Implicit nonlinear complementarity: A new approach to contact dynamics

Emo Todorov

Applied Mathematics and Computer Science & Engineering

University of Washington

ICRA 2010

Multi-joint dynamics with contact

- Continuous-time dynamics (may not have solution):

$$
\begin{aligned}
M(\mathbf{q})\, d\mathbf{w} &= \mathbf{n}(\mathbf{q},\mathbf{w})\, dt + K(\mathbf{q})^T \mathbf{f} \\
K(\mathbf{q})\, \mathbf{w} &= \mathbf{v}
\end{aligned}
$$

$\mathbf{q}, \mathbf{w}$	joint position and velocity
$\mathbf{f}, \mathbf{v}$	contact impulse and velocity
M	joint-space inertia matrix
$\mathbf{n}$	Coriolis, centripetal, gravitational, applied forces
K	contact Jacobian

$\mathbf{f}, \mathbf{v}$ are related through (an approximation to) the laws of contact and friction

- Discrete-time dynamics (always have solution):

Euler discretization with time step Δ yields $d\mathbf{w} \approx \mathbf{w}_{t+\Delta} - \mathbf{w}_t$,

$$
\begin{aligned}
M_t \mathbf{w}_{t+\Delta} &= M_t \mathbf{w}_t + \Delta \mathbf{n}_t + K_t^T \mathbf{f}_{t+\Delta} \\
K_t \mathbf{w}_{t+\Delta} &= \mathbf{v}_{t+\Delta}
\end{aligned}
$$

LCP formulation of contact

We need to solve

$$\begin{aligned} M\mathbf{w} &= \mathbf{c} + K^T\mathbf{f} \\ K\mathbf{w} &= \mathbf{v} \end{aligned}$$

Find $\mathbf{f}, \mathbf{v}$ by solving

$$\boxed{A\mathbf{f} + \mathbf{v}_0 = \mathbf{v}}$$

$A = KM^{-1}K^T$: inverse inertia in contact space
$\mathbf{v}_0 = KM^{-1}\mathbf{c}$: contact velocity in the absence of contact force

Compute $\mathbf{w}$ as

$$\mathbf{w} = M^{-1}\left(\mathbf{c} + K^T\mathbf{f}\right)$$

$\mathbf{f} = \left[f^N; \mathbf{f}^F\right]$ and $\mathbf{v} = \left[v^N; \mathbf{v}^F\right]$ should satisfy the constraints

Complementarity

$$f^N \geq 0, \quad v^N \geq 0, \quad f^N v^N = 0$$

$\mathbf{v}^F$ parallel to $\mathbf{f}^F$

$$\left\langle \mathbf{v}^F, \mathbf{f}^F \right\rangle \leq 0, \quad \left\| \mathbf{f}^F \right\| \leq \mu f^N$$

The latter constraints are nonlinear, however the friction cone can be approximated with a n-sided pyramid, yielding a linear complementarity problem (LCP).

Widely used: ODE, PhysX, Havoc...

Problems with LCP and motivation for our method

- The approximation to the friction cone is inaccurate for small n
- Large n results in too many auxiliary variables that slow down the solver
- Available algorithms are either slow (Lemke) or introduce additional approximations often resulting in spring-damper-like behavior

- The best general-purpose algorithm for solving LCPs is the PATH algorithm, which replaces the LCP with a nonlinear equation (and solves it using a non-smooth Newton method)
- If we are going to replace the LCP with a nonlinear equation, do we need the LCP in the first place? Or can we construct a nonlinear equation directly, without approximating the friction cone and introducing auxiliary variables? Yes we can.

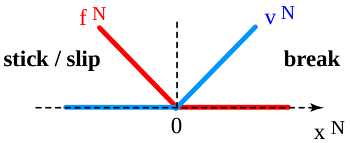
Implicit nonlinear complementarity

Instead of solving $A\mathbf{f} + \mathbf{v}_0 = \mathbf{v}$ under complementarity constraints on $\mathbf{f}, \mathbf{v}$, we design functions $\mathbf{f}(\mathbf{x}), \mathbf{v}(\mathbf{x})$ such that the constraints are satisfied for all $\mathbf{x}$, and then solve the (unconstrained) nonlinear equation

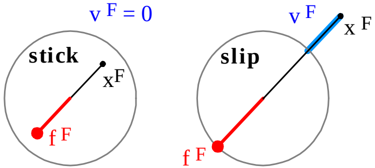
$$A\mathbf{f}(\mathbf{x}) + \mathbf{v}_0 = \mathbf{v}(\mathbf{x})$$

$\mathbf{x}$ is a hybrid variable encoding both contact velocities and contact forces.

normal forces and velocities:



friction forces and velocities:



$\mathbf{v}(\mathbf{x}) = \mathbf{f}(\mathbf{x}) + \mathbf{x}$, thus the (non-smooth) equation becomes

$$\mathbf{r}(\mathbf{x}) \triangleq (A - I)\,\mathbf{f}(\mathbf{x}) - \mathbf{x} + \mathbf{v}_0 = 0$$

The functions f and v

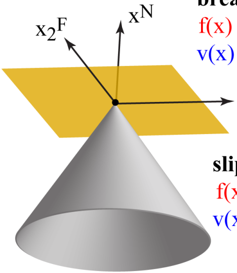
normal forces and velocities:

$$\begin{aligned} f^N(\mathbf{x}) &= \max\left(0,\ -x^N\right) \\ v^N(\mathbf{x}) &= \max\left(0,\ x^N\right) \end{aligned}$$

friction forces and velocities:

$$\begin{aligned} s(\mathbf{x}) &\triangleq \min\left(1,\ \frac{\mu f^N(\mathbf{x})}{\|\mathbf{x}^F\|}\right) \\ \mathbf{f}^F(\mathbf{x}) &= -s(\mathbf{x})\,\mathbf{x}^F \\ \mathbf{v}^F(\mathbf{x}) &= \mathbf{x}^F - s(\mathbf{x})\,\mathbf{x}^F \end{aligned}$$

3D forces and velocities:



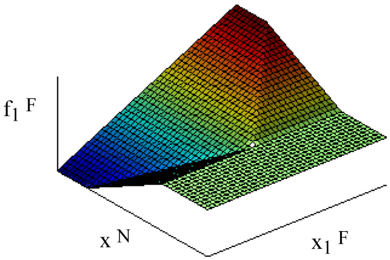
break:
f(x) = 0
v(x) = x

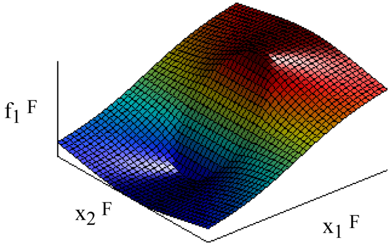
slip:
f(x) = - S(x) x
v(x) = x - S(x) x
S = diag(1, s, s)

stick:
f(x) = - x
v(x) = 0

Shape of the function f

$$\mathbf{f}^F(\mathbf{x}) = -\min\left(1, \frac{\mu \max\left(0, -x^N\right)}{\|\mathbf{x}^F\|}\right)\mathbf{x}^F$$





Root-finding via optimization

Solving $\mathbf{r}(\mathbf{x}) = 0$ is equivalent to minimizing the objective function

$$\ell(\mathbf{x}) = \frac{1}{2} \|\mathbf{r}(\mathbf{x})\|^2$$

This is a non-linear least squares problem, which (in principle) can be handled by a Gauss-Newton method:

$J(\mathbf{x}) = \dfrac{\partial \mathbf{r}(\mathbf{x})}{\partial \mathbf{x}}$	Jacobian/subdifferential of $\mathbf{r}(\mathbf{x})$
$J(\mathbf{x})^T \mathbf{r}(\mathbf{x})$	gradient of $\ell(\mathbf{x})$
$J(\mathbf{x})^T J(\mathbf{x})$	approximate Hessian of $\ell(\mathbf{x})$

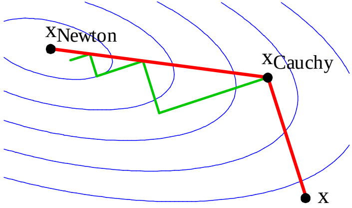
The (stabilized) Newton iteration is

$$\mathbf{x} \leftarrow \mathbf{x} - \left(J(\mathbf{x})^T J(\mathbf{x}) + \lambda I\right)^{-1} J(\mathbf{x})^T \mathbf{r}(\mathbf{x})$$

λ is adapted online in Levenberg-Marquardt fashion.

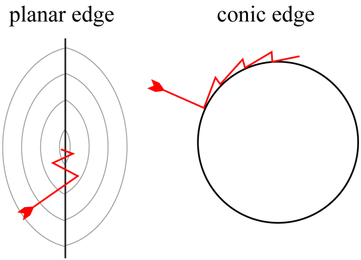
Optimization with edge-aware linesearch

Second-order methods avoid the chattering characteristic of first-order methods (green):



Here we use the "dogleg" method which involves two linesearhes (red). The Cauchy point is the minimum along the gradient.

Non-smoothness can still cause chattering:

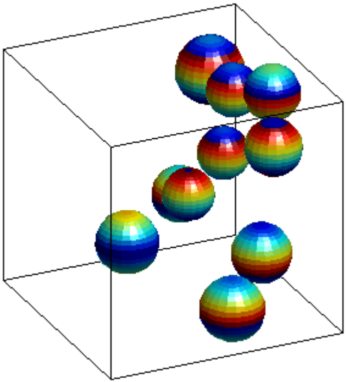


We explicitly consider the intersections of the red lines with the edges where $\mathbf{r}(\mathbf{x})$ is non-smooth (planes and cones).

Numerical results

With $\mu = 1$ and $n_c = 16$ contacts the algorithm takes 5 iterations per time step (no warm start) and accepts the Newton point (without linesearch) on 70% of iterations.

Each iteration here is faster than an iteration of an LCP solver because we are factorizing smaller matrices: $3n_c$ -by- $3n_c$ as opposed to, say, $10n_c$ -by- $10n_c$.



μ	$n_b = 5$, $n_c = 7$	$n_b = 10$, $n_c = 16$	$n_b = 15$, $n_c = 27$
0.1	3.8 99 %	4.8 98 %	6.5 90 %
0.5	2.6 95 %	5.3 85 %	7.5 73 %
1.0	2.8 90 %	4.9 71 %	10.2 60 %
2.0	2.9 88 %	4.6 71 %	16.2 55 %