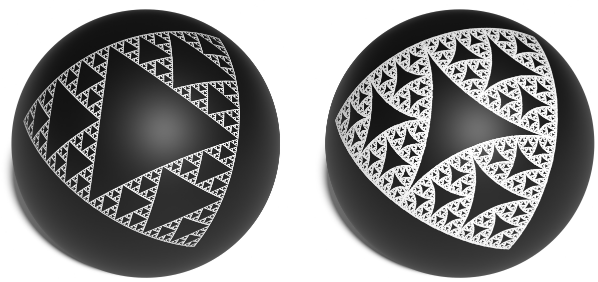

Figure 4: Left: planar Sierpinski triangle drawn over a sphere with geodesic barycentric coordinates [Rustamov 2010]. Right: geodesic Sierpinski triangle generated with our approach.

counterpart. In particular, from the figure we can see that our approach also allows a seamless generalization to a restricted chaos game formulation. The first two fractals are obtained using four and five attractors, respectively, and restricting the choice of the attractors at each iteration by preventing the choice of the same attractor twice in a row: we can appreciate how our approach produces the same patters as the 2D formulation. The third fractal is drawn over a non-convex region, which makes our formulation, in general, ill-defined. This stems directly from our formulation depending on the gradient of the geodesic distance, which is undefined (and thus numerically unstable) at the cut locus. Additionally, eq. (2) is not guaranteed to be contractible outside of convex regions. As a result, when dealing with non-convex regions we can obtain undesirable behaviors, such as the heavy distortion of the pattern depicted in the third frame of fig. 3.

Since our approach produces patterns on surfaces inside a convex geodesic polygon, we considered the geodesic barycentric interpolation [Rustamov 2010] a natural baseline for comparing our results. Indeed, for most IFSs, the k attraction points form a polygon that encloses the produced fractals. This means that we can define a k-sided convex geodesic polygon and apply the geodesic barycentric interpolation to draw the same pattern on the surface.

In fig. 4, we compare a planar Sierpinski triangle drawn on a sphere with geodesic barycentric coordinates against the pattern emerging from the geodesic formulation of the Sierpinski triangle as in eq. (2). We see that in both cases the fractal is enclosed by the geodesic triangle formed by the attraction points. However, as a result of the linear interpolation, the internal triangles appear less curved and more skewed, breaking the recursive structure of the fractal. In contrast, our approach based on geodesic paths produces internal triangles that are more consistent with the underlying curvature, resulting in a better visual preservation of the recursion.

4 Conclusions

We presented a new formulation for iterated function systems (IFSs) based uniquely on the distance from the generating attractor points. We exploited this new formulation to generalize the generation of fractals on arbitrary surfaces through well-established approaches such as the chaos game. Despite being well-defined only inside convex regions, our formulation produces patterns that faithfully reflect their planar counterparts. Compared to standard methods based on decals and geodesic barycentric interpolation, our method produces shapes that better reflect the deformation induced by the underlying geometry, resulting in a more coherent visualization.

References

Michael F Barnsley. 2014. *Fractals everywhere.* Academic press.

Michael F Barnsley, John H Elton, and Douglas P Hardin. 1989. Recurrent iterated function systems. *Constructive approximation* 5, 1 (1989), 3–31.

Michael F Barnsley and Andrew Vince. 2011. The chaos game on a general iterated function system. *Ergodic theory and dynamical systems* 31, 4 (2011), 1073–1079.

Fernando De Goes, Mathieu Desbrun, and Yiying Tong. 2016. Vector field processing on triangle meshes. In *ACM SIGGRAPH 2016 Courses.* 1–49.

David S Ebert, F Kenton Musgrave, Darwyn Peachey, Ken Perlin, and Steve Worley. 2002. *Texturing and modeling: a procedural approach.* Elsevier.

John C Hart. 2001. Perlin noise pixel shaders. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS workshop on Graphics hardware.* 87–94.

Orion Sky Lawlor. 2012. GPU-Accelerated Rendering of Unbounded Nonlinear Iterated Function System Fixed Points. *International Scholarly Research Notices* 2012, 1 (2012), 825782.

Filippo Maggioli, Daniele Baieri, Simone Melzi, and Emanuele Rodolà. 2022a. Newton’s fractals on surfaces via bicomplex algebra. In *ACM SIGGRAPH 2022 Posters.* 1–2.

Filippo Maggioli, Riccardo Marin, Simone Melzi, and Emanuele Rodolà. 2022b. MoMaS: Mold Manifold Simulation for real-time procedural texturing. In *Computer Graphics Forum*, Vol. 41. Wiley Online Library, 519–527.

Claudio Mancinelli, Giacomo Nazzaro, Fabio Pellacini, and Enrico Puppo. 2022. b/surf: Interactive bézier splines on surface meshes. *IEEE Transactions on Visualization and Computer Graphics* 29, 7 (2022), 3419–3435.

Diana Marin, Filippo Maggioli, Simone Melzi, Stefan Ohrhallinger, and Michael Wimmer. 2024. Reconstructing curves from sparse samples on riemannian manifolds. In *Computer Graphics Forum*, Vol. 43. Wiley Online Library, e15136.

Giacomo Nazzaro, Enrico Puppo, and Fabio Pellacini. 2021. geoTangle: interactive design of geodesic tangle patterns on surfaces. *ACM Transactions on Graphics (TOG)* 41, 2 (2021), 1–17.

Marzia Riso, Giacomo Nazzaro, Enrico Puppo, Alec Jacobson, Qingnan Zhou, and Fabio Pellacini. 2022. Boolsurf: Boolean operations on surfaces. *ACM Transactions on Graphics (TOG)* 41, 6 (2022), 1–13.

Raif M Rustamov. 2010. Barycentric coordinates on surfaces. In *Computer Graphics Forum*, Vol. 29. Wiley Online Library, 1507–1516.

Nicholas Sharp. 2021. *Intrinsic Triangulations in Geometry Processing*. Ph. D. Dissertation. Carnegie Mellon University, USA.

Steven Worley. 1996. A cellular texture basis function. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques.* 291–294.