Stable Explicit Integration of Deformable Objects by Filtering High Modal Frequencies

Basil Fierz Jonas Spillmann Matthias Harders
(bfierz | jonas.spillmann | mharders)@vision.ee.ethz.ch
Computer Vision Laboratory
ETH Zurich, Switzerland

ABSTRACT

In a traditional finite element method for the simulation of deformable objects, the stiffness matrix depends on the shape of the tetrahedral elements. Ill-shaped elements containing large or small dihedral angles lead to an arbitrary large condition number of the stiffness matrix, thus slowing down the simulation. In addition, high modal frequencies cannot be simulated stably if an explicit numerical time-integration scheme is considered. We propose an approach consisting of two components to address this problem: First, we isolate the ill-shaped tetrahedra by performing an eigenvalue decomposition, and then remove their high modal frequencies by directly altering their element stiffness matrices. This makes the elements softer in some directions. To prevent element inversion, we define constraints that rigidify the object along those directions. The fast projection method, implemented as a velocity filter, is employed to enforce the constraints after the temporal evolution. With our approach, a significantly larger time step can be chosen in explicit integration methods, resulting in a faster simulation.

Keywords: physically based-modeling, deformable objects, modal analysis, explicit integration, constraints

1 INTRODUCTION

The modeling and dynamic simulation of deformable objects is an active field of research in computer animation. To accomplish this, the domain of the object is commonly discretized into tetrahedral elements. Then, we assume a linear stress-strain relationship and use the finite element method (FEM) to compute the restitution forces which work against the deformation. By assuming that the nodes of the mesh constitute mass points, the equations of motion can be solved numerically to evolve the object in time.

In most standard FEM codes, the internal forces are related to the deformed geometry of an element, and work in the direction of its undeformed geometry. Consequently, the condition of the element stiffness matrix relies heavily on the shape of the element. More precisely, ill-shaped elements with particularly small or large dihedral angles result in a large condition number of the corresponding stiffness matrix, as pointed out by, *e. g.*, Shewchuk [She02a].

In turn, poorly conditioned stiffness matrices cause severe problems for the underlying numerical solver: If implicit integration methods are considered, then the convergence of an iterative solver is slowed down, resulting in a significant computational overhead. In the context of explicit integration methods, small time steps are necessary to reproduce the corresponding high frequencies and thus to stabilize the simulation. This is particularly problematic in the context of real-time simulations that require the maximum time spent to compute one *simulated second* to be less than one *real second*. The time required for the force computation and numerical integration is not dependent on the time step, therefore smaller time steps increase the time to compute one simulated second.

There exist two ways to attack this problem, notably by considering the *geometry*, or by considering the *simulation*. The easiest way to influence the geometry is by employing an appropriate meshing approach that avoids ill-shaped tetrahedra such as [LS07]. However, especially in the context of frequent topology changes in cutting simulations, this can be hard to achieve in real-time. In contrast, changing the geometry of particularly ill-shaped elements locally often results in a cascade of topological operations that propagate through the mesh, which are difficult to control [KS07]. In contrast, approaches that consider the simulation usually limit the strain and thus inhibit degenerate elements [BFA02, TPS09]. However, in the context of an explicit time-integration scheme, these methods cannot prevent instabilities completely.

In this paper, we propose a novel approach to stabilize a simulation of deformable objects in combination with an explicit time-integration method, thereby especially addressing the case of frequent topological changes such as cutting and fracturing. Our approach is a combination of two components: First, we assume that the time step of the simulation is given, based, *e. g.*, on real-time considerations. In cutting simulations, we instead demand that the time step stays constant throughout the simulation. We now identify those elements which cannot be simulated stably with that

time step. We determine the directions of the problematic vibration modes by performing an eigenvalue decomposition of the element stiffness matrix. Then, we assemble a new stiffness matrix by filtering the corresponding modal frequencies.

In order to account for the changed material properties, and to prevent element inversion of the softened elements, the second component rigidifies those elements along the determined directions. To accomplish this, we define a set of constraints and employ the fast projection method [GHF*07], implemented as a velocity filter, to enforce the constraints. We show that by employing our stabilization filter, an explicit solver can take significantly larger time steps than traditional approaches, thus economizing the additional cost of the constraint enforcement phase many times over. In addition, our method avoids the headaches that come with manually identifying and editing ill-shaped elements. Various experiments underline the benefits of our method.

2 RELATED WORK

Since most schemes for the simulation of deformable objects relate the restitution forces to the shapes of the elements, there exist a wealth of approaches handling ill-shaped elements. The approaches can be grouped into those that avoid the creation of ill-shaped elements, and those that reduce their influence on the simulation stability.

The easiest way to handle ill-shaped elements is to prevent them completely, by only generating well-shaped elements at the meshing stage. While Delaunay-based meshing approaches [She98, ACSYD05] and advancing front approaches [Sch97] obtain a very good *average* tet quality, they cannot guarantee not to create any ill-shaped tets. Unfortunately, *one* ill-shaped tet is enough to destabilize the simulation. This problem is alleviated by the approach of Labelle *et al.* [LS07]. They compute a distance field and then fill the domain with tets selected from a list of stencils. Their main benefit is that they can give guarantees on the tet qualities. However, employing their approach to remesh domains on the fly (as in [KFCO06, WT08]) is currently not possible in real-time.

As an alternative to global remeshing, it is also possible to identify ill-shaped elements and use node-snapping and edge-swapping techniques to improve their quality [ISS04, CDRR04]. Recently, Klingner *et al.*have proposed to optimize the mesh using a sequence of different optimization steps [KS07]. However, the resulting topological changes tend to be computationally intensive thus prohibiting their use in real-time simulations.

Instead of the geometry, the underlying simulation can be considered. In the field of dynamic linear systems there has been some work in trying to stabilize explicit integration schemes by determining a suitable time step [GK07, Shi04]. Both look at the properties of the dynamic system and the integration scheme. They determine a time step based on the spectral radius of the matrix of a linear solver. However, this requires calculating the eigenvalues of the system matrix, which is not feasible for interactively changing topologies. Schmedding *et al.* [RS09] define a dynamic damping term such that the simulation is stable. While their method is suitable for mass-spring systems, the extension to FEM-based deformations is not straightforward.

In order to determine the dynamic response of a deformable object to a prescribed deformation, the model frequencies of the system matrix can be computed. This has been done before by Pentland and Williams [PW89] and Hauser *et al.* [HSO03] who identified high modal frequencies as source of small amplitude vibrations causing unpleasant visible artifacts. Moreover, the high modal frequencies require small time steps. As a result they removed the high modal frequencies from the simulation in order to improve the simulation speed. However, their methods require pre-computing the vibration modes for the stiffness matrix. Real-time cutting and fracturing is not feasible using these methods, because calculating the modes online would be too expensive. In contrast to the cited works, we perform the modal analysis on a per-element basis, which results in a smaller system to be solved.

To constrain the deformation of the relaxed elements we filter the velocities, such that deformations in these elements are limited. In doing so, we build on an approach recently proposed by Thomaszewski *et al.* [TPS09]. They extend this concept to continuum based constraints and as such define a neo-hookean material law. This new material is linear within the constraints, and inextensible otherwise. While Thomaszewski *et al.* limit the strain of *all* elements, we identify the ill-shaped elements beforehand and only consider their deformation. Therefore, we obtain a significantly smaller system of constraints whose solution is feasible in real-time.

3 OVERVIEW

Our approach is a combination of two components, notably a method which determines the ill-shaped tets and reduces their high modal frequencies, and a velocity filter which rigidifies these tets afterwards. In this section, we give a short overview of our approach.

As input, we assume an object whose domain is discretized into tetrahedral elements. In addition, we assume that the material properties (Young's modulus, Poisson ratio and density) of the object are given. Further, we assume that an explicit integration method such as the explicit Euler or the Verlet scheme is employed to numerically evolve the object in time.

As a first step, we choose a minimum time step Δt_{min} for our simulation. This time step can be chosen heuristically, *e. g.*, based on the expected time t_{exp} to compute one simulation step of this object on a given hardware. Based on this time, the minimum allowable time step to simulate the object in real-time can be derived. Alternatively, in case of a cutting simulation, we just demand that a time step Δt_{min} chosen at the start of the simulation does not change during the simulation, even though

the cuts might induce ill-shaped tets requiring a smaller time step.

We then identify the elements which cannot be simulated with Δt_{min}, *i. e.*, which have one vibration mode exceeding the maximum allowable modal frequency. This process requires the solution of an Eigenvalue problem of size 12×12 for each element, as detailed in Section 4.3. We then filter the modal frequencies of the found elements along the affected directions by directly altering the stiffness matrices. This process is done *before* the computation of the restitution forces and time evolution.

The second component acts as a velocity filter, which is executed *after* the time evolution. Since we have filtered the modal frequencies of some ill-shaped tets, these elements are particularly sensitive to element inversion. In order to prevent this, we rigidify the elements along the relaxed directions. That is, we define geometric constraints on the positions, and employ a projection method to enforce these constraints directly. The result of the velocity filter are new positions and velocities satisfying the constraints, as discussed in Section 5.

4 FILTERING MODAL FREQUENCIES

In this section, we describe the first part of our approach, which determines the ill-shaped tets and relaxes the material along the bad directions. To accomplish this, we first derive an analytical relation between the element stiffness matrix and the time step of the underlying numerical integration. Equipped with this knowledge, we can determine the bad directions, and the required amount of material relaxation.

4.1 Finite element method

We assume that the restitution forces are computed by employing a linear explicit FEM with warped stiffness [MG04]. We decompose the simulation domain into tetrahedral elements. Within each tetrahedral element, linear shape functions $\mathbf{N}_i, i = 1 \ldots 4$ are employed to interpolate the displacement field $\mathbf{u}$ with $\mathbf{u}(\mathbf{x}) = \mathbf{N}_i(\mathbf{x})\mathbf{u}_i$, where $\mathbf{u}_i = \mathbf{x}_i - \mathbf{x}_{i,0}$ is the displacement of the node i with undeformed position $\mathbf{x}_{i,0}$. The elastostatic equation states that the divergence of the stress σ and the external forces $\mathbf{f}^e$ form an equilibrium,

$$\nabla \cdot \sigma + \mathbf{f}^e = 0. \tag{1}$$

Applying the Galerkin discretization to the variational formulation of (1), we obtain the description of the stress σ_e for each element e:

$$\sigma_e = \mathbf{E}\varepsilon = \mathbf{E}\mathbf{B}_e\mathbf{u}, \tag{2}$$

where $\mathbf{E}$ is the material-dependent elasticity matrix, ε is the Cauchy strain [Hug87], and $\mathbf{B}_e$ is a constant matrix depending on the gradient of the shape functions $\mathbf{N}_i$. The restitution forces of each element are then linearly related to its nodal displacements using the stiffness matrix $\mathbf{K}_e = V_e\mathbf{B}_e^T\mathbf{E}\mathbf{B}_e$, where V_e is the volume of the element e. The global stiffness matrix $\mathbf{K}$ is the superposition of the element stiffness matrices $\mathbf{K}_e$, $\mathbf{K} = \sum_e \mathbf{K}_e$.

Shewchuk [She02a] showed that the condition of $\mathbf{K}_e$ depends on the cotangent of the dihedral angles of the tetrahedron e. If the angles approach 0° or 180°, the cotangent goes to infinity, which results in an arbitrary large condition number of $\mathbf{K}_e$. As a consequence, the condition number of the global stiffness matrix $\mathbf{K}$ depends on the condition numbers of the element stiffness matrices $\mathbf{K}_e$.

In a static FE analysis, we are interested in solving for the unknown deformed nodal positions $\mathbf{x}$, given prescribed boundary conditions. In this case, the problem amounts to solving

$$\mathbf{K}\mathbf{x} = \mathbf{f} \tag{3}$$

for the unknown vector $\mathbf{x}$. As Shewchuk pointed out, the convergence rate of an iterative solver for the linear system decreases with increasing condition number of $\mathbf{K}$, and thus, for ill-shaped tets, the solution of (3) becomes arbitrary expensive.

4.2 Dynamic finite element analysis

In the following, we are focusing on the dynamic simulation of a deformable object. In this case, the equations of motion of the object are

$$\mathbf{M}\ddot{\mathbf{x}} + \mathbf{C}\dot{\mathbf{x}} + \mathbf{K}(\mathbf{x} - \mathbf{x}_0) = \mathbf{f}^e, \tag{4}$$

where $\mathbf{M}$ is the mass and $\mathbf{C}$ the damping matrix. For simplicity and as usual for most problems in computer animation, we assume that the mass is lumped in the nodes, resulting in a diagonal mass matrix. Furthermore, we limit the analysis to the case of an undamped system, *i. e.*, $\mathbf{C} = 0$. We will consider damping as future work. Nevertheless, we can employ viscous point damping without sacrificing our theoretical bounds.

To solve (4), the second order ordinary differential equation (ODE) is transformed into a system of first order ODEs, which are then numerically integrated in order to obtain the trajectories of the mass points. In our case, we assume an explicit time integration scheme [QSS00].

In order to characterize the stability of (4), we employ the Courant-Friedrichs-Lewy (CFL) condition from [She02b], which relates the temporal resolution, *i. e.* the time step of the numerical integration, to the spatial resolution, *i. e.*, the discretization of the mesh. The CFL condition stems from the observation that if a wave (*e. g.*, a pressure wave in our case) is crossing a discrete grid, then the time step must be less than the time for the wave to propagate through an element. More precisely, the CFL condition relates the time step of the numerical integration to the vibration mode of the underlying discretization, which is in turn inversely proportional to the propagation time of a pressure wave:

$$\Delta t < \frac{c}{\sqrt{\omega_{max}^2}}, \tag{5}$$

where Δt is the time step of the numerical integration and ω_{max} is the largest modal frequency of the body. The constant c corresponds to the stability radius of the state transition matrix [QSS00] of the underlying numerical integration scheme. In addition, c may also depend on the damping coefficient. Detailed derivations for different solvers can be found throughout the literature. Gurfil and Klein [GK07] studied the explicit Euler integration and derived $c = 2\xi$, where ξ is the damping coefficient. The explicit Euler is never stable if no damping is present. Müller *et al.* [MHTG05] present a detailed derivation for the leap frog integration scheme without damping, which results in $c = 2$. For the Verlet scheme, Klein [Kle07] derives $c = 2$ for the undamped and $c = 2(\sqrt{\xi^2+1} - \xi)$ for the damped integration scheme.

In order to determine the vibration modes of the mesh, (4) has to be transformed to a diagonal form, which is accomplished with the *whitening transform* [PW89]. This results in the generalized Eigenvalue problem

$$\Lambda\Phi = (\mathbf{M}^{-1}\mathbf{K})\Phi. \tag{6}$$

where the diagonal matrix Λ contains the eigenvalues of $\mathbf{M}^{-1}\mathbf{K}$, and the columns of the matrix Φ correspond to the eigenvectors of $\mathbf{M}^{-1}\mathbf{K}$. The eigenvalues $\lambda_i = [\Lambda]_{ii}$ correspond to the squared modal frequencies ω_i^2, and the eigenvectors $[\Phi]_i$ correspond to the directions along which the vibrations work. By plugging $\omega_{max}^2 = \lambda_{max} := \max_i \lambda_i$ into (5), we can determine the maximum time step allowed for the stable simulation of the object.

However, we are not exactly where we want to be. Firstly, we can now estimate a maximum time step for the whole object, without knowing which elements are particularly ill-shaped. However, as a single tetrahedron may make a simulation unstable, we must determin such tets. Secondly, the numerical solution of (6) is expensive and cannot be done in real-time. Thus, in the following, we show how to approximate (6) by looking at each element in isolation.

4.3 Element vibration modes

Fried [Fri72] showed that it is possible to define a reasonable upper bound for the eigenvalues of $\mathbf{M}$ and $\mathbf{K}$ by considering each element e in isolation. These upper bounds depend on the maximal eigenvalues λ_{max} of the element matrices $\mathbf{M}_e, \mathbf{K}_e$ and the maximum vertex degree n of the mesh,

$$\begin{aligned} \max_e \lambda_{max}(\mathbf{K}_e) \le \lambda_{max}(\mathbf{K}) \le n \max_e \lambda_{max}(\mathbf{K}_e) \\ \max_e \lambda_{max}(\mathbf{M}_e) \le \lambda_{max}(\mathbf{M}) \le n \max_e \lambda_{max}(\mathbf{M}_e) \end{aligned} \tag{7}$$

where $\lambda_{max}(\cdot)$ corresponds to the maximum eigenvalue of the matrix passed as argument. By analogy of [Fri72], Shewchuk derived a similar estimation,

$$\max_e \lambda_{max}(\mathbf{J}_e) \le \lambda_{max}(\mathbf{M}^{-1}\mathbf{K}) \le n \max_e \lambda_{max}(\mathbf{J}_e), \tag{8}$$

where $\mathbf{J}_e = \tilde{\mathbf{M}}_e^{-1/2}\mathbf{K}_e\tilde{\mathbf{M}}_e^{-1/2}$, $\tilde{\mathbf{M}}_e = \mathrm{diag}(m_i, m_j, m_k, m_l)$ and i, j, k, l are the nodes of e. Details are found in [She02b]. As a consequence, the maximum eigenvalue of $\mathbf{M}^{-1}\mathbf{K}$ is roughly proportional to the maximum eigenvalue of $\mathbf{J}_e$ of the worst element e. We employ a slightly different heuristic formulation which is derived from (8).

$$\lambda_{max}(\mathbf{M}^{-1}\mathbf{K}) \le \max_e \frac{1}{m_e}\lambda_{max}(\mathbf{K}_e) \tag{9}$$

where $m_e = \frac{1}{4}V_e\rho$ is the average mass of a corner of the element e with volume V_e and density ρ. In (8) the $\tilde{\mathbf{M}}_e$ and n are related. The masses of the nodes are the sum of the mass contributions of each tet. We use the observation that the ill-shaped elements typically have smaller masses than average mass $\tilde{\mathbf{M}}_e/n$. This gives an estimation of the eigenvalues and thus of the quality of an element. More precisely, if $\omega_{max}^2 = \frac{1}{m_e}\lambda_{max}(\mathbf{K}_e) \ge \left(\frac{c}{\Delta t}\right)^2$, it follows that the element e cannot be stably simulated with the time step Δt.

We note that this bound is very conservative; in practice, meshes can be found whose maximum frequency is up to a magnitude smaller than the frequency derived from (9).

4.4 Stiffness matrix assembly

We now employ the eigenvalues of $\mathbf{K}_e$ to compute the amount of material relaxation required for the stable simulation, and its eigenvectors to compute the direction of material relaxation. We first compute the matrix Λ_e by solving the corresponding 12×12 eigenvalue problem with a QR decomposition. The diagonal matrix $\Lambda_e = [\lambda_i]_{ii}, i = 1 \ldots 12$ contains the squared oscillation frequencies of the element. If the frequencies exceed the maximum oscillation ω_{max} from (9) we filter the corresponding frequencies λ_i from Λ_e and form a new matrix $\Lambda'_e = [\lambda'_i]_{ii}, i = 1 \ldots 12$ with

$$\lambda'_i = \begin{cases} \lambda_i & \text{if } \lambda_i \le \omega_{max}^2 \\ \omega_{max}^2 & \text{if } \lambda_i > \omega_{max}^2 \end{cases} \tag{10}$$

From Λ'_e, we assemble the *filtered* stiffness matrix $\mathbf{K}'_e = \Phi^T\Lambda'\Phi$. The modified material will react less to forces in the direction of the eigenvector corresponding to the modified modes. In the limit the material is infinitely soft along these directions. Thus, these modes have to be rigidified with constraints, as discussed subsequently.

The material modification can be done as a precomputation step for simulations without topological changes. In cutting or fracturing simulations, the filtering is done only for those elements whose topology has changed. This is possible due to the derived per-element approximation of the modal frequencies.

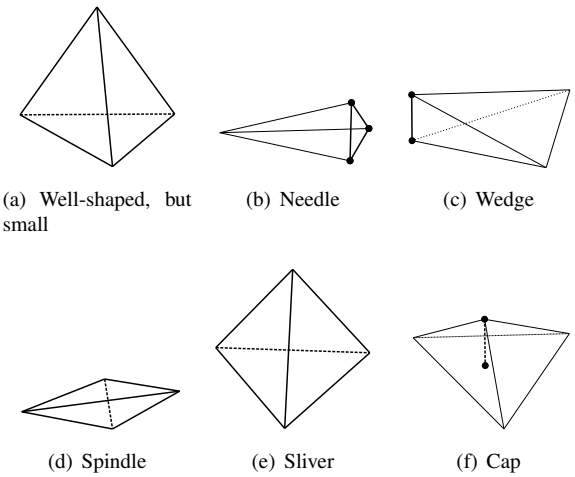
5 CONSTRAINING ELEMENT DEFORMATION

After having filtered the high-frequency vibration modes from ill-shaped elements, we obtain a stable

time integration. However, because the modified elements react less to forces in the direction of the removed modes, they tend to invert or distort. To prevent this, we replace these modes with constraints. These constraints try to keep the filtered elements as close as possible to the ground truth. The ground truth, in this case, is the element without material relaxation, and simulated with a sufficiently small time step.

To constrain the deformations of the filtered elements, we use the fast projection method [GHF*07], which is a manifold projection technique [HWL06]. The conceptual advantage is that fast projection can be implemented as a velocity filter, which directly modifies the future positions and velocities of the points after the numerical time-integration.

An intuitive way to rigidify the elements along the removed directions would be to directly employ the eigenvectors corresponding to the removed vibrations as constraint directions. However, since the eigenvectors and their derivatives cannot be computed symbolically, we instead classify the ill-shaped elements according to [BCER95]. For each type of degeneracy (see Figure 1), we use tailored constraints to rigidify the element along the filtered directions. This is because filtered directions align with the shortest distances in the tetrahedron, which can either be an edge or a height. If more than one vibration mode is filtered, or if multiple directions within the same element have been filtered, we rigidify the whole element.



(a) Well-shaped, but small (b) Needle (c) Wedge

(d) Spindle (e) Sliver (f) Cap

Figure 1: We distinguish 6 different element types [BCER95]. Highlighted line segments mark edge (b), (c) or height (f) constraints. The other elements (a), (d), (e) use volume constraints.

- Well-shaped elements as in Figure 1(a) can cause instability if they are small, compared to the average tet size in the mesh. Because all four heights are relaxed, we have to rigidify the element completely, which is done by constraining the lengths of all six edges of the tetrahedron. The length constraint $\mathbf{C}_e(\mathbf{x},\mathbf{y})$ for an edge $(\mathbf{x},\mathbf{y})$ is

$$C_e(\mathbf{x},\mathbf{y}) = \|\mathbf{y}-\mathbf{x}\|^2 - l_0^2 = 0 \qquad (11)$$

where $\mathbf{l}_0$ is the undeformed length of the edge.

- For needles, Figure 1(b), we use three edge constraints to approximate the relaxed heights. Edge constraints are shared between tetrahedra and thus reduce the overall number of constraints.

- For wedges, Figure 1(c), we use one edge constraints along the shortest edge.

- Spindles and Slivers, Figures 1(d) and 1(e), have relaxed directions perpendicular to the plane spanned by the crossing edges. As such all four heights have to be constrained. Again we use edge constraints to rigidify the whole element.

- Finally, for caps like in Figure 1(f) we set a height constraint along the shortest height. The function $\mathbf{C}_h(\mathbf{x},\mathbf{y},\mathbf{z},\mathbf{w})$ maintaining the distance h_0 between a tetrahedron corner $\mathbf{w}$ and its opposite face $(\mathbf{x},\mathbf{y},\mathbf{z})$ is

$$\begin{aligned}\mathbf{C}_h(\mathbf{x},\mathbf{y},\mathbf{z},\mathbf{w}) &= \frac{(((\mathbf{y}-\mathbf{x})\times(\mathbf{z}-\mathbf{x}))\cdot(\mathbf{w}-\mathbf{x}))^2}{\|(\mathbf{y}-\mathbf{x})\times(\mathbf{z}-\mathbf{x})\|^2} - h_0^2 \\ &= 0 \qquad (12)\end{aligned}$$

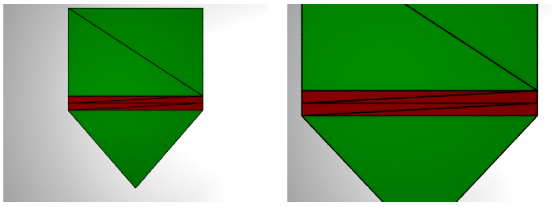
These constraints are symbolically differentiated and stacked in the constraint Jacobian matrix. Then, the fast projection method is employed to compute the feasible positions. The corresponding linear system is solved with an sparse Cholesky decomposition whose run-time is linear in the number of non-zero blocks. Therefore the method is linear in the number of constraints. Experiments indicate that very few iterations provide sufficient accuracy. Notice that in contrast to [TPS09], we do not impose constraints on the well-shaped tetrahedra, leading to significantly fewer constraints, which can be maintained efficiently.

6 RESULTS

In this section, we demonstrate our approach on a variety of dynamic simulations of deformable objects. The tetrahedral meshes have been generated with the approach described in [MT03]. To cut the meshes, we use the approach in [SHGS06] which tries to keep the number of newly created elements small. All experiments have been performed on an Intel Core2 Duo PC running at 2.5GHz.

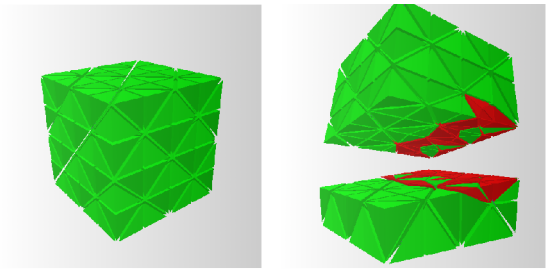
6.1 Stack of Wedges

We illustrate the working of our approach on a particularly simple object consisting of six ill-shaped, wedge-type tets, encompassed by well-shaped tets (see Figure 2). The edge length of the well-shaped tets is 1 m, the Young's modulus is $E = 10$KPa, the Poisson ratio is $\nu = 0.3$, and the density is $\rho = 0.05$kg m^{-3}. The wedges in the middle have one short edge and 5 long edges of approximately the same length. We obtain an edge ratio value of ~ 19, meaning that the longest edge is 19 times longer than the vertical edge. The computation of one simulation step takes 0.06ms. Due to the



(a) (b)

Figure 2: We illustrate the working of our approach on a particularly simple object consisting of six ill-shaped tets (red) encompassed by well-shaped tets (green). For this object, a speed-up of a factor of 3.5 is obtained by employing our approach. Right: Close-up of the six wedges.



(a) (b)

Figure 3: Our approach can also be employed in interactive cutting simulations. Left: Object before being cut. Right: After having cut the object into two parts. The cutting algorithm generates ill-shaped tets (red). By employing our method, we can keep a constant time step throughout the simulation.

ill-shaped tets, a small time step of $\Delta t_{max} = 0.03$ms is necessary to stably simulate the object, as we have determined experimentally. Notice that Δt_{max} could also have been derived from (9), but this would result in a much smaller time step because of the conservative estimation. Consequently, the computation of one physical second takes $1000/0.03 \times 0.06 = 2000ms= 2$ seconds, which is not real-time.

Now we assume that we have to simulate the object with a time step of $\Delta t = 0.7$ms, for example because we aim at a real-time simulation. Thus, we want to relax all elements which *cannot* be simulated with Δt_{min}. By performing the eigenvalue decomposition element-wise, we isolate the six wedges and relax them along the vertical direction. Afterwards, we constrain the small edges of the wedges, resulting in 21 edge constraints that have to be enforced. The resulting relaxed object can now be simulated with a time step of $\Delta t = 0.7$ms, as required. Notice that the new time step is more than 20 times larger than the initial one. The computation of one time step, including the constraint enforcement, takes 0.43 ms. Consequently, the simulation of one physical second takes $1000/0.7 \times 0.43 = 614$ms $= 0.61$ seconds, which is real-time. Table 1 summarizes the timings.

6.2 Cutting a Cube

Our method is particularly attractive in the context of cutting simulations. In Figure 3 we cut a cube of edge length 4 m consisting of 320 well-shaped elements with no dihedral angle smaller than 45°. The Young's modulus of the object is $E = 10$KPa, the Poisson ratio is $\nu = 0.3$, the density is $\rho = 0.06$kg m^{-3}. The cube is simulated with a time step of $\Delta t = 1$ms, and the time to compute one simulation step is 2.6ms. Thus, the simulation of one physical second takes $1000 \times 2.3 =$ 2300ms $= 2.3$ seconds. If we cut the cube, then the cutting method [SHGS06] refines the mesh in the region around the cut plane, thus generating 12 new tets and changing the geometry of 125 tets. Some of these tets will be ill-shaped, since we do not perform any further geometric optimizations. Consequently, the maximum possible time step after having cut the mesh would drop to $\Delta t_{max} = 0.42$ms, which is again experimentally determined. Thus, without our method, the computation of one physical second takes $1000/0.42 \times 2.3 = 5476$ms $= 5.4$ seconds.

We now demand that the time step stays constant throughout the simulation, *i. e.*, we relax all tets which cannot be simulated at $\Delta t = 1$ms. By performing the eigenvalue decomposition element-wise, we isolate 14 ill-shaped tets and introduce 30 constraints. By simulating the relaxed object, we can keep the time step $\Delta t_{rel} = \Delta t = 1$ms constant, as desired. The time to compute one simulation step has slightly increased to 2.4ms, which is due to the constraint handling. The computation of one physical second now takes $1000 \times 2.4 = 2400$ms $= 2.4$ seconds (see Table 1).

6.3 Liver Mesh

Generating tetrahedral meshes from surface meshes is a difficult problem, especially if conforming or surface-interpolating tetrahedral meshes are generated. Our method can be employed to simulate such meshes by employing larger time steps. To illustrate this, we consider a liver model composed of 407 tets (see Figure 4). The liver has a Young's modulus $E = 10$KPa, a Poisson ratio of $\nu = 0.3$, and a density $\rho = 1000$kg m^{-3}. 52 tets (colored red in Figure 4) have modal frequencies which are more than twice as high as the average value. These few elements require that the time step has to be halved. Looking at the classification of these tetrahedra, we can see that only six of them are considered ill-shaped. The rest are well-shaped but are very small and thus have a high modal frequency. If we dynamically simulate the liver without using our approach, we need a time step of $\Delta t_{max} = 1.2$ms. The time to perform one simulation step is 2.0ms, and therefore the time to compute one physical second is $1000/1.2 \times 2.0 = 1666$ms $= 1.6$ seconds.

Assuming that we want to simulate the liver with a time step of $\Delta t_{min} = 2$ms, the method relaxes the 52 elements, resulting in 120 constraints. The time to compute one simulation step increases to 2.6ms. However, the overall time to compute one second is now

Object	Δt	$t_{timestep}$	t_{second}
Wedges	0.03ms	0.06ms	2.0 sec.
	0.7ms	0.43ms	0.6 sec.
Cube (uncut)	1ms	2.3ms	2.3 sec.
	1ms	2.3ms	2.3 sec.
Cube (cut)	0.42ms	2.3ms	5.4 sec.
	1ms	2.4ms	2.4 sec.
Liver	1.2ms	2.0ms	1.6 sec.
	2ms	2.6ms	1.3 sec.

Table 1: This table summarizes the performance gains of our approach. Δt is the (maximum) time step of the numerical integration, $t_{timestep}$ is the time to compute one simulation step, and t_{second} is the time to compute one physical second. We simulate each object twice, once without employing our approach, and once by employing our approach. The last column indicates that in all scenarios we obtain a significant speed-up by employing our approach.

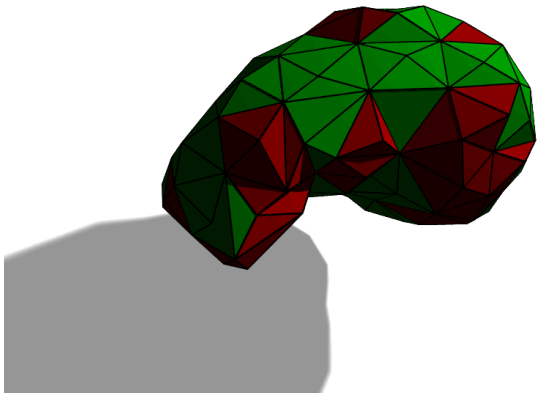


Figure 4: Our approach can also be employed in medical simulations. Here, a liver object is discretized into 407 elements. The ill-shaped tets (red) impose severe time step restrictions which can be relaxed by employing our approach.

$1000/2 \times 2.6 = 1300\text{ms} = 1.3$ seconds, which is a performance gain of 23% (see Table 1).

7 CONCLUSIONS

We have presented a method which allows us to determine ill-shaped elements based on a time step constraint. We relax the material of the ill-shaped elements in the direction of their largest vibration mode. To approximate the unmodified simulation behavior as closely as possible, we apply a velocity filter based on fast projection using constraints along the filtered directions.

Our method does not handle element inversion. But it can be combined with any inversion handling technique available, *e. g.*[ITF04] or [ST08]. Another limitation of our method is that the estimation of the elemental vibration modes is very conservative, which results in filtering more elements than actually necessary. This is because we consider each element in isolation, without looking at its neighborhood. By investigating this issue, the number of constraints could be reduced.

In addition, we want to include damping into the system. The material relaxation should stay unchanged, but updating the velocities in the fast projection step will change slightly. Additionally, we want to look at how compatible our system is with implicit integration schemes.

8 ACKNOWLEDGEMENTS

This research was supported by the EU project PASSPORT FP7 ICT-2007-223894.

REFERENCES

[ACSYD05] Alliez P., Cohen-Steiner D., Yvinec M., Desbrun M.: Variational tetrahedral meshing. ACM Transactions on Graphics (Proc. SIGGRAPH) 24, 3 (2005), 617–625.

[BCER95] Bern M., Chew P., Eppstein D., Ruppert J.: Dihedral bounds for mesh generation in high dimensions. in Proc. Symposium on Discrete Algorithms (1995), Society for Industrial and Applied Mathematics, pp. 189–196.

[BFA02] Bridson R., Fedkiw R., Anderson J.: Robust treatment of collisions, contact and friction for cloth animation. ACM Transactions on Graphics (Proc. SIGGRAPH) 21, 3 (2002), 594–603.

[CDRR04] Cheng S.-W., Dey T. K., Ramos E. A., Ray T.: Quality meshing for polyhedra with small angles. in Proc. Symposium on Computational Geometry (2004), pp. 290–299.

[Fri72] Fried I.: Condition of finite element matrices generated from nonuniform meshes. AIAA Journal 10 (1972), 219–221.

[GHF*07] Goldenthal R., Harmon D., Fattal R., Bercovier M., Grinspun E.: Efficient simulation of inextensible cloth. ACM Transactions on Graphics (Proc. SIGGRAPH) 26, 3 (2007), 49.

[GK07] Gurfil P., Klein I.: Stabilizing the explicit euler integration of stiff and undamped linear systems. Journal of Guidance, Control, and Dynamics 30, 6 (2007), 1659 – 1667.

[HSO03] Hauser K. K., Shen C., O'Brien J. F.: Interactive deformation using modal analysis with constraints. in Proc. Graphics Interface (2003), Canadian Human-Computer Commnication Society, pp. 247–256.

[Hug87] Hughes T.: The Finite Element Method. Prentice-Hall, 1987.

[HWL06] Hairer E., Wanner G., Lubich C.: Geometric Numerical Integration - Structure-Preserving Algorithms for Ordinary Differential Equations. Springer Berlin Heidelberg, 2006.

[ISS04] Ito Y., Shih A. M., Soni B. K.: Reliable isotropic tetrahedral mesh generation based on an advancing front method. in Proc. 13th International Meshing Roundtable (2004), pp. 95–106.

[ITF04] Irving G., Teran J., Fedkiw R.: Invertible finite elements for robust simulation of large deformation. in Proc. Symposium on Computer Animation (2004), Eurographics Association, pp. 131–140.

[KFCO06] Klingner B. M., Feldman B. E., Chentanez N., O'Brien J. F.: Fluid animation with dynamic meshes. ACM Transactions on Graphics (Proc. SIGGRAPH) 25, 3 (2006), 820–825.

[Kle07] Klein B.: FEM - Grundlagen der Anwendungen der Finite-Elemente-Methode im Maschinen- und Flugzeugbau. Vieweg, 2007.

[KS07] Klingner B. M., Shewchuk J. R.: Aggressive tetrahedral mesh improvement. in Proc. IMR (2007), Springer, pp. 3–23.

[LS07] Labelle F., Shewchuk J. R.: Isosurface stuffing: fast tetrahedral meshes with good dihedral angles. ACM Transactions on Graphics (Proc. SIGGRAPH) 26, 3 (2007), 57.

[MG04] Müller M., Gross M.: Interactive virtual materials. in Proc. Graphics Interface (2004), Canadian Human-Computer Communications Society, pp. 239–246.

[MHTG05] Müller M., Heidelberger B., Teschner M., Gross M.: Meshless deformations based on shape matching. ACM Transactions on Graphics (Proc. SIGGRAPH) 24, 3 (2005), 471–478.

[MT03] Müller M., Teschner M.: Volumetric meshes for real-time medical simulations. in Proc. BVM (2003), pp. 279–283.

[PW89] Pentland A., Williams J.: Good vibrations: modal dynamics for graphics and animation. in Proc. SIGGRAPH (1989), ACM, pp. 215–222.

[QSS00] Quarteroni A., Sacco R., Saleri F.: Numerical Mathematics. Springer, 2000.

[RS09] R. Schmedding M. Gissler M. T.: Optimized damping for dynamic simulations. Proc. Spring Conference on Computer Graphics 25 (2009), 205–212.

[Sch97] Schoberl J.: Netgen - an advancing front 2d/3d-mesh generator based on abstract rules. Computing and Visualization in Science 1 (1997), 41–52.

[She98] Shewchuk J. R.: Tetrahedral mesh generation by delaunay refinement. in Proc. 14th Annual Symposium on Computational Geometry (1998), pp. 86–95.

[She02a] Shewchuk J. R.: What is a good linear element? interpolation, conditioning, and quality measures. in Proc. 11th International Meshing Roundtable (September 2002), pp. 115–126.

[She02b] Shewchuk J. R.: What is a Good Linear Element? Interpolation, Conditioning, and Quality Measures. Tech. rep., Carnegie Mellon University, 2002. Online version.

[SHGS06] Steinemann D., Harders M., Gross M., Szekely G.: Hybrid cutting of deformable solids. in Proc. IEEE conference on Virtual Reality (2006), IEEE Computer Society, pp. 35–42.

[Shi04] Shinya M.: Stabilizing explicit methods in spring-mass simulation. in Proc. Computer Graphics International (2004), IEEE Computer Society, pp. 528–531.

[ST08] Schmedding R., Teschner M.: Inversion handling for stable deformable modeling. The Visual Computer 24, 7 (2008), 625–633.

[TPS09] Thomaszewski B., Pabst S., Straßer W.: Continuum-based strain limiting. Computer Graphics Forum (Proc. Eurographics) 28 (4 2009), 569–576.

[WT08] Wojtan C., Turk G.: Fast viscoelastic behavior with thin features. ACM Transactions on Graphics (Proc. SIGGRAPH) 27, 3 (2008), 1–8.