Eurographics Symposium on Geometry Processing 2016
Maks Ovsjanikov and Daniele Panozzo
(Guest Editors)

*Volume 35 (2016), Number 5*

# Iterative Closest Conformal Maps between Planar Domains

Aviv Segall and Mirela Ben-Chen

Technion - Israel Institute of Technology

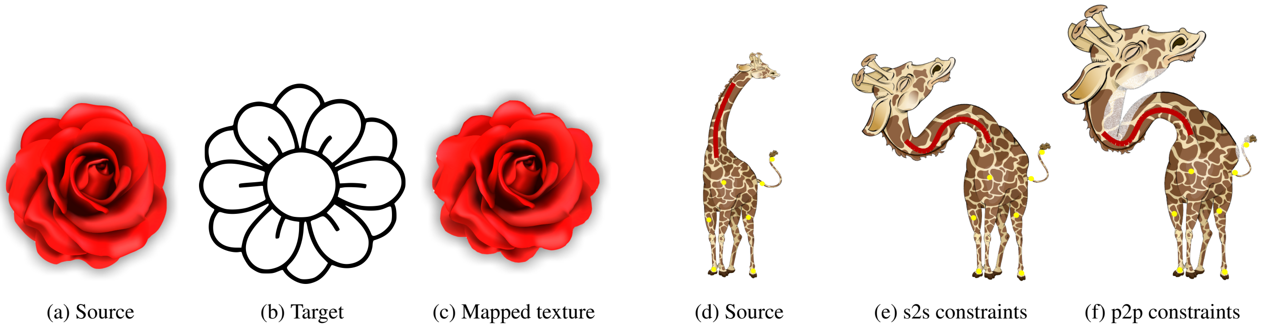

(a) Source (b) Target (c) Mapped texture (d) Source (e) s2s constraints (f) p2p constraints

Figure 1: Left: The rose (a) is given as a source shape and the line drawing (b) as the target. The conformal mapping found via the algorithm is used for transferring the texture from the source shape to the target shape (c). Right: Deformation of the giraffe using stroke-to-stroke constraints, compared to point-to-point constraints. Note that our method (e) generates a deformation with less area distortion (e.g., of the head), and without the foldovers near the head and tail evident when forcing point-to-point matching of points on the curve.

**Abstract**

*Conformal maps between planar domains are an important tool in geometry processing, used for shape deformation and image warping. The Riemann mapping theorem guarantees that there exists a conformal map between any two simply connected planar domains, yet computing this map efficiently remains challenging. In practice, one of the main algorithmic questions is the correspondence between the* boundaries *of the domains. On the one hand, there exist a number of conformal maps between any two domains, thus many potential boundary correspondences, yet on the other, given full boundary prescription a conformal map might not exist. Furthermore, an* approximate *boundary fitting can be enough for many applications. We therefore propose an* alternating minimization *algorithm for finding a boundary-approximating conformal map given only an initial global alignment of the two input domains. We utilize the* Cauchy-Green complex barycentric coordinates *to parameterize the space of conformal maps from the source domain, and thus compute a* continuous *map without requiring the discretization of the domain, and without mapping to intermediate domains. This yields a very efficient method which allows to interactively modify additional user-provided constraints, such as point-to-point and stroke-to-stroke correspondences. Furthermore, we show how to easily generalize this setup to quasi-conformal maps, thus enriching the space of mappings and reducing the area distortion. We compare our algorithm to state-of-the-art methods for mapping between planar domains, and demonstrate that we achieve less distorted maps on the same inputs. Finally, we show applications of our approach to stroke based deformation and constrained texture mapping.*

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—

## 1. Introduction

Conformal maps are often used in computer graphics for mesh parameterization [HLS07] and shape deformation [WG10] among many other applications. Given two simply connected polygonal domains, the Riemann mapping theorem [Rud87] guarantees that there exists a conformal map between them. Constructing this map efficiently, however, is challenging in practice.

One common approach, is to compose two conformal maps, one from the source to the unit disk, and the second from the unit disk to the target, thus reducing the problem to the case where one of the domains is the unit disk. Different methods were developed for this task, two of the renown ones being the Schwartz-Christoffel method [DT02] and circle packing [Ste99], where the first yields a continuous map, from any point in the source domain, and the latter requires a discretization of the interior, yet is guaranteed to converge to the smooth case under refinement [HS96]. This approach has been further generalized in computer graphics to quasi-conformal [CW15] maps and bijective harmonic maps [SH15].

Unfortunately, these methods suffer from a common drawback: as it is necessary to pass through an intermediate convex domain, any distortion incurred in the process could be visible in the final map. Furthermore, the additional degrees of freedom of the problem, namely all Mobius transformations from the unit disk to itself are not considered in this process. Finally, the method should be efficient to allow user control at interactive rates, as well as allow some flexibility since it is not necessarily needed to interpolate the target shape exactly.

We suggest instead to directly map between the source and target domains, by starting with an initial conformal map from the source domain, and iteratively refining it until it matches the boundary of the target domain. We represent a conformal map from the source domain using the *Cauchy transform* [Bel15], which (in the continuous case) spans the entire space of conformal maps from the domain. The Cauchy transform maps continuous functions defined on the boundary of the domain to *holomorphic* functions (complex differentiable functions which are conformal when their derivative does not vanish) in the interior of the domain. Specifically, it can reproduce a holomorphic map from its boundary values. Thus, given the boundary correspondence specified by a conformal map, we can extend this map to the interior of the domain using the Cauchy transform.

We therefore opt for an alternative minimization approach, jointly optimizing for the boundary correspondence and the conformal map. Specifically, we alternate between finding the closest conformal map for a given correspondence and updating the correspondence given a conformal map. This approach leads to a very simple algorithm, which runs at interactive rates and quickly converges to a high quality conformal map. We demonstrate the applicability of our approach using applications to texture transfer and shape deformation. In addition, we show that our method can be easily extended to handle quasi-conformal maps, point-to-point and stroke-to-stroke constraints, and that our results yield lower area distortion than state-of-the-art conformal mapping methods.

### 1.1. Related Work

Many methods for numerically constructing a conformal map from an arbitrary domain to the unit disk can be found in the literature, see e.g. [Kyt12] for a recent review. In engineering applications, the most common methods for this task are the Schwarz-Christoffel Mapping [DT02] and circle packing [Ste99]. However, most of these approaches are quite slow, requiring the solution of non-linear equations or a dense sampling of the domain to achieve sufficient accuracy. Among these, perhaps the closest to our approach is Wegmann's method [Weg86], which also iteratively solves for the boundary correspondence. However, we do not restrict one of the domains to be the unit disk (which can cause crowding [DV98]).

In Computer Graphics, planar conformal maps are popular for shape deformation (e.g. [WBCG09], [VMW15]), yet existing conformal approaches do not allow for stroke-to-stroke constraints which enable the curve constraints and the boundary to slide as we do. Alternatively, the boundary constraints can be specified by requiring the angles of the input polygon to be preserved [WG10]. We demonstrate that our approach leads to a better trade-off between the user's constraints and the resulting area distortion. Another iterative approach which does allow boundary sliding is suggested in [ESA07], yet it requires the discretization of the domain and does not output a smooth conformal map.

### 1.2. Contributions

Our main contribution is a fast iterative algorithm for producing conformal maps between two simply connected planar domains, without prescribing boundary correspondence (§2). In addition, we:

- Introduce stroke-to-stroke constraints, which can be used for deforming a given shape in an intuitive way and for guiding a conformal map between two domains (§3.3).
- Show how to incorporate a quasi-conformal energy to reduce the area distortion (§3.2).
- Show applications of our algorithm to image deformation and constrained texture mapping (§4).

## 2. Iterative Closest Conformal Mapping

Given an input source shape $\Omega_s$ and a target shape $\Omega_t$, both simply connected planar domains, we seek for a conformal map of the source domain, which maps its boundary to the boundary of the target domain. The Riemann mapping theorem [Rud87] states that for any simply connected domain $\Omega \subset \mathbb{C}$, there exists a bijective holomorphic map $f$ from $\Omega$ to the unit disk $U = \{z \in \mathbb{C} : |z| < 1\}$. An immediate corollary of the theorem is that for any two simply connected domains $\Omega_s, \Omega_t$ and the bijections $f_1 : \Omega_s \to U, f_2 : \Omega_t \to U$, one can construct a bijective holomorphic map between the domains $f : \Omega_s \to \Omega_t$, where $f$ is given by $f = f_2^{-1} \circ f_1$.

However, since the space of exact holomorphic maps from one domain to another is quite small and difficult to compute, we would like to relax the problem and gain flexibility to control the behavior of the map. Therefore, we define the following energy, which promotes boundary fitting as a *soft constraint*.

**Energy.** Given a mapping $f : \Omega_s \to \mathbb{C}$ defined over the source domain $\Omega_s$ with boundary curve $S$, we define an energy for measuring its closeness to the target domain $\Omega_t$ with boundary curve $T$ as:

$$E_c(f) = \oint_S d(f(s), T)^2 ds \qquad (1)$$

where $d(w,T)$ is the minimal distance from the point $w$ to the boundary of $\Omega_t$: $d(w,T) = \min_{z \in T} |w - z|$.

**Discretization.** We represent the source and target domains by their boundary, discretized as a source polygon $S$ with $n$ vertices and a target polygon $T$ with $m$ vertices. The space of the holomorphic maps defined over the source polygon is given by a discretization of the *Cauchy transform* for polygons [WBCG09], which yields the *Cauchy-Green coordinates*. These coordinates express a subspace of the holomoprhic maps as complex-valued vectors in the range of a fixed complex matrix, depending only on $S$. Minimizing the discretized energy $E_c$ then boils down to solving a least-squares system using a fixed matrix.

### 2.1. Background - Cauchy-Green coordinates

The Cauchy transform [Bel15] is a widely used operator in complex analysis, which generates the space of holomorphic functions on a domain $\Omega$ from continuous functions $f(z)$ defined on the boundary $\partial\Omega$:

$$u(z) = \frac{1}{2\pi i}\oint_{\partial\Omega}\frac{f(w)}{w-z}dw, \quad z \in \Omega. \tag{2}$$

The function $u$ is holomorphic in the domain, and when $f$ corresponds to the boundary values of a holomorphic function, the Cauchy transform will reproduce $f$.

The Cauchy coordinates [WBCG09] are a discretized version of the integral (2). If we discretize $\Omega$ using a polygon $\{z_i\}_{i=1}^n$, then given samples of the function $f_i = f(z_i)$ and by interpolating them linearly on the edges of the polygon, we can calculate the integral analytically. The integration yields $n$ functions $C_i(z)$, denoted as the *Cauchy-Green coordinates*, such that the value of the integral (2) is given by:

$$u(z) = \sum_{i=1}^{n} C_i(z) f_i. \tag{3}$$

Note, that while the boundary of the domain is discretized as a polygon, the interior of the domain is *continuous*, thus $z$ can take the value of any point in the domain. We provide the expressions for $C_i(z)$ given the input polygon in Appendix A.

### 2.2. Algorithm

**Discrete energy.** The source polygon is sampled at $r$ points (not necessarily at the vertices), denoted by $\{z_j\}$. If we are given the boundary correspondence to the target, namely for each of the sampled points, we have a corresponding point on the target domain boundary $w_j$, then the energy $E_c$ is discretized by:

$$E_c(\{f_i\}) = \sum_{j=1}^{r} |\sum_{i=1}^{n} C_i(z_j) f_i - w_j|^2.$$

Since the correct correspondence is not known in advance, we set $w_j$ as additional optimization variables, and constrain them to be points on the edges of the target polygon $T$. If we pack the coordinates $C_i(z_j)$ of each sample in a coordinate matrix $C \in \mathbb{C}^{r\times n}$, such that row $j$ contains the coordinates of sample $z_j$, then the discretized energy is:

$$E_c(\hat{f},\hat{w}) = ||C\hat{f} - \hat{w}||^2, \tag{4}$$

where $\hat{f} \in \mathbb{C}^n, \hat{w} \in \mathbb{C}^r$ are complex vectors with entries $\{f_i\},\{w_j\}$, respectively.

**Alternating minimization.** We obtain the following minimization problem:

$$\min_{\hat{f},\hat{w}} E_c(\hat{f},\hat{w}), \quad \text{s.t. } \hat{w} \in T. \tag{5}$$

We solve the minimization problem using a local-global approach, by alternating between minimizing it with respect to the conformal map given by $C\hat{f}$ (the global step), and the corresponding points on the target, $\hat{w}$ (the local step).

The initial points $w^0$ are obtained by sampling the target boundary with respect to the arclength of the source boundary sampling, and the coefficients $\hat{f}^0$ are initialized with zero. Then, at each iteration the coefficients $\hat{f}^k$ are updated by minimizing (4) with respect to $\hat{f}$. This is a linear least squares problem, where the minimizer is given by $\hat{f}^k = C^+\hat{w}^{k-1}$. Here $C^+$ is the pseudo-inverse of the matrix $C$, given by $C^+ = (C^*C)^{-1}C^*$, and $C^*$ is the conjugate transpose of $C$.

In the second step of each iteration we minimize the energy with respect to $\hat{w}$, under the constraint that $\hat{w}$ is a set of points lying on the target domain boundary. This is done by finding for each point $z_j$, transformed by the current mapping $\hat{f}^k$, the closest point on an edge (or a vertex) of the target polygon $T$. The algorithm is summarized in Algorithm 1.

```
Data: source polygon $S$, target polygon $T$
Result: set of coefficients $\hat{f}$ defining the closest conformal
        map found from $S$ to $T$
$\hat{z}$ = SamplePolygon(S);
$\hat{w}^0$ = SamplePolygon(T);
$\hat{f}^0 = 0$;
$k \leftarrow 0$;
while $E_c(\hat{f}^k,\hat{w}^k) > threshold$ do
    $\hat{f}^{k+1} \leftarrow C^+\hat{w}^k$;
    $\hat{w}^{k+1} \leftarrow$ ClosestPoint($T, C\hat{f}^{k+1}$);
    $k \leftarrow k+1$ ;
end
return $\hat{f}^k$
```

**Algorithm 1:** Iterative closest conformal mapping

**Convergence.** The Iterative closest conformal map (ICCM) algorithm always converges to a local minimum, since at each iteration the energy is reduced twice. First, by minimizing it with respect to the conformal map given by $\hat{f}$ while keeping $\hat{w}$ fixed and second, by projecting the points on the target polygon $T$, therefore solving for $\hat{w}$ while keeping $\hat{f}$ fixed. Figure 2 shows a typical execution of the algorithm and the energy $E_c$ during the iterations.

## 3. Extensions

### 3.1. P2P Constraints

For gaining more control over the behavior of the conformal map, we can add additional energies to the minimization problem. One possibility is the point-to-point energy $E_{P2P}$ which allows the user to guide the conformal map by specifying points in the source domain $u_i$ and their desired location in the target domain $v_i$. Packing the coordinates of the source points $u_i$ together, we obtain the matrix $C_{P2P}$ where each row contains the coordinates of a source point. Then, the P2P energy is expressed by:

$$E_{P2P}(\hat{f}) = ||C_{P2P}\hat{f} - \hat{v}||^2 \tag{6}$$

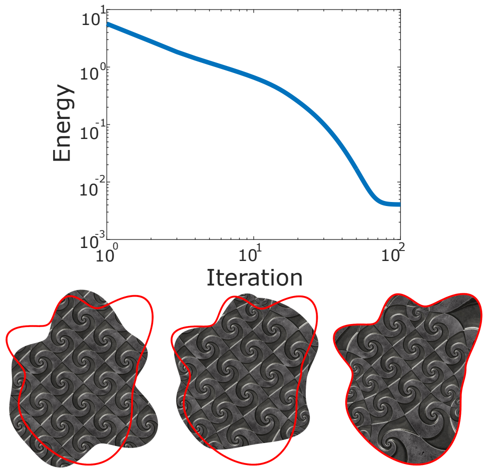


Figure 2: Top: the energy during the iterations of a typical execution of the algorithm on a log-log scale. Bottom, from left to right: the initial domain, the conformal mapping after a single iteration of the algorithm, and the conformal mapping after 100 iterations.

where $\hat{v}$ is the vector of target locations. Minimizing only this energy yields a conformal map which transforms the chosen source points as close as possible to the target positions. With both energies combined, we now seek to minimize $E = E_c(\hat{f}, \hat{w}) + \lambda E_{P2P}(\hat{f})$, where $\lambda$ is a parameter controlling the strength of the P2P energy. This optimization problem can be solved in a very similar way to the ICCM algorithm, the only difference being the fact that now the minimization with respect to $\hat{f}$ should be taken over the weighted least squares problem defined by the two energies. In addition, for the initialization step it is beneficial to calculate $\hat{f}^0$ by minimizing the $E_{P2P}$ energy, yielding a rough initial map, and set $\hat{w}^0$ to be the closest points on the target boundary to the sampled points mapped by $\hat{f}^0$. Figure 3 shows the result of running the algorithm with user defined P2P constraints.

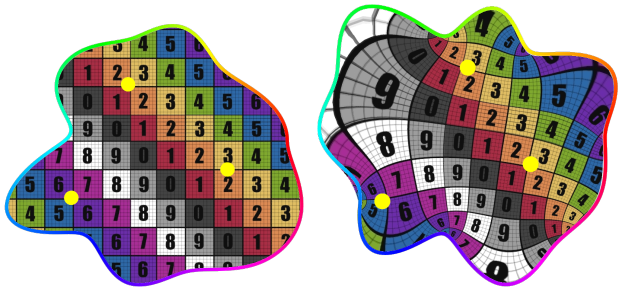


Figure 3: left: source domain and constrained points, right: the mapping obtained by the ICCM algorithm with the specified target boundary and P2P constraints.

### 3.2. Quasi-conformal maps

Since the number of degrees of freedom for a conformal map from one simply-connected domain to another is quite small, a possible way to extend the space of mappings is allowing the map to be quasi-conformal. Instead of looking for a holomorphic function, we will search for a *complex harmonic* function, where the real and imaginary parts are both real harmonic functions. It is known that any complex harmonic function $f$ can be decomposed as the sum of holomorphic and antiholomorphic functions. Thus, we will represent the complex harmonic function $f$ using the Cauchy-Green coordinates by:

$$f(z) = \Phi(z) + \Psi(z) = \sum_{j=1}^{n} C_j(z)\phi_j + \sum_{j=1}^{n} \overline{C_j(z)\psi_j} \tag{7}$$

where $\Phi(z) = \sum_{j=1}^{n} C_j(z)\phi_j$ is a holomorphic function and $\Psi(z) = \sum_{j=1}^{n} \overline{C_j(z)\psi_j}$ is an antiholomorphic function. Denoting by $C$ the matrix of coordinates for the sampled points (similarly to the previous section), the energy $E_c$ now becomes:

$$E_c = \left\| \begin{pmatrix} C & \overline{C} \end{pmatrix} \begin{pmatrix} \hat{\phi} \\ \hat{\psi} \end{pmatrix} - \hat{w} \right\|^2 \tag{8}$$

where $\hat{\phi}$ is the vector with entries $\phi_j$ and $\hat{\psi}$ is the vector with entries $\overline{\psi_j}$.

The dilatation of a mapping $f$ is defined as [AE66]:

$$D_f(z) = \frac{|f_z| + |f_{\bar{z}}|}{|f_z| - |f_{\bar{z}}|} = \frac{1 + |f_{\bar{z}}|/|f_z|}{1 - |f_{\bar{z}}|/|f_z|} \tag{9}$$

where $f_z = (\frac{\partial}{\partial x} + i\frac{\partial}{\partial y})f$ and $f_{\bar{z}} = (\frac{\partial}{\partial x} - i\frac{\partial}{\partial y})f$. The dilatation can be used for measuring the conformal distortion of the mapping: for a conformal map the dilatation is exactly 1 (since $f_{\bar{z}} = 0$, which is equivalent to satisfying the Cauchy-Riemann equations), and as it gets larger, the map distorts angles more. Therefore, for limiting the amount of conformal distortion the dilatation has to be minimized, which can be achieved by minimizing $|f_{\bar{z}}|$. Note that while a quasi-conformal map requires a bounded dilatation in the whole domain, we do not find this global bound, but attempt to minimize it by minimizing the values of $|f_{\bar{z}}|$ on the boundary. From the maximum

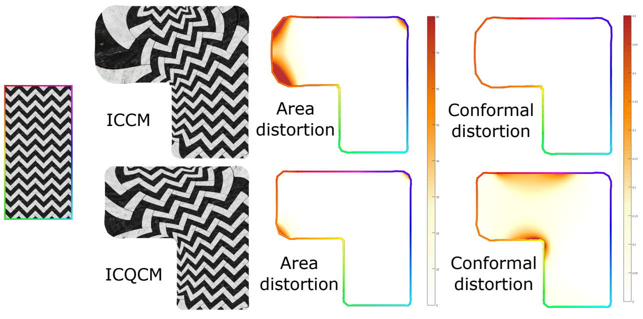


Figure 4: Left: The source polygon. Top row: The conformal map found by the ICCM algorithm, bottom row: quasi-conformal map found by the ICQCM algorithm. Note that we achieve less area distortion in the quasi-conformal mapping at the expense of some conformal distortion.

modulus principle [Kra12], since in our case $f_{\bar{z}}$ is antiholomorphic, the maximum value of $|f_{\bar{z}}|$ occurs on the boundary of the domain, and thus by minimizing $|f_{\bar{z}}|$ on the boundary, it will be minimized inside the domain as well.

In our representation, since $f$ decomposes as a sum of two holomorphic and antiholomorphic functions we get that $f_z = \Phi_z = D\hat{\phi}$ and $f_{\bar{z}} = \Psi_{\bar{z}} = \overline{D}\hat{\psi}$, where $D$ is the matrix with the derivatives of the coordinates for the sampled points, i.e. $D = \frac{\partial}{\partial z}C$. Therefore, we add the energy:

$$E_q = \|f_{\bar{z}}\|^2 = \|\overline{D}\hat{\psi}\|^2, \tag{10}$$

where the derivative is calculated for each element in the matrix, as described in Appendix A. Figure 4 compares the area and conformal distortion of the conformal and quasi-conformal maps achieved using our approach. Note the reduced area distortion near the boundary of the domain, at the expense of a small conformal distortion.

### 3.3. Stroke to Stroke mapping

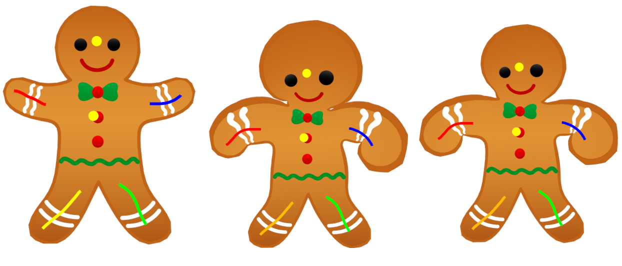
A similar idea to ICCM can be employed for extending the point-to-point constraints to curve-to-curve constraints. In this type of constraints, the user can draw a source stroke $Sk_s$ inside the domain and a target stroke $Sk_t$, and the goal would be to find a conformal mapping of the domain which maps between the drawn strokes. Formally, we define the energy $E_{S2S}$ in a very similar way to the energy $E_c$:

$$E_{S2S}(f) = \int_{Sk_s} d(f(s), Sk_t)^2 ds, \tag{11}$$

where $d(z,T)$ measures the minimal distance of the point $z$ from the curve $T$. The discretization of the energy is achieved using the Cauchy-Green representation of the conformal mapping $f$. First, the source and the target strokes are sampled uniformly at $n$ points $\{s_j\}_{j=1}^{n}$ and $\{t_j\}_{j=1}^{n}$. Next, we calculate the Cauchy-Green coordinates for the points sampled on the source stroke and pack them together in a matrix $C_{sk}$. Finally, the discretized energy is defined by:

$$E_{S2S}(\hat{f},\hat{t}) = \|C_{sk}\hat{f} - \hat{t}\|^2 \tag{12}$$

where $\hat{t}$ is the vector of points sampled on the target stroke. The minimization of this energy is done using a similar iterative algorithm to ICCM, where at each iteration a least squares problem is solved for the coefficients $\hat{f}^{k+1}$, and then the new set of points $\hat{t}^{k+1}$ is calculated by projecting the current mapping of the source points on the target stroke.



(a) Source (b) p2p (c) s2s

Figure 5: Stroke to stroke constraints used for shape deformation. Left: the original shape and the given constraints. Middle: deformation with point-to-point constraints (the points on the strokes are fixed). Right: deformation with stroke-to-stroke constraints (points are allowed to move along the strokes). Note that using the stroke-to-stroke constraints yields a lower area distortion for the hands and the head.

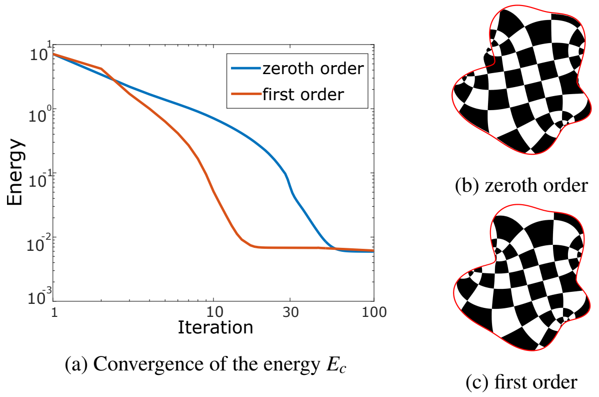
This energy, similarly to the P2P energy, can be used for guiding the conformal map when combined with the closeness energy $E_c$, but can also be useful for deforming a shape. In the latter scenario, it is beneficial to add a regularization term which is defined as an additional energy $E_s(\hat{f}) = \|D^{(2)}\hat{f}\|^2$, where $D^{(2)}$ is a matrix containing the second derivative of the Cauchy-Green coordinates for additional points sampled on the boundary of the domain. This energy is useful for fixing the degrees of freedom (when the number of constraints is smaller than the number of coordinate functions) and for preserving the smoothness of the boundary. Figure 5 shows the deformation found using the stroke-to-stroke constraints, and the comparison to the deformation found using similar point-to-point constraints. Note that since the points are allowed to move freely on the target stroke, the deformation found using the stroke-to-stroke constraints yields a smaller area distortion for the hands and the head.

### 3.4. Higher Order Approximation of the Distance Function

In the global step of the minimization problem (the optimization for $\hat{f}$), we used a zeroth order approximation of the distance function $d(z,T)$ at the mapped point $f(z_j)$, namely the squared distance to the closest point found in the previous step $w_j$. One could use, alternatively, a higher order approximation for the distance function as suggested in [PLH04]. The first order approximation of $d(z,T)$ is the distance to the tangent at the closest point $w_j$. Thus, using complex-variable notation the first order approximation is given by $d(z,T) \approx \mathrm{Re}\left((z - w_j)\overline{N_j}\right)^2$, where $N_j$ is the unit normal at $w_j$ and we have used the representation of a dot product between two vectors $a,b \in \mathbb{R}^2$ in complex form: $\langle a,b\rangle = \mathrm{Re}(a\bar{b})$. Integrating the first order approximation in the global step, we obtain the following minimization problem:

$$\hat{f}^{k+1} = \underset{\hat{f}}{\mathrm{argmin}} \,\|\, \mathrm{Re}\left(\overline{N}\left(C\hat{f} - \hat{w}\right)\right)\|^2 + \lambda\|C\hat{f} - \hat{w}\|^2 \tag{13}$$

where $N$ is the diagonal matrix with the entries $N_j$ on its main diagonal. Note that the zeroth order approximation is used here as a regularization term for stabilizing the energy when the algorithm is close to converge. This is still a simple least squares problem which can be solved by converting the complex variable formula to one with real variables. Note that the local step does not need to change since the distance function can be exactly calculated when the points $f(z_j)$ are fixed, therefore an approximation is not necessary. Figure 6 shows a comparison between the different approximation orders. Notice that while the zeroth order approximation works well for points far away from the curve, the first order approximation behaves better for points close to the curve, and the convergence is achieved much faster.



(a) Convergence of the energy $E_c$

(b) zeroth order

(c) first order

Figure 6: (a) Convergence of the energy using different orders of approximation for the distance function. (b), (c) the conformal map obtained by the zeroth and first order approximations at iteration 30, in which the first order approximation has converged.

## 4. Experimental Results

### 4.1. Implementation Details

We have implemented the ICCM algorithm and its extensions in MATLAB. Given a source polygon $S$ with $n$ vertices and a target polygon $T$, we sample the source polygon at $r$ points for creating the vector of points $P$ used for discretizing the closeness energy $E_c$. It is necessary to sample points at the edges of the polygon for constraining their mapping to be close to the target polygon. While a sparse sampling may result in mapping of points between the sampled ones to points far from the target polygon, sampling too dense might make the algorithm slow. The number of sampled points per edge should depend on its length and on the available computational resources. Since in our experiments the source polygons had approximately uniform edge lengths, we have found that sampling each edge at 4 points (including the vertices) is sufficient for achieving good results. The target polygon is sampled according to the accumulative arclength of the points in $P$, starting from the first point $P_1$. Next, we calculate the coordinates matrix $C$ of size $r \times n$, where row $j$ contains the coordinates of point $P_j$. The other energies $E_s, E_{P2P}, E_{S2S}$ are discretized in a similar way, with the coordinate matrices $D_s^{(2)}, C_{P2P}, C_{S2S}$. Finally, the conformal map is found by minimizing the combined energy:

$$E(\hat{f}) = E_c(\hat{f}, \hat{w}) + \alpha E_s(\hat{f}) + \beta E_{P2P}(\hat{f}) + \gamma E_{S2S}(\hat{f}, \hat{t})$$
$$= \|C\hat{f} - \hat{w}\|^2 + \alpha\|D^{(2)}\hat{f}\|^2 + \beta\|C_{P2P}\hat{f} - \hat{v}\|^2 + \gamma\|C_{S2S}\hat{f} - \hat{t}\|^2$$

Note that $E(\hat{f})$ is a quadratic energy in $\hat{f}$, and thus can be written as $E(\hat{f}) = \|A\hat{f} - b\|^2$. Since the coordinates matrices are constant during the iterations of the algorithm, we can calculate the pseudo-inverse of $A$ in advance, and use it during the iterations minimizing the energy with respect to $\hat{f}$ by multiplying $\hat{f}^{k+1} = A^+ b^k$. After the new coefficients $\hat{f}^{k+1}$ are found at each iteration, the vector $b^{k+1}$ is calculated by updating $\hat{w}^{k+1}$ and $\hat{t}^{k+1}$ to be the closest points on the target boundary and target strokes.

In the case of quasi-conformal mapping, the vector of coefficients $\hat{f}$ (with $n$ elements in the conformal case), is extended to contain $2n$ elements $\hat{f}^q = \begin{pmatrix}\hat{\phi} \\ \hat{\psi}\end{pmatrix}$, where the first $n$ elements represent the holomorphic function $\phi(z)$ and the last $n$ elements represent the antiholomorphic function $\psi(z)$. Additionally, each one of the coordinate matrices is concatenated from the right with its conjugate (i.e. $C^q = \begin{pmatrix}C & \overline{C}\end{pmatrix}$), so that the quasi-conformal function evaluated at the sampled points is given by $f(\hat{z}) = C^q \hat{f}^q$. In this case, the energy $E_q$ is also added as part of the weighted least squares problem.

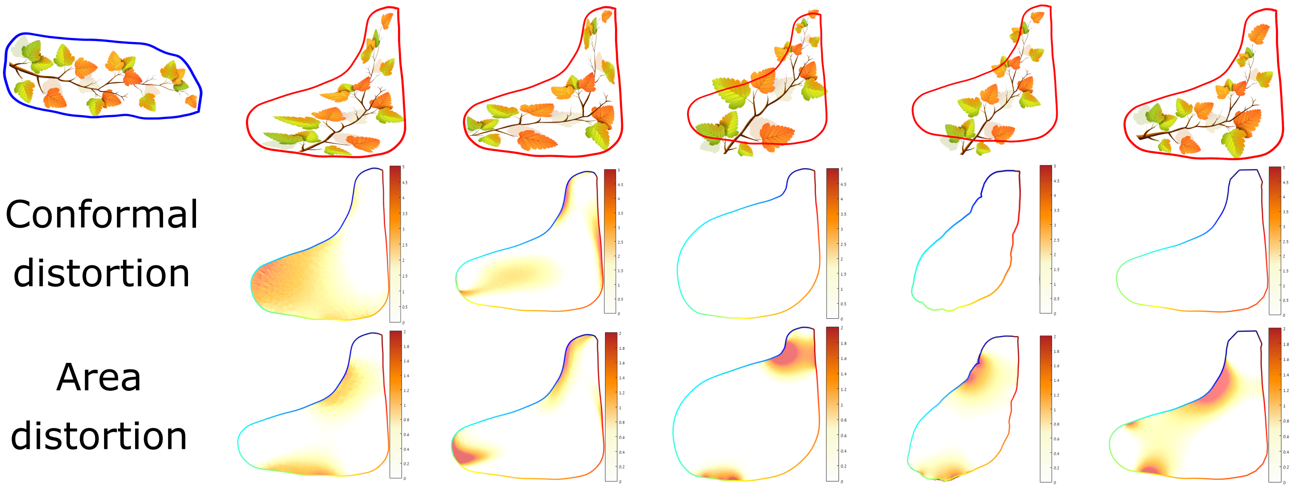
### 4.2. Limitations

One limitation of our method is that it does not have the option to control the area distortion. Therefore, the mappings found by the method may introduce large area distortions in order to minimize the energy. However, by searching for a quasi conformal mapping we have seen that the area distortion can be reduced at the expense of some conformal distortion.

In addition, our method does not prevent flipping, which can appear in the continuous holomorphic map obtained by the algorithm, and may not preserve the order of the points on the boundary. Furthermore, the algorithm depends on the initial boundary correspondence, and for a bad initialization, e.g., a convex part of the source is mapped into two different parts of the target, the optimization will converge to a local minimum.

### 4.3. Comparisons

We have compared our algorithm to several methods for mapping between planar shapes. Figure 7 shows the source polygon in blue, the target polygon in red, the mapping achieved with each of the methods, and the conformal and area distortions. In column (b) we show the results of the method described in [SH15] for constructing a smooth bijective map between arbitrary polygons via a convex regular polygon. We have discretized the interior of the two domains, constructed two harmonic maps from the domains into a convex regular polygon, and composed one with the inverse of the other in order to get a mapping between the two domains. In column (c) we show the results of the method in [ESA07], in which the interior of the domain is discretized, and an energy which measures the distance between the mapping of the source's boundary to the target's boundary and the distortion of the laplacian of the source mesh is minimized. Notice that both of these methods do not produce a conformal map and therefore introduce some conformal distortion. In column (d), we use the method from [WG10] for finding a conformal mapping which maps the angles of the source polygon to the angles of the target polygon. However, the length of the edges is not prescribed, and therefore the mapping does not interpolate the target polygon and can produce large area distortion as can be seen in the results. In column (e) we have used the Szego coordinates which were introduced in [WBCG09]. Our method is shown in column (f). In the example we have also used two p2p constraints in order to construct the initial boundaries correspondence and for guiding the source of the branch and the top leaf to their desired location. Notice that the method produces a conformal map, therefore there is no conformal distortion, but the area distortion is not constrained and therefore it introduces more area distortion than the other methods in some parts of the mesh.



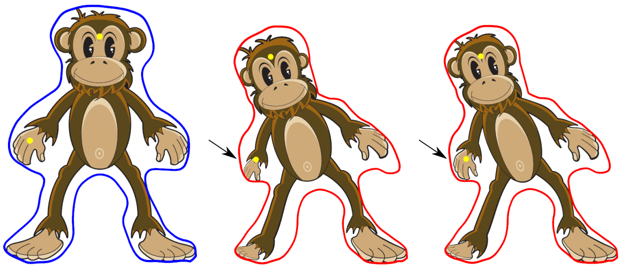
(a) Source (b) [SH15] (c) [ESA07] (d) [WG10] (e) Szego [WBCG09] (f) ICCM

Figure 7: Comparison of our algorithm to state-of-the-art methods for computing maps between planar domains. Note that our algorithm produces a conformal map with no shearing artifacts, but may introduce some area distortions.

### 4.4. Additional Results

**Deformations.** Figure 8 shows a deformation generated by mapping a source sketch to a target sketch, both drawn by the user. Two point-to-point constraints were used for guiding the mapping of the hand and the head of the monkey. Note that the quasi-conformal map better approximates the point-to-point constraints, as well as reduces the area distortion (see the point constraint in the hand).

**Constrained texture mapping.** Figure 9 shows how the algorithm can be used for constrained texture mapping. In this experiment we have used the method from [BCGB08] for calculating a conformal flattening of the given mesh, resulting in a 2D mesh which is conformal to the original. Next, we used our algorithm for finding a conformal map from the boundary of the texture (a square) to the boundary of the 2D mesh, and the point-to-point constraints were used for guiding the conformal map.



(a) Source (b) Conformal (c) Quasi-conformal

Figure 8: Deformation of the monkey using conformal vs quasi-conformal mappings. Note that the quasi-conformal map better approximates the point-to-point constraints, as well as reduces the area distortion (see the point constraint in the hand).

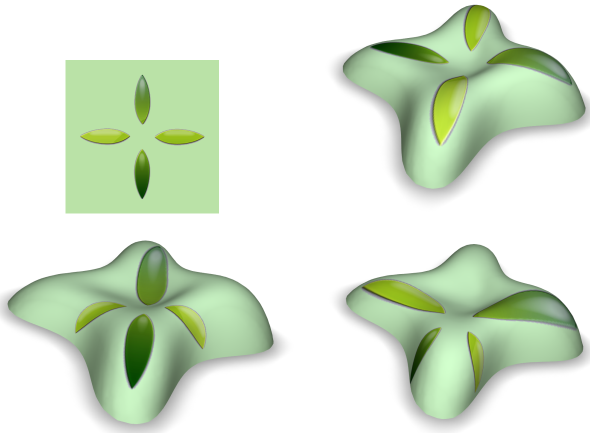


Figure 9: Constrained texture mapping. Top left: the input texture. Top right and bottom row: the texture is mapped to the surface, using point-to-point constraints as a guidance.

**Texture transfer.** In this experiment we calculated a conformal mapping from a hexagon into a flattened mesh, which is conformal to the target 3D mesh. Then, we have transfered the texture from the original shape to the target shape through composition of the two conformal maps. The results are shown in Figure 10.

## 5. Conclusions and Future Work

We have shown a fast algorithm for mapping between planar shapes, and proposed several extensions for controlling the map. In addition, we introduced stroke-to-stroke constraints, a generalization of point-to-point constraints, which can be used as an intuitive deformation framework. We demonstrated how the proposed

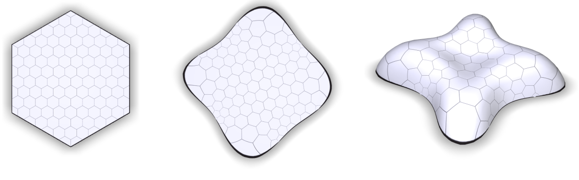

(a) Source (b) Flattened target (c) Target mesh

Figure 10: Texture transfer between shapes. The texture from the hexagon (a) was transfered to the lilium mesh (c) through composition of two conformal maps.

method can be used for deforming a shape by sketching a source and target boundary and for transferring textures.

While the mappings found using this method may introduce a large area distortion, we have found that it can be minimized by considering quasi-conformal mappings, and we believe it can also be controlled by adding an additional energy which will make the optimization non linear. It might also be possible to prevent flips and handle non-convex shapes better by preserving the order of the points when projecting them onto the target boundary. Another interesting direction will be calculation of mappings between three dimensional shapes, using a generalization of the Cauchy-Green coordinates.

### Acknowledgments

The authors acknowledge ISF grant 699/12, Marie Curie CIG 303511, and the German-Israeli Foundation for Scientific Research and Development, Grant No: I-2378-407.6

### References

[AE66] Ahlfors L. V., Earle C. J.: Lectures on quasiconformal mappings. 4

[BCGB08] Ben-Chen M., Gotsman C., Bunin G.: Conformal flattening by curvature prescription and metric scaling. In *Computer Graphics Forum* (2008), vol. 27, Wiley Online Library, pp. 449–458. 7

[Bel15] Bell S. R.: *The Cauchy transform, potential theory and conformal mapping*. CRC press, 2015. 2, 3

[CW15] Chen R., Weber O.: Bounded distortion harmonic mappings in the plane. *ACM Transactions on Graphics (TOG) 34*, 4 (2015), 73. 2

[DT02] Driscoll T. A., Trefethen L. N.: *Schwarz-Christoffel Mapping*, vol. 8. Cambridge University Press, 2002. 2

[DV98] Driscoll T. A., Vavasis S. A.: Numerical conformal mapping using cross-ratios and delaunay triangulation. *SIAM Journal on Scientific Computing 19*, 6 (1998), 1783–1803. 2

[ESA07] Eitz M., Sorkine O., Alexa M.: Sketch based image deformation. In *VMV* (2007), Citeseer, pp. 135–142. 2, 6, 7

[HLS07] Hormann K., Lévy B., Sheffer A.: Mesh parameterization: Theory and practice. 1

[HS96] He Z.-X., Schramm O.: On the convergence of circle packings to the riemann map. *Inventiones mathematicae 125*, 2 (1996), 285–305. 2

[Kra12] Krantz S. G.: *Handbook of complex variables*. Springer Science & Business Media, 2012. 5

[Kyt12] Kythe P.: *Computational conformal mapping*. Springer Science & Business Media, 2012. 2

[PLH04] Pottmann H., Leopoldseder S., Hofer M.: Registration without icp. *Computer Vision and Image Understanding 95*, 1 (2004), 54–71. 5

[Rud87] Rudin W.: *Real and complex analysis*. Tata McGraw-Hill Education, 1987. 1, 2

[SH15] Schneider T., Hormann K.: Smooth bijective maps between arbitrary planar polygons. *Computer Aided Geometric Design 35* (2015), 243–254. 2, 6, 7

[Ste99] Stephenson K.: The approximation of conformal structures via circle packing. *SERIES IN APPROXIMATIONS AND DECOMPOSITIONS 11* (1999), 551–582. 2

[VMW15] Vaxman A., Müller C., Weber O.: Conformal mesh deformations with möbius transformations. *ACM Transactions on Graphics (TOG) 34*, 4 (2015), 55. 2

[WBCG09] Weber O., Ben-Chen M., Gotsman C.: Complex barycentric coordinates with applications to planar shape deformation. In *Computer Graphics Forum* (2009), vol. 28, Wiley Online Library, pp. 587–597. 2, 3, 6, 7

[Weg86] Wegmann R.: An iterative method for conformal mapping. *Journal of computational and applied mathematics 14*, 1 (1986), 7–18. 2

[WG10] Weber O., Gotsman C.: Controllable conformal maps for shape deformation and interpolation. In *ACM Transactions on Graphics (TOG)* (2010), vol. 29, ACM, p. 78. 1, 2, 6, 7

**Appendix A:** Cauchy-Green coordinates formulas.

Given a polygon with $n$ vertices $\{z_j\}$ and a point $z$ inside the polygon, we use the notations $A_j = z_j - z_{j-1}, B_j = z_j - z$ for expressing the Cauchy-Green coordinates as:

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

The expressions for the first and second derivatives of the coordinates are given by:

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

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

When the point $z$ lies on an edge of the polygon, the logarithm in some of the coordinates has to be calculated for a real negative number which lies on the branch of the logarithm. In this case the limit for the logarithm $\log\frac{B_{j+1}}{B_j}$ when the point $z$ lies on the edge $e_j = \{z_j, z_{j+1}\}$ is obtained by $\lim_{z\to e_j}\log\frac{B_{j+1}}{B_j} = \log\left|\frac{B_{j+1}}{B_j}\right| + i\pi$, and the limits of the coordinates are calculated using this limit.

For a point $z$ on a vertex of the polygon the coordinates are calculated by:

$$C_j(z) = \frac{1}{2\pi i}\begin{cases} \frac{B_{j+1}}{A_{j+1}}\log\frac{B_{j+1}}{B_j} & z = z_{j-1} \\ \log\frac{B_{j+1}}{B_{j-1}} & z = z_j \\ -\frac{B_{j-1}}{A_j}\log\frac{B_j}{B_{j-1}} & z = z_{j+1} \end{cases}$$

The derivative of the coordinates is not defined for the vertices of the polygon. However, it can be approximated by calculating the derivative of the coordinates for a point close to the vertex, located inside the polygon.