Complex Wrinkle Field Evolution

Zhen Chen, Danny M Kaufman, Mélina Skouras, Etienne Vouga

Complex Wrinkle Field Evolution

ZHEN CHEN, The University of Texas at Austin, USA
DANNY KAUFMAN, Adobe, USA
MÉLINA SKOURAS, Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, France
ETIENNE VOUGA, The University of Texas at Austin, USA

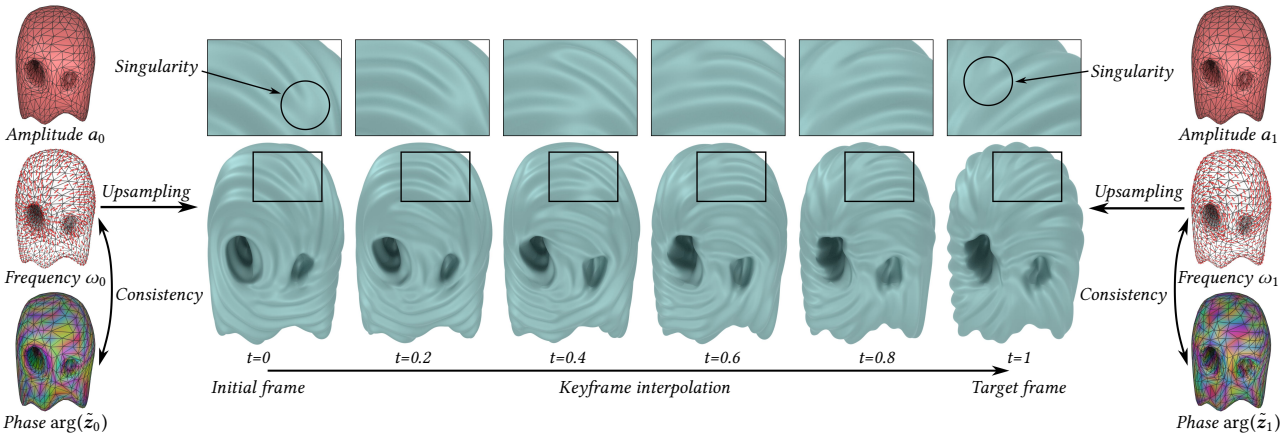

Fig. 1. We propose Complex Wrinkle Fields (*CWF*s), a new discrete wrinkle model that enables the resolution of highly detailed wrinkle patterns on coarse base-mesh geometry. The *CWF* representation consists of a positive number a per vertex encoding the wrinkle amplitude, a one-form ω per edge to model wrinkle frequency, and a complex number $\tilde{z}$ per vertex to represent wrinkle phase, coupled via a weak variational consistency condition ensuring that $\tilde{z}$ can capture singularities while also being as compatible with ω as possible (§ 3.1). We equip the *CWF* representation with a novel temporal interpolation algorithm (§ 4) and a spatial upsampling method (§ 5) that together allow for smooth interpolation between wrinkle patterns represented on surfaces by *CWF*s (leftmost and rightmost column), and base-mesh-independent rendering of arbitrarily high-resolution wrinkle patterns. Together these contributions make it possible to smoothly evolve wrinkle patterns between two prescribed keyframes (middle columns) with automatic merging, splitting, and reconnection of wrinkles as necessary via smooth sliding of singularities across the surface (zoomed-in figures in middle columns). Please check the **Interpolation Results** extra video (00:38–00:53; note this is an additional video separate from the main supplemental video) for the corresponding wrinkle animation.

We propose a new approach for representing wrinkles, designed to capture complex and detailed wrinkle behavior on coarse triangle meshes, called Complex Wrinkle Fields. Complex Wrinkle Fields consist of an almost-everywhere-unit complex-valued phase function over the surface; a frequency one-form; and an amplitude scalar, with a soft compatibility condition coupling the frequency and phase. We develop algorithms for interpolating between two such wrinkle fields, for visualizing them as displacements of a Loop-subdivided refinement of the base mesh, and for making smooth local edits to the wrinkle amplitude, frequency, and/or orientation. These algorithms make it possible, for the first time, to create and edit animations of wrinkles on triangle meshes that are smooth in space, evolve smoothly through time, include singularities along with their complex interactions, and that represent frequencies far finer than the surface resolution.

Authors' addresses: Zhen Chen, zchen96@cs.utexas.edu, The University of Texas at Austin, Austin, TX, USA; Danny Kaufman, dannykaufman@gmail.com, Adobe, Seattle, WA, USA; Mélina Skouras, melina.skouras@inria.fr, Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, Grenoble, France; Etienne Vouga, evouga@cs.utexas.edu, The University of Texas at Austin, Austin, TX, USA.

0730-0301/2023/1-ART1
https://doi.org/10.1145/3592397

CCS Concepts: • **Computing methodologies** → **Mesh geometry models**.

Additional Key Words and Phrases: discrete differential geometry, complex representation, wrinkle fields, singularities

ACM Reference Format:
Zhen Chen, Danny Kaufman, Mélina Skouras, and Etienne Vouga. 2023. Complex Wrinkle Field Evolution. *ACM Trans. Graph.* 1, 1, Article 1 (January 2023), 19 pages. https://doi.org/10.1145/3592397

1 INTRODUCTION

Across widely ranging spatial and temporal scales, wrinkles on surfaces are a fundamental geometric structure. We encounter these structures in our daily interactions with the moving folds and creases in cloth, skin and films, and likewise, we regularly observe them in natural phenomena such as the slow evolution of sand dunes and the rapid rippling of shallow water. Wrinkles on surfaces thus critically enrich otherwise coarse geometric structures with important visual details, while manufactured wrinkle patterns enable the design of complex structures and material behaviors [Chen et al. 2021; Lähner et al. 2018; Zuenko and Harders 2019].

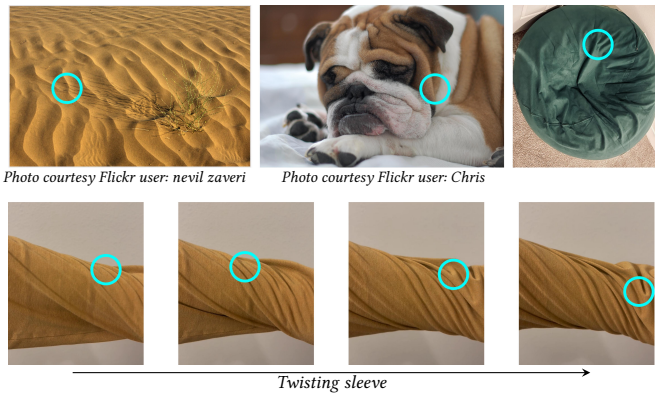

Fig. 2. Wrinkles on surfaces are ubiquitous, from sand dunes (*top-left*) to the creases and folds on a wrinkly dog face (*top-middle*) and a bean bag chair (*top right*). Notice that several key features are evident in all of these examples: frequency and amplitude that vary smoothly over the surface, punctuated by singularities (we circle some examples in aqua) where frequency diverges, amplitude vanishes, and multiple wave crests and troughs commingle in a Y-like pattern. In the bottom row, we show a sequence of wrinkle patterns formed during the twisting of a cloth shirtsleeve, where the singularities appear, merge and slide during the pattern's evolution.

The real-world examples illustrated in Figure 2 demonstrate the characteristic and highly complex features of surface wrinkling behaviors in the wild. Locally, wrinkles are generally characterized by a dominant frequency and amplitude, both of which vary inhomogeneously *but smoothly* over the surface. At the same time *singularities* punctuate wrinkle patterns (see e.g. circled regions in Figure 2). These singularities are the branch points of the phase field where multiple peaks and troughs join together. Near these singularities wrinkles are fundamentally *non-bandlimited*: as we approach a singularity, circulating an arbitrarily small distance around it results in an arbitrarily large shift in phase. Figure 2, bottom row, shows several frames of wrinkle evolution during twisting of a sleeve of a cloth. Singularities appear, merge, and slide over the surface as wrinkles shift, split, and reconnect during their evolution.

In the smooth setting (see § 2.1) wrinkles have an appealing and compact representation as amplitude and phase signals-on-surface; on the other hand the above characteristic features pose significant modeling challenges when it comes to discretizing and evolving wrinkles on triangle meshes. In this work we address these challenges.

1.1 Contributions

First, because the evolving singularities and high-frequency wrinkles in fine wrinkle patterns are generally not captured at practical mesh resolutions, we construct a discretization of wrinkles called the Complex Wrinkle Field (*CWF*) that captures in-element singularities and multiple sub-element wrinkling periods (§ 3). The *CWF* representation enables mesh-independent resolution of fine wrinkles.

Second, to enable smooth design, editing, control, animation, and evolution between wrinkle patterns we construct a mechanics-based algorithm for continuous and temporally-coherent interpolation between *arbitrary* wrinkle patterns on surfaces (§ 4). Our algorithm takes as input two *CWF* endpoints ("keyframes") and solves a boundary value problem approximating a geodesic path in the space of inextensible shells. The results are complex, shifting wrinkle patterns, where singularities evolve automatically to support the necessary branching and merging of wrinkles needed to interpolate between the keyframed patterns.

Third, to complement *CWF*'s ability to model sub-element wrinkle resolution, we derive a new subdivision method that maps *CWF*s on triangle meshes to amplitude and phase on vertices generated by Loop iterates (§ 5). This method resolves arbitrarily fine wrinkled geometry, including singularities, while avoiding artifacts seen in existing baselines (see § 5.3) and so provides high-quality upsampling of wrinkles for rendering and other downstream applications.

We extensively evaluate each of these contributions relative to state-of-the-art alternatives, and then demonstrate their joint application to animating wrinkle evolution on triangle surfaces, and to smooth, user-in-the-loop wrinkle editing (including both local and global manipulations). Across these applications we show that our contributions account for evolving singularities while generating smooth and intuitive interpolants between wrinkle patterns (§ 6). For example, in Figure 3 (bottom row), we show that, with proposed techniques, we can generate wrinkles with the double frequency and half of amplitude of the simulated results from [Chen et al. 2021]—effectively replacing thick cloth with thinner, silkier material without re-simulation. Furthermore, we can swipe through a smooth interpolation of the wrinkle patterns to select the desired look.

Despite the prominence of evolving wrinkles in the natural world, we emphasize that to date there is **no** model or method that supports temporal wrinkle interpolation on surfaces (in the sense of computing a path of spatially- and temporally-continuous wrinkle patterns between prescribed keyframes). Our *CWF* representation and algorithms are the first designed from the ground up to address the challenges imposed by evolving singularities during wrinkle interpolation. These challenges are generally unavoidable, even when the underlying surface and the prescribed keyframes are simple. For example, consider the torus interpolation in Figure 3 (top row) and in the supplementary **Interpolation Results** video at 00:02–00:19. Although neither keyframe contains singularities, it is topologically impossible to follow a smooth path between them without introducing (and then annihilating) singularities along the way.

In summary, Complex Wrinkle Fields (*CWF*) make it possible, for the first time, across all wrinkle inputs, to create, edit, and animate detailed wrinkles on triangle meshes that are smooth in space and evolve smoothly through time; while, at the same time, implicitly including the complex interactions of singularities, and resolving wrinkle frequencies far finer than the surface mesh resolution. Raw data and a reference implementation of our algorithms, as of the time of publication, can be found in the ACM Digital Library. The latest version of the code can also be found on GitHub[1].

[1] https://github.com/zhenchen-jay/Complex-Wrinkle-Field.git

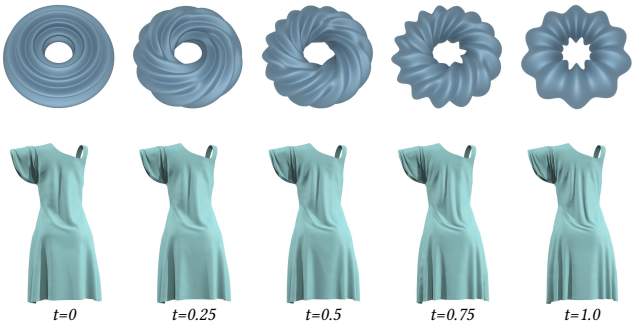

Fig. 3. Two examples of *CWF* interpolation between keyframes specified at $t = 0$ and 1. *Top row:* a torus, with wrinkles keyframed to rotate by ninety degrees; *bottom row:* a dress example from Chen et al. [2021], where we emulate replacing the cloth with a thinner material by uniformly doubling the wrinkle frequency and halving the amplitude. For the rotating torus example, although neither keyframe contains singularities, it is topologically impossible to follow a smooth path between them without introducing (and then annihilating) singularities along the way. Here our *CWF* interpolation allows wrinkles to break apart before reconnecting again. The dress demonstrates the possibility of using our algorithms to reasonably edit physical wrinkles and generate corresponding smooth animation. Videos of these two examples can be found in the **Interpolation Results** supplementary video at 00:02–00:19 (for the torus) and 03:47–04:03 (for the dress).

2 PRELIMINARIES AND RELATED WORK

2.1 Wrinkles on Surfaces

Smooth Setting. A useful [Knöppel et al. 2013, 2015] representation of wrinkles on smooth manifolds $\mathcal{M}$ is by a single complex smooth function $z : \mathcal{M} \to \mathbb{C}$, where the wrinkle amplitude a is encoded as $|z|$, and the associated phase as $\theta = \arg(z)$. Here we assume wrinkles are normal displacements of $\mathcal{M}$ with magnitude $a \cos\theta = \Re(z)$. The smoothness of z guarantees a smooth non-negative amplitude, and a smooth associated phase, except at a set of singular points where $z = 0$ and along branch cuts between them. At these singular points the phase is undefined and the amplitude vanishes. Away from these singularities, the derivative of phase yields the (one-form-valued) wave frequency $\omega = d\theta$. It is important to note that the frequency field singularities are not the usual ones studied in geometry processing where $\omega = 0$; rather, here $\|\omega\| \to \infty$ as one approaches a singularity.

Discrete Setting. Frequency and amplitude are *the* semantically-meaningful parameters for encoding the wrinkling phenomena discussed in §1 and shown in Figure 2. Our goal then is to translate the above spectral representation of wrinkles from the smooth setting to triangle meshes $\mathcal{T} = \{\boldsymbol{V}, \boldsymbol{E}, \boldsymbol{F}\}$. As we motivated in the last section, a practical discretization of $\boldsymbol{z}$ on a triangulation by a set of discrete parameters $\boldsymbol{u}$ should satisfy three basic desiderata:

(1) The parameters $\boldsymbol{u}$ must allow for singularities; i.e., it should be possible to reproduce wrinkle patterns with branch points as illustrated in Figure 2.
(2) High frequency wrinkling should be representable *independent* of the resolution of the underlying triangle mesh, and in particular, wrinkles should be allowed much more than a single period within a triangle. This decoupling (the main motivation for a spectral representation of wrinkles) is needed both because fine wrinkle patterns are common in the real world (as in the skin and cloth images in Figure 2) and because high frequencies are unavoidable in the vicinity of singularities.
(3) We must be able to render a smoothly wrinkled, high-resolution surface for all valid $\boldsymbol{u}$, *and* this surface geometry must change continuously for continuous changes in $\boldsymbol{u}$. This requirement represents a minimum foundation for doing temporally-coherent wrinkle interpolation.

Surprisingly, given how straightforward it is to represent wrinkles in the smooth setting, the most direct strategies for discretizing $\boldsymbol{z}$ (e.g., storing a complex number z_i on each vertex) do not meet the above requirements, as we discuss shortly; constructing the right discrete representation to support temporal interpolation is a challenge in its own right. After reviewing related work we will present our discrete *CWF* model in §3, and compare with some natural alternative choices of discretization and demonstrate their shortcomings.

2.2 Prior Work

Spectral Representations of Wrinkles on Surfaces. Prior work has recognized the significant challenges in representing wrinkles on triangle meshes, and has offered a range of partial solutions. A common approach [Aharoni et al. 2017; Chen et al. 2021; Paulsen et al. 2016] is to design for the wave frequency without regard to phase, using vector-field design tools from geometry processing; when phase is required (e.g., for rendering), it is recovered as a post-process by solving for the phase whose gradient most closely matches the frequency: either via least-squares optimization, or using techniques from the parameterization literature [Diamanti et al. 2015; Ling et al. 2015; Ray et al. 2006; Zhang et al. 2010]. Knöppel et al. [2015] propose a particularly powerful variation of this idea: their algorithm solves for a stripe pattern (phase field) over arbitrary surfaces given a user-provided vector field, where singularities are placed at the centers of the triangles as needed to produce topological dislocations in the pattern. Noma et al. [2022] build on this technique by allowing users to interactively edit the location of singularities in stripe patterns. Beyond designing stripe patterns, Knöppel et al.'s algorithm has been applied for real-time surface parameterization [Lichtenberg et al. 2018] and continuous fiber design [Boddeti et al. 2020].

While these "frequency-first" strategies can produce beautiful, high-frequency wrinkle patterns on surfaces, they all suffer from a key limitation: smooth change in frequency does not yield a smooth change in phase. These strategies are therefore unsuitable for smooth animation of wrinkles due to temporal incoherence: they will produce a temporally discontinuous phase sequence, even when the input frequency sequence undergoes a smooth temporal evolution. For example, in the **main** supplementary video (02:59–03:44) and the **Comparisons** video (00:05–00:59), we show an animation of rotating waves on a torus, where we apply the method described

by Chen et al. [2021] and Knöppel et al. [2015] to every frame. Although the individual frames are spatially smooth, they are not temporally coherent.

Vector Field Design. Over decades, many methods have been developed for vector field design. Vaxman et al. [2016] and de Goes et al. [2016a] summarize the classical ways to design and discretize vector fields on triangle meshes. In general, there are several ways to represent discrete vector fields. One popular approach specifies each vector with respect to a local Cartesian or polar coordinate system [Diamanti et al. 2014; Knöppel et al. 2013]. This representation is agnostic to singularities and well-suited for methods that automatically place them [Bommes et al. 2009; Diamanti et al. 2015; Jakob et al. 2015; Panozzo et al. 2014; Ray et al. 2009]. However, enforcing global integrability of the vector field in this representation is difficult and requires non-linear constraints or integer variables. Another approach represents vector fields implicitly in terms rotation angles on dual edges [Crane et al. 2010; Li et al. 2006; Ray et al. 2008]. This representation is particularly useful for parameterizing vector fields while maintaining full control over singularity placement and index [Fisher et al. 2007; Solomon and Vaxman 2019; Zhang et al. 2006] but integrability is even more elusive: *two* levels of integrability constraints are required (one for integrating the angles to well-defined vectors and another for integrating the vectors to a well-defined scalar field). As we discuss in § 3, neither of these representations are suited for representing temporally-coherent, high-frequency wrinkle fields with moving singularities on discrete triangle meshes.

Vector Field Subdivision. Subdivision operators have been designed that extend Loop subdivision to one-forms on triangle mesh edges [de Goes et al. 2016b; Wang et al. 2006]. We will use this scheme as a part of our approach (§ 5.1). Custers and Vaxman [2020] designed a subdivsion scheme for tangent directional fields on triangle faces. However, these methods do not cover the corresponding subdivision of amplitude and phase, which are essential for upsampling wrinkles.

Vector Field Temporal Interpolation. Temporally interpolating vector fields has long been studied in geometry processing. Zavala-Hidalgo et al. [2003] interpolate high-frequency vector wind fields by decomposing these fields into a basis of empirically-derived orthogonal functions, and then interpolating eigenmodes in time. Chen et al. [2012] propose a framework for designing time-varying vector fields by solving a spatial-temporal Poisson problem; their method implicitly generates bifurcations and singularities in the vector field. Sato et al. [2018] propose an algorithm that takes two entire time sequences of velocity fields and produces time sequences that blend between them. Solomon and Vaxman [2019] use optimal transport theory to match the singularities in two input vector fields, which allows for their direct interpolation. However, vector field interpolation, on its own, does not solve the problem of interpolating wrinkles, where the phase and amplitude must also be interpolated, and, as discussed above, there is no obvious way to recover temporally-coherent phase from frequency.

Spectral Wrinkle Evolution. There are several methods for simulating spectrally-represented wrinkles forwards through time from an initial state: in a line of work, Jeschke, Wojtan, and collaborators [2018; 2015; 2017] propose several approaches to efficiently evolve water waves based on Airy wave theory [1842], which uses a sum of sinusoidal functions to linearly approximate the motion of surface waves on the body of water. Zuenko and Harders [2019] simulate wrinkle motion on 3D surfaces via reaction-diffusion [Turk 1991; Witkin and Kass 1991]. Although these approaches generate animations of wrinkles given initial conditions (initial value problem), they do not address the wrinkle keyframe interpolation problem, which is completely different (boundary value problem).

Shell Geodesics. Heeren et al. [2012] propose a way to find the geodesic between two thin shells based on the minimization of strain dissipation. This idea is further explored for time discretization of geodesic calculus on certain Riemannian manifolds [Rumpf and Wirth 2014], understanding the geometry of the space of shells [Heeren et al. 2014], or finding geodesics between two general elastic shapes [Ezuz et al. 2019; Sassen et al. 2020]. Von-Tycowicz et al. [2015] adopt a similar idea and design a special subspace optimization problem to interpolate between elastic body poses in real time. Our wrinkle interpolation algorithm follows in the footsteps of these methods.

Surface Augmentation. Methods have also been proposed to augment coarse surfaces with fine wrinkles as an alternative to expensive, fine-scale physical simulation. Indirect techniques have been proposed for upsampling cloth [Bergou et al. 2007; Wang 2021] and skin [Rémillard and Kry 2013] by simulating a higher-resolution mesh that is constrained to stay close to an existing, coarse simulation. Rohmer et al. [2010] and Gillette et al. [2015] add fine wrinkle detail to coarse cloth simulations by tracing compression direction fields in a temporal coherent manner. Cutler et al. [2005] allow users to design wrinkle patterns on a set of reference poses, then add weighted combinations of the designed wrinkles to each frame of simulation based on local stresses. Data-driven methods for enriching coarse meshes with fine detail, for example, by representing wrinkles as normal maps [Lähner et al. 2018] or displacements [Chen et al. 2018, 2021; Santesteban et al. 2019; Zhang et al. 2021], are likewise an active and rapidly evolving area of research but require significant preprocessing and generally depend on a corpus of data.

3 COMPLEX WRINKLE FIELD DISCRETIZATION

In the smooth setting, a single complex function z suffices to represent a wrinkle pattern. Unfortunately, z admits no straightforward discretization that safisfies the properties listed in § 2.1. In this section, we introduce our complex wrinkle field (*CWF*) representation, the design decisions that motivate it, and some of the pitfalls with alternative discretizations. In the following sections we will then discuss how to both (a) temporally interpolate between *CWF*s and (b) upsample (and so render) *CWF*s while preserving these properties.

High-level Summary. There are three key design decisions that motivate *CWF*s:

- To represent high-frequency wrinkles on a mesh, unlike in the smooth setting, a phase-based representation is insufficient. The discretization must additionally track frequency.
- For the wrinkle representation to capture singularities, the frequency field cannot be globally integrable in the usual sense

of being the derivative of a real-valued function, *and* cannot even be locally integrable near singularities. We therefore eschew integrability entirely and allow arbitrary frequencies.
- Since frequency is not integrable, we cannot and should not exactly enforce that frequency is the derivative of phase. But we also do not want phase and frequency to be incompatible. We therefore require *soft* compatibility of phase and frequency, in a way that is well-defined at singularities and enforces $\mathrm{d}\arg(z) \approx \omega$ away from singularities.

A *CWF* therefore consists of three sets of degrees of freedom: (1) a one-form ω_{jk} on the edges of $\mathcal{T}$ (representing the wrinkle pattern frequency); (2) a real scalar a_j per mesh vertex (representing the wrinkle amplitude); and (3) a complex number $\tilde{z}_j$ per vertex (encoding the wrinkle phase). There are no constraints on the ω_{jk} and a_j, but $\tilde{z}$ is required to be approximately unit and to satisfy a *weak compatibility* condition with respect to ω. The reader immediately interested in the formal definition of a *CWF* may skip to § 3.1; in what follows we motivate and justify each of the above decisions.

Need for Both Frequency and Phase. The most direct way to discretize z is as a discrete function on the mesh vertices. The strength of this approach is that it uniquely pins down the wrinkle amplitude $|z|$ and wrinkle phase $\arg(z)$ at each vertex. However, z alone is unable to represent wrinkles with high frequencies, as illustrated in Figure 4: at most one wrinkle period per edge is possible, since the phase change $\arg(z_j) - \arg(z_i)$ across the edge is bounded by 2π. By contrast, a frequency one-form ω can encode arbitrarily high frequencies independent of mesh resolution; however, a discrete representation based on frequency alone is unsuitable for applications like ours requiring temporal coherence, since wrinkle phase is underdetermined given frequency alone.

A mixed representation that tracks both phase and frequency simultaneously combines their advantages, and eliminates their disadvantages. This observation motivates a discrete representation that includes both a real-valued one-form ω_{ij} on the mesh edges, and a complex number z_j per mesh vertex. (In the *CWF* representation, we further decompose z_j into parts as $z_j = a_j\tilde{z}_j$ with $a_j, \tilde{z}_j$ as our DOFs rather than z_j itself, for reasons we cover below.)

No Integrability Constraints on Frequency. In the smooth setting, frequency is *defined* as the derivative of phase. A natural question when discretizing frequency is whether to require it, by analogy, to be (discretely) integrable. Several important subtleties make doing so impractical. Note that, in the smooth setting, ω is not globally integrable in the usual sense: ω is the exterior derivative of a section of an S^1-bundle over $\mathcal{M}$-with-singularities-removed. The line integral of ω around a singularity does not necessarily vanish and is instead a multiple of 2π. Requiring $\oint \omega = 0$ around every closed curve in $\mathcal{M}$ would constrain the wrinkle pattern to have *no* singularities. Similarly, in the discrete setting, if a triangle $jk\ell$ contains a singularity, the discrete curl on that triangle $\omega_{jk} + \omega_{k\ell} + \omega_{\ell j}$ does not vanish. Rather than selectively enforcing integrability of ω on only some triangles (which would require explicitly tracking which triangles contain singularities of which index using integer variables, etc.) we instead follow in the footsteps of previous work [Knöppel et al. 2015; Zuenko and Harders 2019] and allow arbitrary ω. The singularities in the wrinkle pattern are then implicitly determined by where ω fails to be integrable.

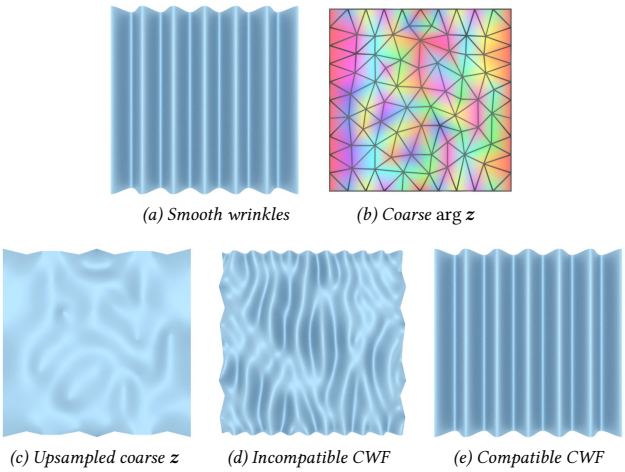

Fig. 4. We show different discretizations of a plane wave associated with $z = \exp(3\pi i x)$. (a): the exact wrinkle pattern z, visualized as a normal displacement with magnitude $\Re(z)$. (b): we sample z onto a coarse mesh and plot $\arg(z_j)$; the discrete phase is badly aliased. (c): attempting to visualize the wrinkles by displacing each vertex by $\Re(z_j)$ yields a surface that bears no resemblance to the expected wrinkle pattern in subfigure (a). (d): the result when we apply our subdivision algorithm from § 5 to visualize a *CWF* with $z_j = \exp(3\pi i x_j)$ and the unrelated, incompatible $\omega_{jk} = (0, 3\pi) \cdot (\boldsymbol{v}_k - \boldsymbol{v}_j)$. Although the rendered wrinkled mesh is smooth, it is riddled with mesh-dependent singularities, and ω has little semantic relationship to the waves in the resulting pattern. We insist on soft compatibility between ω and $\tilde{z}$ to avoid such artifacts. (e): this time, we use $\omega_{jk} = (3\pi, 0) \cdot (\boldsymbol{v}_k - \boldsymbol{v}_j)$, a compatible frequency that gives us a valid *CWF*. When rendered, this discrete *CWF* is indistinguishable from the exact pattern.

Frequency-Phase Consistency. Since we do not require ω to be integrable, ω cannot in general be exactly the derivative of phase. Recall, however, that the motivation for including frequency in our discrete representation is so that it can supply the information missing from z_j (alone) about the jump in phase across a mesh edge. For this purpose, frequency is only useful if, far away from singularities, we *do* have $\omega \approx \mathrm{d}\arg(z)$. (See Figure 4d for an example where frequency and phase are totally uncorrelated.) We therefore require the phase variables in a *CWF* to be "as compatible as possible" with the frequency, (a) given that frequency is not necessarily integrable and (b) accounting for the necessary presence of singularities.

In the smooth setting, one formulation of the above *soft* compatibility condition is to require z to be compatible in a least-squares sense by minimizing an energy similar to

$$\int_{\mathcal{M}} \|\mathrm{d}\arg(z) - \omega\|^2 \, \mathrm{d}A. \tag{1}$$

We cannot use this energy as-is, as it is formally undefined near singularities and at branch cuts. To avoid the discontinuity in $\arg(z)$ at branch cuts, we rewrite the constraint in terms of $\tilde{z} = z/|z|$ [Knöppel

et al. 2015],

$$\tilde{z} = \arg\min_{\tilde{z}} \int_{\mathcal{M}} \|(\mathrm{d} - i\omega)\tilde{z}\|^2 \, \mathrm{d}A \quad s.t.\ |\tilde{z}| = 1, \tag{2}$$

which has the added benefit of eliminating some of the nullspace from Equation (1) (which is invariant under rescalings of z).

Singularity Handling. Equation (2) remains ill-posed at singularities (where neither $\tilde{z}$ nor ω are well-defined) and ill-conditioned close to the singularities, where $\|\omega\| \to \infty$.

There are many potential strategies for relaxing Equation (2) to regularize the behavior at singularities. Our approach is to relax the unit-norm constraint, so as to allow $\tilde{z}$ to vanish at singularities, using a Ginzburg-Landau-type term [Kohn 2006; Viertel and Osting 2019]:

$$\tilde{z} \in \mathrm{Opt}_\omega = \arg\min_{\tilde{z}} \int_{\mathcal{M}} \left[\|(\mathrm{d} - i\omega)\tilde{z}\|^2 + \left(|\tilde{z}|^2 - 1\right)^2 \right] \mathrm{d}A. \tag{3}$$

For later reference, we define:

$$\begin{aligned} E_{\mathrm{compat}}(\tilde{z}, \omega) &= \int_{\mathcal{M}} \|(\mathrm{d} - i\omega)\tilde{z}\|^2 \, \mathrm{d}A, \\ E_{\mathrm{unit}}(\tilde{z}) &= \int_{\mathcal{M}} \left(|\tilde{z}|^2 - 1\right)^2 \, \mathrm{d}A. \end{aligned} \tag{4}$$

We next discretize this variational characterization of $\tilde{z}$ to define the final, discrete soft compatibility conditions we impose to our *CWF* representation.

3.1 Complex Wrinkle Field Representation

Motivated by the above discussion, we define *CWF* to consist of the following degrees of freedom:

(1) a (not necessarily integrable) frequency one-form, encoded as a real number ω_{jk} on each directed edge of the mesh (with $\omega_{jk} = -\omega_{kj}$);
(2) a real number a_j and complex number $\tilde{z}_j$ on each vertex of the base triangle mesh $\mathcal{T}$, which together encode the approximate amplitude and direction of $z_j = a_j\tilde{z}_j$ discretizing the smooth setting's amplitude-phase field z;
(3) with the z_j constrained to be *softly compatible* with the frequency one-form, $\mathrm{d}\arg(z) \approx \omega$, via discretization of Equation (3) on the base triangle mesh.

To discretize the first, least-squares, compatability term of Equation (3), we apply the formulation proposed by Knöppel et al. [2015], as it is designed to be robust in the case that ω_{jk} is larger than 2π on an edge jk,

$$\hat{E}_{\mathrm{compat}}(\tilde{z}, \omega) = \sum_{\text{edges } k\ell} A_{k\ell} \left|\tilde{z}_k \exp(i\omega_{k\ell}) - \tilde{z}_\ell\right|^2, \tag{5}$$

and directly apply piecewise-constant discretization for the second, unit-length penalty term,

$$\hat{E}_{\mathrm{unit}}(\tilde{z}) = \sum_{\text{vertices } j} A_j \left(\left|\tilde{z}_j\right|^2 - 1\right)^2, \tag{6}$$

where the $A_{k\ell}$ and A_j are edge and vertex barycentric area weights, respectively. We then require

$$\tilde{z} \in \hat{\mathrm{Opt}}_\omega = \arg\min_{\tilde{z}} \left[\hat{E}_{\mathrm{compat}}(\tilde{z}, \omega) + \hat{E}_{\mathrm{unit}}(\tilde{z})\right], \tag{7}$$

where $\hat{\mathrm{Opt}}_\omega$ defines our discretized consistency conditions.

Remarks. Note that in Equations (4) and (7) we have made an implicit choice to weight the compatibility and unit-norm terms equally. For regions away from the singularities, $E_{\mathrm{compat}} \approx 0$, so that Equation (4) gives us unit $\tilde{z}$ for any positive scaling of E_{unit}. For regions near the singularities, $E_{\mathrm{compat}} \to \infty$ unless $\tilde{z} \to 0$ so that, for any scaling of E_{unit}, the least-squares energy E_{compat} is the dominant term, which ensures that $\tilde{z} \approx 0$. These observations show that Opt_ω is largely invariant to the relative scaling of the two constraint terms, and we make the simplest choice to weight them equally. Note also that Opt_ω is a nontrivial subspace and contains more than a single element. At minimum, if $\tilde{z} \in \mathrm{Opt}_\omega$, then so is its global phase shift $\exp(i\theta)\tilde{z}$ for every $\theta \in [0, 2\pi)$.

4 INTERPOLATION

In this section we present an algorithm for *temporal interpolation* of *CWF*s: given two *CWF*s as boundary conditions, $\left(\omega^0, a^0, \tilde{z}^0\right)$ and $\left(\omega^1, a^1, \tilde{z}^1\right)$, we seek a smooth interpolant $\gamma(t) = (\omega[t], a[t], \tilde{z}[t])$ through the space of *CWF*s which matches the prescribed boundary conditions at $t = 0, 1$ and which approximates the behavior of real-world wrinkled materials deforming over time.

The challenge with computing such an interpolation is that while there are many smooth choices of interpolant $\gamma(t)$, they are often qualitatively unacceptable. For physically-based wrinkle evolution, wrinkles should *slide* over the surface rather than disappear in one location and then reappear at a target location, and singularities should slide smoothly over the surface. Otherwise plausible-seeming interpolants that are not specifically designed to promote physical fairness of the wrinkle motion regularly produce obvious visual artifacts that are unacceptable for many applications such as animation or editing. For many examples please see § 4.3 and the **Interpolation Results** video.

4.1 Our Approach

To compute physics-inspired, smooth paths between boundary conditions in the space of *CWF*s, we begin with the closely-related problem of computing geodesics in the space of thin shells (see e.g. the work of Heeren et al. [2012]). For our application, the key idea from this work boils down to defining a metric on shell space that measures the length of a path between configurations in terms of the integrated norm of strain-rate along that path. A shortest geodesic in this space is then, effectively, the path that requires doing the least work on the system when deforming the shell along that path.

To compute comparable wrinkling geodesics for *CWF*s we measure a bending strain[2] given in terms of the combined geometry

[2]We make the simplest assumption that the stretching strains in the wrinkled surface represented by a *CWF* are negligible. This decision is justified as compressive stresses should be mostly absorbed by wrinkling. That said, unless both *CWF* keyframes of an interpolation represent isometric deformations of the same underlying surface, there must be some amount of stretching involved and so it should be interesting to consider inclusion of stretching measures as well in future work.

of the underlying wrinkle-free base surface $\mathcal{M}$ and the wrinkle displacements [Chen et al. 2021],

$$\epsilon = \boldsymbol{I}^{-1}\left(\boldsymbol{II} - \bar{\boldsymbol{II}} - a\cos\theta\omega^T\omega\right) \tag{8}$$

$$= \boldsymbol{I}^{-1}\left(\boldsymbol{II} - \bar{\boldsymbol{II}} - \Re(z)[\mathrm{d}\arg z]^T[\mathrm{d}\arg z]\right), \tag{9}$$

where $\boldsymbol{I}$ and $\boldsymbol{II}$ are the first and second fundamental forms of the base surface and $\bar{\boldsymbol{II}}$ is the second fundamental form of the shell's assumed rest state. Following Heeren et al. [2012], we can then define our distance on paths $\gamma(t)$ by

$$d(\gamma) = \int_0^1 \int_{\mathcal{M}} \|\dot{\epsilon}[z(t)]\|^2 \mathrm{d}A\,\mathrm{d}t, \tag{10}$$

where, assuming a hyperelastic homogeneous and isotropic material, we apply the StVK material norm for $\|\cdot\|$; please see the supplemental document (§ 1) for details.

Computing Geodesics. Geodesics are then constrained minimizers γ of Equation (10) satisfying $\tilde{z}(t) \in \mathrm{Opt}_{\omega(t)}$. However, computing these geodesics by directly minimizing Equation (10) is difficult. In the supplemental document (§ 1.2) we derive, after a sequence of approximations and additional simplifying assumptions, an approximate solution to Equation (10) given by

$$a(t) = (1-t)a^0 + ta^1 \tag{11}$$

$$\tilde{z}(t) = \arg\min\ E_{\mathrm{smooth}}(\tilde{z}) + cE_{\mathrm{opt}}(\tilde{z}, \omega) \tag{12}$$

and an explicit formula for $\omega(t)$ given in Equation (34) (§ 1.2 in the supplemental document), where

$$E_{\mathrm{smooth}}(\tilde{z}) = \frac{1}{2}\int_{\mathcal{M}} g_{\mathrm{bd}} \int_0^1 \left|\dot{\tilde{z}}\right|^2 \mathrm{d}t\mathrm{d}A \tag{13}$$

$$g_{\mathrm{bd}} = \frac{\left(a^0\right)^2\left\|\omega^0\right\|^4 + \left(a^1\right)^2\left\|\omega^1\right\|^4}{2} \tag{14}$$

approximates Equation (10) after a and ω are substituted in and simplified—effectively giving a weighted Dirichlet energy in spacetime promoting smoothness of wrinkle phase; and

$$E_{\mathrm{opt}}(\tilde{z}, \omega) = \int_0^1 \left[E_{\mathrm{compat}}(\tilde{z}[t], \omega[t]) + E_{\mathrm{unit}}(\tilde{z}[t])\right]\,\mathrm{d}t \tag{15}$$

provides a penalty term, with stiffness c, for the constraint $\tilde{z} \in \mathrm{Opt}_\omega$ that $\tilde{z}$ and ω are approximately compatible. We use $c = 10^3 g_{\mathrm{ave}}$ in all of our examples, where

$$g_{\mathrm{ave}} = \int_0^1 \int_{\mathcal{M}} a^2\|\omega\|^4\,\mathrm{d}A\,\mathrm{d}t \tag{16}$$

is the spacetime averaged amplitude squared times frequency quartic, in order to make the cE_{opt} unit consistent. If c is too low, the interpolant $\gamma(t)$ fails to satisfy the *CWF* compatibility constraint, yielding noisy wrinkle pattern evolution and many extra singularities. If c is too high, the optimization problem (12) becomes numerically unstable and does not converge to a smooth solution. Please see § 7.2 in the supplemental document for some experiments probing the effect of the choice of c on our results and the suitability of our empirically selected value.

Discretization. Given two *CWF* boundary conditions $\left(\omega^0, a^0, \tilde{z}^0\right)$ and $\left(\omega^1, a^1, \tilde{z}^1\right)$ and a desired number of intermediate frames N, we discretize time into N steps of size $\delta t = 1/N$ and minimize a discretization of Equation (12) over the base triangle mesh $\mathcal{T}$ with a globalized Newton solve. Please see § 1.3 in the supplemental document for additional details, including the full formulas for all our discrete analogues of the above energy terms used in our final solve. Within our Newton implementation we use SuiteSparse's parallel implementation of supernodal sparse Cholesky decomposition [Chen et al. 2008] as our linear solver, and terminate when converged to an objective gradient norm smaller than 10^{-6}.

Linear Substepping. The cost of computing N interpolating frames using our algorithm is dominated by the cost of solving Equation (12), which scales roughly linearly in N. When many frames are needed, for example when rendering videos, this computation is wasteful, since consecutive frames will be almost identical. For large N, we suggest using our algorithm to compute $N' = N/k$ guide frames instead, and then linearly interpolating the *CWF* variables for k steps between each guide frame. As a rule of thumb, linear interpolation is an adequate approximation to a geodesic path between guide frames if the distance traveled by wrinkles during that time does not exceed the average mesh edge length. We use $N = 200$, with $N' = 50$ and $k = 4$ in all of our videos, and provide timing information for the examples in the paper in Table 1 (supplemental document). We further provide some experiments showing the effect of the choice of N' on the interpolant quality in the supplemental document (§ 7.1).

Handling Incompatible Boundary Conditions. The above algorithm assumes that provided boundary conditions are valid *CWF*s. If applied to boundary conditions that severely violate *CWF* soft compatibility, our interpolant will move abruptly near $t = 0$ and 1 (since Equation (15) will enforce the soft compatibility condition everywhere except at the infeasible, prescribed endpoints). Since in applications these boundary conditions will often be provided by the user (as keyframes for animation, e.g.) we propose pre- and post-processing steps that allows use of our algorithm even when the boundary conditions have incompatible frequency and phase.

In particular, notice that the discrete compatibility term in Equation (5) implies the constraint per edge

$$\tilde{z}_k \exp(i\omega_{k\ell}) = \tilde{z}_\ell. \tag{17}$$

This allows us to decompose any frequency one-form ω into a consistent, constraint-satisfying portion, ω_{com}, and a residual $\delta\omega$; that is, there is a unique one-form $\delta\omega$, with $-\pi \le (\delta\omega)_{jk} < \pi$ on each edge jk, such that $\omega_{\mathrm{com}} = \omega - \delta\omega$ and $\tilde{z}$ exactly satisfy Equation (17) on every edge. To handle non-*CWF* boundary conditions, we decompose the frequencies ω^0 and ω^1 into their consistent and inconsistent parts, and apply interpolation to the *CWF*s boundary conditions $(\omega^0_{\mathrm{com}}, a^0, \tilde{z}^0)$ and $(\omega^1_{\mathrm{com}}, a^1, \tilde{z}^1)$ instead. We then interpolate $\delta\omega$ using the same explicit interpolation applied for $\omega(t)$ given in Equation (34) (§ 1.2 in the supplemental document), and add this frequency component back into the final interpolant: $\gamma(t) = (\omega(t) + \delta\omega(t), a(t), \tilde{z}(t))$.

4.2 Results

We demonstrate that our interpolation algorithm yields high-quality wrinkle evolution given a variety of keyframes pairs. Even more interpolation examples, with keyframes created by local edits to wrinkle patterns, are discussed in § 6. Important note: it is difficult to assess the quality of the interpolant, and especially temporal coherence, from a sparse set of stills. Please see the **Comparisons** supplemental video for animations of these examples.

Change in Wrinkle Frequency. Increasing the frequency of a wrinkle pattern on a compact surface like the fertility model (Figures 5 and 6), though conceptually simple, involves a complex path through the space of wrinkle fields since it is impossible to have "fractional" numbers of wrinkles. New wrinkles must be born as the frequency increases, with singularities appearing in pairs and sliding over the surface to "unzip" a new wrinkle.

Change in Wrinkle Direction. We also show several examples of interpolation between wrinkle patterns that have been globally or locally rotated by ninety degrees relative to each other in Figures 1 and 7. The amplitude and phase plots show singularities appearing, splitting, and merging in a complex dance in both cases.

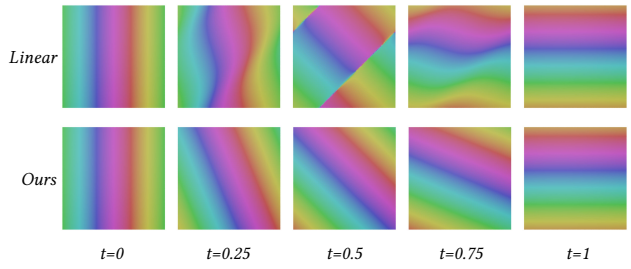

Fig. 8. Failure of linear interpolation given boundary conditions $z_0 = \exp ix$ and $z_1 = \exp iy$. We show the phase patterns at several intermediate frames using linear interpolation (top row) and our proposed interpolant (bottom row). Notice that linear interpolation introduces severe artifacts in the wrinkle pattern: wrinkles shear and degenerate along a sharp line midway through the interpolation.

4.3 Comparison to Other Approaches

We compare our method against some baselines.

Linear Interpolation. It is natural to try to temporally interpolate between *CWF* boundary conditions by linearly interpolating the constituent variables:

$$a(t) = (1-t)a^0 + ta^1;\ \omega(t) = (1-t)\omega^0 + t\omega^1;\ \tilde{z}(t) = (1-t)\tilde{z}^0 + t\tilde{z}^1.$$

Such an interpolant lacks physical meaning and moreover does not stay within the space of *CWF*s, since there is no reason to expect $\omega(t)$ and $\tilde{z}(t)$ to stay approximately compatible during the interpolation. See Figure 10 and Figures 9, 10 in the supplemental document (§ 6) for some examples of artifacts that arise when using this linear scheme. Note that replacing linear interpolation of $\tilde{z}$ and ω with a more complicated non-linear interpolant would not help resolve this incompatibility issue.

For comparison on a more didactic example, consider the simple case of a plane wave rotating by ninety degrees, $z^0 = \exp(ix)$ and $z^1 = \exp(iy)$. On a sufficiently small patch centered at the origin, we expect interpolation to rotate the plane wave, with an interpolant resembling $z(t) = \exp(i[x\cos\frac{\pi t}{2} + y\sin\frac{\pi t}{2}])$. In Figure 8 we see that when using our proposed interpolation algorithm, the interpolant is indeed rigid rotation of the wrinkles. On the other hand, linear interpolation *shears* the wrinkles, which midway through the interpolation degenerate along a singular line.

Knöppel et al. [2015]. In their paper, Knöppel et al. propose a method to extract vertex phase information from a provided frequency one-form by solving an eigenvalue problem. They also propose a *spatial* interpolation scheme for extending phase into triangles by placing singularities at their centers as needed. While well-suited for generating individual, static geometries, Knöppel et al.'s method is not designed to give temporally coherent motion when run on a sequence of frames with smoothly-changing frequency. Indeed, if we apply their method to $\omega(t)$ (computed using Equation (34) in the supplemental document § 1.2) to generate a phase interpolant $z(t)$, we get beautiful individual frames, as expected (Figure 10) but see that the frames are discontinuous in time (see the **main** supplemental video at timestamp 03:44–04:04).

Moreover, Knöppel et al. focus exclusively on phase fields and do not discuss wrinkles with amplitude. We make a good-faith attempt to incorporate amplitude in the method (See Figure 9) by linearly interpolating the keyframe vertex amplitudes in time and then barycentrically blending them into each triangle. The resulting artifacts are due to a mismatch between where Knöppel et al. places phase singularities (at centers of triangles) and where amplitude vanishes (which can only be at vertices).

Chen et al. [2021]. In their recent work, Chen et al. describe a method for rendering high-resolution wrinkled surfaces given frequency and amplitude fields on the surface. To compare against this work, we again use Equations (12), Equation (34) (in the supplemental document § 1.2) to compute $a(t)$ and $\omega(t)$ and apply Chen et al. to each frame of the interpolation. See Figure 10 and the **Comparisons** supplemental video for the results. Unlike Knöppel et al. [2015], Chen et al.'s method does consider wrinkle amplitude, and so can produce smooth wrinkled surface geometry. However, their approach again suffers from temporal incoherence. Moreover, we observe their method also can fail to capture the sliding of singularities over the surface, instead producing degenerate, noisy phase fields; see e.g., Figure 10 around $t = 0.5$.

Keyframes from Physical Simulation. Finally, we compare the behavior of our interpolant to the results of physically simulated wrinkle evolution. We start by simulating the twisting of a 1-meter-high (diameter 0.25 meters, thickness 0.318 mm, Young's Modulus 0.821 MPa, density 472.6 kg/m^3, and Poisson's ratio 0.243), 88k-vertex cylinder mesh, by 80 degrees with C-IPC's [Li et al. 2021] open-source implementation of shell finite elements. The cylinder's top and bottom boundaries are pinned to rotate (time-stepped at 0.04 seconds/frame) in opposite directions at 10 degrees/s without gravity or external forces. To avoid simulating the transient dynamics before wrinkles appear, we initialize the cylinder slightly twisted

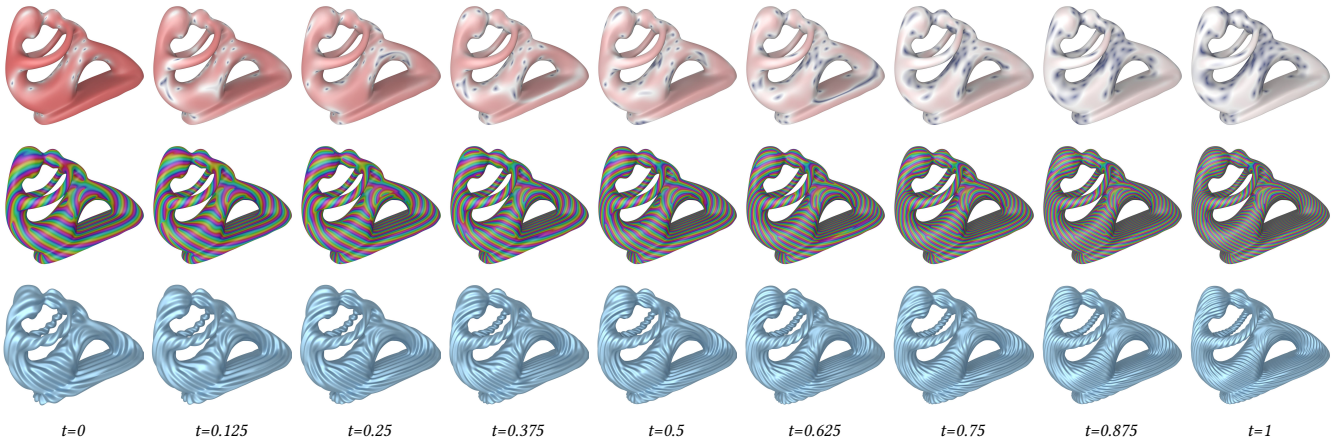

Fig. 5. *CWF* interpolation between two wrinkle patterns on the fertility model, where the second keyframe (last column) has triple the frequency and one-third the amplitude of the first keyframe (first column). We show, from top to bottom, the amplitude, phase, and rendered wrinkles for several intermediate frames during interpolation. On the amplitude plot, blue indicates $|z| = 0$ (singularities) and red indicates larger amplitude. For the animation, please check the **Interpolation Results** video in the supplementary material at timestamp 01:46–02:01.

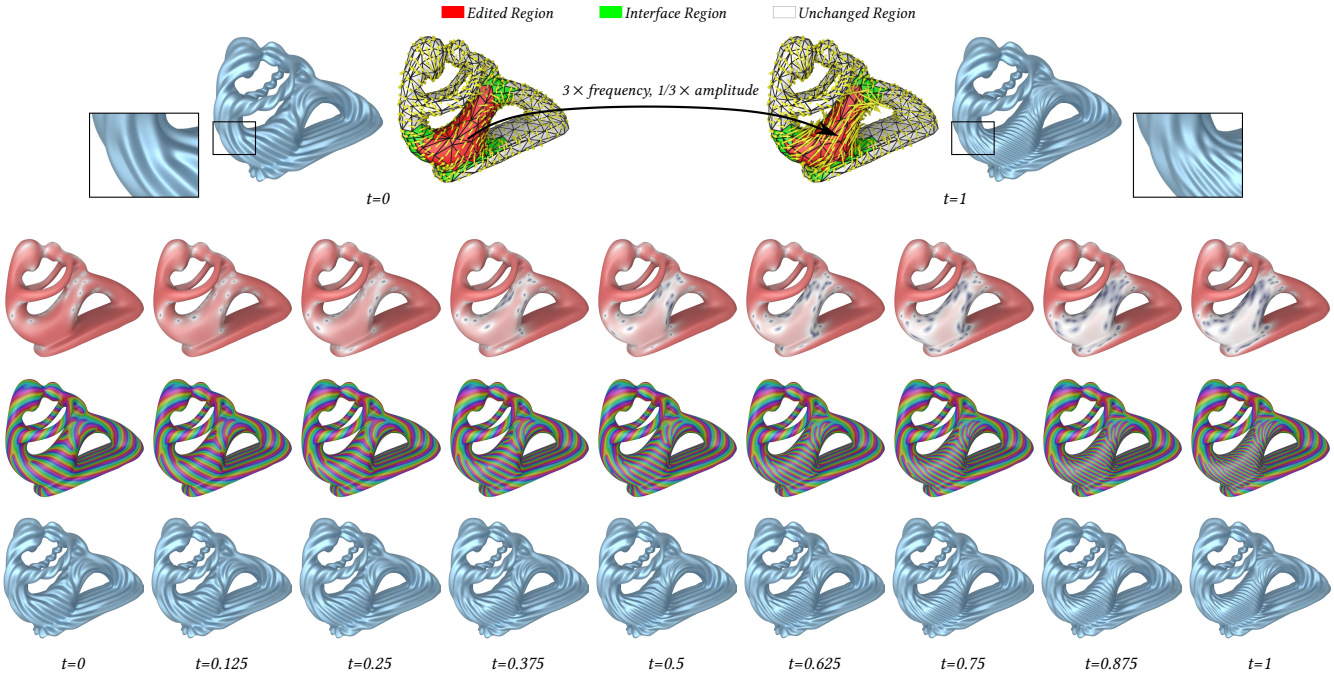

Fig. 6. *CWF* interpolation of two wrinkle patterns on the fertility model, where the second keyframe's wrinkles are triple the frequency of the first keyframe, and their amplitude shrinks by one-third on a **local** patch of surface (the red region on the mesh up top). We design an interface region (the green region; see § 6) to ensure a smooth transition between the red edited region and the unchanged white region. In the supplementary **Interpolation Results** video at timestamp 02:02–02:20, we show the corresponding wrinkle animation for a better temporal visualization.

(by 8 degrees in both direction) and run the simulation for 80 frames (so that cylinder ends are twisted by 40 degrees in both directions at the last frame). During this motion, wrinkles rotate and merge; see Figure 11, top, and the **Comparisons** supplemental video.

To compare these simulated results with corresponding intermediate frames computed using *CWF* interpolation, we take the first and last frame of the simulation as interpolation inputs. Specifically, we decimate the cylinder mesh to 688 vertices and then apply a

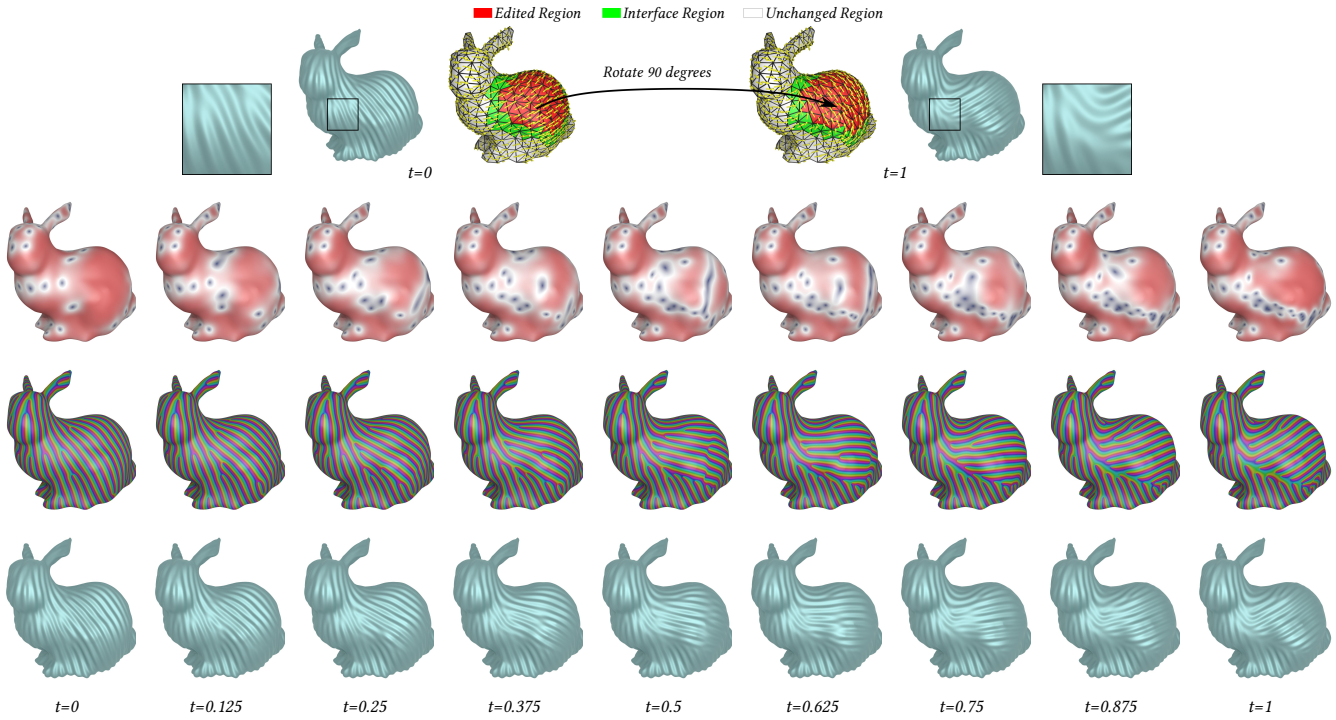

Fig. 7. *CWF* interpolation of two wrinkle patterns on the Stanford bunny, where the user has **locally** rotated the frequency field by ninety degree (red region of the top mesh). Top row: the user-specified keyframes. Second row: the upsampled amplitudes $|z| = a|\tilde{z}|$ of the *CWF*, ranging from blue at singularities ($|z| = 0$) to red where the amplitude attains its maximum. Third row: the upsampled phase of the *CWF*. Bottom row: the rendered wrinkle patterns. We see nontrivial wave mergers via singularities sliding. Please check the **Interpolation Results** video in the supplementary material at timestamp 01:29–01:45 for the corresponding animation.

tension-field-theory-based static solver [Skouras et al. 2014], implemented by Chen at al. [2021], to compute a wrinkle-free base mesh for every frame. We then manually set the frequency at the two endpoint keyframes by counting the number of wrinkles in the corresponding frames of the C-IPC simulation, and scale the wrinkle amplitude to preserve surface area of the cylinder. In Figure 11, bottom, we see that the *CWF* interpolated frames then well-align with the input keyframes and, in-between, obtain qualitatively similar merging and deformation to the simulation results. That said, these interpolated frames do have expected and notable differences from the simulated results: deformation paths differ (e.g., with wrinkles merging at different times) and, of course, the *CWF* model (as it considers only a single primary frequency) does not capture the secondary wrinkles observable in the simulation results.

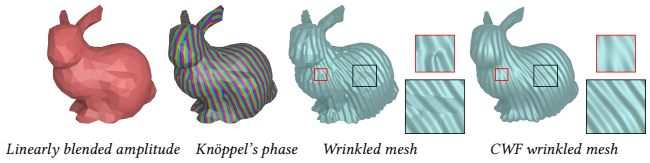

Fig. 9. Knöppel et al. [2015]'s algorithm is designed only for stripe patterns (phase fields), not wrinkles with amplitude. Naively combining the phase field produced by Knöppel et al's method with linearly-interpolated amplitude (through barycentric blending from triangle corners) yields artifacts near wrinkle singularities (the red zoomed-in region). In this case, the coarse mesh has a constant amplitude on each vertex. Moreover, without our *CWF* subdivision algorithm, meshing artifacts are apparent in the rendered result (the black zoomed-in region).

5 WRINKLE UPSAMPLING

Representing fine wrinkle patterns on coarse meshes is only useful if there is a way to actually see them. Given a triangle mesh $\mathcal{T}$ and a *CWF* on that mesh, rendering the wrinkled surface requires both upsampling $\mathcal{T}$ itself (yielding a smooth canvas on which to place wrinkles, free of meshing artifacts) and turning the *CWF* into a displacement map that can be applied to the upsampled mesh.

To that end, we propose the following upsampling strategy:

- For upsampling $\mathcal{T}$, we focus exclusively on Loop subdivision [1987], as it is the most commonly-used triangle mesh subdivision scheme.
- In the remainder of this section, we will describe a rule for mapping per-vertex complex numbers z and per-edge frequencies ω from $\mathcal{T}^k$, the input mesh after k levels of Loop

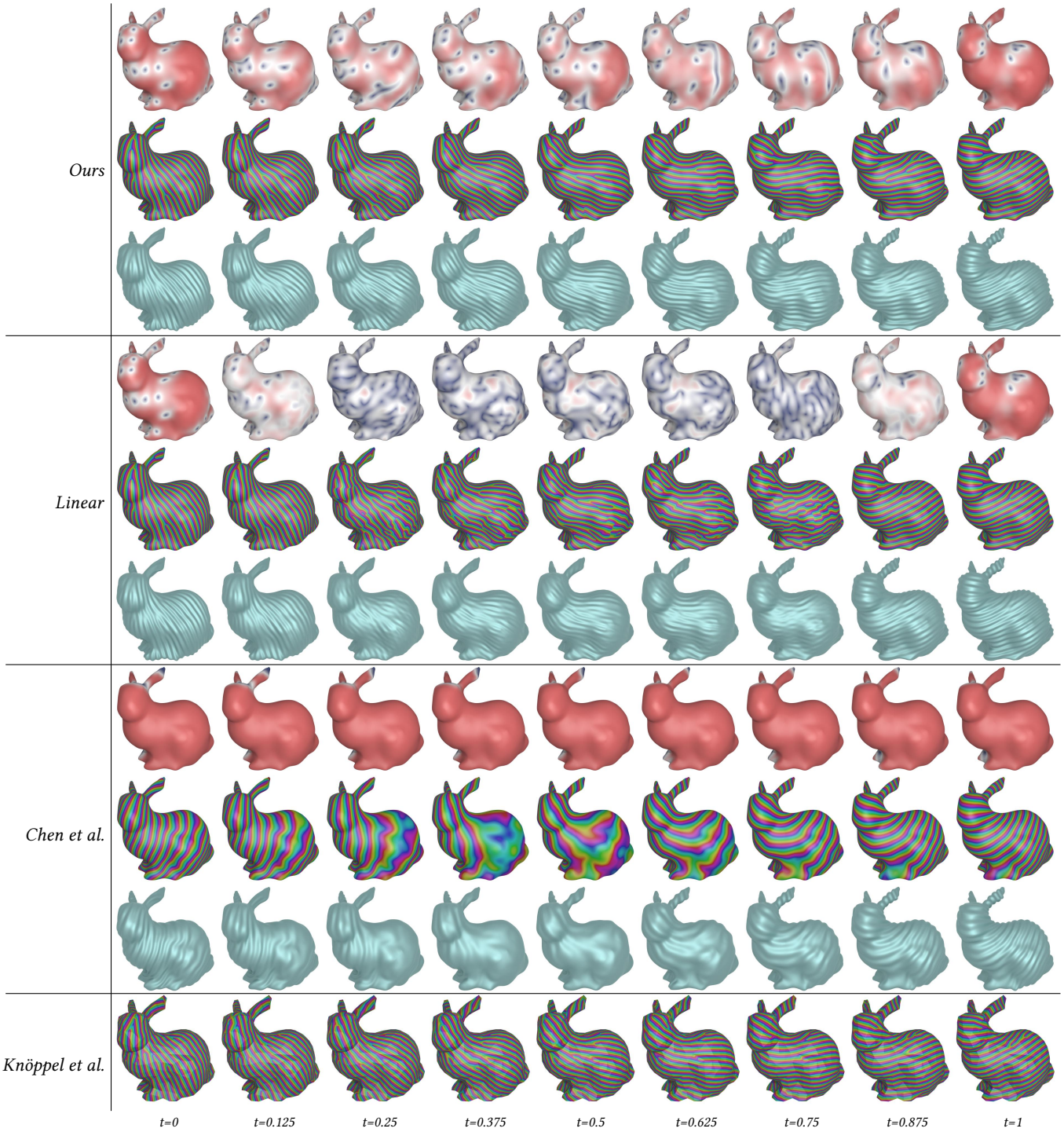

Fig. 10. We compare our interpolation algorithm to several baselines on an example of wrinkles rotating by ninety degrees on the Stanford bunny. Linear interpolation leaves the space of *CWF*s and consequently produces severe artifacts. Both Knöppel et al. [2015] and Chen et al. [2021] produce temporally incoherent results, though these discontinuities are hard to see in static frames and much more obvious in the **main** supplementary video (at 03:44–04:04) and the **Comparisons** supplementary video (at 01:00–01:56). Additionally, Chen et al.'s phase field becomes noisy midway through the interpolation. Note that the Chen et al. phase field differs from the others at $t = 1$ since their method recomputed phase from the provided frequency input.

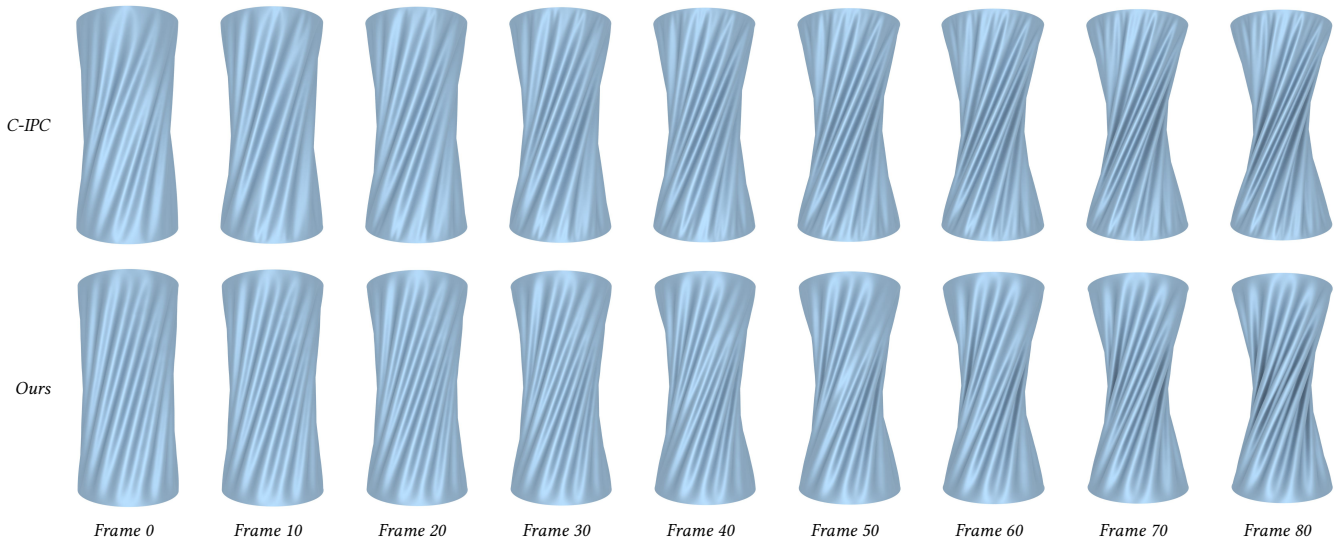

Fig. 11. We compare *CWF* interpolation to the results of physically simulated wrinkle evolution. We simulate the twisting of a cylinder by 64 degrees (from 16 to 80 degrees) with *C-IPC* [Li et al. 2021] *(top)* and compare its results with *CWF* interpolation using the the first and last simulated frame (*bottom*) as boundary conditions. *CWF* smoothly interpolates the given keyframes with qualitatively-similar wrinkle merging and deformation (though it does not capture secondary wrinkles that emerge during simulation). See the **Comparisons** supplemental video (at 02:57–03:13) for an animated comparison.

refinement, to $\mathcal{T}^{k+1}$. The quantities on $\mathcal{T}^0$ can be read directly from the *CWF* ($\boldsymbol{z}_i = a_i\tilde{\boldsymbol{z}}_i$ on the vertices), and then upsampled to any desired level of refinement.
- To materialize the wrinkled mesh at subdivision level k, we apply the displacement $|\boldsymbol{z}_i|\hat{\boldsymbol{n}}_i \cos\arg \boldsymbol{z}_i$ to vertex $\boldsymbol{V}_i$, where $\hat{\boldsymbol{n}}_i$ is the vertex normal.

We use this approach to render all smooth wrinkled surfaces in the paper (we apply 4 or 5 levels of subdivision, depending on the frequency of wrinkles in each example). We have not analyzed this process to prove that the limit surfaces belong to some smoothness class, though in practice all of our results are visually smooth, including near singularities. Although the above procedure is designed to render *CWF*s, it can also be used to visualize $\boldsymbol{z}, \omega$ pairs that do not obey approximate compatibility. We have observed that the resulting upsampled wrinkles are also smooth, although they include extraneous singularities not reflected in $\boldsymbol{z}$ or ω (see Figure 4d).

Preliminaries. We now describe the upsampling maps $\boldsymbol{z}^k \to \boldsymbol{z}^{k+1}$ and $\omega^k \to \omega^{k+1}$. To avoid cumbersome superscripts we will write $\mathcal{T}$ for $\mathcal{T}^k$ and $\mathcal{T}'$ for $\mathcal{T}^{k+1}$, and similarly use primes to denote quantities on $\mathcal{T}^{k+1}$. We write $\boldsymbol{V}$ for the vertices of $\mathcal{T}$ and S_0 for the Loop subdivision mask, so that $\boldsymbol{V}'_j = [S_0\boldsymbol{V}]_j$.

5.1 Upsampling ω

Research in geometry processing [de Goes et al. 2016b; Wang et al. 2006] has established a subdivision mask S_1 on one-forms that is compatible with the Loop subdivision mask and the discrete differential operator, in the sense that for any function f on $\mathcal{T}$, $S_1 \mathrm{d}f = \mathrm{d}S_0 f$. On closed meshes, we use this operator directly, and set $\omega' = S_1\omega$. On meshes with boundary, we modified the standard Loop formula and de Goes et al.'s formula slightly to keep boundary corners sharp (see § 2 of the supplemental document for details).

5.2 Upsampling z

There are two obvious ideas for how to upsample the $\boldsymbol{z}$ values: either by subdividing $\boldsymbol{z}$'s real and imaginary parts separately, or its magnitude and phase. Neither approach works, for essentially the same reasons discussed in § 3 as motivating the use of frequencies in addition to phase in the *CWF* representation: in regions where the frequency is high, phase values at vertices undersample the wrinkles. Computing $\boldsymbol{z}'$ by averaging these aliased samples yields an equally-aliased $\boldsymbol{z}'$. We show a comparison between our approach (described below) and these two naive baselines in Figure 12.

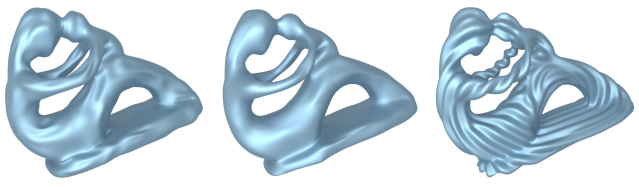

Fig. 12. A comparison of our $\boldsymbol{z} \to \boldsymbol{z}'$ upsampling algorithm (right) to naively applying the Loop subdivision mask to $\boldsymbol{z}$'s amplitude and phase (left), and to $\boldsymbol{z}$'s real and imaginary parts (middle). Neither baseline preserves the frequency of the wrinkles in the original *CWF*.

5.2.1 Our Approach. To upsample z without aliasing the wrinkles, we need to incorporate frequency information from ω into the upsampling procedure. We build up to the final formulas for computing z' on the vertices of $\mathcal{T}'$ by first examining some simpler special cases. We will use the following assumption as our guiding principle for deriving our subdivision rules: in a local patch of the surface, the *phase* of z might change quickly, but the *frequency* does not. This assumption allows to extrapolate and interpolate frequency values away from the edges on which they are defined.

Upsampling on a Line Segment. Let us first consider the case of a line segment with vertices V_0 and V_1, whose z-values are z_0 and z_1, and whose edge frequency is ω. Suppose we need to estimate z at a point $P = (1-\alpha)V_0+\alpha V_1$. Since we assume frequency changes slowly, we can extrapolate z at P from z_0 by $z(P) \approx z_0 \exp(i\alpha\omega)$; similarly from z_1 by $z_1 \exp(i(1-\alpha)\omega)$. These formula do not necessarily agree unless $|z_0| = |z_1|$ and ω and z are exactly compatible. We reconcile the two votes by barycentrically blending them:

$$z(P) = (1-\alpha)z_0 \exp(i\alpha\omega) + \alpha z_1 \exp(i(1-\alpha)\omega). \tag{18}$$

Notice that $z(P)$ interpolates z_0 and z_1.

Upsampling in a Triangle. We use a similar strategy for evaluating z at any point $P = \sum_{j=0}^{3} \alpha_j V_j$ within a triangle $\{V_0, V_1, V_2\}$, whose corresponding z values are z_0, z_1 and z_2, and whose edges have frequencies ω_{01}, ω_{12}, and ω_{20}. Similar to the line segment case, we can estimate $z(P)$ by extrapolating from a triangle corner V_j by $z(P) \approx z_j \exp\left(i\omega_{j\to P}\right)$, where

$$\omega_{j\to P} := \alpha_{j+1}\omega_{j(j+1)} + \alpha_{j+2}\omega_{j(j+2)}$$

is the frequency on the segment V_jP, as measured at vertex V_j using $\omega_{j(j+1)}$ and $\omega_{j(j+2)}$ as a basis for evaluating ω in any direction. Barycentrically blending this formula from three corners gives us:

$$z(P) = \sum_{j=1}^{3} \alpha_j z_j \exp\left[i\left(\alpha_{j+1}\omega_{j(j+1)} + \alpha_{j+2}\omega_{j(j+2)}\right)\right]. \tag{19}$$

This triangle interpolant is similar to the one proposed by Zuenko and Harders [2019] with one key difference: they formulate Equation (19) purely for phase and propose barycentrically interpolating amplitude separately. The Zuenko and Harders approach is unsuitable for triangles containing singularities, since upsampled amplitude does not neccessarily vanish at the singularites, creating visual artifacts (Figure 16).

5.2.2 Loop Subdivision Rules for z. We now generalize Equation (19) to compute z'_i on the Loop-upsampled vertex V'_i of $\mathcal{T}'$. We must consider two cases: V'_i might be a repositioned vertex already present in $\mathcal{T}$ (an *even* vertex), or an entirely new vertex that came from splitting an edge of $\mathcal{T}$ (an *odd* vertex). We separately describe how to compute z' in each case; see Figure 13 for a visual summary of all the rules.

Odd Rule. For an odd vertex, the standard Loop mask is

$$V'_j = \frac{1}{8}(V_0 + V_1) + \frac{3}{8}(V_2 + V_3)$$

on a "diamond" with common edge V_2V_3. This mask can be rewitten

$$\begin{aligned} V'_j &= \frac{1}{2}P_0 + \frac{1}{2}P_1 \\ P_0 &= \frac{1}{4}V_0 + \frac{3}{8}V_2 + \frac{3}{8}V_3 \\ P_1 &= \frac{1}{4}V_1 + \frac{3}{8}V_2 + \frac{3}{8}V_3. \end{aligned} \tag{20}$$

Notice that these formula combine barycentric sampling within a triangle (to compute P_0 and P_1) followed by sampling the midpoint of a line segment (to compute V'_j.) We can convert this interpolation into two single triangle interpolations at P_0, w.r.t. triangle $\{V_0, V_2, V_3\}$ and P_1, w.r.t. triangle $\{V_1, V_2, V_3\}$, together with line segment interpolation at V'_j, w.r.t. $\{P_0, P_1\}$. We can transform these rules for computing V'_j into rules for computing z'_j by applying the formulas from the special cases discussed above: we use Equation (19) to compute z at P_0 and P_1. Next, we would like to use Equation (18) to compute z'_j. There are two obstacles: first, we need the frequency evaluated on the line segment P_0P_1. Second, this line segment generally does not lie on $\mathcal{T}$, but rather cuts through ambient space, whereas frequencies are intrinsic to $\mathcal{T}$'s cotangent space.

We therefore modify the approach of Equation (18) to account for these differences. First, we sample ω at P_0 by interpolating the values of ω at the edges of triangle $V_0V_2V_3$ into the interior (see § 3 of the supplemental document for more details; in particular note that $\omega(P_0)$ is a cotangent vector and not a scalar). We use this frequency to extrapolate $z(P_0)$ to V'_j along the *intrinsic* edge from P_0 to P_1:

$$z_{P_0\to V'_j} = z(P_0) \exp\left(i\omega(P_0)(P_1^* - P_0)/2\right), \tag{21}$$

where P_1^* is the location of P_1 after rigidly unfolding triangle $V_1V_2V_3$ along the common edge V_2V_3 to lie in triangle $V_0V_2V_3$'s tangent plane. We compute $z_{P_1\to V'_j}$ analogously.

Then the final z'_j is the average of the corresponding contributions from P_0 and P_1:

$$z'_j = \frac{1}{2}(z_{P_0\to V'_j} + z_{P_1\to V'_j}). \tag{22}$$

Even Rule. Next, we consider an even vertex V'_0 corresponding to V_0 on $\mathcal{T}$, with neighbors $V_1, \ldots, V_n$. V'_0 has position

$$V'_0 = (1 - \alpha n)V_0 + \sum_{j=1}^{n} \alpha V_j, \tag{23}$$

where α is the traditional even-neighbor Loop weight and depends on the valence n of V_0 [Loop 1987]. As in the odd case, we can refactor Equation (23) as a convex combination of points linearly interpolated within a single triangle:

$$V'_0 = \sum_{j=0}^{n-1} \frac{1}{n} P_j, \quad P_j = (1-2\gamma)V_0 + \gamma V_{j+1} + \gamma V_{j+2}, \tag{24}$$

for $\gamma = \frac{n\alpha}{2}$. We now follow the same recipe as in the odd case: we compute $z(P_j)$ using Equation (19), and write z'_0 as the average of contributions extrapolated from each P_j.

Mesh Boundaries. For even vertices V_0 on the mesh boundary with neighbors V_1 and V_2, we can once again write the Loop subdivision mask in terms of an average of linear interpolations along mesh edges, and apply the same recipe as in the even and odd interior cases above. In this case $V_0' = (P_0 + P_1)/2$, where $P_0 = \frac{1}{4}V_1 + \frac{3}{4}V_0$ and $P_1 = \frac{1}{4}V_2 + \frac{3}{4}V_0$. The odd boundary case involves sampling z on a mesh edge and so Equation (18) can be applied directly.

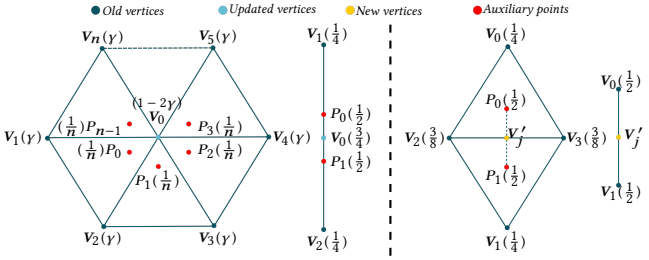

Fig. 13. Subdivision rules for even and odd interior and boundary vertices. First, sample ω and z (using Equation (19)) at the red points P_i, using the barycentric weights specified in parentheses next to the vertices. Finally, extrapolate $z(P_i)$ to the upsampled vertex (blue or yellow) using an analogous approach to Equation (21), and average these extrapolations using the weights in parentheses next to the P_i.

Remark. Notice that the upsampling strategy explained above has the following useful property: it exactly reproduces plane waves, in the sense that if $\mathcal{T}$ is a piece of the plane, ω is the differential of a linear function $\theta(P) = v \cdot P$, and $z(= \tilde{z})$ is consistent with ω, then under refinement the *CWF* $(\omega, 1, \tilde{z})$ converges to (a phase shift of) the smooth plane wave $\psi(P) = \cos(v \cdot P)$.

5.3 Results and Comparisons

We show successive upsampling of a *CWF* on the Stanford bunny in Figure 14. Wrinkles appear as soon as the subdivided mesh has high enough resolution to resolve their frequency. See all other figures of rendered wrinkles in the paper for more examples of our subdivision algorithm at work.

Triangle Interpolation Alternatives. Several alternative formulas have been proposed for interpolating phase into a triangle from samples at its corners; in Figure 15 we compare our choice of Equation (19) to several potential alternatives: the side-vertex scheme used in Jeschke et al's work [2015], and the nine-parameter Clough-Tocher cubic scheme [Farin 1986]. Notice that away from the singularities, all of these three approaches achieve reasonable results; however the side-vertex and Clough-Tocher cubic scheme fail to resolve neighborhoods of singularities. The reason for this failure is that both schemes express phase at interior points in terms of a rational or polynomial function of phase at the triangle vertices. It is mathematically impossible for such functions to produce singularities (where phase must have a branch point where it's undefined).

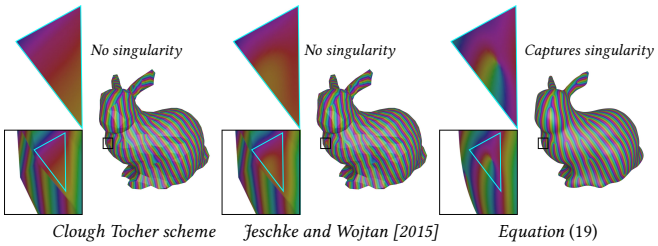

Fig. 15. A comparison where we replace Equation (18) in our upsampling algorithm with some alternatives, for a *CWF* on the Stanford bunny. From left to right: the nine-parameter Clough Tocher cubic scheme [Farin 1986], the side-vertex scheme used by Jeschke and Wojtan [2015], and Equation (19). Only Equation (19) successfully resolves the phase behavior near singularities (zoomed-in regions).

Other Upsampling Methods. In Figure 16, we compare our upsampling algorithm against two baselines from the literature [Chen et al. 2021; Zuenko and Harders 2019]. Zuenko and Harders render fine wrinkles on coarse meshes by first linearly subdividing the mesh, interpolating phase onto the fine mesh using a variant of Equation (19) and amplitude using barycentric interpolation, and then smoothing the result with SPHERIGON [Volino and Thalmann 1998] as a post-process to remove some of the artifacts from the wrinkled mesh geometry. Since they separately upsample wrinkle amplitude and phase, there is no guarantee that amplitude vanishes at phase singularities, leading to noticable noise (in the red and black zoomed-in regions, for instance). Moreover the use of linear subdivision on $\mathcal{T}$ means that the edges of the coarse mesh remain noticeable in the final rendered image, despite the post-processing.

Like in our approach, Chen et al. [2021] use Loop subdivision to upsample $\mathcal{T}$, as well as the wrinkle amplitude and phase, but their method requires that the input frequency and phase are exactly compatible. This restriction makes their approach unsuitable for rendering *CWF*s. In Figure 16 we make a best-effort comparison by projecting ω to the closest globally-integrable frequency using mixed-integer programming, but doing so severely distorts the wrinkle pattern, and many singularities are missing from the final rendered image.

6 WRINKLE DESIGN

In this section, we demonstrate applications of our *CWF* algorithms for user-based design and editing of wrinkled surfaces via interpolation. We first develop tools for the creation and editing of *CWF* keyframes and then show their application. Please also see our **main** and **Interpolation Results** videos for more details and results.

6.1 Adding Wrinkles to Surfaces

For adding wrinkles to a surface a user first provides a mesh $\mathcal{T} = \{V, E, F\}$ and selects k vertices as *source points* s (green points in Figure 17a) and at each of those points a desired wrinkle frequency vector v_s (the yellow arrow). A user then also specifies a region of influence $\text{in}_s \subset V$ for each source point (the white and red regions

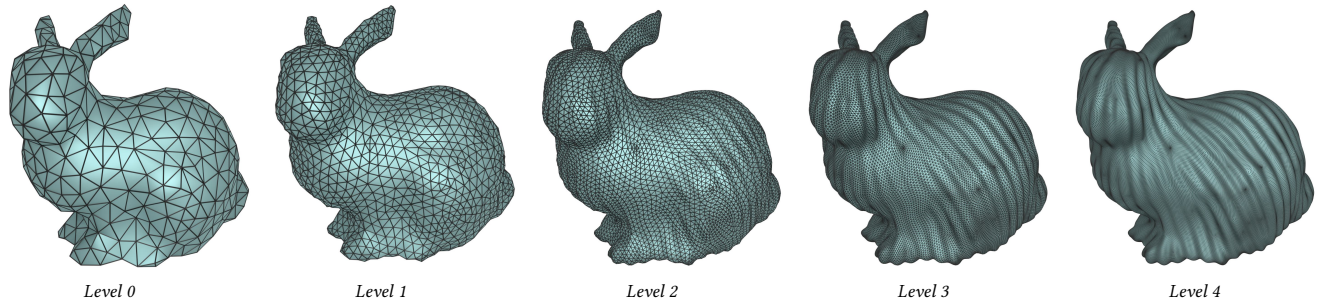

Fig. 14. We successively Loop-upsample the Stanford bunny, alongside the z and ω of a *CWF* on the bunny, and render the resulting displaced surface. After several rounds of subdivision, the wrinkle pattern appears and is stable under additional iterations of subdivision.

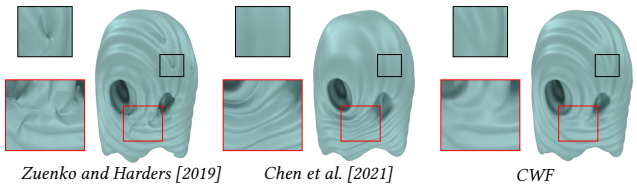

Fig. 16. We compare our upsampling algorithm against two baselines from previous work [Chen et al. 2021; Zuenko and Harders 2019]. Zuenko and Harders perform linear subdivision of $\mathcal{T}$ and separately upsample amplitude and phase in a way that produces noticeable meshing artifacts and discontinuities near the singularities (see zoomed-in regions). Chen et al. [2021]'s upsampling scheme only works for curl-free frequency fields. To use their method, we project ω onto the space of globally integrable one-forms, but doing so distorts the wrinkle pattern, which is missing many singularities.

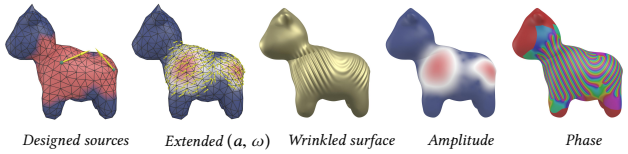

Fig. 17. Adding wrinkles to the Spot cow. From left to right: The user specifies desired wrinkle directions (yellow vectors) and regions of influence for each chosen direction (red regions). The user also assigns an amplitude to each region. We extend a and ω from the user input to the entire mesh. Finally, we solve for $\tilde{z}$ and apply our upsampling algorithm to materialize high-resolution wrinkled geometry. We show the amplitude and phase of the final wrinkle pattern, after upsampling, on the right.

in Figure 17a)[3] and a target amplitude a_s for the wrinkles within the region of influence.

Let S be the set of all source points $\{s_i\}_{i=1}^k$. We call the set of all vertices in any region of influence the *influenced vertices* $\text{in}_V = \bigcup \text{in}_{s_i}$. The influenced faces in_F are those with at least one influenced vertex, and likewise for the influenced edges in_E.

From this input we solve for a wrinkle pattern on $\mathcal{T}$. The process can be divided into two parts: (1) input extension, where we extend the user-provided amplitude and frequency samples to an a and ω on the entire mesh, and (2) wrinkle synthesis, where we compute $\tilde{z}$ from a and ω and use the upsampling technique introduced in § 5 to export the final wrinkled geometry.

We first apply the vector heat method [Sharp et al. 2019] to extend the user-provided frequencies $\boldsymbol{v}_{s_i}$ to a vertex-based vector field $\boldsymbol{v}_i$ on $\boldsymbol{V}$. Then for any influenced edge $\boldsymbol{E}_{ij} \in \text{in}_E$ we set

$$\omega_{ij} = \frac{1}{2}(\boldsymbol{v}_i \cdot \boldsymbol{E}_{ij} + \boldsymbol{v}_j \cdot \boldsymbol{E}_{ij}),$$

and for any edge that is not influenced, we set $\omega_{ij} = 0$.

We compute per-vertex amplitudes by solving

$$\arg\min_a E(a) \quad \text{s.t.} \quad \begin{cases} a_i = 0, & \boldsymbol{V}_i \notin \text{in}_V \\ a_{s_i} = c_{s_i}, & \forall s_i \in \boldsymbol{S} \end{cases} \tag{25}$$

where

$$E(a) = \frac{1}{2} a^T L a + \sum_{i=1}^{k} \sum_{V_j \in \text{in}_{s_i}} (a_j - a_{s_i})^2 A_j + \sum_{F_{\ell mn} \in \text{in}_F} \frac{a_\ell^2 + a_m^2 + a_n^2}{3} (\omega_{\ell m} + \omega_{mn} + \omega_{n\ell})^2$$

and L is the positive semi-definite cotangent Laplacian matrix, A_j is the vertex barycentric area, and $\omega_{\ell m} + \omega_{mn} + \omega_{n\ell}$ is the discrete curl. The first term promotes smoothness of a; the second term measures agreement with the user-specified amplitudes in the affected regions, and the final term couples the a to the ω to ensure that amplitude goes to 0 at the singularities of ω. Finally, for the wrinkle synthesis step, we solve the eigenvalue problem proposed by Knöppel et al. [2015] to determine $\tilde{z}$ and then directly apply the subdivision algorithm from § 5 to $(\omega, a, \tilde{z})$.

Remarks. The procedure above is used to create local wrinkle patterns. If global patterns are desired instead (such as the wrinkle patterns on the bunny in Figure 7), we first compute ω using Knöppel et al. [2013]'s algorithm, and globally scale the frequency and set a global constant amplitude to achieve the desired wrinkle characteristics. Then we find the consistent $\tilde{z}$ values on the vertices and upsample the result using the wrinkle field synthesis algorithm described above.

[3]For simplicity we compute the region of influence as the n-ring neighborhood of the chosen points, for a user-specified n; one could also compute a geodesic disk.

6.2 Editing Wrinkles

Next, we describe a tool for editing wrinkles, given an existing mesh $\mathcal{T}$ and *CWF* $(\tilde{z}^0, a^0, \omega^0)$. There are four steps in the editing pipeline:

- The user picks a region of the mesh to edit (red in Figure 7).
- The user edits the wrinkles in that region. Options we implemented are: rotating the frequency, changing the frequency magnitude, and changing the wrinkle amplitude.
- We blend the new wrinkle pattern in the edited region with the existing one inside an *interface* region around the chosen region (green region in Figure 7). Wrinkles away from the edited region and its interface do not change.
- We interpolate between the original *CWF* and the edited one, using our interpolation algorithm.

The third step requires some elaboration. Let $\boldsymbol{F}_{\text{edit}} \subseteq \boldsymbol{F}$ be the set of faces where the user has chosen to make edits, and let ω^1 and a^1 be the requested frequency and amplitude in that region. We need to construct a new $\tilde{z}^1$, which satisfies (1) $\tilde{z}^1 = \tilde{z}^0$ far away from $\boldsymbol{F}_{\text{edit}}$; and (2) $\tilde{z}^1$ is consistent with ω^1 within $\boldsymbol{F}_{\text{edit}}$. To avoid a discontinuity in frequency and amplitude on the edited surface, we create an *interface* region $\boldsymbol{F}_{\text{inter}}$ of user-specified size around $\boldsymbol{F}_{\text{edit}}$. The remaining mesh faces will not change; we call them $\boldsymbol{F}_{\text{fixed}}$.

Let $\boldsymbol{E}_{\text{edit}}$ be the edges of the triangles in $\boldsymbol{F}_{\text{edit}}$, and $\boldsymbol{E}_{\text{fixed}}$ and $\boldsymbol{V}_{\text{fixed}}$ the edges and vertices of $\boldsymbol{F}_{\text{fixed}}$, respectively. The above requirements can be written as follows:

- $\tilde{z}_j^1 = \tilde{z}_j^0$, for any $\boldsymbol{V}_j \in \boldsymbol{V}_{\text{fixed}}$.
- $|\tilde{z}_j^1| = 1$, for any $\boldsymbol{V}_j \in \boldsymbol{V} \setminus \boldsymbol{V}_{\text{fixed}}$.
- $\tilde{z}_k^1 \exp(i\omega_{kj}) - \tilde{z}_j^1 = 0$ for any edge $\boldsymbol{E}_{jk} \in \boldsymbol{E}_{\text{edit}}$.
- The interface frequency and amplitude fields are smooth.

We first interpolate ω^1 into the interface region. To get a frequency field that is as smooth as possible, we minimize the Dirichlet energy of that field with Dirichlet boundary conditions:

$$\begin{aligned} &\arg\min_{\omega} \sum_{F_{\ell mn}} (\omega_{\ell m} + \omega_{mn} + \omega_{n\ell})^2 + \sum_{\boldsymbol{V}_m} \left(\sum_{\boldsymbol{E}_{mn}} c_{mn}\omega_{mn} \right)^2 \\ &\text{s.t} \quad \begin{cases} \omega_{jk} = \omega_{jk}^0, & \boldsymbol{E}_{jk} \in \boldsymbol{E}_{\text{fixed}} \\ \omega_{jk} = \omega_{jk}^1, & \boldsymbol{E}_{jk} \in \boldsymbol{E}_{\text{edit}}, \end{cases} \end{aligned} \tag{26}$$

where c_{mn} is the cotangent weight w.r.t. edge $\boldsymbol{E}_{mn}$, and the first and second term are the discrete curl and divergence respectively.

Once we have the frequency field, the amplitude field is the optimal solution of the following modified Dirichlet energy:

$$\begin{aligned} &\arg\min_{a} \frac{1}{2} a^T L a + \sum_{F_{\ell mn}} \frac{a_\ell^2 + a_m^2 + a_n^2}{3} (\omega_{\ell m} + \omega_{mn} + \omega_{n\ell})^2 \\ &\text{s.t} \quad \begin{cases} a_j = a_j^0, & \boldsymbol{V}_j \in \boldsymbol{V}_{\text{fixed}} \\ a_j = a_j^1, & \boldsymbol{V}_j \in \boldsymbol{V}_{\text{edit}}, \end{cases} \end{aligned} \tag{27}$$

which asks for an amplitude that is as smooth as possible while ensuring (via the first term) that the amplitude vanishes at singularities. Here L is the positive semi-definite cotangent Laplacian.

Finally, we compute an approximately-consistent $\tilde{z}$ inside the interface and edited regions by solving an optimization problem generalizing the strategy from Knöppel et al. [2015]:

$$\min_{\tilde{\boldsymbol{u}}, s} E_{\text{compat}}(P\tilde{\boldsymbol{u}} + s\boldsymbol{u}^0, \omega) \quad \text{s.t.} \quad \left\| P\tilde{\boldsymbol{u}} + s\boldsymbol{u}^0 \right\|_{M_0}^2 = 1 \tag{28}$$

where $\tilde{\boldsymbol{u}}$ is a vector of size $|\boldsymbol{V}| - |\boldsymbol{V}_{\text{fixed}}|$ representing the unknown values of $\tilde{z}$ in the non-fixed region, $\boldsymbol{u}_j^0 = \tilde{z}_j^0$ for any $\boldsymbol{V}_j \in \boldsymbol{V}_{\text{fixed}}$ (with other entries zero), M_0 is the mesh barycentric mass matrix, and P is the inclusion matrix that extends $\tilde{\boldsymbol{u}}$ to the whole domain by inserting zeros for vertices in the fixed region. The single scalar value s allows for rescaling (but no other changes) within the fixed region. The inclusion of s in the optimization is required since the more straightforward restriction of Knöppel et al. [2015]'s method to the non-fixed region,

$$\min_{\tilde{\boldsymbol{u}}} E_{\text{compat}}(P\tilde{\boldsymbol{u}} + \boldsymbol{u}^0, \omega) \quad \text{s.t.} \quad \left\| P\tilde{\boldsymbol{u}} + \boldsymbol{u}^0 \right\|_{M_0}^2 = 1, \tag{29}$$

enforces an average scaling of $\tilde{\boldsymbol{u}}$ within the non-fixed region separate from, and generally inconsistent with, the scaling of $\boldsymbol{u}^0$ in the fixed region.To transform Equation (28) so that it is once again an eigenvalue problem, we make the substitutions $\boldsymbol{v} = [\tilde{\boldsymbol{u}}, s]$ and $M_0' = P^T M_0 P$. From $\tilde{\boldsymbol{u}}$ we compute $\boldsymbol{u} = P\tilde{\boldsymbol{u}}/s + \boldsymbol{u}^0$ and then set $\tilde{z}_j^1 = \boldsymbol{u}_j/|\boldsymbol{u}_j|$. Notice that $(\omega^1, a^1, \tilde{z}^1)$ is not necessarily a valid *CWF* since $\tilde{z}^1$ may not satisfy the soft compatibility condition exactly (and it is not generally possible to satisfy them without changing the $\tilde{z}_j$ within the fixed region). We can render the wrinkle field nonetheless, and use it as a keyframe for wrinkle evolution by projecting it to a *CWF* using the pre-processing described in § 4.

6.3 Results

We perform a variety of local edits to wrinkles on surfaces and visualize the wrinkle evolution during the editing process. Figures 6 and 19 illustrate local frequency increase, with a corresponding decrease in amplitude in the former example. In Figure 7 a user demands a local rotation of the wrinkles by 90 degrees, while maintaining the frequency magnitude and amplitude. We also support different types of edits on different patches. In Figure 18, we enlarge the frequency of the patch on Spot's body by 2.5 times, while at the same time rotating the direction of wrinkles on the head by 90 degrees. These examples show that we can interpolate between extreme frequency or direction differences, and the singularities nevertheless slide smoothly over the surface without temporal discontinuities or other artifacts seen in results from previous work.

Please see the **Interpolation Results** supplemental video, and the supplemental document (§ 5), for more examples of editing wrinkles, including some examples of modifying the output of wrinkles computed via physical simulation.

7 LIMITATIONS AND FUTURE WORK

In this paper we made several design decisions that could be fruitfully revisited in future work in order to generalize *CWF*s and our interpolation or upsampling algorithms, such as restricting wrinkles to waves with a single (but spatially-varying) frequency, rather than a superposition of frequencies [Rémillard and Kry 2013]; and limiting refinement to Loop subdivision. The comparison to wrinkled materials (e.g., the simulated cloth in Figure 11) indicates significant potential value in extending this work to model additional,

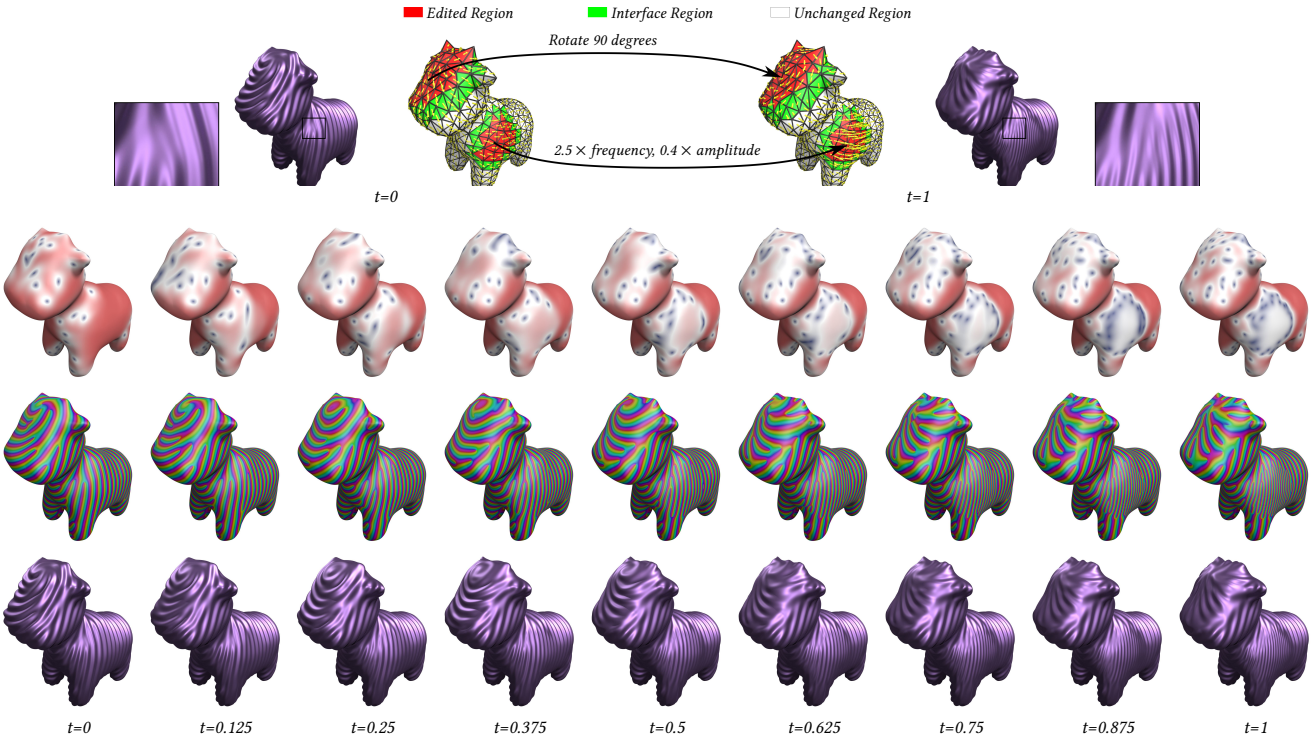

Fig. 18. *CWF* interpolation of two wrinkle patterns on Spot. Please refer to the **Interpolation Results** video in the supplementary materials for the corresponding wrinkle animation (timestamp 02:21-02:37).

secondary, higher-frequency wrinkle fields superimposed upon the first. Another particularly interesting direction of future work is to study the use of *CWF*s for solving PDEs involving waves other than the interpolation boundary value problem: for example, we believe that it's likely that cloth dynamics would benefit from *CWF* kinematics as it would allow simulation of wrinkles that have much higher frequency than the base mesh. Finally, complex wrinkle fields could be used to model skin wrinkles or fingerprints; though as argued by Zuenko et al. [2019], high-quality skin wrinkles would likely require extending *CWF*s to waves with non-sinusoidal profile.

Interpolation Performance. Interpolating *CWF*s is currently far from real-time (please refer the time table in the supplemental document for more details); our implementation took 4 minutes to compute keyframe interpolation on average, with 32 seconds minimum and 12 minutes maximum among all the examples. One problem is that our optimization problem is not a convex optimization: although the other two terms (E_{smooth} and E_{compat}) in Equation (12) are quadratic, E_{unit} is nonlinear and not convex, and this nonlinearity is enough to cause Newton's method difficulty. Performance could be improved by refactoring the optimization problem (including perhaps by improving the initial guess) or parallelizing the linear solver on the GPU. Note that the size of the linear system (in terms of number of non-zeroes) grows linearily in the number of requested frames N. To improve the efficiency, we proposed the use of guide frames (§ 4.1). We study the performance vs. quality tradeoff of the choice of number of guide frames in the supplemental document (§ 7.1); careful tuning of this parameter could achieve a bettter tradeoff than our (conservative) use of $N' = 50$.

Preventing Collisions. When amplitudes are large, it is possible for neighboring wrinkle periods to collide, or for wrinkles to "poke through" distant portions of the base mesh. Our work currently does not detect or prevent such collisions in any way.

Alternative Incompatibility Norm. When defining Opt_ω, we choose to use the $L2$ norm to measure the incompatibllity between ω and $\tilde{z}$ (Equation (2)). Given that singularies are usually sparsely distributed on a wrinkled surface, a promising alternative is the $L1$ norm, which promotes such sparsity, but is more difficult to optimize.

More General Wrinkle Parameterization. Several of our editing examples take as input wrinkle geometry computed by spectral wrinkle simulators [Chen et al. 2021; Zuenko and Harders 2019]. In these cases we could directly convert the output of the simulators into *CWF* variables. For more general wrinkled surfaces (computed by a finite element simulator [Narain et al. 2012], for instance), a missing step is how to decompose a high-resolution, *wrinkled* mesh into the coarse but *smooth* base mesh $\mathcal{T}$ and the *CWF* variables. It's unclear how to best perform this decomposition, especially when wrinkles are complex, aliased, and contain many singularities. In addition to optimization-based approaches, another promising avenue is training a neural network to perform the decomposition.

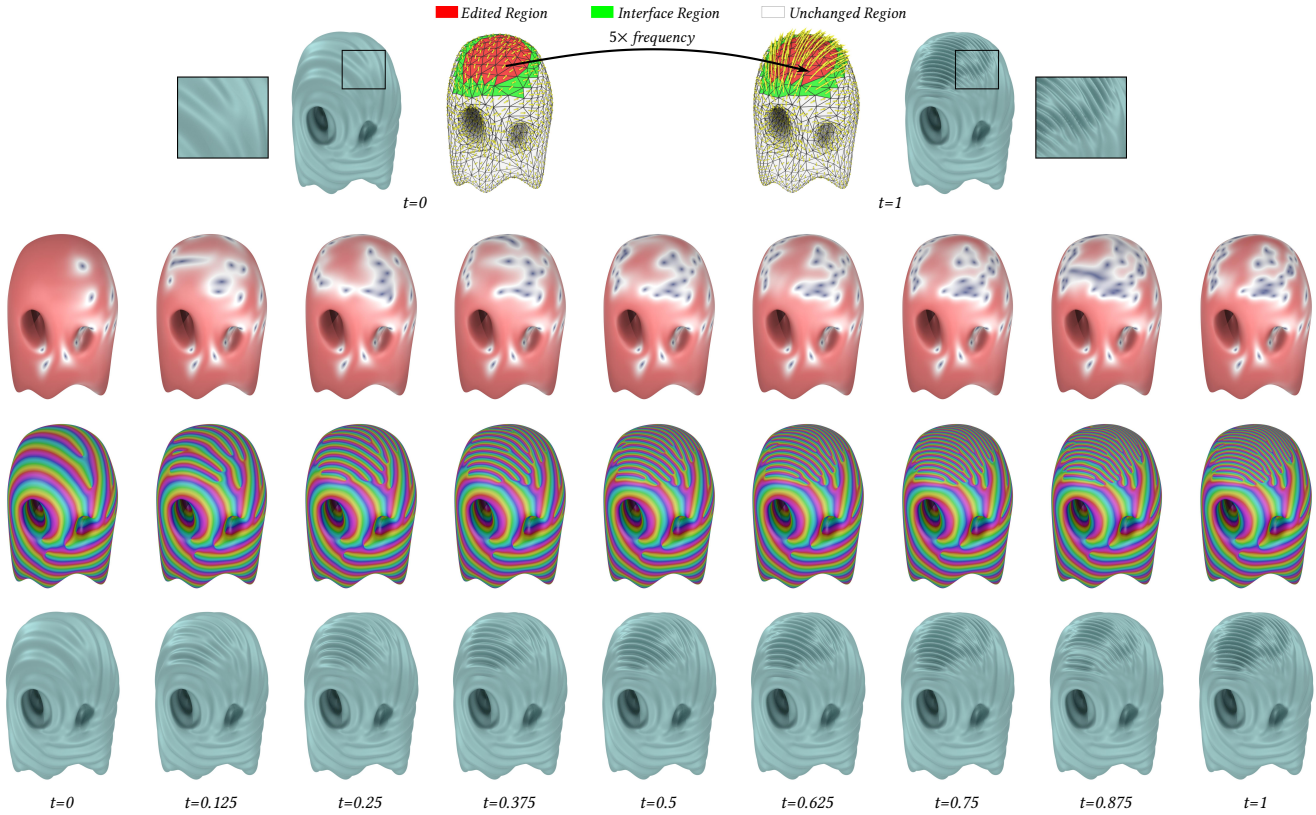

Fig. 19. *CWF* interpolation of two wrinkle patterns on the Phantasma model, where the user has **locally** increased wrinkle frequency by 5×. Our algorithm successfully generates a smooth path between these challenging keyframes. For the corresponding wrinkle animation, please watch the **Interpolation Results** video at timestamp 00:54–01:11 in the supplementary materials.

Improved Wrinkle Physics. Our interpolation is physics-inspired, but due to simplifying assumptions in the derivation such as neglecting the inertia of wrinkles and the effect of base mesh stretching on the wrinkle evolution, our interpolant is not guaranteed to exhibit fully realistic wrinkle dynamics. One possible improvement is to rederive the interpolation formula using the full reduced-order elastic energy derived by Chen et al. [2021] instead of assuming inextensible shells. Additionally, the current model interpolates the *CWF* variables completely independently of any movement or deformation of the base mesh. A scheme that jointly optimizes for a coupled base mesh and *CWF* interpolant could result in wrinkle evolution with higher physical realism. Along similar lines, our current model does not consider inhomogeneities or special features of the base mesh, such as hems or seams, which significantly affect fabric wrinkling in the real world. A possible direction for further exploration is to extend our work by imposing spatial boundary conditions on the *CWF* variables at the locations of stiff seams.

ACKNOWLEDGMENTS

We thank Rahul Narain, Chris Wojtan, Justin Solomon, Max Wardetzky, Ulrich Pinkall, Keenan Crane, and Felix Knöppel for valuable discussions while preparing this work; Peter Schröder and Fernando de Goes for their help tracking down Loop one-form subdivision code; Minchen Li for guidance compiling the IPC codebase; and Kevin Song, Xinya Zhang, and Josh Vekhter for their support and advice. We thank the maintainers of the libigl, Polyscope, C-IPC, and Eigen libraries for contributing high-quality open-source software to our community, and the National Science Foundation (IIS-1910274) and Adobe Systems for their generous support.

REFERENCES

Hillel Aharoni, Desislava Todorova, Octavio Albarran, Lucas Goehring, Randall Kamien, and Eleni Katifori. 2017. The smectic order of wrinkles. *Nature Communications* 8 (07 2017), 15809.

George Biddell Airy. 1842. Tides and waves. (1842).

Miklós Bergou, Saurabh Mathur, Max Wardetzky, and Eitan Grinspun. 2007. TRACKS: Toward Directable Thin Shells. *ACM Trans. Graph.* 26, 3 (jul 2007), 10.

Narasimha Boddeti, Yunlong Tang, Kurt Maute, David W Rosen, and Martin L Dunn. 2020. Optimal design and manufacture of variable stiffness laminated continuous fiber reinforced composites. *Scientific reports* 10, 1 (2020), 1–15.

David Bommes, Henrik Zimmer, and Leif Kobbelt. 2009. Mixed-Integer Quadrangulation. *ACM Trans. Graph.* 28, 3, Article 77 (jul 2009), 10 pages.

Guoning Chen, Vivek Kwatra, Li-Yi Wei, Charles D. Hansen, and Eugene Zhang. 2012. Design of 2D Time-Varying Vector Fields. *IEEE Transactions on Visualization and Computer Graphics* 18, 10 (2012), 1717–1730.

Lan Chen, Juntao Ye, Liguo Jiang, Chengcheng Ma, Zhanglin Cheng, and Xiaopeng Zhang. 2018. Synthesizing cloth wrinkles by CNN-based geometry image superresolution. *Computer Animation and Virtual Worlds* 29 (05 2018), e1810.

Lan Chen, Juntao Ye, and Xiaopeng Zhang. 2021. Multi-Feature Super-Resolution Network for Cloth Wrinkle Synthesis. *Journal of Computer Science and Technology* 36 (06 2021), 478–493.

Yanqing Chen, Timothy A. Davis, William W. Hager, and Sivasankaran Rajamanickam. 2008. Algorithm 887: CHOLMOD, Supernodal Sparse Cholesky Factorization and Update/Downdate. *ACM Trans. Math. Softw.* 35, 3, Article 22 (Oct. 2008), 14 pages.

Zhen Chen, Hsiao-Yu Chen, Danny M. Kaufman, Mélina Skouras, and Etienne Vouga. 2021. Fine Wrinkling on Coarsely Meshed Thin Shells. *ACM Trans. Graph.* 40, 5, Article 190 (aug 2021), 32 pages.

Keenan Crane, Mathieu Desbrun, and Peter Schröder. 2010. Trivial Connections on Discrete Surfaces. *Computer Graphics Forum* 29, 5 (2010), 1525–1533.

Bram Custers and Amir Vaxman. 2020. Subdivision Directional Fields. *ACM Trans. Graph.* 39, 2, Article 11 (feb 2020), 20 pages.

Lawrence D. Cutler, Reid Gershbein, Xiaohuan Corina Wang, Cassidy Curtis, Erwan Maigret, Luca Prasso, and Peter Farson. 2005. An Art-Directed Wrinkle System for CG Character Clothing. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '05)*. Association for Computing Machinery, New York, NY, USA, 117–125.

Fernando de Goes, Mathieu Desbrun, Mark Meyer, and Tony DeRose. 2016b. Subdivision Exterior Calculus for Geometry Processing. *ACM Trans. Graph.* 35, 4, Article 133 (jul 2016), 11 pages.

Fernando de Goes, Mathieu Desbrun, and Yiying Tong. 2016a. Vector Field Processing on Triangle Meshes. In *ACM SIGGRAPH 2016 Courses (SIGGRAPH '16)*. Association for Computing Machinery, New York, NY, USA, Article 27, 49 pages.

Olga Diamanti, Amir Vaxman, Daniele Panozzo, and Olga Sorkine-Hornung. 2014. Designing N-PolyVector Fields with Complex Polynomials. *Comput. Graph. Forum* 33, 5 (aug 2014), 1–11.

Olga Diamanti, Amir Vaxman, Daniele Panozzo, and Olga Sorkine-Hornung. 2015. Integrable PolyVector Fields. *ACM Trans. Graph.* 34, 4, Article 38 (jul 2015), 12 pages.

D. Ezuz, B. Heeren, O. Azencot, M. Rumpf, and M. Ben-Chen. 2019. Elastic Correspondence between Triangle Meshes. *Computer Graphics Forum* 38, 2 (2019), 121–134.

Gerald Farin. 1986. Triangular Bernstein-BÃľzier patches. *Computer Aided Geometric Design* 3, 2 (1986), 83–127.

Matthew Fisher, Peter Schröder, Mathieu Desbrun, and Hugues Hoppe. 2007. Design of Tangent Vector Fields. *ACM Trans. Graph.* 26, 3 (jul 2007), 10.

Russell Gillette, Craig Peters, Nicholas Vining, Essex Edwards, and Alla Sheffer. 2015. Real-Time Dynamic Wrinkling of Coarse Animated Cloth. In *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA '15)*. Association for Computing Machinery, New York, NY, USA, 17–26.

B. Heeren, M. Rumpf, P. SchrÃűder, M. Wardetzky, and B. Wirth. 2014. Exploring the Geometry of the Space of Shells. *Computer Graphics Forum* 33, 5 (2014), 247–256.

B. Heeren, M. Rumpf, M. Wardetzky, and B. Wirth. 2012. Time-Discrete Geodesics in the Space of Shells. *Computer Graphics Forum* 31, 5 (2012), 1755–1764.

Wenzel Jakob, Marco Tarini, Daniele Panozzo, and Olga Sorkine-Hornung. 2015. Instant Field-Aligned Meshes. *ACM Trans. Graph.* 34, 6, Article 189 (oct 2015), 15 pages.

Stefan Jeschke, Tomáš Skřivan, Matthias Müller-Fischer, Nuttapong Chentanez, Miles Macklin, and Chris Wojtan. 2018. Water Surface Wavelets. *ACM Trans. Graph.* 37, 4, Article 94 (jul 2018), 13 pages.

Stefan Jeschke and Chris Wojtan. 2015. Water Wave Animation via Wavefront Parameter Interpolation. *ACM Trans. Graph.* 34, 3, Article 27 (may 2015), 14 pages.

Stefan Jeschke and Chris Wojtan. 2017. Water Wave Packets. *ACM Trans. Graph.* 36, 4, Article 103 (jul 2017), 12 pages.

Felix Knöppel, Keenan Crane, Ulrich Pinkall, and Peter Schröder. 2013. Globally Optimal Direction Fields. *ACM Trans. Graph.* 32, 4, Article 59 (jul 2013), 10 pages.

Felix Knöppel, Keenan Crane, Ulrich Pinkall, and Peter Schröder. 2015. Stripe Patterns on Surfaces. *ACM Trans. Graph.* 34, 4, Article 39 (jul 2015), 11 pages.

Robert V. Kohn. 2006. Energy-driven pattern formation. *Proceedings of the International Congress of Mathematicians* (2006).

Zorah Lähner, Daniel Cremers, and Tony Tung. 2018. DeepWrinkles: Accurate and Realistic Clothing Modeling. In *ECCV*.

Minchen Li, Danny M. Kaufman, and Chenfanfu Jiang. 2021. Codimensional Incremental Potential Contact. *ACM Trans. Graph.* 40, 4, Article 170 (jul 2021), 24 pages.

Wan-chiu Li, Bruno Vallet, Nicolas Ray, and Bruno Levy. 2006. Representing Higher-Order Singularities in Vector Fields on Piecewise Linear Surfaces. *IEEE Transactions on Visualization and Computer Graphics* 12, 5 (2006), 1315–1322.

Nils Lichtenberg, Noeska Smit, Christian Hansen, and Kai Lawonn. 2018. Real-time field aligned stripe patterns. *Computers and Graphics* 74 (aug 2018), 137–149.

Ruotian Ling, Jin Huang, Bert Jüttler, Feng Sun, Hujun Bao, and Wenping Wang. 2015. Spectral Quadrangulation with Feature Curve Alignment and Element Size Control. *ACM Trans. Graph.* 34, 1, Article 11 (dec 2015), 11 pages.

Charles T. Loop. 1987. *Smooth Subdivision Surfaces Based on Triangles.* Master's thesis. University of Utah.

Rahul Narain, Armin Samii, and James F. O'Brien. 2012. Adaptive Anisotropic Remeshing for Cloth Simulation. *ACM Trans. Graph.* 31, 6, Article 152 (nov 2012), 10 pages.

Yuta Noma, Nobuyuki Umetani, and Yoshihiro Kawahara. 2022. Fast Editing of Singularities in Field-Aligned Stripe Patterns. In *SIGGRAPH Asia 2022 Conference Papers (SA '22)*. Association for Computing Machinery, New York, NY, USA, Article 37, 8 pages.

Daniele Panozzo, Enrico Puppo, Marco Tarini, and Olga Sorkine-Hornung. 2014. Frame Fields: Anisotropic and Non-Orthogonal Cross Fields. *ACM Trans. Graph.* 33, 4, Article 134 (jul 2014), 11 pages.

Joseph D. Paulsen, Evan Hohlfeld, Hunter King, Jiangshui Huang, Zhanlong Qiu, Thomas P. Russell, Narayanan Menon, Dominic Vella, and Benny Davidovitch. 2016. Curvature-induced stiffness and the spatial variation of wavelength in wrinkled sheets. *Proceedings of the National Academy of Sciences* 113, 5 (2016), 1144–1149.

Nicolas Ray, Wan Chiu Li, Bruno Lévy, Alla Sheffer, and Pierre Alliez. 2006. Periodic Global Parameterization. *ACM Trans. Graph.* 25, 4 (oct 2006), 1460–1485.

Nicolas Ray, Bruno Vallet, Laurent Alonso, and Bruno Levy. 2009. Geometry-Aware Direction Field Processing. *ACM Trans. Graph.* 29, 1, Article 1 (dec 2009), 11 pages.

Nicolas Ray, Bruno Vallet, Wan Chiu Li, and Bruno Lévy. 2008. N-Symmetry Direction Field Design. *ACM Trans. Graph.* 27, 2, Article 10 (may 2008), 13 pages.

Olivier Rémillard and Paul G. Kry. 2013. Embedded Thin Shells for Wrinkle Simulation. *ACM Trans. Graph.* 32, 4, Article 50 (jul 2013), 8 pages.

Damien Rohmer, Tiberiu Popa, Marie-Paule Cani, Stefanie Hahmann, and Alla Sheffer. 2010. Animation Wrinkling: Augmenting Coarse Cloth Simulations with Realistic-Looking Wrinkles. *ACM Trans. Graph.* 29, 6, Article 157 (dec 2010), 8 pages.

Martin Rumpf and Benedikt Wirth. 2014. Variational time discretization of geodesic calculus. *IMA J. Numer. Anal.* 35, 3 (05 2014), 1011–1046.

Igor Santesteban, Miguel Otaduy, and Dan Casas. 2019. LearningâĂŘBased Animation of Clothing for Virtual TryâĂŘOn. *Computer Graphics Forum* 38 (05 2019), 355–366.

Josua Sassen, Klaus Hildebrandt, and Martin Rumpf. 2020. Nonlinear Deformation Synthesis via Sparse Principal Geodesic Analysis. *Computer Graphics Forum* 39, 5 (2020), 119–132.

Syuhei Sato, Yoshinori Dobashi, and Tomoyuki Nishita. 2018. Editing Fluid Animation Using Flow Interpolation. *ACM Trans. Graph.* 37, 5, Article 173 (sep 2018), 12 pages.

Nicholas Sharp, Yousuf Soliman, and Keenan Crane. 2019. The Vector Heat Method. *ACM Trans. Graph.* 38, 3, Article 24 (jun 2019), 19 pages.

Mélina Skouras, Bernhard Thomaszewski, Peter Kaufmann, Akash Garg, Bernd Bickel, Eitan Grinspun, and Markus Gross. 2014. Designing Inflatable Structures. *ACM Trans. Graph.* 33, 4, Article 63 (jul 2014), 10 pages.

Justin Solomon and Amir Vaxman. 2019. Optimal Transport-Based Polar Interpolation of Directional Fields. *ACM Trans. Graph.* 38, 4, Article 88 (jul 2019), 13 pages.

Greg Turk. 1991. Generating Textures on Arbitrary Surfaces Using Reaction-Diffusion. *SIGGRAPH Comput. Graph.* 25, 4 (jul 1991), 289–298.

Amir Vaxman, Marcel Campen, Olga Diamanti, Daniele Panozzo, David Bommes, Klaus Hildebrandt, and Mirela Ben-Chen. 2016. Directional Field Synthesis, Design, and Processing. *Computer Graphics Forum* (2016).

Ryan Viertel and Braxton Osting. 2019. An Approach to Quad Meshing Based on Harmonic Cross-Valued Maps and the Ginzburg–Landau Theory. *SIAM Journal on Scientific Computing* 41, 1 (2019), A452–A479.

P. Volino and N.M. Thalmann. 1998. The SPHERIGON: a simple polygon patch for smoothing quickly your polygonal meshes. In *Proceedings Computer Animation '98 (Cat. No.98EX169)*. 72–78.

Christoph Von-Tycowicz, Christian Schulz, Hans-Peter Seidel, and Klaus Hildebrandt. 2015. Real-Time Nonlinear Shape Interpolation. *ACM Trans. Graph.* 34, 3, Article 34 (may 2015), 10 pages.

Huamin Wang. 2021. GPU-Based Simulation of Cloth Wrinkles at Submillimeter Levels. *ACM Trans. Graph.* 40, 4, Article 169 (jul 2021), 14 pages.

Ke Wang, Weiwei, Yiying Tong, Mathieu Desbrun, and Peter Schröder. 2006. Edge Subdivision Schemes and the Construction of Smooth Vector Fields. *ACM Trans. Graph.* 25, 3 (jul 2006), 1041–1048.

Andrew Witkin and Michael Kass. 1991. Reaction-Diffusion Textures. *SIGGRAPH Comput. Graph.* 25, 4 (jul 1991), 299–308.

J. Zavala-Hidalgo, M.A. Bourassa, S.L. Morey, J.J. O'Brien, and P. Yu. 2003. A new temporal interpolation method for high-frequency vector wind fields. In *Oceans 2003.*, Vol. 2. 1050–1053 Vol.2.

Eugene Zhang, Konstantin Mischaikow, and Greg Turk. 2006. Vector Field Design on Surfaces. *ACM Trans. Graph.* 25, 4 (oct 2006), 1294–1326.

Muyang Zhang, Jin Huang, Xinguo Liu, and Hujun Bao. 2010. A Wave-Based Anisotropic Quadrangulation Method. *ACM Trans. Graph.* 29, 4, Article 118 (jul 2010), 8 pages.

Meng Zhang, Tuanfeng Wang, Duygu Ceylan, and Niloy J Mitra. 2021. Deep detail enhancement for any garment. In *Computer Graphics Forum*, Vol. 40. Wiley Online Library, 399–411.

Evgeny Zuenko and Matthias Harders. 2019. Wrinkles, Folds, Creases, Buckles: Small-Scale Surface Deformations as Periodic Functions on 3D Meshes. *IEEE Transactions on Visualization and Computer Graphics* PP (05 2019).