Bounded Distortion Harmonic Shape Interpolation

Edward Chien
Bar Ilan University, Israel

Renjie Chen*
Max Planck Institute for Informatics, Germany

Ofir Weber
Bar Ilan University, Israel

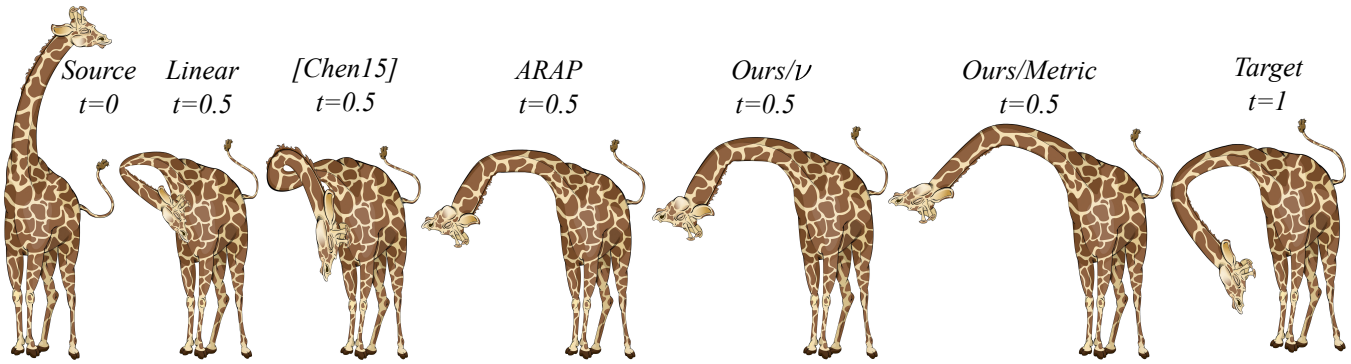

Figure 1: *Interpolation of the Giraffe.*

Abstract

Planar shape interpolation is a classic problem in computer graphics. We present a novel shape interpolation method that blends C^∞ planar harmonic mappings represented in closed-form. The intermediate mappings in the blending are guaranteed to be locally injective C^∞ harmonic mappings, with conformal and isometric distortion bounded by that of the input mappings. The key to the success of our method is the fact that the blended differentials of our interpolated mapping have a simple closed-form expression, so they can be evaluated with unprecedented efficiency and accuracy. Moreover, in contrast to previous approaches, these differentials are integrable, and result in an actual mapping without further modification. Our algorithm is embarrassingly parallel and is orders of magnitude faster than state-of-the-art methods due to its simplicity, yet it still produces mappings that are superior to those of existing techniques due to its guaranteed bounds on geometric distortion.

Keywords: shape interpolation, animation, injective mappings, harmonic mappings, bounded distortion, shape deformation

Concepts: •**Computing methodologies** → *Computer graphics; Animation; Image manipulation; Shape analysis;*

1 Introduction

Traditional hand-drawn animation is a (rather tedious) technique where each image frame is drawn by hand. It was the dominant form of animation in cinema until the dawn of computer animation which allowed for partial automation of the process. Computer animation is a computer-aided process where an animator sets the tone for the behavior of the animation in the form of keyframes, while the computer automatically generates intermediate frames that interpolate these keyframes. A good interpolation algorithm allows the artist to increase the time between the keyframes by producing natural and well-behaved intermediate frames that match the artist's expectations, hence reducing the amount of manual labor.

High-quality interpolation of rigid motions is fairly well understood by the graphics community. Typically, translations are interpolated based on a smooth space curve (e.g. a spline), and rotations are interpolated in angle space (2D) or using quaternions (3D). However, deformation-and-animation of soft bodies is significantly more challenging. Hence, animating and interpolating shapes that undergo large and complex deformation is an active research field in computer graphics and geometry processing.

A modern computer animation system (e.g. character animation) is composed of two major components. Given a source shape (e.g. an image of a character), the first system component allows posing the character and inserting a keyframe at a specific point in time. This is done by deforming the shape in a way that satisfies some constraints while preserving the geometric details and local structure of the original shape. See Figure 1 for an example. The head of the giraffe (left image) is pulled down while the legs stay put (right most image). The second component of the system is fully automatic and is in charge of blending the keyframe shapes. There is a large amount of active research that targets the first component of the system, i.e., designing efficient algorithms for the computation of large and complex deformations, with an ongoing focus in recent years on methods that provide strict guarantees on the geometric qualities of the deformed shapes. These mostly include C^0, mesh-based approaches [Lipman 2012; Weber et al. 2012; Schüller et al. 2013; Aigerman and Lipman 2013; Weber and Zorin 2014; Levi and Zorin 2014; Kovalsky et al. 2014; Kovalsky et al. 2015] but smooth meshless methods are also available [Weber and Gotsman 2010; Poranne and Lipman 2014; Chen and Weber 2015]. These methods aim to produce mappings that do not have inverted elements (flips) and that have bounded amount of conformal and/or isometric distortion. Producing such mappings is challenging since the underlying mathematical optimization is nonlinear and nonconvex. Shape interpolation techniques are also abundant, though methods with strict guarantees on quality are rare.

*Corresponding author

SIGGRAPH '16 Technical Paper, July 24-28, 2016, Anaheim, CA.
ISBN: 978-1-4503-4279-7/16/07
DOI: http://dx.doi.org/10.1145/2897824.2925926

An exception is the technique of Chen et al. [2013] that interpolates flat triangle meshes in a way that produces mappings whose conformal distortion is pointwise bounded by the distortion of the input.

We present a planar shape interpolation method that is designed to produce C^∞ harmonic mappings, which are the most widely used type of mappings in graphics applications due to their smoothness and appealing qualities. The input to our algorithm is two C^∞ harmonic mappings that are "provably good" [Poranne and Lipman 2014] in the sense that they are smooth, locally injective and have bounded conformal and isometric distortion. Such input mappings can be generated using a recent method by Chen and Weber [2015], though our method is applicable to any harmonic input regardless of how it was generated. Our algorithm blends the input mappings and produces a "provably good" harmonic mapping such that its geometric distortion is bounded by that of the input mappings. To the best of our knowledge, this is the first interpolation algorithm that provides bounds on both conformal and isometric distortion.

The most simple widely used interpolation approach is to linearly blend the input mappings. Efficiency is paramount since mathematical optimization is not employed. Moreover, such a process is trivial to parallelize and is typically implemented in the vertex shader since the blending of each point is independent of other points. This comes at a cost, as the quality of such interpolation tends to be quite poor (see the shrinkage of the neck in Figure 1).

Our method has three variants. The first two variants (Section 5) have the remarkable property (on top of its qualitative properties) that their computational complexity is linear in the number of points to be deformed. Just like the simple linear interpolation technique, there is no need to solve an optimization and the process is embarrassingly parallel, allowing for an efficient implementation on graphics hardware. The third variant (Section 6) offers some qualitative benefits over the first two under extreme deformations and is just marginally more involved computationally as it requires solving a small dense linear system (which is done once in a preprocess step for any given source domain). Our method is two orders of magnitude faster than the state-of-the-art [Chen et al. 2013] which requires solving a nonlinear optimization for each animation frame, yet produces similar overall behavior with increased smoothness (our method is meshless) and additional theoretical guarantees on the isometric distortion bounds.

2 Previous Work

Due to the abundance of literature on shape interpolation and deformation techniques, we will restrict ourselves to the most relevant approaches and concentrate on planar methods. We refer the reader to comprehensive reviews of classic methods that are provided in [Wolberg 1998; Alexa 2002]. Some shape interpolation methods consider two (or more) shapes, denoted as source and target, as input. Other methods take a more general point of view where a domain Ω and mappings $f^0, f^1 : \Omega \to \mathbb{R}^2$ are given. We mostly use the latter approach though sometimes it will be convenient to refer to the former one by treating the source shape as our domain, f^0 as the identity mapping and f^1 as the source-to-target mapping. Most shape interpolation methods are based on three main steps: (1) describing the input mappings in terms of some local differential quantities, (2) blending these quantities in a way that interpolates the input, and (3) recovering a new mapping from the blended quantities.

The As-Rigid-As-Possible (ARAP) approach [Alexa et al. 2000] decomposes the 2×2 piecewise constant Jacobians of the input mappings using the polar decomposition: $J_f = R \cdot S$, where R is a rotation matrix and S is a symmetric positive-definite matrix. For each triangle, a rotation angle θ is extracted from R and then S and θ are blended linearly. Such blending in angle space, rather than blending rotation matrices, typically leads to better control over the local distortion. However, in practice, the distortion tends to be significantly higher, as the Jacobians of the individual triangles are blended independently and are, in general, not integrable. In other words, a planar mapping with these Jacobians does not exist and a least squares reconstruction is performed with no control whatsoever on the *maximal* error introduced. A similar approach is taken by [Xu et al. 2006; Weber et al. 2007] where the reconstruction step is shown to be equivalent to a solution to the Poisson equation. Another difficulty arises when the angles are extracted from the rotation matrices due to $2\pi n$ ambiguities. Essentially, the (nonsmooth) principal branch of the matrix logarithm is computed and blending it can lead to nonsmooth and highly distorted results when large rotations are present. A partial remedy is obtained by using a special procedure for consistently choosing rotation angles [Choi and Szymczak 2003; Baxter et al. 2008].

An alternative approach to handling the rotation ambiguity is to use differential coordinates [Kircher and Garland 2008]. Rather than simply using the Jacobians, their ratio (which is invariant to rotations) is interpolated. The reconstruction process first solves for the Jacobians based on the differential coordinates, and then uses a Poisson system to recover the mapping (the spatial coordinates). None of the aforementioned methods is guaranteed to produce mappings that are globally or locally injective.

The methods by Surazhsky and Gotsman [2001; 2003] were specifically designed to produce bijective mappings. The barycentric coordinates of the vertices are blended linearly and then a globally bijective mapping is computed by embedding the intermediate shape inside a convex boundary by using a variant of the celebrated Tutte's embedding theorem [Tutte 1963]. The method is simple (a linear system is solved) and robust but the usability is limited due to the convexity requirement. Efforts to overcome this limitation by embedding a non-convex shape in an artificial convex support structure typically lead to distorted results. Moreover, global bijectivity is too restricting and typically local injectivity is desired.

The state-of-the-art is [Chen et al. 2013] which provides guarantees on local injectivity and a pointwise bound on conformal distortion. That is, the conformal distortion of the mapping at each triangle is not larger than the distortion of the corresponding triangle in the input. The method bypasses both the rotation ambiguity and the integrability problem simultaneously. Rather than blending the Jacobian J_f, it linearly blends the pullback metric $M_f = J_f^T J_f$ which factors out the rotational part of J_f. Chen et al. proved that the blended metric has pointwise bounded conformal distortion. Like the common issue of non-integrable Jacobians, the blended metric cannot be realized directly since its Gaussian curvature is usually not zero. Previous methods addressed the non-integrability of the Jacobian by solving the Poisson equation, which typically reverts the distortion bounds obtained by the blending process. In contrast, Chen et al. apply the innovative step of conformally flattening the metric to embed it in the plane. A conformal scaling of the metric ensures that the conformal distortion does not change, though additional isotropic scaling is introduced in the process. In Section 6 we prove that blending the pullback metric linearly also leads to bounded isometric distortion and show how to construct a planar harmonic mapping with both conformal and isometric distortion bounds. Our algorithm is significantly faster than the nonlinear optimization used in [Chen et al. 2013] as it only requires solving a linear system with a constant left-hand side.

All the methods we mentioned so far are mesh-based and produce continuous piecewise-affine mappings which are not smooth. Visually smoother results can be potentially obtained by refining the underlying meshes at the cost of longer computation time but conver-

gence properties are typically unknown and in most cases a smooth analogue of the algorithm does not even exist. To the best of our knowledge the only method that produces C^∞ mappings with strict quality guarantees is [Weber and Gotsman 2010] which interpolates smooth conformal mappings by blending the so-called angular factor on the shape's boundary, such that the intermediate mappings are also conformal. The first two variants of our method (Section 5) can be seen as generalizations of this approach to the broader class of harmonic mappings.

3 Mathematical Background

For completeness and to set notation, we include a short introduction to basic concepts in complex analysis and planar mappings. For further reading we refer to the books by Ahlfors [1979] and Duren [2004].

3.1 The Complex Derivatives

Consider a planar mapping given by a C^1 function $f : \Omega \to \mathbb{R}^2$ where $\Omega \subseteq \mathbb{R}^2$ is a domain (open, connected set). It is simple to verify that the Jacobian J_f at any point (or any 2×2 real matrix) may be written uniquely as the sum of a similarity and anti-similarity matrix. These sets of matrices are denoted below with $\mathcal{S}_2$ and $\mathcal{A}_2$, respectively:

$$\mathcal{S}_2 = \left\{ \begin{pmatrix} a & -b \\ b & a \end{pmatrix} \middle| \, (a,b) \in \mathbb{R}^2 \right\},$$

$$\mathcal{A}_2 = \left\{ \begin{pmatrix} c & d \\ d & -c \end{pmatrix} \middle| \, (c,d) \in \mathbb{R}^2 \right\}.$$

The matrices in $\mathcal{S}_2$ apply a similarity transformation by rotating and scaling $\mathbb{R}^2$, while the matrices in $\mathcal{A}_2$ perform such transformations after a reflection through the x-axis is applied.

It is also easy to verify that for a given 2×2 real matrix, the parts of the decomposition are the closest similarity and anti-similarity matrices in the sense of the Frobenius norm. In addition, the closest rotation (similarity transformation with determinant 1) is given by normalizing the columns of the similarity part. For the remainder of the paper, this unique decomposition of 2×2 real matrices will be referred to as the *additive decomposition*.

When we identify $\mathbb{R}^2$ with $\mathbb{C}$ using complex notation $z = x + \mathrm{i}y$, we see that the linear transformation applied by a similarity matrix is equivalent to complex multiplication of z by $a + \mathrm{i}b$. Analogously, the linear transformation applied by an anti-similarity matrix is equivalent to complex multiplication of $\bar{z}$ by $c + \mathrm{i}d$. For the remainder of the paper, $\mathbb{R}^2$ and $\mathbb{C}$ will be used interchangeably, with choices being made to assist the presentation.

This leads us naturally to the definitions of the Wirtinger derivatives: $f_z := \frac{1}{2}(f_x - \mathrm{i}f_y)$ and $f_{\bar{z}} := \frac{1}{2}(f_x + \mathrm{i}f_y)$. When the additive decomposition of J_f is considered, these formulae give us the complex numbers corresponding to the similarity and anti-similarity parts, with $f_z = a + \mathrm{i}b$ and $f_{\bar{z}} = c + \mathrm{i}d$.

3.2 Holomorphic (and Anti-holomorphic) Mappings

The set of planar mappings for which $f_{\bar{z}} = 0$ (everywhere) are called holomorphic functions, and are the central objects of study in complex analysis. More commonly, we see this condition written in a different fashion, with u and v denoting the components of $f = (u(x,y), v(x,y))$

$$\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y}, \quad \frac{\partial u}{\partial y} = -\frac{\partial v}{\partial x}.$$

These are called the Cauchy-Riemann equations. The additive decomposition shows us that for holomorphic mappings, J_f is a similarity transformation everywhere, equivalent to multiplication of $\mathbb{C}$ by a complex number f_z. This complex number is the complex derivative of a holomorphic function, and is often denoted more simply with $f' := f_z$.

Let us note that such functions are actually differentiable and integrable (on a simply connected domain) an infinite number of times. These derivatives and anti-derivatives are all holomorphic as well. In addition, sums, products and compositions of holomorphic functions are also holomorphic, and the quotient of two holomorphic functions is holomorphic wherever the denominator does not vanish. For proofs and details, we refer the reader to [Ahlfors 1979].

Planar mappings for which $f_z = 0$ everywhere are called anti-holomorphic functions. Each one is merely the complex conjugate of a holomorphic function, so they are closed under the same arithmetic operations, share the same differentiability and integrability criteria, and satisfy a "flipped" version of the Cauchy-Riemann equations with a negative sign added to one side of each equation.

3.3 Harmonic Planar Mappings

We now introduce harmonic planar mappings. If we have a real-valued function $u(x,y) : \Omega \to \mathbb{R}$, then it is harmonic if it satisfies the Laplace equation

$$\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0.$$

The value of a harmonic function is intuitively the average of nearby values (for more see the Mean Value Property [Duren 2004]). A planar harmonic mapping is a mapping $f : \Omega \to \mathbb{R}^2$ where $f = (u, v)$ and the component functions u and v are harmonic.

With the Cauchy-Riemann equations (and its "flipped" version), it is easy to see that holomorphic and anti-holomorphic functions are harmonic planar mappings, with their real and imaginary parts being harmonic. The converse is not generally true however, with many harmonic planar mappings being neither holomorphic, nor anti-holomorphic.

Instead we have that on a simply-connected domain Ω, any harmonic planar mapping f may be written as the sum of a holomorphic and an anti-holomorphic function.

$$f(z) = \Phi(z) + \overline{\Psi}(z), \tag{1}$$

where Φ and Ψ are holomorphic functions. Intuitively, the parts of the additive decomposition of J_f may be integrated separately to obtain Φ and $\overline{\Psi}$, with harmonicity of f ensuring integrability (see [Duren 2004] Section 1.2 for details). The above representation is unique up to an additive complex constant that can be chosen by setting e.g. $\Psi(z_0) = 0$ for an arbitrary point $z_0 \in \Omega$.

Note that for such a mapping, $f_z = \Phi'$ is holomorphic and $f_{\bar{z}} = \overline{\Psi'}$ is anti-holomorphic. The converse also holds: a planar mapping is harmonic if it has holomorphic f_z and anti-holomorphic $f_{\bar{z}}$.

3.4 Local Geometric Quantities

We return to considering C^1 planar mappings and note that many important quantities are easily expressed in terms of the Wirtinger derivatives. First, we have that:

$$\det(J_f) = |f_z|^2 - |f_{\bar{z}}|^2 .$$

A mapping f is locally injective and sense-preserving (preserves the orientation) at a point z if $\det(J_f(z)) > 0$, so we get that f is a locally injective and sense-preserving mapping wherever

$$|f_z| > |f_{\bar{z}}|\,.$$

As a result, we restrict our attention to mappings that satisfy this inequality everywhere. It is easy to see that we then also have $|f_z| > 0$ (also written $f_z \neq 0$). For the special case of a holomorphic function g (so $g_{\bar{z}} = 0$) satisfying this condition throughout its domain, we have that $g' \neq 0$ everywhere and it is called a *conformal* mapping. Such a mapping preserves the angle between any two intersecting curves under application of the mapping.

Our aim in this paper is to control the amount of angle and metric distortion induced by a mapping. Most distortion measures can be formulated in terms of the smallest and largest singular values of J_f: $0 \leq \sigma_b \leq \sigma_a$. These are the lengths of the minor and major axes of $J_f(\mathbb{S}^1)$ (see Figure 2) and are given by the following equations:

$$\sigma_a = |f_z| + |f_{\bar{z}}|\,, \quad \sigma_b = \Big|\,|f_z| - |f_{\bar{z}}|\,\Big|.$$

For the mappings we are considering, the latter expression simplifies to $\sigma_b = |f_z| - |f_{\bar{z}}| > 0$.

Another relevant quantity is $\mu = \frac{f_{\bar{z}}}{f_z}$, the *first complex dilatation* (a.k.a. the complex Beltrami coefficient). Its modulus $k = \frac{|f_{\bar{z}}|}{|f_z|}$ is called the *little dilatation* and is often used to measure conformal (angle) distortion. For the mappings under consideration, note that $0 \leq k(z) < 1$ throughout the domain and $k(z) = 0$ iff f is conformal.

The Beltrami coefficient μ also encodes other important geometric information. To interpret it appropriately, consider the singular value decomposition (SVD) of the Jacobian: $J_f = U\Sigma V^T$ and recall that the singular values are the diagonal entries of the diagonal matrix Σ and U, V are orthogonal matrices. We use v_a and v_b to denote the columns of V, which are orthonormal vectors (lying on the unit circle) that map to the major and minor axes of the ellipse $J_f(\mathbb{S}^1)$.

If $\mu \neq 0$, it is a fact that $\frac{1}{2}\mathrm{Arg}\,\mu$ (where $\mathrm{Arg}\,\mu \in (-\pi, \pi]$ is the principal value of the argument) is the angle of v_a or $-v_a$ with respect to the x-axis. We will refer to this value as the *stretch direction*. If $\mu = 0$, the mapping is conformal, $J_f(\mathbb{S}^1)$ is a circle, and there is no preferred stretch direction. The diagram in Figure 2 illustrates the case of $\mu \neq 0$, and clarifies the discussion above. Note that the stretch direction for non-zero μ only takes values in

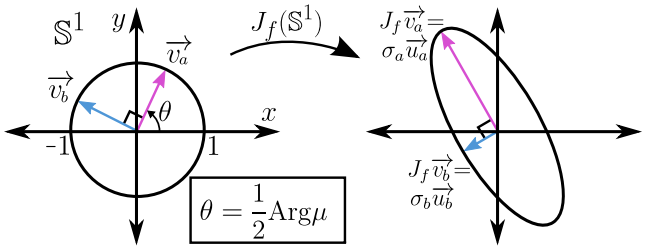

Figure 2: *The stretch direction illustration.*

$(-\frac{\pi}{2}, \frac{\pi}{2}]$, as maximal stretch really occurs along an entire axis.

Lastly, we note that the ratio of the singular values $K = \frac{\sigma_a}{\sigma_b} \in [1, \infty)$ (sometimes called the *large dilatation*) is also commonly taken as a measure of conformal (angle) distortion. The equation $k = \frac{K-1}{K+1}$ relates the two dilatations monotonically, and we see that bounding one bounds the other.

3.5 Logarithms, Arguments, and the Hilbert Transform

We finish the background section with a quick review and introduction of the complex logarithm, arguments, and the Hilbert transform. Lastly, we see how they interact to give us a necessary and important result.

For a complex number z, we consider solutions x to the equation $e^x = z$. If $z = 0$, there is no solution, and if $z \neq 0$, then there are a countable number of solutions of the form: $\{\ln|z| + \mathrm{i}(\mathrm{Arg}(z) + 2\pi n) \mid n \in \mathbb{Z}\}$.

A complex logarithm function on a domain $\Omega \subset \mathbb{C} - \{0\}$ is a continuous choice of one of these solutions for each $z \in \Omega$. Such a choice will be holomorphic [Ahlfors 1979]. Note also that the imaginary part of any such function will give a continuous notion of the "angle" between z as a vector and the positive x-axis. A continuous choice of such an angle is a harmonic function and is referred to as an argument function. When it is possible to choose a complex logarithm it is possible to choose an argument, and vice versa.

We may always make a continuous choice of both if Ω is simply-connected, but it is not guaranteed otherwise. For example, it is not possible to make such a choice for $\Omega = \mathbb{C} - \{0\}$. The standard principal branch, which we denote Log, has imaginary part Arg, and fails to be continuous (and holomorphic) on the negative x-axis. However, it is a complex logarithm function on any domain that avoids this set (and 0). Other typical branches arise by allowing continuity to fail along different rays or curves from 0 to ∞. Lastly, note that sometimes we will use ln to denote Log when the input to the function is in $\mathbb{R}^+$, as a way of emphasizing this.

Consider a non-zero function $g : \Omega \to \mathbb{C}-\{0\}$ defined on a simply-connected domain Ω. In the course of this paper, we will need to define the logarithm of such a function. If $g(\Omega)$ is simply-connected, we may simply choose a logarithm function $\log : g(\Omega) \to \mathbb{C}$ such that $e^{\log(g(z))} = g(z)$. If not, we may not be able to, so we define a single function $\log g : \Omega \to \mathbb{C}$ for which $e^{(\log g)(z)} = g(z)$. This is done by choosing a suitable initial value $(\log g)(z_0)$ for some point $z_0 \in \Omega$ and then integrating:

$$\log g(z) := (\log g)(z_0) + \int_{z_0}^{z} \frac{g'(w)}{g(w)}\,\mathrm{d}w. \tag{2}$$

Cauchy's integral theorem asserts that the integration is independent of the path from z_0 to z as g'/g is holomorphic on Ω. Note that the same problems exist in defining an argument function for g, and taking the imaginary part of such a logarithm defined by integration gives us a single function $\arg g : \Omega \to \mathbb{R}$ which solves these issues.

We introduce very quickly the idea of harmonic conjugates and the Hilbert transform. Given a harmonic function u defined on a simply-connected domain Ω, we aim to find a holomorphic function g such that $u = \mathrm{Re}\,(g)$. To find a suitable imaginary part v, we look to the Cauchy-Riemann equations for v_x and v_y. By harmonicity of u, the vector field they form will be integrable, and will determine v up to an integration constant. The procedure is analogous if one starts with a harmonic function v and would like to find a holomorphic function g such that $v = \mathrm{Im}\,(g)$. These partner harmonic functions are referred to as *harmonic conjugates*, and the procedure of finding them is the Hilbert transform [Bell 1992].

Finally, let us consider a pertinent question: if the argument or magnitude of a non-vanishing holomorphic function g on a simply-

connected domain is specified, to what degree is g determined? First, suppose the argument is specified by a function v (harmonic, of course). Then g will have a holomorphic logarithm for which v is the imaginary part. Performing the (inverse) Hilbert transform we get a harmonic conjugate u (determined up to an integration constant C), and then we exponentiate to get $g = e^{u+iv}$. We see that g is determined up to global scaling by a positive constant e^C. A similar line of reasoning shows us that when the magnitude of g is specified by a function $|g|$ such that $u = \ln |g|$ is harmonic, then g is determined up to a global rotation e^{iC} (where C again denotes an integration constant).

4 The Interpolation Problem

With the necessary mathematical background, we may explicitly state the problem at hand. We note first that for simplicity of the exposition and brevity, the problem (and the variants of our method in later sections) are stated and described for an input of just two mappings. The generalizations to multiple mappings are natural and simple to produce.

Let $f^0, f^1 : \Omega \to \mathbb{R}^2$ denote two locally injective sense-preserving harmonic mappings. We would like an interpolating function $f : [0,1] \times \Omega \to \mathbb{R}^2$ that has at least the following properties (note the domain of f):

1. (interpolation) $f|_{\{0\}\times\Omega} = f^0$ and $f|_{\{1\}\times\Omega} = f^1$
2. (harmonicity) $f|_{\{t\}\times\Omega}$ harmonic $\forall t \in [0,1]$
3. (loc. inj.) $f|_{\{t\}\times\Omega}$ is loc. inj. sense-preserving $\forall t \in [0,1]$
4. (smoothness) $f|_{[0,1]\times\{z\}}$ is C^∞ for all $z \in \Omega$

The last two properties are desirable as the animator typically does not want the interpolated shape to have singular points or local overlaps, or to abruptly change the direction of motion over the course of the animation.

In light of these properties, the following shorthand will be used for the remainder of the article: $f^t := f|_{\{t\}\times\Omega}$. In general, for the quantities discussed in Section 3.4 (and others to be introduced later), the standalone character will represent the quantity for all $t \in [0,1]$, and the same shorthand superscript notation will be used for the quantity for a specific t. For example, we let $\mu : [0,1]\times\Omega \to \mathbb{C}$ be defined by $\mu(t,z) = \frac{f^t_{\bar{z}}(z)}{f^t_z(z)}$ and define $\mu^t := \mu|_{\{t\}\times\Omega}$. For the sake of brevity, such definitions will be used without reminder.

The properties above represent a bare minimum, and do not reflect the fact that we would also like to have the conformal and isometric distortion of f^t bounded by their respective quantities for f^0 and f^1. Ideally, we would like these quantities to be bounded pointwise:

5. (conf. distortion) $k^t \le \max(k^0, k^1)$ for all $t \in [0,1]$
6. (max scaling) $\sigma^t_a \le \max(\sigma^0_a, \sigma^1_a)$ for all $t \in [0,1]$
7. (min scaling) $\sigma^t_b \ge \min(\sigma^0_b, \sigma^1_b)$ for all $t \in [0,1]$

We note here that isometric distortion is often denoted with a single quantity such as $\sigma_a + 1/\sigma_b$ or $\max(\sigma_a, 1/\sigma_b)$. Having both Properties 6 and 7 implies that any of these quantities will be bounded and together form a stronger condition than a bound on any one of these other quantities.

Lastly, we note that we may not have these exact properties under all circumstances, so some loosening (say to global instead of pointwise bounds) may be necessary. Additionally, we will see that for intuitive behavior, additional properties will be desired. These will be introduced as the discussion of our methods progresses.

4.1 A Basic Approach

As we are interested in preserving local geometric quantities, we will look to interpolate the Jacobian J_f and then to integrate it to obtain an interpolation. Furthermore, we would like the interpolation to be harmonic. The decomposition given by Equation (1) suggests a general approach that will solve these problems simultaneously. In particular, we can interpolate the similarity and anti-similarity parts of J_f separately by interpolating $f_z = \Phi'$ and $f_{\bar{z}} = \overline{\Psi'}$, holomorphically and anti-holomorphically respectively. As they are kept holomorphic and anti-holomorphic, they are integrable, and result in holomorphic and anti-holomorphic parts Φ and $\overline{\Psi}$. Summing these mappings to obtain the final interpolated mappings, we see that they will be harmonic and Property 2 will hold.

As might be noted, integration of J_f (or its parts) will result in integration constants that will need to be set. At $t = 0$ and $t = 1$, these integration constants will be chosen so that interpolation is achieved, satisfying Property 1, and for intermediate times, they will be chosen by linear interpolation. As a result:

Lemma 1. *If f_z and $f_{\bar{z}}$ have Property 4, f will have Property 4.*

Lastly, by discussions in Section 3.4, we have:

Lemma 2. *If $|f_z| > |f_{\bar{z}}| \ge 0\ \forall (t,z) \in [0,1]\times\Omega$, f will have Property 3.*

5 The First Two Variants: Parallel Methods

In this section, we present our first two variants, which are almost entirely parallel. They both interpolate f^0_z and f^1_z logarithmically to get an explicit formula for f^t_z. This basically interpolates $\arg f^0_z$ and $\arg f^1_z$ linearly while keeping f^t_z holomorphic. Details are in Section 5.1.

An anti-holomorphic formula for $f^t_{\bar{z}}$ is then determined by linear interpolating geometrically relevant quantities that ensure different desired properties. For the first variant, we introduce the second complex dilatation ν, which ensures conformal distortion is bounded (Property 5) when linearly interpolated. A full explanation and validation is in Section 5.2.

This first variant sometimes leads to unintuitive behavior, so a new property of stretch direction interpolation is formulated and explained in Section 5.3. The following section (5.4) introduces η, which ensures satisfaction of this property when linearly interpolated. This is the basis of our second variant.

It turns out that this basic approach may need to be modified to ensure satisfaction of the distortion bounds (Properties 5-7) when f^0 and f^1 differ greatly from each other. In particular, η may need to be scaled down by a positive constant $\rho < 1$ (Section 5.5).

5.1 Logarithmic Interpolation of f_z

Recall that for a planar mapping g, the quantity $g_z / |g_z|$ represents the closest rotation transformation to J_g at that point (in the Frobenius sense). Thus, it makes sense to have a formula for f^t_z which linearly interpolates the angle of the closest rotation transformation by linearly interpolating arguments of f^0_z and f^1_z.

This is achieved precisely by linearly interpolating the logarithms of f^0_z and f^1_z, which we refer to as logarithmic interpolation. It is expressed succinctly in the following interpolation formula:

$$f^t_z = (f^0_z)^{1-t}(f^1_z)^t. \tag{3}$$

To determine the non-integer powers of f_z^0 and f_z^1, we need to clearly define logarithms $\log f_z^0$ and $\log f_z^1$ as described in Section 3.5. The particular choices made are detailed in the implementation section, but the manipulations below help develop the proper intuition and make it clear that these logarithms are linearly interpolated:

$$\begin{aligned} f_z^t &= e^{(1-t)\log f_z^0} e^{t \log f_z^1} \\ &= e^{(1-t)\log f_z^0 + t \log f_z^1} \\ &= \left|f_z^0\right|^{1-t} \left|f_z^1\right|^t e^{\mathrm{i}\left((1-t)\arg(f_z^0) + t\arg(f_z^1)\right)}. \end{aligned}$$

As can be seen in the polar part of the expression above, the argument is linearly interpolated. Note also the key fact that f_z^t remains non-zero for all $z \in \Omega$ and all t, which will be necessary if we aim to have Property 3.

Let us note also that f_z^t is essentially unique in being a holomorphic interpolation which linearly interpolates the argument. As shown at the end of Section 3.5, a non-zero holomorphic function on a simply connected domain with specified argument is determined up to scaling by a positive global constant. f_z^t is such a function, and also has to interpolate f_z^0 and f_z^1, so this scaling constant is fixed.

Before moving on, we make two final notes. First, we highlight again that the following variants in this section may be viewed as an extension of [Weber and Gotsman 2010]. Interpolation there was restricted to conformal mappings and was performed with linear interpolation of the argument of the derivative on the boundary. The second important note is that we cannot use a logarithmic interpolation approach for the anti-holomorphic part, as $f_{\bar{z}}^0$ and $f_{\bar{z}}^1$ typically vanish at points in Ω, so we cannot sensibly define a logarithm for them. In light of this, we look to interpolate different quantities that reflect the local geometric distortion, and use those to determine $f_{\bar{z}}^t$.

5.2 Bounding conformal distortion

If we are aiming to bound conformal distortion, then we may try to linearly interpolate μ^0 and μ^1 to determine μ, and then solve for $f_{\bar{z}}^t$ with the equation $f_{\bar{z}}^t = \mu^t f_z^t$. Such an interpolation will satisfy Property 5, as $k = |\mu|$ and the norm is a convex function on $\mathbb{C}$. Unfortunately, the result that we will get for $f_{\bar{z}}^t$ is likely not anti-holomorphic, so will generally not be integrable.

Instead, for a planar mapping g, consider the *second complex dilatation* $\nu := \frac{g_{\bar{z}}}{g_z}$. This quantity satisfies $|\nu| = |\mu| = k$ and will be holomorphic as long as g_z does not vanish. Thus, we linearly interpolate this quantity with the simple equation:

$$\nu^t = (1-t)\nu^0 + t\nu^1. \tag{4}$$

As ν^0 and ν^1 are holomorphic, ν^t is holomorphic for all t. Thus, we obtain:

$$f_{\bar{z}}^t = \overline{\nu^t f_z^t}, \tag{5}$$

and we see clearly that we have an anti-holomorphic result.

With formulae for f_z^t and $f_{\bar{z}}^t$, this defines our first variant, which we refer to as the ν variant. It satisfies many of our properties:

Theorem 3. *The ν variant has Properties 1-5, 7.*

Proof. Properties 1 and 2 hold by the discussion in Section 4.1 on our basic approach.

As $\left|\nu^0\right|, \left|\nu^1\right| < 1$, we have $\left|\nu^t\right| < 1$ for all t, which implies that $\left|f_{\bar{z}}^t\right| < \left|f_z^t\right|$. So by Lemma 2, Property 3 holds.

As Equations 3 and 5 are smooth with respect to time everywhere, Property 4 holds by Lemma 1.

Property 5 holds as $k^t = \left|\nu^t\right|$ and the norm is convex on $\mathbb{C}$.

Finally, let us argue that Property 7 holds. Note first the equality:

$$\begin{aligned} \sigma_b^t &= \left|f_z^t\right| - \left|f_{\bar{z}}^t\right| \\ &= (1 - \left|\nu^t\right|)\left|f_z^t\right| \\ &= (1 - \left|\nu^t\right|)e^{\ln\left|f_z^t\right|}. \end{aligned}$$

Now suppose that a particular bound $C > 0$ is obeyed for a fixed time t: $C \le (1 - \left|\nu^t\right|)e^{\ln\left|f_z^t\right|}$. After some rearrangement we have $Ce^{-\ln\left|f_z^t\right|} - 1 + \left|\nu^t\right| \le 0$. Allowing t to vary, we see that $\left|\nu^t\right|$ is a convex function, as is $Ce^{-\ln\left|f_z^t\right|}$, since the exponent varies linearly. Thus, if a bound C holds at the endpoints ($t = 0$ and $t = 1$), then it will hold in between, and we see that by allowing $C = \min(\sigma_b^0, \sigma_b^1)$ we have our desired result. □

In addition to satisfying nearly all our desired properties, the simplicity of the method is a great asset and ensures that this is computationally the fastest and easiest to implement of our methods.

With respect to Property 6, we do not have that it is guaranteed to hold in general, but it is quite challenging to find a counterexample. Nonetheless, we note that since σ_a^t can be expressed as $(1+k^t)\left|f_z^t\right|$ and since k^t and $\left|f_z^t\right|$ are pointwise bounded, their product is still bounded by the product of the individual bounds.

5.3 Interpolating Stretch Direction

Despite all of its strengths, this method sometimes produces unintuitive visual results if the keyframes are drastically different. The reason for this is that the stretch direction is not interpolated properly, as can be seen in Figure 3.

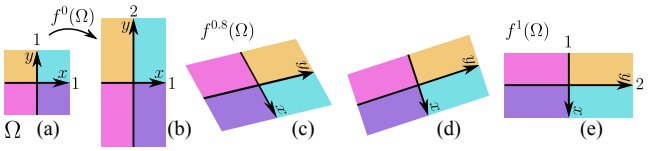

Figure 3: *Interpolation of the stretch direction. (a) is a square domain $[-1,1]^2$. (b) and (e) are the input mappings f^0, f^1, and each stretches the rectangle along the y-axis ($\mu^0 = \mu^1 = -\frac{1}{3}$), while f^1 additionally rotates the domain by $\pi/2$. (c) is the interpolation at $t = 0.8$. Note the stretch direction is slightly different than that of f^0, f^1. (d) is arguably the expected result where the stretch direction is maintained.*

As a result, we might ask that the following properties hold for all points $z \in \Omega$:

8. (interpolation of stretch direction)
   (a) If $\mu^0 = \mu^1 = 0$, then for all time t: $\mu^t = 0$
   (b) If $\mu^0 = 0$ ($\mu^1 = 0$), then $\forall t \in (0,1]$ ($t \in [0,1)$): $\mu^t = c\mu^1$ ($\mu^t = c\mu^0$) for some $c \in \mathbb{R}^+$.
   (c) If $\mu^0, \mu^1 \neq 0$, then for all time t: $\left|\mathrm{Arg}\left(\frac{\mu^t}{\mu^0}\right)\right| + \left|\mathrm{Arg}\left(\frac{\mu^1}{\mu^t}\right)\right| = \left|\mathrm{Arg}\left(\frac{\mu^1}{\mu^0}\right)\right|$.

When the mapping is conformal ($\mu^t = 0$), the stretch direction is ambiguous. The first two subproperties are specifically formulated to deal with these situations. The first states that if the mapping is conformal at the endpoints, then it should remain conformal throughout the interpolation. Note that this automatically holds as

a consequence of Property 5, but is included for clarity and completeness. The second states that if one mapping has a stretch direction, while the other is conformal, the intermediate mappings should have stretch direction matching the non-conformal mapping.

The third subproperty covers the most general case: when both endpoints have a well defined stretch direction. Here, we expect these directions to be interpolated along the shortest angular path between the stretch directions. The condition as written above expresses this succinctly due to the fact that $|\mathrm{Arg}(z)| \leq \pi$ for all $z \neq 0$, giving us the following property.

Arg Property. *Given $z_1, z_2 \in \mathbb{C} - \{0\}$, the angular distance between z_1 and z_2 is $|\mathrm{Arg}(z_1/z_2)| = |\mathrm{Arg}(z_2/z_1)|$.*

5.4 Introducing η

The desire for this property to hold leads us to a different quantity which is still anti-holomorphic, but shares the same argument as μ. For a planar mapping g, this quantity is $\eta = g_z\overline{g_z} = \mu\,|g_z|^2$. Note that up to scaling by a positive constant, this is the only anti-holomorphic function which shares the same argument as μ (its argument is specified everywhere but the shared zeroes of μ and η). The explicit formula for linear interpolation of this quantity is:

$$\eta^t = (1-t)\eta^0 + t\eta^1 \tag{6}$$

With this, we obtain an equation:

$$f_{\bar{z}}^t = \frac{\eta^t}{\overline{f_z^t}} \tag{7}$$

which is anti-holomorphic for all time t, and is thus integrable.

Given formulae for f_z^t and $f_{\bar{z}}^t$, this defines another variant, which we will refer to as the *unscaled η* variant. It will be generalized in the next section to give us the full method, but let us see that this simplified version has Property 8.

Proposition 4. *The unscaled η variant has Property 8.*

Proof. First, note that if we replace every instance of μ with η in Property 8, then this property would hold for the unscaled η variant. This is true as η is simply linearly interpolated (so $\eta/|\eta|$ automatically takes the shortest path in angle space).

Now, let us see that these properties are equivalent. By Equation 3, we have that $f_z^t \neq 0$ everywhere. So, we have that μ^t and η^t share the same zeroes as $f_{\bar{z}}^t$. Furthermore, we have $\eta^t = \left|f_z^t\right|^2 \mu^t$, so that their argument agrees everywhere. Thus, we have equivalence of the properties and our desired result. □

Unfortunately, without further modification, it fails to have bounded conformal distortion in all cases. To see this, consider the special case, where f^0 is the identity mapping. In this case, formulae are simplified, as $f_z^0 = 1$, $f_{\bar{z}}^0 = 0$, and $\eta^0 = 0$. Then we have:

$$k^t = \left|\frac{\eta^t}{(f_z^t)^2}\right| = \frac{t\,|\eta^1|}{|f_z^1|^{2t}}.$$

Thus, to have bounded conformal distortion, we need $t/\left|f_z^1\right|^{2t-2} \leq 1$ (for $t \in [0,1]$ obviously). Some simple analysis shows that this will occur only when $\left|f_z^1\right| \leq \sqrt{e}$. More generally, bounded conformal distortion is only preserved if the ratio $\left|f_z^0\right| / \left|f_z^1\right|$ is not too far from 1.

This variant may also fail to be locally injective, and in rare cases does not possess isometric distortion bounds. Fortunately, all of these issues may be remedied in a fashion which still preserves interpolation of stretch direction.

5.5 Scaling η

Local injectivity and the geometric distortion bounds may be recovered by scaling the linearly interpolated η. In particular, if these properties fail for the unscaled η variant, then we may define a scaled quantity $\tilde{\eta}^t := \rho(t)\eta^t$, where $\rho : [0,1] \to (0,1]$ is a scaling function such that $\rho(0) = \rho(1) = 1$. Then $f_{\bar{z}}^t$ is determined by the following equation:

$$f_{\bar{z}}^t = \frac{\tilde{\eta}^t}{\overline{f_z^t}}. \tag{8}$$

It is a fact that for a particular choice of ρ, we will have local injectivity and the geometric distortion bounds desired. For the conformal distortion bounds, one may see that if we allow $\rho \to 0$, then $f_{\bar{z}}^t \to 0$, and the mappings will become more and more conformal, lowering the conformal distortion.

It turns out that the isometric distortion bounds (and local injectivity) may also be recovered by appropriate scaling, but the argument is more technical. For better flow of the exposition, precise arguments have been relegated to Appendix A for the interested reader.

This procedure defines the scaled η variant, which generalizes the unscaled η variant by checking the geometric distortion bounds for each time t and scaling η suitably to preserve the bounds, if necessary. Equations (3) and (8) give us formulae for f_z^t and $f_{\bar{z}}^t$, which are integrated to obtain the interpolated mappings. Figure 4 compares the interpolated result on a bar model with and without the scaling of η.

Theorem 5. *The scaled η variant has Properties 1-3, 5-8.*
Proof. Properties 1 and 2 hold by the discussion in Section 4.1 on our basic approach.

Properties 3, 5-7 hold by the discussion in Appendix A.

Finally, Property 8 holds by Proposition 4 and the fact that scaling a complex number by a non-zero real constant does not change its argument. □

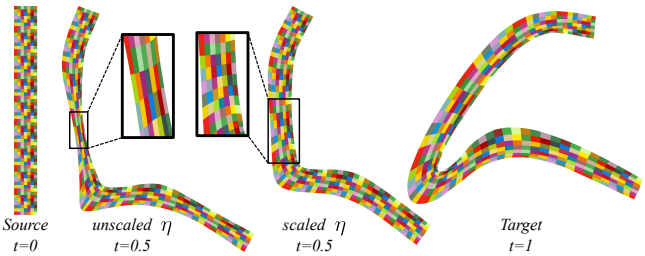

Figure 4: *Scaling η. Comparison of the results with and without scaling. The zoom-in shows the problematic area where the conformal distortion for the unscaled η variant is unbounded.*

The scaled η variant does not however guarantee smoothness with respect to time (Property 4), as no restrictions are made on the differentiability properties of ρ. It should be noted that with some small effort, procedures could be developed that would smooth any given ρ into various differentiability classes.

Finally we note that the global scaling by ρ sometimes leads to a qualitative non-locality to the interpolation. The intense scaling needed in one portion of a mapping may result in another portion being scaled to near-conformality, even when this is not desired. The decision to globally scale η was made in order to make sure that it stays anti-holomorphic.

6 Metric Pullback Method

For the third variant, an alternate procedure for interpolating f_z is considered, while maintaining linear interpolation of η. The basis of this method is linear interpolation of the metric tensor (first introduced in [Chen et al. 2013]). Section 6.1 defines the metric tensor precisely and shows that it preserves conformal and isometric distortion when linearly interpolated. Section 6.2 explicitly describes the procedure for performing this linear interpolation (along with linear interpolation of η) on the boundary to determine f_z^t.

Linear interpolation of η then determines $f_{\bar{z}}^t$ and Section 6.3 verifies that the procedure results in satisfaction of the geometric distortion bounds in a global sense in addition to all other previously defined properties. In addition to these guarantees, it seems to perform the best qualitatively of our three variants. Further comparison is done in Sections 8 and 9.

6.1 The Metric Tensor and Linear Interpolation

Consider a planar mapping $h : \Omega \to \mathbb{R}^2$. The metric tensor is the matrix expression for the pullback metric $h^* g_\bullet$ where $g_\bullet$ denotes the standard Euclidean metric given by the classical dot product. To simplify notation and to match more closely with previous works, we use M_h to denote this quantity. It is expressed simply in terms of the Jacobian: $M_h = J_h^T J_h$, and is clearly symmetric and positive-definite (when J_h is nonsingular).

With some basic linear algebra it is simple to see that given the metric tensor M_h at a point, there is an orientation-preserving Jacobian J_h which realizes it, and it is determined up to post-rotation. As one might note, there is no guarantee that a choice of such a Jacobian at every point will be integrable, regardless of the choice of rotation. As a result, a metric specified at every point may not be the pullback metric of a planar mapping h. This issue is dealt with later, so we continue forward with the following discussion holding at a differential level, at a single point.

To investigate the geometric properties of the Jacobians associated with a given metric tensor, we consider a more detailed formula for M_h. Using the additive decomposition of J_h into similarity and anti-similarity parts, we obtain a formula in terms of h_z, $h_{\bar{z}}$, and η:

$$\begin{pmatrix} |h_z|^2 + |h_{\bar{z}}|^2 & 0 \\ 0 & |h_z|^2 + |h_{\bar{z}}|^2 \end{pmatrix} + 2 \begin{pmatrix} \mathrm{Re}\,(\eta) & \mathrm{Im}\,(\eta) \\ \mathrm{Im}\,(\eta) & -\mathrm{Re}\,(\eta) \end{pmatrix} \quad (9)$$

In light of this expression, we define the useful quantity: $\mathcal{A} := |h_z|^2 + |h_{\bar{z}}|^2$. The isometric distortion constants are easily expressed in terms of this quantity and η:

$$\sigma_a^2 = |h_z|^2 + |h_{\bar{z}}|^2 + 2|\eta| = \mathcal{A} + 2|\eta| \quad (10)$$

$$\sigma_b^2 = |h_z|^2 + |h_{\bar{z}}|^2 - 2|\eta| = \mathcal{A} - 2|\eta| \quad (11)$$

A conformal distortion measure, the square of the large dilatation K^2, may also be easily expressed in terms of $\mathcal{A}$ and η:

$$K^2 = \frac{\sigma_a^2}{\sigma_b^2} = \frac{\mathcal{A} + 2|\eta|}{\mathcal{A} - 2|\eta|}. \quad (12)$$

Now, consider linear interpolation of the metric tensor. In particular, consider two planar mappings f^0 and f^1 and linearly blend their metrics M_f^0 and M_f^1 to get the formula $M_f^t = (1-t)M_f^0 + tM_f^1$. The uniqueness of the additive decomposition in Equation (9) implies that $\mathcal{A}^t = (1-t)\mathcal{A}^0 + t\mathcal{A}^1$ and $\eta^t = (1-t)\eta^0 + t\eta^1$. From Equations (10) and (11) it is clear that the isometric distortion quantities are bounded as $\mathcal{A}^t$ is linear with respect to time and $|\eta^t|$ is convex with respect to time. We will also see that K^2 is quasiconvex with respect to time, but the argument is a little more involved, so it is delayed until the proof of Theorem 6.

6.2 Interpolation on the Boundary

As we noted previously, the blending of the metric tensor everywhere may not give us metrics that are realizable as the pullback metric for a planar mapping. In [Chen et al. 2013], this was dealt with by using a discrete curvature flow to flatten the metric, which preserves the conformal distortion bounds, but unfortunately invalidates the isometric distortion bounds.

In the variant we are presenting here, we effectively only blend the metric linearly on the boundary of Ω, allowing us to preserve the geometric distortion bounds there. As our resulting mappings are harmonic, we obtain global bounds by the work in [Chen and Weber 2015].

This effective linear blending is accomplished by linearly blending $\mathcal{A}$ and η on $\partial\Omega$. This determines $|f_z^t|^2$ on the boundary as a root of the following quadratic:

$$|f_z^t|^2 + |f_{\bar{z}}^t|^2 = (1-t)\mathcal{A}^0 + t\mathcal{A}^1$$

$$|f_z^t|^2 + \frac{|\eta^t|^2}{|f_z^t|^2} = \mathcal{A}^t$$

$$\left(|f_z^t|^2\right)^2 - |f_z^t|^2 \mathcal{A}^t + |\eta^t|^2 = 0$$

This quadratic has two solutions, given by the equation below:

$$|f_z^t|^2 = \frac{\mathcal{A}^t \pm \sqrt{(\mathcal{A}^t)^2 - 4|\eta^t|^2}}{2} \quad (13)$$

It is a simple (but tedious) exercise to see that the discriminant is positive, so there are two distinct real solutions.

If one were to solve for $|f_{\bar{z}}^t|^2$ instead, the exact same quadratic would be obtained. As $\mathcal{A}^t$ is to be blended linearly, a choice of one solution for $|f_z^t|^2$ will mean that $|f_{\bar{z}}^t|^2$ will take the other solution. Thus, we take the positive root, so that $|f_z^t| > |f_{\bar{z}}^t|$.

With $|f_z^t|$ specified on the boundary, we then solve the Dirichlet problem to obtain a harmonic function $u : \Omega \to \mathbb{R}$ with value $\ln|f_z^t|$ on the boundary. Next, the process described in Subsection 3.5 is used to obtain a harmonic conjugate v, and the resulting $\log f_z^t$ is exponentiated to obtain a formula for f_z^t over all of Ω (and for all time t). There is of course, an integration constant which results in f_z^t only being determined up to a global rotation e^{iC}. The integration constant is chosen at the endpoints to ensure interpolation and is merely linearly interpolated for intermediate times.

Lastly, to obtain an anti-holomorphic formula for $f_{\bar{z}}^t$, we maintain linear blending of η everywhere (not just on the boundary). This results in the same formula as that given in Equation (7) (though f_z^t is different). We refer to this variant as the metric variant.

An application of this variant is illustrated in Figure 5, along with a comparison to our first two variants and ARAP. Qualitatively, it can be seen that the metric and scaled η variants outperform the ν variant and ARAP. The quantitative plots and comparison are discussed further in Section 8. Animations of some of these outputs are also available in the accompanying video.

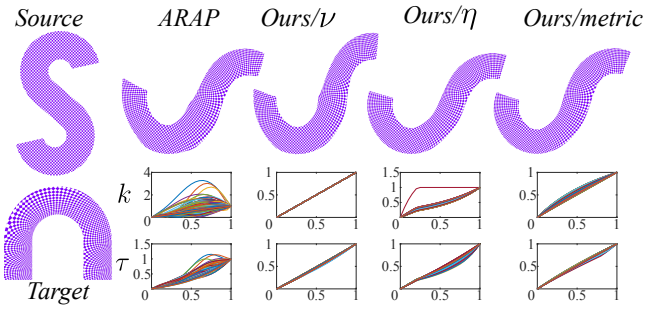

Figure 5: *S-2-U. Qualitative and geometric distortion comparison for our variants and ARAP.*

6.3 Variant Validation

As noted already, we use the results of Chen and Weber [2015] (Theorem 4) to give us global bounds on conformal distortion and isometric distortion. For a planar mapping g, we denote the global bound on conformal distortion $\mathring{k} = \sup_{z\in\overline{\Omega}} k(z) \in [0, 1)$. Similarly, we let $\mathring{\sigma_a} = \sup_{z\in\overline{\Omega}} \sigma_a(z) \in (0, \infty)$ and $\mathring{\sigma_b} = \inf_{z\in\overline{\Omega}} \sigma_b(z) \in (0, \infty)$ denote the global bounds on the isometric distortion. Note that in our applications, the mappings are always assumed to be defined on a region slightly larger than the compact figure to be animated, so the supremums and infinums will always be realized in $\overline{\Omega}$. Below are the modified properties on conformal and isometric distortion:

5'. (conf. distortion) $\mathring{k}^t \leq \max(\mathring{k}^0, \mathring{k}^1)$ for all $t \in [0, 1]$

6'. (max scaling) $\mathring{\sigma_a}^t \leq \max(\mathring{\sigma_a}^0, \mathring{\sigma_a}^1)$ for all $t \in [0, 1]$

7'. (min scaling) $\mathring{\sigma_b}^t \geq \min(\mathring{\sigma_b}^0, \mathring{\sigma_b}^1)$ for all $t \in [0, 1]$

This variant has all of our essential properties, it interpolates stretch direction and has the above modified distortion properties, and also.

Theorem 6. *The metric variant has Properties 1-4, 5'-7', 8.*

Proof. Properties 1 and 2 follow by the discussion in the basic approach section and the short note about the e^{iC} global rotation in Section 6.2.

Property 3 will follow from Property 7'. It ensures us that $\mathring{\sigma_b}^t > 0$ for all t so that $\sigma_b^t(z) > 0$ for all z and t, and we have the property by Lemma 2.

For Property 4, note first that Equation (13) for $\left|f_z^t\right|^2$ is smooth. This follows as $\left|\eta^t\right|^2$ is smooth and the discriminant is positive (see Lemma 7 in Appendix B). As $\mathcal{A}^t > 0$, we have that $\left|f_z^t\right|^2 > 0$, so $\left|f_z^t\right| = \sqrt{|f_z^t|^2}$ is smooth. Furthermore, we have then that $\ln\left|f_z^t\right|$ is a smoothly varying boundary input to the Dirichlet problem, whose solution in the interior will then vary smoothly. The Hilbert transform is also a smooth operation, and we obtain a smooth formula for f_z^t. Finally, from Equation 7, we see that $f_{\bar{z}}^t$ is also smooth, so we have the property by Lemma 1.

For Property 5', we use reasoning that is analogous to that used to prove Property 7 in Theorem 3. Note first that it is equivalent to bound K^2. We have the equality:

$$(K^t)^2 = \frac{(\sigma_a^t)^2}{(\sigma_b^t)^2} = \frac{\mathcal{A}^t + 2\left|\eta^t\right|}{\mathcal{A}^t - 2\left|\eta^t\right|}.$$

This is just Equation (12) with t superscripts. Now suppose that a particular bound $C \geq 1$ is obeyed for a fixed time t: $C \geq \frac{\mathcal{A}^t+2|\eta^t|}{\mathcal{A}^t-2|\eta^t|}$. After some rearrangement we have $(C-1)\mathcal{A}^t - (C+1)2\left|\eta^t\right| \geq 0$. Allowing t to vary, we see that $-(C+1)2\left|\eta^t\right|$ is a concave function, as is $(C-1)\mathcal{A}^t$ since it is linear. Thus, if a bound C holds at the endpoints ($t = 0$ and $t = 1$), then it will hold in between, and we see that by allowing $C = \max((K^0)^2, (K^1)^2)$ we have a pointwise bound on $\partial\Omega$. Theorem 4 of [Chen and Weber 2015] implies that if f_z does not vanish inside (our f_z is obtained by exponentiation) bounding the distortion on the boundary is sufficient to bound it everywhere, which gives us Property 5'.

For Properties 6' and 7', recall that we argued at the end of Section 6.1 for pointwise bounds on $\partial\Omega$. These followed as Equations (10) and (11) were shown to be convex and concave, respectively. [Chen and Weber 2015] is used again to show that the bounds are global.

Finally, we have that Property 8 holds as we are still linearly interpolating η everywhere. As η^t is the same as it would be in the unscaled η variant, Proposition 4 gives us this property. □

7 Implementation

Our algorithm is implemented as a plug-in to Autodesk Maya 2016. The GUI elements, such as keyframe insertion, are implemented through Maya's C++ API. The rendering is done in OpenGL and the algorithm itself is implemented in Matlab (through Matlab's COM engine). We used Matlab's built-in ability to parallelize vector operations using Nvidia's CUDA on the GPU. This is simply achieved by using the gpuArray function which copies the vector to the GPU memory. The rest is fully transparent to the programmer. Every embarrassingly parallel operation such as computing the logarithm of every entry in the vector or summing vectors is automatically processed on the GPU. In fact, the same Matlab code can run on the CPU in case a CUDA-enabled GPU is not available. The results were obtained on an i7-3770 core with Nvidia Quadro K6000 GPU.

Our method is formulated in the smooth case assuming that the input is a C^∞ harmonic mapping and as such, it is indifferent to the particular underlying representation of the input. However, the method of [Chen and Weber 2015] was used to generate the input mappings in most of the examples in this paper, because the method is guaranteed to produce bounded distortion harmonic mappings expressed in closed-form.

Chen and Weber assume that the domain is bounded by a simply connected polygon P which is offset in the outward normal direction to form the polygon $\hat{\mathrm{P}} = \{z_1, z_2, ..., z_n\},\ z_j \in \mathbb{C}$ which is denoted as the cage. The infinite-dimensional space of harmonic mappings is spanned by Equation (1) where the holomorphic functions are approximated by Φ and Ψ from the finite-dimensional space spanned by the Cauchy complex barycentric coordinates [Weber et al. 2009]:

$$\Phi(z) = \sum_{j=1}^{n} C_j(z)\varphi_j, \quad \Psi(z) = \sum_{j=1}^{n} C_j(z)\psi_j. \qquad (14)$$

Visualization of the mapping is achieved by applying a texture onto a high resolution triangulation $\mathcal{T}$ of the domain ($\approx$100,000 triangles were used for all the results presented in this paper). The holomorphic basis functions $C_j(z)$ possess a simple closed-form expression, hence f_z and $f_{\bar{z}}$ are given by a simple formula [Chen and Weber 2015]. The blended derivatives: f_z^t and $f_{\bar{z}}^t$ are holomorphic and anti-holomorphic functions respectively and are precisely integrable. However, their antiderivatives do not in general belong to the holomorphic subspace spanned by Equation (14). A similar difficulty arises in [Weber and Gotsman 2010] and was treated by least-squares projection of the derivatives into the Cauchy's coordinates subspace. While this is a viable approach that results in a

closed-form expression for the mapping f^t, it can lead to violation of the geometric distortion bounds.

Instead, $f^t(v_i)$ is evaluated at the vertices of $\mathcal{T}$ using numerical integration which can be done efficiently for any required accuracy. At preprocessing, we compute a spanning tree for the graph induced by $\mathcal{T}$ whose root vertex v_0 (anchor) is chosen by the user. For each directed edge $e_{i,j}$ of the spanning tree, we approximate the edge difference using a simple trapezoidal quadrature rule:

$$\Phi^t(v_j)-\Phi^t(v_i) = \int_{v_i}^{v_j} f_z^t(z)dz \approx \frac{(v_j - v_i)(f_z^t(v_j) + f_z^t(v_i))}{2}.$$

This turned out to be extremely accurate and higher order rules such as Simpson's rule did not lead to noticeable improvement. Furthermore, since the Lipschitz constants for the derivative of the Cauchy coordinates were derived by Chen and Weber, a bound on the maximal approximation error can be obtained. The evaluation is done on the GPU for all edges in parallel (as explained at the beginning of this section) and then these edge values are accumulated along the branches of the spanning tree starting from v_0. The value at v_0 determines the degree of freedom of the integration constant. The anchor position is interpolated linearly throughout the animation. i.e. $\Phi^t(v_0) = (1-t)f^0(v_0) + tf^1(v_0)$, but it should be noted that other choices (e.g., a user-defined trajectory curve) are possible. Similarly, $\overline{f_{\bar{z}}^t}$ is integrated to obtain Ψ by choosing $\Psi^t(v_0) = 0$.

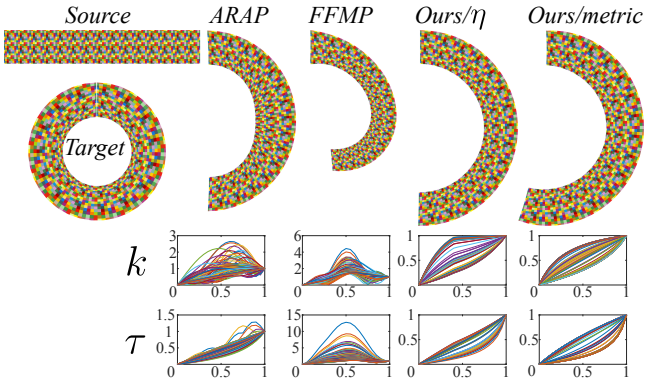

Figure 6: *Bar to ring. Qualitative and geometric distortion comparison.*

7.1 Log-based Methods

In this subsection, we explain how the actual blending of f_z^t and $f_{\bar{z}}^t$ is done. In order to accelerate the computation, we can separate quantities that are time-dependent from those that are not. For example, the variants in Section 5 require that $\nu^0(v_i), \nu^1(v_i), \eta^0(v_i), \eta^1(v_i), \log(f_z^0(v_i)), \log(f_z^1(v_i))$ will be evaluated at each vertex v_i of $\mathcal{T}$. These quantities depend only on the input mappings f^0, f^1 and can be evaluated and stored during the deformation phase. Alternatively, since the number of animation frames between each two consecutive keyframes is typically large (say 50), the computation time of these quantities can be amortized such that it become negligible. All these quantities are stored in vector form on the GPU memory (using gpuArray).

On the other hand, $\nu^t(v_i), \eta^t(v_i), f_z^t(v_i)$ given by Equations (4),(6),(3) respectively are time-dependent and are evaluated for every $t \in [0,1]$. Given a specific time $t \in [0,1]$, the blendings of ν^t, η^t, f_z^t are computed in vector form, followed by the computation of $f_{\bar{z}}$ (Equation (5) or (7) depending on the particular variant).

7.2 Evaluating the Log

As explained in Section 3.5, since f_z^0, f_z^1 are nonvanishing, a continuous logarithm branch exists. Matlab's built-in complex logarithm implementation is designed to compute the principal branch which is not appropriate in most cases. Numerical integration is a straightforward way to compute the logarithm (Equation (2)), however we can do better than that. For each edge $e_{i,j}$ of the spanning tree we compute the difference between the desired values of the log as follows:

$$\log(f_z(v_j)) - \log(f_z(v_i)) = \mathrm{Log}\left(\frac{f_z(v_j)}{f_z(v_i)}\right). \qquad (15)$$

Recalling the property of $\mathrm{Arg} = \mathrm{Im}(\mathrm{Log})$ noted at the end of Section 5.3, the above formula is *exact* as long as the difference between the arguments at the edge's endpoints is in the range $(-\pi/2, \pi/2)$. This should always be the case for any sensible input. If theoretically, this is not the case, the edge can always be subdivided. Finally, like before, edge values are accumulated along the tree branches to form the vertex values. The value at the root is determined by $\log(f_z^0(v_0)) = \mathrm{Log}(f_z^0(v_0))$ and $\log(f_z^1(v_0)) = \mathrm{Log}\left(\frac{f_z^1(v_0)}{f_z^0(v_0)}\right) + \log(f_z^0(v_0))$, again utilizing the property of Arg to ensure that the imaginary parts of these anchor values differ by no more than π.

7.3 Scaling η

For scaling η, it is clear that it is best to minimize the amount of scaling (or to maximize ρ) while still achieving the geometric distortion bounds. In order to achieve this, at each time frame t, the bounds are experimentally checked on a dense set of points. We pause here to note that in Appendix A, the algorithm was described as checking for violation of distortion bounds throughout Ω, but in practice, we only check on $\partial\Omega$, as it is sufficient to ensure that the distortion bounds hold everywhere, and is faster. With this modification, the algorithm technically only guarantees the global distortion bounds (Properties 5'-7'), but it is simple to modify the implementation to check throughout the interior of the domain as well.

For this section, the notation $k^t, \sigma_a^t, \sigma_b^t$ is used to denote the values obtained with the unscaled η variant. Looking at Equations (16), (17), and (18), we see that if the distortion measures are violated at a point z, then the smallest respective scalings (or rather largest scaling coefficients) needed to recover the bounds are:

$$\rho_k^t(z) = \frac{\max(k^0, k^1)(z)}{k^t(z)}$$
$$\rho_a^t(z) = \frac{\max(\sigma_a^0, \sigma_a^1)(z) - \left|f_z^t(z)\right|}{|f_{\bar{z}}^t(z)|}$$
$$\rho_b^t(z) = \frac{\left|f_z^t(z)\right| - \min(\sigma_b^0, \sigma_b^1)(z)}{|f_{\bar{z}}^t(z)|}$$

If any of the denominators in the above formulae are zero, the value is simply set to 1, as no scaling is needed then.

Within the implementation, the violation is checked and the scaling is computed simultaneously by the following formula.

$$\rho(t) = \min_{z\in\partial\Omega}\left(\min\left(\rho_k^t(z), \rho_a^t(z), \rho_b^t(z), 1\right)\right)$$

The scaling values are compared to each other and to 1 (in case no violation occurs), and then this is minimized over all boundary points.

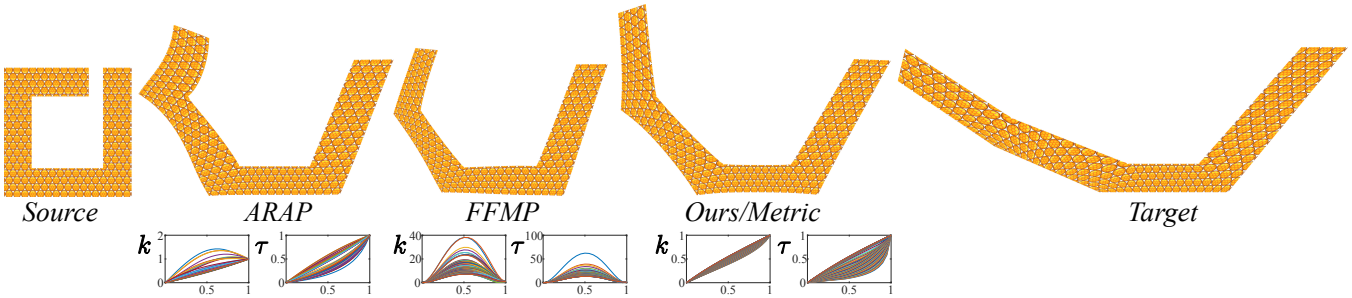

Figure 7: *Square bar, with input specified by harmonic coordinates. Qualitative and geometric distortion comparison.*

7.4 Metric-based Method

The implementation of the metric variant does not require the evaluation of the log (Section 7.2) but is slightly more involved than the log-based variants due to the need to compute the Hilbert transform. Computing η^t is done as explained in Section 7.1. Next we compute $\ln \left| f_z^t(w_i) \right|$ on the boundary, which is done by evaluating Equation (13) and then taking the square root and logarithm of the result. These operations are performed on a set of dense uniform sample points w_i lying on P. As usual, all the computations are done in vector-form on the GPU.

Our next goal is to find a harmonic function with $\ln \left| f_z^t \right|$ as boundary values and then to find a conjugate harmonic function (the Hilbert transform). This is done simultaneously using the Hilbert complex barycentric coordinates as explained in Section 9 of [Weber and Gotsman 2010]. The computation is based on the complex-valued boundary element method (BEM) and requires solving a rather small dense linear system. Computing the Hilbert coordinates is quite efficient and takes less than a second for all the models we present here. But more importantly, it is done once in preprocessing and is independent of the animation.

Obtaining $\log(f_z^t)$ at the vertices of $\mathcal{T}$ boils down to a matrix-vector multiplication. The matrix is stored on the GPU (using gpuArray) and since it is dense, the multiplication is also embarrassingly parallel. Finally, $\log(f_z^t)$ is exponentiated to obtain f_z^t. $f_{\bar{z}}^t$ is obtained by (7) and the actual mapping f^t is realized by the integration procedure described earlier.

8 Results

In this section, we present experimental results that demonstrate the relative speed and efficacy of our variants when compared with several existing methods. In addition, they allow us to contrast the three variants and talk about their relative strengths and weaknesses. We will also occasionally refer to the accompanying video, as the problem at hand is an interpolation problem, and the best way to appreciate the results is to compare the actual animations with respect to the time domain.

We begin by referring to Figure 1, which compares the output of several algorithms at time $t = 0.5$. The first shows the poor quality of the linear interpolation at $t = 0.5$, which simply linearly interpolates the positions of all the points. This naive approach clearly does not preserve geometric distortion, and rarely produces locally injective mappings.

The second output shows the algorithm of [Chen and Weber 2015] with the point-to-point (P2P) handles interpolated linearly. As can be seen, the behavior is unintuitive despite its bounds on geometric distortion. In order to obtain natural behavior, the trajectory of the handles would clearly need to be non-linear. We also note that it is not as robust as our variants, because the feasibility of the problem is not guaranteed if there are too many handle points. For further demonstration of this concept, we refer to the raptor and giraffe examples in the video.

The third and fourth outputs are the result of ARAP [Alexa et al. 2000] and our ν variant, and are quite similar qualitatively. We note here that for all examples of the ARAP method, we used the rotation angles obtained in Section 7.2 ($\theta = \mathrm{Im}\,(\log)$) rather than the principal branch (as the method suggests) as it performs poorly otherwise. If one focuses on the upper portion of the giraffe's neck for these outputs, it can be seen that the stretch direction at the halfway point is perpendicular to those of the source and target images (which run along the neck roughly). This contrasts with the final output which is the result of our metric variant, and maintains the stretch direction. Usually this variant produces the best qualitative behavior of all of our variants, as is shown here.

Next, let us extend the comparison of our three variants and ARAP presented in Figure 5 by discussing the plots of the geometric distortion for the interpolations within the figure. More specifically, the plots are of normalized quantities k for conformal distortion, and $\tau = \max(\sigma_a, 1/\sigma_b)$ for isometric distortion, for 100 points of high distortion. The quantities are normalized separately for each point, with a simple linear scaling so that these values are 1 at time 1. Like most of the examples here, the source is the domain and f^0 is the identity mapping, so the distortion values are 0 at time 0. More points were not plotted to prevent crowding of the plots.

We make two notes about these plots. The first (and most important) is that ARAP clearly fails to have bounded geometric distortion, as many distortion values exceed 1 for intermediate time values, and that all of our variants achieve bounded geometric distortion. The second is that for the plot of normalized k for the ν variant, we see that it is strictly linearly interpolated, as it should be when $\nu^0 = 0$.

Figure 6 is similar but replaces our ν variant with an output from the FFMP algorithm [Kircher and Garland 2008]. Qualitatively, our variants outperform both existing methods and we encourage the reader to view the video to see this even more clearly. With respect to the plots, we see again that ARAP fails to have bounded distortion, as does FFMP. The distortion plots for FFMP are not surprising, as the method is invariant to affine transformations, which can certainly affect both conformal and isometric distortion. Lastly, we note that if one looks closely at the distortion plots for the scaled η variant, one can see where the method fails to be smooth at around time $t = 0.4$. In general, this failure of smoothness is hard to notice visually, but its presence is clear in this example from the plots.

In Figure 9, we compare running times with [Chen et al. 2013], which produces high-quality results with bounded conformal distortion which are visually very similar to our metric-based variant.

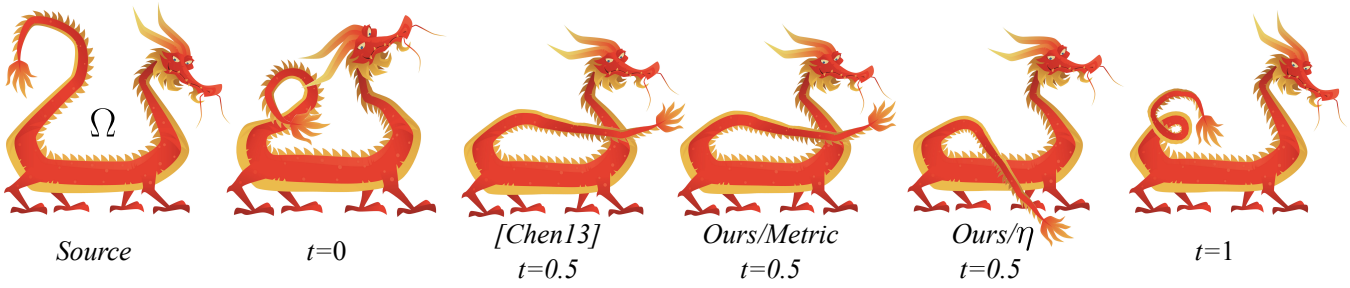

Figure 8: *Dragon, with moving tail. Ω in first image and $f^0 \neq \mathrm{Id}$ in second image. Qualitative comparison.*

We compare the runtime of a single frame of our metric-based variant with that of Chen et al's.

For fairness, we ran our method on the CPU as the Newton solver needed for the nonlinear optimization of [Chen et al. 2013] cannot run on a GPU. As can be seen from the graphs, our method is two order of magnitudes faster. For high resolution meshes our GPU implementation is, in addition, about 5 times faster than our CPU one. We also use this opportunity to note that our variants are faster than nearly all of the pre-existing methods: ARAP, FFMP, [Chen et al. 2013], and [Chen and Weber 2015] with linearly interpolated handles. Only the simplest and most naive method, linear interpolation, is faster, and our simplest variant, the ν variant has similar running times.

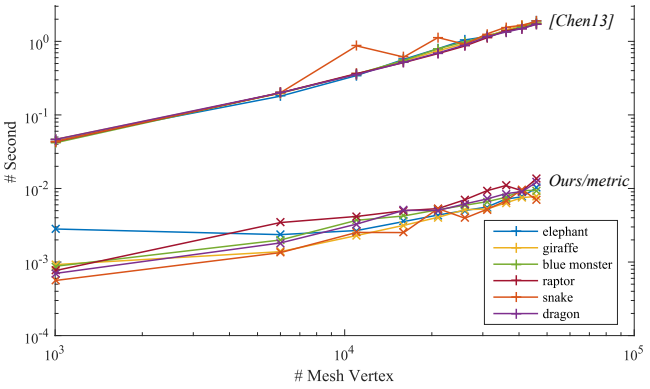

Figure 9: *Running times for a single animation frame on the CPU.*

The next two figures, Figures 7 and 8, show some examples that come from a broader context of application. For the rest of the examples in this section, the input mappings were generated with [Chen and Weber 2015], but the input for Figure 7 was generated with harmonic coordinates as described in Section 10 of [Weber and Gotsman 2010]. It also demonstrates qualitative superiority of the metric variant over ARAP and FFMP, as well as the fact that these pre-existing methods do not have bounded distortion.

As noted previously, we have also used the source as the domain and considered f^0 to be the identity for nearly all the examples. In Figure 8, we have a demonstration of the scaled η variant and the metric variant on two input mappings which both differ from the identity. They are also compared with [Chen et al. 2013]. This example is useful for highlighting the qualitative similarity of the metric variant with Chen et al's method, and for noting the difference between the metric variant and the scaled η variant. We see that the speed of rotation of the Dragon's tail differs in each, likely due to the fact that the scaled η variant linearly interpolates the angle of the closest rotation while the metric variant does not.

Finally, we have Figure 10, which has a large collection of results obtained by our metric-based variant, and serves as a further advertisement for the video. As noted previously, this variant seems to behave the best qualitatively.

9 Summary and Discussion

In this paper, we presented three variants of a novel shape interpolation technique for locally-injective harmonic mappings. The basic method relies on Equation (1) which allows us to blend the Jacobians in a fashion which keeps them integrable and keeps the mapping harmonic for all intermediate times. The ν and scaled η variants blend the similarity part f_z of the Jacobian logarithmically, and determine the anti-similarity part $f_{\bar{z}}$ by linearly interpolating the geometrically relevant quantities in their names. The metric variant determines f_z by effectively linearly blending the metric tensor on the boundary, while maintaining linear interpolation of η (which determines $f_{\bar{z}}$). The mathematical simplicity of this basic approach makes our variants simple to implement, largely parallel, and efficient.

All three of our variants manage to bound the geometric distortion with respect to the input mappings, to varying degrees. The ν variant achieves pointwise bounded conformal and σ_b distortion, and usually achieves a pointwise bound on σ_a distortion. The scaled η variant achieves pointwise bounds on all measures, and can be loosened to obtain global bounds (via the work in [Chen and Weber 2015]). The metric variant also achieves these same global bounds by achieving pointwise bounded conformal and isometric distortion on $\partial\Omega$.

Beyond their behavior with respect to the distortion, the variants also possess other advantages and disadvantages. The ν variant is simple and fast, but fails to interpolate stretch direction, leading to unintuitive results when the keyframes differ greatly. The scaled η variant properly interpolates stretch direction, but the global scaling factor can lead to loss of locality, and it can fail to be smooth with respect to time. The metric variant usually has the best qualitative results, is smooth, and respects stretch direction, but is slightly slower and less trivial to implement due to the Hilbert transform which is approximated numerically using BEM.

In comparison to other existing methods, we would like to stress that our variants are almost the only ones that provide mathematical guarantees on all three measures of geometric distortion with the exception of [Chen and Weber 2015] which can fail due to feasibility reasons. The algorithm of [Chen et al. 2013] has conformal distortion guarantees, but no isometric distortion guarantees.

With respect to qualitative behavior, we feel that the variants that interpolate stretch direction (scaled η and metric) outperform nearly all other pre-existing methods, e.g., ARAP and FFMP. The only algorithm which is comparable is that of Chen's [2013], which is

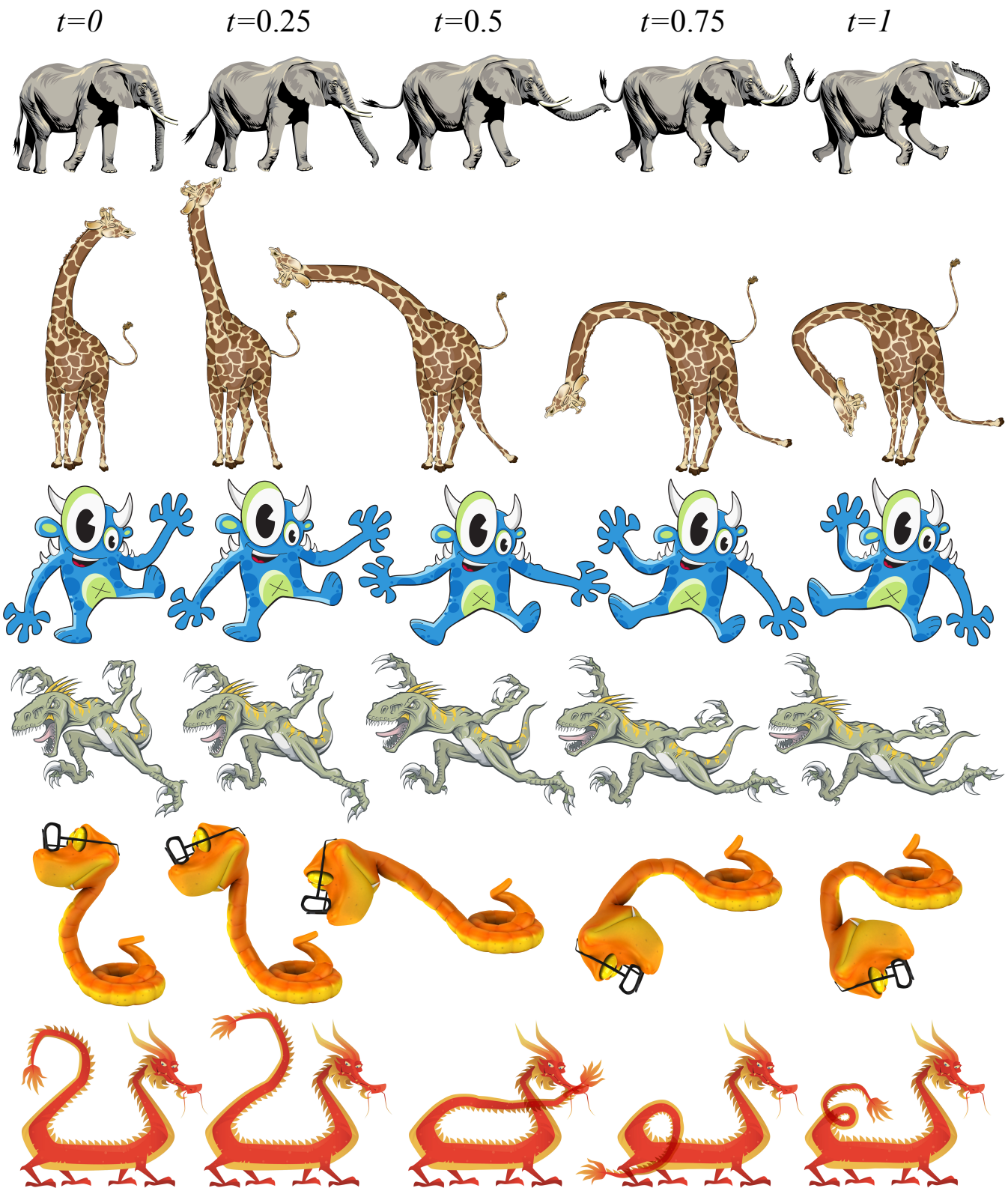

Figure 10: *Interpolation obtained by the metric-based method.*

slower by a wide margin. In fact, save for the naive linear interpolation, our variants are faster than all other pre-existing methods. In light of these comparisons, we feel that our algorithms offer a strong combination of qualitative superiority, speed, and mathematical certification.

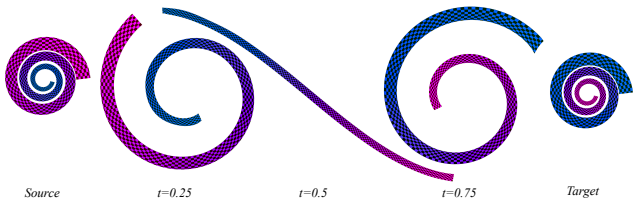

Figure 11: *Application of our method (scaled η variant) to meshes.*

9.1 Limitations

We are aware that the restriction to harmonic input (and output) that is expressed in closed form may be viewed as a disadvantage if such mappings are not available or desired. Investigations have begun into applying the method to discrete harmonic and non-harmonic mesh-based mappings. Figure 11 is the result of a method that follows the spirit of the basic method presented in this paper. The inputs for this image are two non-harmonic meshes, and the mappings are piecewise linear, with piecewise constant quantities $f_z, f_{\bar{z}}, \eta$ for each triangle. These were blended per triangle with the procedure of the scaled η variant. With the mesh input, there is no guarantee on integrability with this procedure, so a Poisson step had to be performed as the final step before production of the mapping. As can be seen, the output is quite natural.

Furthermore, experiments have been conducted with discrete harmonic mappings on meshes (with cotangent weights). While our method still fails to give exactly integrable discrete mappings, we have noticed that the Poisson error converges to zero under refinement of the mesh, as expected.

Finally, we acknowledge the limitation to planar mappings, and will pursue further methods that are based on metric tensor blending, which generalizes to three dimensions (unlike the mathematical machinery of complex analysis).

9.2 Future Work and Applications

There are several avenues for future work and applications of the ideas and methods in this paper. First it would be interesting to explore the use of our interpolation methods for 3D volumetric reconstruction from 2D slices (e.g. MRI data). In addition, the core idea of splitting a harmonic mapping into holomorphic and anti-holomorphic parts is also useful for shape deformation, as is done in [Levi and Weber 2016]. Finally, [Knöppel et al. 2013] is another work where this decomposition is used for vector field design on meshes. It would be interesting to combine these works for the sake of interpolation of vector fields.

Acknowledgements

This research was partially funded by the Israel Science Foundation (grants No. 1869/15 and 2102/15) and by the Max Planck Center for Visual Computing and Communication. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the GPU. Finally we thank the anonymous reviewers for their valuable comments and suggestions.

References

AHLFORS, L. 1979. *Complex analysis*, vol. 7. McGraw-Hill Education.

AIGERMAN, N., AND LIPMAN, Y. 2013. Injective and bounded distortion mappings in 3D. *ACM Transactions on Graphics (TOG) 32*, 4, 106.

ALEXA, M., COHEN-OR, D., AND LEVIN, D. 2000. As-rigid-as-possible shape interpolation. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 157–164.

ALEXA, M. 2002. Recent advances in mesh morphing. In *Computer graphics forum*, vol. 21, Wiley Online Library, 173–198.

BAXTER, W., BARLA, P., AND ANJYO, K. 2008. Rigid shape interpolation using normal equations. In *Proceedings of the 6th international symposium on Non-photorealistic animation and rendering*, ACM, 59–64.

BELL, S. R. 1992. *The Cauchy transform, potential theory and conformal mapping*, vol. 7. CRC press.

CHEN, R., AND WEBER, O. 2015. Bounded distortion harmonic mappings in the plane. *ACM Transactions on Graphics (TOG) 34*, 4, 73.

CHEN, R., WEBER, O., KEREN, D., AND BEN-CHEN, M. 2013. Planar shape interpolation with bounded distortion. *ACM Transactions on Graphics (TOG) 32*, 4, 108.

CHOI, J., AND SZYMCZAK, A. 2003. On coherent rotation angles for as-rigid-as-possible shape interpolation. In *Proceedings of the 15th Canadian Conference on Computational Geometry*, 111–114.

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

KIRCHER, S., AND GARLAND, M. 2008. Free-form motion processing. *ACM Transactions on Graphics (TOG) 27*, 2, 12.

KNÖPPEL, F., CRANE, K., PINKALL, U., AND SCHRÖDER, P. 2013. Globally optimal direction fields. *ACM Transactions on Graphics (TOG) 32*, 4, 59.

KOVALSKY, S. Z., AIGERMAN, N., BASRI, R., AND LIPMAN, Y. 2014. Controlling singular values with semidefinite programming. *ACM Transactions on Graphics (TOG)*, 4.

KOVALSKY, S. Z., AIGERMAN, N., BASRI, R., AND LIPMAN, Y. 2015. Large-scale bounded distortion mappings. *ACM Transactions on Graphics (TOG) 34*, 6, 191.

LEVI, Z., AND WEBER, O. 2016. On the convexity and feasibility of the bounded distortion harmonic mapping problem. *ACM TOG 35*, 4.

LEVI, Z., AND ZORIN, D. 2014. Strict minimizers for geometric optimization. *ACM Transactions on Graphics (TOG) 33*, 6, 185.

LIPMAN, Y. 2012. Bounded distortion mapping spaces for triangular meshes. *ACM Transactions on Graphics (TOG) 31*, 4, 108.

PORANNE, R., AND LIPMAN, Y. 2014. Provably good planar mappings. *ACM Transactions on Graphics (TOG) 33*, 4, 76.

SCHÜLLER, C., KAVAN, L., PANOZZO, D., AND SORKINE-HORNUNG, O. 2013. Locally injective mappings. In *Computer Graphics Forum*, vol. 32, Wiley Online Library, 125–135.

SURAZHSKY, V., AND GOTSMAN, C. 2001. Controllable morphing of compatible planar triangulations. *ACM Transactions on Graphics 20*, 4, 203–231.

SURAZHSKY, V., AND GOTSMAN, C. 2003. Intrinsic morphing of compatible triangulations. *International Journal of Shape Modeling 9*, 02, 191–201.

TUTTE, W. 1963. How to draw a graph. *Proc. London Math. Soc 13*, 3, 743–768.

WEBER, O., AND GOTSMAN, C. 2010. Controllable conformal maps for shape deformation and interpolation. *ACM Transactions on Graphics (TOG) 29*, 4, 78.

WEBER, O., AND ZORIN, D. 2014. Locally injective parametrization with arbitrary fixed boundaries. *ACM Transactions on Graphics (TOG) 33*, 4, 75.

WEBER, O., SORKINE, O., LIPMAN, Y., AND GOTSMAN, C. 2007. Context-aware skeletal shape deformation. In *Computer Graphics Forum*, vol. 26, Wiley Online Library, 265–274.

WEBER, O., BEN-CHEN, M., AND GOTSMAN, C. 2009. Complex barycentric coordinates with applications to planar shape deformation. *Computer Graphics Forum 28*, 2, 587–597.

WEBER, O., MYLES, A., AND ZORIN, D. 2012. Computing extremal quasiconformal maps. *Computer Graphics Forum 31*, 5, 1679–1689.

WOLBERG, G. 1998. Image morphing: a survey. *The visual computer 14*, 8, 360–372.

XU, D., ZHANG, H., WANG, Q., AND BAO, H. 2006. Poisson shape interpolation. *Graphical models 68*, 3, 268–281.

A Recovery of Bounds by Scaling η

As noted in Section 5.5, the scaling constant ρ is global, but we first discuss the necessary scaling for a single point. Let $\tilde{f}_{\bar{z}}^t$ denote the value defined in Equation (8) and revert to using $f_{\bar{z}}^t$ to refer to the value obtained with Equation (7) in the unscaled η variant. Similarly, let $k^t, \sigma_a^t, \sigma_b^t$ refer to the values obtained with the unscaled η variant, and use $\tilde{k}^t, \tilde{\sigma}_a^t, \tilde{\sigma}_b^t$ to refer to the respective quantities obtained with scaled $\tilde{\eta}$.

A.1 Recovery at a Point

It is easiest to see that the conformal distortion bounds can be recovered. Suppose that for some fixed time t, we have $k^t = |\eta|^t / \left|f_z^t\right|^2 > \max(k^0, k^1)$. We may assume $\max(k^0, k^1) > 0$, because if $\max(k^0, k^1) = 0$, then $k^t = k^0 = k^1 = 0$ (as $\eta^t = \eta^0 = \eta^1 = 0$). Then it is clear to see that for some constant $\rho(t) \in (0, 1]$ we will have:

$$\tilde{k}^t = \frac{\left|\tilde{\eta}^t\right|}{\left|f_z^t\right|^2} = \rho(t) k^t < \max(k^0, k^1) \tag{16}$$

For the bound on σ_a, suppose that for some t we have $\sigma_a^t = \left|f_z^t\right| + \left|f_{\bar{z}}^t\right| > \max(\sigma_a^0, \sigma_a^1)$. By Equation (8), we have $\left|\tilde{f}_{\bar{z}}^t\right| = \rho(t) \left|f_{\bar{z}}^t\right|$. We desire a constant $\rho(t) \in (0, 1]$ (likely different from the one needed to bound conformal distortion), so that the following expression holds:

$$\begin{aligned} \tilde{\sigma}_a^t &= \left|f_z^t\right| + \rho(t) \left|f_{\bar{z}}^t\right| \\ &\leq \max(\sigma_a^0, \sigma_a^1) = \max(\left|f_z^0\right| + \left|f_{\bar{z}}^0\right|, \left|f_z^1\right| + \left|f_{\bar{z}}^1\right|). \end{aligned} \tag{17}$$

It is clear that a suitable scaling $\rho(t)$ will exist if $\left|f_z^t\right| < \max(\sigma_a^0, \sigma_a^1)$. As we obtain f_z^t by logarithmic interpolation, note the following:

$$\begin{aligned} \left|f_z^t\right| &= \left|f_z^0\right|^{1-t} \left|f_z^1\right|^t \leq \max(\left|f_z^0\right|, \left|f_z^1\right|) \\ &\leq \max(\left|f_z^0\right| + \left|f_{\bar{z}}^0\right|, \left|f_z^1\right| + \left|f_{\bar{z}}^1\right|) = \max(\sigma_a^0, \sigma_a^1). \end{aligned}$$

Given that the distortion bounds are automatically achieved at the endpoints, we may assume $t \in (0, 1)$, and the first inequality becomes strict unless $\left|f_z^0\right| = \left|f_z^1\right|$.

If this case occurs, then $\left|f_z^t\right| = \left|f_z^0\right| = \left|f_z^1\right|$, and Equation (17) becomes equivalent to $\rho(t) \left|f_{\bar{z}}^t\right| \leq \max(\left|f_{\bar{z}}^0\right|, \left|f_{\bar{z}}^1\right|)$. There is clearly a suitable scaling here, unless $\max(\left|f_{\bar{z}}^0\right|, \left|f_{\bar{z}}^1\right|) = 0$, in which case $\left|f_{\bar{z}}^t\right| = \left|f_{\bar{z}}^0\right| = \left|f_{\bar{z}}^1\right| = 0$ and the distortion bound holds trivially.

An analogous argument shows that the bound on σ_b is recovered, so we omit a detailed explanation. For use in Section 7.3 we note the analogous equation to Equation (17). If the bound is violated at a time t, we would like a scaling constant $\rho(t)$ so that:

$$\left|f_z^t\right| - \rho(t) \left|f_{\bar{z}}^t\right| \geq \min(\left|f_z^0\right| - \left|f_{\bar{z}}^0\right|, \left|f_z^1\right| - \left|f_{\bar{z}}^1\right|). \tag{18}$$

Lastly, we have that local injectivity follows automatically with recovery of the bound on σ_b, as local injectivity is equivalent to $\sigma_b > 0$.

A.2 Global Recovery

To obtain a suitable global scaling constant, we must ensure that distortion bounds are respected at every point in the domain. In particular, if a bound is violated for a fixed time t, we choose a $\rho(t)$ that is less than or equal to the minimum of the scalings required at each point. Compactness of $\overline{\Omega}$ ensures that a positive value will suffice.

Finally, it must recover all of the distortion bounds, so some minimum of the recovery scaling factors for each quantity $(k^t, \sigma_a^t, \sigma_b^t)$ needs to be taken. Further details are in Section 7.3.

B Lemma on Discriminant Positivity

Lemma 7. $(\mathcal{A}^t)^2 - 4\left|\eta^t\right|^2 > 0$, $\forall t \in [0, 1]$.

Proof. By the triangle inequality, we have that $\left|\eta^t\right| \leq (1-t)\left|\eta^0\right| + t\left|\eta^1\right|$. This means that we have the following inequality:

$$\begin{aligned} (\mathcal{A}^t)^2 - 4\left|\eta^t\right|^2 &\geq (\mathcal{A}^t)^2 - 4\left((1-t)\left|\eta^0\right| + t\left|\eta^1\right|\right)^2 \\ &= (1-t)^2(\det J_f^0)^2 + t^2(\det J_f^1)^2 \\ &\quad + 2t(1-t)\left(\mathcal{A}^0(\sigma_b^1)^2 + 2(\sigma_b^0)^2\left|\eta^1\right|\right) \end{aligned}$$

For the last equality, many manipulations have been omitted here for the sake of brevity, but it may be easily checked by expressing all quantities in terms of $\left|f_z^0\right|, \left|f_{\bar{z}}^0\right|, \left|f_z^1\right|, \left|f_{\bar{z}}^1\right|$. Now, note that the three terms in the last expression are all non-negative for $t \in [0, 1]$. Furthermore, the first term is strictly positive when $t \neq 1$ and the second term is strictly positive when $t \neq 0$. □