Implementation of

<u>Staggered Projections for Frictional Contact in Multibody systems</u>

in card house

Authors: Danny M. Kaufman, etc.

Presenter: Lake Chen and Yuchen Luo

Outlines

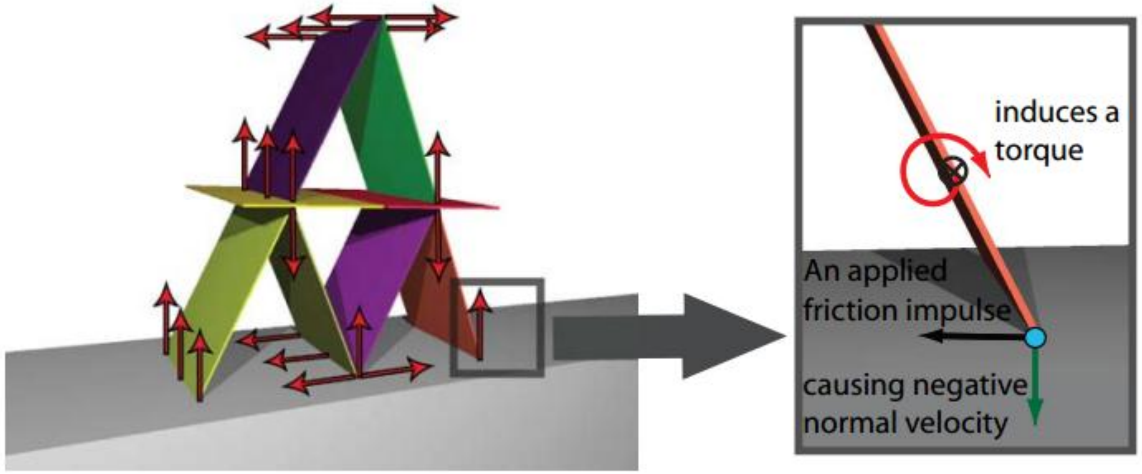
- Introduction
- Previous Work
- New Method
- Important Equations
- Test Case and analysis
- Result and Discussion
- Workload Distribution
- Summary

Introduction

- Grand Challenges in rigid body simulation

1. Jitter free
2. Stacking of multiple objects
3. Friction Prediction

- Friction Prediction is the topic covered in this presentation



Previous Work

- Penalty-based Method--- stiffness, stability issues
- Linear Complementarity Programming (LCP)-based Approach

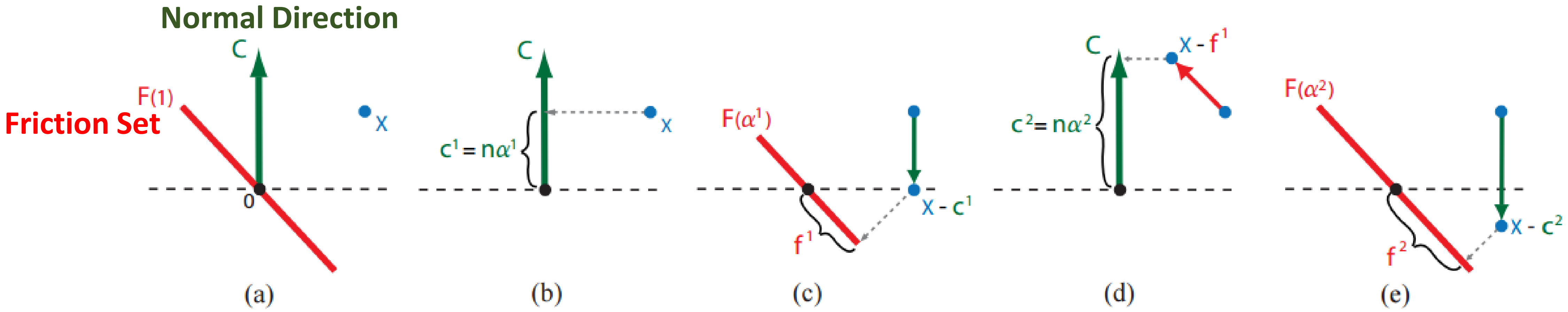
  1) acceleration-level: NP-hard

  2) velocity-level : non-convex

  3) iterative LCP : errors, artifacts and lots of iterations

  4) direct LCP : long computation time

New Method---
Velocity-based Staggered Projection Method



(a) Predicted Velocity: X, without constrain

(b) Project the x-f^0 to obtain normal impulse

(c) Project the x-c^1 on friction set and obtain f^1

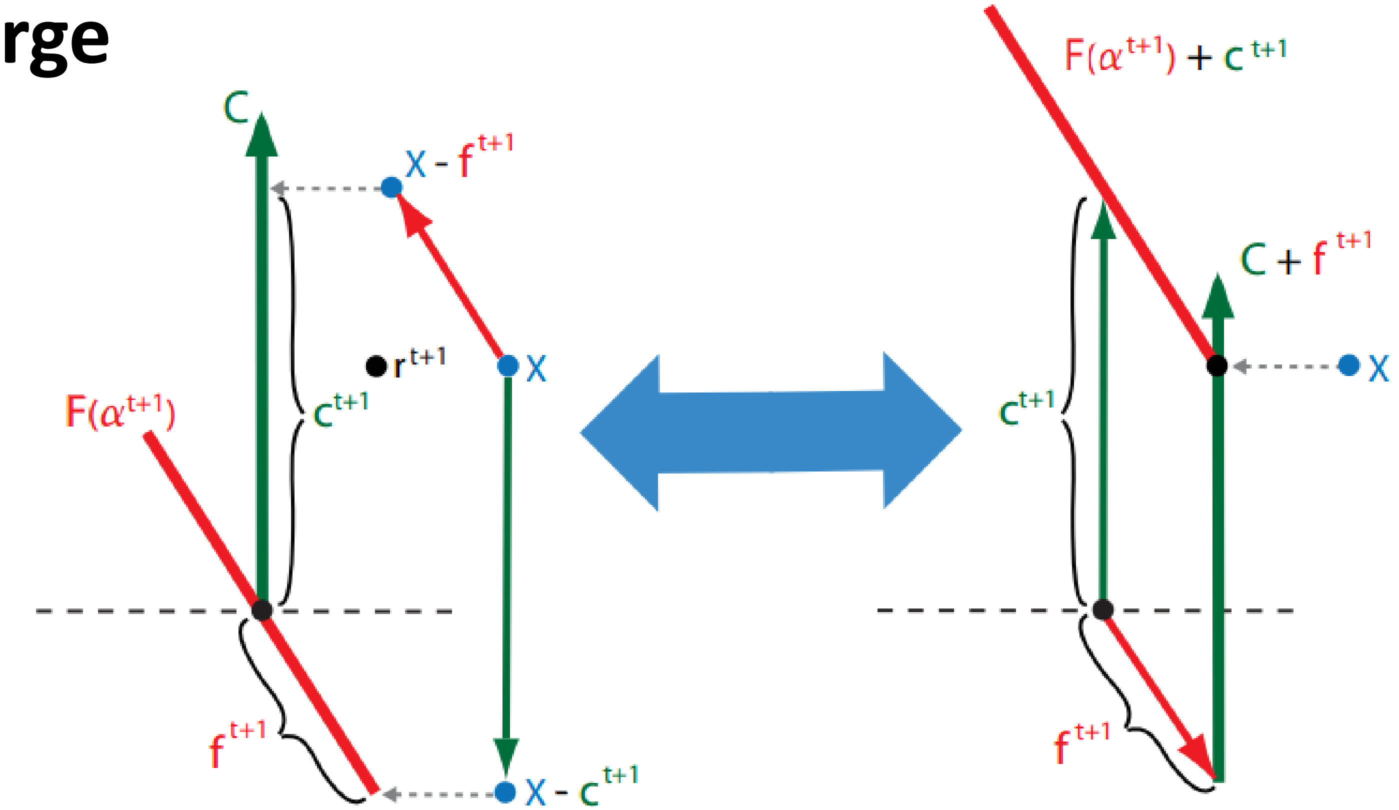
(e) Project the x-f^1 to obtain normal impulse c^2

(e) Project the x-c^2 on friction set and obtain f^2

$c^{t+1} = P_C(x - f^{t+1})$ $\quad f^{t+1} = P_{F(\alpha^{t+1})}(x - c^{t+1})$

New Method(continue)---
Velocity-based Staggered Projection Method

Till converge



Important Equations

Friction:

$$\beta^{t+1} = \underset{\beta}{\operatorname{argmin}}\left(\beta^T \mathsf{D}^T \dot{\mathsf{q}}^{t+1} \;:\; E^T \beta \le \operatorname{diag}(\mu)\alpha^{t+1},\ \beta \ge 0\right)$$

Normal :

$$\dot{\mathsf{q}}^{t+1} = \underset{\mathsf{u}}{\operatorname{argmin}}\left(\frac{1}{2}\mathsf{u}^T \mathsf{M} \mathsf{u} - \mathsf{u}^T\left(\mathsf{M}\dot{\mathsf{q}}^p + \mathsf{D}\beta^{t+1}\right) \;:\; \mathsf{N}^T \mathsf{u} \ge 0\right)$$

$$\mathsf{P}_{\mathsf{S}}(\mathsf{v}) \overset{\text{def}}{=} \underset{\mathsf{u} \in \mathsf{S}}{\operatorname{argmin}}\ (\mathsf{u}-\mathsf{v})^T \mathsf{M}^{-1}(\mathsf{u}-\mathsf{v})$$

$$\mathsf{f}^{t+1} = \mathsf{P}_{\mathsf{F}(\alpha^{t+1})}(\mathsf{x} - \mathsf{c}^{t+1}),$$

$$\mathsf{c}^{t+1} = \mathsf{P}_{\mathsf{C}}(\mathsf{x} - \mathsf{f}^{t+1}).$$

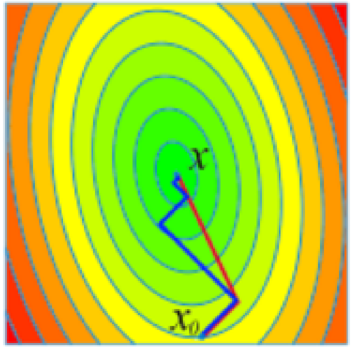
$$\mathsf{f}^{i+1} \leftarrow \mathsf{P}_{\mathsf{F}(\alpha^{i})}(\mathsf{x} - \mathsf{N}\alpha^{i}),$$

$$\mathsf{N}\alpha^{i+1} \leftarrow \mathsf{P}_{\mathsf{C}}(\mathsf{x} - \mathsf{f}^{i+1}).$$

Important Equations(continue)---Quadratic Programming

$$P_S(v) \stackrel{\text{def}}{=} \underset{\boxed{u \in S}}{\operatorname{argmin}} \boxed{(u-v)^T M^{-1} (u-v)}$$

constrain Target function



Quadratic programming is a subfield of nonlinear optimization which deals with quadratic optimization problems subject to optional boundary and/or general linear equality/inequality constraints:

$$\min_x \left[\frac{1}{2} x^T A x + b^T x\right] \quad \text{subject to:}$$

1) $l \leq x \leq u \quad l, x, u \in R^N$

2) $Cx \circ d \quad$ where $C \in R^{KxN}$, $d \in R^K$, $\circ$ is an arbitrary combination of $\leq = \geq$

Framework Pseudocode

```
initialize constant-size matrix → q̇, q̇_p, f, c, M…
while(1){
        contact detection → contacts (store body index, vertex index, sdf source index)
        update inertia matrix → M (change rotational inertia)
        update contact matrix → N, D, α, β
        assemble overall impulse matrix using list of contacts → N, D
        predict velocity → q̇_p
        while (not converged or exceed maximum iteration){
                call quadratic programming solver to get impulse magnitude → α, β → c, f
        }
        update velocity using impulse → q̇
        update system position
}
```

$\dot{q}$: system velocity
$\dot{q}_p$: system predicted velocity
f: system friction impulse
c: system normal impulse
M: system inertia matrix
N: generalized normal impulse
D: generalized frictional impulse
α: normal impulse magnitude
β: frictional impulse magnitude

Test Case

- Collision Detection and Response
- Normal Impulse
- Frictional Impulse
- Stacking
- Card House with No Friction
- Card House with Friction

Workload Distribution

- Equation Derivation(Yuchen and Lake)
- Quadratic Programming (Yuchen)
- Code framework (Lake)
- Debug and test cases (Yuchen and Lake)
- Slides (Yuchen and lake)
- Post Processing Animation (Lake)
- Present (Yuchen and Lake)

Summary

Table 1: Schedule of Milestones of the Project

Date	Milestone	Description
Oct. 27 2016	Project start	Start creating program architecture.
Nov. 03 2016	Complete program skeleton	All classes and methods are created. Animation scene and setup completed.
Nov. 10 2016	Implement normal force	Implement normal force interaction: Stack of cubes in rest.
Nov. 21 2016	Implement single friction	Implement solution that resolves friction on a single body: Cube in slope.
Nov. 29 2016	Implement multiple friction	Implement solution that resolves frictions on multiple bodies: card house.
Dec. 6th 2016	Mass splitting jitter free method implemented	Modify the finished staggered projection method to increase convergence speed
Dec. 13th 2016	Final solution implementation and presentation	Finalize all the code and optimize the efficiency prepare for the presentation.