The 4th Canadian Conference on Nonlinear Solid Mechanics (CanCNSM 2013)
McGill University, July 23-26, 2013
Montréal, Canada

Paper ID 814

Standard Interface for data analysis of solvers in multibody dynamics

Abstract
After more than twenty years spent on the development of simulation models and integration algorithms, there is no convenient way to compare the performance of the many algorithms that solve the Complementarity Problem (CP) in an unbiased manner. The constraint models in multibody systems naturally take the form of CPs and the current methods for the formulations lead to singular CPs which are difficult to solve. Since the underlying models are nonsmooth and nonlinear, mathematical theory supporting the design of simulation methods is sparse. For these reasons, we find that an important tool in the field of physical simulation will be a benchmark database that facilitates the fair comparison of solver algorithms. In any field where theory is limited, sound empirical methods must be used to measure progress and set future research directions. We present an HDF5 database, which provides a standard interface for the capture of data needed to reconstruct the time-stepping subproblem from any open-source physics engine *e.g.* Bullet, ODE, Gazebo, ChronoEngine *etc.* Timing data, as well as unilateral and bilateral constraint information is collected at the time-step level. This dataset may then be accessed later to reformulate the subproblem using different dynamics models. Applying different solvers to the CPs provides a comparison of solver performance and error analysis. This standard software interface provides a convenient and fair way to compare the solvers in the different physics engines as well as customized algorithms for multibody systems.

Keywords
solver, multibody dynamics, error, comparison, Complementarity Problem

Ying Lu, RPI Robotics Lab, Rensselaer Polytechnic Institute, luy4@rpi.edu
Jedediyah Williams, RPI Robotics Lab, Rensselaer Polytechnic Institute, willij16@cs.rpi.edu
Claude Lacoursière, HPC2N/UMIT, Umeå University, claude@hpc2n.umu.se
Jeff Trinkle,RPI Robotics Lab, Rensselaer Polytechnic Institute, trinkle@gmail.com

1. Introduction

The dynamics of the multibody system is nonsmooth and nonlinear because of the discontinuous nature of nonpenetration and stick-slip friction constraints. There are linearized models to approximate the nonlinear constraints, which introduce more constraint equations than the nonlinear formulations, making the simulation slow for large number of contacts. Currently, solution methods in popular physics engines like Bullet, ODE, Gazebo, Solfec, *etc*, fall into two categories: one involves solving the dynamics directly, and the other involves solving a linearized approximation of the model. Despite the importance of solver choice, little has been done to analyze their differences, particularly with respect to solution of data from "real" simulation. In this paper, we provide an standard interface to make such comparisons using data collected from simulation experiments in any physics engine. In doing so, we hope to provide insight into simulator design choices, including better understanding of the tradeoff between speed and accuracy for various purposes of simulation.

In the following sections, we first introduce the Complementarity Problem from the constraints, then the standard HDF5 interface is presented. The results of comparison between different time-stepping formulations and solvers are analyzed in the last section.

2. Previous Work

The methods to solve the Complementarity Problems fall into two main categories: direct method and iterative method [1] [2]. The methods of Lemke and Dantzig are direct methods based on simplex pivoting [2], which are faster and more accurate when solving systems with few contacts. But when it comes to large system with hundreds of contacts, they have a tendency to fail. Iterative methods include Projected Gauss-Seidel (PGS), projected Jacobi, fixed-point [3], and generalized Newton methods. PGS has been popular as a parallel solver, but because of the zeros in the diagonal entries of the matrix that characterizes the system's dynamics, the blocked PGS has been more popular.

For the time-stepping subproblem, the mixed linear Complementarity Problem (MLCP) and Linear Complementarity Problem (LCP) are popular and available in most of physics engines. Lemke and PATH [4] have been widely used in solving the LCPs and MLCPs, respectively. The other time-stepping subproblem formulation is the Nonlinar Complementarity Problem (NCP), which fully models the system's nonlinar and nonsmooth nature [5]. A popular method to solve the NCP problems is to rewrite the NCP constraint equations into an equivalent function such as the Fischer-Burmeister NCP function [6], or Alart-Curnier function [7]. Chen et al. [8] have modified the Fischer-Burmeister function by adding a new term to the original function to improve performance. After we write the NCP functions, we may solve the nonlinear system of equations with a generalized Newton method or convert it to an optimization problem [3].

3. Complementarity Problems

In this section, the complementarity condition will be built based on the physical constraints which occur in the simulation of multibody dynamics.

3.1 Normal Constraints

Since the normal constraints differ for unilateral contacts and bilateral joints, we present the following two cases:

Unilateral Contacts

An example of potential contact and the contact frame between two spheres is depicted in figure 1. The two bodies are in contact when the gap distance ψ_n is 0. An impulse is generated by the collision between two bodies if and only if the gap distance is 0. Because we consider the bodies ideally rigid, the gap

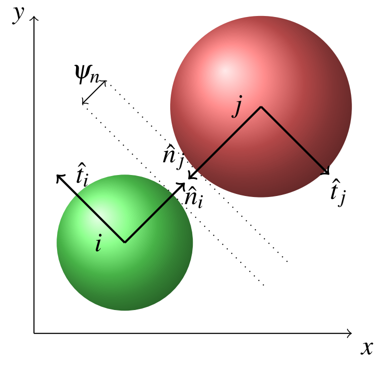

Figure 1. contact frame

distance should always be non-negative. Based on this, the forces between the two bodies can only be repulsive forces to push them apart when the distance between the two bodies is 0. To write the constraints in a complementarity form:

$$\mathbf{0} \leq \psi_n(\mathbf{q},t) \perp \lambda_n \geq \mathbf{0} \tag{1}$$

where ψ_n is the gap distance between the two bodies forming the contact, and is a function of the generalized position $\mathbf{q}$ and time t. The normal force λ_n is applied to each body in opposite directions to prevent penetration. The $\perp$ implies orthogonality, which states that a contact force can only exist when the distance between the two bodies is zero.

Bilateral Joints

For a bilateral joint such as a revolute or prismatic joint, the contact points are constrained to coincide, which removes some degrees of freedom from the system. The normal constraint equation for a bilateral joint may be written:

$$\phi_n(\mathbf{q},t) = \mathbf{0} \tag{2}$$

3.2 Frictional constraints

Coulomb's dry friction law is used to build the friction model. For a three-dimensional multibody system, we use $[\hat{n},\hat{t},\hat{o}]$ to represent the primary axes of the contact frame, where the normal contact force is λ_n and the tangential friction force is: $[\lambda_t,\lambda_o]$. This defines a friction cone at the contact frame. There are three physical cases for a potential contact at the friction level.

- **Detaching**: The gap distance between the two bodies is 0. The normal force is 0, and the frictional force at this contact will be $[\lambda_t,\lambda_o] = [0,0]$. The relative tangential velocity can be any arbitrary value.
- **Sliding**: The gap distance between the two bodies is 0, the frictional force satisfies $\lambda_t^2+\lambda_o^2 = \mu^2\lambda_n^2$, where μ is the coefficient of friction. The relative sliding velocity in the tangential direction at the contact point has the opposite direction with the frictional force and the magnitude is greater than 0.
- **Sticking** The gap distance between the two bodies is 0, the frictional force satisfies $\lambda_t^2+\lambda_o^2 < \mu^2\lambda_n^2$. The magnitude of relative tangential velocity at the contact degenerates to 0.

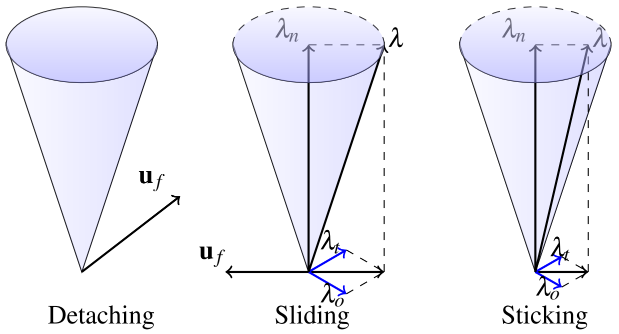

Figure 2. Three friction cases

In figure 2, v_f is the relative velocity at the contact point. Here we introduce some notation that will be used in the following sections. The generalized position and velocity in the world frame are $\mathbf{q}$ and $v = \dot{\mathbf{q}}$. $\mathbf{G}_n$ and $\mathbf{G}_f$ are the transformation matrices dependent on position $\mathbf{q}$ and map the normal and frictional forces from the contact frame to the world frame. Conversely, $\mathbf{G}_n^T$ and $\mathbf{G}_f^T$ will transform the velocity from the world frame to the contact frame to get the relative velocities. The normal and tangential relative velocity are denoted as $\mathbf{u}_n = \mathbf{G}_n^T v$ and $\mathbf{u}_f = \mathbf{G}_f^T v$.

Friction laws here satisfy the maximum dissipation principle at each contact. For the ith contact:

$$(\lambda_{it},\lambda_{io}) \in \underset{(\lambda'_{it},\lambda'_{io})\in\mathscr{F}_i}{\arg\max} \left(-(u_{it}\lambda'_{it}+u_{io}\lambda'_{io})\right) \tag{3}$$

where $\mathscr{F}_i$ is the Coulomb friction cone at the ith contact point, and $\mathscr{F}_i = \left\{(\lambda_{it},\lambda_{io}) : (\lambda_{it})^2 + (\lambda_{io})^2 \leq (\mu_i\lambda_{in})^2\right\}$. The maximum dissipation principle here can not be determined from the Karush-Kuhn-Tucker (KKT) condition, so we apply the Fritz-John conditions and arrive at the compact friction formulation [9]

$$(\mathbf{U}\lambda_n)\circ\mathbf{u}_t + \lambda_t \circ \mathbf{s} = \mathbf{0} \tag{4}$$

$$(\mathbf{U}\lambda_n)\circ\mathbf{u}_o + \lambda_o \circ \mathbf{s} = \mathbf{0} \tag{5}$$

$$\mathbf{0} \leq (\mathbf{U}\lambda_n)\circ(\mathbf{U}\lambda_n) - \lambda_t\circ\lambda_t - \lambda_o\circ\lambda_o \perp \mathbf{s} \geq \mathbf{0} \tag{6}$$

where $\mathbf{s}$ is the vector of sliding speed at the contact point, and $\mathbf{U}$ is a diagonal matrix of the coefficients of friction. The operator $\circ$ denotes the Hadamard product:

$$\mathbf{u}\circ\mathbf{v} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} \circ \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} = \begin{bmatrix} u_1v_1 \\ u_2v_2 \\ \vdots \\ u_nv_n \end{bmatrix} \tag{7}$$

3.3 Discretized friction model

Since the friction constraints are nonlinear, LCP solvers like Lemke and PATH cannot solve this model directly. An approximation to the quadratic friction cone in the Coulomb friction model is a polyhedron with n_d facets in the tangent plane that discretize the nonlinear model. The friction inside or on the boundary of the friction cone defined as $\mathscr{F}$ will be approximated as the sum of the n_d components. The linearized form of the maximum dissipation conditions are:

$$\alpha_i \in \underset{\mathbf{0}\leq\alpha_i,\mathbf{e}^T\alpha_i\leq\mu_i\lambda_{in}}{\arg\max} \left(v^T\mathbf{D}^T\mathbf{G}_f^T\alpha_i\right) \tag{8}$$

where α_i is a n_d component vector where each non-negative component represents a magnitude of the friction force in one of the n_d discrete friction directions. The sum of the n_d components is no larger than $\mu_i\lambda_{in}$. The vector $\mathbf{e}$ is a column vector with all 1s and length n_d. $\mathbf{D} = [\mathbf{d}_1,\mathbf{d}_2,\cdots,\mathbf{d}_{nd}]$, where

$$\mathbf{d}_i = \begin{bmatrix} \cos\left((i-1)\dfrac{2\pi}{n_d}\right) \\ \sin\left((i-1)\dfrac{2\pi}{n_d}\right) \end{bmatrix} \tag{9}$$

To keep consistent with the NCP model, we denote $\mathbf{G}_d = \mathbf{G}_f\mathbf{D}$. Using this linearized model, the frictional constraints are:

$$\mathbf{0} \leq \alpha \perp \mathbf{G}_d^T v + \mathbf{E}\mathbf{s} + \frac{\partial\psi_f}{\partial t} \geq \mathbf{0} \tag{10}$$

$$\mathbf{0} \leq \mathbf{s} \perp \underbrace{\mathbf{U}\lambda_n - \mathbf{E}^T\alpha}_{\sigma} \geq \mathbf{0} \tag{11}$$

where $\mathbf{E}$ is block diagonal matrix of elements $\mathbf{e}$.

4. Time-Stepping Formulations

4.1 Newton-Euler Equation

The Newton-Euler Equation for the system is:

$$\mathbf{M}(\mathbf{q})\dot{v} = \mathbf{G}_n\lambda_n + \mathbf{G}_f\lambda_f + \lambda_{\text{app}} \tag{12}$$

where $\mathbf{M}$ is the generalized mass matrix with mass properties for all the bodies. For the jth rigid body we have:

$$\mathbf{M}_j = \begin{bmatrix} m_j\mathbf{I}_{3\times3} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_j \end{bmatrix} \tag{13}$$

$\mathbf{I}_j$ is the inertia matrix for the jth body and λ_{app} is the applied force i.e. gravity.

Since $\dot{v} \approx (v^{\ell+1} - v^{\ell})/h$ and the impulse $\mathbf{p} = \lambda h$, we apply a backward Euler derivative formula to equation (12) to get the following time-stepping model:

$$\mathbf{M}(v_{\ell+1} - v_{\ell}) = \mathbf{G}_n^{\ell+1}\mathbf{p}_n^{\ell+1} + \mathbf{G}_f^{\ell+1}\mathbf{p}_f^{\ell+1} + \mathbf{p}_{\text{app}}^{\ell} \tag{14}$$

4.2 NCP Formulation

Equations (1), (4), (5), (6), (14) form the system of equations for the NCP formulation. One general method to solve the NCP is to write the constraints into so-called NCP-functions, such as the proximal (prox) function [10] or Fischer-Newton function. Another method is to write them as an optimization problem with an merit or objective function.

In this paper, The NCP constraints are written into an equivalent prox function [11], and then solved using a fixed-point variant of Gauss-Seidel.

A proximal point function maps a point to the closest point in a feasible set $\mathscr{C}$. The NCP condition in vector form $\mathbf{0} \leq \mathbf{x} \perp \mathbf{y}(\mathbf{x}) \geq \mathbf{0}$ can be rewritten as the following nonsmooth equation:

$$\phi(\mathbf{x},\mathbf{y}) = \mathbf{x} - \text{prox}_{\mathscr{C}}(\mathbf{x} - r\mathbf{y}(\mathbf{x})) = 0 \tag{15}$$

where r is a parameter that affects the convergence of the problem. For relatively large r, the problem is prone to diverge. Inside the range of r which guarantees the convergence of the problem, the larger r will converge faster while the smaller ones will converge slower. There are some strategies mentioned in [12] for choosing r effectively, but for larger problems it is still an area that needs further exploration. For the prox function, when the term $(\mathbf{x} - r\mathbf{y}(\mathbf{x}))$ is inside the feasible set, then $\text{prox}_{\mathscr{C}}(\mathbf{x} - r\mathbf{y}(\mathbf{x})) = \mathbf{x} - r\mathbf{y}(\mathbf{x})$ while when the term $(\mathbf{x} - \mathbf{r}\mathbf{y}(\mathbf{x}))$ falls outside the feasible set, it will be projected onto the nearest point on the boundary of the set.

Another popular NCP-function is Fischer-Burmeister function: Suppose we have the NCP condition in the vector form $\mathbf{0} \leq \mathbf{x} \perp \mathbf{y}(\mathbf{x}) \geq \mathbf{0}$, the Fischer-Burmeister function is

$$\phi_{i\text{FB}}(x_i,y_i) := x_i + y_i - \sqrt{x_i^2 + y_i^2}. \tag{16}$$

This function has nice properties since it is smooth everywhere except at the point $(0,0)$. However, the Fischer-Burmeister still has some drawbacks, for example it is too flat in the positive orthant, which is the region we are most interested in for Complementarity problems. A modified NCP-function is introduced by Chen et al [8]:

$$\phi_{i\lambda}(x_i,y_i) = \lambda\phi_{i\text{FB}}(x_i,y_i) + (1-\lambda)x_i^+y_i^+ \tag{17}$$

where $\lambda \in (0,1)$ is an arbitrary parameter and the term $x_i^+y_i^+$ not only penalizes violations of the complementarity condition but also makes the function continuously differentiable on $\mathbb{R}^2$

4.3 LCP Formulation

We substitute the linearized friction model into equation (14) to get the following:

$$\mathbf{M}(\nu_{\ell+1}-\nu_\ell)=\mathbf{G}_n^{\ell+1}\mathbf{p}_n^{\ell+1}+\mathbf{G}_d^{\ell+1}\alpha^{\ell+1}+\mathbf{p}_{\text{app}}^{\ell} \tag{18}$$

Together with equations (1), (10), and (11), (18) defines an MLCP formulation. We can solve equation (18) and substitute the velocity into the other three equations to get the LCP form:

$$\mathbf{0}\le\underbrace{\begin{bmatrix}\mathbf{G}_n^T\mathbf{M}^{-1}\mathbf{G}_n & \mathbf{G}_n^T\mathbf{M}^{-1}\mathbf{G}_d & \mathbf{0}\\ \mathbf{G}_d^T\mathbf{M}^{-1}\mathbf{G}_n & \mathbf{G}_d^T\mathbf{M}^{-1}\mathbf{G}_d & \mathbf{E}\\ \mathbf{U} & -\mathbf{E}^T & \mathbf{0}\end{bmatrix}}_{\mathbf{A}}\underbrace{\begin{bmatrix}\mathbf{p}_n\\ \mathbf{p}_f\\ \mathbf{s}\end{bmatrix}}_{\mathbf{z}}$$

$$+\underbrace{\begin{bmatrix}\mathbf{G}_n^T(\nu+\mathbf{M}^{-1}\mathbf{p}_{\text{app}})+\frac{\psi_n}{h}+\frac{\partial\psi_n}{\partial t}\\ \mathbf{G}_d^T(\nu+\mathbf{M}^{-1}\mathbf{p}_{\text{app}})+\frac{\partial\psi_f}{\partial t}\\ \mathbf{0}\end{bmatrix}}_{\mathbf{b}}\perp\underbrace{\begin{bmatrix}\mathbf{p}_n\\ \mathbf{p}_f\\ \mathbf{s}\end{bmatrix}}_{\mathbf{z}}\ge\mathbf{0} \tag{19}$$

Here we arrived at the standard LCP form which is $\mathbf{0}\le(\mathbf{Az}+\mathbf{b})\perp\mathbf{z}\ge\mathbf{0}$, allowing us to call any LCP solver to solve for $\mathbf{z}$.

5. Data Hierarchy

In order to compare different solver performance in solving the CPs, we have developed a data format that utilizes the Hierarchical Data Format (HDF5). Timing data, body information, and constraint violations at the simulation time-step level are collected. We capture all the data needed to reconstruct the time-stepping subproblem from any open-source physics engine and save it into our standardized dataset. By loading data later, both NCP and LCP formulations can be reconstructed and different solvers may be used to solve each problem again in order to do performance comparison and error analysis.

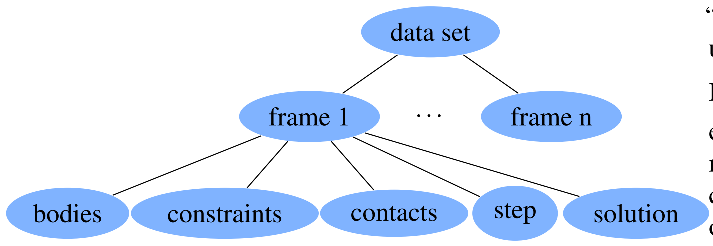

Figure 3. Simulation data hierarchy

A high-level view of the data hierarchy is shown in figure 3, where the frames 1 to n represent the n simulation steps, each including the results of collision detection, the original solver's solution, constraint forces, velocities, positions, *etc* as independent modules. The class is defined in more detail in table 1 with each column containing the lower level properties of figure 3. In the "body" class, the external forces like gravity are saved under "forces". The body id is unique for each body and used to index the bodies for reference. The inertia tensors for all bodies are saved in the field "inertia". Quaternions are stored to represent the body orientations.

In the "constraints" class, the information related to the bilateral joints are stored. The Jacobian matrix which transforms from the joint frame to the world frame is saved in the filed of "jacobian". For different kinds of joints, the size of Jacobian matrix varies, so we use a "row" vector to record the size of each Jacobian component for each joint. For some bilateral joints, there may be a joint limit, which is stored in the field "bounds". "Pairs" is the pair of body ids of the two bodies that form the joint.

Table 1. Detail of the HDF5 standard interface

bodies	constraints	contacts	solution
forces	jacobian	gap	z
ids	bounds	mu	total_error
inertia	pairs	normals	normal_error
masses	rows	pairs	friction_error
positions	violation	points	iterations
quaternions			slide_or_stick
velocities			

For each contact, the signed gap distance between the two bodies is saved in the field of "gap". The coefficient of friction is saved in the field of "mu". For unilateral contacts, we may reconstruct the Jacobian matrix which transforms from the contact frame to the world frame, so the fields of "normals" and "points" are saved. The normals are the unit vector from the contact point and is perpendicular to and away from the surface of the 1^{st} body in the pair. The points are represented by vectors from the center of mass of the 1^{st} body to the contact point on the body. The "step" is a field containing the single value of time step size in seconds.

After solving the complementarity problems, the error and iteration information are saved in the field of "solution". Here "z" is the solution to the equation (19). We measure the total error using a uniform standard objective function based on the Fischer-Burmeister function in equation (16): $\xi=\frac{1}{2}\phi_{\text{FB}}^T\phi_{\text{FB}}$. The total error is measured by evaluation of the objective function. The normal error is the violation of the normal complementarity condition while the frictional error is evaluated by the sum of magnitude error and the directional error. The magnitude error is $\mathbf{s}^T\sigma$, which is the frictional complementarity condition violation in equation (11). The directional error is measured by taking the dot product of the corresponding frictional force and the relative velocity, which is $\lambda_f^T\nu_f$. The number of iterations is also saved in this field. In the case of pivoting method, the number of pivots is saved. To help analyse the simulation and state change between the different friction cases: we also saved the "slide_or_stick", which stores whether the contact is sliding or sticking at the current frame.

After we get the data in the standard format, we load the data using our HDF5 reader into the RPI-MATLAB-Simulator,

Table 2. Available dynamics formulations with their compatible solvers.

Dynamics Model	Solvers			
LCP	Lemke [2]	Fischer-Newton [13]	PGS	Minmap-Newton [13]
mLCP	PATH [4]	Fixed-point	Minres [14]	Jacobi
NCP	PGS	Fixed-point [5]	Minres [14]	
mNCP	Fixed-point	non-smooth Newton [13]		

which is a physics engine with several dynamics formulations and solvers available (table 2). We reconstruct the time-stepping subproblem as an LCP or NCP, then pass the problem to each solver and record performance.

6. Result and Analysis

6.1 Error analysis of data from Gazebo

Using the data hierarchy described in section 5, we generated a dataset with problems simulated in Gazebo, which calls the solver in ODE. We developed a simulation with the Atlas robot [15] and a hose lying on the table. Figure 4 shows a frame in which the hand has grasped the end of the hose. The results of every time step in the simulation were saved in the HDF5 file and then the two most complex frames (measured by the number of unilateral contacts) were imported into MATLAB for testing.

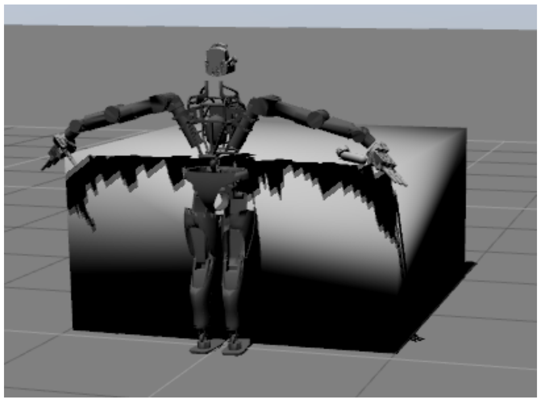

Figure 4. The Atlas robot picking up the hose

We iterated over the two frames for 40 iterations of PGS. Figure 5 shows the absolute value of the constraint violations (interbody penetrations, friction model violations, *etc.*) across the simulation. The errors after 40 iterations are mostly bounded by 10^{-3} and 10^{-4}. This is accurate enough for the simulation to appear correct with no obvious interpenetrations and stay stable for a long enough time.

To explore the effectiveness of additional iterations, we studied the error distributions of the 10th and 40th iterates of the PGS solver at each time step across the entire simulation. We could see from figure 6 that the center of mass of the error distribution for the 10th iterates is to the right of the distribution of the 40th iterates.

With this observation, we could set the maximum iteration number as 40, with which the error is mostly bounded by at most 10^{-3}. Then keep track of the solution with minimum violations as the solution for this simulation step. If we set the maximum number of iterations much bigger than 40, it will take much longer time to do the iterations with only a minor reduce in the error of violation.

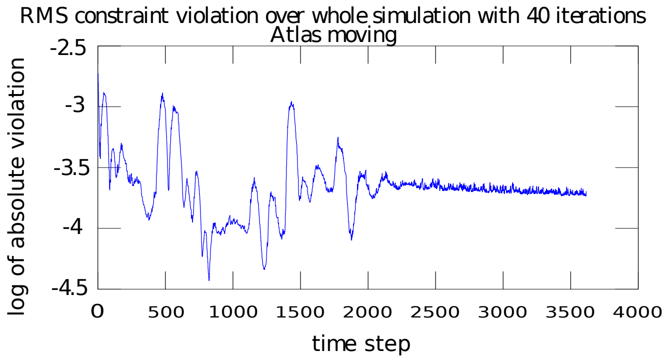

Figure 5. RMS constraint violation in absolute value with 40 iterations

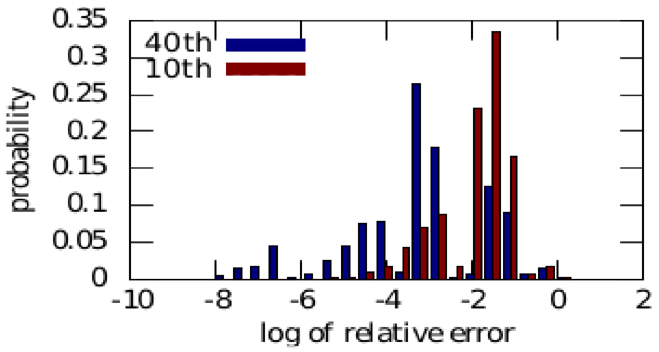

Figure 6. Average error distribution at the 10th and 40th iteartion when Atlas moving

6.2 Error analysis of data using RPI-MATLAB-Simulator

We ran an experiment of 15 spheres falling into a box from separate random initial positions. The box is defined by five faces: one bottom and four sides. Figure 7 shows a step when all the spheres have fallen onto the bottom and some of them are rolling on the bottom of the box. We solved 10 continuous timesteps using various solution methods and plot the convergence of the solvers.

Figure 8 shows the results from the Lemke solver, which is a direct method based on pivoting. The top plot is the average error after each pivot over all the 10 timesteps. We can see that the

Lemke solver has a high accuracy since the error is on the order of 10^{-10}. However, it is worth pointing out that the frictional error is on the order of 10^{-5}, which is even larger than the total error. This is possible due to the linearization of the friction cone. When we evaluate the total error by using the objective function, the discretized frictional forces are considered, but when we compute the frictional force, it is the resultant force along all the discretized directions.

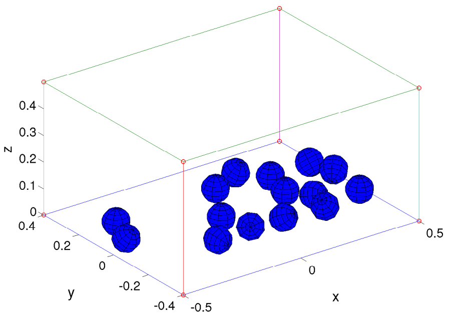

Figure 7. A scene of 15 spheres falling into a box

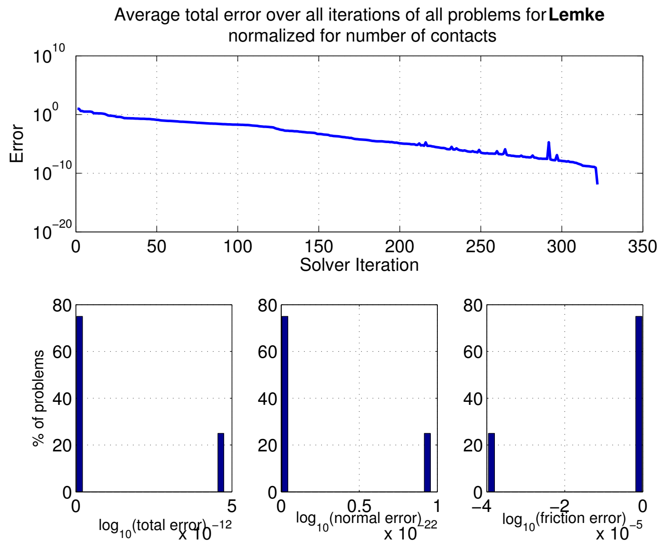

Figure 8. Lemke: performance with the detailed total, normal, frictional error

Figure 9 is the Projected Gauss Seidel method with the constrained force mixing (CFM) metric to make up for the zero element in the diagonal position of the matrix by adding a small number ε to the diagonal of the big matrix **A** in equation (19). The PGS method converged to the order of 10^{-1}, and afterwards, the error doesn't change with the increase of the solver iterations. This can be explained in that we find a local minimum solution. This trend is often seen in the PGS-type solver, which is one reason for including a maximum number of iterations such as in ODE.

6.3 Error analysis of data from Algoryx Simulation

This experiment is simulation on the stacking of forty identical cylindrical logs. These logs fall from separated initial positions and eventually pile up. Figure 10 shows a scene of the logs in the process of piling up.

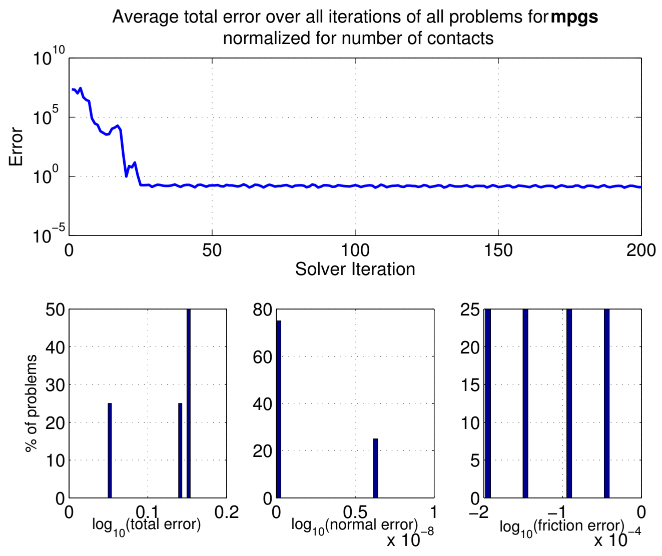

Figure 9. PGS: solver convergence with total, normal, frictional error

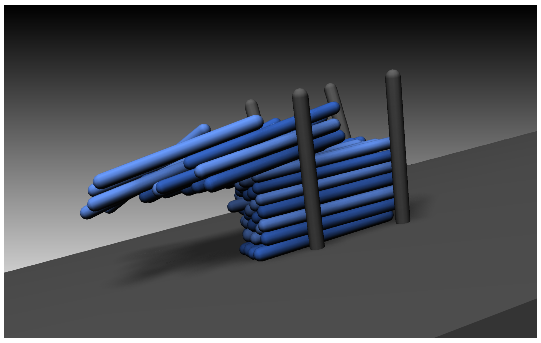

Figure 10. A scene of 40 cylindrical logs piling up

Figure 11 is the performance of Lemke solver on the Algoryx simulation. For the first 90 pivots, the complementarity conditions are not satisfied in the system of equations, until after about 92th pivots, the complementarity conditions are all satisfied and we arrived at the solution, then the error is in the order of 10^{-20}, which is effectively zero with machine precision. But like we saw before in the falling spheres example, the frictional error is still relatively large compared with the total error.

The Fischer-Newton method involves rewriting the Complementarity condition in the form of Fischer-Burmeister function in equation (16), and then solving the nonlinear system of equations using the generalized Newton's method. The performance of Fischer-Newton method is shown in figure 12. We didn't see the quadratic convergence for the Newton method, but the solution converged to the order of 10^{-5} in about 40 iterations. From the bottom detailed error bar plot, the normal error is still more accurate than the friction due to the approximation by linearization. The main bottleneck with minimizing error lies in determining how to solve the frictional force more effectively and accurately.

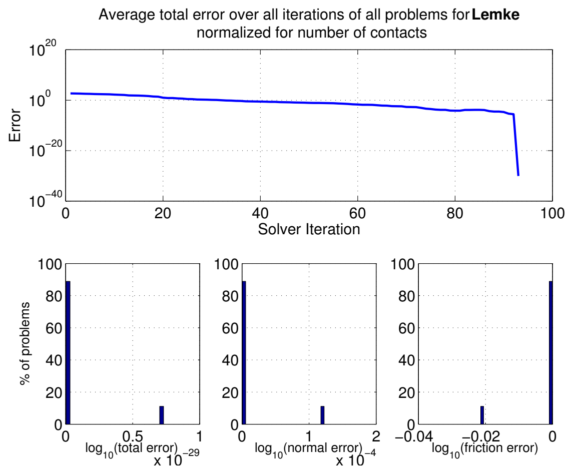

Figure 11. Lemke:performance with the detailed total, normal, frictional error

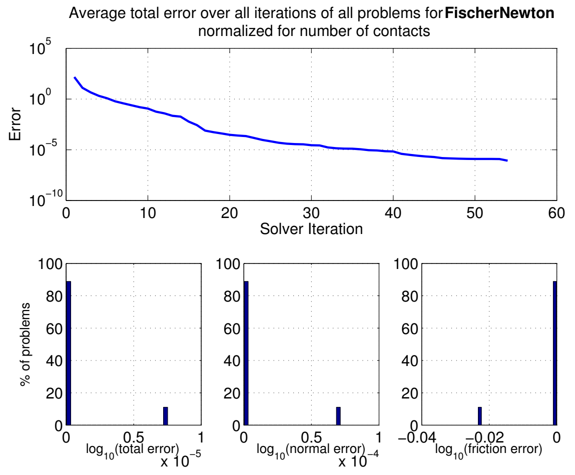

Figure 12. Fischer-Newton method: solver convergence and total, normal, frictional error

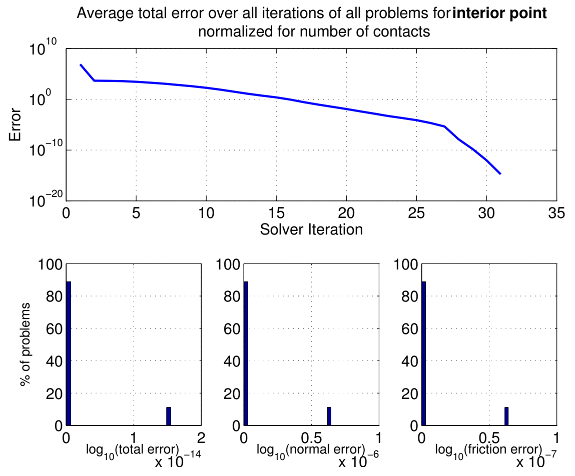

Figure 13. Interior-point method: solver convergence and total, normal, frictional error

The interior-point method introduces a positive value γ to ensure invariance of the complementarity conditions. So we have the system of equations:

$$(\mathbf{Ax}-\mathbf{y}+\mathbf{b})_i = 0 \tag{20}$$

$$\mathbf{x}_i\mathbf{y}_i-\gamma = 0 \tag{21}$$

$$\mathbf{x}_i \geq 0;\ \ \mathbf{y}_i \geq 0 \tag{22}$$

where γ is a relaxed complimentarity measure [13]. Figure 13 shows the error drops to the order of 10^{-15} in about 30 iterations. The frictional error is on the order of 10^{-7}, which is also accurate enough in the simulation.

Figure 14 is the method to rewrite the NCP formulation in the form of proximal function (15), in this method, the friction is solved as the frictional cone rather than using the linearized approximation. Before solving the system of nonlinear nonsmooth equations, we use quadratic programming to solve the normal force first, and then use the solution as the initial value to start the Gauss-Seidel-type iterative procedure. From the detailed friction error on the bottom of figure 12, the friction error is on the order of 10^{-5}, which is more accurate than the LCP model.

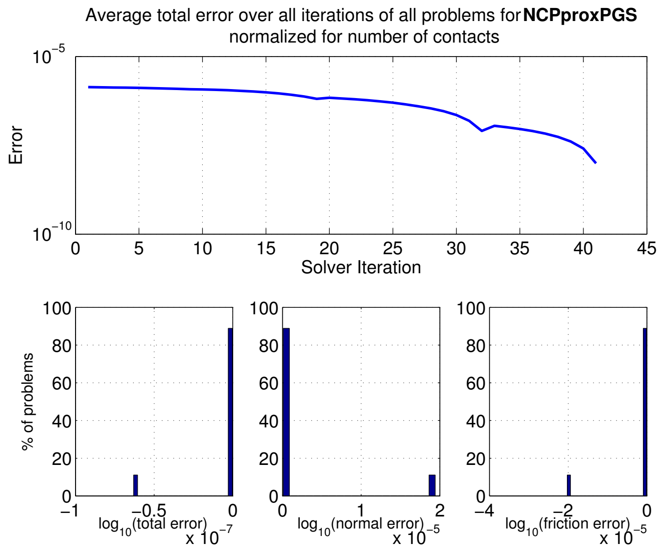

Figure 14. NCPprox method: solver convergence and total, normal, frictional error

7. Future Work

We currently have only one NCP solver, based on the proximal function and solved in a Gauss-Seidel manner. More solvers based on the NCP formulation will be added to our simulator. We had implemented a write-interface for ODE in order to store simulation data, and hope to extend the number of supported open-source physics engines. This will make direct comparison of solver performance possible in different physics engines.

Acknowledgments

This work was supported by DARPA W15P7T-12-1-0002 and Lockheed-Martin. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding agencies.

References

[1] J. Bender, K. Erleben, J. Trinkle, and E. Coumans. Interactive simulation of rigid body dynamics in computer graphics. Technical report, Eurographics Association, 2012.

[2] R. Cottle, J. Pang, and R. E. Stone. *The linear complementarity problem*. SIAM, 2004.

[3] T. Heyn, A. Tasora, M. Anitescu, H. Mazhar, and D. Negrut. A parallel algorithm for solving complex multibody problems with stream processors, 2009.

[4] M. C. Ferris and T. S. Munson. Complementarity problems in gams and the path solver. *Journal of Economic Dynamics and Control*, 24:2000, 1998.

[5] F. Bertails-Descoubes, F. Cadoux, G. Daviet, and V. Acary. A nonsmooth newton solver for caputuring exact coulomb friction in fiber assemblies. *ACM Transactions on Graphics*, January 2011.

[6] A. Fischer. A newton-type method for positive-semidefinite linear complementarity problem. *Journal of Optimization Theory and Applications*, 86:585–608, 1995.

[7] P. Alart and A. Curnier. A mixed formulation for frictional contact problems prone to newton like solution methods. *Comput. Methods Appl. Mech. Eng.*, 92(3):353–375, Nov 1991.

[8] B. Chen, X. Chen, and C. Kanzow. A penalized fischer-burmeister ncp-function: Theoretical investigation and numerical results, 1997.

[9] B. Nguyen. *Locally Non-convex Contacts Models and Solution Methods for Accurate Physical Simulation in Robotics*. PhD thesis, Rensselaer Polytechnic Institute, 2011.

[10] R. I. Leine and H. Nijmeijer. *Dynamics and Bifurcations of Non-Smooth Mechanical Systems*. Academic Press, 1992.

[11] T. Schindler, B. Nguyen, and J. Trinkle. Understanding the difference between prox and complementarity formulations for simulation of systems with contact. *IROS, IEEE*, pages 1433–1438, 2011.

[12] M. Forg, T. Geier, L. Neumann, and H. Ulbrich. R-factor strategies for the augmented lagrangian approach in multibody contact mechanics, June 2006.

[13] K. Erleben. num4lcp. Published online at code.google.com/p/num4lcp/, October 2011. Open source project for numerical methods for linear complementarity problems in physics-based animation.

[14] T. Heyn, M. Anitescu, A. Tasora, and D. Negrut. Using krylov subspace and spectral methods for solving complementarity problem in many-body contact dynamics simulation. *Int. J. Numer. Meth. Engng*, 00:1–21, 2012.

[15] Boston Dynamics. www.bostondynamics.com/robot_Atlas.html, 2013.