Efficient Tactile Simulation with Differentiability for Robotic Manipulation

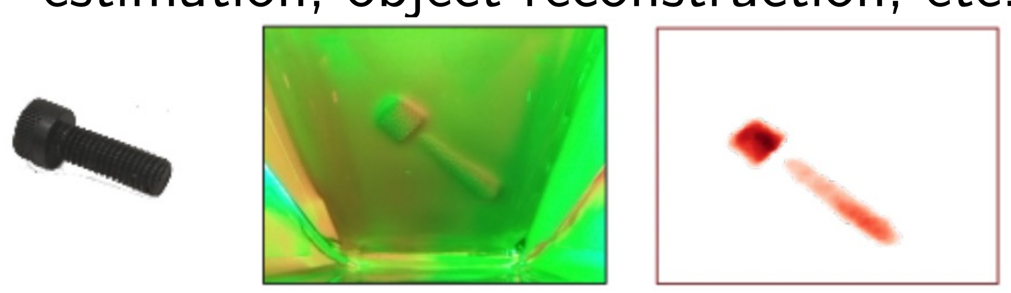
Jie Xu, Sangwoon Kim, Tao Chen, Alberto Rodriguez, Pulkit Agrawal, Wojciech Matusik, Shinjiro Sueda

Massachusetts Institute of Technology, Texas A&M University

http://tactilesim.csail.mit.edu/

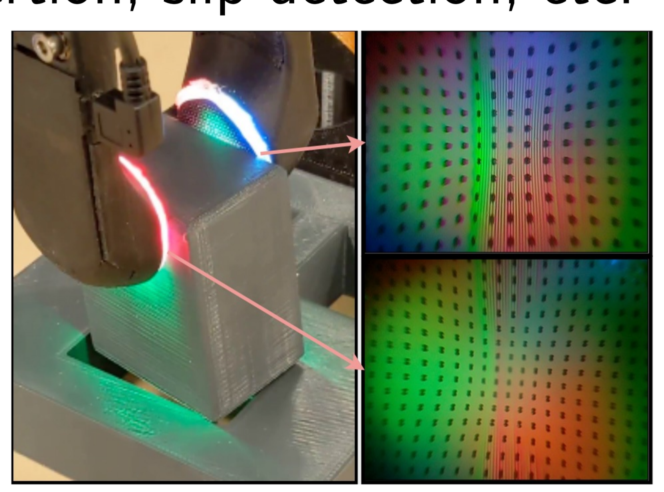
Motivation

- **Dense Tactile Feedback for Manipulation**
  - Tactile <u>Normal-Direction</u> Feedback
    - Static spatial relation.
    - Applications: edge following, pose estimation, object reconstruction, etc.



(Bauza et. al. 2019)

  - Tactile <u>Shear-Direction</u> Feedback
    - Dynamic tangential motions.
    - Applications: stable grasp, precise insertion, slip detection, etc.



(Kim, Rodriguez 2022)

- **However, the training process usually requires time-consuming and labor-intensive real hardware experiments.**

GOAL

Build an Efficient Tactile Simulation for Sim-to-Real Tactile-Based Robot Control

Our Approach

- **Efficient Tactile Simulation**
  - **Built upon DiffRedMax (Xu et. al. 2021), implemented in C++.**
  - **Tactile Sensor Representation:** each tactile sensing point i is represented as a tuple $\langle \mathrm{B}_i, \mathrm{E}_i, \xi_i \rangle$ (Fig. 1).

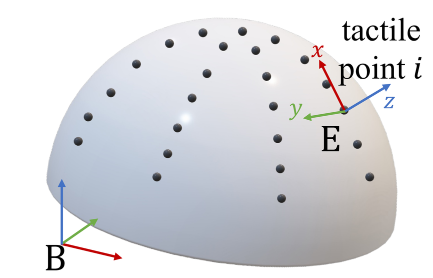


Fig.1: Tactile sensor representation

  - **Penalty-Based Tactile Model:**
    - <u>First step</u>: compute penalty-based tactile forces on tactile points.

$$\vec{\boldsymbol{f}}_n = (-k_n + k_d \dot{d}) d \vec{\boldsymbol{n}}, \qquad \vec{\boldsymbol{f}}_t = -\frac{\vec{\boldsymbol{v}}_t}{\|\vec{\boldsymbol{v}}_t\|} \min(k_t \|\vec{\boldsymbol{v}}_t\|, \mu \|\vec{\boldsymbol{f}}_n\|)$$

    - <u>Second step</u>: transform the forces into the tactile point frame.

$$T_{\{sx,sy,n\}} = \left(\vec{\boldsymbol{f}}_n + \vec{\boldsymbol{f}}_t\right)^{\mathsf{T}} \{\vec{\boldsymbol{x}}, \vec{\boldsymbol{y}}, \vec{\boldsymbol{z}}\}$$

  - **Features**
    - <u>Efficiency:</u> 1050 FPS for a ball-rolling experiment (Fig. 3) with 40Hz 200×200 tactile force field computation on a single core of Intel Core i7-9700K CPU.
    - <u>Arbitrary tactile sensor geometry layout:</u> specify any number of sensing points in arbitrary geometry layouts.
    - <u>Differentiability:</u> provide fast analytical first-order gradients for the entire dynamics chain (e.g. the gradients of the reward/loss w.r.t. policy parameters).
    - <u>User-friendly simulation interface:</u> C++ backend with Python frontend interfaces, simple configuration file format for simulation scene/robot descriptions.
- **Sim-to-Real via Normalized Tactile Flow Map**

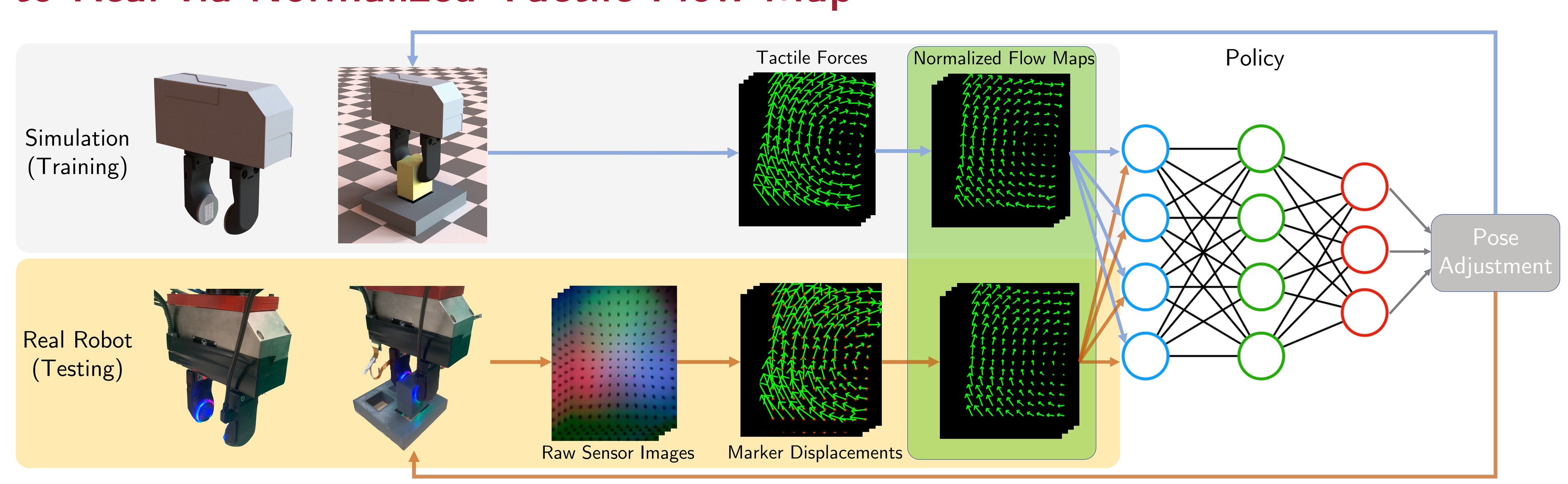


Fig.2: Sim-to-real pipeline

Experiments

- **Simulation Experiments**

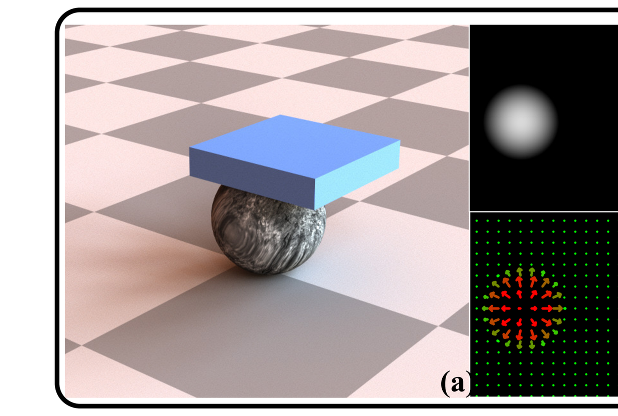


Fig.3: **Speed**: Ball Rolling Experiment

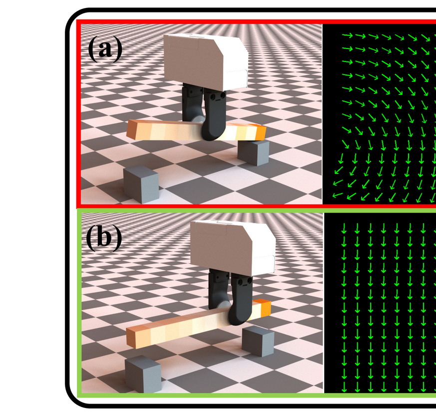


Fig.4: **RL Training**: Stable Grasp Task

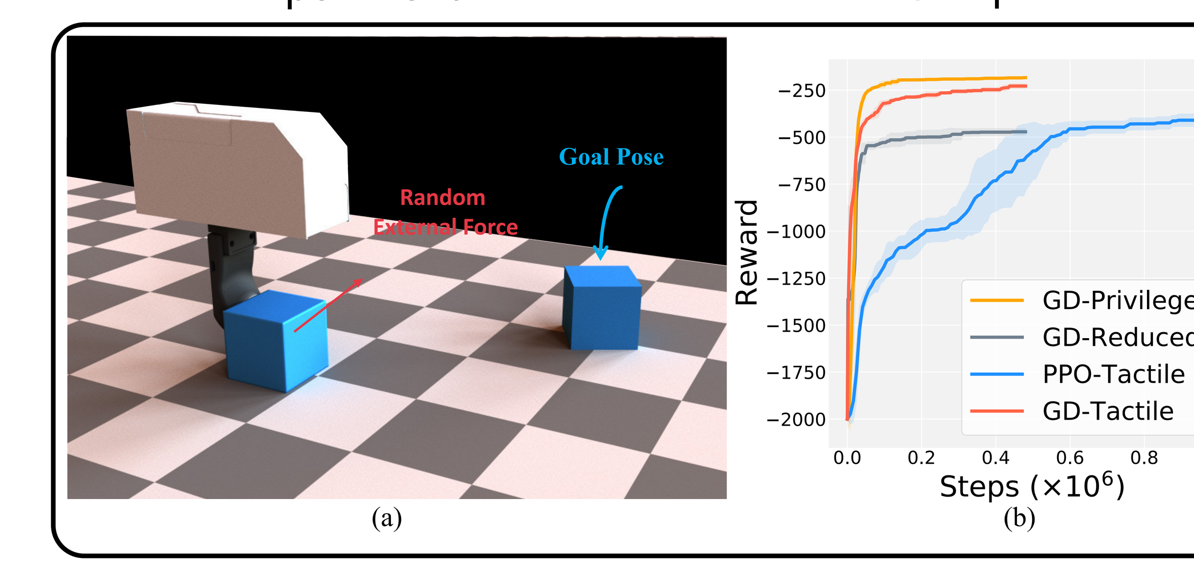


Fig.5 **Differentiability**: Tactile-Based Box Pushing Task

- **Zero-Shot Sim-to-Real Experiments: Tactile RL Insertion**
  - <u>Hardware</u>: 6-DoF ABB IRB 120 robot arm; WSG-50 parallel jaw gripper; GelSlim 3.0 tactile sensor.
  - <u>Domain randomization</u>: parameters, tactile readings.

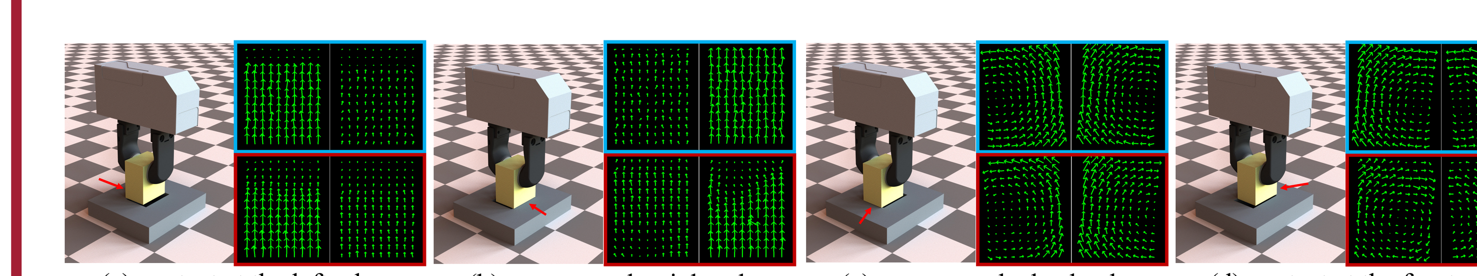


Fig.6: Comparison of the normalized tactile flow maps.

Rotation Only	Translation Only	Rotation+Translation
Succ: 100% Attempts: 1.53	Succ: 100% Attempts: 2.33	Succ: 83% Attempts: 4.81

Table 1: Zero-shot sim-to-real results

Dirty Laundry List

DOESN'T work for <u>very soft</u> tactile pad (e.g. TacTip).
- limited capability of penalty-based rigid-body dynamics.
- linear assumption between marker displacements and forces.

How to better leverage <u>differentiability</u> is challenging.
- local minimal problem.
- gradient explosion/vanishing.

<u>Sim-to-Real</u> is still NOT perfect.
- lower success rate on Rotation+Translation task than Dong et. al. 2021 (89.6%).
- generalizable policy for various object shapes?