2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

Convex and analytically-invertible dynamics with contacts and constraints: Theory and implementation in MuJoCo

Emanuel Todorov

***Abstract*— We describe a full-featured simulation pipeline implemented in the MuJoCo physics engine. It includes multi-joint dynamics in generalized coordinates, holonomic constraints, dry joint friction, joint and tendon limits, frictionless and frictional contacts that can have sliding, torsional and rolling friction. The forward dynamics of a 27-dof humanoid with 10 contacts are evaluated in 0.1 msec. Since the simulation is stable at 10 msec timesteps, it can run 100 times faster than real-time on a single core of a desktop processor. Furthermore the entire simulation pipeline can be inverted analytically, an order-of-magnitude faster than the corresponding forward dynamics. We soften all constraints, in a way that avoids instabilities and unrealistic penetrations associated with earlier spring-damper methods and yet is sufficient to allow inversion. Constraints are imposed via impulses, using an extended version of the velocity-stepping approach. For holomonic constraints the extension involves a soft version of the Gauss principle. For all other constraints we extend our earlier work on complementarity-free contact dynamics – which were already known to be invertible via an iterative solver – and develop a new formulation allowing analytical inversion.**

I. Introduction

Contacts enable robots to interact with the environment and get a job done. Yet their discontinuous nature complicates simulation, planning and control. One way to sidestep these complications is to focus on the smooth dynamics in between contact events, and handle the transitions via problem-specific and often adhoc methods that tend to limit the capabilities of the robot. Another way is to smooth the contacts using spring-dampers. Here we do not refer to detailed models of material deformations (which are accurate yet slow), but rather to phenomenological spring-damper contact models in the context of rigid-body dynamics. That approach has been superseded by the velocity-stepping complementarity approach [1], [2], where the contact impulse (i.e. the integral of the contact force over the simulation timestep) is computed by solving a linear or nonlinear complementarity problem. Apart from its mathematical elegance, this approach avoids instabilities and unrealistic penetrations and handles interactions among simultaneous contacts – allowing much larger timesteps. However it also has shortcomings: it calls for a computation that is NP-hard in its exact form [3], and furthermore it only solves the simulation problem while planning and control remain difficult due to contact discontinuities.

This work was supported by the NSF and DARPA. E. Todorov is with the departments of Applied Mathematics and Computer Science & Engineering, University of Washington. Thanks to Yuval Tassa for discussions and comments on the manuscript.

A. Complementarity-free contact dynamics

The shortcomings of the now-standard complementarity approach motivated complementarity-free methods, developed independently in [4] and [5]. The idea is to relax the strict complementarity condition and instead enforce it approximately. This relaxation transforms the NP-hard problem into a convex optimization problem: a conic program when the friction cone is treated exactly, and a quadratic program when the cone is approximated with a pyramid. A systematic analysis of the effects of this approximation for complex robots remains to be done. Nevertheless, synthetic tests [6] as well as experience with model-based control [7] indicate that the approximation is accurate. Note also that the complementarity formulation is not necessarily a gold standard [8]. All contact models presently used in rigid-body simulations are phenomenological, and the only way to validate them is to measure the contact interactions between real robots and their environment – which is rarely done. Contact softness in particular may be better modeled by the complementarity-free approach. Such softness arises from regularization terms that are also essential for inversion.

B. Inverse dynamics with contacts and constraints

The goal of inverse dynamics it to compute the control forces *and* constraint impulses given the positions, velocities and accelerations. This is complicated by the fact that rigid-body contact dynamics are not actually invertible. Consider pushing against a wall. The contact force cannot be recovered from the kinematics, unless of course we measure the material deformations – but such deformations are ignored in the rigid-body approximation and in the complementarity approach. Indeed one of the notable advantages of the complementarity approach is that it considers the control force (as well as all other non-contact forces) before deciding what contact impulse to apply. This avoids penetration, but also makes it impossible to invert the dynamics or smooth the contacts. If on the other hand we were to use spring-dampers, the contact dynamics would be smooth and trivially invertible – but as mentioned above, spring-dampers have their own limitations that tend to outweigh their advantages.

Our complementarity-free approach [4] combines the best of both worlds: it allows smoothing and inversion, and at the same time considers both the inertia and the control forces in computing the contact impulses. The related approach [5] is not amenable to inversion because it uses a two-step optimization method, reminiscent of the staggered projection method developed for complementarity problems [3].

II. FORWARD DYNAMICS

A. Notation

Throughout the paper we use the following notation:

q	joint position
v	joint velocity
u	control force
h	discrete timestep
D	armature, implicit damping inertia
$M(q)$	total joint-space inertia
$c(q,v)$	gravity, Coriolis, centripetal forces
$p(q,v)$	spring-dampers, other passive forces
$J_{\mathrm{E}}(q)$	equality constraint Jacobian
$f_{\mathrm{E}}(q,v,u)$	equality constraint impulse
$J(q)$	contact Jacobian
$f(q,v,u)$	contact impulse
$\mathrm{v}^+, \mathrm{v}^-, \mathrm{v}^*$	impulse-space velocities defined later

Since the treatment of equality constraints and contacts is related, we use the same symbols (f, J) and the subscript $_{\mathrm{E}}$ to distinguish between the two. In addition, the following model parameters will be defined later:

ϵ	impulse regularization scaling
κ	error reduction time constant
η	loss due to dry joint friction
d	number of contact friction dimensions
$\mu_1, \cdots, \mu_d$	coefficients of elliptical friction cone

Multiple friction coefficients are needed to handle tangential, torsional and rolling friction; thus d can be up to 5.

B. Overall computation

We consider multi-joint systems subject to holonomic constraints such as loop joints, and contact impulses arising from dry joint friction, joint and tendon limits, frictionless and frictional contacts. The continuous-time dynamics are

$$M\,dv + c\,dt = (p+u)\,dt + J^T f + J_{\mathrm{E}}^T f_{\mathrm{E}} \tag{1}$$

We have not divided by dt because f_{E}, f are impulses.

The computation is carried out in two phases. Phase I corresponds to smooth dynamics and preparation for impulse dynamics, and relies on standard methods [9]. Phase II corresponds to impulse dynamics; it is the more the challenging part and is also where the novelty of our approach lies.

In Phase I we compute $M, D, c, p, J_{\mathrm{E}}, J$. In our implementation in the MuJoCo physics engine, $M(q) - D$ is computed with the Composite Rigid Body (CRB) algorithm, then D is added and the resulting M is LDL-factorized taking advantage of branch-induced sparsity. $c(q,v)$ is computed with the Recursive Newton-Euler (RNE) algorithm – which is actually an algorithm for inverse dynamics:

$$\mathrm{RNE}(q,v,\dot{v}) = (M(q) - D)\,\dot{v} + c(q,v) \tag{2}$$

Here we use it to compute $\mathrm{RNE}(q,v,0) = c(q,v)$. Note that CRB and RNE do not take into account the extra inertia D, which is why we have to add it to the output of these algorithms to obtain the total inertia M. The contact Jacobian $J(q)$ is computed using collision detection. The quantities $p(q,v), J_{\mathrm{E}}(q)$ are computed from analytical formulas or user callbacks. The extra diagonal inertia D models the armature inertia of motors as well as implicit damping.

We do not use Featherstone's $O(n)$ forward dynamics. This is because the impulse phase needs the inertia matrix to be computed and factorized, and once this is done, using RNE is faster. Thus our method has $O(n^3)$ worst-case performance. However, as Featherstone showed in [10], branch-induced sparsity typical for robotic systems makes the present method very similar to his $O(n)$ method.

In Phase II we compute (f_{E}, f) given (q, v, u) as explained in the following subsections. This is done in two stages: we first eliminate f_{E} by expressing it as a function f, and then project the dynamics in contact space and solve for f.

Before proceeding with the impulse computation, we need to transition to discrete time. Let $h > 0$ be the timestep. Replace $dv(t)$ with $v(t+h) - v(t)$. All relevant quantities except $v(t+h)$ are defined at time t, thus we omit t and write $v(t+h)$ as v'. The discrete-time dynamics are

$$M(v' - v) = (p + u - c)\,h + J^T f + J_{\mathrm{E}}^T f_{\mathrm{E}} \tag{3}$$

Since M is symmetric positive-definite, v' can be found as

$$v' = v + M^{-1}\left((p + u - c)\,h + J^T f + J_{\mathrm{E}}^T f_{\mathrm{E}}\right)$$

Now that we have transitioned to discrete time, we can clarify how the implicit damping terms in D are computed. Recall that "implicit" refers to evaluating quantities at the next timestep rather than the current timestep, and makes numerical integration more stable. In the case of a damped 2nd-order system $\overline{M}\dot{v} = -Bv$, implicit damping is implemented in discrete time as

$$\overline{M}(v' - v) = -Bv'h$$

These dynamics can be written in the familiar (explicit) form

$$M(v' - v) = -Bvh$$

by modifying the inertia as $M = \overline{M} + Bh$. Thus we implement implicit damping by defining $D \equiv \mathrm{armature} + Bh$, and adding $-Bv$ to the passive force $p(q,v)$. We require D to be diagonal and non-negative, so as to preserve the sparsity and positive-definiteness of M.

C. Constraint dynamics

The dynamics of equality-constrained systems can be obtained from the Gauss principle [11], which we now recall. Suppose we have unconstrained continuous-time dynamics $Ma = \tau$ subject to acceleration constraints $J_{\mathrm{E}}a = a_{\mathrm{E}}^*$. Then, given $M, \tau, J_{\mathrm{E}}, a_{\mathrm{E}}^*$, the constrained acceleration is the solution to the convex optimization problem

$$\min_{a\,:\,J_{\mathrm{E}}a = a_{\mathrm{E}}^*} (Ma - \tau)^T M^{-1} (Ma - \tau) \tag{4}$$

Since we aim to invert the dynamics later, and hard constraints are non-invertible, we must soften the constraints somehow. We propose to do this by softening the Gauss principle, i.e. replacing the hard constraint with a soft penalty.

Proposition. The acceleration of the constrained system is defined as the solution to the convex optimization problem

$$\min_a \left(Ma-\tau\right)^T M^{-1}\left(Ma-\tau\right)+ \left(J_{\mathrm{E}}a-a^*_{\mathrm{E}}\right)^T R_{\mathrm{E}}^{-1}\left(J_{\mathrm{E}}a-a^*_{\mathrm{E}}\right) \tag{5}$$

where the regularizer R_{E} is a diagonal positive matrix.

In the limit $R_{\mathrm{E}} \to 0$ the solution to problem (5) converges to the solution to problem (4), which is non-invertible. However for any $R_{\mathrm{E}} \succ 0$ the resulting dynamics are invertible, as shown by construction later.

We now return to discrete time and impose a soft constraint on velocity rather than acceleration: $J_{\mathrm{E}}v' = \mathrm{v}^*_{\mathrm{E}}(q,v)$. Here $\mathrm{v}^*_{\mathrm{E}}$ is the desired next-step velocity in constraint space, computed by any suitable constraint stabilization mechanism. Our specific choice will be described later.

Theorem 1. The dynamics under the proposed soft Gauss principle are

$$v' = \widehat{v} + \widehat{M}^{-1}J^T f \tag{6}$$

where $\widehat{M}$ and $\widehat{v}$ are defined as

$$\begin{aligned} \widehat{M} &\equiv M + J_{\mathrm{E}}^T R_{\mathrm{E}}^{-1} J_{\mathrm{E}} \\ \widehat{v} &\equiv v + \widehat{M}^{-1}\left((p+u-c)\,h + J_{\mathrm{E}}^T R_{\mathrm{E}}^{-1}\left(\mathrm{v}^*_{\mathrm{E}} - J_{\mathrm{E}}v\right)\right) \end{aligned} \tag{7}$$

Proof. Apply the soft Gauss principle with

$$a = \frac{v'-v}{h},\; a^*_{\mathrm{E}} = \frac{\mathrm{v}^*_{\mathrm{E}} - J_{\mathrm{E}}v}{h},\; \tau = p+u-c+\frac{J^T f}{h}$$

Solving (5) analytically yields (6, 7). ■

We can interpret $\widehat{M}$ as the apparent inertia that takes into account the constraints, and $\widehat{v}$ as the next-step velocity that takes into account all forces except for the contact impulse.

For hard constraints the dynamics remain in the general form (6), but the definitions (7) are replaced with

$$\begin{aligned} \widehat{M}^{-1} &\equiv M^{-1} - M^{-1}J_{\mathrm{E}}^T A_{\mathrm{E}}^{-1} J_{\mathrm{E}} M^{-1} \\ \widehat{v} &\equiv v + \widehat{M}^{-1}\left(p+u-c\right)h + M^{-1}J_{\mathrm{E}}^T A_{\mathrm{E}}^{-1}\left(\mathrm{v}^*_{\mathrm{E}} - J_{\mathrm{E}}v\right) \end{aligned}$$

where $A_{\mathrm{E}} \equiv J_{\mathrm{E}}M^{-1}J_{\mathrm{E}}^T$ is the inverse inertia in constraint space. This can be shown by solving (4), or by taking the limit $R_{\mathrm{E}} \to 0$ and using the matrix inversion lemma.

Equations (6, 7) represent modified dynamics which implicitly take the constraints into account. We did not compute the impulse f_{E} explicitly because it can only be computed after v' is known, i.e. in the context of inverse dynamics.

D. Contact dynamics

We now have everything in place for the contact computation stage. First we project (6) in contact space via multiplication by the contact Jacobian J:

$$Jv' = J\widehat{v} + J\widehat{M}^{-1}J^T f$$

Then we write the contact-space dynamics as

$$Af + \mathrm{v}^- = \mathrm{v}^+ \tag{8}$$

where $A \equiv J\widehat{M}^{-1}J^T$ is the inverse of the apparent inertia, $\mathrm{v}^- \equiv J\widehat{v}$ is the next-step velocity before the contact impulse, and $\mathrm{v}^+ \equiv Jv'$ is the next-step velocity after the contact impulse, all expressed in contact space.

To complete the forward dynamics computation we have to solve (8) for (f, v^+) given (A, v^-). This involves twice as many unknowns as the number of equations in (8), thus we need additional information – which comes in the from of inequality constraints reflecting the laws of contact and friction. We now introduce these inequalities, and at the same time explain what exactly is included in the contact space.

In our current implementation in MuJoCo the contact solver can handle three types of objects: friction loss in the joints, limits on joint angles and distances, and frictional contacts. The elements f_i of the vector f satisfy different inequality constraints depending on the type of object they represent, as follows:

$$\begin{aligned} &\text{friction:} && \eta(i) - f_i \geq 0 \\ & && \eta(i) + f_i \geq 0 \\ &\text{limit:} && f_i \geq 0 \\ &\text{contact:} && f_i \geq 0 \\ & && f_i^2 - \sum_{j=1}^{d(i)} \left(\frac{f_{i+j}}{\mu_j(i)}\right)^2 \geq 0 \end{aligned} \tag{9}$$

Assembling the left hand sides of the above inequalities in the vector $\phi(f)$, we can write (9) as $\phi(f) \geq 0$.

In (9) subscripts denote vector elements as usual, while i in brackets denotes parameters of the object whose data starts at position i in the vector f. In the case of friction loss, $\eta(i)$ is the (load-independent) joint torque that is lost to friction before the joint starts accelerating. Limits can be defined for revolute, prismatic and ball joints (the latter limits are cylinders in the angle-axis representation of the joint quaternion), as well as for tendon lengths (tendons are strings whose spatial path is defined by via points and wrapping objects), and distances between geometric shapes (i.e. frictionless contacts).

In the case of frictional contacts, $d(i) \in \{2,3,5\}$ is the dimensionality of the friction space and $\mu_j(i)$ are the friction coefficients in the $d(i)$ dimensions. The contact contributes $1+d(i)$ elements to the vector f with the following semantics. Consider a 3D frame whose first axis is aligned with the contact normal. Impulses (f_i, f_{i+1}, f_{i+2}) cause relative translation along the contact frame axes, while $(f_{i+3}, f_{i+4}, f_{i+5})$ cause relative rotation around the axes. Thus f_i is the normal impulse, (f_{i+1}, f_{i+2}) is the tangential friction impulse, f_{i+3} is the torsional friction impulse, and (f_{i+4}, f_{i+5}) is the rolling friction impulse. The contact inequalities in (9) specify that the normal impulse must be non-negative, and that the impulse vector must lie within the elliptical friction cone. In the special case when $d(i) = 2$ and $\mu_1(i) = \mu_2(i) = \mu(i)$, this reduces to the more familiar definition of a friction cone:

$$\mu(i)\, f_i \geq \sqrt{f_{i+1}^2 + f_{i+2}^2}$$

We now return to the computation of the contact impulse f and next-step velocity v^+. Conditions (8, 9) are still

insufficient to determine a unique solution. There are two paths forward: complementarity-based and complementarity-free. The former approach introduces additional constraints (complementarity conditions) to obtain a unique solution. For example, the contact normal inequality $f_i \geq 0$ is augmented with $\mathrm{v}_i^+ \geq 0$ and $f_i \mathrm{v}_i^+ = 0$. In the presence of frictional contacts, this approach yields an NP-hard problem in the forward dynamics. Furthermore its inverse cannot be defined. Thus we will not pursue it further in this paper. Instead we will rely on the complementarity-free approach.

E. Complementarity-free contact dynamics

In our approach to contact dynamics [4], the impulse f is defined via minimization of the (regularized and offset) next-step kinetic energy subject to (9):

$$\min_{f\,:\,\phi(f)\geq 0} \left(\mathrm{v}^+ - \mathrm{v}^*\right)^T A^{-1} \left(\mathrm{v}^+ - \mathrm{v}^*\right) + f^T R f \qquad (10)$$

The regularizer R is a diagonal positive matrix, which is needed because A can be singular and also because it introduces smoothing that is necessary to define the inverse later. Kinetic energy is measured relative to a desired contact velocity v^* which can be computed by any suitable contact stabilization mechanism (details below).

Substituting (8) in (10), the impulse f is found as

$$\min_{f\,:\,\phi(f)\geq 0} \frac{1}{2} f^T (A + R) f + f^T \left(\mathrm{v}^- - \mathrm{v}^*\right) \qquad (11)$$

This is a convex optimization problem and has a unique global minimum. We solve it using our (yet unpublished) generalization of the projected Gauss-Seidel method (GPGS) that can handle cone and pyramid constraints.

III. Inverse dynamics

A. Overall inverse computation

Given (q, v, v'), we compute $J, J_{\mathrm{E}}, D, p, \mathrm{v}^*, \mathrm{v}_{\mathrm{E}}^*$ as in the forward dynamics, and define $\dot{v} \equiv (v' - v)\, h^{-1}$. Then we apply the RNE algorithm to compute the sum of all forces acting on the system except for the Coriolis, centripetal and gravity force $c(q, v)$ and the extra inertial force $D\dot{v}$. More precisely, from (1) and (2) we have

$$\mathrm{RNE}(q, v, \dot{v}) + D\dot{v} = p + u + \frac{J^T f + J_{\mathrm{E}}^T f_{\mathrm{E}}}{h} \qquad (12)$$

Once the impulses f, f_{E} are computed as explained below, the control force u is recovered from (12) and we are done.

Note that we did not need to compute or factorize the inertia matrix M. It will turn out that M is not needed to recover the impulses either.

B. Inverse constraint dynamics

Recall that in the forward dynamics we did not compute the constraint impulse f_{E}, but only modified the dynamics so as to take it into account implicitly. Here f_{E} can be computed explicitly because v' is known.

Theorem 2. The constraint impulse f_{E} which caused the observed state transition $(q, v) \to v'$ satisfies

$$J_{\mathrm{E}}^T f_{\mathrm{E}} = J_{\mathrm{E}}^T R_{\mathrm{E}}^{-1} \left(\mathrm{v}_{\mathrm{E}}^* - J_{\mathrm{E}} v'\right) \qquad (13)$$

Proof. Combining (6) and (7) yields

$$\left(M + J_{\mathrm{E}}^T R_{\mathrm{E}}^{-1} J_{\mathrm{E}}\right)(v' - v) = (p + u - c)\, h + J^T f + J_{\mathrm{E}}^T R_{\mathrm{E}}^{-1} \left(\mathrm{v}_{\mathrm{E}}^* - J_{\mathrm{E}} v\right)$$

Subtracting this from (3) yields the result (13). ■

We can further recover the actual f_{E} if we assume that the equality constraints are non-redundant (i.e. that J_{E} has full rank). However this assumption is not needed to complete the inverse dynamics computation, because (12) only depends on $J_{\mathrm{E}}^T f_{\mathrm{E}}$ which we already have from (13).

C. Inverse contact dynamics: General case

In our previous work [4] we showed how the convex contact model described above can be inverted in an unconstrained setting, by converting the inequality constraints into log-barrier penalty functions. Here we present a more general version of this result, allowing hard inequality constraints – which in turn make it possible to use projected and active-set methods in the forward dynamics. We first state the abstract result and then specialize it to dynamic simulation.

Theorem 3. Let $A \in \mathbb{R}^{n,n}$ be symmetric positive semi-definite, $r, s : \mathbb{R}^n \to \mathbb{R}$ be convex, $\phi : \mathbb{R}^n \to \mathbb{R}^n$ be convex, and $b, c \in \mathbb{R}^n$. Define $x_b^*, x_c^* \in \mathbb{R}^n$ as the (unique global) solutions to the following convex optimization problems:

$$x_b^* : \min_{x\,:\,\phi(x)\geq 0} r(x) + s(Ax + b) + \frac{1}{2} x^T A x + x^T b \qquad (14a)$$

$$x_c^* : \min_{x\,:\,\phi(x)\geq 0} r(x) + x^T (A \nabla s(c) + c) \qquad (14b)$$

Then $Ax_b^* + b = c$ implies $x_b^* = x_c^*$.

***Proof*.** Since x_b^*, x_c^* are defined as solutions to convex optimization problems, they can be equivalently characterized by the corresponding Karush-Kuhn-Tucker (KKT) conditions. The KKT conditions for problem (14a) are:

$$\nabla r(x) + A \nabla s(Ax + b) + Ax + b + \left(\frac{\partial \phi}{\partial x}\right)^T \lambda = 0$$
$$\phi(x) \geq 0,\ \lambda \geq 0,\ \phi(x)^T \lambda = 0$$

The KKT conditions for problem (14b) are:

$$\nabla r(x) + A \nabla s(c) + c + \left(\frac{\partial \phi}{\partial x}\right)^T \lambda = 0$$
$$\phi(x) \geq 0,\ \lambda \geq 0,\ \phi(x)^T \lambda = 0$$

These two sets of conditions are identical when $Ax + b = c$. Therefore, if the solution x_b^* to the first problem satisfies $Ax_b^* + b = c$, then x_b^* will also be the solution to the second problem, and so $x_b^* = x_c^*$. ■

This theorem can be applied to complementarity-free contact dynamics by identifying x with f, b with v^-, c with v^+, and absorbing the linear term $-f^T \mathrm{v}^*$ in $r(f)$. Thus we have the following corollary.

Corollary. Given $A, \mathrm{v}^*, \mathrm{v}^-$, the impulse is computed by solving the (forward dynamics) convex optimization problem

$$\min_{f\,:\,\phi(f)\geq 0} r(f) + s\left(Af + \mathrm{v}^-\right) + \frac{1}{2} f^T A f + f^T \left(\mathrm{v}^- - \mathrm{v}^*\right)$$

Given $A, \mathrm{v}^*, \mathrm{v}^+$, the impulse is computed by solving the (inverse dynamics) convex optimization problem

$$\min_{f\,:\,\phi(f)\geq 0} r\left(f\right) + f^T\left(A\nabla s\left(\mathrm{v}^+\right) + \mathrm{v}^+ - \mathrm{v}^*\right)$$

Both computations yield the same impulse f.

D. Inverse contact dynamics: Analytical special case

We now focus on the specific formulation (11) used in the forward dynamics, where instead of a general impulse regularizer $r\left(f\right)$ we had

$$r\left(f\right) = \frac{1}{2} f^T R f - f^T \mathrm{v}^*$$

In this case the inverse can be computed analytically. In particular, problem (14b) can be written in least-squares form

$$\min_{f\,:\,\phi(f)\geq 0} \frac{1}{2}\left(f-y\right)^T R\left(f-y\right) \tag{15}$$

where the vector y is defined as

$$y \equiv R^{-1}\left(\mathrm{v}^* - \mathrm{v}^+\right)$$

Including a general velocity regularizer $s\left(\mathrm{v}^+\right)$ would merely add a constant to y and still allow the inverse to be computed analytically, but we omit it here so as to match the forward dynamics formulation (11) used by our GPGS solver.

We now make a key observation: the terms corresponding to the different objects used to construct the contact space are decoupled, both in the objective function and in the constraints. Thus problem (15) decomposes into a collection of smaller problems – one for each friction loss, limit, and frictional contact object. These smaller problems can be solved analytically as follows.

Theorem 4. The solution to the inverse dynamics optimization problem (15) is

$$\begin{aligned}
&\text{friction:} && f_i = \max\left(-\eta\left(i\right), \min\left(\eta\left(i\right), y_i\right)\right)\\
&\text{limit:} && f_i = \max\left(0, y_i\right)\\
&\text{contact:} && f_i, \cdots, f_{i+d(i)} = \text{ConeProject}\left(y, i\right)
\end{aligned}$$

ConeProject extracts the contact-specific data $y_i, \cdots, y_{i+d(i)}$ from y, and then calls Algorithm 5 below to compute the nearest vector within the friction cone.

E. Projection on a friction cone

Here we develop the analytical procedure for projecting a vector on a friction cone with certain properties. Let $\mathcal{C}$ be an elliptical cone defined as

$$\mathcal{C} \equiv \left\{x \in \mathbb{R}^{d+1} : x_0 \geq 0,\ x_0^2 \geq \sum\nolimits_{j=1}^{d} \mu_j^{-2} x_j^2\right\}$$

We seek the vector $x \in \mathcal{C}$ which minimizes the weighted distance to a given vector $y \in \mathbb{R}^{d+1}$:

$$\min_{x\,:\,x\in\mathcal{C}} \sum\nolimits_{k=0}^{d} r_k\left(x_k - y_k\right)^2$$

Here $r \in \mathbb{R}^{d+1}$ is a vector of positive weights, i.e. the diagonal of the matrix R above. Throughout this section k is an index starting at 0 while j is an index starting at 1.

This general problem cannot be solved analytically because both the cone and the distance metric are elliptical, making the problem equivalent to finding the roots of a polynomial of order $2\left(d+1\right)$. Our goal then is to identify conditions that simplify the problem, in particular make the cone circular and the metric Cartesian. The metric is made Cartesian by the change of variables

$$\widehat{x}_k \equiv r_k^{1/2} x_k, \quad \widehat{y}_k \equiv r_k^{1/2} y_k \tag{16}$$

The minimization problem now becomes

$$\min_{\widehat{x}\,:\,\widehat{x}\in\widehat{\mathcal{C}}} \sum\nolimits_k \left(\widehat{x}_k - \widehat{y}_k\right)^2$$

where the transformed friction cone $\widehat{\mathcal{C}}$ is

$$\widehat{\mathcal{C}} = \left\{\widehat{x} \in \mathbb{R}^{d+1} : \widehat{x}_0 \geq 0,\ r_0^{-1}\widehat{x}_0^2 \geq \sum\nolimits_j \mu_j^{-2} r_j^{-1} \widehat{x}_j^2\right\}$$

This cone is circular when $\mu_j^2 r_j = const$, which can be enforced by requiring that there exists a scalar $\widehat{\mu}$ such that

$$\mu_j^2 r_j = \widehat{\mu}^2 r_0 \tag{17}$$

The transformed friction cone then takes the familiar form

$$\widehat{\mathcal{C}} = \left\{\widehat{x} \in \mathbb{R}^{d+1} : \widehat{x}_0 \geq 0,\ \widehat{\mu}^2\widehat{x}_0^2 \geq \sum\nolimits_j \widehat{x}_j^2\right\}$$

The resulting optimization problem can now be solved analytically. If $\widehat{y} \in \widehat{\mathcal{C}}$ then $\widehat{x} = \widehat{y}$ and we are done. Otherwise $\widehat{x}$ lies on the surface of the cone $\widehat{\mathcal{C}}$ and can be found via Lagrange multipliers: there exists a scalar λ such that

$$\widehat{x} - \widehat{y} + \lambda\left(\,\widehat{\mu}^2\widehat{x}_0, -\widehat{x}_1, \cdots, -\widehat{x}_d\right)^T = 0$$

We can find λ by expressing $\widehat{x}$ as a function of $\widehat{y}$ and λ:

$$\widehat{x}_0 = \frac{\widehat{y}_0}{1 + \,\widehat{\mu}^2\lambda}, \quad \widehat{x}_j = \frac{\widehat{y}_j}{1-\lambda} \tag{18}$$

Substituting this $\widehat{x}$ in the surface equality constraint and solving for λ yields two general solutions:

$$\lambda^{\pm} = \frac{1 \mp \beta}{1 \pm \widehat{\mu}^2\beta}, \quad \beta = \left(\frac{\sum_j \widehat{y}_j^2}{\widehat{\mu}^2\widehat{y}_0^2}\right)^{1/2} \tag{19}$$

with corresponding vectors $\widehat{x}^{\pm}$ given by (18). If both vectors have non-negative first components $\left(\widehat{x}_0^{\pm} \geq 0\right)$, the optimal solution is the one closer to $\widehat{y}$. If only one vector has non-negative first component, it is the optimal solution. If both have negative first components, the optimal solution is $\widehat{x} = 0$.

We must also handle two special cases where the above general method involves division by 0. If $\widehat{y}_0 = 0$ we have $1 + \,\widehat{\mu}^2\lambda = 0$. In that case the solution can be shown to be

$$\widehat{x}_0 = \frac{\widehat{\mu}}{1+\widehat{\mu}^2}\left(\sum\nolimits_j \widehat{y}_j^2\right)^{1/2}, \quad \widehat{x}_j = \frac{\widehat{\mu}^2}{1+\widehat{\mu}^2}\widehat{y}_j \tag{20}$$

The other special case is $\widehat{y}_j = 0$ for all $j \geq 1$, resulting in $1 - \lambda = 0$. In that case the solution is the tip of the friction cone $\widehat{x} = 0$.

We now clarify how the contact model can be constructed so as to obey the restriction (17). The regularizing weights $r_0, \cdots, r_d$ together with the friction coefficients $\mu_1, \cdots, \mu_d$ and the transformed coefficient $\widehat{\mu}$ appear to have $2d+2$

degrees of freedom, however d of them are removed due to (17), leaving us with $d+2$ degrees of freedom. The friction coefficients clearly need to be under the control of the user, thus only two of the regularizers can be specified independently. It is then natural to construct the contact model by specifying the following independent parameters:

$$\begin{array}{ll} \text{friction coefficients:} & \mu_1, \cdots, \mu_d \\ \text{normal regularizer:} & r_0 \\ \text{mean friction regularizer:} & r_{\mathrm{F}} \end{array} \tag{21}$$

Defining

$$r_j \equiv \frac{r_{\mathrm{F}} \mu_j^{-2}}{\left\langle \mu_j^{-2} \right\rangle}, \quad \widehat{\mu}^2 \equiv \frac{r_{\mathrm{F}}}{r_0 \left\langle \mu_j^{-2} \right\rangle} \tag{22}$$

where $\langle . \rangle$ denotes the mean value, we can verify that (17) is satisfied and furthermore $r_{\mathrm{F}} = \langle r_j \rangle$ as intended. We now summarize the algorithm.

Algorithm 5. Given the contact model parameters (21) and the vector y, compute the vector x as follows:

1) if y is inside the friction cone, set $x = y$ and return;
2) if $y_j = 0$ for all $j \geq 1$, set $x = 0$ and return;
3) compute r_j and $\widehat{\mu}^2$ from (22);
4) compute $\widehat{y}$ from (16);
5) if $\widehat{y}_0 = 0$, compute $\widehat{x}$ from (20) and go to step 8;
6) compute $\widehat{x}^{\pm}$ from (18, 19);
7) choose the optimal $\widehat{x}$ among $\{\widehat{x}^{\pm}, 0\}$;
8) compute x by inverting (16).

As a sanity check, we generated random optimization problems in this family and compared our analytical solution to the solution found by the MATLAB `fmincon` iterative solver. The two solutions agreed within the tolerance level specified for the iterative solver.

IV. Computational complexity

Let $n = \dim(v)$ be the number of degrees of freedom and $m = \dim(f)$ the number of impulses. We will ignore holonomic constraints in this analysis because the computational cost is dominated by the contact impulses.

The forward dynamics involve computing and factorizing the inertia M which is $O\left(n^3\right)$. Computing $A = JM^{-1}J^T$ is $O\left(m^2 n + n^2 m\right)$. Applying the GPGS iterative solver is $O\left(m^2 \text{ maxiter}\right)$. Note however that the branch-induced sparsity of M as well as the sparsity of J make the actual performance better than these worst-case estimates. Our implementation exploits sparsity.

The inverse dynamics involve RNE which is $O(n)$, as well as our new analytical impulse solver which is $O(m)$. We use a precomputed approximation to the diagonal of A to set the regularizer R, avoiding any higher-order operations. Thus the worst-case performance is dominated by the computation of the contact Jacobian J which is $O(mn)$. Again, sparsity makes the actual performance better.

The above computational complexity analysis does not include collision detection – which is identical in both the forward and inverse dynamics. In simulations relevant to robotics, we have found collision detection to be a small fraction of the computational cost.

V. Analysis and tuning of the impulse dynamics

Here we analyze the behavior of the dynamics defined above. We also show how to set the solver parameters $R, R_{\mathrm{E}}, \mathrm{v}^*, \mathrm{v}_{\mathrm{E}}^*$ so as to obtain a stabilization mechanism.

For contacts we focus on the case when all inequality constraints are inactive, i.e. $\phi(f) > 0$ at the solution f found by the impulse solver. Physically this corresponds to non-sliding contacts (modulo contact softness). In that case, from (15) we have

$$f = R^{-1}\left(\mathrm{v}^* - Jv'\right) \tag{23}$$

Note how similar this contact impulse is to the constraint impulse (13), despite the fact that the two were defined and computed differently. This observation motivates analysis in the combined space of constraints and contacts, with coordinates x defined as follows. For constraints, limits and contact normals x_i is the violation/penetration distance. For frictional dimensions x_i is defined relative to an arbitrary offset (we only care about the velocity in that case).

Define the Jacobian $\mathcal{J}$, regularization matrix $\mathcal{R}$ and desired next-step velocity $\dot{x}^*$ in x-space by stacking the corresponding quantities for constraints and contacts:

$$\mathcal{J} \equiv \begin{bmatrix} J_{\mathrm{E}} \\ J \end{bmatrix}, \; \mathcal{R} \equiv \begin{bmatrix} R_{\mathrm{E}} & 0 \\ 0 & R \end{bmatrix}, \; \dot{x}^* \equiv \begin{bmatrix} \mathrm{v}_{\mathrm{E}}^* \\ \mathrm{v}^* \end{bmatrix}$$

Then the velocity and inverse inertia in x-space are

$$\dot{x} = \mathcal{J}v, \; \mathcal{A} = \mathcal{J}M^{-1}\mathcal{J}^T$$

As before, $\dot{x}' = \dot{x} + h\ddot{x}$ will denote the actual next-step velocity. We will also need the x-space acceleration that the non-impulsive forces cause:

$$a \equiv \mathcal{J}M^{-1}\left(p - c + u\right)$$

Finally we must decide how the desired next-step velocity $\dot{x}^*$ is computed; different choices give rise to different dynamics. Motivated by the idea of Baumgarte stabilization [12], we consider a virtual PD controller that causes acceleration $-\mathcal{B}\dot{x} - \mathcal{K}x$. Thus $\dot{x}^*$ is defined as

$$\dot{x}^* \equiv \dot{x} - h\mathcal{B}\dot{x} - h\mathcal{K}x \tag{24}$$

We now have everything in place to obtain the dynamics.

Theorem 6. The x-space dynamics are

$$\left(I + \mathcal{A}\mathcal{R}^{-1}\right)\ddot{x} + \mathcal{A}\mathcal{R}^{-1}\mathcal{B}\dot{x} + \mathcal{A}\mathcal{R}^{-1}\mathcal{K}x = a \tag{25}$$

Proof. Substituting (23) and (13) in (3) yields

$$M\left(v' - v\right) = \left(p - c + u\right)h + J_{\mathrm{E}}^T R_{\mathrm{E}}^{-1}\left(\mathrm{v}_{\mathrm{E}}^* - J_{\mathrm{E}}v'\right) + J^T R^{-1}\left(\mathrm{v}^* - Jv'\right)$$

Multiplying by $\mathcal{J}M^{-1}$ and using the above definitions,

$$\dot{x}' - \dot{x} = ah + \mathcal{A}\mathcal{R}^{-1}\left(\dot{x}^* - \dot{x}'\right)$$

Using the definitions of $\dot{x}'$ and $\dot{x}^*$ yields the result (25). ■

We can now gain a better understanding of what the impulse solver is doing. Suppose we set $\mathcal{R} = \mathcal{A}\epsilon$ for some positive ϵ. Since $\mathcal{A}$ and $\mathcal{R}$ have the same units (they were added together in (11)), ϵ is a dimensionless constant. Also

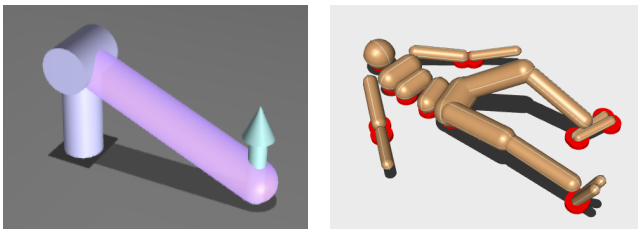

Fig. 1. The two MuJoCo models used in the simulations. Left: single-joint mechanism with dry friction. The arrow shows the contact impulse. Right: 27-dof humanoid. The red cylinders show the active contacts. Joint limits prevent more contacts.

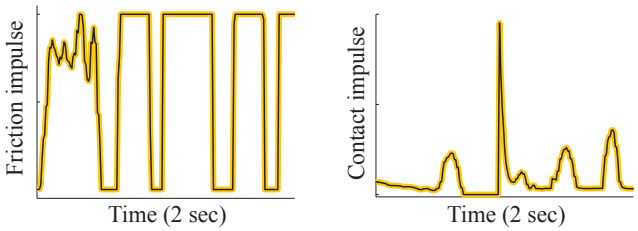

Fig. 2. Comparison of the friction impulse and contact impulse computed by the forward (black) and inverse (orange) dynamics, while the single-joint system was being perturbed randomly. The two curves in each plot are within 1E-5 of each other, thus the difference is not visible.

set $\mathcal{B} = \overline{\mathcal{B}}\,(1+\epsilon)\,, \mathcal{K} = \overline{\mathcal{K}}\,(1+\epsilon)$ for some $\overline{\mathcal{B}}, \overline{\mathcal{K}}$. Assuming for the moment that $\mathcal{A}$ is invertible, (25) becomes

$$\ddot{x} + \overline{\mathcal{B}}\dot{x} + \overline{\mathcal{K}}x = \frac{\epsilon}{1+\epsilon}a \tag{26}$$

Therefore in the limit $\epsilon \to \infty$ we have a spring-damper driven by a. In the limit $\epsilon \to 0$ the spring-damper becomes autonomous and removes constraint violations regardless of a. The latter limit corresponds to hard constraints and contacts. In practice we use small ϵ. Since our solver implements the above dynamics implicitly, we can get very close to the $\epsilon \to 0$ limit without instabilities, numerical errors, or need for small timesteps.

While (26) is illuminating, it does not apply in general because $\mathcal{A}$ can be singular; for example, two frictional contacts on the same body make $\mathcal{A}$ singular. Furthermore we cannot set $\mathcal{R} = \mathcal{A}\epsilon$ because we want $\mathcal{R}$ to be diagonal, so that the dynamics can be inverted analytically.

In our implementation we set $\mathcal{R}$ to a diagonal matrix:

$$\mathcal{R}_{ii} = \max\left(\mathcal{A}_{ii}\epsilon, r_{\min}\right), \tag{27}$$

Similarly the stiffness and damping are diagonal:

$$\mathcal{B}_{ii} = 2\,(1+\epsilon)\,\kappa^{-1},\ \mathcal{K}_{ii} = (1+\epsilon)\,\kappa^{-2} \tag{28}$$

We can now carry the analysis of (25) further by approximating $\mathcal{A}$ with its diagonal, resulting in

$$\ddot{x} + 2\kappa^{-1}\dot{x} + \kappa^{-2}x \approx \frac{\epsilon}{1+\epsilon}a$$

Note that the choice of stiffness and damping coefficients in (28) made the autonomous part of the dynamics critically-damped, where κ^{-1} is the natural frequency and so κ is the time constant of the x-space dynamics. It is also informative to look at penetrations. If we consider a free-floating object resting on the ground in the presence of gravity g, we have $a = g$, and so the penetration is

$$x \approx \frac{\kappa^2 \epsilon g}{1+\epsilon}$$

independent of the mass of the object.

There are two final caveats. First, the $\mathcal{R}_{ii}$ corresponding to the friction dimensions of each contact must be further adjusted so as to satisfy (22). Second, in friction dimensions we only care about velocity, thus we set the corresponding $\mathcal{K}_{ii} = 0$. In that case the velocity decays to 0 at rate $2\kappa^{-1}$, and κ again has the meaning of a time constant.

To summarize, the user specifies the impulse regularization scaling ϵ and the error reduction time constant κ, and then the solver parameters $R, R_{\mathrm{E}}, \mathrm{v}^*, \mathrm{v}^*_{\mathrm{E}}$ are computed automatically using (27, 28, 24) with the above modifications.

Instead of using a diagonal approximation, we can obtain a stabilization mechanism that is closer to critical damping as follows. Define a diagonal $\mathcal{R}$ as above. Now solve the following equations for $\mathcal{B}$ and $\mathcal{K}$:

$$\mathcal{A}\mathcal{R}^{-1}\mathcal{B} = 2\kappa^{-1}\left(I + \mathcal{A}\mathcal{R}^{-1}\right)$$
$$\mathcal{A}\mathcal{R}^{-1}\mathcal{K} = \kappa^{-2}\left(I + \mathcal{A}\mathcal{R}^{-1}\right)$$

If $\mathcal{A}$ is invertible these equations can be solved exactly, resulting in exact critical damping. If not, then a pseudo-inverse can be used. The resulting $\mathcal{B}$ and $\mathcal{K}$ are no longer diagonal, but that does not complicate the computation.

VI. Simulations

A. Correctness

We demonstrate numerically the correctness of the inversion using a single-joint mechanism (Figure 1 left) with two impulses: dry friction in the joint and contact with the ground. Figure 2 compares these impulses as computed by the forward and inverse dynamics, while the mechanism is perturbed with random control forces.

Recall that the impulses in the forward dynamics are computed with our GPGS iterative solver which may not discover the optimal solution within the number of iterations we allow in runtime. Indeed for more complex simulations the forward and inverse do not agree so closely, making the inverse dynamics approach more appealing for optimal control and estimation applications.

B. Speed of computation

Figure 3 illustrates the computational efficiency of our algorithms and their implementation in MuJoCo. The tests were done on an Intel i7-3930K processor, Windows 7, single-threaded computation taking advantage of AVX instructions (with custom BLAS-like routines). We dragged the humanoid model around in the virtual environment using a 3D mouse. This generated many different contact configurations with different Jacobian size. We run both the forward and inverse dynamics and timed them using high-resolution timers. The timing data and size of the Jacobian at each simulation step were saved in a file. Afterwards, we found all simulation steps in which the Jacobian had a given

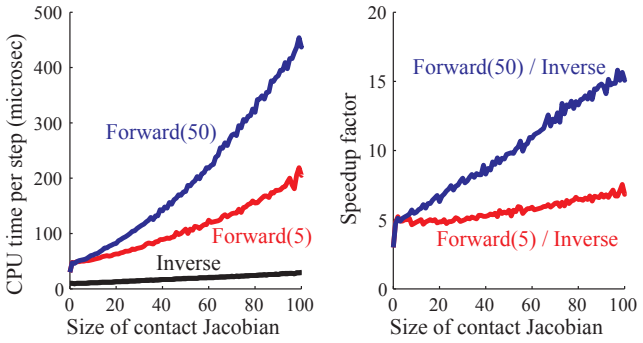

Fig. 3. Comparison of the speed of the forward and inverse dynamics computations for the humanoid. Left: CPU time per simulation step is shown for different sizes of the contact Jacobian. The PGS iterative solver in the forward dynamics was run for 5 or 50 iterations. Right: the speedup factor is the ratio of forward dynamics CPU time to inverse dynamics CPU time.

size (plotted on the x-axis in Figure 3), and computed the median of the corresponding CPU times. We performed the test twice, with 5 and 50 iterations of the GPGS solver in the forward dynamics. To generate larger Jacobians we enabled all contact friction directions. For 10 contacts with sliding friction only, the Jacobian size would be 30.

We find the results in Figure 3 quite remarkable. On a single core of a desktop processor, we can evaluate the inverse dynamics of a 27-dof humanoid subject to 100 impulses in 30 microseconds. In the absence of impulses the time goes down to 10 microseconds. Since the simulation timestep is 10 milliseconds, the inverse dynamics are being computed between 300 and 1000 times faster than real time depending on the number of impulses. The forward dynamics are slower – by an order of magnitude – but still much faster than real time. Note that increasing the number of GPGS iterations from 5 to 50 is expensive with large number of impulses, but the inverse dynamics are faster regardless of the number of GPGS iterations. This is because computing the full A matrix is avoided.

VII. Conclusions and Future Work

We described a full-featured simulation pipeline for multi-joint dynamics subject to constraints and contacts. Forward simulation requires an iterative solver, while inverse dynamics are computed analytically and faster.

The inversion is possible because our formulation allows a certain amount of softness. One can think of the impulses as being generated by smart spring-dampers, which are aware of each other and furthermore scale their stiffness and damping automatically with inertia. The forward formulation considers the interactions among all impulses, involving the dense matrix A and the diagonal regularizer R. The mathematical relation between the forward and inverse is such that the A matrix is no longer needed in the inverse, and we are left only with the R matrix allowing us to decompose the problem.

Softness is controlled by the two parameters ϵ, κ whose effects we analyzed. Small values of these parameters make the forward dynamics harder to simulate (requiring smaller timesteps or larger number of solver iterations). They also make the output of the inverse dynamics more sensitive to the input, but the actual computation is not affected since it always relies on the same analytical procedure.

Our softness parameters ϵ, κ are generally related to the constraint-force mixing (CFM) and error reduction parameter (ERP) introduced in the Open Dynamics Engine (ODE), although the specifics are different. CFM regularization in ODE is applied to the full system while we only apply regularization in the impulse space, and furthermore the problem solved by ODE does not become convex even after regularization. ERP implements a first-order constraint stabilization mechanism while we use a second-order mechanism.

Our results have many potential applications in data analysis, simulation, estimation and control. We have already leveraged the new analytical inverse dynamics in the context of state estimation, using it to enforce a physics consistency prior [14]. Future applications to optimal control are particularly exciting. We were recently able to synthesize complex full-body movements with direct trajectory optimization automatically, without relying on motion capture or manual scripting [13]. While this was done offline, we estimate that in model-predictive control (MPC) mode our existing optimizer will be an order of magnitude slower than real time. The analytical inverse dynamics developed here may provide the missing order of magnitude speedup. If we could replicate [13] in MPC mode, it is likely to transform robotic control as well as interactive games.

References

[1] D. Stewart and J. Trinkle, "An implicit time-stepping scheme for rigid-body dynamics with inelastic collisions and coulomb friction," *International Journal Numerical Methods Engineering*, vol. 39, pp. 2673–2691, 1996.

[2] M. Anitescu, F. Potra, and D. Stewart, "Time-stepping for three-dimensional rigid body dynamics," *Computer Methods in Applied Mechanics and Engineering*, vol. 177, pp. 183–197, 1999.

[3] D. Kaufman, S. Sueda, D. James, and D. Pai, "Staggered projections for frictional contact in multibody systems," *ACM Transactions on Graphics*, vol. 164, pp. 1–11, 2008.

[4] E. Todorov, "A convex, smooth and invertible contact model for trajectory optimization," *ICRA*, 2011.

[5] E. Drumwright and Shell, "Modeling contact friction and joint friction in dynamic robotic simulation using the principle of maximum dissipation," *International Workshop on the Algorithmic Foundations of Robotics*, 2010.

[6] E. Drumwright and E. Shell, "An evaluation of methods for modeling contact in multibody simulation," *ICRA*, 2011.

[7] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," *IROS*, 2012.

[8] A. Chatterjee and A. Ruina, "A new algebraic rigid body collision law based on impulse space considerations," *Journal of Applied Mechanics*, 1998.

[9] R. Featherstone, *Rigid Body Dynamics Algorithms*. Springer, 2008.

[10] ——, "Efficient factorization of the joint-space inertia matrix for branched kinematic trees," *International Journal of Robotics Research*, 2005.

[11] F. Udwadia and R. Kalaba, "A new perspective on constrained motion," *Proceedings of the Royal Society*, 1992.

[12] J. Baumgarte, "Stabilization of constraints and integrals of motion in dynamical systems," *Computer Methods In Applied Mechanics And Engineering*, 1972.

[13] I. Mordatch, E. Todorov, and Z. Popovic, "Discovery of complex behaviors through contact-invariant optimization," *SIGGRAPH*, 2012.

[14] K. Lowrey, Y. Tassa, T. Erez, S. Kolev, and E. Todorov, "Physically-consistent sensor fusion for contact-rich behaviors," *manuscript under review*, 2014.