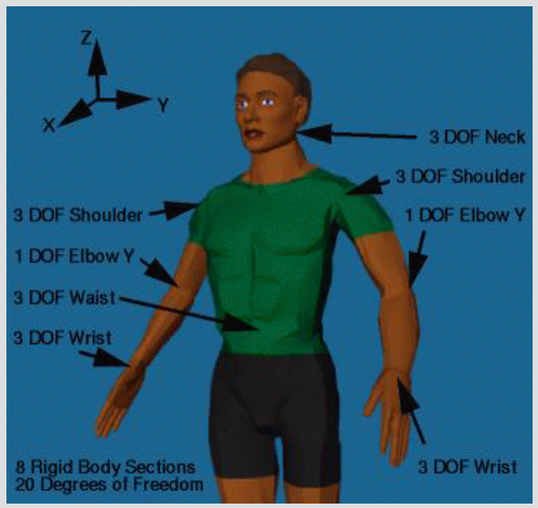
Workshop introduction

A short history of robotic simulation

My introduction to robotic simulation (2001)



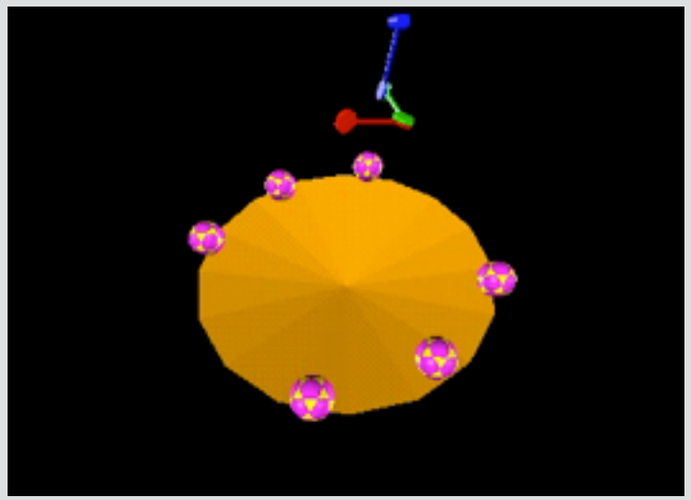
"Adonis dances the Macarena"

Simulated using SD/FAST, then MathEngine (now Vortex), then Open Dynamics Engine

M. Matari´c, V. Zordan, and M. Williamson. Making complex articulated agents dance: an analysis of control methods drawn from robotics, animation, and biology. Autonomous Agents and Multiagent Systems, 2(1):23–44, March 1999.

1. Joints came apart, and it was hard to do many of the kinds of robotic kinematics and control techniques I was learning about
2. The key differentiator of ODE, etc. from previous engines (to the user) is innate ability to deal with contact.

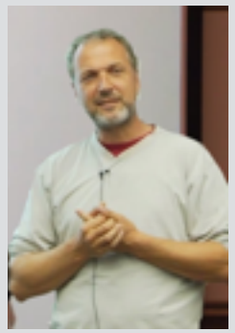
My thought: robotic simulation just needed good software development.



Impulse (B. Mirtich)

The assumption: the solutions are known, just need good software development to implement them
Impulse: no joints pulling apart

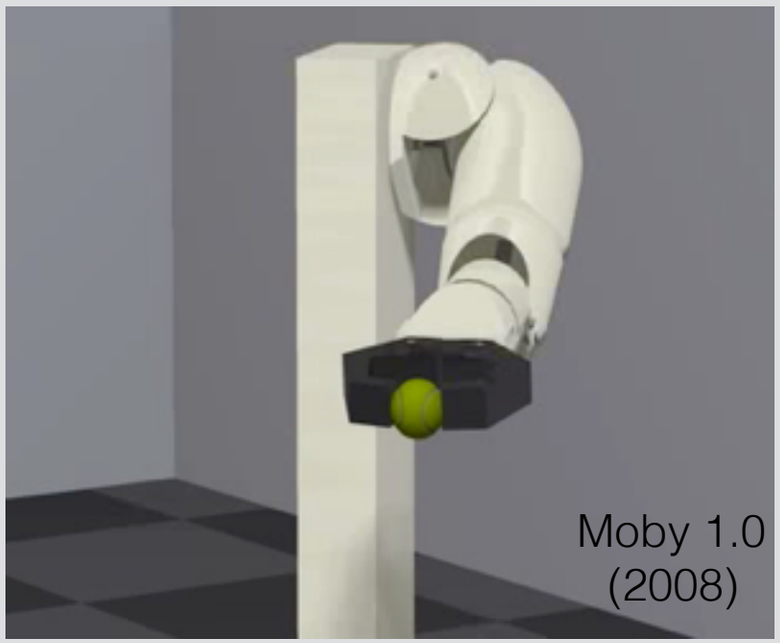
My first foray into to robotic simulation: penalty methods.



Don’t write a simulator!

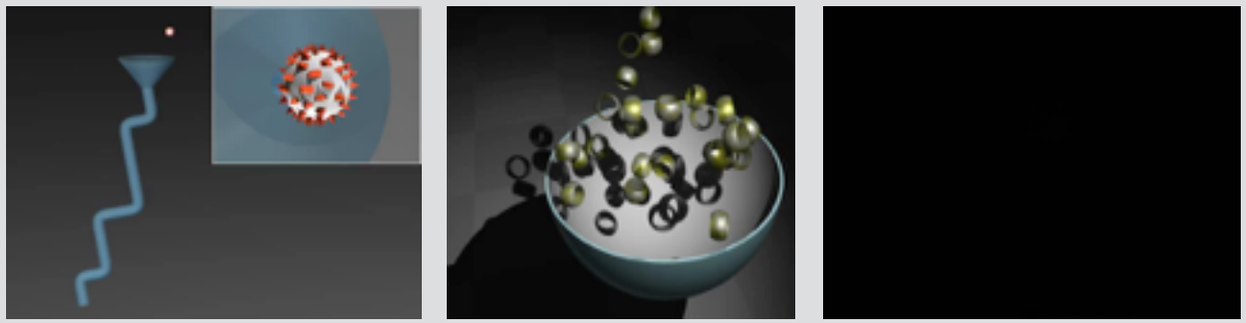
After this experience, I thought there was considerable room for improvement.
Crummy PR2 simulation from my first simulator.

Hard contacts made my simulations faster and robotic grasping more robust.



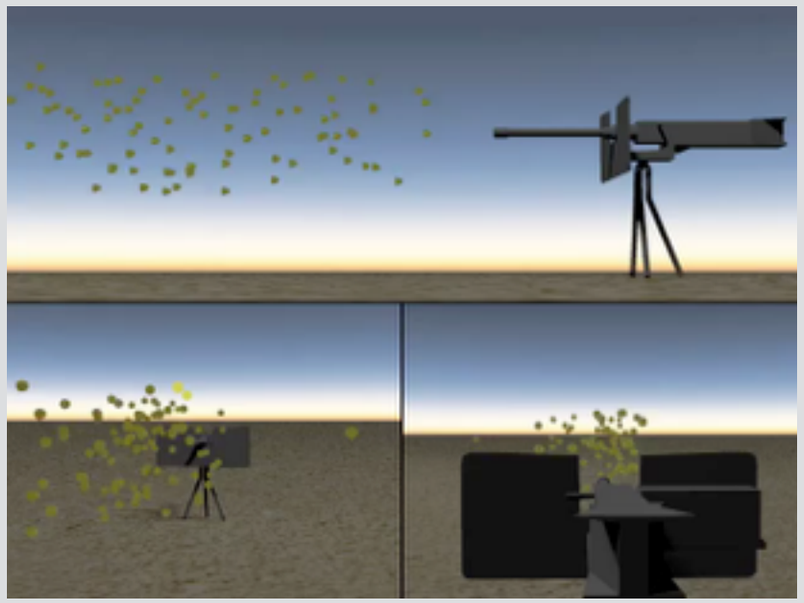
Considered robotic grasping within simulation solved.

My next few years in simulation focused on fast simulation of non-convex and thin-shelled geometries…



(Joint work with Dylan Shell)

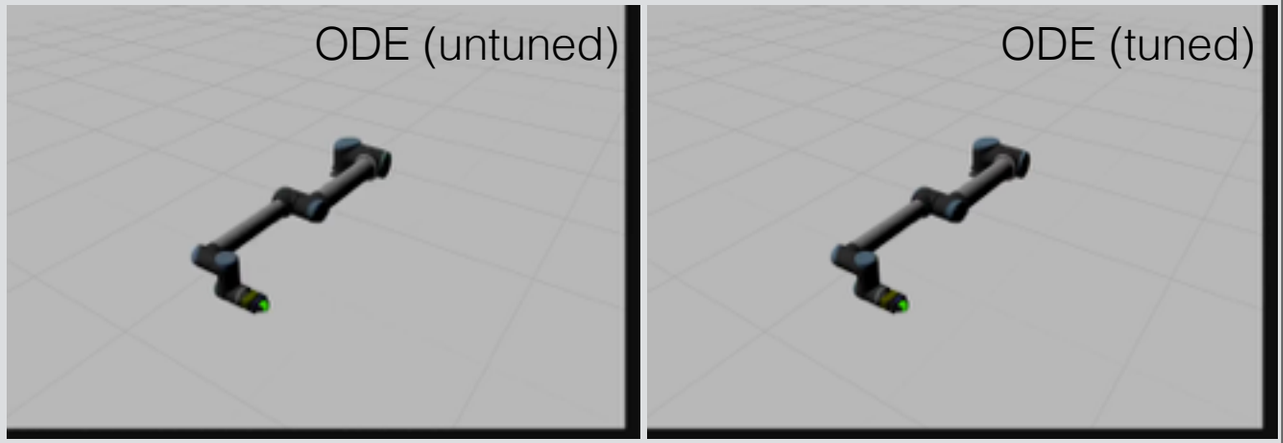
…and preventing “tunneling”.



(Joint work with Dylan Shell)

Segue: my original assumptions were challenged.

DRC showed that simulation parameters can be tuned to yield desired performance, even under (presumably) reduced accuracy.



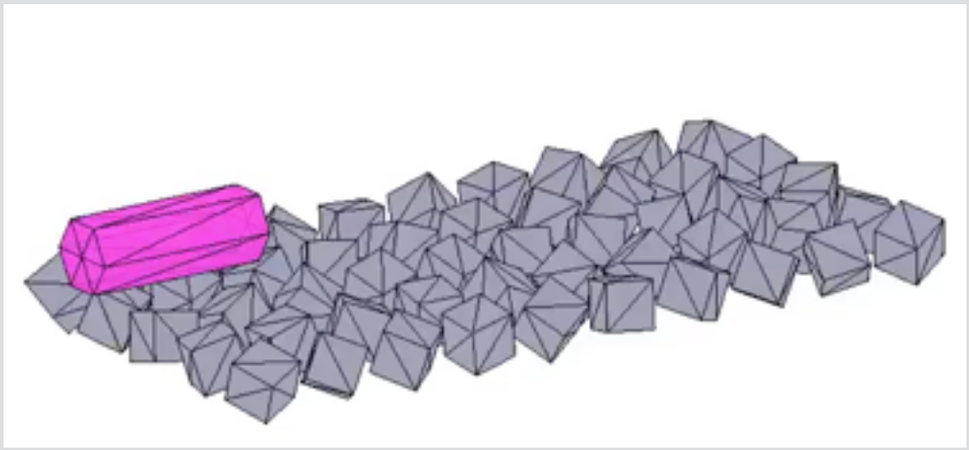
J. R. Taylor, E. M. Drumwright, and J. Hsu. Analysis of grasping failures in multi-rigid body simulations. In Proc. SIMPAR, 2016.

Robotics community continues to push in direction of faster simulation rates.

> “Accuracy is nice. But what I really want is *fast* simulations.”
>
> “I prefer inaccurate simulations. That ensures my controllers will be robust when I move them to real robots.”
>
> “When you do simulation, you’re doomed to succeed.”

* These improvements often come from approximating solutions to models.
* We don’t have a complete understanding of the tradeoffs.
* We don’t have a piece of software that we can trust to produce solutions.

You might expect rigid body simulation software to crash when bugs occur.



$\dot{e} = G\omega$ $\quad G = \begin{bmatrix} -e_x & e_w & -e_z & e_y \\ -e_y & e_z & e_w & -e_x \\ -e_z & -e_y & -e_x & e_w \end{bmatrix}$

1. The flip side of this observation is that even inaccurate simulations can be stable.
2. “Given enough eyeballs, all bugs are shallow.” —Linus Torvalds