Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2014)
Vladlen Koltun and Eftychios Sifakis (Editors)

Fast Grid-Based Nonlinear Elasticity for 2D Deformations

Rajsekhar Setaluri[1] Yu Wang[2] Nathan Mitchell[1] Ladislav Kavan[2] Eftychios Sifakis[1]

[1]University of Wisconsin-Madison [2]University of Pennsylvania

Figure 1: *Image warp of a flexible, yet incompressible, elastic sphere pinched between two fingers. The effect was created by providing spatially-varying compression resistance values to make the sphere significantly more area-preserving than its compressible background.*

Abstract

We present a deformation technique that constructs 2D warps by using spline curves to specify the starting and target shapes of selected key contours. We generate a two-dimensional deformation map from these contours by simulating a non-linear elastic membrane deforming in accordance with user-specified constraints. Although we support and demonstrate elastic models inspired by physical membranes, we highlight a custom material model for this specific application, which combines the benefits of harmonic interpolation and area-preserving deformations. Our warps are represented via a standard Cartesian lattice and leverage the regularity of this description to enable efficient computation. Specifically, our method resolves the targeting constraints imposed along arbitrarily shaped contours with sub-grid cell precision, without requiring an explicit remeshing of the warp lattice around the constraint curve. We describe how to obtain a well-conditioned discretization of our membrane model even under elaborate constraints and strict area preservation demands, and present a multigrid solver for the efficient numerical solution of the deformation problem.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling

1. Introduction

Image deformation techniques such as guided warping are valuable tools in photographic editing and visual effects production. The dramatic aesthetic effect of image deformations is evident even in early pioneering work in the field [BN92] and remains a ubiquitous tool in post-production. To a certain degree most image warping techniques are based on the concept of the image domain as an elastic membrane, which deforms subject to constraints while keeping a shape that is as natural and regular as possible. This principle has been successfully demonstrated in numerous image deformation techniques, targeting application areas such as shape manipulation [ACOL00, SMW06, JBPS11], image resizing [WTSL08], lens correction [CAA09] and video retargeting [WLSL10]. Nevertheless, several important aspects of the deformable membrane concept for image deformation have not been thoroughly explored; these include the use of nonlinear membrane models to capture complex behaviors such as local area preservation, the accommodation

DOI: 10.2312/sca.20141124

of irregularly-shaped geometric constraints within a regular discretization data structure, and the utilization of efficient multi-resolution techniques to solve the discrete equations governing the computed deformation map. Our objective is to improve the scope and utility of image deformation techniques by developing a methodology that *jointly* addresses these important algorithmic and modeling features.

Techniques which establish an affine relationship between user specified constraints and the generated image deformation have excellent potential for real-time performance and interactive manipulation [JBPS11]. Certain applications, however, benefit greatly from the inclusion of constraints that are inherently nonlinear, such as bending resistance for straight lines [CAA09] and global area preservation [WXW*06]. We specifically focus on the ability of nonlinear models to enforce (local and global) area preservation; this feature can be leveraged to efficiently generate squash-and-stretch deformations and also avoid folding artifacts by ensuring that any local region in the image does not deform into a near-zero or negative-volume configuration. Folding avoidance has previously been addressed by means of inequality constraints [Lip12, WTSL08]. Our proposed approach is to encode compression resistance directly into the material model of the simulated deformable membrane. In principle this is a soft constraint, although our method allows setting its stiffness arbitrarily without adversely affecting the numerical conditioning of the discrete governing equations even for strongly compression-resistant cases. This treatment enables graceful degradation when the user-imposed constraints make folding unavoidable, and enables tuning of deformation behaviors by specifying spatially-varying values of compression resistance (see Fig.1).

Our proposed methodology guides an image warp using user-specified contour pairs, which indicate original and target shapes of important feature lines (see Fig.1). Central to our method is our use of a standard, regular Cartesian grid to compute the deformation field; however, we do not impose any restrictions on the constraint curves, which are free to pass through the interior of our grid elements, yet are resolved at sub-grid resolution by our method. As a consequence, we do not require the constraint curves to align with edges of the discretization mesh, and therefore we do not need to perform a remeshing step when the user introduces (or adds to) the set of constraint curves. Using a regular grid to discretize the membrane dynamics promotes regularity of our data structures and provides a natural path to efficient multiresolution numerical solvers. We leverage this property by designing an efficient multigrid solver to compute the solution of the membrane deformation problem. We observe that the performance of our grid-based multigrid approach is superior to irregular meshes solved using highly-optimized sparse direct solvers [SG06]. The differences are even more significant when using multiple CPU cores, because our multigrid approach offers excellent potential for parallelization.

Importantly, the performance of our method comes without restricting the set of desirable features. In particular, our formulation can accommodate any elastic membrane model from the typical repertoire of materials surveyed by engineering disciplines. We do, however, define a custom material model, labeled Compression-Resistant Poisson Membrane (CRPM) which departs from "real-world" materials in certain ways. In particular, our model allows local tuning of the membrane's tendency for area preservation; areas of the membrane can be modeled as strongly incompressible (thus preventing inversion), while other areas can be allowed to compress without being forced to fold over. When image constraints make volume loss inevitable, this allows the membrane to gracefully compress without inverting, and when folding or inversion does occur our method is able to recover in a smooth fasion (once the constraints are lifted).

2. Previous work

Our work is related to the pioneering work of [BN92], especially the concept of guiding a warp by corresponding feature lines. A number of interpolation techniques have been adapted to the specific task of generating image-space deformations. Methods based on as-rigid-as-possible transformations [ACOL00, IMH05] aim to minimize distortion and preserve shape features. Moving least squares techniques [SMW06, WSBZ08] can accommodate a range of transformations, such as rigid, affine and similarity transforms. Thin plate splines [Boo89] have been leveraged for shape interpolation in both two and three dimensions. Bounded biharmonic weights [JBPS11] allow real-time performance and provide smoothness guarantees for the interpolation basis functions. However, linear methods trade off advanced membrane models for efficient runtime computation.

In terms of nonlinear methods, bounded distortion maps [Lip12] and locally injective mappings [SKPSH13] can guarantee inversion-free deformations; in contrast to our "soft" enforcement of area preservation they employ hard bounds. In addition [Lip12] ensures certain distortion bounds on triangle meshes. Hard constraints for inversion avoidance have been employed in the context of content-aware image resizing [WTSL08]. Other authors have incorporated constraints such as preservation of salient features [WTSL08] or 3-dimensional consistency [SD96] into the warp formulation, and employed warping techniques in applications of image and video stabilization [LGJA09].

Several authors have cast the image deformation problem as an energy minimization task [RT07, KFG09] thus avoiding folding and biasing the warp towards "natural" deformations. Terms of such an optimization functional can be adapted to preserve straight lines in re-projection tasks [CAA09], or enforce constraints at a sub-grid resolution for video retargeting [WLSL10]. Feature preservation during deformations is facilitated by using differential coordinates [Ale03], or storing relative differences between vertices and

their neighborhood [SK04] and can be complemented with terms that encourage global area preservation [WXW*06]. Optimization approaches have been applied to interpolation tasks for vector graphics images [FSH11] with attention to constraints and discontinuities.

The Finite Element Method, which represents the theoretical underpinning of our method, is relevant also in many related topics, such as creation of free form vector graphics [BBG12]. Similarly to our technique, Schiwietz and colleagues [SGW07] proposed a grid-based Finite Element Method for image deformations. Our method features the advantages of sub-grid accurate curve constraints and advanced material models. Multigrid for 3D deformations on irregular meshes has been studied by Shi and colleagues [SYBF06]. More recently, Kaufmann and colleagues [KWSH*13] study Finite Element-based image warping on adaptive meshes, exploring many different nonlinear energies and shape functions. Recent methods such as [KWSH*13] and [SKPSH13] rely on highly optimized sparse direct solvers for efficiency. Sparse direct solvers became popular in computer graphics since the thorough study by Botsch [BBK05]. However, we demonstrate that our proposed multigrid approach features superior performance on regular grids, especially when multiple CPU cores are available. Our work on elastic deformations expands on recent results on grid-based discretizations, mostly for 3D models [MZS*11, PMS12] while our treatment of constraints is motivated by recent results on grid-embedded discretizations of elliptic problems [BVBZ*10].

3. Elastic image warping

We associate an image of dimensions $L_x \times L_y$ with the rectangular domain $\Omega_0 = [0, L_x] \times [0, L_y] \subseteq \mathbf{R}^2$. Our computed image warp transforms an input point $x \in \mathbf{R}^2$ on the image to its new location $\hat{x} \in \mathbf{R}^2$. We encode this transform via a displacement map $u(x) = \hat{x} - x$, defined over the expanded domain $\Omega = [-d, L_x + d] \times [-d, L_y + d]$ obtained by padding Ω_0 with a margin of width d (of course d can be zero if desired). We typically enforce a zero-displacement constraint on the boundary of the padded image domain, i.e. $u(x) = 0$ if $x \in \partial\Omega$, therefore offering the boundary $\partial\Omega_0$ the flexibility to deform to a certain degree (depending on the width of the image margin) in order to meet any imposed constraints. Positional constraints are specified as a collection of source and target spline pairs. Each spline pair $\mathcal{S}_i(s)$ and $\hat{\mathcal{S}}_i(s), s \in [0,1]$ defines the constraint $u(\mathcal{S}_i(s)) = \hat{\mathcal{S}}_i(s) - \mathcal{S}_i(s)$; thus $\mathcal{S}_i(s)$ corresponds to the original shape of a feature curve and $\hat{\mathcal{S}}_i(s)$ defines its warped target. We summarize the constraints as $u(x) = \eta(x),\ x \in \Gamma := \cup_i \mathcal{S}_i$ where $\eta(x)$ are user-specified displacements on constraint curves.

3.1. Nonlinear membrane models

We seek to reconstruct a complete displacement field $u(x)$ that spans the entire image domain Ω, while satisfying user-specified constraints. We cast this as an optimization problem: we seek the function $u(x)$ that minimizes a regularity functional $E[u]$, subject to $u(x) = \eta(x),\ x \in \Gamma$ and $u(x) = 0,\ x \in \partial\Omega$. An intuitive form of such a regularity functional can be derived from the potential energy stored in an elastic membrane as a consequence of its deformation; we thus "stretch" a hypothetical membrane with the image content imprinted on it in such a way that the user-specified curves are displaced to their prescribed targets. For a detailed discussion on elastic material models we refer the reader to the classic textbook of Bonet and Wood [BW08] or the tutorial of Sifakis and Barbič [SB12]. In this section, we review the basic principles, with a focus on the specific material descriptions appropriate for our warping task.

An elastic membrane can undergo deformations that vary along its spatial extent. Thus, the deformation energy is typically defined locally, via an *energy density* function $\Psi(x)$ that conveys the potential energy per unit undeformed area in the vicinity of location x. Once an appropriate definition of $\Psi(x)$ has been adopted, the total membrane energy can be obtained via integration as $E = \int_\Omega \Psi(x) dx$. The energy density itself depends on the local shape of the displacement field. Common material models define the energy density as a function $\Psi(\mathbf{F})$ of the *deformation gradient* $\mathbf{F} := \partial\hat{x}/\partial x = \mathbf{I} + \partial u/\partial x$. The formula for Ψ often includes auxiliary quantities such as the volume change ratio $J = \det(\mathbf{F})$ that measures the degree of compression or expansion ($J = 1$ is a perfectly area-preserving deformation). Our method can naturally support many different energy functions; see [SB12, KWSH*13] for a list of energies typically used in graphics. We found that the following membrane model is particularly suitable for our purposes:

$$\Psi(\mathbf{F}) = \underbrace{\frac{\mu}{2}\|\mathbf{F}\|_F^2}_{\Psi_P} + \underbrace{\frac{\kappa}{2}(J-1)^2}_{\Psi_I} \tag{1}$$

The energy density in Eq. (1), which we refer to as Compression-Resistant Poisson Membrane (CRPM), does not directly correspond to a physical material, but offers a number of useful properties. First, $\Psi(\mathbf{F})$ is rotation-invariant, which means that it can support large rotations without artifacts. The term Ψ_P can be recognized as the energy equivalent of a harmonic membrane, i.e. one where the displacements satisfy a Laplace equation $\Delta u = 0$. Note that the Ψ_P term alone leads to harmonic coordinates [JMD*07], relying completely on the boundary conditions to produce a non-trivial solution. Intuitively, the Ψ_P term represents an elastic membrane that wants to shrink to zero area, but is not permitted to do so due to the fixed boundary. The Ψ_I term is an adjustable control that favors element deformations that preserve area. In contrast to the popular as-rigid-as-possible concept [IMH05], the Ψ_I term does not penalize all non-rigid deformations and encourages natural squash-and-stretch behavior. By adjusting the parameters μ and κ we can produce behaviors ranging from an extremely taut

material, to a moderately compressible or even highly incompressible membrane. This allows us to avoid folding or wrinkling behaviors by adjusting the tension/compressibility of the material, either for the entire image or for particular regions. In contrast to incompressible models that employ hard inversion barriers [SKPSH13], our newly introduced CRPM model enjoys the following beneficial properties: it remains well defined under folding or inversion (which may be desired by the user), its energy has continuous gradients throughout the configuration space, and the gradient and Hessian of the energy function can be computed with minimal effort as shown in section 3.4. These properties enable an efficient multigrid procedure, described in section 3.5.

3.2. Grid discretization and constraints

We discretize the elastic membrane models of the previous section on regular Cartesian grids. As a consequence, the total membrane energy is given by the sum of the energies associated with each grid cell Ω_k as $E = \sum_k E_k = \sum_k \int_{\Omega_k} \Psi(x)dx$. Ultimately, the energy E_k associated with each grid cell will be expressed as a function of the displacements $u_{00}^{(k)}, u_{10}^{(k)}, u_{01}^{(k)}, u_{11}^{(k)}$ discretely sampled at the four cell vertices. Given this formulation, the question arises how constraints of the form $u(x) = \eta(x),\ x \in \Gamma$ can be resolved. This would be trivial if the deformation u was sampled on an irregular mesh that was specifically generated to conform to the shape of constraint curves; mesh vertices along the constraint curve would be explicit degrees of freedom and could be directly constrained. In our Cartesian grid approach, a different constraint strategy must be followed. A seemingly natural option would be to collect a discrete sampling of points along the constraint curve, embed them in the lattice (as linear combinations of nodes), and enforce the displacement constraints only at those locations. Unfortunately, this approach would be highly sensitive to the number of points generated along the constraint curve: too few points introduce the possibility of continuity artifacts by letting the material bulge through the curve. Too many points present the danger of generating an over-constrained problem that would not have a solution.

To avoid these difficulties, we propose to enforce displacements at sub-grid resolution by satisfying constraints *on average* within each cell. Given discrete displacement values at the vertices of a cell Ω_k, we can reconstruct a continuous displacement field $\tilde{u}^{(k)}(x)$ using bilinear interpolation as $\tilde{u}^{(k)}(x) = \sum_{i,j=0}^{1} u_{ij}^{(k)} \mathcal{N}_{ij}(x)$, where $\mathcal{N}_{ij}(x)$ is the bilinear shape function associated with each vertex. Enforcing the displacement constraint $\tilde{u}^{(k)}(x) = \eta(x)$ on average within cell Ω_k yields the following condition:

$$\underbrace{\int_{\Gamma_k} \eta(x)dx}_{b_k} = \int_{\Gamma_k} \tilde{u}^{(k)}(x)dx = \sum_{i,j} u_{ij}^{(k)} \underbrace{\int_{\Gamma_k} \mathcal{N}_{ij}(x)dx}_{w_{ij}^{(k)}} \qquad (2)$$

where $\Gamma_k = \Gamma \cap \Omega_k$. This condition can be summarized by flattening the four nodal displacements into a combined vector $U^{(k)} = (u_{00}^{(k)}, \ldots, u_{11}^{(k)}) \in \mathbf{R}^8$ and writing the above constraint equivalently as $\mathbf{W}_k U^{(k)} = b_k$, where $\mathbf{W}_k$ is a 2×8 matrix. This averaged constraint approach is inspired by the treatment of embedded Dirichlet conditions in 2D Poisson problems by Bedrossian et al [BVBZ*10], where they demonstrate that such constraints are compatible with an energy minimization problem in the sense that they do not lead to over-constrained systems, yet they achieve second order convergence to the continuous formulation, under refinement. However, their overall methodology is notably different from ours, as they apply this technique to boundaries surrounding the active domains, not curves that are interior to them, and they modify the energy term to reflect the fact that the active domain is bounded by the boundary curve.

3.3. Soft & hard constraints – saddle point formulation

Let us assume a given set of constraint functions $C_1, \ldots, C_m$. For example, the spline constraints discussed in the previous section correspond to $C_k(U) = \|b_k - \mathbf{W}_k U^{(k)}\|_2$, but let us first tackle constraints in full generality. Typically, we would proceed by enforcing $C_1(U) = 0, \ldots, C_m(U) = 0$ exactly by using Lagrange multipliers (hard constraints). Another common strategy is to formulate a modified objective:

$$\tilde{E}(U) = E(U) + \sum_{i=1}^{m} \frac{d_i}{2} C_i(U)^2 \qquad (3)$$

where $E(U)$ is the original energy and $d_i > 0$ are weights of the individual constraints. This formulation corresponds to pursuing the constraints in a soft sense. Note that if all $d_i \to \infty$, we would converge to the former case of hard constraints. While soft constraints offer additional flexibility, the drawback is that very high weights d_i lead to stiff systems, adversely affecting the conditioning of the optimization problem. Special numerical techniques such as the Augmented Lagrangian Method have been developed to alleviate this issue [NW06]. We propose a novel formulation that allows us to *smoothly* transition from soft to hard constraints while avoiding the trap of ill-conditioning. Specifically, we propose to consider the following modified energy function:

$$\hat{E}(U, p) = E(U) + \sum_{i=1}^{m} \alpha_i p_i C_i(U) - \sum_{i=1}^{m} \frac{\alpha_i^2 p_i^2}{2 d_i} \qquad (4)$$

where E, d_i are as before, p_i are auxiliary parameters and α_i are arbitrary scaling coefficients which we can use to balance the magnitude of the individual components in order to achieve good numerical conditioning. This is analogous to improving the conditioning of a linear system by careful row/column scaling, while leaving the solution unaffected. Note that by setting all $d_i := \infty$, the last term falls out and $\hat{E}$ becomes a classical Lagrangian, with p assuming the role of Lagrange multipliers. However, our augmented energy $\hat{E}$ still allows us to specify finite d_i, corresponding to soft

constraints. The key property is that the critical points of $\tilde{E}$ (Eq. 3) correspond to the critical points of $\hat{E}$ (Eq. 4). By direct differentiation, we can easily show that if (U^*, p^*) is a critical point of $\hat{E}$, then U^* is a critical point of $\tilde{E}$ (the supplemental material gives a formal proof). We call this a "saddle point formulation" because even for a convex energy E, the critical points (U^*, p^*) of $\hat{E}$ will be saddle points due to the last term which involves negative p_i^2. This is the cost we need to pay for the arbitrary scaling factors α_i, used to obtain a well-conditioned problem. Note that the equivalence of the critical points of $\tilde{E}$ and $\hat{E}$ does *not* depend in any way on the particular values of α_i. In section 3.5 we discuss how the resulting indefinite systems can be solved efficiently.

Let us now discuss the specific form of Eq. 4 for our CRPM energy. There are, in fact, two types of stiff terms in the energy $E(u)$ which could lead to problematic numerical conditioning. The first has to do with the incompressibility term Ψ_I, which becomes problematic when a large value for κ is used. The second contributing factor would be the targeting terms $\frac{d_k}{2}\|b_k - \mathbf{W}_k U^{(k)}\|_2^2$ introduced in the beginning of this section, especially if their coefficients are set to high values to emulate hard constraints. In essence, we treat both of these categories of stiff terms using the same saddle point formulation. Two different sets of auxiliary paramters will be introduced: The first of them (p) is associated with incompressibility constraints, and forms a field that spans the entire image; specifically, we associate one parameter for every cell of the grid. The second set of parameters (q) relates to targeting constraints, and is localized around the spline curves guiding the deformation; here, we associate two parameters (one for each of the x, y coordinates) with every cell that is intersected by the spline targeting curve. Let us denote the set of all grid cells as $\mathcal{G}$ and the set of grid cells which intersect a source spline curve as $\mathcal{H}$. Typically, $\mathcal{H}$ is only a small subset of $\mathcal{G}$. Using this notation, the final form of our saddle point formulation of the CRPM energy is:

$$\hat{E}(U,p,q) = \sum_{i\in\mathcal{G}} \hat{E}_i^{(1)}(U,p) + \sum_{i\in\mathcal{H}} \hat{E}_i^{(2)}(U,q) \qquad (5)$$

where

$$\hat{E}_i^{(1)}(U,p) = \frac{\mu_i}{2}\int_{\Omega_i} \underbrace{\|\mathbf{F}\|_F^2}_{\Psi'_P} dx + \alpha_i p_i \int_{\Omega_i} \underbrace{[J(x)-1]}_{\Psi'_I} dx - \frac{\alpha_i^2}{2\kappa_i} p_i^2$$

$$\hat{E}_i^{(2)}(U,q) = \beta_i q_i^T (b_i - \mathbf{W}_i U^{(i)}) - \frac{\beta_i^2}{2d_i}\|q_i\|_2^2$$

Note that p_i is a scalar and q_i a 2D vector. Our examples use constant values of α_i, β_i, d_i across the entire domain; however, in some examples we vary μ_i, κ_i using an image mask.

3.4. Numerical approximation and solution

Before we can find the critical points of Eq. (5) using a numerical optimization algorithm, we need to define appropriate discrete approximations of the two continuous integrals. For brevity of notation, we shall use the nodal *deformation* values $\hat{x}_{ij} = \hat{u}_{ij} + x_{ij}$, with the understanding that those are a simple transformation of the displacement values in U. The first term (Ψ'_P) can be integrated exactly by computing a continuous expression of the deformation gradient $\mathbf{F}$ from the bilinearly interpolated displacement values within the cell. The final result is:

$$\int_{\Omega_k} \|\mathbf{F}\|_F^2 dx = \frac{1}{2}\Big(\|\hat{x}_{10}-\hat{x}_{00}\|_2^2 + \|\hat{x}_{11}-\hat{x}_{01}\|_2^2 + \|\hat{x}_{01}-\hat{x}_{00}\|_2^2 + \|\hat{x}_{11}-\hat{x}_{10}\|_2^2\Big) \qquad (6)$$

The analytic calculation of this expression is facilitated by the fact that the interpolated deformation field $\tilde{u}^{(k)}(x)$ is a bilinear function; as a consequence, its gradient $\mathbf{F}(x)$ is a *linear* function of nodal displacements within each cell Ω_k. Note that this convenient property is specific to 2D, while in 3D the trilinear nature of $\tilde{u}^{(k)}(x)$ would lead to a deformation gradient with entries that could include products of *two* of the monomials x, y and z. The same convenient property can be used to compute Ψ'_I exactly, by leveraging the fact that the term $J-1 = \det(\mathbf{F})-1$ includes only up to bilinear terms and can thus be integrated exactly using a single quadrature point at the center of Ω_k. The final discrete expression is:

$$\int_{\Omega_k}[J(x)-1]dx = \text{Area}(\Omega_k)[\det(\mathbf{F}_*)-1], \text{where} \qquad (7)$$

$$\mathbf{F}_* = \tfrac{1}{2h}\begin{pmatrix} \hat{x}_{10}+\hat{x}_{11}-\hat{x}_{00}-\hat{x}_{01} & \hat{x}_{01}+\hat{x}_{11}-\hat{x}_{00}-\hat{x}_{10} \end{pmatrix}$$

In the last expression, $\mathbf{F}_* \in \mathbf{R}^{2\times 2}$ denotes the deformation gradient evaluated at the cell center, h is the grid spacing, and $\hat{x}_{ij}$ are interpreted as column vectors. The simplicity of the CRPM material was essential in deriving these closed form expressions. For more general materials such as Neohookean membranes, we would have to resort to numerical quadrature to obtain a discrete energy approximation [PMS12]; this is one of the benefits of our proposed CRPM energy.

Having established the discrete form of Eq. (5), the next step is to find a critical point close to the initial guess. Because our CRPM energy is of an order higher than quadratic, we employ an iterative Newton-Raphson procedure. Interestingly, the energy from Eq. 5 is only one order away from quadratic; the only order three term comes from the determinant J (quadratic in 2D) multiplied by the auxiliary variables p. Consequently, the calculation of the Hessian matrix can be optimized by taking advantage of the fact that many terms do not change between steps. The Hessian matrix is symmetric, but indefinite by construction. Therefore, we can use an appropriate Krylov subspace algorithm such as MINRES [PS75] for an iterative solution. The following section outlines an even more efficient multigrid solution process.

3.5. Multigrid solution

The regularity of our grid-based discretization combined with good numerical conditioning produced by the saddle point formulation (Eq. 5) have created favorable conditions

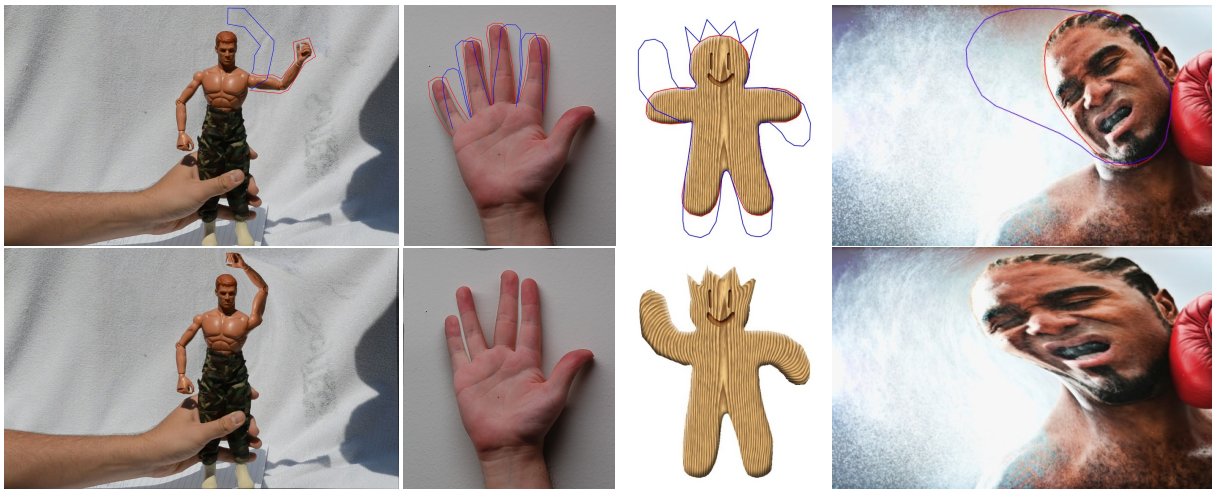

Figure 2: *Warp examples, from left to right: A toy figure is warped to articulate the arm in a different pose (a), A hand pose is adjusted to increase the separation of some fingers, while bringing others closer together (b), Woody demonstrates large rotations and sharp non-smooth target curves (c), Exaggerated squash-and-stretch deformation of a boxer face is facilitated by our incompressibility constraints (d).*

for using a multigrid solver [TOS01] which has the potential to have a solution cost that scales linearly with the total number of degrees of freedom (i.e. grid nodes). However two special circumstances, namely the indefiniteness of the Hessian matrix used in the Newton-Raphson iteration (resulting from the saddle-point formulation), and the embedded enforcement of constraints necessitate certain modifications to a classical multigrid approach. We start by reviewing the fundamental methodology, and detail the interventions necessitated by our particular application. For brevity of notation, we use the single variable u which concatenates all discrete degrees of freedom including nodal displacements (U), the auxiliary parameters associated with the incompressibility constraints (p) and their counterparts associated with the embedded spline constraints (q). We also write the entire Newton-Raphson system as $\mathbf{L}u = f$, where $\mathbf{L} = \partial^2 \hat{E}(u)/\partial u^2$ will be the Hessian of the energy in Eq. (5) and $f = -\partial \hat{E}(u)/\partial u$ is the negated gradient.

The general formulation of the Multigrid V-Cycle we apply to this problem is given in Algorithm 1. We proceed to describe all the individual algorithmic building blocks.

Operator hierarchy. We utilize a sequence of discrete approximations of the operator $\mathbf{L}$, denoted as $\mathbf{L}^h, \mathbf{L}^{2h}, \mathbf{L}^{4h}$, corresponding to different grid resolutions. In our case, constructing these operators is straightforward, since we can simply discretize the elasticity potential at any given grid resolution, and recompute the per-cell embedded spline constraints without altering the control splines themselves.

Smoother. The multigrid algorithm employs a *smoother*, or *relaxation* routine. This is an iterative algorithm designed to reduce the residual of the discrete equations (in some appropriate norm) after repeated application. The exact variant of a smoother is dependent on the numerical properties of the operator $\mathbf{L}$; if this was a symmetric and positive definite matrix, Gauss-Seidel is the standard option, or a Damped Jacobi iteration may be preferred for reasons of parallelism. In our case, however, $\mathbf{L}$ is a symmetric yet indefinite matrix. As a consequence, Gauss-Seidel or Jacobi iteration are not guaranteed to converge. Instead, we employ the Kaczmarz method [TOS01] which effectively performs a change of variable $u = \mathbf{L}^T y$ to convert the indefinite (and possibly non-symmetric) system $\mathbf{L}u = f$ into the symmetric positive definite equivalent $\mathbf{L}\mathbf{L}^T y = f$. The Kaczmarz technique lever-

Algorithm 1 Pseudocode for a V-Cycle of the Multigrid Correction Scheme for the solution of the linear equation $\mathbf{L}u = f$

procedure MG_VCYCLE($u_0, f, \mathbf{L}$) ▷ u_0: initial guess
  $u^h \leftarrow u_0, b^h \leftarrow f$ ▷ total of $L+1$ levels
  for $l = 0$ **to** $L-1$ **do**
    Smooth($\mathbf{L}^{2^l h}, u^{2^l h}, b^{2^l h}$)
    $r^{2^l h} \leftarrow b^{2^l h} - \mathbf{L}^{2^l h} u^{2^l h}$
    $b^{2^{l+1} h} \leftarrow$Restrict($r^{2^l h}$)
    $u^{2^{l+1} h} \leftarrow 0$
  end for
  Solve $u^{2^L h} \leftarrow (\mathbf{L}^{2^L h})^{-1} b^{2^L h}$ ▷ Using non-MG solver
  for $l = L-1$ **down to** 0 **do**
    $u^{2^l h} \leftarrow u^{2^l h} +$Prolongate($u^{2^{l+1} h}$)
    Smooth($\mathbf{L}^{2^l h}, u^{2^l h}, b^{2^l h}$)
  end for
  $\phi \leftarrow u^h$
end procedure

ages the fact that Gauss-Seidel or Damped Jacobi (with appropriate damping parameter) can be guaranteed to converge on the modified system. In fact the change of variable never takes place explicitly; its effect is instead emulated by updating the primary variable u directly. We use a Damped Jacobi iteration for smoothing the entire domain, and additionally sweep regions near position constraints with a Gauss-Seidel traversal for a number of additional sweeps per smoother iteration. Both algorithms are summarized below.

```
procedure KaczmarzGaussSeidel(A,x,b)
    for i = 1 to N do                          ▷ total of N equations
        δ ← (b_i − a_i^T x)/‖a_i‖_2^2          ▷ a_i is the i-th row of A
        x ← x + δ a_i
    end for
end procedure
procedure KaczmarzDampedJacobi(A,x,b,δ,ω)
    for i = 1 to N do                          ▷ total of N equations
        δ_i ← (b_i − a_i^T x)/‖a_i‖_2^2        ▷ a_i is the i-th row of A
    end for
    x ← x + ωAδ                                ▷ ω ∈ (0,1] is the damping factor
end procedure
```

Transfer operators. Communication of data between levels of the multigrid hierarchy is performed by two subroutines, the *prolongation* operator and the *restriction* operator. Prolongation is essentially an upsampling operator; once a correction has been produced by a coarse level of the multigrid hierarchy, we *prolongate* it to the immediate finer resolution and incorporate it into our current iterate. Conversely, the restriction operator downsamples the residual of the equations of a given resolution, to produce right-hand-sides for the equations of the immediate coarser level. In our case, these transfer operations need, in principle, to be applied to all discrete variables $u = (U, p, q)$, and the residuals of their respective equations. As far as the positional degrees of freedom (U) and the incompressibility-associated augmented variables (p) this process is intuitive and straightforward; all of these variables correspond to 2D spatial fields and have a multi-resolution nature. For example, when a converged solution has been achieved, each p_k associated with a given grid cell Ω_k equals $p_k = (\kappa/\alpha)(J_k - 1)$, i.e. it measures the deviation of the cell volume from its undeformed value; if the p values of a 2×2 block of cells are averaged, the result will be the same measure evaluated on their parent cell, on the immediate finer resolution. Thus, for both U and p variables, we employ geometrically-inspired transfer operators. The prolongation of U is performed by bilinear interpolation, while p is prolongated by piecewise-constant upsampling. The respective restriction operators are their adjoint (transpose) operators.

The situation is more complicated with the degrees of freedom q associated with the embedded positional constraints. In contrast with U and p which form 2D fields, q resides on a lower dimensionality subdomain (along cells intersected by the boundary). Multigrid theory dictates that a proper restriction/prolongation of these values should happen along their native domain. A further complication arises from the fact that different cells intersect the constraint curve for different arc lengths, and consequently variables and their associated equations are "unevenly" weighted. As a consequence, proper restriction/prolongation would need to employ intricate weighting schemes. Instead of crafting intricate transfer operators that cope with these issues, we employ a simpler (and theoretically very well established, see [TOS01]) alternative. Multigrid theory suggests that good convergence can be preserved even *without* prolongating/restricting such variables, if we can devote extra smoothing effort in their vicinity (i.e. near spline constraints). Intuitively, the reason why this approach works can be described as follows: if we have intensively smoothed equations in regions associated with q variables, the residuals of their equations are near-zero. Thus, irrespective of the weights that a transfer operator would use, the restricted value would also be near-zero. By smoothing with higher intensity near constraint curves we can exclude these variables from inter-grid transfer completely. The reason why this additional smoothing is practically tractable, is that the measure of the region that requires extra smoothing asymptotically vanishes under refinement, when we transition to higher resolutions. In practice, our aggregate smoother routine sweeps 5 times near the constraint curve, then iterates twice across the entire domain, and concludes with 5 additional sweeps near constraint-intersected cells again. The boundary smoothing effort typically dominates for resolutions lower than 32^2, but becomes less and less significant as we transition to more refined grids.

4. Results

We tested our technique on a number of images (figure 2), observing that our CRPM energy typically results in quite intuitive behavior. As can be seen in the toy figure example in figure 2 and the roundabout in figure 6, our method is robust to large rotations, where linear methods may exhibit folding artifacts (figure 6). The combination of harmonic membrane with incompressibility results in natural squash-and-stretch effects, as demonstrated in figures 1 and 2 (boxer face). Our embedded sub-grid spline constraints allow for accurate targeting, as shown in 4.

We compare our grid-based multigrid solver to recent image deformation techniques that employ irregular meshes and sparse direct solvers [KWSH*13, SKPSH13]. General triangle meshes offer higher flexibility by allowing us to place vertices and edges at particular places in the image, making our embedded constraints unnecessary. On the flip side, if the constraint locations or the resolution change, it is necessary to recompute the triangulation. This exposes this approach to the danger of undesired discontinuities ("popping"). Our constraints discussed in section 3.2 are simply

Grid size	800×800	400×400	300×300	200×200	150×150	100×100	50×50
Multigrid time (six cores) [s]	0.832	0.271	0.178	0.096	0.076	0.055	0.029
Multigrid time (one core) [s]	4.19	1.37	0.932	0.544	0.37	0.233	0.11
Multigrid memory [MB]	85	21	12	5.3	3	1.3	0.3
Number of mesh vertices	640192	155752	86610	40999	21690	11316	2517
PARDISO time (six cores) [s]	5.15	1.34	0.689	0.381	0.229	0.168	0.119
PARDISO time (one core) [s]	10.29	2.48	1.30	0.615	0.341	0.216	0.123
PARDISO memory [MB]	762	190	100	43.4	21.1	10.5	1.81

Table 1: *Performance comparison between our multigrid solver on a regular grid and sparse direct solver (PARDISO) on an irregular triangle mesh with approximately the same number of degrees of freedom.*

smooth splines, oblivious to the fact that we are working with a discretized deformation field. Also, construction of well-behaved meshes is a non-trivial task which typically results in dependency on external software packages [She96]; construction of regular grids is trivial. However, the chief advantage of our multigrid approach is performance, studied in the following example.

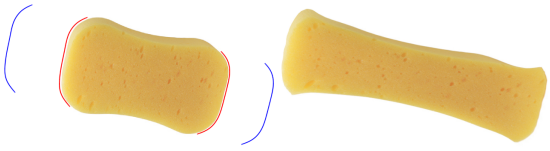

Figure 3: *Stretchy sponge used for performance tests. Original image (left), resulting deformation (right).*

Using the deformation shown in figure 3 as an example, we compare our grid-based solver to triangle meshes. We use the same CRPM energy with triangle meshes and, following [KWSH*13, SKPSH13], we employ a sparse direct solver. Specifically, we use the highly optimized multi-threaded PARDISO solver [SG06]. Our triangle meshes adapt to the targeting curves, and we took effort to create meshes with approximately the same number of degrees of freedom as our regular grids, see table 1. With modest resolutions, single core runs of PARDISO and our multigrid solver achieve about the same speed. However, memory consumption is significantly higher for the direct solver. This translates to a performance advantage for the multigrid solver, especially at higher resolutions, where the direct solver needs to use a lot of memory. Random access to large blocks of RAM typically incurs performance penalties due to the limitations of fast hardware caches. We register an even more significant difference by comparing the multi-threaded versions of our multigrid solver and PARDISO, because sparse direct solvers are difficult to parallelize. In our experiment, we see that PARDISO enjoys only limited benefit (up to $2\times$) from multiple cores (we used a 6-core CPU). On the other hand, our multigrid solver is easy to parallelize, and its performance improves significantly (up to $5\times$) on multiple cores. We conclude that multigrid methods represent an important alternative to direct solvers, given the demand for high resolutions and ever increasing numbers of CPU cores.

We also experimented with reproducing an image warping example from Locally Injective Mappings [SKPSH13], specifically, Figure 12 (Mona Lisa). We mimic the LIM energy in our framework by using:

$$\Psi(\mathbf{F}) = \frac{\mu}{2}\|\mathbf{F}-\mathbf{R}\|_F^2 + \frac{\kappa}{2}(\min(0, J-\varepsilon))^2 \qquad (8)$$

where $\mathbf{R}$ is a best-fit rotation, corresponding to an as-rigid-as-possible energy. With high κ, this energy introduces strong resistance when relative element areas drop below ε (we used $\varepsilon = 0.2$, i.e., we start penalizing shrinkage below 20% of the original area). An important difference to the infinite barriers employed in [SKPSH13] is that our formulation allows for inverted elements, even though they are of course highly penalized. We believe that in typical image deformation scenarios, allowing for inverted elements is in fact more practical, because the user-specified targeting curves may *require* inversion; in this case, the strict barrier functions force LIM to ignore some of the user-provided constraints [SKPSH13]. Such behavior is not acceptable in image warping – in certain special circumstances the user

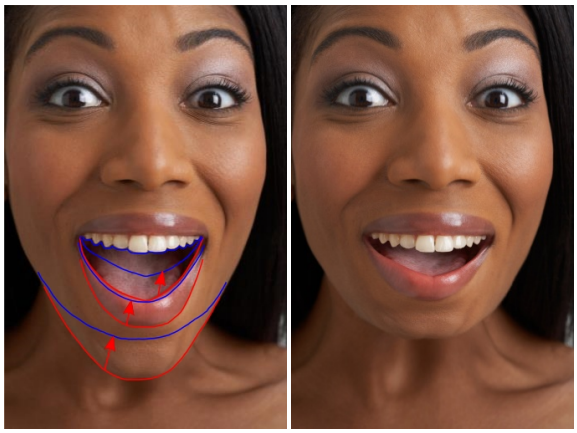

Figure 4: *Closing of the mouth is facilitated by precisely targeted curves, designed to move the lip upwards while keeping the teeth in place.*

may actually desire foldovers. If an inversion-free result exists, our solver is likely to find it because our saddle point formulation allows for arbitrarily high κ in Eq. 8. As we can see in the example in figure 5, inversion occurs within the first few iterations, but is resolved in subsequent iterations. While both LIM and our method converge to the same final result, the paths of individual iterations are different. Interestingly, the performance of both methods is comparable for single-thread execution (all iterations of our method take 65.3s with the MINRES solver compared to 73.8s for LIM), however, our multi-threaded implementation requires only 12.2s. We did not attempt to measure the multi-threaded performance of LIM because the publicly available implementation [SKPSH13] was not optimized for parallel processing, i.e., the comparison would not be fair.

5. Limitations and Future Work

The premise of our work is that high-quality image warping can be achieved by modeling the image as an elastic medium that is (a) endowed with nonlinear material properties and (b) discretized on a regular Cartesian grid. These traits provide benefits in terms of modeling flexibility and numerical efficiency, but also give rise to certain limitations. Specifically, our use of nonlinear membrane models allows us to reproduce interesting behaviors such as area preserving warps. Nonlinear materials, however, are inherently susceptible to non-unique solutions. Linear models such as the Poisson equation, or interpolation techniques such as thin plate splines and RBF networks can guarantee a unique solution. Also, using nonlinear membrane models necessitates the use of nonlinear numerical solvers; our use of a multigrid scheme helps lessen the cost of such nonlinear problems, but purely linear formulations will always have the potential for faster performance. In our future work we aim to investigate adaptivity in the context of methods that use regular elements, e.g. quadtrees.

Our image warps are represented on regular Cartesian grids. This eliminates the need for remeshing and improves the regularity of data structures, generating opportunities for parallel acceleration. However, by adopting this approach we forgo the direct and precise control one would enjoy by remeshing around constraint curves, as we only enforce constraints on a cell-averaged basis. Non-regular meshes can leverage adaptivity to provide greater detail around geometrically intricate features, while our approach commits to a uniform resolution across the entire image.

Finally, although our method promotes properties such as area preservation, there is currently no mechanism to preserve salient features [WTSL08] or to consider the 3D nature of the scene [SD96]. As future work, we would like to investigate incorporating constraints such as length preservation for guide curves, or using bending resistance to discourage distortion of straight-line features.

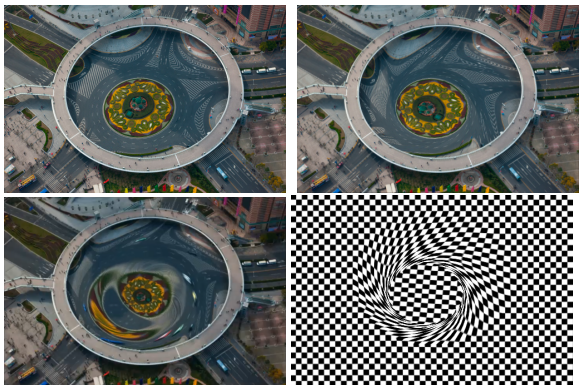

Figure 6: *The island at the center of a roundabout is rotated by 90 degrees while keeping the elevated overpass fixed. Our method (top) produces swirling motion free of folding or inversions, as shown by the checkerboard pattern (bottom right). Thin-plate spline warping leads to unnatural deformation (bottom left).*

6. Acknowledgements

We are grateful to Perry Kivolowitz for motivating this work and providing crucial feedback. We thank Peter Kaufmann and Christian Schueller for sharing their source code and expertise; Tiantian Liu and Nathan Marshak for help with the accompanying video. This research is supported in part by NSF IIS-1253598, NSF CNS-1218432, NSF IIS-1350330, and US Army TARDEC.

References

[ACOL00] ALEXA M., COHEN-OR D., LEVIN D.: As-rigid-as-possible shape interpolation. In *Proceedings of ACM SIGGRAPH '00* (2000), pp. 157–164. 1, 2

[Ale03] ALEXA M.: Differential coordinates for local mesh morphing and deformation. *The Visual Computer 19*, 2 (2003), 105–114. 2

[BBG12] BOYÉ S., BARLA P., GUENNEBAUD G.: A vectorial solver for free-form vector gradients. *ACM Transactions on Graphics (TOG) 31*, 6 (2012), 173. 3

[BBK05] BOTSCH M., BOMMES D., KOBBELT L.: Efficient linear system solvers for mesh processing. In *Mathematics of Surfaces XI*. Springer, 2005, pp. 62–83. 3

[BN92] BEIER T., NEELY S.: Feature-based image metamorphosis. In *Proceedings of SIGGRAPH '92* (1992), pp. 35–42. 1, 2

[Boo89] BOOKSTEIN F. L.: Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Trans. Pattern Anal. Mach. Intell. 11*, 6 (June 1989), 567–585. 2

[BVBZ*10] BEDROSSIAN J., VON BRECHT J., ZHU S., SIFAKIS E., TERAN J.: A second order virtual node method for elliptic problems with interfaces and irregular domains. *Journal of Computational Physics 229*, 18 (2010), 6405–6426. 3, 4

[BW08] BONET J., WOOD R.: *Nonlinear continuum mechanics for finite element analysis*. Cambridge University Press, 2008. 3

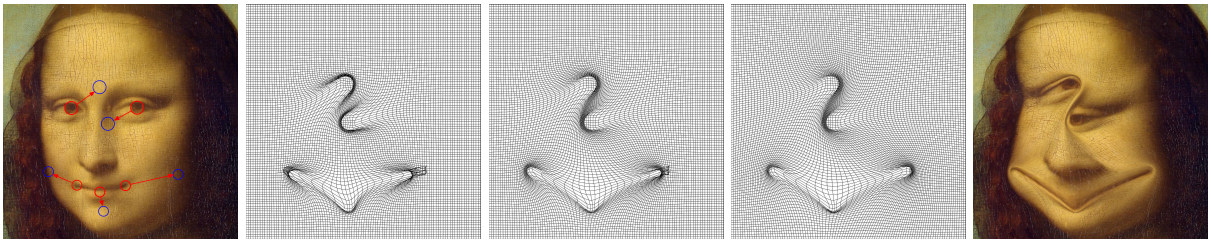

Figure 5: *In contrast to Locally Injective Mappings [SKPSH13], our method allows for inverted elements in intermediate steps. However, when the optimization is finished, all elements successfully recover from inversion, producing a fold-over free result.*

[CAA09] CARROLL R., AGRAWAL M., AGARWALA A.: Optimizing content-preserving projections for wide-angle images. In *ACM Transactions on Graphics (TOG)* (2009), vol. 28, ACM, p. 43. 1, 2

[FSH11] FINCH M., SNYDER J., HOPPE H.: Freeform vector graphics with controlled thin-plate splines. In *ACM Transactions on Graphics (TOG)* (2011), vol. 30, ACM, p. 166. 3

[IMH05] IGARASHI T., MOSCOVICH T., HUGHES J. F.: As-rigid-as-possible shape manipulation. *ACM Trans. Graph. 24*, 3 (July 2005), 1134–1141. 2, 3

[JBPS11] JACOBSON A., BARAN I., POPOVIĆ J., SORKINE O.: Bounded biharmonic weights for real-time deformation. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH) 30*, 4 (2011), 78:1–78:8. 1, 2

[JMD*07] JOSHI P., MEYER M., DEROSE T., GREEN B., SANOCKI T.: Harmonic coordinates for character articulation. In *ACM Transactions on Graphics (TOG)* (2007), vol. 26, p. 71. 3

[KFG09] KARNI Z., FREEDMAN D., GOTSMAN C.: Energy-based image deformation. In *Computer Graphics Forum* (2009), vol. 28, pp. 1257–1268. 2

[KWSH*13] KAUFMANN P., WANG O., SORKINE-HORNUNG A., SORKINE-HORNUNG O., SMOLIC A., GROSS M.: Finite element image warping. In *Computer Graphics Forum* (2013), vol. 32, pp. 31–39. 3, 7, 8

[LGJA09] LIU F., GLEICHER M., JIN H., AGARWALA A.: Content-preserving warps for 3D video stabilization. *ACM Trans. Graph. 28*, 3 (July 2009), 44:1–44:9. 2

[Lip12] LIPMAN Y.: Bounded distortion mapping spaces for triangular meshes. *ACM Transactions on Graphics (TOG) 31*, 4 (2012), 108. 2

[MZS*11] MCADAMS A., ZHU Y., SELLE A., EMPEY M., TAMSTORF R., TERAN J., SIFAKIS E.: Efficient elasticity for character skinning with contact and collisions. *ACM Trans. Graph. 30*, 4 (July 2011), 37:1–37:12. 3

[NW06] NOCEDAL J., WRIGHT S. J.: Numerical optimization 2nd. 4

[PMS12] PATTERSON T., MITCHELL N., SIFAKIS E.: Simulation of complex nonlinear elastic bodies using lattice deformers. *ACM Trans. Graph. 31*, 6 (Nov. 2012), 197:1–197:10. 3, 5

[PS75] PAIGE C., SAUNDERS M.: Solution of sparse indefinite systems of linear equations. *SIAM Journal on Numerical Analysis 12*, 4 (1975), 617–629. 5

[RT07] REHMAN T., TANNENBAUM A.: Multigrid optimal mass transport for image registration and morphing. In *Electronic Imaging 2007* (2007), International Society for Optics and Photonics, pp. 649810–649810. 2

[SB12] SIFAKIS E., BARBIC J.: FEM simulation of 3D deformable solids: a practitioner's guide to theory, discretization and model reduction. In *ACM SIGGRAPH 2012 Courses* (2012), pp. 20:1–20:50. 3

[SD96] SEITZ S. M., DYER C. R.: View morphing. In *Proceedings of SIGGRAPH '96* (1996), pp. 21–30. 2, 9

[SG06] SCHENK O., GÄRTNER K.: On fast factorization pivoting methods for sparse symmetric indefinite systems. *Electronic Transactions on Numerical Analysis 23* (2006), 158–179. 2, 8

[SGW07] SCHIWIETZ T., GEORGII J., WESTERMANN R.: Freeform image. In *PG'07. 15th Pacific Conference on Computer Graphics and Applications, 2007.* (2007), IEEE, pp. 27–36. 3

[She96] SHEWCHUK J. R.: Triangle: Engineering a 2d quality mesh generator and delaunay triangulator. In *Applied computational geometry towards geometric engineering*. Springer, 1996, pp. 203–222. 8

[SK04] SHEFFER A., KRAYEVOY V.: Shape preserving mesh deformation. In *ACM SIGGRAPH 2004 Sketches* (2004), ACM, p. 39. 3

[SKPSH13] SCHÜLLER C., KAVAN L., PANOZZO D., SORKINE-HORNUNG O.: Locally injective mappings. *Computer Graphics Forum (proceedings of EUROGRAPHICS/ACM SIGGRAPH Symposium on Geometry Processing) 32*, 5 (2013), 125–135. 2, 3, 4, 7, 8, 9, 10

[SMW06] SCHAEFER S., MCPHAIL T., WARREN J.: Image deformation using moving least squares. *ACM Trans. Graph. 25*, 3 (July 2006), 533–540. 1, 2

[SYBF06] SHI L., YU Y., BELL N., FENG W.-W.: A fast multigrid algorithm for mesh deformation. In *ACM Transactions on Graphics (TOG)* (2006), vol. 25, ACM, pp. 1108–1117. 3

[TOS01] TROTTENBERG U., OOSTERLEE C., SCHÜLLER A.: *Multigrid*. Academic Press, 2001. 6, 7

[WLSL10] WANG Y.-S., LIN H.-C., SORKINE O., LEE T.-Y.: Motion-based video retargeting with optimized crop-and-warp. *ACM Transactions on Graphics (TOG) 29*, 4 (2010), 90. 1, 2

[WSBZ08] WENG Y., SHI X., BAO H., ZHANG J.: Sketching mls image deformations on the gpu. In *Computer Graphics Forum* (2008), vol. 27, Wiley Online Library, pp. 1789–1796. 2

[WTSL08] WANG Y.-S., TAI C.-L., SORKINE O., LEE T.-Y.: Optimized scale-and-stretch for image resizing. *ACM Trans. Graph. 27*, 5 (Dec. 2008), 118:1–118:8. 1, 2, 9

[WXW*06] WENG Y., XU W., WU Y., ZHOU K., GUO B.: 2D shape deformation using nonlinear least squares optimization. *The visual computer 22*, 9-11 (2006), 653–660. 2, 3