Differentiable Simulation Methods for Robotic Agent Design

Tao Du

MIT CSAIL
Robots are ubiquitous.

- Agriculture (DJI)
- Warehouse (Boston Dynamics)
- Marine study (MIT)
- Prosthetics (UPMC)
- Manipulation (MIT)
Building the best robot in the world

The best robot requires the best body and brain combination.
Building the best robot in the world

The best robot requires the best body and brain combination.

**Body**
- Morphology
- Topology
- Actuation
- ...

**Brain**
- Perception
- Planning
- Control
- ...
Building the best robot in the world

We have seen lots of breakthroughs on the brain side.

**Body**
- Morphology
- Topology
- Actuation
- ...
Building the best robot in the world

However, almost everything on the body side is still solved manually.

THOR (SUTD)

Professional photography
(Netflix documentary)
Robot design: still driven by humans

Consider designing a flying robot...

The design space exhibits diversity and is highly unstructured.
Traditional pipeline

The process depends heavily on human expertise.
Traditional pipeline

The process depends heavily on human expertise.
Traditional pipeline

The process depends heavily on human expertise.
The process depends heavily on human expertise.
The process depends heavily on human expertise.
Traditional pipeline

The process depends heavily on human expertise.
Reality: tedious and error-prone process
We propose computational robot design pipelines.
We propose computational robot design pipelines.
Scope of this talk

We focus on exploring continuous parameters only.
Scope of this talk

We focus on exploring continuous parameters only.
Contributions

A computational pipeline that jointly optimizes shape and control
Contributions

Advanced differentiable simulators for complex physics
Contributions

A solution to sim-to-real problems using differentiable simulation
Part I: Computational Robot Co-design
Contributions

A computational pipeline that jointly optimizes shape and control.
The key player: differentiable simulation

Recap: physics simulation
- States: $s(t)$, e.g., positions and velocities of rigid bodies.
- Actions: $a(t)$, e.g., external forces or torques.
- System parameters: $\theta$, e.g., mass of rigid bodies, viscosity of fluids.
The key player: differentiable simulation

Recap: physics simulation
- States: $s(t)$, e.g., positions and velocities of rigid bodies.
- Actions: $a(t)$, e.g., external forces or torques.
- System parameters: $\theta$, e.g., mass of rigid bodies, viscosity of fluids.
The key player: differentiable simulation

Recap: physics simulation
- States: \( s(t) \), e.g., positions and velocities of rigid bodies.
- Actions: \( a(t) \), e.g., external forces or torques.
- System parameters: \( \theta \), e.g., mass of rigid bodies, viscosity of fluids.
The key player: differentiable simulation

Differentiable simulation
- Imagine you have a loss defined on the final state.
The key player: differentiable simulation

Differentiable simulation

- Imagine you have a loss defined on the final state.
- By flipping each arrow, you get gradients.
The key player: differentiable simulation

Differentiable simulation

- Imagine you have a loss defined on the final state.
- By flipping each arrow, you get gradients.
- With gradients, we can optimize the loss.

\[
\frac{\partial L}{\partial \theta} \quad \frac{\partial L}{\partial s_0} \quad \frac{\partial L}{\partial a_0}
\]

\[
\theta \quad \text{Sim} \quad s_0 \quad s_1 \quad L
\]
Contributions

A computational pipeline that jointly optimizes shape and control
Contributions

A computational pipeline that jointly optimizes shape and control

Continuous Design Parameters

Shape Design

Controller Design

DiffSim

Experiment
Computational Multicopter Design

Tao Du, Adriana Schulz, Bo Zhu, Bernd Bickel, Wojciech Matusik

ACM SIGGRAPH Asia 2016
The key player: differentiable simulation
Results: maximizing payload

Original design

Optimized design
Behind the scene: source of efficiency

Old payload: 1047g

New payload: 1392g
Behind the scene: source of efficiency

Old payload: 1047g
New payload: 1392g

The power of computational methods: finding **blind spots** in human experts and discovering novel morphology.
Part II: Advanced Differentiable Simulation
Contributions of my thesis

Two advanced differentiable simulators for complex physics
DiffPD: Differentiable Projective Dynamics

Tao Du, Kui Wu, Pingchuan Ma, Sebastien Wah, Andrew Spielberg, Daniela Rus, and Wojciech Matusik

ACM TOG (accepted with minor revisions)
Motivation

Computational design needs quick simulation and gradients.
But this is difficult for soft bodies due to their too many degrees of freedom.
Motivation

Computational design needs quick simulation and gradients. But this is difficult for soft bodies due to their too many degrees of freedom.

People know how to speed up forward simulation. Lots of SIGGRAPH papers are about this topic (e.g., model reduction, projective dynamics, and new data structures).
Motivation

Computational design needs quick simulation and gradients. But this is difficult for soft bodies due to their too many degrees of freedom.

People know how to speed up forward simulation. Lots of SIGGRAPH papers are about this topic (e.g., model reduction, projective dynamics, and new data structures).

Can we borrow these techniques to speed up backpropagation?
DiffPD in a nutshell

The high-level idea in one page

Implicit time integration = Finding a root of this nonlinear system:

\[ x_{i+1} - h^2 M^{-1} f(x_{i+1}) = x_i + h v_i \]
DiffPD in a nutshell

The high-level idea in one page

Implicit time integration = Finding a root of this nonlinear system:

\[ x_{i+1} - h^2 M^{-1} f(x_{i+1}) = x_i + hv_i \]

Solving it requires an iterative solver linearizing the system frequently (slow).
DiffPD in a nutshell

The high-level idea in one page
Implicit time integration = Finding a root of this nonlinear system:
\[ x_{i+1} - h^2 M^{-1} f(x_{i+1}) = x_i + hv_i \]
Solving it requires an iterative solver linearizing the system frequently (slow).
Backpropagation solves a linear system with the same matrix.

\[ \begin{bmatrix} I - h^2 M^{-1} \frac{\partial f}{\partial x_{i+1}} \end{bmatrix} \]

\[ \frac{\partial L}{\partial x_i} \rightarrow x_i \rightarrow \text{Sim} \rightarrow x_{i+1} \rightarrow L \rightarrow \frac{\partial L}{\partial x_{i+1}} \]
DiffPD in a nutshell

The high-level idea in one page

Implicit time integration = Finding a root of this nonlinear system:

\[ x_{i+1} - h^2 M^{-1} f(x_{i+1}) = x_i + hv_i \]

Solving it requires an iterative solver linearizing the system frequently \((\text{slow}).\)

Backpropagation solves a linear system with the same matrix.

**Idea for quick forward simulation -> quick backpropagation!**

\[ [I - h^2 M^{-1} \frac{\partial f}{\partial x_{i+1}}] \]

\[ \frac{\partial L}{\partial x_i} \quad x_i \quad \text{Sim} \quad \rightarrow \quad \frac{\partial L}{\partial x_{i+1}} \]
Benchmark tests

Does this idea work?

DoFs: 8019, time steps: 25, dt: 10ms, tol: 1e-4

DiffPD: 15.9s, Cholesky: 216s \(13.5x\), PCG: 157.5 \(9.9x\)
Examples: system identification

DoFs: 16776, time steps: 200, dt: 5ms, tol: 1e-4
Goal: optimize frictional coefficients so that the duck falls in the white circle
Examples: closed-loop control

<table>
<thead>
<tr>
<th>Initial guess</th>
<th>Optimized by DiffPD</th>
<th>Optimized by RL</th>
</tr>
</thead>
</table>

Goal: Optimize a neural network controller so that the shark swims.

**DiffPD**: 1h24m, RL: 31h33m *(22.3x)*, Newton’s Methods: 11h40m *(8.2x)*
Lesson: deriving gradients can be an art.

We learned the same lesson from differentiable fluid simulation too:

Part III:
Sim-to-real Transfer
Contributions

A solution to sim-to-real problems using differentiable simulation
Underwater Soft Robot Modeling and Control with Differentiable Simulation

Tao Du*, Josie Hughes*, Sebastien Wah, Wojciech Matusik, Daniela Rus

IEEE RA-L/RoboSoft 2021
Motivation: controlling a starfish robot

Challenges
- System identification with solid-fluid coupling.

Our core idea
- Use DiffPD to iteratively calibrate the system and optimize the motion.
Method: sim-real alternation

**Step 1:** Accept control signals from diffsim.

**Step 2:** Gather real-world data.

**Step 3:** System-ID with diffsim.

**Step 4:** Optimize the controller with diffsim.
Result: 4x performance improvement
Result: sim-to-real gap

After convergence, we have observed between sim and real:
- Similar horizontal offsets;
- Similar cyclical motion patterns.
Conclusions
Designing robots is still challenging.
Computers can help you co-design.
Computing gradients can be an art...

Projective Dynamics [Bouaziz S, Martin S, Liu T, Kavan L, Pauly M. SIGGRAPH 14]
...and it unlocks certain applications.
However, gradients have limitations too.

Topological changes are tricky to deal with.
Thank you for your attention!

Tao Du
MIT EECS and CSAIL

Homepage
http://people.csail.mit.edu/taodu

Email
taodu@csail.mit.edu

Project and paper links
Copter  DiffPD  DiffStokes  Starfish
Acknowledgments: image credits

Pg. 6
THOR: https://ieeexplore.ieee.org/document/7989755

Pg. 7
Microrobot: https://www.micro.seas.harvard.edu/research
Ingenuity: https://mars.nasa.gov/technology/helicopter/

Pg. 8
Human head clip art: http://4.bp.blogspot.com/_OEWNrlUA7Gc/S8cNqthAcal/AAAAAAAABxQ/R8vCvS3BVbY/s1600/Brain_training-1.jpg

Pg. 16

Pg. 61
Projective Dynamics: https://www.cs.utah.edu/~ladislav/bouaziz14projective/bouaziz14projective.html