Conformal Mappings of the Hyperbolic Plane to Arbitrary Shapes

Eryk Kopczyński[1] and Dorota Celińska-Kopczyńska[2]

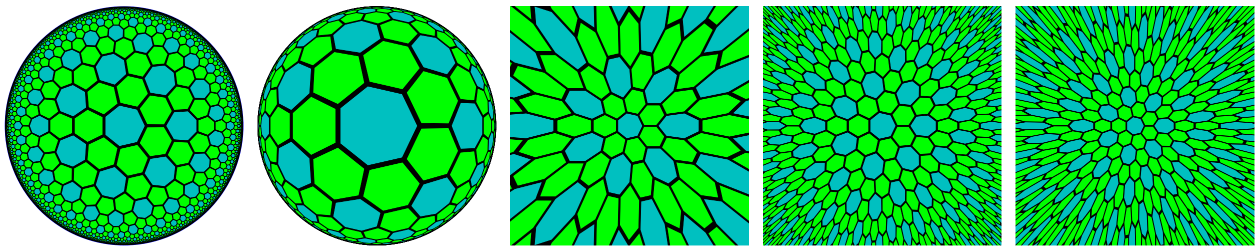
[1]Institute of Informatics, University of Warsaw, Poland; erykk@mimuw.edu.pl
[2]Faculty of Economic Sciences, University of Warsaw, Poland; dcelinska@wne.uw.edu.pl

Abstract

Hyperbolic tessellations have been typically projected into discs or other simple shapes. In this paper, we study the problem of projecting hyperbolic tessellations to arbitrary shapes, given by a source bitmap image. We present an algorithm for finding such conformal mappings and provide discussion of the exemplar results. The previous approaches are vulnerable to numerical precision issues or computationally expensive; our proposition deals with long and narrow shapes efficiently.

Introduction

It is well known that, due to the roughly spherical shape of the surface of Earth, every map will introduce some kind of distortion. Such a map can be conformal (angles between lines are mapped faithfully, e.g., stereographic or Mercator projection), equi-area (areas on the map are proportional to actual areas), transform geodesics to straight lines (e.g., gnomonic projection), equidistant (a specific subset of distances are mapped proportionally – e.g., in the equirectangular projection the distances on equator and meridians are mapped proportionally), but it is impossible in general to satisfy all of these properties at once. This is because the surface of the sphere is non-Euclidean, and the same phenomenon happens when trying to map the hyperbolic plane $\mathbb{H}^2$. Figures 1 (a–e) depict a tessellation of the hyperbolic plane in several projections: Poincaré disk model (conformal), Klein disk model, Gans model, azimuthal equidistant and azimuthal equi-area. All the projections in Figures 1 (a–e) are *azimuthal*, i.e., geodesics passing through the point in the center are mapped to straight lines, and circles centered in the center point are mapped to circles.



(a) **(b)** **(c)** **(d)** **(e)**

Figure 1: *Azimuthal projections of $\mathbb{H}^2$: (a) Poincaré disk model, (b) Klein disk model, (c) Gans model, (d) azimuthal equidistant, (e) azimuthal equi-area.*

We believe the Poincaré disk model is the most aesthetically pleasing of these projections. Because of its conformal nature, the shapes are recognizable (just scaled) in every part of the picture. The closer we get to the boundary, the smaller the scale is, yielding a nice self-similar pattern. The Poincaré disk model is also among the most recognized projections among the general public, mostly because it was featured by M.C. Escher in his *Circle Limit* series.

However, the Poincaré disk model is not the only conformal projection of the hyperbolic plane. The well-known upper half-plane model, as well as the recently popular band model, are also conformal (Figures 2 (a–d)). We may perceive the band model as the hyperbolic analogue of the Mercator projection of the sphere. Take a straight line H in the hyperbolic plane, then map this line to an Euclidean straight line E. The rest of the hyperbolic plane is mapped conformally, yielding a band of a finite width.

It is useful to think of conformal projections as functions from $\mathbb{H}^2$ to a subset of complex numbers $\mathbb{C}$; for example, the Poincaré disk model maps $\mathbb{H}^2$ to $\mathbf{D} = \{z \in \mathbb{C} : |z| < 1\}$, while the upper half-plane model maps $\mathbb{H}^2$ to $\mathbf{P} = \{z \in \mathbb{C} : \Im z > 0\}$, and the band model maps $\mathbb{H}^2$ to $\mathbf{B} = \{z : 0 < \Im z < 1\}$. In this presentation, conformality corresponds to the notion of a biholomorphic complex function, in the following sense: composing a conformal projection $\pi : \mathbb{H}^2 \to S \subseteq \mathbb{C}$ with a biholomorphic function $f : S \to T \subseteq \mathbb{C}$ yields another conformal projection. A function f is *biholomorphic* if both f and its inverse are complex differentiable in every point. For example, the upper half-plane model can be obtained by composing the Poincaré disk model with $f(z) = 1/(z - i) + 1/2$, and the band model can be obtained by composing the upper half-plane model with $f(z) = \log(z)/\pi$, where log is the complex logarithm. The spiral projection from Figure 2 (d) has been obtained from the band model with $f(z) = \exp(zc)$, where exp() is the complex exponential function and the parameter $c \in \mathbb{C}$ controls the shape of the spiral.

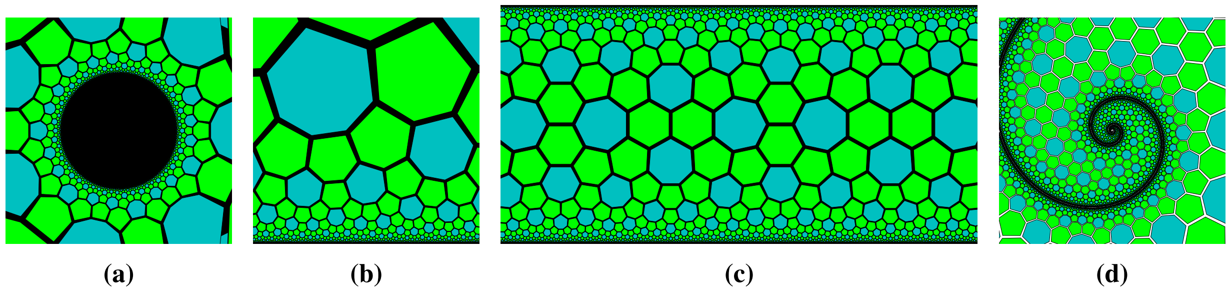


Figure 2: *Conformal projections of* $\mathbb{H}^2$*: (a) inverted Poincaré model, (b) upper half-plane model, (c) band model, (d) spiral.*

Conformal projections of the hyperbolic plane have been studied by Bulatov [2], who was mostly using specific formulas to map $\mathbb{H}^2$ to various shapes. Schwartz-Christoffel mapping gives a formula for mapping $\mathbf{P}$ to an arbitrary polygon, such as a square [3]; however, this formula is computationally difficult [4]. According to the Riemann mapping theorem, there exists a biholomorphic mapping from any non-empty simply connected open subset of $\mathbb{C}$ to $\mathbf{D}$; by composing such a mapping with the Poincaré disk model we can get a projection of the hyperbolic plane of any (simply connected) shape.

In this paper, we study the problem of mapping the hyperbolic plane $\mathbb{H}^2$ to an arbitrary (simply connected) shape. We are the most interested in map the hyperbolic plane to are ones that feature long, narrow paths, such as the Elegant Cat picture (Figure 3 (a)). We want to numerically find a mapping $f : X \to \mathbb{H}^2$, where X is the interior of the given image. Combined with a tessellation of $\mathbb{H}^2$ we get a picture like in Figure 3 (b). The problem of finding the Riemann mappings numerically has been studied with applications in engineering and physics in mind. The Zipper algorithm [8] is one of the most efficient known algorithms; up to our knowledge, this algorithm has not been used to map hyperbolic tessellations. [9] describes a simple iterative algorithm for conformal mapping based on iterative approach, with artistic applications different than ours. Both of these algorithms are not suitable for the kind of shapes we are interested in due to numerical precision issues that arise whenever dealing with big distances in hyperbolic geometry (in particular, the iterative approach [9] converges very slowly for our shapes). Our core contribution is a simple algorithm which avoids the precision issues by being based on the band model.

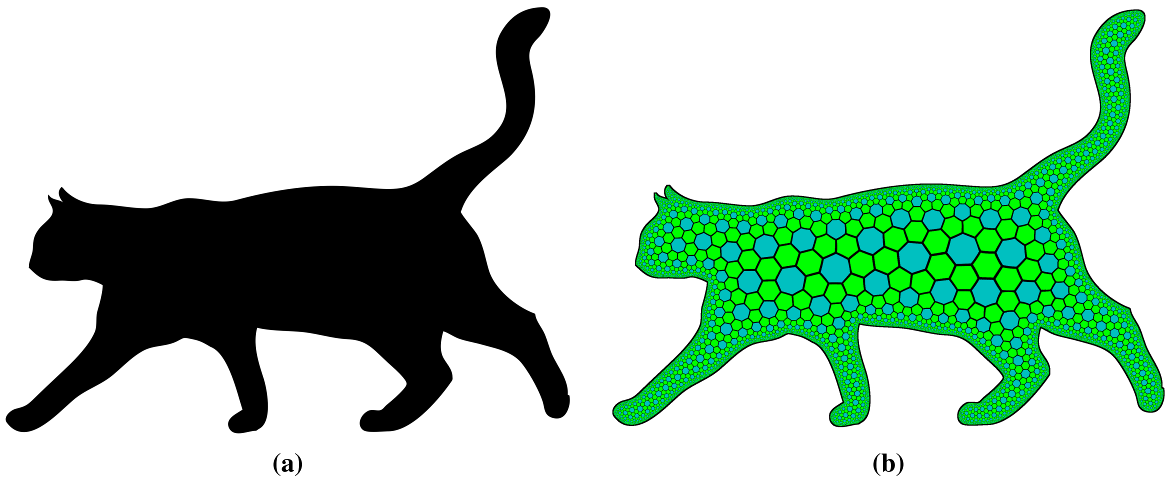


Figure 3: *Elegant Cat Silhouette [5]: (a) source, (b) the result image.*

Our Algorithm

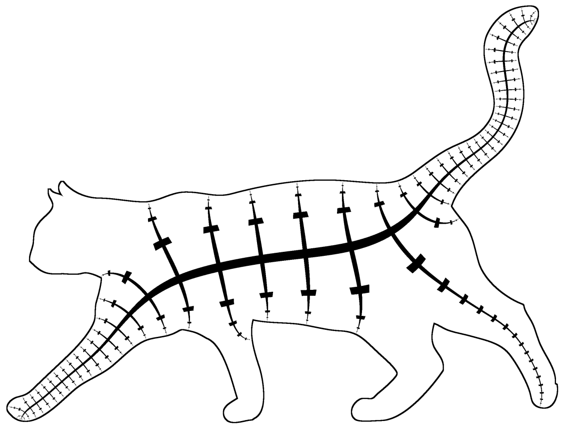
We treat our shape as a discrete set of pixels. For example, the original Elegant Cat Silhouette [5] is a bitmap with resolution 2328x1758, with 1,430,020 black, internal, pixels. We will find a mapping $f : P \to \mathbf{B}$, where P is the set of internal pixels in the image, and $\mathbf{B}$ is the hyperbolic plane in the band model. Let us denote by B the boundary of P, i.e., pixels that are not in P but are adjacent to P. The set B will be mapped to the boundary of the band, consisting of $-\infty$, ∞, $\mathbf{B}_0 = \{z : \Im z = 0\}$ and $\mathbf{B}_1 = \{z : \Im z = 1\}$. We pick two points $p_{-\infty}$ and p_∞ on the boundary of P; the point $p_{-\infty}$ will be mapped to $-\infty$, and p_∞ will be mapped to ∞. These two points split the boundary of P into two parts, the bottom part B_0 and the top part B_1. The bottom part will be mapped to $\mathbf{B}_0$ and the top part will be mapped to $\mathbf{B}_1$.

Treated as functions from $\mathbb{R}^2$, holomorphic functions are harmonic: $\Delta f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = 0$. Since we are looking for a mapping from a discrete set of pixels rather than a continuous subset of $\mathbb{R}^2$, we replace the Laplace operator Δ with its discrete variant: $\Delta f = \sum_{q \in N(p)} f(q) - f(p)$, where $N(p)$ is the set of 4 pixels adjacent to p; in other words, for every pixel p, $f(p)$ should be the average of f for the four neighbors of p. Since we are mapping to $\mathbf{B}$, we get the following boundary condition: $\Im f(p) = 0$ in the bottom part, $\Im f(p) = 1$ in the top part, $\Re f(p_{-\infty}) = -M$, $\Re f(p_\infty) = M$; for p in the top and bottom part, $\Re f(p)$ should be close to the value of the adjacent internal pixels. We use M instead of ∞ because ∞ gets "rounded down" to a finite number due to our discretization. Since the real and imaginary are completely independent, we can solve for them independently (note that this would not be the case if we used another target projection instead of $\mathbf{B}$). This way, we get a system of n equations of form $\Im f(p) = (\Im f(q_1) + \Im f(q_2) + \Im f(q_3) + \Im f(q_4))/4$; for $\Re$, we get a similar system of equations, but we skip neighbors in $B_0 \cup B_1$.

We then solve this system of equations using Gaussian elimination. We do not know the value of M, but we can assume $M = 1$ and, after the computation, rescale the real part so that the combination of two harmonic functions $\Re f$ and $\Im f$ is conformal.

Numerical Precision Issues

Figure 4 (a) explains why our approach, based on the band model, lets us avoid numerical precision issues inherent to other algorithms, typically based on Poincaré disk and half-plane models. We have picked points $p_{-\infty}$ and p_∞ to be the tip of the front leg and the tip of tail. The long internal line is the set of points mapped to points $z \in \mathbf{B}$ such that $\Im z = 1/2$; we will call this line the "equator". The short internal lines are hyperbolic straight lines orthogonal to the equator (the "meridians"); crosses are drawn each 1 unit. About 50 crosses are clearly visible on the equator, and the hyperbolic distance between points that are actually on the boundary would be infinite. Hence, the visible endpoints are about 25 units away from the center. The Poincaré disk model maps the point d units to the right from the center to $z = \sinh(d)/(1 + \cosh(z))$, which for $d = 25$ equals roughly $1 - 2.8 \cdot 10^{-11}$. This is dangerously close to the precision of the `double` floating-point format; and in some shapes, the distances will be even greater. As a result, such precision error could significantly distort the fine details of the tessellation we are mapping to our source image. This issue is not restricted to the Poincaré disk model: say, any representation of a point in $\mathbb{H}^2$ as a pair of double-precision floating-point numbers (128 bits) will be able to map only 2^{128} distinct points. Since the hyperbolic circle of radius r has area $2\pi(\cosh(r) - 1) \approx \pi e^r$, any such representation will map two points in hyperbolic distance greater than 1 into a single point. Many algorithms trying to map the hyperbolic plane to long, narrow shapes fail to achieve the necessary precision, and thus the fine details of the tessellation we are mapping to our source image, or maybe even whole parts of the picture are destroyed. Examples of such distortions are shown in [4].


(a)

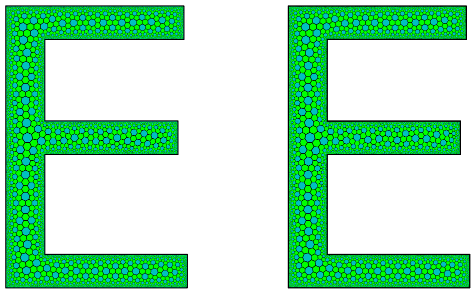

(b)

Figure 4: *Intuition for numerical precision issues: (a) hyperbolic straight lines in Elegant Cat Silhouette, (b) letter E: the result of shifting.*

One way of resolving these issues is used in [6]. A regular tessellation of $\mathbb{H}^2$ can be generated without any precision issues using finite automata [7]; we then get a precise representation of $\mathbb{H}^2$ by representing each point with the closest vertex of our regular tessellation and coordinates relative to that closest vertex. This approach is quite complicated though, and it does not yield any way of representing the ideal points (i.e., points on the boundary of our projection).

Instead, our approach is based on the band model, which represents all points close to the equator $\Im z = 1/2$ with high precision. By choosing the two ends of a long, narrow path ($p_{-\infty}$ and p_∞) to be mapped to $-\infty$ and ∞, we guarantee that all the points on this path will be represented precisely. More complicated, branching shapes will require another approach, described later.

To verify the accuracy of our numerical approximations, we have applied our method to a disk of diameter 1000 and compared the result with the Poincaré disk model. The mapping produced by our algorithm matches the Poincaré disk model within 2 pixels.

Another consequence of hyperbolic geometry is the instability of the mapping found. In Figure 4 (a), the hyperbolic distance from the visible points on the rear leg to the equator is about 16 units. Suppose we translate our hyperbolic tessellation by 0.0001 units. This would have no visible effect on the line AB, but the tessellation on the other legs would look completely different. This is because, when we consider two points X and Y on the equator in small distance ε and points X' and Y' in distance d from AB such that the orthogonal projections of X' and Y' on line AB are X and Y, respectively, the distance between X' and Y' is on the order of $\varepsilon \sinh(d)$ (as long as $\varepsilon \sinh(d) = O(1)$), which grows exponentially with d; therefore, a minor movement on the equator results in a complete change of the mapping on the other legs. For a clearer vision, we display this effect in Figure 4 (b) on an image of letter E. We notice that the heptagons in the middle bar are placed differently between the left and the right picture. Sadly, this instability makes conformal projections incompatible with some forms of animation (e.g., making the Elegant Cat run).

Optimizations

One issue with our algorithm is that the Gaussian elimination is potentially computationally expensive. Let us say that pixel p depends on q if $f(q)$ appears in the current equation for $f(p)$ with a non-zero coefficient. When we eliminate $f(p)$, all pixels depending on p become dependent on all pixels q that p depended on, except themselves. Consider solving a system of equations produced by a rectangle of size $X \times Y$ pixels. If we eliminate pixels in the reading order (from top to bottom, and from left to right in each line), each pixel p we eliminate (except the first and last line) will mutually depend on $Y - 1$ other pixels adjacent to the already eliminated region; eliminating p will take time $O(Y^2)$ (due to updating the coefficients of other pixels), and the whole algorithm will take time $O(XY^3)$. The memory complexity of this approach is $O(XY^2)$.

To combat this, we apply a different order of elimination: first we eliminate each pixel with both coordinates odd, so that the remaining pixels form 2×2 squares; then, we eliminate pixels with x coordinate odd, making the remaining pixels form 4×2 squares; then we eliminate pixels such that $x/2$ is odd and $y \bmod 4 \neq 0$, thus making our 4×2 join into 4×4 squares; and so on. In the kth iteration (which constructs $2^k \times 2^k$ squares out of $2^{k-1} \times 2^{k-1}$) we remove $O(n/2^k)$ pixels, and each of them depends on $O(2^k)$ pixels, thus k-th iteration will take time $O(2^k n)$. The number of iterations we require is $\log \min(X, Y)$, hence this method improves the time complexity of our algorithm to $O(n \min(X, Y)) = O(XY^2)$. This is the worst case complexity; for the source images interesting for us, that have long but narrow regions, the number of dependent pixels in k-th operation will be bounded by the width of the region. Further performance improvement is achieved by rotating the image first by 45 degrees – this allows our algorithm to use the dependence more efficiently.

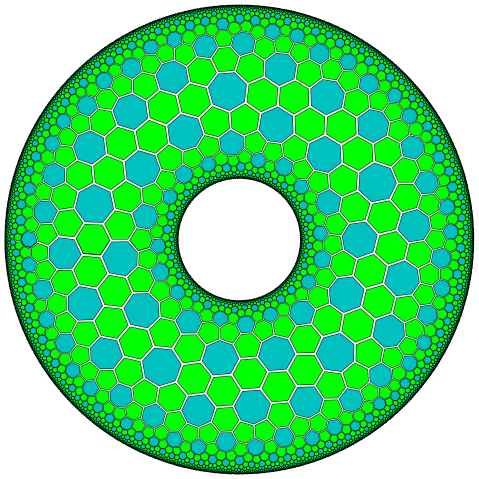
While our algorithm deals with long and narrow regions efficiently, it is quite slow for pixels p that are far away from the boundary. Another potential optimization is to notice that conformal mappings will be roughly linear in the neighborhoods of such points p, so we should not lose much precision by making our grid less precise in such regions. However, we have not managed to successfully implement an optimization of this kind so far. Our current algorithm works about 5 minutes for a disk of diameter 1000, on an Intel i5 CPU 2.67 GHz (using a single thread).

What to Do about the Exterior?

After mapping the internal points of the source image, a new problem appears: what to do with the set of points P' that is the complement of P, i.e., the white points in the source image? We would like to have another hyperbolic tessellation mapped conformally to P'. One possible approach is to apply inversion to P' (considered as an infinite set), thus getting a bounded, simple connected region, and then apply the same algorithm to map this inversion. The result would look similar to the inverted Poincaré model in Figure 2 (a).

We have decided to take another approach. Let R be a rectangle containing P, and $P' = R - P$. The set P' is homeomorphic to an annulus. As found by Bulatov [2], periodic tessellations can be mapped to annuli. Consider a periodic tessellation T of $\mathbf{B}$, with period t (in the band coordinates). The complex function

$f(z) = \exp(2\pi iz/t)$ maps $\mathbf{B}$ to the ring $\mathbf{R}_t = \{z \in \mathbb{C} : |z| \in (\exp(-2\pi/t), 1)\}$. We color each point z in $\mathbf{R}_t$ with the same color as $f^{-1}(z)$ in T; while the function f is not injective, it has period t, so that color is defined unambiguously. Figure 5 (a) depicts our example tessellation on an annulus.


(a)

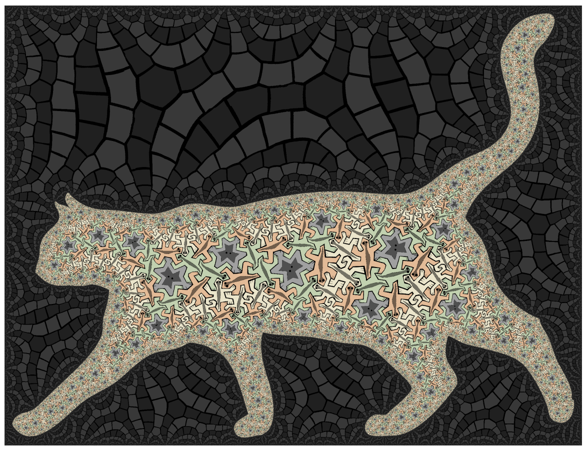

(b)

Figure 5: *(a) Conformal mapping to an annulus, (b) Elegant Cat Silhouette, tessellated both inside and outside.*

Our algorithm can be easily modified to find similar periodic conformal mappings for P'. We will map P' to $\mathbf{B}$ under the assumption that the value of $\Re f$ increases by $M \in \mathbb{R}$ when we go around P. We map the internal boundary B_1 to $\mathbf{B}_1$, and the external boundary B_0 to $\mathbf{B}_0$. We determine $\Im f(p)$ in exactly the same way as in the section describing the algorithm. For $\Re f(p)$, we cut our P' with a straight line segment L connecting B_0 and B_1, and for the equations $\Re f(p)$ that depend on points q such that p and q are on the opposite sides of L, we add or subtract M. We do not know the value of M in advance, but we can set $M = 1$ and then rescale when we combine two harmonic functions into a conformal mapping.

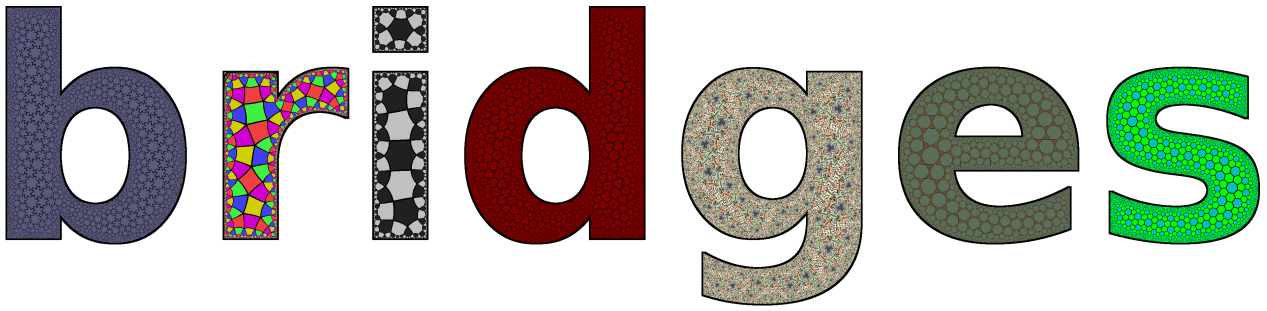


Figure 6: *Bridges pattern mapped with our algorithm.*

This method will work not only for mapping the exterior, but also for all shapes homeomorphic to annuli (Figure 6). However, one issue here is that the obtained M does not have to be a multiple of the period of the tessellation we have chosen for our exterior (and in fact, the period of any hyperbolic tessellation). In the case of the exterior P', we have control over the size of the margin we add to the source image – the larger the margin, the smaller M will be, so we can pick the size of the margin to make M a multiple of the period, or close enough that the error will not be noticeable to observers. Figure 5 (b) depicts the Elegant Cat Silhouette with both the inside and outside mapped to hyperbolic tessellations from HyperRogue [6]. For annuli other than the exterior, we have no control of the margin, so stretching the tesselation slightly may be necessary. We see no way to make our method work with shapes with more than one hole, such as the capital B.

Branching Shapes

Our algorithm is successful at conformally mapping the hyperbolic plane to long, narrow shapes that do not branch. Most hyperbolic art is based on periodic tessellations in the Poincaré disk model; our algorithm lets us make such art into arbitrary shapes. Our algorithm is also perfect for mapping non-periodic hyperbolic image rendered in the band model, as in Figure 7 (a) (see [6] for an explanation of the source art). However, the described algorithm is less successful at mapping branching shapes, such as the letter E, or Triskele [1] depicted in Figure 7 (b). We can pick the points p_∞ and $p_{-\infty}$ to be the ends of two branches, but then the third branch will not be computed correctly due to the precision issues. (This phenomenon does not occur in the Elegant Cat Silhouette because the other legs are short enough.)

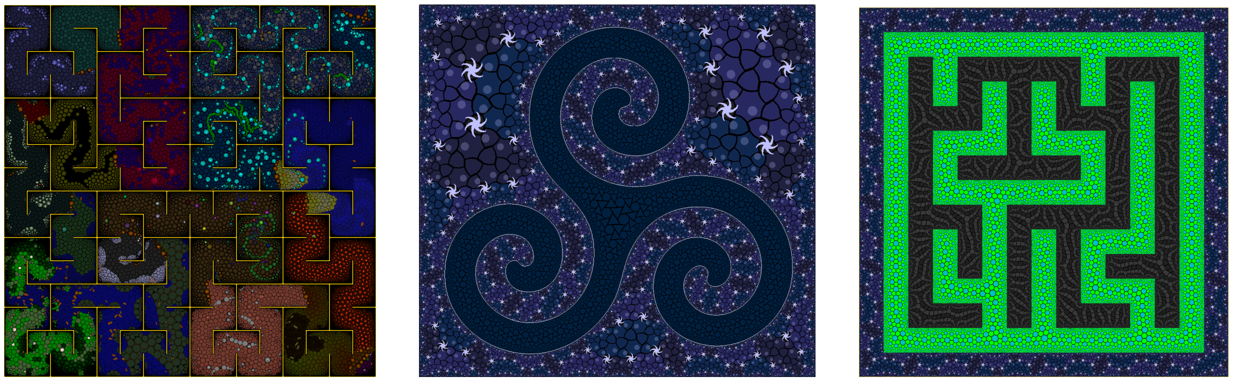

(a) **(b)** **(c)**

Figure 7: *Examples of mapped branching shapes: (a) Hilbert curve, (b) Triskele, (c) maze.*

We took the following approach to map the Triskele (Figure 7 (b)): let A, B, C be the ends of the three branches; first, we run our algorithm taking $p_\infty = A$ and $p_{-\infty} = B$ obtaining mapping f_{AB}, then we take $p_\infty = A$ and $p_\infty = C$, obtaining mapping f_{AC}. The mapping f_{AC} will be precise on the line from A to C (but not close to B), while the mapping f_{AB} will be precise on the line from A to B (but not close to C). If we tried to tessellate the AB line using f_{AB} and the remaining part using f_{BC}, the two tessellations would not combine seamlessly, because they have been computed independently. However, we fix this by choosing a point D close to the seam and finding a hyperbolic isometry which makes f_{AB} and f_{AC} agree in the neighborhood of D. After composing f_{AC} with such a hyperbolic isometry, the seam is no longer visible. The same approach also can be used to map source images with more branching points, like the exterior of the Triskele, trees or mazes (Figures 7 (b,c)).

Our Implementation

Our implementation can be found at `http://github.com/zenorogue/newconformist/`. The source image is given as a bitmap (png), but newconformist can generate common shapes such as rectangles, circles, or Hilbert curves. The tessellation is also provided in .png format, in either Poincaré disk or band model. (Of course Poincaré disk tessellation images do not have enough resolution to be remapped to long, narrow stripes – but if they are periodic and the periods are given, we can find a representant of each orbit that is close to the center of the Poincaré disk.) Newconformist automatically finds suitable points $p_{\pm\infty}$ on the boundary and detects branching shapes; then it generates suitable extra endpoints and central points for them. While it takes a substantial time to generate conformal mappings, they can be saved to intermediate files that can be afterward loaded – so, if we want to experiment with various tessellations, we can compute the mapping only once.

Summary

In this paper, we have proposed an algorithm to map the hyperbolic plane $\mathbb{H}^2$ to arbirtary (simply connected) shapes. Using the example of the Elegant Cat Silhouette, we presented the basic algorithm and the possibilities for the optimizations. Although there were other approaches taken to provide solutions to similar problems, the advance of our proposition lies in avoiding numerical precision issues, to which other algorithms were vulnerable. Our approach is successful in conformally mapping the hyperbolic plane to long, narrow shapes. The presented algorithm can be utilized in development of the visualization tools for hyperbolic geometry, for the purposes of both mathematicians (e.g., gaining better intuitive understanding of the concept), as well as the artists (e.g., as a means for creating artworks).

References

[1] AnonMoos. "Spiral triskele, found in celtic artwork, used by celtic reconstructionists and occassionally as a christian trinity symbol", 2007. `https://en.wikipedia.org/wiki/Triskelion#/media/File:Triskele-Symbol1.svg` (Jan 22, 2018), public domain.

[2] V. Bulatov. "Conformal models of the hyperbolic geometry", 2010. `http://bulatov.org/math/1001/index.html` (as of Jan 20, 2017).

[3] C. Fong. "The conformal hyperbolic square and its ilk." *Bridges Conference Proceedings*, Jyväskylä, Finland, Aug. 9–13, 2016, pp. 179–186.

[4] C. Fong and D. Dunham. "A poor man's hyperbolic square mapping." *Bridges Conference Proceedings*, Stockholm, Sweden, July 25–29, 2018, pp. 59–66. `http://archive.bridgesmathart.org/2018/bridges2018-59.pdf`.

[5] GDJ. "Elegant cat silhouette", 2016. `https://www.iconspng.com/image/78076/elegant-cat-silhouette` (Jan 22, 2018), free for personal and commercial use.

[6] E. Kopczyński, D. Celińska, and M. Čtrnáct. "HyperRogue: Playing with hyperbolic geometry." *Bridges Conference Proceedings*, Waterloo, Canada, July 27–31, 2017, pp. 9–16. `http://archive.bridgesmathart.org/2017/bridges2017-9.pdf`.

[7] M. Margenstern. "Small universal cellular automata in hyperbolic spaces: A collection of jewels", vol. 4. Springer Science & Business Media, 2013.

[8] D. Marshall and S. Rohde. "Convergence of a variant of the zipper algorithm for conformal mapping." *SIAM Journal on Numerical Analysis*, vol. 45, no 6, 2007, pp. 2577–2609.

[9] D. Swart. "Warping pictures nicely." *Bridges Conference Proceedings*, Coimbra, Portugal, July 27–31, 2011, pp. 303–310. `http://archive.bridgesmathart.org/2011/bridges2011-303.pdf`.