Cloth Simulation Using Soft Constraints

Mihai Frâncu
Politehnica University
Bucharest, Romania
mihai.francu@cs.pub.ro

Florica Moldoveanu
Politehnica University
Bucharest, Romania
florica.moldoveanu@cs.pub.ro

Abstract

This paper describes a new way of using projective methods for simulating the constrained dynamics of deformable surfaces. We show that the often used implicit integration method for discretized elastic systems is equivalent to the projection of regularized constraints. We use this knowledge to derive a Nonlinear Conjugate Gradient implicit solver and a new projection scheme based on energy preserving integration. We also show a novel way of adding damping to position based dynamics and a different view on iterative solvers. In the end we apply these fresh insights to cloth simulation and develop a constraint based finite element method capable of accurately modeling thin elastic materials.

Keywords

Implicit integration, constraints, projection, PBD, damping, iterative solver, cloth, FEM

1 INTRODUCTION

For decades now the preferred method of simulating elastic systems in computer graphics has been the implicit integration of the equations of motion. The method is very attractive due to its unconditional stability and large time steps. It has been widely used for simulating cloth and finite difference or finite element soft bodies in general. Constraint based methods on the other hand have not received that much attention, with the exception of rigid body dynamics. Only recently there has been an increase in the number of papers on the subject in relation to soft bodies and we believe there is room for improvement. Many regard the method as being an inaccurate approximation of natural phenomena which are better described by elasticity theory. In this paper we aim to reconcile the two methods and show that they are two faces of the same problem; this can prove useful for the further development of both approaches.

1.1 Related work

Constraint based methods have appeared originally in their acceleration based formulation for rigid body dynamics [Bar94]. Later on, velocity or impulse based methods gained more popularity [AH04, Erl07]. Position based methods are actually a nonlinear version of velocity based ones, in the sense that they can still be expressed as velocity filters, but constraints are enforced at positional level [ST96]. Such a method was made popular in games by [Jak01] and was later refined and extended by [MHHR07] under the name Position Based Dynamics (PBD). Part of the inspiration for this method came from molecular dynamics where methods like SHAKE or RATTLE are widely used [BKLS95]. A more detailed study for the application to cloth simulation in computer graphics was done in [Gol10]. Here the method of *fast projection* is developed based on an implicit treatment of constraint directions [HCJ*05] and a better energy preserving integrator is also derived. A similar method was used to develop the unified Nucleus solver in Autodesk Maya [Sta09]. Position based methods rely on *projection* for solving differential algebraic equations (DAE), which is ultimately an optimization problem [HLW06]. Another part of inspiration came from *strain limiting* techniques used in elastic cloth simulation [Pro96, BFA02].

Constraint based methods are often criticized for the fact they simulate only nearly inextensible materials and are prone to *locking*. In order to address this [EB08] use fast projection in conjunction with a BDF-2 integrator on a conforming triangular mesh. They also give a brief proof for fast projection being the limit of infinitely stiff elastic forces. Other authors prefer to use quad-predominant meshes or diamond subdivision [Gol10].

Constraint *regularization* was employed mainly in [Lac07] for making rigid dynamics with contact and friction more tractable numerically. We take the name *soft constraints* from [Cat10] where an older idea is used: regularization under the mask of Constraint

Force Mixing (CFM) [Smi06]. Recently constraint regularization has been used for particle based fluid simulation [MM13]. Another application was intended for the simulation of deformable elastic models using a constraint based formulation of the linear Finite Element Method (FEM) [SLM06]. Similar position based approaches can be found in [BML*14] and [BKCW14]. The FEM constraint approach is similar in philosophy with *continuum strain limiting* [TPS09].

The implicit integration of the equations of motion has become pervasive for cloth since the seminal work of [BW98]. The method was also applied for FEM simulation [MSJT08]. Its main attraction is its unconditional stability for very stiff equations and large time steps. By implicit integration we usually mean the Implicit Euler (IE) method, but other implicit integrators were also employed, like BDF-2 [CK02], Implicit Midpoint [OAW06] or Newmark [SSB13]. These integration methods offer better energy conservation and more responsive simulation, in contrast to IE which artificially dampens out high frequency details in exchange for stability. Other variations include approximations made to the force Jacobian [HET01] or an implicit-explicit (IMEX) approach [EEH00, BMF03]. Most approaches however use only one Newton solver iteration. More recently a new view on IE as an optimization problem was presented in [LBOK13].

A special class of integrators labeled *variational* can be deduced directly from the discretization of the Euler-Lagrange equations of motion [SD06]. They are also symplectic integrators, i.e. they preserve area in phase space, which also means they are closer to preserving energy and momenta [HLW06]. Many of them are explicit methods (e.g. Symplectic Euler, Verlet, Leapfrog) so care must be taken to the time step size. Variational implicit methods like Implicit Midpoint or Newmark are more stable and can be converted to projection schemes (e.g. through our constraint space transformation). This is why we used them as inspiration for our energy conservation strategy.

A new alternative that is totally different from implicit integration of elastic systems or our approach is *exponential integration* [MSW14] which relies on evaluating trigonometric matrices (in terms of exponential functions).

1.2 Contributions

We present in this paper a constraint based simulator that is able to reproduce fully the elastic properties of cloth. We base our results on the fact that constraint projection methods are in fact equivalent to implicit integration of stiff springs (Section 2). Our approach is not entirely new as it is based on the idea of constraint regularization [Lac07]. We chose to use a PBD method instead as it corresponds to the nonlinear case [ST96] and it handles fast deforming bodies more robustly. Catto [Cat10] uses Baumgarte stabilization (ERP) and CFM and relates them to the stiffness and damping of an implicit harmonic oscillator. We give a more general and accurate correspondence to elastic parameters.

In Section 3 we derive a simple implicit integration solver based on the Variational Implicit Euler approach in [LBOK13] and the Nonlinear Conjugate Gradient method which is very similar to PBD. From it we obtain the equations of regularized constraint projection (Section 4). Using this transformation we derive a new projection method with better energy conservation (Section 5). In Section 6 we present a novel and effective way of adding more damping to PBD. Section 7 shows how *relaxation* can be used for block solving and regularization and how to transform the Conjugate Gradient method into Conjugate Residuals. In Section 8 we present constrained mass-spring systems for cloth and how to prevent locking. In Section 9 we present a nonlinear Saint Vennant-Kirchoff (StVK) elasticity model implemented through soft constraints. We take the area and planar strain constraint from [SLM06] and derive a method that takes into account the discretization and elastic properties of cloth. The closest approach to our method is [BKCW14] but they use energy as a constraint instead of a minimization objective. [BML*14] is using the same energy objective as us but their numerical method is different.

2 OPTIMIZATION EQUIVALENCE

In this section we would like to show that implicitly integrating stiff elastic forces is no different than using a constraint based formulation with regularization. The most general way to show this is by employing an optimization formulation for both methods. Let us start with Implicit Euler:

$$\mathbf{M}\Delta\mathbf{v} = h\mathbf{f}(\mathbf{x}_0 + \Delta\mathbf{x}), \tag{1}$$

$$\Delta\mathbf{x} = h\Delta\mathbf{v} \tag{2}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{x}$ are positions, $\mathbf{v}$ are velocities, h is the time step, $\mathbf{x}_0 = \mathbf{x}^{(n)} + h\mathbf{v}^{(n)}$ and $\mathbf{f}(\mathbf{x})$ are conservative forces, i.e. $\mathbf{f}(\mathbf{x}) = -\nabla_{\mathbf{x}} U(\mathbf{x})$. We can reformulate (1) as:

$$\tfrac{1}{h^2}\mathbf{M}\Delta\mathbf{x} = -\nabla_{\mathbf{x}} U(\mathbf{x}_0 + \Delta\mathbf{x}), \tag{3}$$

This is a nonlinear equation which is typically solved using the Newton method, but it can also be regarded as the optimality condition of an optimization problem:

$$\text{minimize } \tfrac{1}{2h^2}\Delta\mathbf{x}^T\mathbf{M}\Delta\mathbf{x} + U(\mathbf{x}_0 + \Delta\mathbf{x}). \tag{4}$$

It is shown in [GHF*07] that a similar formulation can be used for a *projection* method using Lagrange multipliers and implicit constraint directions:

$$\text{min. } \tfrac{1}{2h^2}\Delta\mathbf{x}^T\mathbf{M}\Delta\mathbf{x} - \boldsymbol{\lambda}^T\mathbf{c}(\mathbf{x}_0 + \Delta\mathbf{x}) + U_{ext}(\mathbf{x}^{(n)}), \tag{5}$$

where $\mathbf{c}(\mathbf{x})$ are the constraint functions that need to be zero. Note that we modified the formulation so that the external forces potential is included in the objective function. Note that the external forces are treated explicitly, i.e. using the position at the beginning of the frame, which is also the case for (4).

The potential term $U_c = \lambda^T \mathbf{c}(\mathbf{x})$ gives us the internal constraint forces. If we take the gradient of this constraint potential we get the *principle of virtual work*:

$$\mathbf{f}_c = \nabla_{\mathbf{x}} \mathbf{c}(\mathbf{x}) \lambda = \mathbf{J}^T \lambda. \tag{6}$$

If we express the total potential in general as $U = U_{int} + U_{ext}$, then so far the objectives in (4) and (5) are the same in the first (inertial) term and U_{ext}. For U_{int} we have an expression in the second case, but we have not yet specified one for the implicit integration. And we are not forced to provide one, but in reality, following the approach in [BW98], this internal potential is usually made up of quadratic elastic potentials with the purpose of enforcing certain constraints [Lan70]:

$$U_e(\mathbf{x}) = \tfrac{k}{2} \|\mathbf{c}(\mathbf{x})\|^2. \tag{7}$$

More generally we can replace stiffness k by a matrix:

$$U_e(\mathbf{x}) = \tfrac{1}{2} \mathbf{c}(\mathbf{x})^T \mathbf{E} \mathbf{c}(\mathbf{x}),$$

which is extremely useful when dealing with different stiffnesses in a mass-spring system or continuum based constitutive laws.

The potential energy in (7) gives forces of the form:

$$\mathbf{f}_e(\mathbf{x}) = -k \mathbf{J}^T \mathbf{c}(\mathbf{x}). \tag{8}$$

By comparing (6) and (8) we can see that they act in the same direction and by requiring that they have the same magnitude we obtain the regularization condition:

$$\mathbf{c}(\mathbf{x}) + \varepsilon \lambda = 0, \tag{9}$$

where $\varepsilon = 1/k$. We call this a *soft constraint* and by enforcing it we basically set the internal potential energy in (5) to be the same as in (4), thus making the two problems equivalent. It is clear now that when stiffness k goes to infinity ($\varepsilon \to 0$) implicit integration of springs becomes the constrained dynamics problem in (5). In the general case ε gets replaced by $\mathbf{E}^{-1}$.

In conclusion, not only is constraint based dynamics a limit case of implicit integration, but it can be made equivalent by replacing the strict constraint condition with a "softer" one. This permits us to solve a mass-spring system or any other discretized elastic system by casting the problem into the following form:

$$\begin{aligned} \mathbf{M}\Delta\mathbf{x} &= h^2 \left(\mathbf{J}^T \lambda - \nabla_{\mathbf{x}} U_{ext}(\mathbf{x}^{(n)}) \right), \\ 0 &= \mathbf{c}(\mathbf{x}_0 + \Delta\mathbf{x}) + \varepsilon \lambda \end{aligned}$$

This equivalence opens up a whole range of opportunities, especially for bringing results from implicit integration into the world of constraint based simulation. This was not considered possible in the past, as projection methods were regarded as an approximation of true elasticity based ones [Lac07, LBOK13].

3 NONLINEAR CONJUGATE GRADIENT SOLVER

The most important analogy we make in this paper is that between Implicit Euler integration and PBD. PBD starts from a candidate position (that includes the effect of external forces) and then runs an iterative process that does not involve the second derivative of the constraint function. This process is actually a minimization algorithm based on *sequential quadratic programming* (SQP) [WN99] that involves solving a linear system at every iteration, called *fast projection* [GHF*07]. This process can be further optimized by employing an inexact one step SOR-Newton scheme [Jak01, MHHR07] that reduces the cost of each iteration by running only one relaxation step.

The same logic can be applied to the Implicit Euler method expressed as the quadratic minimization problem in (4). If we choose the initial guess state to be one that incorporates the external forces, i.e. positions and velocities after an unconstrained step, we arrive at an approach similar to fast projection. The only difference is that the former works in configuration space, while the latter works in constraint space.

If we consider the initial candidate state consisting of $\tilde{\mathbf{v}} = \mathbf{v}^{(n)} + h\mathbf{f}_{ext}$ and $\tilde{\mathbf{x}} = \mathbf{x}^{(n)} + h\tilde{\mathbf{v}}$ we can rewrite (1) using a first order Taylor expansion around $\tilde{\mathbf{x}}$:

$$\mathbf{M}\delta\mathbf{v} = h\left(\mathbf{f}(\tilde{\mathbf{x}}) + \mathbf{K}\delta\mathbf{x}\right), \tag{10}$$

where $\mathbf{K} = \nabla_{\mathbf{x}}\mathbf{f} = -\frac{\partial^2 U}{\partial \mathbf{x}^2}$ is the *tangential stiffness matrix* and $\delta\mathbf{x} = h\delta\mathbf{v}$. Most authors choose to solve the implicit integration problem using only one Newton step, meaning we only need to solve one single linear system: $\mathbf{S}\delta\mathbf{v} = \mathbf{t}$. This works well in practice, but only if $\mathbf{K}$ contains second derivatives of the constraint function. This is because these terms contain information about the change of the constraint direction, so without them we need an iterative algorithm that keeps updating the constraint gradient. By dropping the second derivative term from $\mathbf{K}$ (see [BW98]) we get:

$$\mathbf{K} = -k\mathbf{J}^T\mathbf{J}. \tag{11}$$

This is equivalent to linearizing the force in (8) as in [EB08]. Using this formula at every Newton iteration we get the series of linear systems we need to solve:

$$(\mathbf{M} + h^2 k \mathbf{J}_i^T \mathbf{J}_i)\delta\mathbf{v}_{i+1} = h\mathbf{f}(\mathbf{x}_i), \tag{12}$$

where $\mathbf{J}_i^T = \nabla_{\mathbf{x}}\mathbf{c}(\mathbf{x}_i)$ and $\mathbf{x}_{i+1} = \mathbf{x}_i + h\delta\mathbf{v}_{i+1}$.

Uncostrained step to $\tilde{\mathbf{x}}, \tilde{\mathbf{v}}$
Compute Jacobian $\mathbf{J}$ and forces $\mathbf{f}$ using (8)
Compute residual $\mathbf{r} = \mathbf{d} = h\mathbf{f}$ and its square $\delta = \mathbf{r}^2$
for iter = 1:maxIter **do**
  Compute $\mathbf{q} = \mathbf{S}\mathbf{d} = (\mathbf{M} + h^2 k \mathbf{J}^T \mathbf{J})\mathbf{d}$
  Compute impulse $\mathbf{p} = \alpha\mathbf{d}$, where $\alpha = \delta / \mathbf{q}^T\mathbf{d}$
  Integrate: $\mathbf{v} \leftarrow \mathbf{v} + \mathbf{p}$, $\mathbf{x} \leftarrow \mathbf{x} + h\mathbf{p}$
  Recompute Jacobian $\mathbf{J}$ and forces $\mathbf{f}$
  Compute residual $\mathbf{r} = h\mathbf{f}$ and its square $\delta' = \mathbf{r}^2$
  Compute $\beta = \delta'/\delta$ and then $\delta' \leftarrow \delta$
  Compute new search direction $\mathbf{d} = \mathbf{r} + \beta\mathbf{d}$

Algorithm 1: NCG implicit solver

Nonlinear Conjugate Gradient (NCG) [She94] is a natural solution for solving the above problem, given its linear version is very popular for solving the one Newton step approach. The only changes we need to make to linear CG is to replace the system matrix at every step with $\mathbf{S}_i = \mathbf{M} + h^2 k \mathbf{J}_i^T \mathbf{J}_i$ (the Hessian of the objective function) and the residual with $\mathbf{r}_i = h\mathbf{f}(\mathbf{x}_i)$. We use a Fletcher-Reeves formula and perform the inner line search in only one iteration - see Algorithm 1.

Note that the NCG method is not necessarily faster than traditional CG linear implicit solvers (we found that it takes roughly 40% more time without optimizations). We can also add back the second derivative term if we want. Also, visually there is no big difference between the two methods. The only advantages you would get with the NCG method are smaller spring elongations and more stability for large time steps. But the main reason for devising the scheme is the similarity with PBD which we further exploit in the next section.

4 CONSTRAINT SPACE

Given that we already know that the regularized projection method is equivalent to implicit integration and that the formulation in the previous section is already very similar to PBD, we would like to transform the system in (12) to one corresponding to fast projection. So by multiplying (12) on the left-hand side by $\mathbf{T} = \frac{1}{hk}\mathbf{A}^{-1}\mathbf{J}$, where $\mathbf{A} = \mathbf{J}\mathbf{M}^{-1}\mathbf{J}^T$, we get:

$$(h^2\mathbf{A} + \varepsilon\mathbf{I})\delta\lambda + \mathbf{c}(\mathbf{x}) = \mathbf{0}, \tag{13}$$

which is precisely the system we need to solve at every iteration of fast projection for the regularized constraints in (9). In order to get this result we made the substitution $\delta\mathbf{v} = h\mathbf{M}^{-1}\mathbf{J}^T\lambda$ which derives from the optimality conditions of the constraint projection optimization problem. To back our claims you can see in Figure 1 that NCG and PBD behave almost the same and very closely to the exact and CG semi-implicit solvers.

We call $\mathbf{T}$ a *constraint space transformation* from configuration space and show that the inverse transform is also possible by multiplying (13) to the left hand side by $\mathbf{Q} = hk\mathbf{M}^{-1}\mathbf{J}^T$. Note that $\mathbf{TQ} = \mathbf{I}_m$ (m - number of constraints) and $\mathbf{QT} = \mathbf{I}_n$ (n - degrees of freedom). We thus found a quick way of switching from one interpretation to the other. Also, by setting $\varepsilon \to 0$ (infinite stiffness) we recover the classic iterative projection formula:

$$h^2\mathbf{A}\delta\lambda + \mathbf{c}(\mathbf{x}) = \mathbf{0}, \tag{14}$$

Note though that not all implicit formulations can be converted this way to a constraint based formulation and that is because the methods are equivalent as optimization problems but the numerical methods used may differ in significant ways, e.g. the use of the second derivative. Still their results converge towards the same solution.

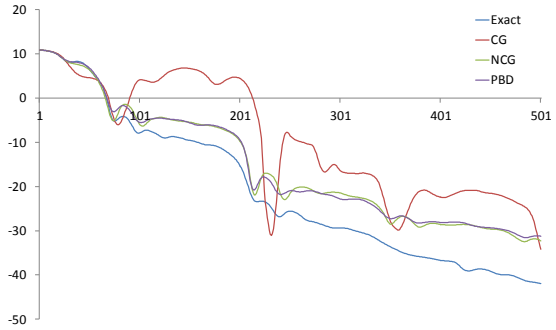

Figure 1: The energy evolution over 500 frames of a 15x15 piece of cloth using NCG (green), PBD (purple) and CG (red) and exact (blue) semi-implicit solvers.

5 ENERGY CONSERVATION

Implicit methods in general suffer from artificial numerical dissipation, whether they are used for an elasticity based formulation or a constraint based one. This is usually regarded as a good stability property and the extra damping is considered beneficial by the computer graphics community. Still in many cases like the example of cloth, this integration technique acts like a low-pass filter that removes high frequency motion and thus prevents the formation of high-detail wrinkles and responsive folds.

In the projection methods literature there exist energy preserving solutions like *symmetric projection* [HLW06] or the Velocity Verlet method proposed in [Gol10]. Another popular integration method that can conserve energy exactly is the Implicit Midpoint method [OAW06]. There exist other explicit variational integrators (e.g. Symplectic Euler) with very good energy conservation properties but they suffer from the same time step limitations as any other explicit method.

Given the fact that we are not able to solve the nonlinear equations generated by implicit methods exactly, the

accumulated errors will make Implicit Midpoint much less stable than Implicit Euler. One could alleviate this problem by using techniques suited for explicit methods, e.g. smaller/adaptive time steps or adding in a damping term. Our solution is to employ an integration scheme taken from [Lac07] which gives us more flexibility:

$$\tfrac{1}{h}\mathbf{M}\Delta\mathbf{v} = \mathbf{f}_{ext} + (1-\alpha)\mathbf{f}(\mathbf{x}^{(n)}) + \alpha\mathbf{f}(\mathbf{x}^{(n+1)}), \quad (15)$$

$$\tfrac{1}{h}\Delta\mathbf{x} = (1-\beta)\mathbf{v}^{(n)} + \beta\mathbf{v}^{(n+1)}, \quad (16)$$

where α and β are between 0 and 1 and we can distinguish the following special cases: Explicit Euler ($\alpha=\beta=0$), Implicit Euler ($\alpha=\beta=1$), Implict Midpoint ($\alpha=\beta=\frac{1}{2}$), and Symplectic Euler ($\alpha=0,\beta=1$ or $\alpha=1,\beta=0$).

Moving slightly away from the Implicit Midpoint method and making it more implicit permits us to have low artificial numeric dissipation while still being able to solve the system approximately and obtain a stable, yet responsive simulation. Using the constraint space transformation , i.e. multiplying (15) and (16) on the left hand side by $\mathbf{T}$, we obtain the following regularized projection:

$$(h^2\alpha\beta\mathbf{A}_i + \varepsilon\mathbf{I})\delta\lambda_{i+1} + \alpha\mathbf{c}(\mathbf{x}_i) = \mathbf{0}, \quad (17)$$

where $\mathbf{x}_{i+1} = \mathbf{x}_i + \beta h\mathbf{M}^{-1}\delta\mathbf{f}_{i+1}$, $\mathbf{v}_{i+1} = \mathbf{v}_i + h\mathbf{M}^{-1}\delta\mathbf{f}_{i+1}$ and $\delta\mathbf{f}_{i+1} = \mathbf{J}_i^T\delta\lambda_{i+1}$. Also the integration of the candidate state changes to:

$$\tilde{\mathbf{x}} = \mathbf{x}^{(n)} + h\mathbf{v}^{(n)} + h^2\mathbf{g} + h^2\beta(1-\alpha)\mathbf{M}^{-1}\mathbf{J}^T\lambda^{(n)},$$
$$\tilde{\mathbf{v}} = \mathbf{v}^{(n)} + h\mathbf{g} + h(1-\alpha)\mathbf{M}^{-1}\mathbf{J}^T\lambda^{(n)},$$

where we replaced external force by gravity for brevity and the Jacobian $\mathbf{J}$ is computed at position $\mathbf{x}^{(n)}$.

At the limit $\varepsilon \to 0$ this whole procedure is of course equivalent to projecting the candidate positions using the same system (14) as in fast projection and PBD. The difference appears only in the regularization term which gets replaced by $\varepsilon\alpha^{-1}\beta^{-1}$. The above formulation is also good when stepping both positions and velocities at the same time. Alternatively we could estimate the new velocity only at the end using equation (16):

$$\beta\mathbf{v}^{(n+1)} = \tfrac{1}{h}(\mathbf{x}^{(n+1)} - \mathbf{x}^{(n)}) - (1-\beta)\mathbf{v}^{(n)}$$

We could have reached a similar result using the fast projection formalism in [Gol10] but our constraint space transformation method ensures that we also obtain the correct regularization term. For example, we can apply the same technique to obtain the BDF-2 based projection method presented in [EB08] and find that the new regularization term is $\frac{9}{4}\varepsilon$. Still we prefer our method as we do not need to store previous positions and velocities and, being a one step scheme, it is better suited for non-smoothness. Actually more related to ours is the Newmark scheme [SSB13] for which we can use the same projection method regularized with ε/β, using the Newmark β factor between 0 and $1/2$.

6 DAMPING

Now that we have reduced the amount of artificial damping, we can add back some real damping. Our method will be based on the damping force expression used in [BW98] which is also a special case of the widely used method of *Rayleigh damping* [SSB13]. In order to extend (10) to contain the damping force we need to consider the total force as $\mathbf{f}(\mathbf{x},\mathbf{v}) = \mathbf{f}_e(\mathbf{x}) + \mathbf{f}_d(\mathbf{v})$, i.e. sum of elastic and damping forces:

$$(\mathbf{M} - h^2\mathbf{K} - h\mathbf{D})\delta\mathbf{v} = h\mathbf{f}(\tilde{\mathbf{x}}),$$

where $\mathbf{D} = \frac{\partial\mathbf{f}}{\partial\mathbf{v}} = \nabla_{\mathbf{v}}\mathbf{f}$. This is equivalent to having a damping force:

$$\mathbf{f}_d = d\nabla_{\mathbf{x}}\mathbf{c}(\mathbf{x})\dot{\mathbf{c}}(\mathbf{x}) = d\mathbf{J}^T\mathbf{J}\mathbf{v},$$

where d is the damping coefficient and $\dot{\mathbf{c}}(\mathbf{x}) = \mathbf{J}\mathbf{v}$.

Rayleigh damping makes the approximation $\mathbf{D} = \zeta\mathbf{M} + \gamma\mathbf{K}$, but we will only be using the second term as it makes the derivations simpler and it is only damping along the constraints we are interested in (we can achieve drag friction in other ways). The implicit integration formula (10) now becomes:

$$(\mathbf{M} - h(h+\gamma)\mathbf{K})\delta\mathbf{v} = h\mathbf{f}(\tilde{\mathbf{x}}). \quad (18)$$

If using the approximation in (11) we notice that $\mathbf{f}_d = \gamma\mathbf{K}\mathbf{v}$, where $\gamma = d/k$ is the ratio between the damping and the stiffness coefficients.

We can incorporate this damping force into the optimization formulation using *Rayleigh dissipation functions* [Lac07]. So we transform (18) to constraint space and get the following projection:

$$h(h+\gamma)\mathbf{A}\delta\lambda_{i+1} + \mathbf{e}_i = \mathbf{0}, \quad (19)$$

where $\mathbf{e}_i = \mathbf{c}(\mathbf{x}_i) + \gamma\mathbf{J}\mathbf{v}_i$; the second term is nothing more than the relative velocity along the constraint times γ. So this is a simple way to add more damping along the constraint directions which looks more natural than plain viscous drag in all directions. The downside is that you may have to pay the price of more iterations in order to keep the same amount of constraint violation as before damping. The final formula after adding regularization and energy preservation is:

$$(h\alpha(h+\gamma)\mathbf{A}_i + \varepsilon\mathbf{I})\delta\lambda_{i+1} + \alpha\mathbf{e}_i = \mathbf{0}. \quad (20)$$

As you can see now the projection formula includes both a stiffness parameter ($\varepsilon = 1/k$) and a damping parameter ($\gamma = d/k$) and so the method is fully equivalent to the implicit integration of a damped elastic system. Note that even in the case of infinite stiffness, the damping parameter can remain finite and we get (19).

7 ITERATIVE SOLVERS

As we already mentioned, the most popular methods for solving PBD are nonlinear relaxation methods: Jacobi, Gauss-Seidel or Successive Over Relaxation (SOR). Jacobi for instance has the following update formula: $\delta\lambda_{i+1} = \delta\lambda_i + \mu\mathbf{r}_i$, where $\mu_j = \omega/A_{jj}$, $\omega < 1$ and A_{jj} comes from the diagonal of $\mathbf{A}$. What these methods actually do is they solve each equation and its corresponding unknown separately (local solve) and iterate over all equations for a number of times in order to refine the solution. The local solve formula for one constraint c in the PBD method is:

$$h^2\nabla c(\mathbf{x})^T\mathbf{M}\nabla c(\mathbf{x})\delta\lambda + c(\mathbf{x}) = 0, \qquad (21)$$

where $\mathbf{M}$ and $\mathbf{x}$ correspond to the $s \geq 1$ particles involved in the constraint. We could also solve for more than one constraints simultaneously and we would obtain a system $\mathbf{L}\delta\lambda + \mathbf{c}(\mathbf{x}) = \mathbf{0}$ that we could solve using direct matrix inversion or a direct solver.

Looking more closely at the equation (13) we notice that it is not that different from (14). We can regard the change to the diagonal of the system matrix as a scaling through a relaxation factor as in [Jak01]. As noted in [MHHR07] this factor is highly nonlinear and we are now able to express this exact nonlinear relationship to the linear spring stiffness value: $\omega_j = (1 + (h^2 k_j A_{jj})^{-1})^{-1} < 1$, where k_j is the stiffness of spring j. On the other hand, if we use $\omega > 1$ we obtain SOR which may converge faster and produce better inextensibility, i.e. stiffer cloth for less iterations.

Another application of the equivalence between implicit integration and regularized PBD is to see what happens to the Conjugate Gradient method when we transform from configuration to constraint space. Let us start with the update formula for Steepest Descent:

$$\delta\mathbf{v}_{i+1} = \delta\mathbf{v}_i + \alpha_i\rho_i,$$

where ρ_i is the residual of the system in (12) which is related to the residual $\mathbf{r}_i$ by $\rho_i = \mathbf{T}\mathbf{r}_i$ and $\mathbf{r}_i = \mathbf{Q}\rho_i$, and

$$\alpha_i = \frac{\rho_i^T\rho_i}{\rho_i^T\mathbf{S}\rho_i}.$$

We would like to see how the constraint space transformation affects the update formula in constraint space. So we write α_i in terms of $\mathbf{r}_i$:

$$\alpha_i = \frac{\mathbf{r}_i^T\mathbf{Q}^T\mathbf{Q}\mathbf{r}_i}{\mathbf{r}_i^T\mathbf{Q}^T\mathbf{S}\mathbf{Q}\mathbf{r}_i}.$$

For the nominator we find that $\mathbf{Q}^T\mathbf{Q} = h^2k^2\tilde{\mathbf{A}}$, where $\tilde{\mathbf{A}} = \mathbf{J}\mathbf{M}^{-2}\mathbf{J}^T$ and the denominator is $\mathbf{Q}^T\mathbf{S}\mathbf{Q} = h^2k^3\mathbf{A}\mathbf{S}$. By using the properties of matrices $\mathbf{T}$ and $\mathbf{Q}$ we get that $\mu_i = k\alpha_i$ and by considering infinite stiffness, i.e. $\mathbf{S} \to h^2\mathbf{A}$ when $\varepsilon \to 0$, we get:

$$\mu_i = \frac{\mathbf{r}_i^T(h^2\tilde{\mathbf{A}})\mathbf{r}_i}{\mathbf{r}_i^T(h^2\mathbf{A})^2\mathbf{r}_i}.$$

If we ignore the tilde (or all the masses are 1) we see that we have obtained the formula for the Minimum Residual method which was used in [FM14]. It may be that the method breaks when $\mathbf{M}^{-1}$ is far different from its square and the correct formula needs to be used. Still we think this derivation is a strong argument for why CG does not work in constraint based methods and we need to transform it to a method that looks more like a minimum residual version of CG, e.g. Conjugate Residuals (CR) [Saa03].

8 CLOTH MODEL

The most straightforward application of the presented methods to cloth is through the use of a model made of particles and springs or rigid links connecting them. Such links correspond to a constraint function like $\mathbf{c}(\mathbf{x}_i, \mathbf{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\| - l_{ij}$, where l_{ij} is the initial rest length of the link and i and j are the indices of the two particles. Using three types of links for stretching, shearing and bending, one can obtain a full model of cloth. Details on how to build such links for quad meshes are given in many papers [Pro96, OAW06]. The main advantage of our method of *soft constraints* is that the stiffness of each constraint can now be expressed naturally as an elastic parameter (related to Young's modulus) instead of using non-linear attenuation factors like in [Jak01] and [MHHR07].

Figure 2: Simulation of a cloth model consisting of 6910 vertices and 13674 triangles using soft constraints

For irregular triangle meshes, one can also build bending links [Erl05], but one cannot distinguish between stretching and shearing links anymore. For quad meshes shearing is simple to express and is usually less stiff than stretching. Using the same stiffness coefficient for shearing as for stretching (infinite in the

case of PBD) leads to locking artifacts, as the cloth does not have enough degrees of freedom to move naturally. Lowering the stiffness value may help with the locking problem, but this causes stretching, which must be avoided at all cost for cloth.

Many other solutions have been proposed for locking [EB08], but we chose to use the model presented in [BW98] which separates out the stretching components (warp and weft) from shearing for each triangle. In general a constraint involving a number of particles implies calculating gradients corresponding to each particle $\nabla_i c(\mathbf{x}) = \frac{\partial c(\mathbf{x})}{\partial \mathbf{x}_i}$. Then we can solve that constraint independently using (21):

$$\delta\lambda = -\frac{c(\mathbf{x})}{\sum_{i=1}^{s} m_i^{-1} \|\nabla_i c(\mathbf{x})\|^2} = -\frac{c(\mathbf{x})}{\xi}. \tag{22}$$

The action of the bending links is dependent on stretching so we might want to use other measures for the curvature of the cloth. We could use directly the constraint between two triangles (4 vertices) defined in [MHHR07], as it expresses the same dihedral angle as in [BW98]. Still we chose to derive our own formulas using (22) and the assumption made in [BW98] that the lengths of the normals remain constant.

9 FINITE ELEMENT METHOD

The continuum formulation in [BW98] is actually a special treatment of the finite element method. The three constraints correspond to the strain components ε_{uu}, ε_{vv} and ε_{uv} that make up the planar symmetric Green-Lagrange strain tensor:

$$\varepsilon(\mathbf{x}) = \tfrac{1}{2}(\nabla\mathbf{w}\nabla\mathbf{w}^T - \mathbf{I}), \tag{23}$$

where $\mathbf{w} : \mathbb{R}^2 \to \mathbb{R}^3$ is a mapping from an undeformed mesh parametrization (u, v coordinates) to deformed vertex positions. Then the actual components are:

$$\varepsilon_{uu} = \tfrac{1}{2}(\mathbf{w}_u^T\mathbf{w}_u - 1), \tag{24}$$

$$\varepsilon_{vv} = \tfrac{1}{2}(\mathbf{w}_v^T\mathbf{w}_v - 1), \tag{25}$$

$$\varepsilon_{uv} = \varepsilon_{vu} = \mathbf{w}_u^T\mathbf{w}_v, \tag{26}$$

where by the subscript of $\mathbf{w}$ we signify partial derivation with respect to to u and v. By considering strain constant over a triangle (linear FEM) we can derive simple formulas for $\mathbf{w}_u$ and $\mathbf{w}_v$ like in [BW98] or [VMTF09].

The integral of the strain energy over a triangle is:

$$U_{\text{fem}} = \tfrac{a}{2}\hat{\varepsilon}(\mathbf{x})^T\mathbf{E}\hat{\varepsilon}(\mathbf{x}), \tag{27}$$

where a is the area of the triangle, $\hat{\varepsilon}^T = (\varepsilon_{uu}, \varepsilon_{vv}, \varepsilon_{uv})$ and $\mathbf{E}$ is a matrix that depends on the Young modulus E and the Poisson ratio ν (or equivalently on the Lamé coefficients) like the one given in [VMTF09] or [TWS07]. Note that the former expresses isotropic elasticity while the latter expresses orthotropic elasticity, i.e. different stiffness along warp and weft directions.

We use (27) to derive the true constraint function using the regularization framework as in [SLM06]:

$$\mathbf{c}(\mathbf{x}) = a^{\frac{1}{2}}\hat{\varepsilon}(\mathbf{x}). \tag{28}$$

The resulting three constraints (c_u, c_v, c_s) are are similar to the ones in [BW98] and their gradients form the Jacobian $\mathbf{J}_{\text{fem}} = (\nabla c_u^T, \nabla c_v^T, \nabla c_s^T)$. Note that in the most rigorous approach the area of the triangle $a(\mathbf{x})$ is also varying and its derivative should also be considered. We chose not to do so in our computations, but alternatively we could add an extra area constraint. Some authors use area and volume constraints together with edge constraints to improve on mass-spring soft body models [THMG04].

Now we can formulate the regularization condition:

$$a^{\frac{1}{2}}\hat{\varepsilon}(\mathbf{x}) + \mathbf{E}^{-1}\lambda = \mathbf{0}. \tag{29}$$

In the end we can apply the block local solve formula from Section 7, which is equivalent to other StVK linear FEM approaches like the one in [VMTF09]. We choose to apply this block approach only for the stretch components together, as the shear stress component is related only through a diagonal term to strain, and thus decoupled from the normal directions. The resulting 2x2 local linear system for the two stretching constraints is:

$$(h^2\mathbf{A} + \tilde{\mathbf{E}}^{-1})\delta\lambda + \hat{\varepsilon}(\mathbf{x}) = 0,$$

where in the case of isotropic materials

$$\tilde{\mathbf{E}}^{-1} = \frac{1}{E\sqrt{a}}\begin{pmatrix} 1 & \nu \\ -\nu & 1 \end{pmatrix}.$$

Notice that we divided equation (29) by the constant area term $a^{1/2}$ and obtained a CFM matrix that contains all the relevant continuous material parameters: Young's modulus, Poisson ratio and the triangle area (discretization measure). We can also add damping through the Rayleigh damping technique presented in Section 6 or the projection in Section 5 for better energy conservation. In the end we obtain a very accurate (iterations permitting) and physically correct model for simulating thin nonlinear elastic materials like the one in [VMTF09] based only on constraints (Figure 3).

10 RESULTS

We implemented a cloth simulator solely on a modular constraints architecture using C++ (single threaded). Depending on the level of accuracy or performance the user can choose between different constraint types, e.g. links or FEM triangles for stretching and shearing, different types of bending constraints, static collision or

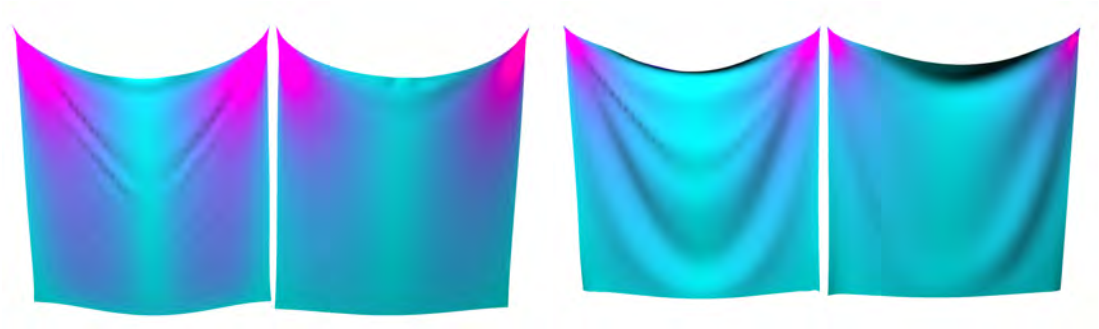

Figure 3: Two snapshots of a side by side real-time simulation of two 40x40 cloth pieces with the same Young's modulus E: regularized FEM constraints (left) and soft links (right); superimposed in purple is the strain map. FEM offers more realistic folds and the strain is better distributed throughout the cloth.

self collision constraints etc. Note that all these constraints are treated in the same solver loop. Regularization was implemented by modifying the diagonal of the system matrix using equation (13) or as described in Section 7. The resulting scheme can be modified to use the projection in Section 5 or we could add more damping (Section 6). These options can be added depending on the needs of the simulator and we denote them collectively as soft constraint methods for enhancing PBD. This also is why we do not provide any pseudo-code and hope that the readers will assemble themselves the simulator of choice.

Given our simulator is fully constraint based our collision response techniques are the same as the ones used in [Jak01, MHHR07] and for self collision we adapted the methods in [BFA02]. Friction is treated more accurately by being solved at every contact iteration in a similar fashion to cone complementarity programming [TA10]. We implemented cloth-mesh collisions by testing for triangle-point and edge-edge intersections between two triangles. For acceleration we used AABB trees for both the static mesh (pre-computed) and the cloth (rebuilt at every frame). A similar approach was used for accelerating self-collisions too.

We tested simulations mostly visually looking for obvious artifacts like jittering or instability. Our most common test scenario was a piece of cloth hanging by two corners, falling from a horizontal or vertical position, with different parameters or tessellation. Given the multitude of methods used and the differences between them it is hard to find a metric that measures well the quality of the simulation. We opted to measure the total energy - kinetic and potential (gravitational and elastic) and no damping, and chart its evolution in time (Figure 4). The NCG solver behaves well and has good convergence, but decays non-monotonically. The regularized PBD method is smoother, dissipates energy slower, but the Gauss-Seidel solver is less accurate. Energy preserving projection with $\alpha = \beta = 0.6$ offers even slower energy decay while the higher energy line is due to the kinetic energy of the oscillation. This is a good energy preserving property but it looks jittery and unnatural and we may need to add extra (non-artificial) damping. The closer we get to $\alpha = \beta = 0.5$ the more horizontal the graph becomes, i.e. full energy conservation, but this is dangerous territory for stability. We get the same results with a Symplectic Euler integrator with very small time steps and a small damping factor.

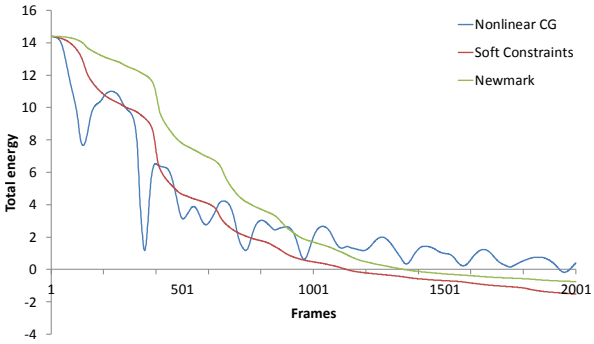

Figure 4: Total energy evolution in time for the simulation of a 10x10 rubber cloth (k = 2 N/m, 25 iterations) using NCG implicit integration (blue), regularized PBD (red) and regularized energy preserving projection (green).

For our damping method we measured the total energy minus the elastic potential in order to give a clearer picture of the velocity reduction (Figure 5). As you can see a damping factor of $\gamma = 10h$ gives a significant energy dissipation compared to soft projection (or PBD just as well). Reaching this level of dissipation so quickly is not possible using the method we compared against, i.e. reducing the relative velocity along the constraint direction (basically velocity projection). We set for the energy preserving projector $\alpha = \beta = 0.55$ and $\gamma = h$ in order to obtain as little artificial damping as possible while at same time damping the simulation just a bit less than PBD would normally do.

All simulations were performed in real time at 60 Hz, i.e. under 16 ms of computation time depending on

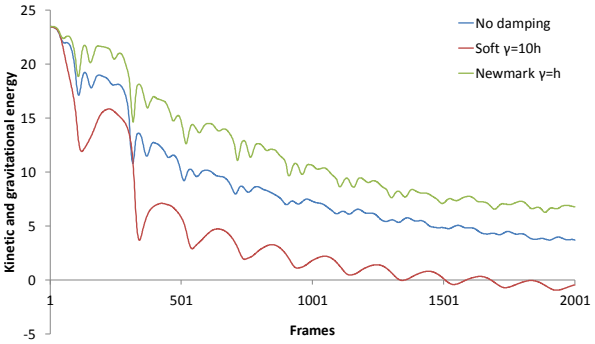

Figure 5: Damping response for the simulation of a 40x40 piece of cloth ($k = 2000$ N/m, 25 iterations) using regularized PBD (blue), aggressive damping (red) and slightly damped energy preserving projection (green).

cloth size, and with lower framerate for the dress in Figure 2 (up to 20 ms or more for the solver only).

11 CONCLUSIONS

We have shown that implicit integration of elastic systems is equivalent as an optimization problem to fast projection. Based on the analogy to PBD we derived a Nonlinear Conjugate Gradient implicit solver. Its drawback is that it is using an approximated force Jacobian but this is compensated by running more than one inexact Newton iterations.

After developing a method of switching between the two representations (configuration and constraint space) we proved that the regularized PBD method (soft constraints) replicates elastic behavior. We also showed how to preserve energy better or how to dissipate more when solving constraints. Note though that the viscous drag term of the Rayleigh damping matrix cannot be treated implicitly in this framework. Also, one can use a parallel version of the Conjugate Residuals algorithm to speed up the simulation. Finally we showed that accurate FEM simulation of cloth using constraints is possible and is no different from implicit integration. We believe that these are new and useful results for PBD.

We hope to use the Kawabata evaluation system in the future for real fabric modeling like in [VMTF09]. We also intend to optimize our simulator using parallel algorithms for multi-core and GPGPU.

12 ACKNOWLEDGMENTS

We would like to thank Framebreed animation studio for the 3D model of the dress and the body in Figure 2. We also want to thank Lucian Petrescu and Dongsoo Han for their input. Part of the research presented in this paper was supported by the Sectoral Operational Program Human Resources Development, 2014-2015, of the Ministry of Labor, Family and Social Protection through a Financial Agreement between University POLITEHNICA of Bucharest and AM POS DRU Romania, project ID 132395.

13 REFERENCES

[AH04] M. Anitescu and G. D. Hart. A Constraint Stabilized Time-Stepping Approach for Rigid Multibody Dynamics with Joints, Contact and Friction. Int. J. Numer. Meth. Eng., 60(14):2335–2371, 2004.

[Bar94] D. Baraff. Fast Contact Force Computation for Nonpenetrating Rigid Bodies. In ACM SIGGRAPH 1994 Papers, pages 23–34, 1994

[BW98] D. Baraff and A. Witkin. Large Steps in Cloth Simulation. In ACM SIGGRAPH 1998 Papers, pages 43–54, 1998.

[BKLS95] E. Barth, K. Kuczera, B. Leimkuhler, and R. D. Skeel. Algorithms for constrained molecular dynamics. J. Comp. Chem, 16:1192–1209, 1995.

[BKCW14] J. Bender, D. Koschier, P. Charrier, and D. Weber. Position-based simulation of continuous materials. Computers & Graphics, 44:1–10, 2014.

[BML*14] S. Bouaziz, S. Martin, T. Liu, L. Kavan, and M. Pauly. Projective dynamics: Fusing constraint projections for fast simulation. ACM Trans. Graph., 33(4):154:1–154:11, 2014.

[BFA02] R. Bridson, R. Fedkiw, and J. Anderson. Robust Treatment of Collisions, Contact and Friction for Cloth Animation. In ACM SIGGRAPH 2002 Papers, pages 594–603, 2002.

[BMF03] R. Bridson, S. Marino, and R. Fedkiw. Simulation of clothing with folds and wrinkles. In Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pages 28–36, 2003.

[Cat10] E. Catto. Soft Constraints: Reinventing the Spring. Game Developer Conference, 2010.

[CK02] K.-J. Choi and H.-S. Ko. Stable but responsive cloth. In Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '02, pages 604–611, 2002.

[EEH00] B. Eberhardt, O. Etzmuß, and M. Hauth. Implicit-explicit schemes for fast animation with particle systems. In In Eurographics Computer Animation and Simulation Workshop, pages 137–151, 2000.

[EB08] E. English and R. Bridson. Animating developable surfaces using nonconforming elements. ACM Trans. Graph., 27(3):66:1–66:5, 2008.

[Erl05] K. Erleben et al. Physics-based Animation. Charles River Media, 2005.

[Erl07] K. Erleben. Velocity-based Shock Propagation for Multibody Dynamics Animation. ACM Trans. Graph., 26, 2007.

[FM14] M. Francu and F. Moldoveanu. An improved jacobi solver for particle simulation. In VRIPHYS 14, pages 125–134, 2014.

[Gol10] A. R. Goldenthal. Implicit Treatment of Constraints for Cloth Simulation. PhD thesis, 2010.

[GHF*07] R. Goldenthal, D. Harmon, R. Fattal, M. Bercovier, and E. Grinspun. Efficient Simulation of Inextensible Cloth. In ACM SIGGRAPH 2007 Papers, 2007.

[HLW06] E. Hairer, C. Lubich, and G. Wanner. Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations. Springer, 2006

[HET01] M. Hauth, O. Etzmuss, and U. Tübingen. A high performance solver for the animation of deformable objects using advanced numerical methods. In In Proc. Eurographics 2001 (2001), pages 319–328, 2001.

[HCJ*05] M. Hong, M.-H. Choi, S. Jung, S. Welch, and J. Trapp. Effective constrained dynamic simulation using implicit constraint enforcement. In Robotics and Automation, ICRA 2005, pages 4520–4525, 2005.

[Jak01] T. Jakobsen. Advanced Character Physics. In Game Developers Conference Proceedings, pages 383–401, 2001.

[Lac07] C. Lacoursière. Ghosts and machines: regularized variational methods for interactive simulations of multibodies with dry frictional contacts. PhD thesis, Umeå University, 2007.

[Lan70] C. Lanczos. The Variational Principles of Mechanics. Dover Publications, 1970.

[LBOK13] T. Liu, A. W. Bargteil, J. F. O'Brien, and L. Kavan. Fast simulation of mass-spring systems. ACM Trans. Graph., 32(6):214:1–214:7, 2013.

[MM13] M. Macklin and M. Müller. Position based fluids. ACM Trans. Graph., 32(4):104:1–104:12, July 2013.

[MSW14] D. L. Michels, G. A. Sobottka, and A. G. Weber. Exponential integrators for stiff elastodynamic problems. ACM Trans. Graph., 33(1):7:1–7:20, Feb. 2014.

[MHHR07] M. Müller, B. Heidelberger, M. Hennix, and J. Ratcliff. Position Based Dynamics. J. Vis. Comun. Image Represent., 18(2):109–118, 2007.

[MSJT08] M. Müller, J. Stam, D. James, and N. Thürey. Real time physics: Class notes. In ACM SIGGRAPH 2008 Classes, pages 88:1–88:90, 2008.

[OAW06] S. Oh, J. Ahn, and K. Wohn. Low damped cloth simulation. Vis. Comput., 22(2):70–79, 2006.

[Pro96] X. Provot. Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behavior. In In Graphics Interface, pages 147–154, 1996.

[Saa03] Y. Saad. Iterative Methods for Sparse Linear Systems. Society for Industrial and Applied Mathematics, 2nd edition, 2003.

[SLM06] M. Servin, C. Lacoursière, and N. Melin. Interactive simulation of elastic deformable materials. In SIGRAD 2006 Conference Proceedings, pages 22–32, 2006.

[She94] J. R. Shewchuk. An Introduction to the Conjugate Gradient Method Without the Agonizing Pain. Technical report, Pittsburgh, PA, USA, 1994.

[SSB13] F. Sin, D. Schroeder, and J. Barbic. Vega: Nonlinear fem deformable object simulator. Comput. Graph. Forum, 32(1):36–48, 2013.

[Smi06] R. Smith. Open dynamics engine v0.5 user guide. 2006.

[Sta09] J. Stam. Nucleus: Towards a unified dynamics solver for computer graphics. In Computer Aided Design and Computer Graphics, pages 1–11, 2009.

[SD06] A. Stern and M. Desbrun. Discrete geometric mechanics for variational time integrators. In ACM SIGGRAPH 2006 Courses, SIGGRAPH '06, pages 75–80, 2006.

[ST96] D. Stewart and J. C. Trinkle. An Implicit Time-Stepping Scheme for Rigid Body Dynamics with Coulomb Friction. Int. J. Numer. Meth. Eng., 39:2673–2691, 1996.

[TA10] A. Tasora and M. Anitescu. A Convex Complementarity Approach for Simulating Large Granular Flows. J. Comput. Nonlinear Dynam., 5, 2010.

[THMG04] M. Teschner, B. Heidelberger, M. Muller, and M. Gross. A versatile and robust model for geometrically complex deformable solids. In Proceedings of the Computer Graphics International, pages 312–319, 2004.

[TPS09] B. Thomaszewski, S. Pabst, and W. Straßer. Continuum-based strain limiting. Comput. Graph. Forum, 28(2):569–576, 2009.

[TWS07] B. Thomaszewski, M. Wacker, and W. Straßer. Advanced topics in virtual garment simulation – part 1, 2007.

[VMTF09] P. Volino, N. Magnenat-Thalmann, and F. Faure. A simple approach to nonlinear tensile stiffness for accurate cloth simulation. ACM Trans. Graph., 28(4):105:1–105:16, 2009.

[WN99] S. J. Wright and J. Nocedal. Numerical optimization, volume 2. Springer New York, 1999.