

Compressed vibration modes of elastic bodies

Brandt, Christopher; Hildebrandt, Klaus

DOI

[10.1016/j.cagd.2017.03.004](https://doi.org/10.1016/j.cagd.2017.03.004)

Publication date

2017

Document Version

Accepted author manuscript

Published in

Computer-Aided Geometric Design

Citation (APA)

Brandt, C., & Hildebrandt, K. (2017). Compressed vibration modes of elastic bodies. *Computer-Aided Geometric Design*, 52-53, 297-312. <https://doi.org/10.1016/j.cagd.2017.03.004>

Important note

To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright

Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy

Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

Compressed Vibration Modes of Elastic Bodies

Christopher Brandt Klaus Hildebrandt

Delft University of Technology

Abstract

The natural vibration modes of deformable objects are a fundamental physical phenomenon. In this paper, we introduce *compressed vibration modes*, which, in contrast to the natural vibration modes, are localized (“sparse”) deformations. The localization is achieved by augmenting the objective which has the vibration modes as minima by a L^1 term. As a result, the compressed modes form a compromise between localization and optimal energy efficiency of the deformations. We introduce a scheme for computing bases of compressed modes by solving sequences of convex optimization problems. Our experiments demonstrate that the resulting bases are well-suited for reduced-order shape deformation and for guiding the segmentation of objects into functional parts.

1. Introduction

Vibration modes and their frequencies are fundamental for analyzing and simulating the dynamics of physical objects. In this work, we introduce an approach for the compression and localization of vibration modes of elastic bodies. The *compressed vibration modes* have a localized support while preserving properties of the natural vibration modes, *e.g.*, they form an orthonormal system in the space of configurations of an elastic body and the low-frequency modes correspond to low-energy deformations. The degree of localization can be controlled by a continuous parameter μ .

For applications, *e.g.*, for reduced-order simulations, the localization provides benefits. The vectors describing the compressed vibration modes of a discrete object are sparse, which results in less memory requirements for storing a basis and fewer arithmetic operations for adding or scaling the vectors. In our experiments, we used the compressed vibration modes for reduced-order simulation and deformation-based shape modeling. A second aspect is that the localization of the vibration modes adds a novel aspect to the modal analysis of deformable objects. Our experiments illustrate that the compressed modes localize in a structured way. We see potential in exploring this structure for modal and shape analysis and using it for applications. As a first step in this direction, we use the compressed modes for shape segmentation into functional parts.

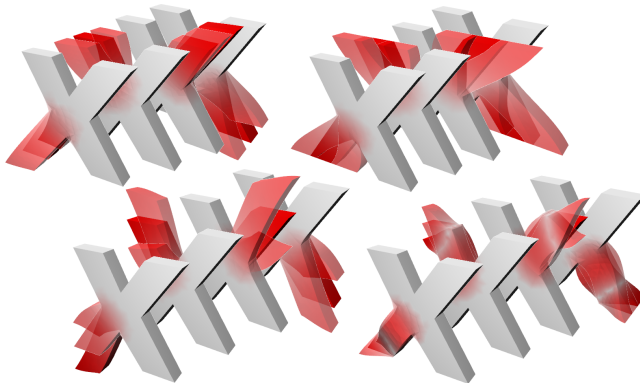


Figure 1: The first sixteen compressed vibration modes of an X-shaped mesh ($\mu = 10$). We grouped the same modes (up to symmetry) for each of the four 'legs'.

To define the compressed vibration modes, we first characterize the natural vibration modes as the minimizers of an optimization problem, then we add a L^1 regularizing term to the objective to enforce compression. The idea of using L^1 regularization to localize modes of variational problems was introduced in [1] and applied to Schrödinger's equation. The compressed vibration modes, we introduce, specialize this idea to the localization of modes of vibration of elastic bodies. There are significant differences in how we define and compute the compressed modes compared to [1] and other work on the compression of modes like [2, 3, 4]. Whereas they compute all compressed modes in one optimization, we devise a reformulation of the L^1 regularized eigenproblem, which allows for computing the modes sequentially. This results in computation timings that scale linearly in the number of modes, whereas previous methods scale superlinearly. Moreover, to compute the modes, we devise an algorithm for solving the L^1 regularized optimization problem, which involves a convexification of a constraint as well as a linearization of the L^1 term. The algorithm allows to stably compute a large number of modes, as the runtime for each additional mode remains (almost) constant. The shown experiments demonstrate the benefits of this computational method over the ADMM method for the computation of compressed modes. An additional point is that our reformulation of the compressed eigenproblem allows for solving the L^1 constrained version of the problem. This formulation has the benefit over the L^1 regularized problem that the L^1 norm remains constant for all modes. This aspect has not been treated in prior work and seems difficult to achieve with prior problem formulations.

2. Related Work

Vibration modes. Vibration modes are fundamental for describing, analyzing and simulating the dynamics of objects. The low-frequency vibration modes correspond to low-energy deformations of an object, which makes spaces spanned

by low-frequency vibration modes attractive for reducing the computational cost of simulations and optimizations. In recent work in graphics, vibration modes are used for reduced-order models for deformation-based editing [5], deformable object simulation [6, 7, 8, 9], sound synthesis [10, 11], and spacetime control of animations [12, 13, 14, 15]. Besides the dimensional reduction of simulations and optimizations, vibration modes are used for the analysis of objects [16, 17] and the segmentation of objects into functional parts [18]. One drawback of using vibration modes for dimensional reduction is that the resulting basis vectors are dense, which can be problematic when a large basis is used or applications, like games, impose strict limits on the memory available for the reduced simulation. This problem has been addressed in [19] by applying a data compression scheme for storing the basis vectors. We present, for the first time, a localization of vibration modes, which allows for creating bases of low-energy deformations that are sparse. One resulting benefit for reduced-order methods is a reduced memory requirement.

L^1 regularization for eigenvalue problems. *Compressed modes for variation problems* were introduced by Ozoliņš et al. [1] and used for computing localized bases for Schrödinger’s equation. For the numerical computation of the compressed modes, they used a *splitting orthogonality constraint (SOC)* scheme. The approach was extended to the computation of compactly supported multiresolution bases for the Laplace operator on planar domains by Ozoliņš et al. [20] and the sparse approximation of differential operators in the Fourier domain by Mackey et al. [21]. Compressed eigenfunctions of the Laplace–Beltrami operator of curved surfaces, called *compressed manifold modes*, were considered by Neumann et al. [2] and used for mesh segmentation and functional correspondences. For computing the compressed manifold modes, they proposed a scheme based on the alternating direction method of multipliers (ADMM, [22]) and demonstrated that it outperforms the SOC scheme. Boscaini et al. [3] used the compressed manifold modes for building class-specific descriptors for non-rigid shapes. Houston [23] proposed a natural ordering for the compressed manifold modes along with an adaptation of the algorithm that is reported to significantly reduce the number of ADMM iterations required in the optimization. Since the definition of compressed modes includes a unit L^2 norm constraint, the feasible set for the optimization is not a linear space, but a curved manifold. A generic algorithm for L^1 regularized minimization problems over manifolds called the manifold alternating direction method of multipliers (MADMM) was introduced by Kovnatsky et al. [4] and used for the computation of compressed manifold modes. Concurrent to our work, Bronstein et al. [24] proposed a discretization of the L^1 norm for linear Lagrange finite elements on meshes and an iteratively reweighted least-squares scheme for computing compressed manifold modes. In this paper, we opt for a different approach for computing the compressed vibration modes using convexification of the constraints and linearization of the L^1 term by duplication of the variables.

ℓ_0 regularized eigenvalue problems. Sparsity in finite dimensional eigenvalue problems can be enforced by regularization with a ℓ_0 “norm”, which counts the number of non-zero entries in a vector. A recent scheme for solving the resulting *sparse eigenvalue problems* is the *truncated power method* introduced by Yuan et al. [25]. In order to compute sparse eigenvectors of a symmetric matrix with less than k non-zero entries, the matrix is iteratively applied to some initial vector, and subsequently all but the k largest absolute values are set to 0. Unfortunately such a simple approach is not suited in our scenario since we are interested in finding approximations of continuous functions with localized support, and the ℓ_0 “norm” of a vector representing such a function in some discretization scheme does not measure the size of the support since the mapping between the function and the vector depends on the underlying mesh. If weights, which consider the area associated to each vertex, are used to account for the underlying mesh, the complexity of the problem rises and methods like the truncated power method are no longer applicable.

L^1 regularization for empirical eigenvalue problems. Related to the construction of subspaces using compressed modes is the problem of constructing localized bases that span a given space of deformations. Meyer et al. [26] considered spaces specified by an animator’s rig and compute localized, not necessarily orthogonal bases of the rig space via Varimax rotations. It is demonstrated that the resulting basis vectors localize in a structured way. Neumann et al. [27] introduced a scheme for computing localized bases in a spaces of captured facial deformations using a L^1 regularized principal component analysis. In contrast to the approach proposed here, a large sequence of input shapes is required and no elastic energies are taken into account. As a result, this method reduces and localizes a set of input deformations, as opposed to producing a new set of previously unknown deformations based on an analysis of the elastic energy of an object.

Sparsity in geometry processing. Sparsity enforcing regularization has also been used in the context of shape deformation. Gao et al. [28] introduced localization encouraging deformation energies. Specifically, they modify the As-Rigid-As-Possible energy by replacing terms measuring least-squares deviations from the rest configuration by terms involving p -norms, which, for small p , tends to localize the deformations when using the modified energy in modeling tasks (*i.e.* minimizing it under soft or hard constraints). More examples, where ℓ_1 or ℓ_0 regularization has been employed for geometry processing tasks, surface smoothing [29] and optimal spline approximation [30, 31]. For a recent survey on compressed sensing for geometry processing, we refer to [32].

3. Background: Deformation Energies and Vibration Modes

In this section, we introduce basic concepts concerning deformation energies and vibration modes. Due to space restrictions, we consider only the discrete setting and the concepts needed to define the compressed vibration modes in the

next section. For more background on elasticity and finite element discretization, we refer to the textbook [33].

In the following, we will deal with triangle surface meshes and tetrahedral volume meshes with fixed connectivity $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{F}, \mathcal{T})$ (vertices, edges, faces and, in case of tetrahedral meshes, tetrahedrons) and initial vertex positions given by a vector $\mathbf{x} \in \mathbb{R}^{3|\mathcal{V}|}$. We refer to the mesh with these vertex positions as the *rest configuration*. We express a deformation of the mesh via a vector $\mathbf{u} \in \mathbb{R}^{3|\mathcal{V}|}$, which stacks the displacements of each vertex in world coordinates.

A discrete deformation energy $E_{\mathcal{M}, \mathbf{x}} : \mathbb{R}^{3|\mathcal{V}|} \rightarrow \mathbb{R}$ is a measure for the effort it takes to deform an object from its rest configuration into a deformed state given by some displacement vector $\mathbf{u}$, *i.e.* the new vertex positions are given by $\mathbf{x} + \mathbf{u}$, while assuming that the object consists of a hyperelastic material. The energy E depends on the connectivity of the mesh, its rest configuration and material properties. For our experiments, we used a finite element discretization of *St. Venant–Kirchhoff materials* [33] for simulating elastic solids and *Discrete Shells* [34] for elastic shells.

In order to define a L^2 scalar product and L^1 and L^2 norms on the space of displacements of a mesh, we associate a mass m_i to every vertex. We assume that the material of the elastic bodies has a constant density ρ , and, for thin shells, we assume a constant thickness ϵ . Then, for elastic solids, the mass m_i equals ρ times a fourth of the combined volume of all tets containing vertex v_i , and, for thin shells m_i equals $\rho\epsilon$ times a third of the combined area of the triangles containing v_i . The *mass matrix* $\mathbf{M}$ is the $3|\mathcal{V}| \times 3|\mathcal{V}|$ diagonal matrix that stacks the masses on the diagonal (in the same order as the displacements are stacked). The L^2 scalar product is

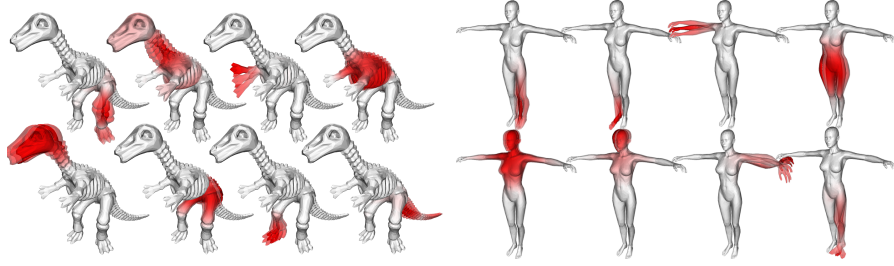
$$\langle \mathbf{u}, \mathbf{v} \rangle := \mathbf{u}^T \mathbf{M} \mathbf{v}, \quad (1)$$

with corresponding L^2 norm $\|\mathbf{u}\|_2 = \langle \mathbf{u}, \mathbf{u} \rangle^{\frac{1}{2}}$, and the L^1 norm is

$$\|\mathbf{u}\|_1 := \sum_{i=0}^{3|\mathcal{V}|} M_{ii} |u_i|, \quad (2)$$

where M_{ii} is the i^{th} entry on the diagonal of $\mathbf{M}$ and u_i the i^{th} entry of $\mathbf{u}$. In [2] the ℓ_1 -norm was used for computing compressed manifold modes, *i.e.* the masses associated to the vertices were not considered when computing the sparsity regularizer. We illustrate the effect of the different 1-norms on the compressed vibration modes in Figure 4, where we demonstrate that our method is robust against remeshing of the object.

Natural vibration modes. We consider the modes of vibration of an elastic object at its rest configuration. The Hessian $\mathbf{H}$ of $\mathbf{E}$ at $\mathbf{x}$ is the matrix of second derivatives of $\mathbf{E}$ with respect to displacements of the object, *i.e.*, $\mathbf{H}_{ij} = \frac{\partial^2}{\partial u_i \partial u_j} E_{\mathcal{M}, \mathbf{x}}(0)$. At the rest configuration, the Hessian is a symmetric and positive semi-definite matrix. The vibration modes are the eigenvectors and the



(a) Eight of the first fifty compressed vibration modes on the dinosaur model (tetrahedral mesh, St.-V.-K. energy, $\mu = 20$).
(b) Eight of the first fifty compressed vibration modes on the Victoria mesh (surface mesh, thin shells energy, $\mu = 0.0001$).

Figure 2: Two sets of compressed vibration modes for different geometry types and elastic energies.

frequencies are the square roots of the eigenvalues of the generalized eigenvalue problem

$$\mathbf{H}\mathbf{u}_i = \lambda_i \mathbf{M}\mathbf{u}_i. \quad (3)$$

The problem can be re-written as the following sequence of optimization problems:

$$\mathbf{u}_i := \begin{cases} \arg \min_{\mathbf{u}} \mathbf{u}^T \mathbf{H} \mathbf{u} \\ \text{subject to } \mathbf{u}^T \mathbf{M} \mathbf{u} = 1 \text{ and } \forall j < i : \mathbf{u}^T \mathbf{M} \mathbf{u}_j = 0 \end{cases}$$

We will use this definition as the starting point for defining the compressed vibration modes.

4. Compressed Vibration Modes

Vibration modes are in general global and thus a displacement in direction of any mode deforms the whole object. Since reduced order methods need to store the whole subspace basis, this is a drawback for applications imposing strict bounds on the memory requirements, *e.g.*, when the GPU memory is shared for different computational tasks (Games or VR) or larger bases or bases for multiple objects are needed (real-time simulation). Moreover, we found that localized deformations are interesting and natural quantities of the mesh: they localize in a structured way and thus give a useful tool for analyzing the mesh and exploring the space of deformations. Thus, we aim at finding a sparse, *i.e.* localized basis for deformations of a mesh, which represent a trade-off between optimality w.r.t. the deformation energy $E_{\mathcal{M}, \mathbf{x}}$ and sparsity, *i.e.* few non-zero entries per vector.

To this end, we define the compressed modes as solutions $\mathbf{u}_i$ of the following sequence of optimization problems:

$$\mathbf{u}_i := \begin{cases} \arg \min_{\mathbf{u}} \mathbf{u}^T \mathbf{H} \mathbf{u} + \mu \|\mathbf{u}\|_1 \\ \text{subject to } \mathbf{u}^T \mathbf{M} \mathbf{u} = 1 \text{ and } \forall j < i : \mathbf{u}^T \mathbf{M} \mathbf{u}_j = 0 \end{cases} \quad (4)$$

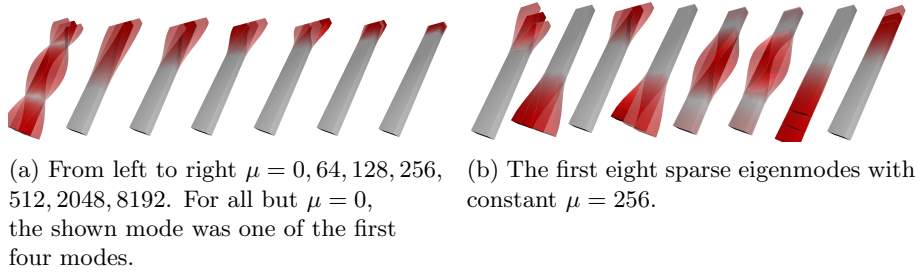


Figure 3: Compressed vibration modes of a tetrahedral bar mesh.

Note that this is a direct extension of the iterative definition for natural vibration modes as given above, with the addition of a sparsity regularizer: the term $\mu \|\mathbf{u}\|_1$ represents a trade-off between minimizing the elastic energy introduced by the mode, and its sparsity. Higher μ results in more localized deformations, see Figure 3a. By iteratively adding orthogonality constraints, the resulting modes cover a broad range of deformations, see Figure 3b. The compressed vibration modes strongly depend on the structure of the underlying object and thus analyzing their support reveals some of the structure of the object in return (this will be demonstrated in Section 7, where we use compressed vibration modes for object segmentation). It is not obvious how to choose the parameter μ such that a certain localization is achieved. We show several examples for sets of sparse vibration modes and discuss the effects of different sparsity parameters μ in Section 8. Additionally, we introduce an alternative definition for compressed vibration modes, where the L^1 norm is being constrained instead of regularized, below.

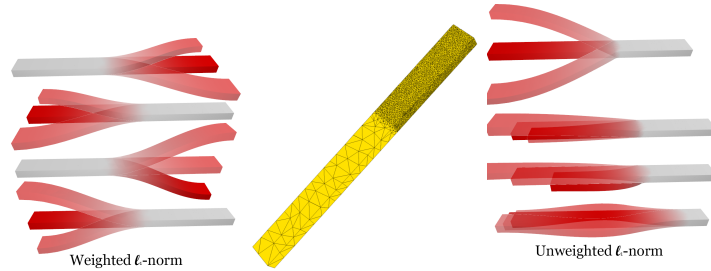


Figure 4: Comparison of the first four modes when using the mass-weighted discrete L^1 norm (left) when optimizing for compressed vibration modes of a irregular bar mesh (middle) and the (unweighted) L^1 norm (right) (which was used in [2]).

The definition (4) for compressed vibration modes has several differences to the previous definitions of compressed modes as given in [1, 2, 3, 4] (aside from us looking at localized vibrations of elastic bodies for the first time). They look for a full set of vectors in a single optimization problem, constraining the whole set to be orthonormal, whereas we define them as solutions of a sequence

of smaller optimization problems, while adding an orthogonality constraint in each iteration. This has the advantage that each of the minimization problems has a comparatively low number of variables. When trying to compute a large number of modes for high resolution meshes, solving for all modes at once becomes prohibitive. Note that for $\mu = 0$ both formulations yield the solutions to the generalized eigenvalue problem. Another advantage is that the choice of the sparsity parameter μ becomes consistent: the sparsity of the modes does not depend on the number of modes K but only on μ , whereas in [2], the same choice of μ would lead to different sparsity for bases of different sizes. Another difference in our definition is that we use a proper discretization of the L^1 norm of piecewise linear displacements, as opposed to the unweighted vector ℓ_1 norm used in [2].

5. Computation of Compressed Vibration Modes

The optimization problem (4) is non-convex due to the constraint on the L^2 norm of $\mathbf{u}$ and the L^1 term is non-differentiable. For solving this type of problem, a Splitting Orthogonality Constraint and an Alternating Direction Method of Multipliers scheme have been proposed in [1] and [2]. A comparison of the two approaches in [2] indicates that the ADMM scheme is more effective. In our experiments, see Section 8, ADMM did not produce satisfying results for the computation of compressed vibration modes, neither in terms of computation times, nor in terms of convergence to an acceptable minimum. Therefore, we opt for a novel optimization scheme, which we refer to as *iterated convexification for compressed modes* (ICCM). It combines a convexification of the problem with the classical approach of turning the L^1 term into a linear term and linear inequality constraints by duplicating the variables. We will describe both steps in detail below. ICCM is parameter free and its implementation is straightforward, given a quadratic programming library.

Convexification. We replace the constraint $\mathbf{u}^T \mathbf{M} \mathbf{u} = 1$, by a hyperplane constraint $\mathbf{u}^T \mathbf{M} \mathbf{c}_0 = 1$, where $\mathbf{c}_0$ is initialized as a random, normalized vector. Essentially this replaces the constraint that the deformation should be on the unit sphere, with the constraint that the deformation should be on a plane that is tangential to the unit sphere. After we find a solution $\mathbf{u}$ with this constraint, we set $\mathbf{c}_1 = \mathbf{u} / \|\mathbf{u}\|_{L^2}$ and solve the optimization problem again, this time with the hyperplane constraint $\mathbf{u}^T \mathbf{M} \mathbf{c}_1 = 1$. This is being repeated until some convergence criterion is met. In our experiments we used a lower bound on the deltas between the energy values of the last and current solution as the stopping criterion. After finding a mode, we initialize the next optimization problem by again using a random hyperplane constraint $\mathbf{c}_0$, which additionally has the property that the problem admits a feasible solution, *i.e.* $\forall j < i : \mathbf{c}_0^T \mathbf{M} \mathbf{u}_j = 0$. While this convexification might lead to finding a local instead of a global minimum, it provides a fast and suitably stable way to solve a large scale, non-convex and non-differentiable optimization problem. The performance and consistence of this convexification is being further analyzed in Section 8.

Linearization of the L^1 term. After convexifying the L^2 constraint, each mode is computed via solving a sequence of convex optimization problems. In order to turn these into quadratic programs, we get rid of the non-differentiable part introduced by the L^1 regularizer by using non-negative variables, that is,

$$\mathbf{u} = \mathbf{u}^+ - \mathbf{u}^- \quad \mathbf{u}^+, \mathbf{u}^- \geq 0, \quad (5)$$

where the inequality is meant component wise. This is a widely used approach, first proposed in Tibshirani et al. [35]. With this change of variables the L^1 term can be bounded by a linear term

$$\|\mathbf{u}\|_1 \leq \mathbf{1}^T \mathbf{M} \mathbf{u}^+ + \mathbf{1}^T \mathbf{M} \mathbf{u}^-. \quad (6)$$

After expressing the energy functional and all constraints in terms of $\mathbf{u}^+$ and $\mathbf{u}^-$, as well as adding the non-negativity constraints (5) and employing the convexification above, the optimization problem (4) turns into an inner and outer sequence of quadratic programs. In each inner iteration we adapt the convex hyperplane constraints until a convergence criteria is met:

$$\mathbf{u}_{i,k} := \begin{cases} \arg \min_{\mathbf{u}=\mathbf{u}^+-\mathbf{u}^-} \begin{pmatrix} \mathbf{u}^+ \\ \mathbf{u}^- \end{pmatrix}^T \begin{pmatrix} \mathbf{H} & -\mathbf{H} \\ -\mathbf{H} & \mathbf{H} \end{pmatrix} \begin{pmatrix} \mathbf{u}^+ \\ \mathbf{u}^- \end{pmatrix} \\ \quad + \mu (\mathbf{1}^T \mathbf{M} \mathbf{u}^+ + \mathbf{1}^T \mathbf{M} \mathbf{u}^-) \\ \text{subject to} \begin{pmatrix} \mathbf{u}^+ \\ \mathbf{u}^- \end{pmatrix}^T \begin{pmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{pmatrix} \begin{pmatrix} \mathbf{c}_k \\ -\mathbf{c}_k \end{pmatrix} = 1 \\ \quad \text{and } \forall j < i : \begin{pmatrix} \mathbf{u}^+ \\ \mathbf{u}^- \end{pmatrix}^T \begin{pmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{pmatrix} \begin{pmatrix} \mathbf{u}_j \\ -\mathbf{u}_j \end{pmatrix} = 0 \\ \quad \text{and } \mathbf{u}^+, \mathbf{u}^- \geq 0 \end{cases} \quad (7)$$

$$\mathbf{c}_{k+1} = \mathbf{u}_{i,k} / \|\mathbf{u}_{i,k}\|_2.$$

After that we add the last mode from this sequence to the set of computed sparse vibration modes, *i.e.*

$$\mathbf{u}_i := \mathbf{u}_{i,k} / \|\mathbf{u}_{i,k}\|_2$$

and then start the inner iteration above again, with the additional orthogonality constraint induced by the new mode. Note that the linear term that bounds the L^1 norm from above, (6), is actually equal to the L^1 norm for solutions of (7), since otherwise one could trivially construct a feasible lower energy solution (see the appendix where we show this for the L^1 constrained formulation). Thus, (6) is a valid, linear, replacement for the non-differentiable term $\|\mathbf{u}\|_1$.

Each inner iteration of ICCM amounts to solving a quadratic program which can be done using highly efficient specialized solvers (in our implementation we used *Mosek*). Typical iteration times and full times for computing a set of

modes can be taken from Figure 14 and Table 4. Table 3 shows a comparison to the computation times resulting from ADMM as implemented in [2]. We will analyze the resulting compressed vibration modes and discuss stability and computation times of the algorithm in more detail in Section 8.

6. L^1 constrained vibration modes

In addition to using L^1 regularization to enforce compression of the modes, we can consider a L^1 constrained formulation: we remove the term $\mu\|\mathbf{u}\|_1$ from the objective functional and instead add the constraint $\|\mathbf{u}\|_1 = s$ for some $s \in \mathbb{R}^{>0}$. The resulting optimization problem is only well-posed, if there are solutions with unit L^2 norm and L^1 norm lower or equal to s . Whether feasible solutions exist for a given value of s can be checked by a numerical optimization software such as *MOSEK*. Also note that the correctness of the linearized L^1 term used in our optimization has to be shown for the constrained version, the proof can be found in the appendix. For the L^1 constrained problem, the inner iterations of ICCM remain quadratic programs and the algorithm results in comparable computation times.

The attractiveness of the L^1 constrained version becomes apparent when analyzing the behavior of the elastic term $\mathbf{u}^T \mathbf{H} \mathbf{u}$. For many meshes, the vibration modes have energy levels with vastly different magnitudes. Thus, the L^1 regularizer has a different effect on the sparsity of the support of the compressed vibration modes: While the first modes have the desired sparsity pattern, later modes might have a dense support when using the same value of μ . This can be seen in Figure 5 where, in the L^1 regularized case ($\mu = 255$), the support of the first eight modes covers roughly one third of the mesh, but higher frequency modes have a support that covers roughly 90% of the mesh. They are essentially dense modes, which shows that the L^1 term does not have significant influence on the optimization. If we fix the value of the sparsity term of the first mode in the L^1 regularized case and use it as the value s in the L^1 constrained case (here $s = 11.5$), we are able to compute a set of modes with consistent sparsity pattern. The plots in Figure 3b show the respective values of the L^1 and elasticity terms. From this it can be seen that a regularization is not suited since the elastic terms explode for higher frequencies. This prevents us from having to search for a suited value of μ when computing each mode, which would be infeasible given the computation times of the modes.

We want to remark that while ICCM can be used for computing a constrained version of the compressed modes, it is not clear how to employ previously proposed minimization schemes to solve the constrained version. To the best of our knowledge, this is the first time a L^1 constrained version for compressed modes has been considered. In [36], Rustamov et al. propose an L^1 constrained optimization problem to obtain smooth functions on meshes, *Multiscale Biharmonic Kernels*. Their motivation to use an L^1 constraint is similar to ours: the parameter that controls the magnitude of the L^1 -term controls the localization of the solutions directly. For this problem, no non-convex constraint is present, such that the minimization can be directly formulated as a quadratic program

with linear (in)equality constraints.

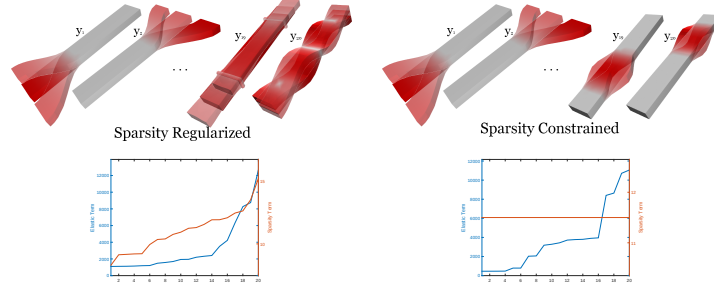


Figure 5: Left: The first, second, 19th and 20th L^1 regularized compressed vibration modes for $\mu = 255$. Right: The first, second, 19th and 20th L^1 constrained compressed vibration modes for $s = 11.5$, which is the value of the L^1 norm of the first L^1 regularized vibration mode for $\mu = 255$. The plots show the values of the elastic and sparsity terms in both cases.

Note, that the previous definitions of compressed modes takes all K modes into account in a single optimization problem, and thus applies the L^1 regularization to the whole matrix containing the modes. The problem of finding a suitable parameter μ remains similar: when computing a set of K and a set of $\tilde{K}$ modes, it is not clear how to choose the values of μ , such that a similar sparsity pattern is achieved. Also note that the problem of varying degrees of localization for a fixed value of μ but a larger number of modes K is not specific to our setting of compressed vibration modes and we think that other methods using compressed eigenmodes will benefit from the constrained formulation.

Lastly, we want to clarify that both (the regularized and the constrained) formulations are valid and there are situations in which either might be preferable. When using the L^1 constrained formulation, we specify an approximate volume (or area) that is covered by the support of the vibration mode, which might not be desirable: in Figures 2a and 2b we see modes whose supports cover vastly different amounts of area or volume respectively, but we intuitively convey these parts as functional units of the mesh.

7. Applications

Before we cover the analysis of our optimization scheme, compare to previous methods and show examples for sets of compressed vibration modes, we will cover two applications that benefit from the localized deformations that have been introduced in this paper.

Compressed deformation bases, elastic simulation and deformation. A general advantage of compressed vibration modes is that they can be stored efficiently by storing a list of the n non-zero indices along with the non-zero entries instead of storing all $N \gg n$ entries. This results in a significant reduction of the

space required to store the deformation basis, which becomes essential in applications that impose strict memory requirements, *e.g.* when the GPU memory needs to be shared by different processes (games or VR) or when large or multiple reduced bases have to be stored at the same time (interactive simulations). Moreover, mapping from the subspace given by a sparse basis to the full space is cheaper (given a certain sparsity) than mapping from a dense basis to the full space since sparse matrix vector products can be computed more efficiently. In Table 4, we list the relative memory size of the sparse basis compared to the dense basis. We store the sparse deformations as a list of integers representing the indices of the non-zero vectors along with the three Doubles that represent the displacement vector. The dense deformations are simply stored as $N \cdot 3$ Double values representing all displacement vectors.

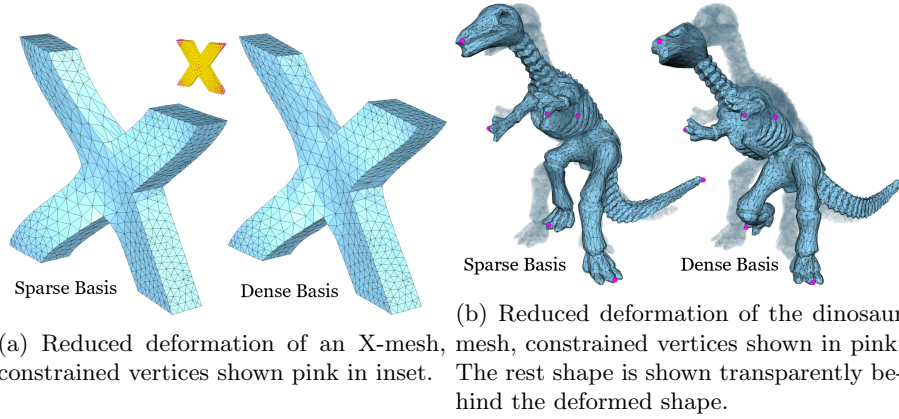


Figure 6: Reduced deformation using a basis of sparse and dense vibration modes respectively, for more details see Section 7.

This raises the question of whether in a reduced-order simulation a basis of compressed vibration modes can compete with a dense basis of the same dimension. To test this, first we used the compressed vibration modes in an elastic simulation setup. In the following it is convenient to use the change of coordinates

$$\hat{\mathbf{u}} := \mathbf{M}^{1/2} \mathbf{u}, \quad (8)$$

under which the mass matrix becomes identity. Likewise, let $\hat{\mathbf{H}} = \mathbf{M}^{-1/2} \mathbf{H} \mathbf{M}^{-1/2}$, *i.e.* the elasticity Hessian expressed in these coordinates.

We consider the linearized equations of motion induced by the elasticity Hessian,

$$\ddot{\hat{\mathbf{u}}} + \beta \hat{\mathbf{H}} \dot{\hat{\mathbf{u}}} + \hat{\mathbf{H}} \hat{\mathbf{u}} = 0, \quad (9)$$

where $\beta \in [0, 1]$ is a damping parameter. We reduce (9) using the basis of compressed vibration modes $\hat{\mathbf{U}} = (\hat{\mathbf{u}}_1, \dots, \hat{\mathbf{u}}_K)$, that is, $\hat{\mathbf{U}}$ is the $3|\mathcal{V}| \times K$ matrix which contains the compressed vibration modes, expressed in terms of the

changed coordinates, as columns. Let $\tilde{\mathbf{H}} = \hat{\mathbf{U}}^T \hat{\mathbf{H}} \hat{\mathbf{U}}$ and let $\Phi = (\phi_1, \dots, \phi_K)$ be the matrix containing the eigenvectors of $\tilde{\mathbf{H}}$ as its columns and $\Lambda = \text{diag}(\lambda_i)$ be the diagonal matrix containing the eigenvalues of $\tilde{\mathbf{H}}$. We then consider the reduced, decoupled system

$$\ddot{\mathbf{x}} + \beta \Lambda \dot{\mathbf{x}} + \Lambda \mathbf{x} = 0. \quad (10)$$

These systems can be solved, given appropriate initial conditions, by using the closed form solutions for each dimension individually. As a post-processing step, we apply *rotation-strain warping*, see [37], to the results of the linearized simulation. For our experiments, we implemented a tool for interactive simulation, where initial velocities can be injected into the system by clicking on vertices of the mesh. Two examples of resulting simulations are included to the supplementary video. When comparing results obtained in subspaces of the same dimension constructed from compressed and natural vibration modes, we obtain results of comparable quality, while the compressed basis takes about 20% of the memory of the dense basis.

By design the compressed modes $\mathbf{u}$ induce low-energy deformations, *i.e.*, $\mathbf{u}^T \mathbf{H} \mathbf{u}$ is small. Due to the localization, the compressed modes are less efficient than the natural vibration modes. However, we expect that the energy of the compressed modes converges to that of the natural modes when μ converges to zero. We provide experimental evidence in Figure 7, where we show that the energy $\mathbf{u}^T \mathbf{H} \mathbf{u}$ of the lowest compressed vibration mode (blue) (which is constrained to be orthogonal to the linearized rigid motions) converges to the eigenvalue of the first natural vibration (again orthogonal to the rigid motions) (orange) when μ goes to zero. A visual evaluation of this convergence can be seen in Figure 3a.

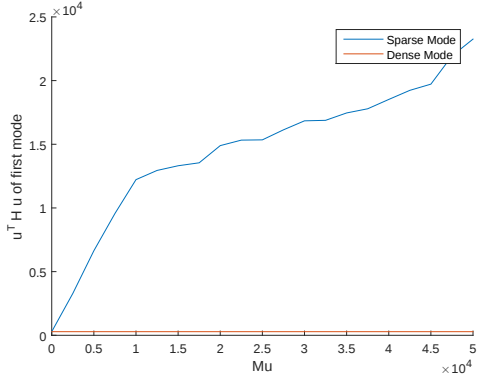


Figure 7: Convergence of sparse vibration modes to natural vibration modes as μ goes to 0.

As a second application of the compressed basis, we use compressed vibration modes in a reduced deformation-based shape editing setup. In Figure 6, we compare the results of such modeling sessions, using a basis of compressed and dense vibration modes of various sizes (10, 20 and 30 modes for the X-mesh; 20, 40 and 60 modes for the dinosaur).

More precisely, we minimize a weighted sum of the St.-Venant-Kirchhoff energy and a least squares term that drags a subset of the vertices to user specified

Mesh	# vertices	K	Energy using compressed modes	Energy using dense modes
X	1098	10	2137.93	424.82
		20	287.33	292.27
		30	109.01	192.17
Dinosaur	27664	20	53.76	372.20
		40	20.47	152.46
		60	12.12	43.93

Table 1: Energy values of optimized static state solutions for the deformations shown in Figure 6.

positions. Instead of minimizing this energy over the full space, we optimize over the subspaces spanned by the compressed and dense vibration modes respectively. The compressed basis requires about 35%/20% of the storage space of the dense basis for the X-mesh and dinosaur mesh, respectively. The energy levels of the optimized solutions are listed in Table 1. It is remarkable that in almost all cases a lower energy deformation was found in the subspace spanned by compressed vibration modes.

In our supplementary video, we show a modeling session using a sparse basis to demonstrate the flexibility of deformations that can be acquired using a basis of compressed vibration modes. We found that it is often more intuitive to achieve a modeling goal using the sparse vibration modes as a reduced basis.

Elastic segmentation. The compressed vibration modes provide us with information about the dynamics of an object: the supports of the modes mark areas which are well suited as segments of the object that undergo deformations while the rest of the shape remains still. The size of these areas, *i.e.* the semantic level that should be covered by the modes, can be controlled via the parameter of the L^1 regularizer, as can be seen, for example, in Figures 3a and 12. This makes the compressed vibration modes well-suited to aid in segmentation tasks, where elastic behavior of the mesh should be considered. We implemented a simple segmentation algorithm, where we segment the mesh according to the supports of the modes by only considering a mode if there is not already another mode that covers it by more than 90%. We decide whether a tetrahedron belongs to the segment defined by a mode, if the deformation vector for this mode is larger than that of other modes that cover a nearby segment. Results of this segmentation scheme can be seen in Figure 8, where we show its performance on a volumetric and a surface mesh, using the modes from the experiments shown above. In both cases, $K = 50$ compressed modes were produced and the steps above lead to the depicted segmentations. The number of segments depends on the parameter μ and the elastic properties of the shape itself. For the dinosaur we end up with 10 segments, while for the centaur we get 18. In both cases, we can see some triangles and tetrahedrons that do not belong to any partition, or segments with rough borders. This could easily be fixed by extending the steps above, but we wanted to highlight the simplicity of this algorithm which

already yields a useful segmentation that is based on elastic properties of the object.

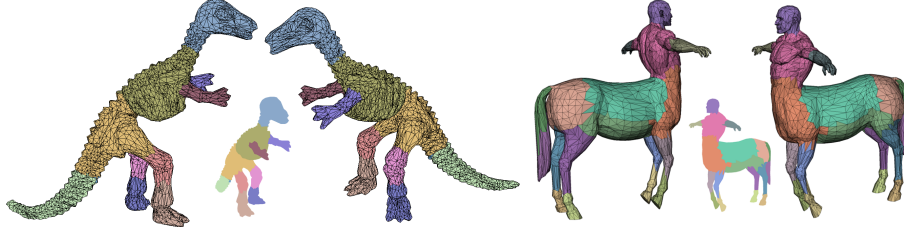


Figure 8: A simple segmentation scheme using our compressed vibration modes applied to the dinosaur (volumetric) and the centaur (surface) mesh. See Section 7 for details.

Segmentation using elasticity has previously been proposed by Huang et al. [18], where the natural vibration modes of an elastic object were weighted and used in a variant of k -means clustering to partition the mesh. The goal of the segmentation is to be able reproduce the 'likely, or typical' deformations of the vibration modes by using rigid transformations on each of the resulting segments. In contrast, we directly produce a set of likely, or typical sparse deformations whose support we use as the segments.

8. Experiments

Consistency of ICCM. A property of ICCM is that it will always converge to some local minimum, up to a desired precision: the sequence of functional values of the $\hat{\mathbf{u}}_{i,k}$ in (7) is monotonically decreasing for each fixed i and increasing k . Indeed, when minimizing the functional to find $\hat{\mathbf{u}}_{i,k}$, the previous solution $\hat{\mathbf{u}}_{i,k-1}$ is within the hyperplane to which the search space is restricted in that iteration, by construction. If a mode with lower energy value is found within the hyperplane, the functional value of the normalized solution will be strictly smaller since the functional is strictly convex. Moreover, each of the sub-problems can be solved robustly and up to a desired precision since the interior point methods provided by solvers like *MOSEK* offer various optimality guarantees for quadratic programs with inequality constraints. For ADMM, such convergence guarantees are not available, and indeed, as stated above, the density, size and structure of the matrices, as given in our setting of compressed vibration modes, prevented ADMM from converging to a meaningful solution for simple models, which brought up the necessity to develop an alternative optimization scheme.

For $\mu = 0$, the compressed vibration modes coincide with the natural vibration modes that are solutions of the generalized eigenvalue problem $\mathbf{H}\mathbf{u} = \lambda\mathbf{M}\mathbf{u}$. In this case, our algorithm can be seen as a scheme that uses convexification to solve an eigenvalue problem and it is important to verify that the solutions of our approach coincide with the ground truth (which can be obtained using reliable eigen-solvers). Indeed, for all meshes used in our experiments, we were able

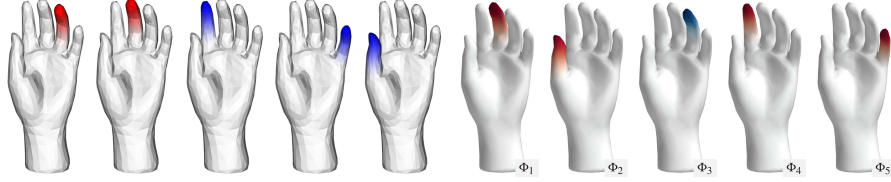


Figure 9: Comparison between the compressed manifold modes produced by our scheme (left) and Neumann et al. [2] (right). Both methods yield the same results up to the ordering and sign changes, while the computation times of our method are significantly lower.

to find the first 20 vibration modes among the first 25 modes produced using our minimization scheme. The reason that we do not get a one-to-one correspondence to the ground truth is that we use a convexification that approximates the constraint $\|\mathbf{u}\|_2 = 1$. This may lead to finding a local minimum, instead of the global minimum. In our experiments, we were still able to find consistent sets of compressed vibration modes, by computing slightly more modes and cutting off the highest frequency modes.

In case $\mu > 0$, we first compare to the compressed manifold modes shown in Neumann et al. [2]: in Figure 9 we show the output of their adapted ADMM optimization scheme and our proposed ICCM scheme. We were able to reproduce the same modes, up to changes in the order in which they were found. This, in combination with the enhanced computation timings shown below, shows ICCM is a valid alternative to ADMM for the problem of finding compressed modes. Moreover, ADMM did not converge after 20000 iterations to any suitable solution when solving for compressed vibration modes: we used the elasticity Hessian and the corresponding mass matrix in the ADMM implementation of Neumann et al. [2] and received the modes shown in Figure 10, which aren't meaningful solutions to the optimization problem. See Figure 1 for a comparison to the output of our method.



Figure 10: Compressed vibration modes produced by using ADMM for $\mu = 10, 20, 30, 50, 75$. See Figure 1 for the output of our method.

Note that for $K > 1$, ICCM and ADMM try to solve two slightly different optimization problems (computing a sequence of modes or computing all modes at once), so the following quantitative comparison of the two algorithms was performed for $K = 1$. Additionally, for this comparison only, ICCM used the unweighted L^1 norm, such that the objective values will be comparable. We performed the comparison on various meshes and different triangulations. The values of μ were chosen such that the resulting modes had a support of about 20% of the mesh area. We used a randomized mode for both methods as an initialization (in ICCM we need an initial hyperplane constraint, ADMM needs an initial value to start an iterative procedure) and ran both algorithms 100 times with different random initializations. In Table 2 we list both the average

as well as the best objective values found. From those values we conclude that in setups where ADMM does converge (which is not always the case, see above), both algorithms perform equally well. Thus, we met our goal of creating an algorithm that is able to compute vibration modes consistently and whose computation times scale linearly in the number of modes that should be produced.

Mesh	# vertices	μ	Obj. value of optimum for ICCM (average,best)	Obj. value of optimum for ADMM [2] (average,best)
Hand	868	0.0001	4.1755E-4, 5.5042E-4	4.2055E-4, 6.0002E-4
Hand	4054	0.0001	9.4314E-4, 1.4113E-3	9.4731E-4, 1.1586E-3
Fertility	4994	0.0002	2.1663E-2, 2.5329E-2	2.1683E-2, 2.6280E-2
Fertility	9994	0.00005	1.2519E-2, 1.3222E-2	1.2531E-2, 1.3815E-2
Bunny	34834	0.0001	5.1078E-2, 6.4719E-2	5.1078E-2, 7.0158E-2

Table 2: Comparison of objective values of the optima computed using our algorithm (ICCM) and using ADMM as proposed and implemented in Neumann et al. [2].

Lastly, note that ADMM requires the choice of a penalty parameter, for which Neumann et al. use an automatic adjustment strategy. It is unclear whether different strategies might lead to meaningful results the examples where ADMM did not converge. Our proposed algorithm is parameter free and converged in all examples for all choices of sparsity regularization and number of modes.

Compressed vibration modes and compressed manifold modes are unstable in their order: usually there is a large set of compressed vibration modes with almost the same objective value, so small changes of the mesh, even isometric deformations, lead to a different order of the sparse eigenmodes. Therefore, none of our applications rely on a precise ordering of the modes. Note, however, that the modes can always be ordered after a certain number has been computed. [23] provides a natural ordering for compressed manifold modes that can be extended to compressed vibration modes as well. In our experiments, we always computed about 10%-25% more modes than required, ordered them and cut off the superfluous modes. This way we were able to get a consistent set of the K lowest compressed vibration modes.

In Figure 1, we show the first sixteen modes of an X-shaped volume mesh. The modes were computed in the order that is shown there, and putting them into groups of four shows how the symmetry of the mesh is properly reflected in the modes: each mode appears four times, once for each leg of the X.

Sparsity control. Examples for compressed vibration modes, where the sparsity term was used as a regularizer are shown in Figures 1, 3a, 3b, 12, 2a and 2b. In Figure 3a, we show how the bar can be segmented into parts of arbitrary size (limited by the mesh resolution) by tuning the sparsity parameter μ . For each of those parts, we get multiple vibrations that span a space of deformations for

that part of the mesh. In case of the dinosaur mesh shown in Figure 2a it is remarkable how the support of the modes is concentrated around parts of the skeleton that we would intuitively categorize as segments.

While for the rest of the experiments we used tetrahedral volume meshes with the St.-Venant-Kirchhoff energy, in Figures 12 and 2b we show that our method is not limited to this setup: there, compressed vibration modes for triangle surface meshes are shown, where the discrete shells energy was used as the underlying deformation energy. Figure 12 also demonstrates how we can localize the modes in different levels of scaling: for $\mu = 0.01$ we get deformations of single fingers as compressed vibration modes and for $\mu = 10^{-4}$ we get deformations of the legs and arms.

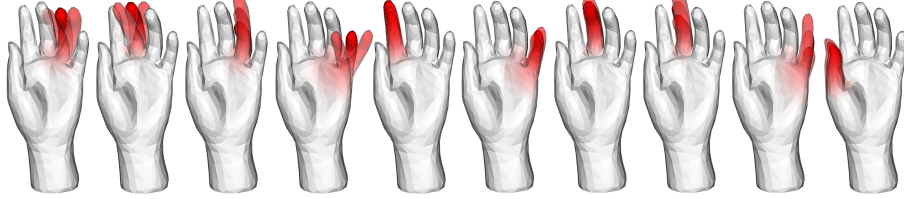


Figure 11: Sparsity constrained vibration modes of a hand mesh.

In Figure 11, we show compressed vibration modes where we constrained the L^1 norm such that the first mode approximately covers one of the fingers. Shown are the first ten modes. Higher modes exhibit more complicated vibrations of the fingers and other parts of the hands are only covered by very high modes. This shows the tendency of compressed vibration modes to concentrate on parts that are easier to deform in isolation (*i.e.* the energy of these deformations is lower than that of comparably sized other parts).

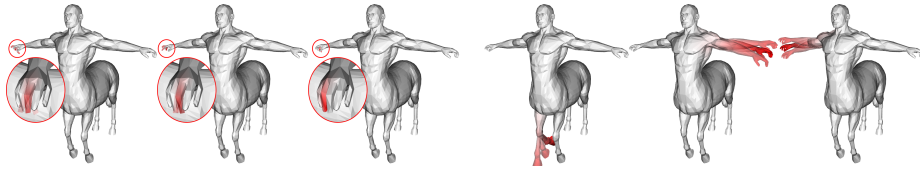


Figure 12: A surface mesh of a centaur with compressed vibration modes of the discrete shells energy. From left to right we show the first modes three for $\mu = 0.01$ and then the first three modes for $\mu = 10^{-4}$

In Figure 13, we show an advantage of both the correct approximation of the L^1 norm as well as the sparsity constrained formulation: we show that modes computed on a model with 28k vertices are similar to those computed on a simplification of that model with 5k vertices when using the same parameter to constrain the L^1 norm. The L^1 norm is only a meaningful measure when taking into account the volume associated to each vertex sample (*i.e.* using the entries of $\mathbf{M}$ when computing the L^1 norm). Also, there is no guarantee that the

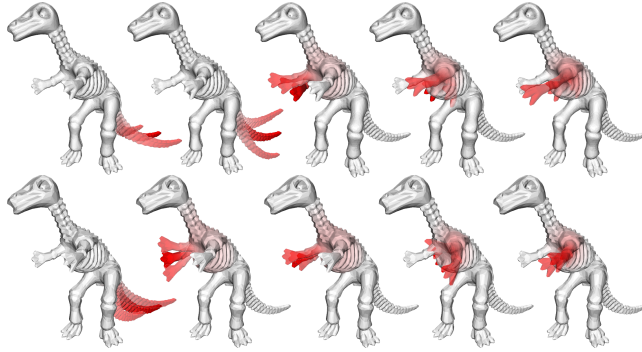


Figure 13: The first five sparsity constrained vibration modes on a high resolution and low resolution version of the dinosaur mesh (same value of s for both meshes).

elastic energies of both models behave the same, and thus a L^1 regularization can not be expected to have a similar effect on the localization of the modes. A L^1 constraint, however, still leads to a similar support of the modes on the two versions of the mesh, independently of the energy levels induced by the elastic energies on them.

Timings. We first show that the computation times of the proposed ICCM scheme scale better when the number of modes, K , increases. Therefore, we compare the timings of our iterative optimization scheme as opposed to using ADMM to find a full set of modes at once. For the latter we use the implementation provided by the authors of [2]. Since ADMM did not converge to satisfying solutions when applied for sparse vibration modes (see previous paragraph), we compare the timings of ICCM and ADMM for the computation of compressed manifold modes (*i.e.* sparse eigenmodes of the surface Laplacian). We used the same custom laptop to measure computation times for both algorithms and use the convergence criteria proposed in [2] for ADMM. When computing the modes using ICCM, we stopped the inner iterations to find a mode, when the changes in the objective functional were below 10^{-8} times the current value of the objective functional.

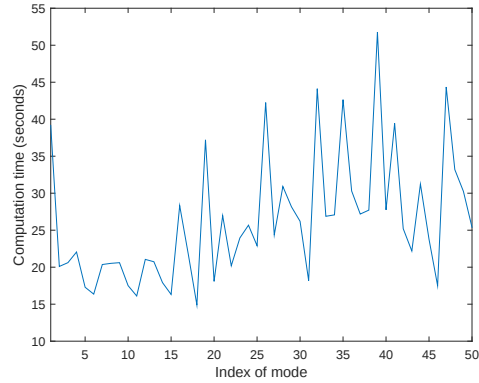


Figure 14: Times for the computation of each individual mode of the dinosaur model shown in Figure 13, lower row, when using ICCM.

As shown before, in Figure 9, for the right values of μ in each algorithm the produced modes are visually indistinguishable (after sign flips and reordering). The resulting timings are listed in Table 3. We provide visual comparisons of the computed modes in Figure 9 and as inset images in Table 3.

For some models and values of K , ADMM did not converge after 20k iterations using the convergence criteria given in the implementation by Neumann et al. [2], we marked these timings in the Table. We get faster computation times in general for smaller models and have a better scaling in computation times when computing a larger number of modes. For large meshes and low values of K , ADMM benefits from the fact that the iterations can be performed at comparably low costs, but as the size of the problem grows, each iteration becomes more costly, and more iterations are required for the method to converge. Thus, ICCM eventually outperforms ADMM as the number of desired modes grows. With our scheme, for each additional mode, we have to solve another sequence of quadratic programs with an additional equality constraint (orthogonality to the last mode). Whenever an additional equality constraint is added to the system, a change of variables can be performed, such that the variables automatically fulfill the constraints. As a result, the computation time per mode remains approximately constant, even for higher modes. In contrast, when using ADMM, as proposed in [2], the number of modes K that have to be computed has to be fixed before starting the computation and the number of variables is $N \cdot K$, where N is the number of vertices of the mesh. During minimization several steps are being performed that scale superlinear in the number of modes. Additionally, in our tests, ADMM requires more iterations in order to converge when the number of modes is high.

An overview of the computation times for several examples of compressed vibration modes can be found in Table 4. The computation times strongly depend on the size of the model and the number of modes, but also on the shape itself and the resulting elasticity Hessian. However, as pointed out before, the time to compute each mode is roughly the same, which means that computing

Mesh	# vertices	times for $K = 10$ (us / ADMM [2])	times for $K = 40$ (us / ADMM [2])	times for $K = 80$ (us / ADMM [2])
Hand See Figure 9	868	2.26s / 4.17s	12.47s / 144.64s*	23.25s / 435s*
Fertility 	4994	32s / 30s	123s / 422s	254s / 899s
Bunny 	34834	1007s / 312s	4072s / 3459.82s	8110s / 17205s*

Table 3: Comparison of computation times for compressed manifold modes when using ICCM (our scheme) versus the modified ADMM algorithm, as proposed in [2]. The inset pictures show the first three modes of each method, first ours, then ADMM.

*: ADMM did not converge after 20k iterations under the convergence criteria given in the implementation by Neumann et al.

Mesh	# vertices	constrained?	parameter	average time per mode	average # of iterations per mode	rel. size of sparse versus dense basis
Bar (Fig. 3)	418	no	$\mu = 512$	1.97s	12.06	38%
Bar (Fig. 5)	418	yes	$s = 11.5$	2.1s	11.02	38%
Centaur (Fig. 12)	2645	no	$\mu = 0.01$	16.13s	8	2%
Hand (Fig. 11)	2584	yes	$s = 6.46$	14.72s	17.51	17%
Dinosaur lo-res. (Fig. 13, lower row)	4713	no	$\mu = 20$	25.17s	10.22	16%
Dinosaur hi-res. (Fig. 13, upper row)	27664	yes	$s = 6.5$	518.5s	13.4	16%

Table 4: Computation times for different sets of compressed vibration modes when using our proposed algorithm.

a large number of modes is entirely possible. This is shown in Figure 14 where we show the computation times for each individual mode of the dinosaur mesh.

Computation of the Hessian. Before (compressed) vibration modes can be computed the matrix $\mathbf{H}$ has to be set up. In the following, we list several options on how to compute the Hessian. For a general discrete elastic energy, once the evaluation of the energy with respect to a fixed rest-shape $\mathbf{x}$ and a variable displacement vector $\mathbf{u}$ is implemented as a function that takes an array of $3|\mathcal{V}|$ values and returns a single value denoting the energy associated to this displacement, one can use automatic differentiation (*e.g.* ADOL-C, see [38]) in order to obtain the Hessian matrix. Often, however, the Hessian can be evaluated faster by using explicit formulas. For energies that use discrete bending forces, Tamstorf et al. [39] documented closed form expressions for the Hessian of the bend angle, which allows explicit formulas for the energy Hessians of a large class of elastic energies. The Hessian of the discretized St. Venant–Kirchhoff energy for tetrahedral meshes is available as part of the Vega FEM library [40]. Also note, that for our purposes, we only require the evaluation of the energy Hessian at the rest configuration. A simple formula for the Hessian at the rest configuration for a certain class of deformation energies is detailed in [16].

9. Conclusion

We introduce compressed vibration modes of elastic bodies, which are orthonormal systems of displacements of objects that induce local and low-energy deformations. For the computation of the modes, we devise a novel minimization scheme which proves to be stable and the resulting computation time scales linearly in the number of modes, as opposed to previous methods. The compressed vibration modes are shown to be intuitively controllable via either tuning the sparsity regularizer or imposing a sparsity constraint (the latter of which has not been done before for comparable modes). The modes are shown to be stable under refinement and a correction of the L^1 term leads to invariance of the triangulation.

We show how the modes can extend applications such as reduced elastic simulation and deformation or mesh segmentation. Their compressible structure is attractive for cases in which memory limitations are present (*e.g.* games and VR, where GPU memory has to be handled efficiently). Moreover they tend

to exhibit a semantic structure, in the sense that the support of the modes is often located on intuitive parts of objects, such as hands, arms, legs or fingers, depending on the choice of μ .

Challenges and limitations. We are convinced that compressed vibration modes can be a useful tool in various areas of geometry processing and we can see direct benefits for simulation, shape space exploration and mesh analysis. While computation times scale linearly, the timings per mode for detailed meshes is quite high, and does not allow for interactive re-computation of the modes, under different parameters or different rest configurations. Thus, it might be worthwhile to further investigate alternative L^1 optimization schemes, or to provide a multi-resolution scheme for the proposed ICCM optimization.

Another direction of future work might be to exchange the L^1 regularization by a regularization that penalizes the volume of the support of the modes. This would lead to a mass weighted ℓ_0 problem. Solving such problem poses a challenging problem.

Acknowledgements

We want to thank Razvan-Florin Hulea for carrying out experiments that served as a motivation for this project and the anonymous reviewers for their helpful comments and suggestions. For our experiments, we used models from the Non-Rigid World [41] and TOSCA [42] data sets, Cyberware 3D-scan data samples, Christian Schulz, and the Aim@Shape repository.

- [1] V. Ozoliņš, R. Lai, R. Caflisch, S. Osher, Compressed modes for variational problems in mathematics and physics, *Proceedings of the National Academy of Sciences* 110 (46) (2013) 18368–18373.
- [2] T. Neumann, K. Varanasi, C. Theobalt, M. Magnor, M. Wacker, Compressed manifold modes for mesh processing, *Comput. Graph. Forum* 33 (5) (2014) 35–44.
- [3] D. Boscaini, J. Masci, S. Melzi, M. M. Bronstein, U. Castellani, P. Vndergheynst, Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks, *Computer Graphics Forum* 34 (5) (2015) 13–23.
- [4] A. Kovnatsky, K. Glashoff, M. M. Bronstein, MADMM: a generic algorithm for non-smooth optimization on manifolds, *arXiv preprint arXiv:1505.07676*.
- [5] K. Hildebrandt, C. Schulz, C. von Tycowicz, K. Polthier, Interactive surface modeling using modal analysis, *ACM Transactions on Graphics* 30 (5) (2011) 119:1–119:11.

- [6] K. K. Hauser, C. Shen, J. F. O'Brien, Interactive deformation using modal analysis with constraints., in: *Graphics Interface*, Vol. 3, 2003, pp. 16–17.
- [7] J. Barbič, D. L. James, Real-time subspace integration for St. Venant–Kirchhoff deformable models, *ACM Trans. Graph.* 24 (3) (2005) 982–990.
- [8] C. von Tycowicz, C. Schulz, H.-P. Seidel, K. Hildebrandt, An efficient construction of reduced deformable objects, *ACM Trans. Graph.* 32 (6) (2013) 213:1–213:10.
- [9] Y. Yang, D. Li, W. Xu, Y. Tian, C. Zheng, Expediting precomputation for reduced deformable simulation, *ACM Trans. Graph.* 34 (6) (2015) 243:1–243:13.
- [10] J. N. Chadwick, S. S. An, D. L. James, Harmonic shells: A practical non-linear sound model for near-rigid thin shells, *ACM Trans. Graph.* 28 (5) (2009) 119:1–119:10.
- [11] D. Li, Y. Fei, C. Zheng, Interactive acoustic transfer approximation for modal sound, *ACM Trans. Graph.* 35 (1) (2015) 2:1–2:16.
- [12] J. Barbič, F. Sin, E. Grinspun, Interactive editing of deformable simulations, *ACM Trans. Graph.* 31 (4) (2012) 70:1–70:8.
- [13] K. Hildebrandt, C. Schulz, C. von Tycowicz, K. Polthier, Interactive space-time control of deformable objects, *ACM Trans. Graph.* 31 (4) (2012) 71:1–71:8.
- [14] C. Schulz, C. von Tycowicz, H.-P. Seidel, K. Hildebrandt, Animating deformable objects using sparse spacetime constraints, *ACM Transactions on Graphics (TOG)* 33 (4) (2014) 109:1–109:10.
- [15] S. Li, J. Huang, F. de Goes, X. Jin, H. Bao, M. Desbrun, Space-time editing of elastic motion through material optimization and reduction, *ACM Trans. Graph.* 33 (4) (2014) 108:1–108:10.
- [16] K. Hildebrandt, C. Schulz, C. von Tycowicz, K. Polthier, Eigenmodes of surface energies for shape analysis, in: *Advances in Geometric Modeling and Processing*, Vol. 6130 of *Lecture Notes in Computer Science*, Springer, 2010, pp. 296–314.
- [17] K. Hildebrandt, C. Schulz, C. von Tycowicz, K. Polthier, Modal shape analysis beyond Laplacian, *Computer Aided Geometric Design* 29 (5) (2012) 204–218.
- [18] Q.-X. Huang, M. Wicke, B. Adams, L. Guibas, Shape decomposition using modal analysis, *Computer Graphics Forum* 28 (2) (2009) 407–416.
- [19] T. R. Langlois, S. S. An, K. K. Jin, D. L. James, Eigenmode compression for modal sound models, *ACM Trans. Graph.* 33 (4) (2014) 40:1–40:9.

- [20] V. Ozoliņš, R. Lai, R. Caffisch, S. Osher, Compressed plane waves yield a compactly supported multiresolution basis for the Laplace operator, *Proceedings of the National Academy of Sciences* 111 (5) (2014) 1691–1696.
- [21] A. Mackey, H. Schaeffer, S. Osher, On the compressive spectral method, *Multiscale Modeling & Simulation* 12 (4) (2014) 1800–1827.
- [22] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, Distributed optimization and statistical learning via the alternating direction method of multipliers, *Foundations and Trends in Machine Learning* 3 (1) (2011) 1–122.
- [23] K. Houston, Compressed manifold modes: Fast calculation and natural ordering, *arXiv preprint arXiv:1507.00644*.
- [24] A. M. Bronstein, Y. Choukroun, R. Kimmel, M. Sela, Consistent discretization and minimization of the L^1 norm on manifolds, *arXiv preprint arXiv:1609.05434*.
- [25] X.-T. Yuan, T. Zhang, Truncated power method for sparse eigenvalue problems, *Journal of Machine Learning Research* 14 (2013) 899–925.
- [26] M. Meyer, J. Anderson, Key point subspace acceleration and soft caching, *ACM Trans. Graph.* 26 (3) (2007) 74:1–74:8.
- [27] T. Neumann, K. Varanasi, S. Wenger, M. Wacker, M. Magnor, C. Theobalt, Sparse localized deformation components, *ACM Trans. Graph.* 32 (6) (2013) 179:1–179:10.
- [28] L. Gao, G. Zhang, Y. Lai, L^p shape deformation, *Science China Information Sciences* 55 (5) (2012) 983–993.
- [29] L. He, S. Schaefer, Mesh denoising via l_0 minimization, *ACM Trans. Graph.* 32 (4) (2013) 64:1–64:8.
- [30] H. Kang, F. Chen, Y. Li, J. Deng, Z. Yang, Knot calculation for spline fitting via sparse optimization, *Computer-Aided Design* 58 (2015) 179–188.
- [31] C. Brandt, H.-P. Seidel, K. Hildebrandt, Optimal spline approximation via ℓ_0 -minimization, *Computer Graphics Forum* 34 (2) (2015) 617–626.
- [32] L. Xu, R. Wang, J. Zhang, Z. Yang, J. Deng, F. Chen, L. Liu, Survey on sparsity in geometric modeling and processing, *Graphical Models* 82 (2015) 160 – 180.
- [33] J. Bonet, R. D. Wood, *Nonlinear Continuum Mechanics for Finite Element Analysis*, Cambridge University Press, 2008.
- [34] E. Grinspun, A. N. Hirani, M. Desbrun, P. Schröder, Discrete shells, in: *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2003, pp. 62–67.

- [35] R. Tibshirani, Regression shrinkage and selection via the lasso, *Journal of the Royal Statistical Society. Series B (Methodological)* (1996) 267–288.
- [36] R. M. Rustamov, Multiscale biharmonic kernels, *Computer Graphics Forum* 30 (5) (2011) 1521–1531.
- [37] J. Huang, Y. Tong, K. Zhou, H. Bao, M. Desbrun, Interactive shape interpolation through controllable dynamic deformation, *IEEE Transactions on Visualization and Computer Graphics* 17 (7) (2011) 983–992.
- [38] A. Griewank, D. Juedes, J. Utke, Algorithm 755: ADOL-C: A package for the automatic differentiation of algorithms written in C/C++, *ACM Trans. Math. Softw.* 22 (2) (1996) 131–167.
- [39] R. Tamstorf, E. Grinspun, Discrete bending forces and their Jacobians, *Graphical Models* 75 (6) (2013) 362 – 370.
- [40] F. S. Sin, D. Schroeder, J. Barbič, Vega: Non-linear fem deformable object simulator, *Computer Graphics Forum* 32 (1) (2013) 36–48.
- [41] A. M. Bronstein, M. M. Bronstein, R. Kimmel, Efficient computation of isometry-invariant distances between surfaces, *SIAM Journal on Scientific Computing* 28 (5) (2006) 1812–1836.
- [42] A. M. Bronstein, M. M. Bronstein, R. Kimmel, *Numerical geometry of non-rigid shapes*, Springer, 2008.

Appendix A. Analysis of the ICCM scheme for the L^1 constrained optimization

As part of ICCM, we bound the weighted L^1 norm of a mode $\mathbf{u} = \mathbf{u}^+ - \mathbf{u}^-$ with $\mathbf{u}^+, \mathbf{u}^- > 0$ from above, via the linear term $\ell(\mathbf{u}^+, \mathbf{u}^-) = \mathbf{1}^T \mathbf{M} \mathbf{u}^+ + \mathbf{1}^T \mathbf{M} \mathbf{u}^-$. When computing sparse vibration modes via ICCM, in case the L^1 term is used as a regularizer, the minima found in each iteration have an L^1 norm that is equal to $\ell(\mathbf{u}^+, \mathbf{u}^-)$. In the following, we prove that this also holds for the L^1 constrained problem. Note that $\|\mathbf{u}^+ - \mathbf{u}^-\| = \ell(\mathbf{u}^+, \mathbf{u}^-)$ if for each i it is either $\mathbf{u}_i^+ = 0$ or $\mathbf{u}_i^- = 0$, i.e. we need to show that for no coordinate both the positive and the negative part of the variables are greater than 0.

In the constrained optimization, after having computed the first $i-1$ modes, the inner loop of ICCM has the following form (where $\mathbf{c}$ is the current hyperplane, used to convexify the L^2 constraint):

$$(\mathbf{u}^+, \mathbf{u}^-) := \begin{cases} \arg \min_{\mathbf{u}^+ - \mathbf{u}^-} \begin{pmatrix} \mathbf{u}^+ \\ \mathbf{u}^- \end{pmatrix}^T \begin{pmatrix} \mathbf{H} & -\mathbf{H} \\ -\mathbf{H} & \mathbf{H} \end{pmatrix} \begin{pmatrix} \mathbf{u}^+ \\ \mathbf{u}^- \end{pmatrix} \\ \text{subject to} \quad \begin{pmatrix} \mathbf{u}^+ \\ \mathbf{u}^- \end{pmatrix}^T \begin{pmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{pmatrix} \begin{pmatrix} \mathbf{c} \\ -\mathbf{c} \end{pmatrix} = 1 \\ \text{and } \forall j < i : \begin{pmatrix} \mathbf{u}^+ \\ \mathbf{u}^- \end{pmatrix}^T \begin{pmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{pmatrix} \begin{pmatrix} \mathbf{u}_j \\ -\mathbf{u}_j \end{pmatrix} = 0 \\ \text{and } \mathbf{u}^+, \mathbf{u}^- \geq 0 \\ \text{and } \ell(\mathbf{u}^+, \mathbf{u}^-) = s \end{cases} \quad (\text{A.1})$$

In the following we make the assumptions that there is indeed a feasible solution (in particular s is large enough to allow $\begin{pmatrix} \mathbf{u}^+ \\ \mathbf{u}^- \end{pmatrix}^T \mathbf{M} \begin{pmatrix} \mathbf{c} \\ -\mathbf{c} \end{pmatrix} = 1$) and that we choose s smaller than the L^1 norm of the solution of the same problem without the constraint $\ell(\mathbf{u}^+, \mathbf{u}^-) = s$. We will now show the following: either we have that $\ell(\mathbf{u}^+, \mathbf{u}^-) = \|\mathbf{u}^+ - \mathbf{u}^-\|_1$ or there is a mode $\tilde{\mathbf{u}}$ with L^1 norm smaller than s but with the same value for the quadratic term. Indeed, suppose that there is j , such that $\mathbf{u}_j^+ > \mathbf{u}_j^- > 0$ (the case $\mathbf{u}_j^- > \mathbf{u}_j^+ > 0$ can be handled equivalently). Let $\tilde{\mathbf{u}} = \tilde{\mathbf{u}}^+ - \tilde{\mathbf{u}}^-$, where $\tilde{\mathbf{u}}_i^+ = \mathbf{u}_i^+$ and $\tilde{\mathbf{u}}_i^- = \mathbf{u}_i^-$ for $i \neq j$ and $\tilde{\mathbf{u}}_j^+ = \mathbf{u}_j^+ - \mathbf{u}_j^-$ and $\tilde{\mathbf{u}}_j^- = 0$. Note that $\ell(\tilde{\mathbf{u}}_j^+, \tilde{\mathbf{u}}_j^-) = \|\tilde{\mathbf{u}}^+ - \tilde{\mathbf{u}}^-\|_1$ and that $\tilde{\mathbf{u}}$ still satisfies all constraints from (A.1) and that it has the same value for the quadratic term (this is clear by construction of the energy functional and constraints after splitting the variables into positive and negative parts).

This however would imply, that in the optimization problem (A.1), when we replace the constraint $\ell(\mathbf{u}^+, \mathbf{u}^-) = s$ by the constraint $\|\mathbf{u}^+, \mathbf{u}^-\|_1 = s$, this constraint would not be active, which contradicts our assumption, that s was chosen smaller than the L^1 norm of the solution of (A.1) without the constraint $\ell(\mathbf{u}^+, \mathbf{u}^-) = s$.