"Toward High-Performance Computing Support for the Analysis, Simulation, and Planning of Robotic Contact Tasks," a workshop at Robotics Science and Systems, June 27, 28 2011, held on the campus of the University of Southern California.

Supported by NSF Grant: CRI: CP-I: 0855024

Executive Summary of NSF Workshop

The workshop was opened by Jeff Trinkle, who articulated the goals to the participants. The main goal of the first day was to establish an understanding of the state of the art in the following subareas:

- Simulation capabilities and support functions
- Future needs in robotics, engineering, games, etc.
- Guidelines for model/method selection
- Issues in high-performance computing

The main goal of the second day was to discuss the needs of the robotics in terms of a computational simulation tool and support functions. Three break-out groups would be formed to discuss the following subtopics:

- Software base and user support
- Hardware and platforms
- Validation and benchmarks

In a final wrap-up meeting on the second day the issues and recommendations of the three break-out groups were discussed with additional consideration given to the possibility of applying for an NSF Community Infrastructure – Acquisition, Development, Deployment, and Operation (CI-ADDO) grant.

The conclusions of the second day's discussions were:

- It is premature to develop an NSF-supported community software resource
- A longer, larger workshop (perhaps through BIRS) involving the mechanics and numerical methods communities is in order
- A repository should be developed that contains contact problems with known analytical solutions and instances of time-stepping subproblems that are difficult to solve

State-of-the-art update from EuroMech Colloquium 516

The week after this workshop, Trinkle attended EuroMech Colloquium 516 in Grenoble France, July 6-8, 2011. Two simulation tools were presented that define the state of the art in rigid body simulation: Extended Dynamics Engine (XDE) by Xavier Merlhoit. XDE is free for academic use, but closed-source. The most compelling case for a paradigm shift in multibody dynamics simulation tools was Algoryx Simulation, by Claude Lacoursiere. The simulation engine is based on a variational integrator formulation derived from augmented Lagrangians (http://www.physics.umu.se/english/research/statistical-physics-and-networks/complex-mechanical-systems/realtime-wire-simulation/). Unfortunately this simulator is closed-source and not free for academic use. The most capable open-source physics engines are Siconos, (http://siconos.gforge.inria.fr/) developed at Bernard Brogliato's group at INRIA, Grenoble, and Solfec, (http://code.google.com/p/solfec/) developed by Tomasz Koziara's group at University of Glasgow. The former has been used for very challenging hair simulations with thousands of hairs and tens of thousands of contacts. The latter has been developed for shared-memory computing systems. Simulations with thousands of polyhedral and tens of thousands of contacts have been solved on problems with up to 128 CPUs. The friction law is not compromised in either of these simulators. The idea of a repository discussed in the workshop in Los Angeles has already been implemented by Vincent Acary (see https://gforge.inria.fr/projects/fclib).

List of attendees:

Organizers:
Chris Carothers, chrisc@cs.rpi.edu
Dan Negrut, negrut@wisc.edu
Jeff Trinkle, trinkle@gmail.com
Invited Speakers:
Erwin Coumans, erwin.coumans@gmail.com
Danny Kaufman, kaufman@cs.columbia.edu
Kevin Lynch, kmlynch@northwestern.edu
Dinesh Manocha, dm@cs.unc.edu
Todd Murphey, t-murphey@northwestern.edu
Dinesh Pai, pai@cs.ubc.ca
Siddhartha Srinivasa, siddh@cmu.edu

Poster Presenters:
Dan Goldman, daniel.goldman@physics.gatech.edu
Beatriz Leon, beatriz.leon@icc.uji.edu
Hammad Mazhar, hmazhar@wisc.edu
Thorsten Schindler, thorsten.schindler@mytum.de
Dylan Shell, dshell@cse.tamu.edu

Other Particpants:
Srinivas Akella, sakella@uncc.edu,
John Hsu, johnnhsu@willowgarage.com
Abhi Jain, abhi.jain@jpl.nasa.gov,
Matt Mason, matt.mason@cs.cmu.edu
Rasmus Tamstorf, Rasmus.Tamstorf@disneyanimation.com
Yuval Tassa, yuval.tassa@gmail.com
Tom Vose, tomvose@gmail.com

hfvonbremen@csupomona.edu
t3wu@ucsd.edu
Marco.B.Quadrelli@jpl.nasa.gov
amin.atrash@gmail
subhrajit@gmail.com
cdellin@cmu.edu
dharmon@cs.nyu.edu
etom@uw.edu

<u>Appendix A:</u>
<u>Raw Notes from workshop "Toward High-Performance Computing Support for the Analysis, Simulation, and Planning of Robotic Contact Tasks"</u>

<u>Jeff's notes during talks on first day: June 27, 2011:</u>
Erwin Coumans 9:10 – 9:35
- Data standards are important to follow.
- OpenCL is to support GPU programming.
- CUDA is not industry-wide.
- Contact generation is coupled with dynamics.
- Represents bodies as many points.
- What are the main approximations you make?
- What is coming?
- How do you parallelize LCP?

Sidd Srinivasa 9:44 – 10:09
- Why is manipulation hard?
- Representation of C-Space.
- Failures in grasping. Localization error is the problem.

Dan Negrut 10:17 – 10:42
- Mihai and Allesandro 2008 – friction relaxation to write as a cone complementarity problem
- What about existence with the CCP?
- What about comparison with real measurements/experiments?
- Projected Gauss Jacobi method has issues with convergence.

Kevin Lynch 1:32 – 1:57
- Sliding assumption breaks down near plate's center and far from center.
- How could you benefit from a physics engine?
- Inverse problems, i.e. design system inputs for specified outcome using simulation.

Dinesh Manocha 2:00 – 2:25
- 98% of computation time in motion planning is collision checking.
- They would prefer to use distance and penetration, but it is too computationally intensive.
- Narrow passage can be helped by incorporating dynamics like Nilanjan IROS 2008.
- What about the PPU of PhysX? The PPU is dead.
- Everyone should release their code.

Todd Murphey 2:33 – 2:58
- How do you get exact solutions?
- Sim 20, others
- Linearization data
- Under what circumstances can you believe the simulation?
- His method in TREP exactly conserves constraints. How is this possible?

Dinesh Pai 3:34 – 3:59
- Must have deformation in real systems.

Compute volumes of penetrations. The guys at the CSA (Yves (blanking on last name) and Christian Lange do this in their simulation code).

Contact friction not yet implemented.

Danny Kaufman 4:05 – 4:30

Models do not predict the system behavior accurately.

Accurate and robust algorithms are needed.

Start from core principles, e.g., Hamilton's principle.

Chris Carothers 4:38 – 5:03

Day 2, morning only: June 28, 2011:
Meet 8:30 – 9:45 at the patio outside food court.
Reconvene 9:45 – 11:00 in Room 101

Break-out Group Topics
- Hardware/Software/Scalability
- Set up a community to further cause of
- Benchmarking, problem repository, validation

Chris' notes: Hardware and Software Issues

Attendees: Srinivas Akella, Tom Vose, Erwin Courmans, Sid, Chris Dellin, Ramus Tamstorf, Dinesh Manocha and Chris Carothers.

Supercomputer vs one's own Linux cluster.
- Millions of particles (i.e., grains of sand) is problem for SC platforms.

Desktop GPU engines. Physics Engines should be cross platform. We need one language that fits all – but we don't have it. Collada doesn't do it, it is not human-readable. We need to use wrappers to develop a general language.

Plans for Bullet engine -- make the code more re-usable.

Physics folks don't want to share high-level code. They only want to work with their own high-level view.

Build systems are a challenge because they need to be easy and pleasing all the users. Bullet supports five different build systems.

High-level API
Low-level -- sphere-shere
Base - core algorithms

Sid -- 90% of time is spent in collision algorithms for motion planning. We need a common repository to share data on collision detection of what works and does not. There can be legal issues (Flamingo example) in sharing data.

Onboard vs. off-board computations.
Vision processing is done over wireless in off-board fashion on GPU. Is it possible to pre-compute and store things, e.g., self collisions. Do near neighbors in high-dim spaces.

Today have to code for a multi-level architecture designs -- e.g., NUMA vs. GPU.

Codes that sample the hardware performance and pick the algorithm configuration that performs best is a good service to have -- e.g., Atlas does this.

For the community by a mailing list -- this group is a start. What about Facebook? -- some don't like end user terms of service.

Make results reproducible via data and simulation.

Make accurate simulation a part of the big NSF Robotics Program that is coming.

Create a "Contact GEMs" and "LCP Solver GEMS" book + website where the articles are 4 pages maybe more but put code with the article.

Todd's and Kevin's Notes: Application/User support

Todd's Notes:

1) Specific problems are useful, but we don't necessarily need to dictate what examples to use when we could instead let some examples filter to the top as being important.

2) The scale of benchmarks matters. We should have some benchmarks that really highlight the low-level problems that can occur when simulating mechanical systems (like the mass examples that have catastrophic outcomes in simulation of impact). Other benchmarks could focus more on larger-scale problems involving many masses.

3) What should we compare to? Two natural possibilities are comparing to experiments and comparing to analytic or nearly analytic benchmark solutions. We should distinguish between ensemble characteristics of a simulation and individual trajectories as well, since different applications have different needs in this regard.

4) We need to distinguish between situations where tuning parameters are acceptable and situations where they are not.

5) A repository of standardized problems should include the ability to test subroutines, like collision detection, LCP solvers, etcetera.

6) For funding we should keep in mind who the "customers" will be for careful numerical techniques. Manufacturing is probably a good focus point.

7) Lastly, we may wish to use Wikipedia as an example of how we should share information.

Kevin's Notes:

Breakout session notes, June 27 2011, from workshop "Toward High-Performance Computing Support for the Analysis, Simulation, and Planning of Robotic Contact Tasks"

Important distinction: "correct" or "highly accurate" simulation of a mathematical model of contact and impact (Kevin: this is what I want), whether the model is continuous or discrete, vs. simulations that predict the real world (i.e., the model must also be right). Keep clear the distinction between models of the world (lots of impact and friction models out there, compliant or not, etc.) and the implementation of a particular model. (All software "correctly" implements itself; the software itself is not the mathematical model.) Note: I think Dinesh Pai later referred to this as verification (correct implementation) vs. validation (predicts the world), though that might conflict with the use of "verification" in "software verification."

Different questions we could ask:

* correct solution of a timestep or instantaneous acceleration (e.g., solving an LCP or NCP)
* simple systems but longer time interval of simulation (e.g., may require collision detection), and comparison of the simulation results to analytical solutions of the mathematical model (where such solutions can be found)
* reproducing qualitative large-scale behavior, like the squares in layers of vibrated grains Danny Kaufman showed, reproducing experimental results by Melo, Umbanhowar, and Swinney (note: Dan Goldman says these features are actually quite robust to different models)

could classify benchmark problems by the pieces of the simulator involved (e.g., collision detection or contact point detection may not be needed for some problems)

Could make a community-supported benchmark website with the following items:

1) collision detection problems (download the scenario, test it on your algorithm)
2) LCP problems corresponding to contact scenarios (i.e., maybe not just a random collection of LCP's, but ones that arise from interesting contact problems)
3) results of real experiments so simulation folks can connect their simulation results to real experiments. Example: Goldman's work dropping spheres into grains and using an accelerometer to measure the impact accelerations; the squares observed by Melo et al.; Vose's vibrating plate
4) a set of pathological models that break some simulators (e.g., the simple scenario of two connected balls inside a ring shown by Kaufman)
5) a set of scenarios with analytic solutions for particular mathematical models

If the website reached a critical mass, testing your algorithm on these examples could become a standard and perhaps even required for publishing in some venues.

could categorize problems by type (intermittent contacts/impacts or not, number of DOFs, constraints or not, flexible materials or not, etc.) and benchmark metrics could be computation time, accuracy, ...

Many people don't want to release their code, as it's their bargaining chip with companies, etc. (Negrut). But may be willing to make it available for academic use, perhaps a Berkeley license.

Are there any analogies to ROS? If there is a simulation code base that people can contribute to and build from, will it catalyze efforts toward open code and better benchmarking? Or is simulation too immature for this, and we should focus just on benchmarking? Perhaps the benchmarking site can have links to simulation software rather than trying to maintain a code base. Perhaps community moderated. Perhaps a wikipedia page. Would help if the open source software had some instruction materials.

Would be a big contribution if we could standardize problem specification, so people could share scenarios. Negrut suggested using the ADAMS specification, bulletproofed by decades of development. Collada came up in our convergence meeting, with people complaining that it is too complicated. Negrut said he could talk to people at ADAMS about making their problem specification parser public.

Proposal/funding: could latch on to the open manufacturing initiative (e.g., simulation for manufacturing), or new robotics initiative. perhaps an NSF virtual community, or cyber-enabled discovery and innovation. Can we find a good analogy to hitch to, to show why a community-supported benchmarking site or code repository would have significant impact? ROS? Need industry behind it. Grand Challenges? make it a program instead of a proposal?

John Hsu's notes:

Discussions on database with problem+solution data sets and benchmark data.

Some implementation Issues:

Some problem sets need multiple states (in time) for warm starts.

Need to be easy to participate for different audiences with different LCP solvers. Data sets need to be easy to parse, or provide parsers (translation tools) for different popular simulator formats.

Solutions can range from analytical solution to experimental data to macroscopic system behaviors, transient vs. single instance in time.

One idea is to implement test cases as a "Blooper's Reels". Show pathological examples (and solutions) for collisions and lcp examples.

Scope - possibly start by building a database of problems+solutions with a narrower scope in robotics?

Previous efforts - What similar attempts if any exists? Some mentions of prior database examples that might have lost steam / funding over time:

pdb.org?

cpnet?

Funding and Maintenance - How to maintain funding, interest and maintenance, so project does not run out of steam and stays useful.

Lawyer stuff - Potential legal issues with sensitive data sets (should stick to "opensource" data?), require some form of disclosure disclaimer?

Jeff's notes on group discussion after recombining the break-out groups:

Build a repository for problem instances. Format should be simple.

Efficient problem set description language is needed. Standardize the problem specification.

Time series of solutions and problems makes it too hard initially.

Stick to rigid bodies initially.

Constraint functions, complementarity problems, function representation that includes minimization and complementarity problems. What solution did they get? Here's the outputs from other solvers.

Such a database will help the multibody dynamics community bridge to the optimization community.

Collada is not human readable. URDF – universal robot description format is human readable. It is Willow's home-built description language.

Delimit the scope of what the web page will take – scope, real-time or not, rigid or not.

Model our effort on successful similar past efforts. Linear alg efforts – LaPACK. Protein DB. In these efforts, there was a backer with funds

Within community we need to communicate. The repository should help achieve this.

Which problems take unreasonably large amounts of time to solve? Is there a relationship?

What do you want to predict from models

Requests from roboticists. They want two knobs for the simulation – stability and speed. ODE tweaked by Schaal to work?

Super computer vs linux
- Desktop GPU – should be cross platform
- One language that fits all
- Produce a suite of contact "gems"
- Multiple build systems, repository, on-board vs off-board
- Make results more reproducible.
- Adopt what Adams uses as a file format? (Dan) .adm files.
- Maybe Matlab .mat files,

Benchmarking (Todd and Kevin)
- Ensemble statistics
- What to benchmark – pathological cases that break algorithms
- Calibration – should you be allowed to calibrate
- How would we share code?
- Wikipedia articles on FEM
- Wiki page on rigid body dynamics
- Can we leverage the push in manufacturing that Obama pushed?
- What are the metrics?
- Std LCP and CD sites
- Dan – 5 components of use: LCP matrix and vector, pathological cases, bench mark problems