Controllable Conformal Maps for Shape Deformation and Interpolation

Ofir Weber Craig Gotsman
Technion – Israel Institute of Technology
weber@cs.technion.ac.il gotsman@cs.technion.ac.il

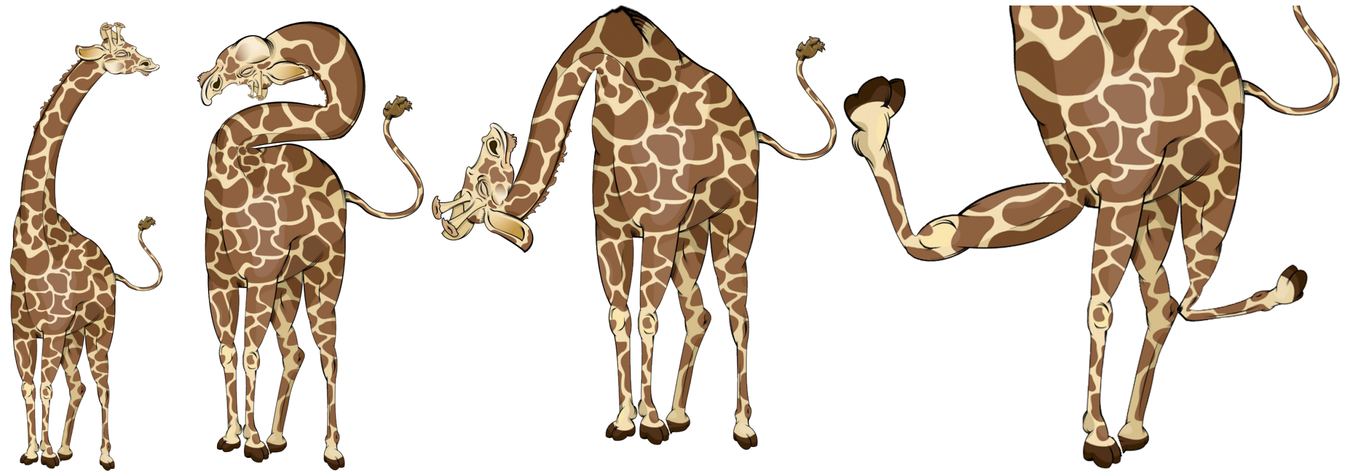

Figure 1: *Conformal deformation of a giraffe with sharp bends at neck and legs. Original model (left) and three deformed versions.*

Abstract

Conformal maps are considered very desirable for planar deformation applications, since they allow only local rotations and scale, avoiding shear and other visually disturbing distortions of local detail. Conformal maps are also orientation-preserving C^{∞} diffeomorphisms, meaning they are extremely smooth and prevent unacceptable "foldovers" in the plane. Unfortunately, these maps are also notoriously difficult to control, so working with them in an interactive animation scenario to achieve specific effects is a significant challenge, sometimes even impossible.

We describe a novel 2D shape deformation system which generates conformal maps, yet provides the user a large degree of control over the result. For example, it allows discontinuities at user-specified boundary points, so true "bends" can be introduced into the deformation. It also allows the prescription of angular constraints at corners of the target image. Combining these provides for a very effective user experience. At the heart of our method is a very natural differential shape representation for conformal maps, using so-called "conformal factors" and "angular factors", which allow more intuitive control compared to representation in the usual spatial domain. Beyond deforming a given shape into a new one at each key frame, our method also provides the ability to interpolate between shapes in a very natural way, such that also the intermediate deformations are conformal.

Our method is extremely efficient: it requires only the solution of a small dense linear system at preprocess time and a matrix-vector multiplication during runtime (which can be implemented on a modern GPU), thus the deformations, even on extremely large images, may be performed in real-time.

ACM Reference Format
Weber, O., Gotsman, C. 2010. Controllable Conformal Maps for Shape Deformation and Interpolation. *ACM Trans. Graph. 29*, 4, Article 78 (July 2010), 11 pages. DOI = 10.1145/1778765.1778815 http://doi.acm.org/10.1145/1778765.1778815.

1 Introduction

Planar shape deformation and animation are fundamental applications in computer graphics. Despite the voluminous literature on the subject, a perfect solution still does not exist. The main challenge is to be able to create high quality deformations of a given 2D shape with as little user input as possible without losing control over the result. In other words, an ideal deformation system should allow the user intervention when it is required but infer all the missing data automatically. Once the user fixes a small set of constraints, the system should find the *best* deformed shape that satisfies these constraints. What is considered best depends, of course, on the application. Research in recent years has focused on finding deformations that best preserve the fine details of the source shape. This means that the deformation should be smooth, avoid unnecessary variations, and locally should resemble only rotation and possibly uniform scale. Shear and non-uniform scale should be avoided, and "foldovers" of the image should not be allowed under any circumstances. These desirable properties are precisely those of conformal maps - injective harmonic maps whose two components satisfy the Cauchy-Riemann equations (which mean that their gradients are perpendicular to each other and have the same magnitude). They preserve the angles between intersecting curves and also orientation, having positive Jacobian (determinant) throughout the domain, thus are ideal for shape deformation.

Conformal maps are central to complex function theory [Ahlfors 1979] and have been the subject of intense study for centuries. Most well-behaved complex functions are holomorphic, and with the additional requirement that their derivative does not vanish, they become conformal. At first glance, this would seem to imply that there are plenty of conformal maps, so it should not be too difficult to generate one satisfying some reasonable user-supplied constraints. In fact, the celebrated Riemann mapping theorem [Ahlfors 1979, Ch. 6] guarantees that there exists a (unique up to a few degrees of freedom) conformal map between any two simply-connected regions of the plane. However, the reality of working with conformal maps is quite different. It is notoriously difficult to generate the Riemann maps, and when the boundaries of the two regions are significantly different, the conformal map between the two can be surprisingly complex, with enormous differences in scale being quite common. It is even more difficult to generate conformal maps when more sophisticated constraints are present. For example, it is impossible to impose a correspondence between more than three pairs of points along the two respective boundaries. Thus controlling conformal maps in a natural way to achieve desired results is quite difficult. This is the main problem we address in this paper.

A common approach to generating deformations in practice is to search for the solution within some meaningful subspace of the deformation space. Finding the most suitable subspace is the main challenge. A very popular approach is to use a fixed set of basis functions (depending only on the source shape), also known as *barycentric coordinates*, and then the deformation is expressed as a linear combination of these basis functions. Possible choices are mean-value coordinates [Floater 2003] or harmonic coordinates [Joshi *et al.* 2007]. Recently, Weber *et al.* [2009] suggested using holomorphic basis functions since they may lead to conformal maps. Deformation methods based on barycentric coordinates usually lead to efficient algorithms since the basis functions can be computed in a preprocess step. This is the approach we use in this paper.

The input to our algorithm is a polygonal boundary curve enclosing a simply connected planar region and a small set of *corner* points along the boundary. During the animation session, the user controls the target shape by specifying the orientation of the tangent vector at any boundary point. At a corner point, two distinct tangents should be specified, giving the user the freedom to change the corner angle as well as its orientation. This gives the user a large amount of control over the behavior of the deformation. Moreover, the resulting deformation is always a conformal map which satisfies the user constraints exactly.

Despite it not being possible, in general, to generate a conformal map that maps one planar region to another with an exact prescription of the boundary behavior, as we show in the next sections, there always exists a conformal map that follows an exact angle prescription along the boundary. We provide the theory and the exact recipe to efficiently compute this conformal map using a new type of barycentric coordinates, which we call Hilbert coordinates.

2 Previous Work

In the 2D domain, the subject of this paper, the deformation problem boils down to bounding some region of the plane (usually a portion of an image) by a polygonal *source* "cage" P, and, using a set of controls, deform this to another *target* planar polygon Q. In the simplest scenario, we would like to generate some well-behaved mapping between the interior of P and the interior of Q. The easiest way to achieve this is through barycentric coordinates, which express the coordinates of any x in the interior of P as a linear combination of the coordinates of the vertices of P. The image of x is then defined to be the same linear combination of the vertices of Q. Over the years, many recipes for barycentric coordinates have been proposed, and we mention here just the most well-known: the three-point family [Floater *et al.* 2006] (which includes the cotangent, mean-value and Wachspress coordinates), the harmonic coordinates [Joshi *et al.* 2007] and the maximum entropy coordinates [Hormann and Sukumar, 2008]. Joshi *et al.* define harmonic coordinates using a discrete triangulation of the interior of P, and generate a discrete harmonic map between the interior of P and Q. All these mappings distort P as much as is needed in order to fit Q exactly, sometimes yielding quite visually-unsatisfying results. If the exact boundary mapping between P and Q is relaxed, the mapping being controlled only by a small set of "positional constraints", it is possible to optimize the mapping to contain less shear effects and as much local similarities or rigid transformations (which are considered less distorting) as possible. In the discrete world, these are sometimes called the "As-Similar-As-Possible" (ASAP) [Levy *et al.* 2002; Schaefer *et al.* 2004; Igarashi *et al.* 2005; Karni *et al.* 2009] or "As-Rigid-As-Possible" (ARAP) [Karni *et al.* 2009] mappings.

Realizing that conformal mappings are very desirable in the deformation context, more recent work has restricted the mappings generated to be continuous holomorphic mappings. Thus P will typically not be mapped exactly to Q, rather close to it. These mappings may also be generated using barycentric coordinate functions, such as the Green coordinates [Lipman *et al.* 2008], and the equivalent Cauchy coordinates [Weber *et al.* 2009], which may be controlled directly through the target polygon Q, or through a more intuitive "point-to-point" user interface. Unfortunately, while better than the more distorted traditional deformations, they are still quite difficult to control, and foldovers are not eliminated.

3 Contributions

Our main contribution is a novel 2D shape deformation system which generates controllable conformal maps. We introduce the notion of singular points to 2D shape deformation which leads to the ability to generate more realistic deformations compared to existing methods. To the best of our knowledge, this is also the first deformation method that *guarantees* that the Jacobian of the map will not vanish, thus preventing foldovers completely. This approach also leads to a simple way to interpolate shapes in a natural way such that the intermediate deformations are also conformal.

From a technical point of view, we derive a new type of complex barycentric coordinates which are a generalization of the Cauchy coordinates [Weber et al. 2009], and show an interesting connection between Cauchy coordinates and harmonic coordinates which, in turn, leads to a an elegant computational method for harmonic coordinates. We describe the Hilbert barycentric coordinates which leads to an efficient computation of the so-called Hilbert transform. This, in turn, allows us to construct conformal maps via their derivatives. The efficiency of our deformation method is comparable to methods based on other barycentric coordinates. This permits a very fast deformation algorithm which can be easily implemented on a GPU.

4 Conformal Shape Deformation

Real-valued barycentric coordinates are affine-invariant. Until recently this ability was considered an advantage. This was challenged by Lipman *et al.* [2008] who demonstrated that when barycentric coordinates are used for deformation, affine transformations introduce shear which in turn destroys the fine details of the shape. To avoid this, Lipman *et al.* [2008] suggested blending the

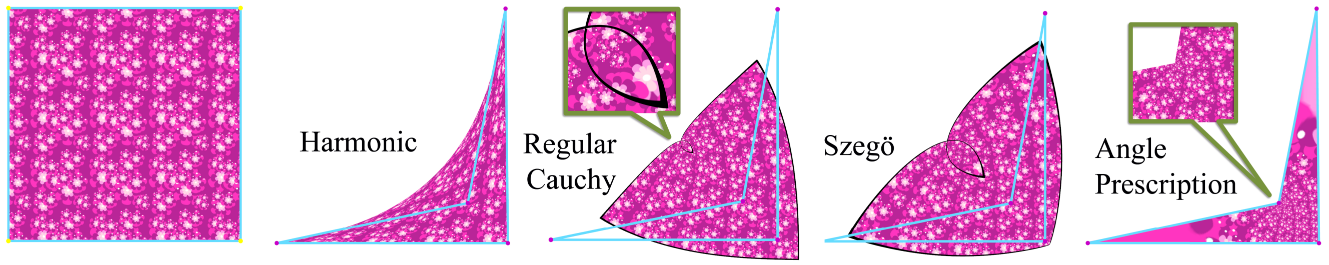

Figure 2: *Deforming a square. (left to right) Original image enclosed by a cage of 4 vertices and resulting deformations controlled by identical target cages, generated by a number of methods: harmonic coordinates (note how the image "spills" outside the target cage), regular Cauchy coordinates (note how the winding number of the boundary curve has changed from 2π to 4π and the derivative vanishes at a point inside the loop), Szegö coordinates (which tries to map closer to the cage), our deformation using exact angle prescription at the corners results in a bijective conformal map without any foldovers.*

cage normals using another set of barycentric coordinates such that the linear combination of vertices augmented with linear combination of normals is no longer affine-invariant. More recently, Weber *et al.* [2009] extended this idea by allowing the barycentric coordinates to be complex-valued functions. Since holomorphic functions form a linear subspace, complex holomorphic barycentric coordinates leads to holomorphic deformations. However, a holomorphic deformation doesn't always mean the map is conformal since the derivative of the function may vanish inside the domain. This leads to local foldovers which may even cause the winding number of the boundary curve to change. Such visual artifacts (see Fig. 2) are undesirable and in this work we provide a concise way to prevent them.

Complex barycentric coordinates (when they are holomorphic) have the advantage of being detail preserving, however, this comes with a price. It is impossible to achieve the Lagrange property i.e., to interpolate the target cage, since it is not possible, in general, to prescribe the boundary mapping of a conformal map. This does not contradict Riemann's conformal mapping theorem since that theorem only guarantees that the source domain will be mapped onto the target domain, with no possibility to influence the mapping between the two boundaries. At first glance, the fact that interpolation cannot be achieved using complex barycentric coordinates is disappointing. However, careful examination shows that actually *none* of the existing real-valued barycentric coordinates has the "containment" property as really expected by a user of an interaction deformation tool, namely that $f(\Omega)$ does not "spill out" of $f(\partial\Omega)$.When the target cage is convex (even if the source cage is not), the Rado-Kneser-Choquet theorem [Duren 2004, Ch. 3] states that the corresponding harmonic map will be a bijection. However, when the target cage is not convex, even harmonic maps cannot guarantee that the Jacobian of the mapping will have positive determinant. This means that even though the target shape interpolates the target cage, some portions of the source region may "spill" outside the target region. Thus, the outer boundary of the target region does not interpolate the target cage (see Fig. 2). In fact, we do not know of barycentric coordinates (real or complex) that *guarantee* a bijective mapping (not necessarily harmonic or conformal) between arbitrary source and target cages. Whether it is possible to create such coordinates is an open question.

In light of this unfortunate situation, we now show how to relax the boundary interpolation requirement, in order to be able to construct conformal deformations which are more appealing than deformations that can be achieved using current complex barycentric coordinates. Instead of prescribing the exact boundary mapping at each point along the edges of the cage we will prescribe only the angular change at each point along the boundary. As we will see next, this can be achieved exactly.

The Jacobian of deformations achieved by real-valued barycentric coordinates can easily vanish or even be negative, resulting in local foldovers. Using complex holomorphic barycentric coordinates guarantees that the Jacobian of the mapping (the absolute value of the first complex derivative) is non-negative (since it is a similarity transformation). Our method is superior to that in the sense that it guarantees that the Jacobian is strictly *positive* inside the domain.

The key to success is the use of a representation of conformal maps that always exists and is unique. The representation should be minimal in the sense that it should not contain any redundant information yet still contain all the information required to uniquely reconstruct the deformed shape. The representation we use forms a linear subspace, namely that a linear combination of two valid shape representations is also a valid shape representation. By doing so, we can reduce the optimization problem (of finding a particular conformal map) to finding a particular shape representation within the linear subspace. This leads to an efficient and simple deformation algorithm which has some provable shape-preserving properties.

4.1 Shape Representation

Given a complex mapping f, consider the following function:

$$\mathrm{Log}(f') = \mathrm{Log}(|f'|) + i\,\mathrm{Arg}(f')$$

This log derivative of f encodes two local geometric properties of f: 1) The *conformal factor*, $\mathrm{Log}(|f'(z)|)$, describes the local scale change induced by f, and 2) the *angular factor*, $\mathrm{Arg}(f'(z))$, describes the local orientation change induced by f at z. As we will soon see, given one of these two functions (conformal factor or angular factor), satisfying some mild conditions on the boundary of a domain, it is possible to uniquely recover the conformal map having that boundary behavior up to some global transformation. It is not possible to prescribe both functions.

Thus our representation of choice for a conformal map f of a given simply connected 2D domain Ω will be its *angular factor* on $\partial\Omega$ - a piecewise-smooth real-valued scalar function defined on $\partial\Omega$, which we denote by $\theta(t)$ when parameterized by arc length. More specifically, given a conformal map $f(z)$ of a source region Ω, define:

$$\theta(z) = \mathrm{Im}\left(\mathrm{Log}\left(f'\left(z\right)\right)\right) = \mathrm{Arg}(f'(z))$$

With slight abuse of notation, the conformal *shape representation* of f is now defined as the parameterized version of θ on $\partial\Omega$:

$$\theta\left(t\right) = \lim_{z \to w(t)} \theta\left(z\right), \quad z \in \Omega,\ w(t) \in \partial\Omega \qquad (1)$$

where $w(t)$ is the point on $\partial\Omega$ corresponding to parameter t. $\theta(t)$ has the geometric interpretation as the local change in tangent direction of f on $\partial\Omega$.

The derivative of a holomorphic function f is also holomorphic. Since f is conformal its derivative does not vanish, hence the multivalued $\mathrm{Log}(f'(z))$ function is also holomorphic. Furthermore, since the real and imaginary parts of any holomorphic functions are harmonic functions, it follows that $\theta(z)$ is harmonic on Ω. Hence, the limit (1) always exists except at some finite number of points.

4.2 Shape Reconstruction

A harmonic function $v(x,y)$ is said to be the (harmonic) conjugate of the harmonic function $u(x,y)$ defined on some open domain $\Omega \subset \mathbb{R}^2$ if and only if u and v satisfy the Cauchy-Riemann equations in Ω. Alternatively, v is the harmonic conjugate of u if and only if $f(z)=f(x+iy)=u(x,y)+iv(x,y)$ is holomorphic on Ω. Any harmonic function admits a harmonic conjugate whenever its domain is simply connected, and it is unique up to an additive constant.

Given the angular factor $\theta(t)$ defined on the boundary of a simply connected domain Ω, it is always possible to reconstruct a conformal map f which has angular factor θ on $\partial\Omega$, and is unique up to translation and scale, using a two step process. In the first step we construct the derivative f'. In the second step we seek a function f having that derivative.

Existence: Given the function $\theta(t)$ on $\partial\Omega$, it is always possible to solve the Dirichlet problem, extending $\theta(t)$ harmonically into the interior of Ω, thus obtaining $\theta(z)$. Since Ω is simply connected, $\theta(z)$ always admits a harmonic conjugate function, denoted by $\phi(z)$. It is then possible to construct the holomorphic function $g(z)=\phi(z)+i\theta(z)$. Since the complex exponential function is holomorphic, so is $\exp(g(z))$. A consequence of Cauchy's integral theorem is that if a function is holomorphic in a simply connected domain it always has an antiderivative, which means that $\exp(g(z))$ is the derivative of some holomorphic function f. In other words $f'=\exp(g(z))$. Since f' cannot vanish, f is conformal.

Uniqueness: Assume that two conformal maps f_1 and f_2 have the same angular factor $\theta(t)$ on the boundary. Since the derivatives f_1' and f_2' cannot vanish, $\mathrm{Log}(f_1')=\phi_1+i\theta_1$ and $\mathrm{Log}(f_2')=\phi_2+i\theta_2$ are holomorphic. It follows that θ_1 and θ_2 are harmonic functions. The uniqueness of the solution to the Dirichlet problem and the fact that θ_1 and θ_2 coincide on the boundary implies that $\theta_1=\theta_2$. Since both ϕ_1 and ϕ_2 are harmonic conjugate to $\theta_1=\theta_2$, this means that $\phi_1=\phi_2+k$ where k is a real scalar. It follows that $f_1'=\exp(\phi_1+i\theta_1)=\exp(\phi_2+k+i\theta_1)=\exp(k)\exp(\phi_2+i\theta_2)=\exp(k)f_2'$. We conclude that f_1' and f_2' differ only by scale, thus f_1 and f_2 differ only by scale and translation.

This shows that there is a one-to-one correspondence between conformal maps of a given simply connected domain Ω and real-valued functions defined on $\partial\Omega$. This greatly simplifies the task of searching for a particular conformal map since we may restrict our search to the low-dimensional linear subspace of real functions defined on the boundary of the domain.

In order to reconstruct a conformal map from its representation $\theta(t)$ in practice, we will need efficient techniques for finding a harmonic conjugate function and reconstructing a holomorphic function from its derivative. These will be described in the following sections.

4.3 Shape Interpolation

The process of animation creation consists of several steps. The animator selects a particular point in time, poses and deforms the objects in the scene, and sets a keyframe. The animator then "plays" the animation, which in turn deforms the objects in the scene by interpolating the shapes between each two consecutive keyframes. The animator then repeats this process in order to correct and refine the animation. This is a tedious process and a good animation tool strives to minimize the manual effort needed.

Our conformal shape deformation framework has an inherent ability to interpolate shapes in a shape-preserving way. Achieving that using our shape representation is very simple. Given two (or more) deformed shapes A and B, we simply blend the shape representations of A and B in a linear manner to obtain a new shape representation:

$$\theta^C\left(t\right) = (1-\mu)\theta^A\left(t\right) + \mu\theta^B\left(t\right)$$

such that $\mu \in [0,1]$ is a parameter that controls the interpolation. The same arguments that allowed the successful reconstruction of a conformal map from $\theta^A(t)$ and $\theta^B(t)$ can be applied to $\theta^C(t)$. This leads to a very simple and efficient algorithm for shape interpolation with some guaranteed properties. The deformations obtained using our method are very natural, avoid local intersections and are conformal everywhere. This is demonstrated in Fig. 3 but even more so in the accompanying video. In contrast to naïve linear morphing methods, which operate directly in the spatial domain, no shrinkage effects occur and the overall boundary length and shape area is nicely preserved (when it is equal in A and B). Also, as opposed to other interpolation methods which are limited to rotations of up to 180 degrees, our method is capable of interpolating shapes which undergo arbitrarily large rotations (even much larger than 2π) between shapes.

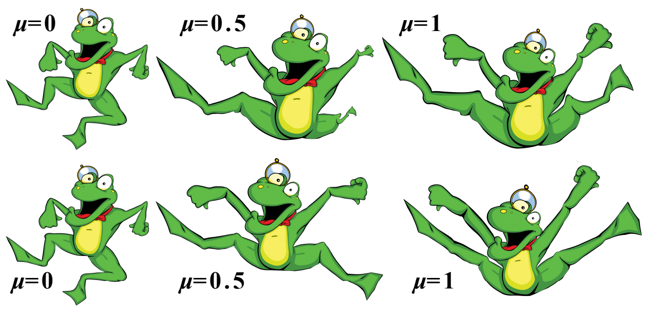

Figure 3: *Deformation and interpolation: (top row) Regular Cauchy coordinates. Note the shrinkage and rounding of the legs. (bottom row) Our method. Note the sharp bend in elbow and knee.*

5 Cauchy Coordinates

Since planar deformations may be viewed as a function from the complex plane to itself, it is advantageous to develop the necessary theory in the complex plane. We start by reviewing some basic facts from complex function theory. Consider the following complex function of two variables:

$$C(w,z)=\frac{1}{2\pi i}\frac{1}{w-z}$$

called the *Cauchy kernel*. The *Cauchy transform* of a function f - a piecewise smooth function defined on the boundary of Ω - a region of the complex plane - is defined as the following complex boundary integral [Bell 1992], producing a new function g on Ω, as illustrated in Fig. 4:

$$g(z)=\oint_{\partial\Omega}C(w,z)f(w)dw=\frac{1}{2\pi i}\oint_{\partial\Omega}\frac{f(w)}{w-z}dw \qquad (2)$$

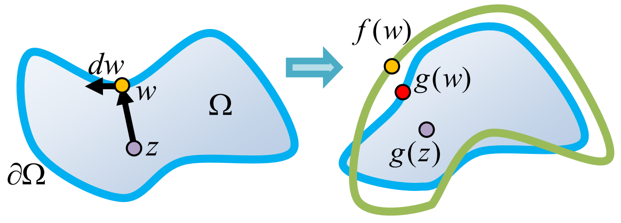

Figure 4: *Continuous planar mapping from* Ω *to* $g(\Omega)$ *generated by the Cauchy transform of* $f(\partial\Omega)$. *If* f *is not holomorphic then* $g(\partial\Omega)\neq f(\partial\Omega)$.

The Cauchy transform has various desirable properties. One is that, under mild assumptions on f, g is always holomorphic on Ω [Bell 1992, Theorem 3.1]. Hence, when using this transform in the context of planar shape deformation, if the derivative of f does not vanish, the mapping will be conformal. Specifically, the Cauchy transform reproduces any holomorphic function from its boundary values. Unfortunately, as Fig. 4 illustrates, the Cauchy transform does not possess the interpolation property, namely in general $f(w)\neq g(w)$ on $\partial\Omega$ if f is not holomorphic. This is not surprising since it is well known that (in contrast to harmonic maps), in general, it is impossible to generate a conformal map while prescribing its exact behavior on the boundary.

5.1 Regular Cauchy Coordinates

Complex barycentric coordinates were introduced by Weber *et al.* [2009] who allowed barycentric coordinates, which are traditionally real-valued basis functions, to assume complex values. When the basis functions are holomorphic, their linear combinations, using complex coefficients, may be used to generate detail preserving deformations of a 2D region. Weber et al. [2009] describe several recipes for such complex barycentric coordinates, the most fundamental being the Cauchy coordinates, based on the Cauchy transform (2) described above, which were proved to be equivalent to the 2D version of the Green coordinates [Lipman et al. 2008]. The Cauchy coordinates were derived by discretizing $\partial\Omega$ into a set of straight lines, namely a polygon (the so-called *cage*), reducing the integral (2) to:

$$g(z)=\frac{1}{2\pi i}\sum_{j=1}^{n}\oint_{e_j}\frac{f(w)}{w-z}dw \qquad (3)$$

Under the assumption that f is linear on each edge and continuous at the cage vertices, the integral (3) has a closed-form expression leading to an elegant formula for the (regular) Cauchy coordinate function $C_j(z)$ at vertex j (see Fig. 5, left, for the exact notations):

$$g(z)=\sum_{j=1}^{n}C_j(z)f_j \qquad C_j(z)=\frac{1}{2\pi i}\left(\frac{B_{j+1}}{A_{j+1}}\log\left(\frac{B_{j+1}}{B_j}\right)-\frac{B_{j-1}}{A_j}\log\left(\frac{B_j}{B_{j-1}}\right)\right)$$

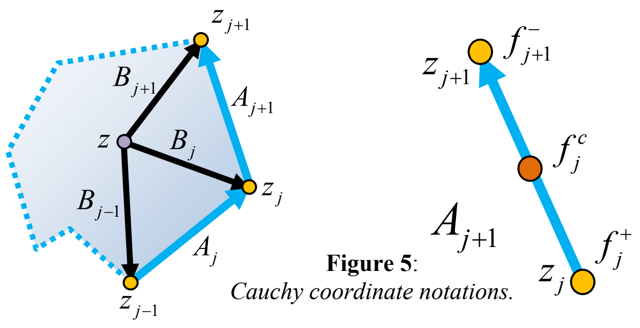

Figure 5: *Cauchy coordinate notations.*

where n is the number of vertices of the cage and f_j is the value of f at the jth vertex. Note that the Cauchy coordinates $C_j(z)$ are not defined on $\partial\Omega$, however the limit, as z approaches $\partial\Omega$, exists and the function is well behaved.

Conformal maps are angle preserving, however, the boundary curve is not considered to be part of the domain. For example, consider a conformal map f of a square to a disk, where it is clear that the boundary angles are changed. Examining $\theta(z)=\mathrm{Log}(|f'(z)|)$ on the boundary will reveal that the function is not continuous at the corners of the square.

The Cauchy transform spans the entire linear space of holomorphic functions. However, its discretized version, defined in (3), which we use in order to approximate the holomorphic function $\theta(z)$, spans only an n dimensional subspace of the holomorphic functions. It turns out that this subspace is relatively limited, for two main reasons. First, the Cauchy basis functions do not contain any singularity. They vary smoothly everywhere, even in the vicinity of the cage vertices. Since any linear combination will inherit the properties of the basis functions, it is not possible to properly express near-singular holomorphic functions. In particular, it cannot support changes in angle at the cage vertices. The second reason is due to the implicit assumption that f behaves linearly on the polygonal edges. Most holomorphic functions behave quite differently and a linear approximation is just not good enough.

5.2 Generalized Cauchy Coordinates

We now show how to extend the regular Cauchy coordinates in two ways. First, we allow f to have quadratic rather than linear behavior on each edge. Second, and most important, we allow f to have two distinct values at each singular vertex (a subset of the cage vertices), introducing a discontinuity for f at the vertex if the values are different. These two properties are achieved by prescribing three values for f on the edge A_{j+1}: f_j^+, f_j^c and f_{j+1}^- (see Fig. 5, right) such that:

$$f_j^+=f(z_j^+),\qquad f_j^c=f\left(\frac{z_j+z_{j+1}}{2}\right),\qquad f_{j+1}^-=f(z_{j+1}^-)$$

Given these, f can be expressed on the edge A_{j+1} as follows:

$$\begin{aligned}f(w)=&\frac{f_j^+}{A_{j+1}^2}\left(2w^2-wz_j-3wz_{j+1}+z_jz_{j+1}+z_{j+1}^2\right)\\&+\frac{f_j^c}{A_{j+1}^2}\left(-4w^2+4wz_j+4wz_{j+1}-4z_jz_{j+1}\right)\\&+\frac{f_{j+1}^-}{A_{j+1}^2}\left(2w^2-3wz_j-wz_{j+1}+z_j^2+z_jz_{j+1}\right)\end{aligned}$$

The next step is to plug f into (3) and evaluate the integral. The derivation is long and technical and is omitted here for brevity. Fortunately, as with the regular Cauchy coordinates, the integral has a closed-form solution which gives rise to the *generalized Cauchy coordinates*:

$$CE_j(z) = \frac{1}{\pi i A_{j+1}^2}\left(A_{j+1}\left(B_{j+1}+B_j\right) - 2B_{j+1}B_j \log\left(\frac{B_{j+1}}{B_j}\right)\right)$$

$$CV_j^-(z) = \frac{B_{j-1}}{2\pi i A_j^2}\left(\left(B_j+B_{j-1}\right)\log\left(\frac{B_j}{B_{j-1}}\right) - 2A_j\right)$$

$$CV_j^+(z) = \frac{B_{j+1}}{2\pi i A_{j+1}^2}\left(\left(B_{j+1}+B_j\right)\log\left(\frac{B_{j+1}}{B_j}\right) - 2A_{j+1}\right)$$

While the number of regular Cauchy coordinate functions is n (one coordinate for each vertex), here we have one coordinate function for each edge (CE) and an additional two coordinate functions (CV^- and CV^+) for each vertex, resulting in $3n$ coordinate functions. Finally, the generalized discrete Cauchy transform is:

$$g(z) = \sum_{j=1}^{n}\left(CE_j(z) f_j^c + CV_j^-(z) f_j^- + CV_j^+(z) f_j^+ \right)$$

It is important to note that even though the basis functions are bounded holomorphic functions at any point inside Ω, the limit of $CV_j^-(z)$ and $CV_j^+(z)$ at the vertex z_j does not exist, rather has a logarithmic singularity there. The singularity drastically improves the expressive power of the discrete Cauchy transform, and we refer to this as the *discontinuous case*. If discontinuity is not needed (i.e., we assume that $f_j^+=f_j^-$), it is possible to reduce the number of basis functions by combining the two coordinates at each vertex:

$$CV_j(z) = CV_j^-(z) + CV_j^+(z)$$

In this case the total number of basis functions is reduced to $2n$ and the transform has the following form, which we refer to as the *continuous case*:

$$g(z) = \sum_{j=1}^{n}\left(CV_j(z) f_j + CE_j(z) f_j^c \right)$$

Note that in this case the limit of $CV_j(z)$ at the vertices exists as in the regular Cauchy coordinates. Furthermore, if the values at the mid-edges equal the average of the values at the edge endpoints, i.e. $f_j^c=(f_{j+1}^- + f_j^+)/2$, the generalized Cauchy coordinates will reduce to regular Cauchy coordinates. Note also that generalized Cauchy coordinates reproduce both linear and quadratic functions.

Fig. 6 shows a visualization of the real and imaginary parts of the generalized Cauchy coordinates for a non-convex polygon.

6 Dirichlet Problems and Hilbert Transforms

In Section 4.2 we explained how to reconstruct a conformal map from its shape representation (prescribed angular factor on the boundary). This required the solution of a Dirichlet problem and the computation of a harmonic conjugate function. In this section, we describe an efficient, elegant and accurate numerical method for such computations.

The operator that takes a function u to its harmonic conjugate function v is the *Hilbert transform*. The classic definition of the Hilbert transform operates on real-valued functions of a single real variable, however, for our applications we prefer to use an alternative definition where u and v are real-valued functions of a single *complex* variable defined on the upper part of the complex plane such that the transform is applied to their restriction to the real axis. This is equivalent since it is always possible to uniquely extend a real-valued function defined on the real axis to be harmonic on the upper complex plane. Once the extended harmonic function is found, it is possible to find its harmonic conjugate and again, its restriction to the real axis is considered as the transformed real-valued function.

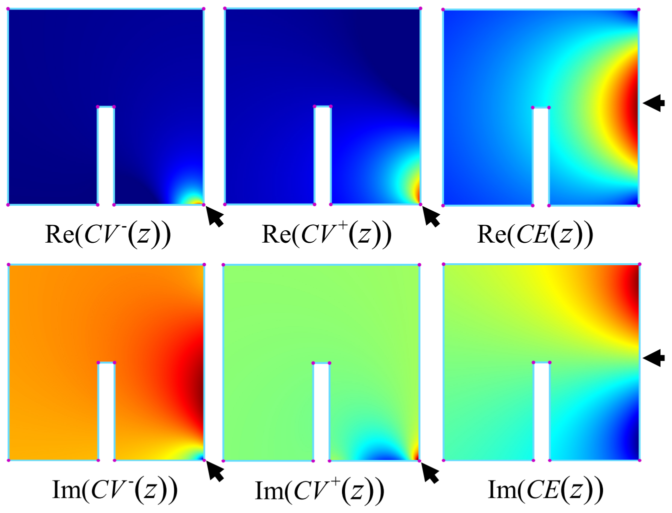

Figure 6: *Visualization of the generalized Cauchy coordinates for the marked vertex and edge.*

The same logic can be used to define the Hilbert transform on more general domains. For a simply connected domain Ω, consider the Hilbert transform [Bell 1992] as an operator that takes a real-valued function $u(s)$ on $\partial\Omega$ to the holomorphic function $\mathcal{H}(z)=u+iv$ on Ω. The transform always exists and is unique up to additive imaginary constant. However, in general, it does not possess a closed-form expression and computing it requires some effort.

The Boundary Element Method (BEM) [Kythe 1995] is a numerical computational method of solving linear partial differential equations such as Laplace's equation. The main idea is to formulate the problem as a boundary integral equation and then discretize the *boundary* into finite elements and approximate the equation as a linear combination of a set of basis functions with unknown coefficients. The coefficients are obtained as a solution to a dense linear system. Once the coefficients are found, the solution can be evaluated at any arbitrary point inside the domain. In contrast to the Finite Element Methods (FEM), BEM is based on discretization of the boundary alone and avoids discretization of the entire domain. The advantage is that the complexity of the boundary, which dictates the complexity of the solution, is decoupled from the complexity of the domain. The Complex Variable Boundary Element Method (CVBEM) [Hromadka et al. 1987] is a variant of BEM for solving two dimensional problems based on complex numbers. An advantage of CVBEM over BEM is that the derivation of the boundary integrals is done using complex analysis which greatly simplifies the derivation and leads to closed-form expressions. But more importantly, as we will elaborate on later, this technique is more natural if our ultimate goal is to construct *dual pairs* of harmonic functions (which together form a holomorphic complex function). We now show how CVBEM combined with our generalized Cauchy coordinates leads to an efficient and elegant method for the computation of the Hilbert transform.

6.1 The Dirichlet Problem

Given the values $u(s)$ of a function on $\partial\Omega$, we want to solve the Dirichlet problem - find a harmonic extension $u(z)$ to Ω preserving the given boundary values. Once $u(z)$ is found we want to find its harmonic conjugate function $v(z)$. As we shall see next, the two problems can be solved simultaneously.

Since the real and imaginary parts of any holomorphic function $g(z)$ are harmonic and since the Cauchy transform spans the entire space of holomorphic functions, it follows that *any* harmonic function h may be expressed as:

$$h(z) = \mathrm{Re}\left(g(z)\right) = \mathrm{Re}\left(\frac{1}{2\pi i}\oint_{\partial\Omega}\frac{f(w)}{w-z}dw\right) \tag{4}$$

for some complex f. The following error functional measures the difference on $\partial\Omega$ between some arbitrary harmonic function $h(z)$ and the prescribed boundary conditions $u(s)$:

$$Err\left(h(z)\right) = \oint_{\partial\Omega}\left(u(s) - h(s)\right)^2 ds \tag{5}$$

We would like to find the complex function f defined on $\partial\Omega$ that minimizes this error. Note that there always exists a function f which produces zero error since the Dirichlet problem always has a solution. To solve the optimization problem in practice we substitute the continuous Cauchy transform with the finite-dimensional discrete transform $g(z)=\sum_{j=1}^{s}C_j(z)f_j$ and approximate the integral in (5) as a sum over a k-sampling of $\partial\Omega$. Denoting the generalized Cauchy coordinates by $C_j(z)=\varphi_j(z)+i\psi_j(z)$ and the complex coefficients by $f_j=f_j^x+if_j^y$, the optimization problem becomes:

$$\underset{\{f_j\}_{j=1}^{s}}{\mathrm{argmin}}\sum_{i=1}^{k}\left(u_i - h(w_i)\right)^2 \tag{6}$$

where: $\quad h(w_i) = \mathrm{Re}\left(g(w_i)\right) = \sum_{j=1}^{s}\varphi_j(w_i)f_j^x - \psi_j(w_i)f_j^y$

w_i are the locations of the k boundary samples and s is the total number of basis functions. Since the error (5) is quadratic, the optimization can be solved by solving a rather small dense (over-determined) linear system of equations with $2s$ real variables. Once the optimal real coefficients f_j^x and f_j^y are found, we can plug them into $\mathrm{Re}\left(\sum_{j=1}^{s}C_j(z)f_j\right)$ obtaining a formula for the desired harmonic function $h(z)$ at any interior point z. It is important to note that the function $h(z)$ only approximates the given boundary conditions. However, we cannot hope for better than this since an analytic expression for the Dirichlet problem does not exist (in general). Moreover, using this construction we can guarantee that $h(z)$ is harmonic in the continuous (as opposed to discrete) sense.

The reader may have noticed that in order to solve (6), the generalized Cauchy coordinates need to be evaluated on the boundary of the domain, where they are singular. We define the values of the coordinate functions on $\partial\Omega$ to be their limit (in case it exists) when approaching the boundary from the interior of Ω:

$$C_j(w) = \lim_{z\to w} C_j(z), \quad z\in\Omega,\ w\in\partial\Omega$$

These limits exist everywhere except at the singular vertices. We refer to Appendix A for the appropriate formulae.

6.2 The Hilbert Transform

We now turn to finding the harmonic conjugate function. The main reason why we chose to solve the Dirichlet problem using the Cauchy transform can now be revealed. The desired conjugate is simply: $p(z)=\mathrm{Im}\left(\sum_{j=1}^{s}C_j(z)f_j\right)$. Since the discrete Cauchy basis functions are holomorphic, it follows that $p(z)$ is harmonic and that $h(z)$ and $p(z)$ are *exactly* harmonic conjugate functions. We thus conclude that the Dirichlet problem and finding a harmonic conjugate are two equivalent problems since once we solve for the complex coefficients f_j, we obtain a closed-form expression for $h(z)$ as well as for $p(z)$. Finally, the discrete Hilbert transform that we seek is: $\quad \mathcal{H}(z) = \sum_{j=1}^{s}C_j(z)f_j$

The inverse of the Hilbert transform which takes the harmonic function v to the holomorphic function $\mathcal{H}^{-1}(z)=v-iu$ is easily obtained by multiplying $\mathcal{H}(z)$ by the complex number i.

7 The Antiderivative

In Section 6 we showed how to compute the Hilbert transform which enables us to construct a holomorphic function such that its real (or imaginary) part coincides with a given real-valued function along the boundary. Our ultimate goal is to reconstruct a conformal map from its prescribed angular factor along the boundary $\theta(t)$. Applying the inverse Hilbert transform to $\theta(t)$, results in the holomorphic function $g(z)=\mathrm{Log}(f'(z))$ from which we can evaluate $f'(z)=\exp(g(z))$ anywhere in Ω and in particular on $\partial\Omega$.

The next step is to reconstruct the actual mapping f. Even though computing f' using the discrete Hilbert transform as described in Section 6 guarantees that the approximated f' is holomorphic, its antiderivative does not possesses a closed-form expression. However, using the following observation we can avoid numerical “integration” of f'. Since f is holomorphic it can be approximated using the discrete Cauchy transform $f(z)=\sum_{j=1}^{s}C_j(z)d_j$. Differentiating with respect to z results in $f'(z)=\sum_{j=1}^{s}D_j(z)d_j$ where $D_j(z)=d\left(C_j(z)\right)/dz$ is the derivative of the generalized Cauchy basis functions, which have a closed-form expression. Following the same logic of Section 6, we employ the following discrete optimization problem:

$$\underset{\{f_j\}_{j=1}^{s}}{\mathrm{argmin}}\sum_{i=1}^{k}\left|f'(w_i) - \sum_{j=1}^{s}D_j(w_i)d_j\right|^2 \tag{7}$$

where w_i are the locations of k boundary samples and d_j are the complex coefficients that we solve for. In contrast to Section 6, here the linear system of equations is over the field of complex numbers. Once the linear system is solved and the coefficients d_j are found, they can be plugged into the discrete Cauchy transform formula in order to compute $f(z)$ at any arbitrary interior point z.

The expressions for the first derivative of the generalized Cauchy coordinates and their limits when an internal point approaches the boundary are given in Appendix A.

8 Deformation by Angle Prescription

Having laid down all the mathematical building blocks for our deformation algorithm, we can describe it from the user point of view. The user first defines a polygon with n vertices to serve as the boundary of the domain and a set of p singular points are selected among the cage vertices. To simplify the user interface, we allow the user to prescribe the angular change $\theta(t)$ only at the singular points rather than on any point on the boundary. Since the singular points are considered to be corner points we actually have two distinct angles for each one of them, resulting in $2p$ degrees of freedom. The values of $\theta(t)$ are then interpolated linearly along the boundary. We give the user the freedom to choose between two possible user interfaces.

8.1 Rotational Handles

This is the most direct user interface. A rotational handle manipulator is attached to each singular vertex. In addition, the user is required to place one anchor handle inside the domain in order to fix the global scale and translation degrees of freedom. To control the amount of rotation, the user first selects a particular handle. Rotating the handle adds the same amount of rotation to the tangent vectors just before and just after the selected corner, which results in an overall orientation change at the corner. Dragging the handle rather than rotating it causes the two tangent vectors to rotate in opposite directions, which has the effect of changing the corner angle rather than its orientation. During interaction, the system constantly updates the location of the handles to match the deformed location of the singular corners.

8.2 Cage-Based

Alternatively, the user can manipulate the vertices of the polygonal boundary curve (the cage). In this case, all the vertices of the cage are considered to be singular. The corner angles are taken to be equal to the angle of the edges of the target cage, which results in a deformed shape bound by straight lines. Fig. 8 show an example of mapping a sinusoidal bar to a triangle. Since the edges of the target cage are straight, the image of the bar is a perfect triangle.

In addition, the user has an additional parameter per edge. This can be thought of as a tension parameter. When it has a large value, the angular change $\theta_i^+(t)$ just after the ith singular vertex equals the angular change $\theta_{i+1}^-(t)$ just before the $(i+1)$th singular vertex. This results in a constant θ along the edge, so the edge is mapped to a straight line in the target image. When tension is low we reduce the angles by an amount proportional to the tension parameter. This has the effect of bending the edge to form a circular arc, creating a more relaxed conformal map with more moderate variations in scale. This is demonstrated in Fig. 7. The three degrees of freedom can be either prescribed directly at an internal anchor point or can be deduced from the vertices of the target cage by some heuristic such as finding a scale and translation that minimizes the distance between the target cage and the boundary of the target image.

9 Implementation Details

During interaction, the user constantly changes the boundary condition θ_i $i\in[1, 2p]$ at the singular vertices which results in immediate visual feedback in the form of the deformed conformal map. Since this is an interactive application running in real-time, it is advisable to solve the linear systems during runtime as quickly as possible, even at the price of a longer preprocessing time.

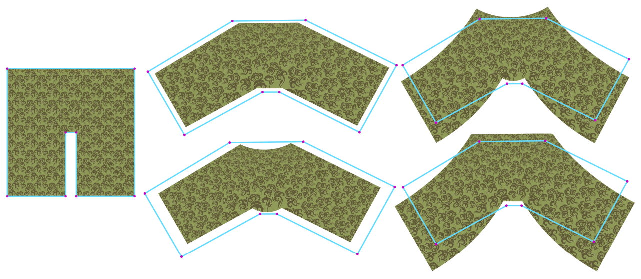
Figure 7: *Deformation using angle prescription: (Left) original shape enclosed by a cage. (Top left) all target edges are straight. (Top right) Tension is relaxed at all edges. (Bottom left and right) Tension is relaxed at only some of the edges.*

The number of basis functions needed in order to achieve good approximations depends on the complexity of the domain. Convex regions usually require less basis functions while highly concave and twisted regions require more. Hence, in practice we usually increase the number of basis functions by sampling the original edges of the cage, adding some *virtual* vertices. However, away from the singular vertices of the cage we expect that the mapping will behave well and will not contain radical changes, so at virtual vertices we can make do with the continuous case of generalized Cauchy coordinates, keeping the number of basis functions manageable. In contrast, at the singular vertices of the cage we always use the discontinuous case of generalized Cauchy coordinates in order to allow maximum flexibility. Denote by r the total numbers of vertices in the super sampled cage. We have one CE_j basis function per edge, one CV_j basis function per "regular" vertex and two basis functions CV_j^- and CV_j^+ per singular vertex which results in totally $s=2r+p$ basis functions. A typical value for p is 4-20 and a typical value for s is 200.

Once the user defines a cage with n vertices and a set of p singular vertices, the polygon is sampled and refined to have r vertices. We then uniformly sample the polygon with k samples, avoiding sampling the r virtual vertices, and evaluate the s generalized Cauchy coordinates on each sample. Choosing k to be an order of magnitude larger than r led to good results in our experiments.

Next, we construct the real matrix $C_{k\times 2s}$ containing the real and imaginary parts of the coordinates, φ and $-\psi$, at each sample in interleaved columns. Computing its pseudo-inverse and combing every two consecutive rows into a single row of complex numbers results in the matrix $C^+_{s\times k}$. We can express the solution to the optimization problem (6) in matrix form:

$$f_{s\times 1} = C^+_{s\times k} u_{k\times 1}$$

$u_{k\times 1}$ are the given values of u on the k boundary samples. Since we assume that u changes linearly between singular vertices, we can further express the samples $u_{k\times 1}$ as a matrix-vector product $u_{k\times 1}=S_{k\times 2p}\theta_{2p\times 1}$ where $S_{k\times 2p}$ is a "uniform" sampling matrix having exactly two non-zero elements in each row corresponding to two singular vertices. The rate of change is proportional to the arc length. It follows that:

$$f_{s\times 1} = \underbrace{C^+_{s\times k} S_{k\times 2p}}_{T_{s\times 2p}} \theta_{2p\times 1} = T_{s\times 2p}\theta_{2p\times 1}$$

Once the matrix T is computed, we obtain a formula for evaluating the discrete Hilbert transform at any point z inside Ω:

$$\mathcal{H}(z)=\sum\nolimits_{j=1}^{s}C_j(z)\sum\nolimits_{i=1}^{2p}T_{ji}\theta_i=\sum\nolimits_{i=1}^{2p}\theta_i\underbrace{\sum\nolimits_{j=1}^{s}C_j(z)T_{ji}}_{\mathcal{H}_i(z)}$$

We denote by $\mathcal{H}_i(z)$, the i^{th} Hilbert coordinate which has a closed-form expression. Once the $2p$ Hilbert coordinates are computed in the preprocess step, the Hilbert transform can be computed during interaction by a simple matrix-vector multiplication:

$$\mathcal{H}(z)=\sum_{i=1}^{2p}\mathcal{H}_i(z)\theta_i \tag{8}$$

Since θ is in fact the imaginary part (rather than the real part) of the holomorphic function $\mathcal{H}(z)$, we need the inverse Hilbert transform. This is obtained simply by multiplying θ by the complex number i before computing (8).

The $2p$ Hilbert coordinates are evaluated during the preprocess step at the k boundary samples and are stored in a matrix. During interaction, the user provides a vector of $2p$ angles which is multiplied by the Hilbert coordinates matrix. This results in a new vector of k complex numbers. Taking the exponent of each element in the vector produce a k-vector of boundary derivatives.

Our final task is then to construct a conformal map that has those derivatives along the boundary. The linear system (7) is also prefactored during preprocessing. Finding the optimal complex coefficients is achieved during interaction by a matrix-vector multiplication. Finally, each point $z \in \Omega$ is mapped to its target position using the discrete Cauchy transform. Note that computing the generalized Cauchy coordinates at all the internal points and storing them in a large matrix is also done in preprocessing.

There are three degrees of freedom that must be fixed. One is due to the Hilbert transform which corresponds to a constant addition to ϕ and has the geometric interpretation of global scale. The other two are due to the fact that the complex matrix containing the coordinates of the derivatives has co-rank one. This can be easily fixed by augmenting the real matrix $C_{k\times 2s}$ and the complex matrix of derivatives with one additional row.

We have implemented our algorithm as a plug-in to Autodesk Maya™. Computation of the pseudo-inverse matrices required in order to solve the linear systems (6) and (7) is done by SVD on our Intel i7 processor, using the Intel MKL library. This requires up to 15 seconds for our most complex shape. Since our method is based on barycentric coordinates, during interaction, we only need to perform dense matrix vector multiplications (an "embarrassingly parallel" process). This is done on a Nvidia Quadro FX 5800 graphics processor. The deformation is computed interactively even for huge images, giving the user immediate feedback, which is crucial for an effective animation session.

10 Computing Harmonic Coordinates

Harmonic coordinates [Joshi *et al.* 2007] have many useful properties, making them attractive for many graphics applications such as shape deformation and color and data interpolation. They are smooth, have the Lagrange (interpolation) property, and reproduce constant and linear functions. But their main advantage over the celebrated mean-value coordinates [Horman and Floater 2006] is that they are non-negative, even for non-convex boundaries. This property is especially important for shape deformation.

On the downside, harmonic coordinates do not possess a closed-form expression for general boundary shapes. Hence a good numeric approximation is sought. Joshi *et al.* [2007] compute the coordinates by discretizing the entire domain (2D or 3D) into piecewise–linear finite elements and solving a discrete Laplace equation for these elements. Martin *et al.* [2008] applied the Method of Fundamental Solutions [MFS] for computing harmonic coordinates on 3D polyhedral domains. The harmonic function is constructed as a linear combination of radial basis functions (the fundamental solution of the Laplace equation) with a set of real coefficients.

We obtain an alternative derivation for the computation of harmonic coordinates for simply connected 2D domains. The i^{th} harmonic coordinate is nothing but the real part of the Hilbert coordinate $\mathrm{Re}(\mathcal{H}_i(z))$. This derivation of harmonic coordinates possesses the same constant and linear reproduction properties as the discrete harmonic coordinates with the additional advantage of being *continuously* harmonic over Ω. Fig. 9 and the accompanying video demonstrates that high quality approximations for harmonic coordinates can be achieved when the generalized Cauchy coordinates are used, even with a very small number of basis functions.

11 Conclusions and Discussion

We have presented a novel method for 2D shape deformation with some *guaranteed* shape-preserving properties. Our method produces pure conformal maps, hence local foldovers are completely eliminated. We allow the user to augment the shape with a small set of singular points along the boundary.

Supported by a richer subspace spanned by a generalization of the complex Cauchy barycentric coordinates, we are able to create realistic deformations that have controllable sharp bends. The user has the ability to precisely prescribe the angular change along the boundary. The main tool is a natural representation for conformal maps. We achieve superb image quality and high performance thanks to efficient computation of the Hilbert transform, which is based on a new set of barycentric coordinates which we called Hilbert coordinates. Our method comes with a built-in ability to correctly interpolate shapes possessing the same shape-preserving properties. This is an important feature missing from all other deformation algorithms based on barycentric coordinates.

The optimization problem that we solve is linear. However, the final map is a non-linear function of the boundary angles. Although positional constraints can be added to (7), by doing so, we may lose the shape-preservation property (i.e. the map might not be injective anymore). An interesting direction for future research would be to solve a non-linear optimization problem in order to accommodate positional constraints.

Another direction for future research may be to define energies other than (5), optimizing both for orientation and scale constraints. Since satisfying both perfectly is impossible, the requirement can be relaxed to achieve a more balanced mapping with less scale variations and more local control.

We believe that Hilbert coordinates might be useful for other applications as well. For example, computing the Riemann map of a polygon to the unit disk is an important building block in many graphics and geometry applications. Initial results show that this can be done. Finally, we would like to find an analog to our theory in three dimensions, as an immediate extension does not seem to be possible. In fact, conformal maps in 3D rarely exist. Nonetheless, the theory of quasi-conformal maps may shed some light on this subject.

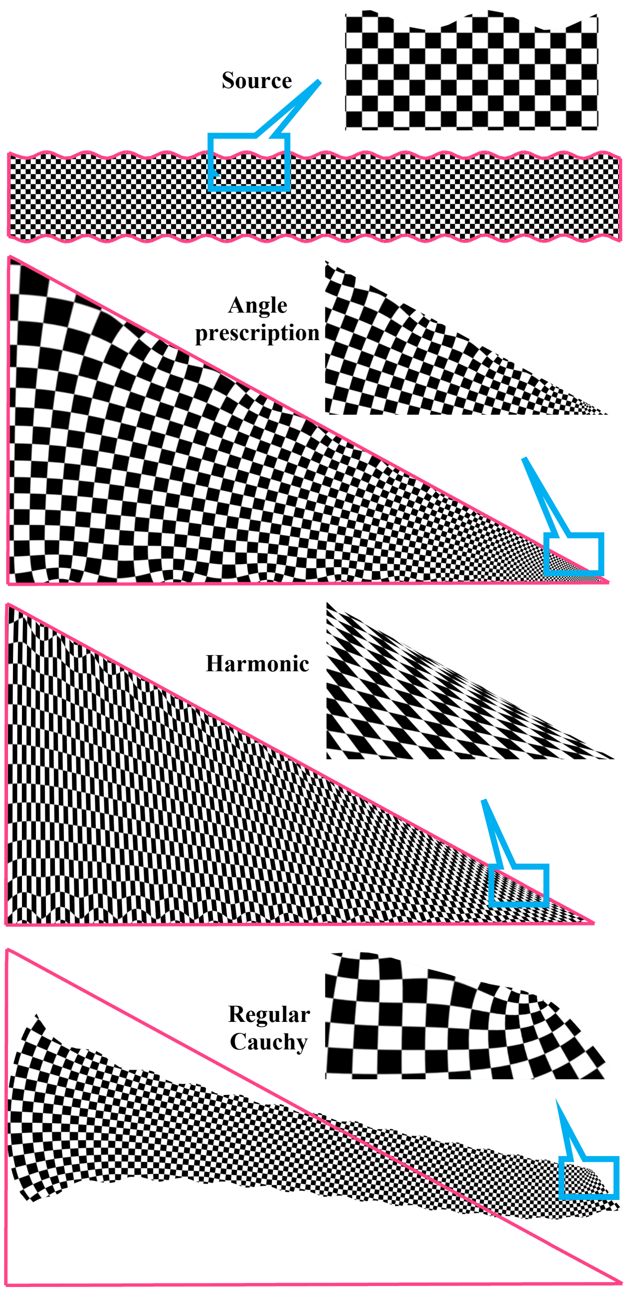

Figure 8: *Deformation of a bar with sinusoidal boundary into a triangle. (top to bottom) Source with a cage having 200 vertices, Conformal deformation using exact angle prescription, Result using harmonic coordinates (note the shear), Result using regular Cauchy coordinates (the shape does not follow the cage edges position nor orientation).*

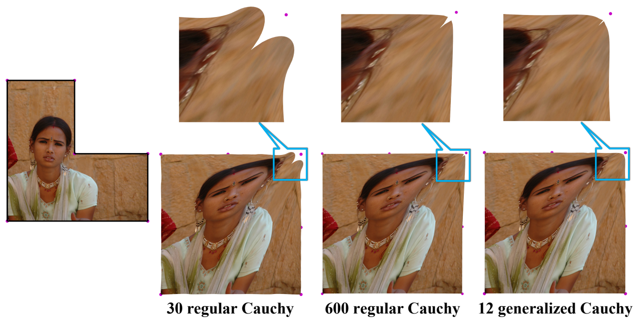

Figure 9: *Image deformation using our derivation of harmonic coordinates. (extreme left) original L-shaped image enclosed by a cage with 8 vertices, (left to right) deformation to a square using various numbers and types of basis functions.*

References

AHLFORS L. 1979. *Complex Analysis, 3rd Edition*. McGraw-Hill Science.

BELL S.-R. 1992. *The Cauchy Transform, Potential Theory and Conformal Mapping*. CRC-Press.

DUREN, P. 2004. *Harmonic mappings in the plane. Cambridge University Press.*

FLOATER M. S., HORMANN K., KÒS G. 2006. A general construction of barycentric coordinates over convex polygons. *Adv. Comp. Math. 24*, 1-4.

FLOATER M. S. 2003. Mean-value coordinates. *Comp. Aided Geom. Design 20*, 1, 19-27.

HORMANN, K., FLOATER M. S. 2006. Mean value coordinates for arbitrary planar polygons. *ACM Transactions on Graphics 25, 4.*

HORMANN, K., SUKUMAR, N. 2008. Maximum entropy coordinates for arbitrary polytopes. *Computer Graphics Forum*, 27, 5.

HROMADKA II, T.V., LAI, C., 1987. The Complex Variable Boundary Element Method in Engineering Analysis, Springer-Verlag Publishers.

IGARASHI T., MOSCOVICH T., HUGHES J.-F. 2005.-RIGID-AS-Possible shape manipulation. *ACM Transactions on Graphics (Proc. SIGGRAPH), 24*, 3.

JOSHI, P., MEYER, M., DEROSE, T., GREEN, B., AND SANOCKI, T. 2007. Harmonic coordinates for character articulation. *ACM Transactions on Graphics (Proc. SIGGRAPH), 26*, 3.

KARNI, Z., FREEDMAN, D., GOTSMAN, C. 2009. Energy-based content-aware image deformation. *Computer Graphics Forum (Proc. SGP), 28, 5.*

KYTHE, K., P. 1995. *An introduction to boundary element methods*. CRC Press.

LÉVY B., PETITJEAN S., RAY N., MAILLOT J. 2002. Least squares conformal maps for automatic texture atlas generation. *Proc. SIGGRAPH.*

LIPMAN, Y., LEVIN, D., COHEN-OR, D. 2008. Green Coordinates. *ACM Transactions on Graphics (Proc. SIGGRAPH), 27*, 3.

MARTIN, S., KAUFMANN, P., BOTSCH, M., WICKE, M., GROSS, M. 2008. Polyhedral finite elements using harmonic basis functions. *Computer Graphics Forum* 27, 5.

SCHAEFER S., MCPHAIL T., WARREN J. 2004. Image deformation using moving least squares. *ACM Transactions on Graphics (Proc. SIGGRAPH), 23*, 3.

WEBER, O., BEN-CHEN, M., GOTSMAN, C. 2009. Complex barycentric coordinates with applications to planar shape deformation. *Computer Graphics Forum* 28, 2.

Appendix A

The first derivative of the generalized Cauchy coordinates:

$$DE_j = \frac{2}{\pi i A_{j+1}^2}\left(\left(B_{j+1}+B_j\right)\log\left(\frac{B_{j+1}}{B_j}\right)-2A_{j+1}\right)$$

$$DV_j^- = \frac{1}{2\pi i A_j^2}\left(\frac{B_{j-1}A_j}{B_j}+3A_j-\left(4B_{j-1}+A_j\right)\log\left(\frac{B_j}{B_{j-1}}\right)\right)$$

$$DV_j^+ = \frac{1}{2\pi i A_{j+1}^2}\left(\frac{B_{j+1}A_{j+1}}{B_j}+3A_{j+1}-\left(4B_{j+1}-A_{j+1}\right)\log\left(\frac{B_{j+1}}{B_j}\right)\right)$$

The limit of the generalized Cauchy coordinates when z approaches the boundary:

$$\left.\begin{aligned} CV_j^-(z) &= \frac{-t}{2\pi i}\left(\left(1-2t\right)\left(\log\left(\frac{1-t}{t}\right)+\pi i\right)-2\right) \\ CV_j^+(z) &= \frac{1}{2\pi i}\left(\frac{(1-t)A_j}{A_{j+1}}+1\right)\left(\left(\frac{2(1-t)A_j}{A_{j+1}}+1\right)\log\left(\frac{A_{j+1}}{(1-t)A_j}+1\right)-2\right) \end{aligned}\right\} \begin{array}{c} z\in A_j \\ z=\left(1-t\right)z_{j-1}+tz_j \\ t\in\left(0,1\right) \end{array}$$

$$\left.\begin{aligned} CV_j^-(z) &= \frac{1}{2\pi i}\left(\frac{tA_{j+1}}{A_j}+1\right)\left(\left(\frac{2tA_{j+1}}{A_j}+1\right)\log\left(\frac{tA_{j+1}}{tA_{j+1}+A_j}\right)+2\right) \\ CV_j^+(z) &= \frac{1-t}{2\pi i}\left(\left(1-2t\right)\left(\log\left(\frac{1-t}{t}\right)+\pi i\right)-2\right) \\ CE_j(z) &= \frac{1}{\pi i}\left(1-2t+2t\left(1-t\right)\left(\log\left(\frac{1-t}{t}\right)+\pi i\right)\right) \end{aligned}\right\} \begin{array}{c} z\in A_{j+1} \\ z=\left(1-t\right)z_j+tz_{j+1} \\ t\in\left(0,1\right) \end{array}$$

$$CV_j^-(z_{j-1}) = CV_j^+(z_{j+1}) = 0$$
$$CV_j^-(z_j) = \text{undefined}$$
$$CV_j^+(z_j) = \text{undefined}$$
$$CE_j(z_j) = \frac{1}{\pi i}$$
$$CE_j(z_{j+1}) = -\frac{1}{\pi i}$$
$$CV_j(z_j) = CV_j^-(z_j)+CV_j^+(z_j) = \frac{1}{2\pi i}\log\left(\frac{\left|A_{j+1}\right|}{\left|A_j\right|}\right)+\frac{\measuredangle z_{j-1}z_jz_{j+1}}{2\pi}$$

The limit of the first derivative of the generalized Cauchy coordinates when z approaches the boundary:

$$\left.\begin{aligned} DV_j^-(z) &= \frac{1}{2\pi i A_j}\left(3+\left(1-4t\right)\left(\log\left(\frac{t}{1-t}\right)-\pi i\right)-\frac{t}{1-t}\right) \\ DV_j^+(z) &= \frac{1}{2\pi i A_{j+1}}\left(4+\frac{A_{j+1}}{(1-t)A_j}-\left(3+\frac{4(1-t)A_j}{A_{j+1}}\right)\log\left(\frac{A_{j+1}}{(1-t)A_j}+1\right)\right) \end{aligned}\right\} \begin{array}{c} z\in A_j \\ z=\left(1-t\right)z_{j-1}+tz_j \\ t\in\left(0,1\right) \end{array}$$

$$\left.\begin{aligned} DV_j^-(z) &= \frac{1}{2\pi i A_j}\left(4+\frac{A_j}{tA_{j+1}}-\left(3+\frac{4tA_{j+1}}{A_j}\right)\log\left(\frac{A_j}{tA_{j+1}}+1\right)\right) \\ DV_j^+(z) &= \frac{1}{2\pi i A_{j+1}}\left(3+\left(4t-3\right)\left(\log\left(\frac{1-t}{t}\right)+\pi i\right)-\frac{1-t}{t}\right) \\ DE_j(z) &= \frac{2}{\pi i A_{j+1}}\left(\left(1-2t\right)\left(\log\left(\frac{1-t}{t}\right)+\pi i\right)-2\right) \end{aligned}\right\} \begin{array}{c} z\in A_{j+1} \\ z=\left(1-t\right)z_j+tz_{j+1} \\ t\in\left(0,1\right) \end{array}$$

$$DV_j^-(z_{j-1}) = \text{undefined}$$
$$DV_j^+(z_{j+1}) = \text{undefined}$$
$$DE_j(z_j) = \text{undefined}$$
$$DE_j(z_{j+1}) = \text{undefined}$$
$$DV_j^-(z_j) = \text{undefined}$$
$$DV_j^+(z_j) = \text{undefined}$$
$$DV_j(z_j) = DV_j^-(z_j)+DV_j^+(z_j) = \text{undefined}$$