Workshop on Virtual Reality Interaction and Physical Simulation VRIPHYS (2014)
J. Bender, C. Duriez, F. Jaillet, and G. Zachmann (Editors)

An Improved Jacobi Solver for Particle Simulation

M. Frâncu and F. Moldoveanu

University POLITEHNICA Bucharest, Romania

Abstract

This paper presents a new method for simulating particles for computer graphics and video games, based on an improved Jacobi solver and a hybrid approach between velocity time stepping and position based dynamics. Current constrained dynamics solvers use relaxation iterative methods like Gauss-Seidel or Jacobi. We propose a new iterative method based on a minimum residual variant of the Conjugate Gradient algorithm and show that it can be formulated as an improvement to the Jacobi method. We also describe an adaptation of position based dynamics to better handle contact and friction and allow tight two way coupling with velocity level methods.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically Based Modeling

1. Introduction

Particle systems can be used to model all kind of mechanical phenomena including granular matter, fluids, cloth, deformable objects and even rigid bodies. Granular matter has been used extensively in visual effects and computer generated animations such as Spiderman or Rise of the Guardians [ABC*07]. Cloth and soft bodies in general are now ubiquitous in movies and becoming more so in games. This is why the simulation methods still need to become faster and more robust in order to handle a growing number of objects.

In this paper we present a unified approach to simulating granular matter and cloth using a constrained dynamics approach. Our hybrid dynamics method offers tight coupling between all simulated objects as all constraints are treated in the same solver loop. Also our improved Jacobi solver shows convergence similar to the popular Gauss-Seidel method, thus allowing for more efficient parallel implementations.

2. Related work

Granular matter has been an area of research in computational mechanics for decades. The method of choice is usually the *discrete element method* (DEM) which treats the granules as elastic billiard balls. The DEM method was used in graphics too [BYM05, ATO09, Har07]. Another approach was a continuum based one, considering the granular matter a special kind of fluid [ZB05, NGL10]. This was followed by a Lagrangian version derived from the *smooth particle hydrodynamics* method for simulating fluids [AO11, IWT12].

An alternative to DEM is the *non-smooth constrained dynamics* approach where the particles are considered fully rigid and this is the path we are following. In fact the method was developed for the more general case of rigid bodies, but that can be turned into an advantage given the granules can have any shape other than spherical [BYM05]. A great deal of articles have been written on the subject of multi-body dynamics with contact and friction [ST96, AH04, AT10, Lac03], many in the computer graphics community [Bar94, Erl07, BETC14, TBV12, KSJP08], and some explicitly on the subject of granular flow [TA10, RA05, LSB10].

Rigid bodies can be simulated with other methods than constrained dynamics, e.g. the penalty method [BZX14]. Other approaches consider the rigid body as a collection of particles and contact forces are computed using DEM or in other ways [TSIHK06, Jak01]. The particles can move under rigid transformations [Har07] or be constrained together to form a composite rigid object [Cou12, MMCK14].

Cloth has been traditionally simulated as a mass-spring model [Pro96] and great effort has been put into making the simulation stable at large time steps [BW98]. In games *position based dynamics* is usually preferred (PBD) [Jak01, MHHR07, GHF*07]. The approach in [BBD09] and [How11] is similar to our hybrid algorithm, but has separate passes for velocity and position. Volumetric deformable objects can be modeled by connecting the particles together in a lattice fashion, by adding internal pressure or through shape matching [MHHR07, BMOT13]. A unified particle

DOI: 10.2312/vriphys.20141231

based simulator using constraints is described in [Sta09]. Many similar approaches to ours involving PBD are presented in a couple of recent papers [MMCK14, DCB14].

Iterative methods are currently the preferred way of solving constrained mechanical systems for real-time. Using exact methods can become infeasible when adding contact and friction for more than a few hundred bodies [BETC14]. The fastest and most robust iterative method used in the present is Gauss-Seidel (GS) [Cat05, Erl07]. GS also knows improvements such as line search with conjugate directions [SHNE10a] or subspace minimization [SHNE10b].

Jacobi is another relaxation method, closely resembling GS, but it converges slower and needs modifications to remain stable. Still it is preferred to GS for parallel implementations as it can process each constraint independently of the others [TBV12].

The Conjugate Gradient (CG) method has a good reputation for solving linear systems as it has better convergence than matrix splitting methods like Jacobi or GS [Saa03]. Even though it was used for implicit integration of mass-spring models [BW98] it has never gained traction in constrained dynamics simulations. There have been attempts at using it [RA05], but many argued against its applicability to efficient simulations mainly due to erratic convergence [Erl07, Ton12, Mor05]. Our method is based on a minimum residual variant of gradient descent algorithms as it gurantees decreasing residual energy and is more stable. After optimizing it we arrived at a version of Jacobi with improved convergence. A minimum residual method (GPMINRES) was used in [HATN12] but it is only one of several methods described, it is more complex than ours and it was chosen for a different reason, i.e. to handle positive semi-definite matrices. The line search Jacobi algorithm offers similar improvements [TBV12, CPS92], but we found little resemblance in the mathematical formulation to our method.

3. Preliminaries

3.1. Constrained dynamics

The mathematical formulation of the constrained dynamics of a particle system has the form of a *differential complementarity problem* (DCP) [ST96]:

$$\mathbf{M}\dot{\mathbf{v}} = \mathbf{J}^T\boldsymbol{\lambda} + \mathbf{F}, \tag{1}$$

$$\dot{\mathbf{x}} = \mathbf{v}, \tag{2}$$

$$\mathbf{C}_b(\mathbf{x}) = \mathbf{0}, \tag{3}$$

$$\mathbf{C}_u(\mathbf{x}) \geq \mathbf{0}, \boldsymbol{\lambda}_u \geq \mathbf{0}, \tag{4}$$

$$\boldsymbol{\lambda}_u^T\mathbf{C}_u(\mathbf{x}) = \mathbf{0}, \tag{5}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{v}$ is the generalized velocity, $\mathbf{x}$ is the generalized position, $\mathbf{F}$ is the generalized external force, $\boldsymbol{\lambda}$ are the Lagrange multipliers, $\mathbf{C}(\mathbf{x})$ is the vector constraint function and $\mathbf{J} = \nabla\mathbf{C}(\mathbf{x})$ its Jacobian. For a particle system the mass matrix is a block diagonal matrix made up of $d \times d$ square element matrices $\mathbf{M}_i = \text{diag}(m_i)$, where d is the number of space dimensions and m_i are the particle masses. The constraint function is split in two, $\mathbf{C} = (\mathbf{C}_b\,\mathbf{C}_u)^T$: bilateral (equality) constraints and unilateral (inequality) constraints. The same goes for the Lagrange multipliers $\boldsymbol{\lambda} = (\boldsymbol{\lambda}_b\,\boldsymbol{\lambda}_u)^T$, and the Jacobian $\mathbf{J} = (\mathbf{J}_b\,\mathbf{J}_u)^T$.

The system (1)-(5) can be discretized using a semi-implicit integrator (a low-order symplectic Euler scheme that closely preserves energy) and through constraint linearization, resulting in a *mixed linear complementarity problem* (MLCP) known as *velocity time-stepping* [AH04]:

$$\mathbf{v}^{(k+1)} = \mathbf{v}^{(k)} + h\mathbf{W}\mathbf{J}^T\boldsymbol{\lambda} + h\mathbf{W}\mathbf{F}, \tag{6}$$

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + h\mathbf{v}^{(k+1)}, \tag{7}$$

$$\mathbf{0} = \mathbf{J}_b\mathbf{v}^{(k+1)} + \frac{\gamma}{h}\mathbf{C}_b(\mathbf{x}^{(k)}), \tag{8}$$

$$\mathbf{0} \leq \mathbf{J}_u\mathbf{v}^{(k+1)} + \frac{\gamma}{h}\mathbf{C}_u(\mathbf{x}^{(k)}) \perp \boldsymbol{\lambda}_u \geq \mathbf{0}, \tag{9}$$

where the superscript between brackets indicates the value at time t_k or $t_{k+1} = t_k + h$, h is the time step, $\mathbf{W} = \mathbf{M}^{-1}$, $\mathbf{J} = \nabla\mathbf{C}(\mathbf{x}^{(k)})$, $\gamma \leq 1$ is the Baumgarte stabilization factor [Cat05], and the perpendicularity symbol indicates complementarity like in (5).

3.2. Coulomb friction

Unilateral constraints are mostly used to model non-penetrating contact and can also be used to model friction. This is why we differentiate between normal and tangential constraints $(\mathbf{C}_n, \mathbf{C}_t)$ and their corresponding Jacobians $(\mathbf{J}_n, \mathbf{J}_t)$ and multipliers $(\boldsymbol{\lambda}_n, \boldsymbol{\lambda}_t)$. By looking at a single contact point we identify the following elements: a contact normal $\mathbf{n}$ and two tangential vectors $\mathbf{t}_1$ and $\mathbf{t}_2$ forming an orthonormal frame, a contact force made up of a normal component preventing penetration $\mathbf{F}_n = \lambda_n\mathbf{n}$ and a friction force:

$$\mathbf{F}_t = \lambda_{t_1}\mathbf{t}_1 + \lambda_{t_2}\mathbf{t}_2, \tag{10}$$

and a relative tangential velocity $\mathbf{v}_t$, whose magnitude we'll denote by v_t (sliding speed).

We can now express the Coulomb friction model as consisting of two laws [BETC14, ST96]: *dynamic friction* - if the bodies are sliding ($v_t \neq 0$) the friction force acts against the tangential velocity and has the expression

$$\mathbf{F}_t = -\mu\lambda_n\frac{\mathbf{v}_t}{v_t}, \tag{11}$$

and *static friction* - if the bodies are sticking ($v_t = 0$) the contact force has to be inside the friction cone defined by

$$F_t = \sqrt{\lambda_{t_1}^2 + \lambda_{t_2}^2} \leq \mu\lambda_n. \tag{12}$$

The two laws can be summed up together using the *principle of maximum dissipation* [BETC14]:

$$(\lambda_{t_1}\,\lambda_{t_2})^T = \arg\min_{F_t \leq \mu\lambda_n}(\mathbf{F}_t \cdot \mathbf{v}_t). \tag{13}$$

Equation (13) applied to all the contacts in the system together with the MLCP in (6)-(9) form a *differential variational inequality* (DVI) problem [AT10].

One way of solving the DVI is to discretize the friction cone as a polyhedral cone (faceted pyramid) and convert the problem to a *linear complementarity problem* (LCP) [ST96]. To get from this model to the one used in computer graphics and video games that uses iterative solvers we need to make the following simplifications [PNE10]: use a square pyramid defined by the directions $\mathbf{t}_1$ and $\mathbf{t}_2$, use box constraints for friction ($\boldsymbol{\lambda}_t \le |\mu\boldsymbol{\lambda}_n|$) and contact ($0 \le \boldsymbol{\lambda}_n \le \infty$), and drop the principle of maximum dissipation. In the end we get a *box linear complementarity problem* (BLCP) [Cat05]:

$$\mathbf{y} = \mathbf{A}\boldsymbol{\lambda} + \mathbf{b}, \tag{14}$$

$$\boldsymbol{\lambda}_{lo} \le \boldsymbol{\lambda} \le \boldsymbol{\lambda}_{hi}, \tag{15}$$

where

$$\mathbf{A} = h\mathbf{J}\mathbf{W}\mathbf{J}^T, \tag{16}$$

$$\mathbf{b} = \mathbf{J}\mathbf{v}^{(k)} + h\mathbf{J}\mathbf{W}\mathbf{F} + \frac{\gamma}{h}\mathbf{C}(\mathbf{x}^{(k)}), \tag{17}$$

assuming $\mathbf{C}_t(\mathbf{x})$ is always 0, and $\boldsymbol{\lambda}_{lo}$ and $\boldsymbol{\lambda}_{hi}$ are the lower and upper bounds of the Lagrange multipliers as described in the box constraints above. For a more rigorous complementarity formulation see [Lac03, PNE10].

3.3. Position based dynamics

The DCP without unilateral constraints (1)-(3) is called a *differential algebraic equation* (DAE) or a differential equation on a manifold [HLW06]. A different way other than (6)-(8) of discretizing the DAE is through a nonlinear formulation at position level with implicit constraint directions [GHF*07]:

$$\mathbf{x}^{(k+1)} = \tilde{\mathbf{x}} + \Delta\mathbf{x}, \tag{18}$$

$$\mathbf{v}^{(k+1)} = \tilde{\mathbf{v}} + \frac{1}{h}\Delta\mathbf{x}, \tag{19}$$

$$\mathbf{C}_b(\mathbf{x}^{(k+1)}) = \mathbf{0}, \tag{20}$$

where

$$\tilde{\mathbf{v}} = \mathbf{v}^{(k)} + h\mathbf{W}\mathbf{F}, \tag{21}$$

$$\tilde{\mathbf{x}} = \mathbf{x}^{(k)} + h\tilde{\mathbf{v}}, \tag{22}$$

$$\Delta\mathbf{x} = h^2\mathbf{W}\nabla\mathbf{C}_b^T(\mathbf{x}^{(k+1)})\boldsymbol{\lambda}. \tag{23}$$

The values $\tilde{\mathbf{v}}$ and $\tilde{\mathbf{x}}$ represent the unconstrained velocity and position integration steps.

The system (18)-(20) can be solved through *projection* [HLW06], which in turn can be expressed as a series of *sequential quadratic programming* (SQP) steps [GHF*07]. In result we get a Newton process for solving equation (20):

$$\mathbf{0} = \mathbf{C}(\mathbf{x}^l) + h^2\mathbf{J}^l\mathbf{W}(\mathbf{J}^l)^T\boldsymbol{\lambda}^l, \tag{24}$$

$$\mathbf{x}^{l+1} = \mathbf{x}^l + h^2\mathbf{W}(\mathbf{J}^l)^T\boldsymbol{\lambda}^l, \tag{25}$$

where the superscript l indicates the iteration number, $\mathbf{x}^0 = \tilde{\mathbf{x}}$, and $\mathbf{J}^l = \nabla\mathbf{C}(\mathbf{x}^l)$. The *position based dynamics* (PBD) method employs nonlinear iterative solvers for (20) [Jak01, MHHR07]. PBD can also handle contact and friction as described in [ST96] by adding equations (4)-(5) to (18)-(20) in order to form a *nonlinear complementarity problem* (NCP).

3.4. Iterative methods

The most popular methods for solving the BLCP (14)-(15) are relaxation or matrix splitting methods. They are widely used for solving large sparse linear systems [Saa03] and they have been adapted for LCPs using projection [CPS92]. We will further denote the projection step through the function: $\text{proj}(\lambda) = \text{clamp}(\lambda, \lambda_{lo}, \lambda_{hi})$ and consider h included in $\boldsymbol{\lambda}$.

3.4.1. Projected Jacobi

Projected Jacobi has the form:

$$\boldsymbol{\lambda}^{l+1} = \text{proj}(\boldsymbol{\lambda}^l - \omega\mathbf{D}^{-1}\mathbf{r}^l), \tag{26}$$

where l denotes the iteration number, $\mathbf{D} = \text{diag}(\mathbf{A})$, $\mathbf{r}^l = \mathbf{A}\boldsymbol{\lambda}^l + \mathbf{b}$ is the residual vector and $\omega < 2/\rho(\mathbf{A})$ is a factor that assures convergence. In order to simplify the algorithm, the velocities are updated at each iteration:

$$\mathbf{v}^{l+1} = \mathbf{v}^l + \mathbf{W}\mathbf{J}^T\Delta\boldsymbol{\lambda}^{l+1}, \tag{27}$$

where $\Delta\boldsymbol{\lambda}^{l+1} = \boldsymbol{\lambda}^{l+1} - \boldsymbol{\lambda}^l$ is the magnitude of the corrective impulse. One can also divide this impulse by the number of incident constraints to the current body being updated (n_{b_i}) and omit ω altogether from (26). Using (27) for the velocity time stepping problem in (14)-(17) we can also derive a simpler expression for the residual:

$$\mathbf{r}^l = \mathbf{J}\mathbf{v}^l + \boldsymbol{\delta}, \tag{28}$$

where $\boldsymbol{\delta} = \frac{\gamma}{h}\mathbf{C}(\mathbf{x}^0)$ can be precomputed ($\mathbf{x}^0 = \mathbf{x}^{(k)}$). The residual is thus nothing else than the relative velocity along the constraint direction plus a position stabilization term $\boldsymbol{\delta}$ depending on the position at the beginning of the time step $\mathbf{x}^0$ and the Baumgarte factor divided by the time step.

3.4.2. Projected Gauss-Seidel (PGS)

PGS is very similar to Jacobi with the difference that it uses the updated velocities right away, speeding up convergence:

$$\lambda_i^{l+1} = \text{proj}(\lambda_i^l - \frac{\omega}{d_i}\bar{\mathbf{r}}), \tag{29}$$

where i is the body index, $\omega \ge 1$ is a *successive over-relaxation* (SOR) factor, and $d_i = \mathbf{D}_{ii}$ is the inverse effective mass seen by the constraint, e.g. for a constraint between two particles $d_i = \frac{1}{m_{i_1}} + \frac{1}{m_{i_2}}$, where i_1 and i_2 are the particle indices. The residual $\bar{\mathbf{r}}$ is computed based on the most up-to-date velocities $\bar{\mathbf{v}} = (v_1^{l+1} \dots v_{i-1}^{l+1} v_i^l \dots v_n^l)^T$.

4. Minimum residual methods

In this section we will present a couple of methods inspired by the Conjugate Gradient algorithm and in the end describe a way of improving the convergence of the Jacobi method, making it comparable to Gauss-Seidel.

4.1. Projected Minimum Residual

The Steepest Descent (SD) algorithm for solving linear systems [She94] has the following update rule:

$$\boldsymbol{\lambda}^{l+1} = \text{proj}(\boldsymbol{\lambda}^l - \alpha^l \mathbf{r}^l), \tag{30}$$

where for α we use the formula from the Minimum Residual (MR) method [Saa03]:

$$\alpha^l = \frac{(\mathbf{r}^l)^T \mathbf{A}\mathbf{r}^l}{(\mathbf{A}\mathbf{r}^l)^T(\mathbf{A}\mathbf{r}^l)}. \tag{31}$$

We found that α needs to vary smoothly in order to prevent erratic converge and also that for stability reasons $\alpha \leq \alpha_0 = 1/\rho(\mathbf{A})$ must hold. We tried setting α to a close estimate of α_0 instead of (31) (computed at the beginning of the frame using the power iteration method for determining the largest eigenvalue of a matrix) and found it to be a robust optimization. Note also the similarity with Jacobi where the value of α differs for each constraint and can be written as

$$\alpha_i = \frac{\omega}{d_i}. \tag{32}$$

This is why the two methods have similar convergence rates.

4.2. Projected Conjugate Residuals

The Conjugate Gradient (CG) makes an improvement over SD by adding a descent direction and an extra update step:

$$\boldsymbol{\lambda}^{l+1} = \text{proj}(\boldsymbol{\lambda}^l - \alpha^l \mathbf{d}^l), \tag{33}$$

$$\mathbf{d}^{l+1} = \mathbf{r}^{l+1} + \beta^{l+1}\mathbf{d}^l, \tag{34}$$

where again we use a formula for β from the Conjugate Residuals (CR) method [Saa03]:

$$\beta^{l+1} = \frac{(\mathbf{r}^{l+1})^T \mathbf{A}\mathbf{r}^{l+1}}{(\mathbf{r}^l)^T \mathbf{A}\mathbf{r}^l}. \tag{35}$$

We found experimentally that β grows very rapidly from 0 to 1 and then stays almost constant. Also clamping it below 1 improved stability (see Figure 1). Given that the choice of β is not unique [HZ06], we chose to approximate (35) with:

$$\beta^l = \min\left(a\left(\frac{l}{l_{max}}\right)^b, 1\right), \tag{36}$$

where $a \geq 1$ and $b < 1$ and l_{max} is the maximum number of iterations. We obtained very good results and an even more stable simulation. In order to increase convergence one can make β grow steeper by lowering b and increasing a, but this may introduce jitter. We found $a = 1$ and $b = 0.6$ to be good values. Both CR and the power function optimized version have very good convergence, similar to Gauss-Seidel or even better in many cases (see Figure 2).

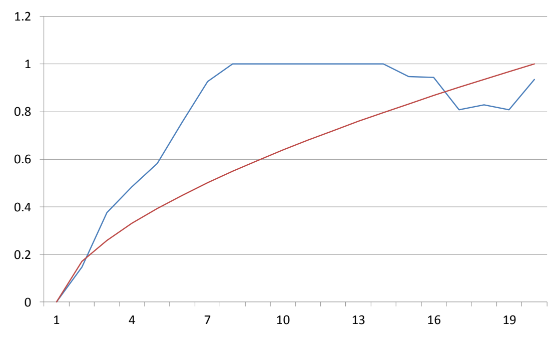


Figure 1: Plot of β over 20 iterations of the CR solver: original formula clamped below 1 (blue) and power function approximation (red) - PBD simulation.

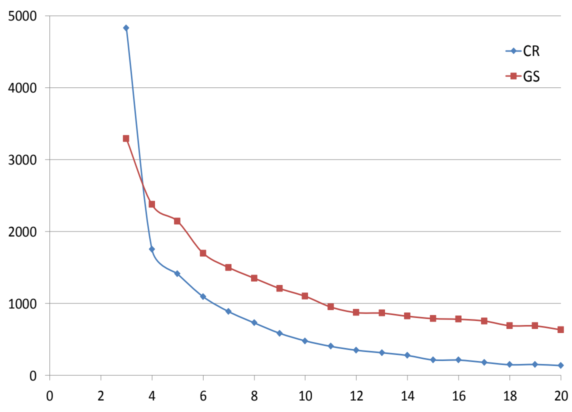


Figure 2: Plot of the total stretching at equilibrium of a 50x50 piece of cloth relative to the number of iterations for GS (red) and optimized CR (blue).

4.3. Improved Jacobi

If we set α fixed in the CR method just like we did for MR we can make the following simplification by using the descent direction (34) from the previous iteration:

$$\Delta\boldsymbol{\lambda}^{l+1} = -\alpha\mathbf{d}^l = -\alpha(\mathbf{r}^l + \beta^l\mathbf{d}^{l-1}) = -\alpha\mathbf{r}^l - \beta^l\Delta\boldsymbol{\lambda}^l. \tag{37}$$

This way we don't need to store the descent direction any more and we get a simpler version of CR. We still have to provide a value for α and we could use α_0, but we can make things even simpler by using the values from (32) and obtain a version of Jacobi with increased convergence:

$$\boldsymbol{\lambda}^{l+1} = \text{proj}(\boldsymbol{\lambda}^l - \frac{\omega}{d_i}\mathbf{r}^l - \beta^l\Delta\boldsymbol{\lambda}^l), \tag{38}$$

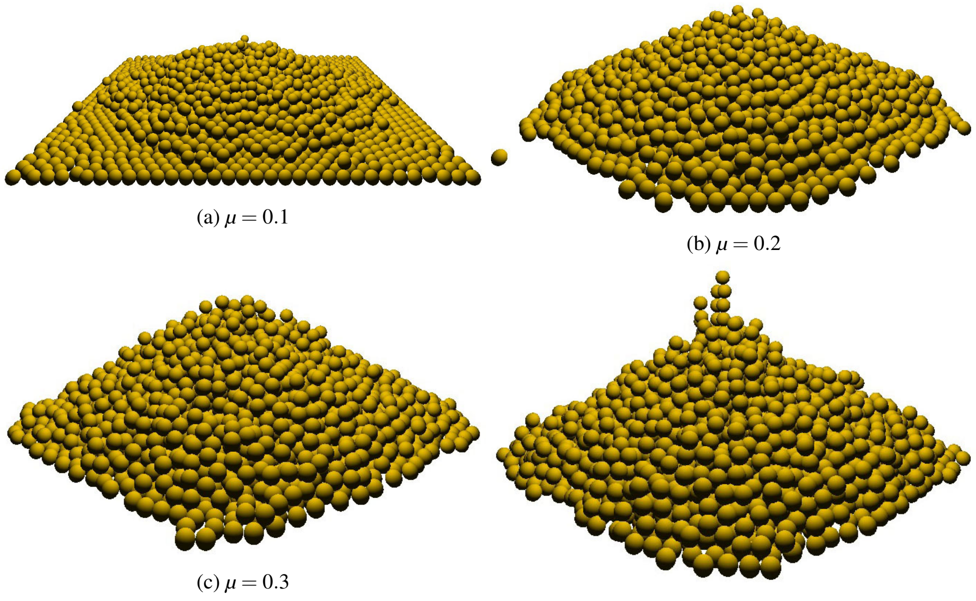

(a) $\mu = 0.1$ (b) $\mu = 0.2$ (c) $\mu = 0.3$ (d) $\mu = 0.4$

Figure 3: Sand piles formed by dropping 3000 particles using VTS with different friction coefficients (15 iterations, $\gamma = 0.5$).

where $\omega \leq 1$. We tested this improvement on GS too, but we had to lower the value of ω a lot in order to stabilize the simulation. Also our focus is on the Jacobi method as it is easier to paralellize.

5. Handling friction

The most popular way to handle friction in games and computer graphics is to use the square pyramid model [Erl07]. We chose not to do the simplification in [Cat05] (constant normal push) in order to have accurately coupled friction. This approach adds two tangential constraints for friction at each contact and so performance is affected.

Another way of solving the DVI (6)-(9), (13) is by casting it to a *cone complementarity problem* (CCP) [AT10]. In simple words this reduces to calculating the friction impulse that zeros the tangential velocity and projecting it to the smooth cone [Ton12]. In order to do this we need to add a relaxation term to (9):

$$\mathbf{0} \leq \mathbf{J}_u \mathbf{v}^{(k+1)} + \frac{\gamma}{h}\mathbf{C}_u(\mathbf{x}^{(k)}) - \mu\sqrt{v_{i,t_1}^2 + v_{i,t_2}^2} \perp \boldsymbol{\lambda}_u \geq \mathbf{0}, \quad (39)$$

or equivalently to each component of $\mathbf{b}$ (17) and replace it with $\mathbf{b}'$ in (14):

$$b_i' = b_i - \mu v_{i,t}, \quad (40)$$

where $v_{i,t} = \|\mathbf{v}_i\|$ is the tangential sliding speed at the ith contact and $\mathbf{v}_i = (v_{i,t_1}\ v_{i,t_2})^T$. As it works along a single direction, using a CCP approach can prove to be faster. We found that even without the relaxation term in (40) there are no noticeable convergence issues.

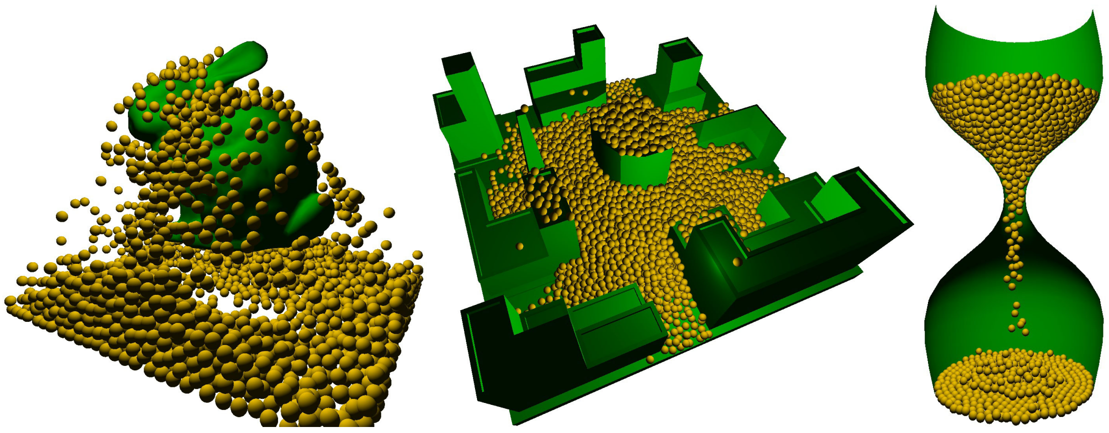
The last resort for optimizing friction is to apply it at the end, usually in a single iteration, as a post-processing step like in [MHHR07]. This process can be viewed as a simple kind of staggered approach [KSJP08, BETC14] and it usually generates weak and unrealistic friction forces.

6. Hybrid dynamics

The position based dynamics (PBD) method is very well suited for simulating complex articulated structures like cloth, where the constraint directions are rapidly changing. The velocity time stepping (VTS) method (14)-(15) cannot handle these situations properly as it's computing the Jacobian only at the initial time t_k. PBD offers the best accuracy for bilateral constraints and that is why we would like to use it to minimize position drift and for coupling with cloth.

PBD can handle contacts by iteratively solving the NCP (18)-(20) as a sequence of LCP sub-problems [ST96, Jak01, MHHR07]. Some authors and engines use it as a post-processing stabilization step [AH04]. But it does suffer from a couple of drawbacks: the calculated impulses are usually too high unless a high collision tolerance is used, and the friction forces are very roughly approximated after the solver has completed [Jak01, MHHR07]. Some existing PBD simulators already address these problems like for instance [MMCK14], but our hybrid approach is different.

Regarding the first problem we noticed that the impacts in VTS are handled more inelastically than PBD. That is because the VTS solver tries to bring the normal velocity close to zero immediately, while penetration is eliminated over the



(a) Bunny (b) City (c) Hourglass

Figure 4: Granular matter falling over or trough various meshes.

course of many frames. On the other hand PBD applies any impulse necessary to zero the constraint error. So, in order to soften the impacts, we choose to use relative velocities too (along the constraint directions) and relax the enforcement of position corrections inside PBD.

Also in order to accommodate friction inside the PBD solver we need to explicitly update the velocities at each iteration. In [Jak01] velocities are implicitly updated by the fixed time step Verlet integrator, while in [MHHR07] they are approximated at the end by a finite difference.

Finally, in order to address both issues we need to accumulate the normal and friction forces over all iterations and properly clamp them as in [Cat05]. For this purpose we consider that the constraint directions don't change at every iteration and we clamp the impulse magnitudes directly. The end result is a hybrid method between VTS and PBD that we called Sequential Positions (SP). We present a pseudocode description in Algorithm 1 using our improved Jacobi method, but any other iterative method could be used.

The two constants k_v and k_c can vary between 0 and 1 and control the mix between the velocity and position error. If $k_v = 0$ and $k_c = 1$ we get PBD, and if $k_v = 1$ and $k_c = \gamma$ we get a sort of nonlinear VTS. We experimented with different other values and found that for example $k_v = 0.1$ can in many cases soften the impacts of PBD. The relaxation factor ω is the same as in (38) and has to be provided as input. The same goes for the mixing coefficients k_v and k_c, and a and b for approximating β. Although they provide flexibility, we hope to reduce the number of tweakable constants in the future.

One modification we could make to Algorithm 1 is to delay the first integration update of positions until after a few velocity iterations (with $k_v > 0$). This would make sure that the constraint violations are not as big as it would normally happen for PBD and the contact impulses are smaller too.

Algorithm 1 Sequential Positions (using improved Jacobi)

```
Unconstrained velocity step $\mathbf{v} = \tilde{\mathbf{v}}$ using (21)
Unconstrained position step $\mathbf{x} = \tilde{\mathbf{x}}$ using (22)
Detect collisions
for $l = 0$ to $l_{max} - 1$ do
    Compute $\beta(l)$ using (36)
    for each constraint $C_i$ do
        Compute position error $\delta = C_i(\mathbf{x})$
        Compute constraint direction $\mathbf{J}_i = \nabla C_i(\mathbf{x})$
        Compute velocity error $\nu = \mathbf{J}_i \mathbf{v}$
        $\alpha = \omega m_e$, where $m_e = 1/d_i$ is the effective mass
        $r = k_v \nu + \frac{k_c}{h} \delta$
        Compute $\Delta\lambda_i$ using $\alpha, \beta$ and (38)
        $\bar{\lambda} = \lambda_i$
        $\lambda_i = \text{proj}(\lambda_i + \Delta\lambda_i)$
        $\Delta\lambda_i = \lambda_i - \bar{\lambda}$
    end for
    $\mathbf{v} = \mathbf{v} + \mathbf{W}\mathbf{J}^T \Delta\boldsymbol{\lambda}$
    $\mathbf{x} = \mathbf{x} + h\mathbf{W}\mathbf{J}^T \Delta\boldsymbol{\lambda}$
end for
```

Overall, when dealing with contacts, we found the method to show its strength over VTS only at large number of iterations (see Figure 10 and the Results section). As a fallback one could precompute the constraint error and direction for this kind of constraints and also make the algorithm faster. But the main advantage of the method is that it permits very tight two way coupling between cloth and colliding particles.

7. Further optimization

We believe that our improved Jacobi method can profit from even more optimizations, like for instance sleeping policies [BETC14]. We tried warm starting [Cat05] and found that it helps improve convergence, but not as dramatically as for GS. Our parallel implementation of the algorithm is a straightforward clone of Jacobi plus the improvement in (37). We used C++ and OpenCL for our implementation. The algorithm runs in two passes: a loop over all the constraints computing the required impulses and a loop over all the particles accumulating and applying these impulses. For the second pass we use adjacency lists for each particle. This approach permitted us to exploit multi-core architectures and simulate more particles in real time. Still we believe our method can be optimized even further. For example it could be used in a block solver like in [TBV12] where each block can be solved using GS in the local memory of a GPU compute unit.

8. Applications

8.1. Granular matter

Granular matter is a material with very curious properties as it sometimes behaves like a solid and other times like a fluid [JNB96]. For example, due to static friction piles of grains maintain their shape, but if the angle of repose is reached avalanches can be triggered. For similar reasons sand in an hourglass flows at a steady rate as the pressure doesn't increase indefinitely with height as in the case of hidrostatic pressure. We attempted to reproduce such behaviour with our methods and you can see the resulting sand piles in Figure 3. Notice how the angle of repose increases with the value of the friction coefficient.

Collision detection between particles is done at discrete times using a grid to prune the possible intersecting candidates, followed by sphere-sphere tests. We allow a small tolerance in our tests in order to catch potential changes in the constraint active set while the solver is running. The collision tolerance is more important for stabilizing PBD when used for particle collisions, even if increasing it means more constraints to solve per frame. This is again why we think SP is a better method as it needs lower tolerance, especially when we integrate the positions after at least a pure velocity iteration. Note also that in PBD collision detection is done after the unconstrained integration step and before the solver, while in VTS and SP it is done right before the solver as there are no position updates at that time.

We also implemented collisions with triangle meshes, either static or dynamic (see coupling with cloth below). For this we used the same grid to filter point-triangle candidates and then performed point-triangle or ray-triangle tests (in order to prevent tunneling). You can see in Figure 4 some simulations we did of particles falling over meshes and an hourglass. We chose the VTS simulation method here as it performs better and is more stable at low iteration count. Although all of our tests involved convex object shapes (i.e. spheres) we are confident that the methods described apply just as well for arbitrary concave bodies, as is the case of rigid bodies, which we plan to address in a future paper.

8.2. Cloth

Cloth is modeled many times as a particle system connected with stretching, shearing and bending springs [Pro96]. In our case we replace the springs with hard constraints that can be softened in the case of shearing and bending using a stiffness parameter like in [MHHR07]. Note that the constrained dynamics methods correspond in theory to infinitely stiff springs so there is no need for specifying the true elastic stiffness of the material. Behaviour corresponding to low stiffness, e.g. under-critically damped oscillations, can only arise because of numerical inaccuracy of the iterative solvers or due to the aforementioned stiffness factors.

We applied to this cloth model our improved Jacobi iterative method using PBD and the results were very good. We also used the SP method and the biggest gain was its very accurate friction model. You can see a comparison between PBD and SP for handling friction in Figure 5. For PBD we applied friction after the position solver using constant vertical push, while in SP we recomputed the friction impulse at every iteration using the updated normal impulse.

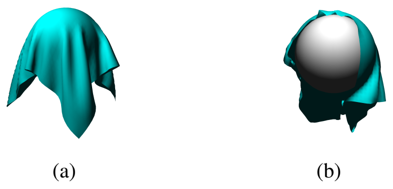


Figure 5: Cloth falling freely over a sphere with friction coefficient $\mu = 0.5$: (a) using the SP method the cloth remains stable on the sphere and (b) using the PBD method the cloth falls very quickly off the sphere.

Another advantage of SP is that it permits us to solve both cloth and particles with frictional contact in the same iterative method, allowing direct two way coupling. We present some results in Figure 6 where the cloth bounces the particles up several times until they reach a steady state together. We performed the same kind of mesh collision tests as described in the previous section, but contact response was different in order to allow coupling. We used barycentric coordinates [Jak01] and a modified impulse [BFA02] in order to correct the positions and velocities of the triangle vertices.

9. Results

First we tested the accuracy of our improved Jacobi method on bilateral constraints only. Our test scenario consisted of a 100x100 piece of cloth falling from a horizontal position and hanging by two corners. The simulation used a PBD method with a timestep of 16 ms, one substep and 15 iterations. You can see the evolution in time of the system for three different solvers in Figure 7. Clearly our method performs better.

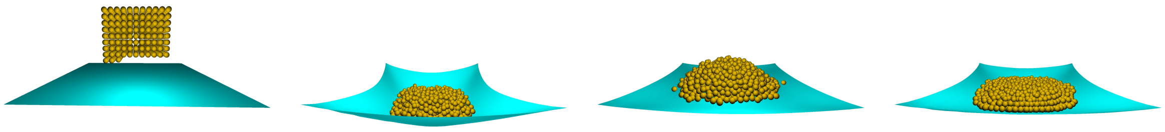


Figure 6: Simulation of 1000 particles falling on a 20x20 piece of cloth fixed at its corners (using the SP method).

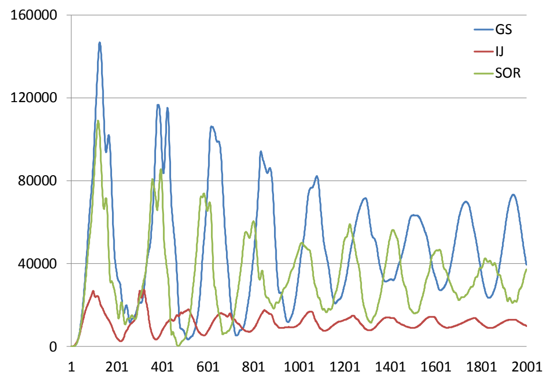


Figure 7: Plot of constraint error (L1 norm) for PBD cloth simulation with different solvers (frame number on the horizontal axis): Gauss-Seidel (blue), SOR (green) with $\omega = 1.2$, and improved Jacobi (red) with $\omega = 0.5$, $a = 1$ and $b = 0.6$.

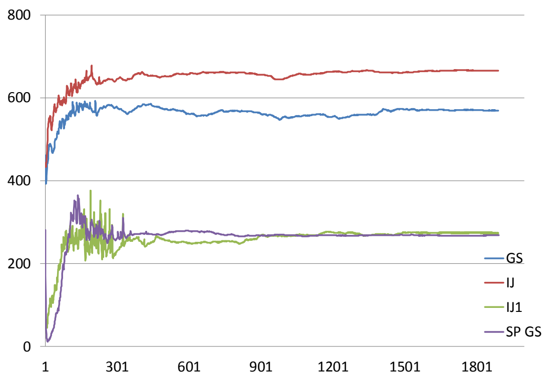


Figure 8: Plot of unilateral constraint error (L1 norm) for 3000 particles falling in a box (VTS, $\gamma = 0.5$): Gauss-Seidel (blue), improved Jacobi with $\omega = 0.5$, $a = 1$ and $b = 0.6$ (red), improved Jacobi with $a = 2$ (green), and Sequential Positions using GS with $k_c = 1$ and $k_v = 0.1$ (purple).

Next we measured the positional errors for unilateral frictionless constraints by dropping 3000 particles in a box using a VTS method. As you can see from the results (Figure 8), the improved Jacobi method is not always more accurate than GS and we had to tweak it ($a = 2$) in order to get better results. Still, even without tweaking, our method behaves similarly to GS and at a similar cost (see Table 1) without having its drawbacks: constraint order bias and batching for parallel implementations [Har09, TBV12].

	Gauss-Seidel	Improved Jacobi
Cloth	39 ms	51 ms
Particles	5.2 ms	6.5 ms

Table 1: Frame time measurements made on a Intel Core i7 3770 CPU (single-threaded) for the two presented scenarios: hanging cloth (PBD) and falling particles (VTS).

We also tested heterogeneous mass values with large ratios between them and found that improved Jacobi handles them just as robustly as GS. Adding friction to the mix may introduce jitter but it can be alleviated by lowering ω. You can see the results of a falling particles simulation with inverse mass values between 0.01 and 1000 and friction ($\mu = 0.2$) in Figure 9.

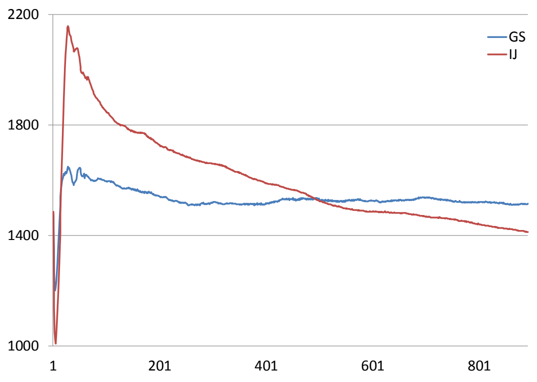


Figure 9: Plot of unilateral constraint error (L1 norm) for 3000 particles in a box with varying masses and friction: GS (blue), improved Jacobi (red) with $\omega = 0.4$, $a = 1$, $b = 0.6$.

In our falling particles experiments we used 15 iterations and one substep at 60 Hz, but the iteration count can be set even lower without breaking our method. For very low iteration numbers we recommend decreasing ω and b more in order to avoid jitter. Of course high velocities and small object sizes can put even GS in difficulty. In this situations

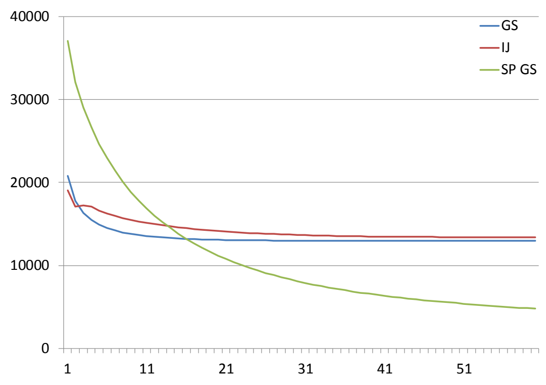


Figure 10: Plot of relative velocity along constraints relative to the number of iterations for 1000 particles falling in a box with contact and friction: Gauss-Seidel (blue), improved Jacobi (red) with $\omega = 0.5$, $a = 1$ and $b = 0.6$, and Sequential Positions using GS with $k_c = 1$ and $k_v = 0.1$ (green).

increasing the number of iterations doesn't always work and we need to lower the time step, i.e. add more substeps.

In order to measure convergence per frame for both VTS and SP methods we switched to using the velocity error. We present results from frame 100 (Figure 10) for 1000 falling particles with friction (60 iterations per frame). Again improved Jacobi behaves very similarly to GS. Also our SP method converges better than VTS, but only for a high enough number of iterations. You can see in Figure 8 that the positional error of the SP-GS method is also good but, for the visual aspect, impacts may require softening.

10. Conclusions

We have presented a new iterative method for solving constrained dynamics based on the Conjugate Residuals and Jacobi methods and showed that it has very good convergence properties, similar to Gauss-Seidel. In many cases, especially for bilateral constraints, our method provides more accurate solutions for a lower number of iterations than GS. We have also proposed a new solver scheme that we called Sequential Positions acting as a hybrid between position based dynamics and velocity time stepping methods. SP handles friction better, provides a mechanism for mixing between PBD and VTS and softens the impacts. It also allows direct two way coupling between cloth and rigid particles.

11. Future work

We would like to use more accurate modeling for rolling friction and focus more on rigid bodies. Another goal is to improve collision detection and include it together with our improved Jacobi solver in a unified GPU physics pipeline.

12. Acknowledgements

The work has been funded by the Sectoral Operational Programme Human Resources Development 2007-2013 of the Ministry of European Funds through the Financial Agreement POSDRU/159/1.5/S/132395.

References

[ABC*07] Ammann C., Bloom D., Cohen J. M., Courte J., Flores L., Hasegawa S., Kalaitzidis N., Tornberg T., Treweek L., Winter B., Yang C.: The birth of sandman. In *ACM SIGGRAPH 2007 Sketches* (New York, NY, USA, 2007), SIGGRAPH '07, ACM. 1

[AH04] Anitescu M., Hart G. D.: A Constraint-Stabilized Time-Stepping Approach for Rigid Multibody Dynamics with Joints, Contact and Friction. *International Journal for Numerical Methods in Engineering 60*, 14 (2004), 2335–2371. 1, 2, 5

[AO11] Alduán I., Otaduy M. A.: SPH Granular Flow with Friction and Cohesion. In *ACM SIGGRAPH 2011 Papers* (New York, NY, USA, 2011), SIGGRAPH '11, ACM. 1

[AT10] Anitescu M., Tasora A.: An Iterative Approach for Cone Complementarity Problems for Nonsmooth Dynamics. *Comput. Optim. Appl. 47*, 2 (Oct. 2010), 207–235. 1, 3, 5

[ATO09] Alduán I., Tena A., Otaduy M. A.: Simulation of high-resolution granular media. In *Proc. of Congreso Español de Informática Gráfica* (2009), vol. 1. 1

[Bar94] Baraff D.: Fast Contact Force Computation for Non-penetrating Rigid Bodies. In *ACM SIGGRAPH 1994 Papers* (New York, NY, USA, 1994), SIGGRAPH '94, ACM, pp. 23–34. 1

[BBD09] Bender J., Bayer D., Diziol R.: Dynamic simulation of inextensible cloth. *IADIS International Journal on Computer Science and Information Systems 4*, 2 (2009), 86–102. 1

[BETC14] Bender J., Erleben K., Trinkle J., Coumans E.: Interactive Simulation of Rigid Body Dynamics in Computer Graphics. *Computer Graphics Forum 33*, 1 (2014), 246–270. 1, 2, 5, 6

[BFA02] Bridson R., Fedkiw R., Anderson J.: Robust Treatment of Collisions, Contact and Friction for Cloth Animation. In *ACM SIGGRAPH 2002 Papers* (New York, NY, USA, 2002), SIGGRAPH '02, ACM, pp. 594–603. 7

[BMOT13] Bender J., Müller M., Otaduy M. A., Teschner M.: Position-based methods for the simulation of solid objects in computer graphics. In *EUROGRAPHICS 2013 State of the Art Reports* (2013), Eurographics Association. 1

[BW98] Baraff D., Witkin A.: Large Steps in Cloth Simulation. In *ACM SIGGRAPH 1998 Papers* (New York, NY, USA, 1998), SIGGRAPH '98, ACM, pp. 43–54. 1, 2

[BYM05] Bell N., Yu Y., Mucha P. J.: Particle-based Simulation of Granular Materials. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (New York, NY, USA, 2005), SCA '05, ACM, pp. 77–86. 1

[BZX14] Barbic J., Zhao Y., Xu H.: Implicit Multibody Penalty-based Distributed Contact. *IEEE Transactions on Visualization and Computer Graphics 99*, PrePrints (2014), 1. 1

[Cat05] Catto E.: Iterative Dynamics with Temporal Coherence. *Game Developer Conference* (Jan. 2005). 2, 3, 5, 6

[Cou12] Coumans E.: Destruction. In *Game Developers Conference Proceedings* (2012). 1

[CPS92] COTTLE R., PANG J., STONE R.: *The Linear Complementarity Problem*. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104), 1992. 2, 3

[DCB14] DEUL C., CHARRIER P., BENDER J.: Position-based rigid body dynamics. In *Proceedings of the 27th International Conference on Computer Animation and Social Agents* (May 2014). 2

[Erl07] ERLEBEN K.: Velocity-based Shock Propagation for Multibody Dynamics Animation. *ACM Trans. Graph. 26*, 2 (June 2007). 1, 2, 5

[GHF*07] GOLDENTHAL R., HARMON D., FATTAL R., BERCOVIER M., GRINSPUN E.: Efficient Simulation of Inextensible Cloth. In *ACM SIGGRAPH 2007 Papers* (New York, NY, USA, 2007), SIGGRAPH '07, ACM. 1, 3

[Har07] HARADA T.: Real-Time Rigid Body Simulation on GPUs. In *GPU Gems 3*, Nguyen H., (Ed.). Addison Wesley Professional, Aug. 2007, ch. 29. 1

[Har09] HARADA T.: Parallelizing the Physics Pipeline: Physics Simulations on the GPU. In *Game Developers Conference Proceedings* (2009). 8

[HATN12] HEYN T., ANITESCU M., TASORA A., NEGRUT D.: Using Krylov Subspace and Spectral Methods for Solving Complementarity Problems in Many-Body Contact Dynamics Simulation. *International Journal for Numerical Methods in Engineering* (2012). 2

[HLW06] HAIRER E., LUBICH C., WANNER G.: *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*. Springer Series in Computational Mathematics. Springer, 2006. 3

[How11] HOWES L.: Cloth Simulation in the Bullet physics SDK. In *OpenCL Programming Guide*, Munshi A., Gaster B. R., Mattson T. G., Func J., Ginsburg D., (Eds.). Addison Wesley, 2011, ch. 17, p. 425–448. 1

[HZ06] HAGER W. W., ZHANG H.: A survey of nonlinear conjugate gradient methods. *Pacific Journal of Optimization 2*, 1 (2006), 35–58. 4

[IWT12] IHMSEN M., WAHL A., TESCHNER M.: High-resolution Simulation of Granular Material with SPH. In *Workshop on Virtual Reality Interaction and Physical Simulation* (2012), The Eurographics Association, pp. 53–60. 1

[Jak01] JAKOBSEN T.: Advanced Character Physics. In *Game Developers Conference Proceedings* (2001), pp. 383–401. 1, 3, 5, 6, 7

[JNB96] JAEGER H. M., NAGEL S. R., BEHRINGER R. P.: Granular solids, liquids, and gases. *Rev. Mod. Phys. 68* (Oct 1996), 1259–1273. 7

[KSJP08] KAUFMAN D. M., SUEDA S., JAMES D. L., PAI D. K.: Staggered Projections for Frictional Contact in Multibody Systems. *ACM Transactions on Graphics (SIGGRAPH Asia 2008) 27*, 5 (2008), 164:1–164:11. 1, 5

[Lac03] LACOURSIÈRE C.: Splitting Methods for Dry Frictional Contact Problems in Rigid Multibody Systems: Preliminary Performance Results. In *The Annual SIGRAD Conference. Special Theme – Real-Time Simulations. Conference Proceedings from SIGRAD2003* (2003). 1, 3

[LSB10] LACOURSIÈRE C., SERVIN M., BACKMAN A.: Fast and stable simulation of granular matter and machines. 1

[MHHR07] MÜLLER M., HEIDELBERGER B., HENNIX M., RATCLIFF J.: Position Based Dynamics. *J. Vis. Comun. Image Represent. 18*, 2 (Apr. 2007), 109–118. 1, 3, 5, 6, 7

[MMCK14] MACKLIN M., MÜLLER M., CHENTANEZ N., KIM T.-Y.: Unified particle physics for real-time applications. *ACM Transactions on Graphics (TOG) 33*, 4 (2014), 104. 1, 2, 5

[Mor05] MORAVÁNSZKY A.: NovodeX Demo Exercise. 2

[NGL10] NARAIN R., GOLAS A., LIN M. C.: Free-flowing Granular Materials with Two-way Solid Coupling. In *ACM SIGGRAPH Asia 2010 Papers* (New York, NY, USA, 2010), SIGGRAPH ASIA '10, ACM, pp. 173:1–173:10. 1

[PNE10] POULSEN M., NIEBE S., ERLEBEN K.: Heuristic Convergence Rate Improvements of the Projected Gauss–Seidel Method for Frictional Contact Problems. Skala V., (Ed.), Václav Skala - UNION Agency, pp. 135–142. 3

[Pro96] PROVOT X.: Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behavior. In *In Graphics Interface* (1996), pp. 147–154. 1, 7

[RA05] RENOUF M., ALART P.: Conjugate gradient type algorithms for frictional multi-contact problems: applications to granular materials. *Computer Methods in Applied Mechanics and Engineering 194*, 18-20 (May 2005), 2019–2041. 1, 2

[Saa03] SAAD Y.: *Iterative Methods for Sparse Linear Systems*, 2nd ed. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2003. 2, 3, 4

[She94] SHEWCHUK J. R.: *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*. Tech. rep., Pittsburgh, PA, USA, 1994. 4

[SHNE10a] SILCOWITZ-HANSEN M., NIEBE S., ERLEBEN K.: A nonsmooth nonlinear conjugate gradient method for interactive contact force problems. *The Visual Computer 26*, 6-8 (2010), 893–901. 2

[SHNE10b] SILCOWITZ-HANSEN M., NIEBE S., ERLEBEN K.: Projected Gauss-Seidel Subspace Minimization Method for Interactive Rigid Body Dynamics - Improving Animation Quality using a Projected Gauss-Seidel Subspace Minimization Method. In *GRAPP* (2010), Richard P., Braz J., Hilton A., (Eds.), INSTICC Press, pp. 38–45. 2

[ST96] STEWART D., TRINKLE J. C.: An Implicit Time-Stepping Scheme for Rigid Body Dynamics with Coulomb Friction. *International Journal for Numerical Methods in Engineering 39* (1996), 2673–2691. 1, 2, 3, 5

[Sta09] STAM J.: Nucleus: Towards a unified dynamics solver for computer graphics. In *Computer-Aided Design and Computer Graphics, 2009. CAD/Graphics '09. 11th IEEE International Conference on* (Aug 2009), pp. 1–11. 2

[TA10] TASORA A., ANITESCU M.: A Convex Complementarity Approach for Simulating Large Granular Flows. *J. Comput. Nonlinear Dynam. 5* (05/2010 2010), 031004 (10 pages). 1

[TBV12] TONGE R., BENEVOLENSKI F., VOROSHILOV A.: Mass Splitting for Jitter-free Parallel Rigid Body Simulation. *ACM Trans. Graph. 31*, 4 (July 2012), 105:1–105:8. 1, 2, 7, 8

[Ton12] TONGE R.: Solving Rigid Body Contacts. *Game Developer Conference* (2012). 2, 5

[TSIHK06] TANAKA M., SAKAI M., ISHIKAWAJIMA-HARIMA, KOSHIZUKA S.: Rigid Body Simulation Using a Particle Method. In *ACM SIGGRAPH 2006 Research Posters* (New York, NY, USA, 2006), SIGGRAPH '06, ACM. 1

[ZB05] ZHU Y., BRIDSON R.: Animating Sand As a Fluid. In *ACM SIGGRAPH 2005 Papers* (New York, NY, USA, 2005), SIGGRAPH '05, ACM, pp. 965–972. 1