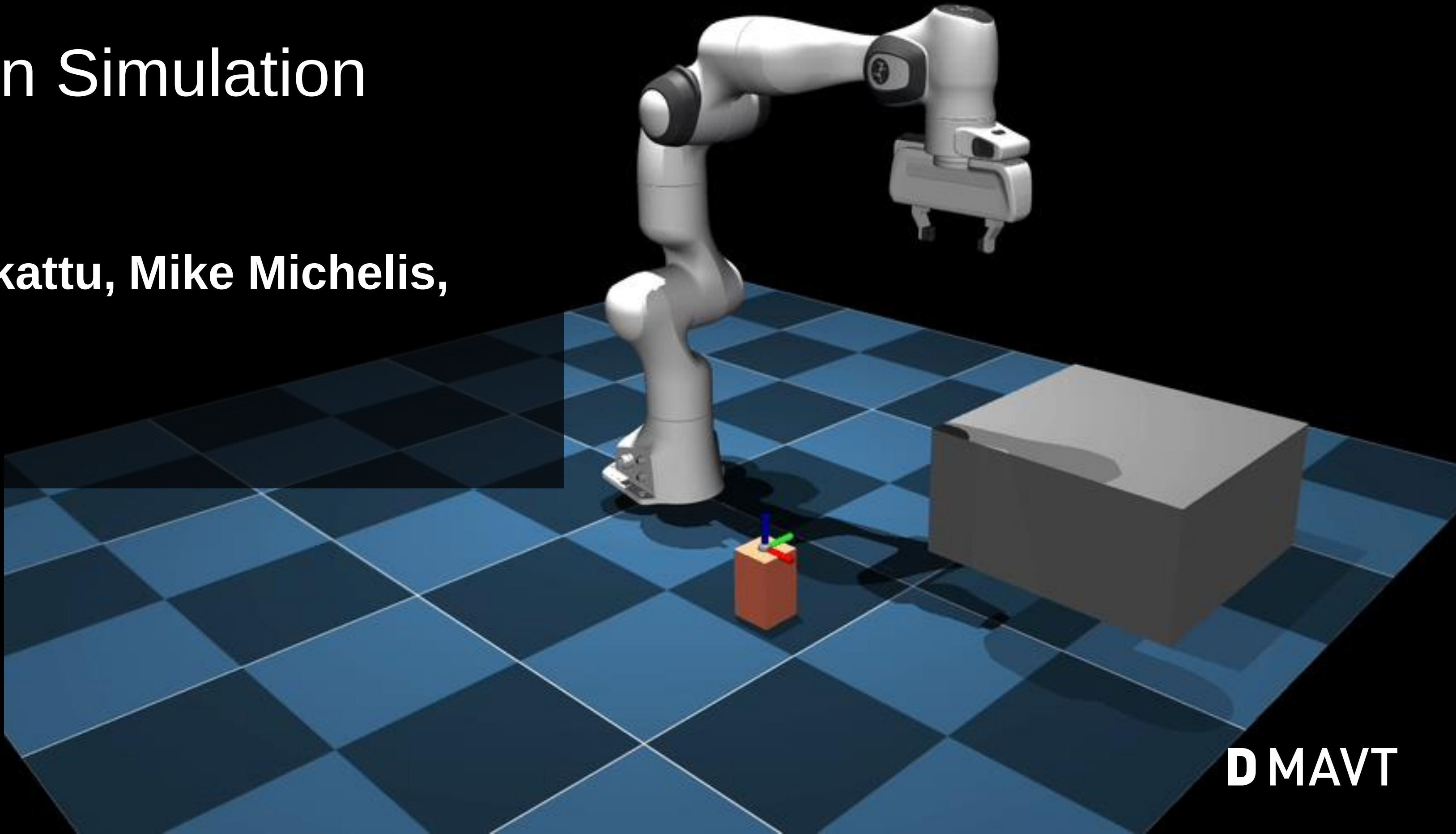
ETH zürich

Real World Robotics Course

Methods and Challenges in Simulation
Focus Talk Unit 5

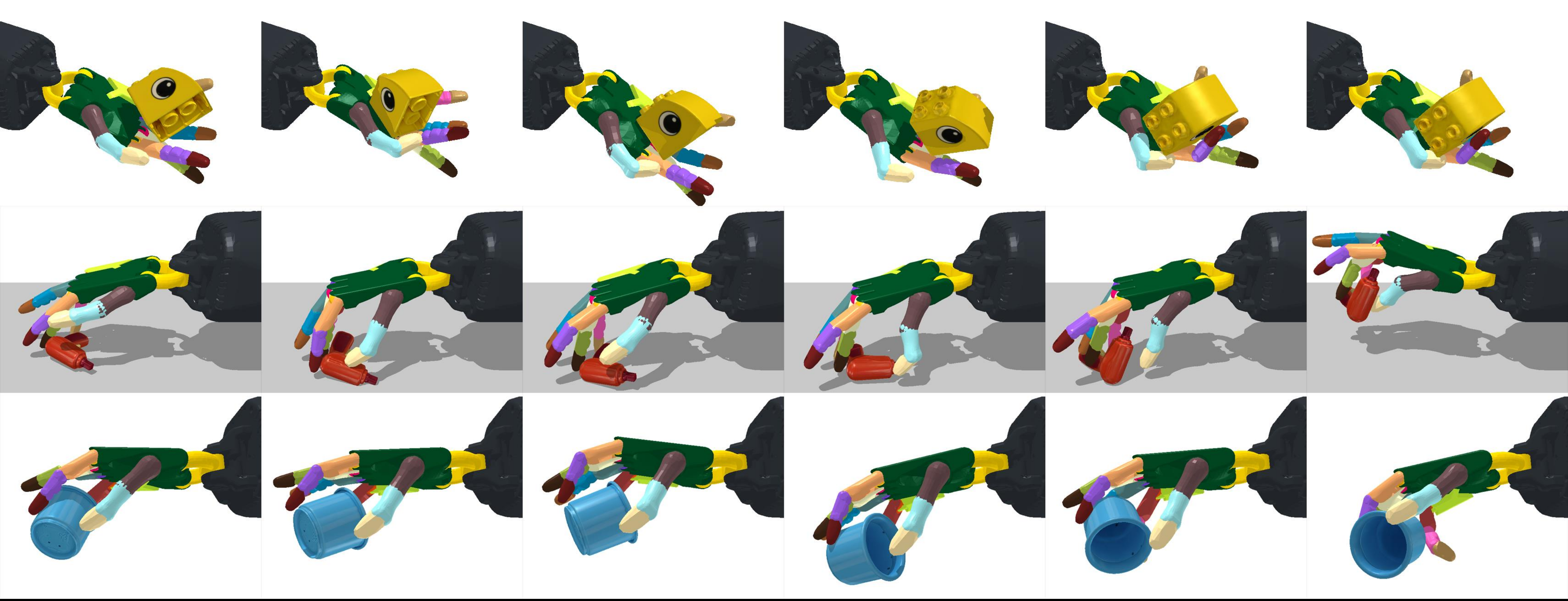


Yasunori Toshimitsu, Manuel Mekkattu, Mike Michelis, Robert Katzschmann
Soft Robotics Lab
ETH Zurich

Soft Robotics Laboratory

D MAVT

How to simulate dexterous object manipulation?



1. Chen, Tao, Jie Xu, and Pulkit Agrawal. "A system for general in-hand object re-orientation." Conference on Robot Learning. PMLR, 2022.
   https://taochenshh.github.io/projects/in-hand-reorientation

SoftRobotics Laboratory

Why are simulators important?

Why simulators?

***Safe* and *fast* environment for testing robot controllers**

Parallelized simulation for reinforcement learning (RL)

Model-based control based on simulators

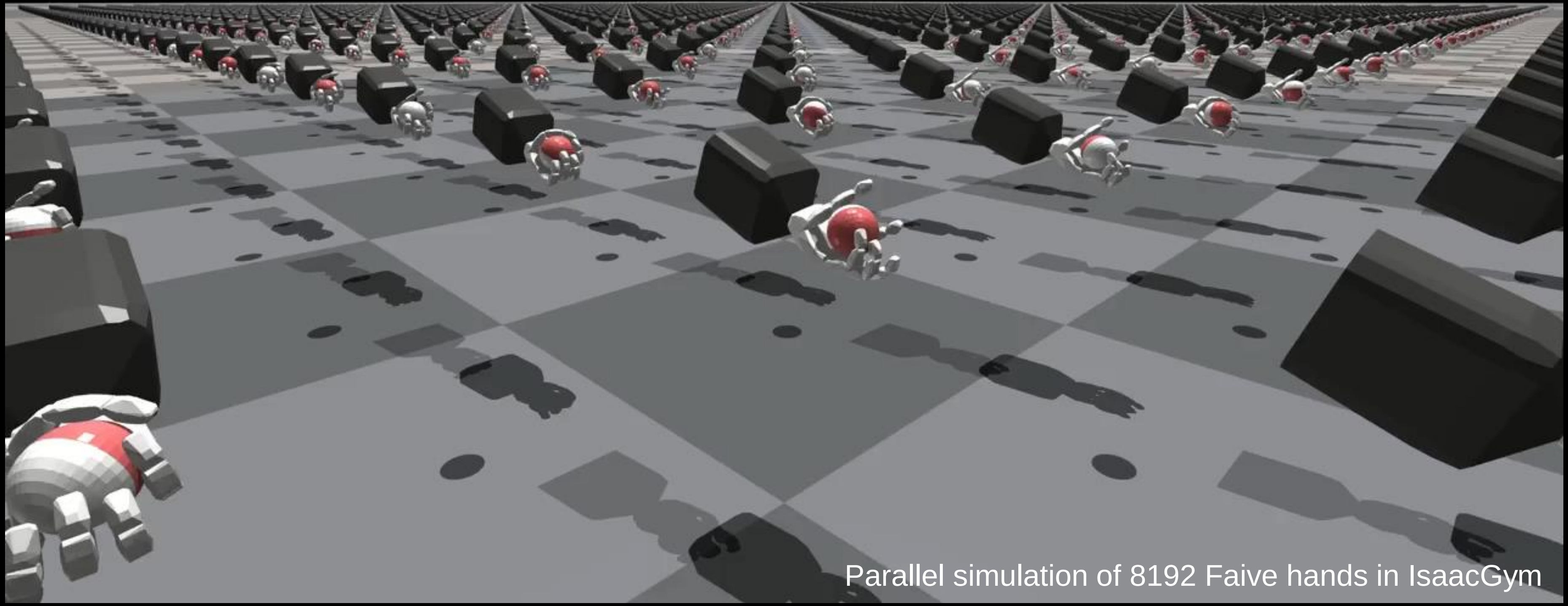
Failure in real environments are expensive

Darpa Robotics Challenge (DRC): https://www.darpa.mil/program/darpa-robotics-challenge

Simulations are safe environments

Darpa Robotics Challenge (DRC): https://www.darpa.mil/program/darpa-robotics-challenge

Parallelized Simulation for Reinforcement Learning (RL)



Parallel simulation of 8192 Faive hands in IsaacGym

Proximal Policy Optimization (PPO)

- Reinforcement Learning (RL) algorithm
- Scales well to parallel environments[3]

Domain Randomization[1]

- Randomize physics
- Add noise to observations
- Make it robust for physical deployment

Parallelized Simulation

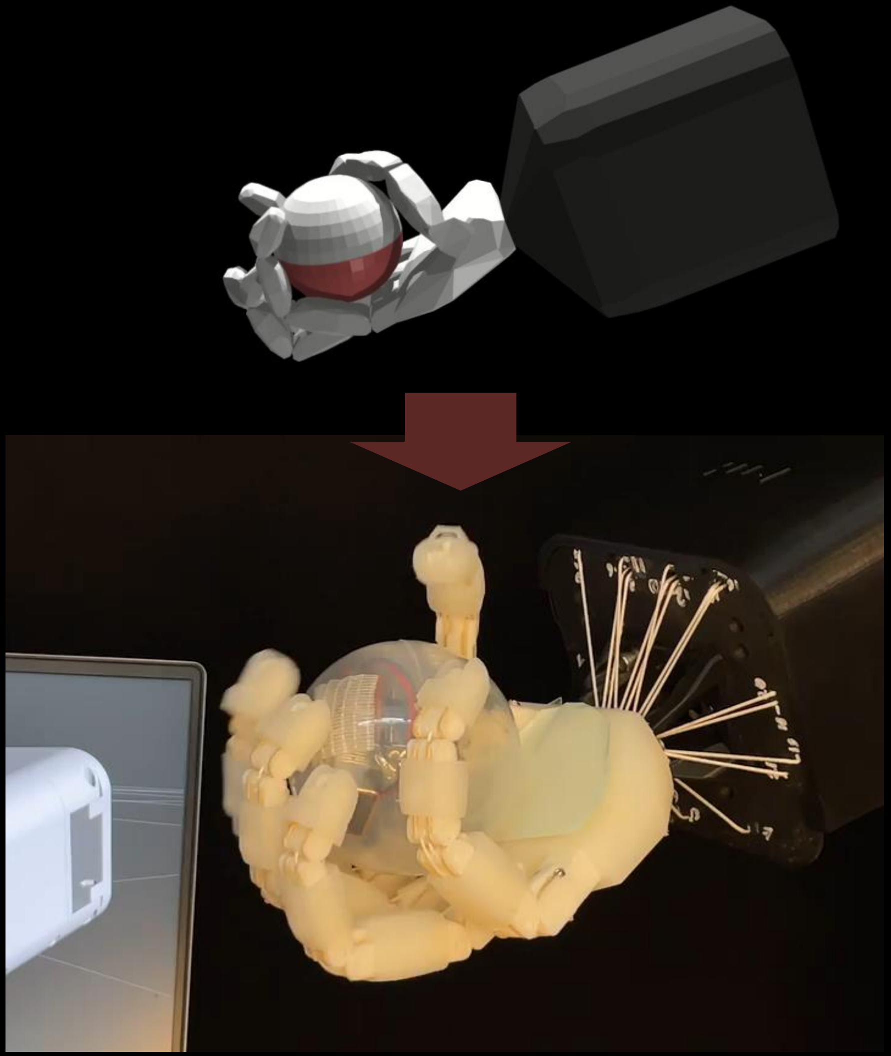
- 1000’s of robots in parallel on GPU[2,3]
- Wide exploration of initial conditions, parameters and control policies

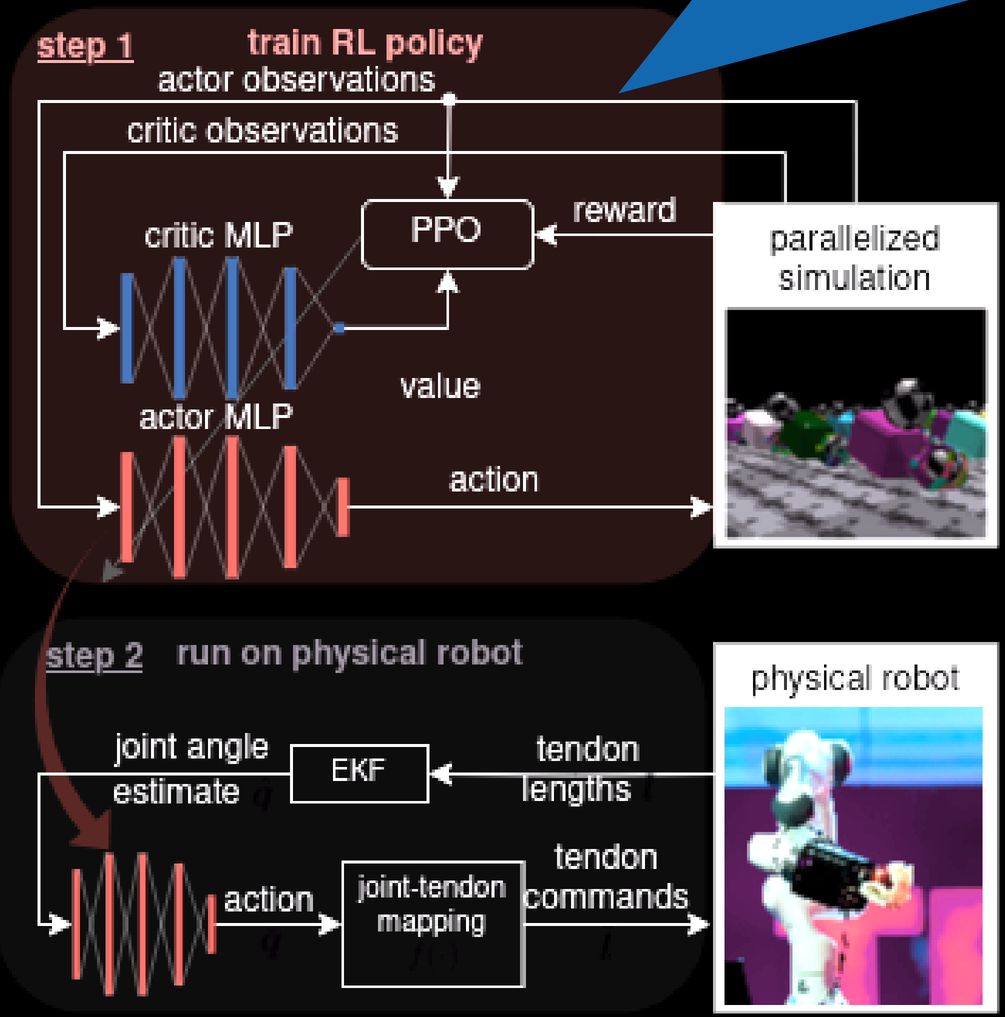
1. OpenAI, Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, et al. 2018. “Learning Dexterous In-Hand Manipulation.” *arXiv [cs.LG]*. arXiv. http://arxiv.org/abs/1808.00177
2. Makoviychuk, Viktor, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, et al. 2021. “Isaac Gym: High Performance GPU Based Physics Simulation For Robot Learning.” *Https://openreview.net › Forumhttps://openreview.net › Forum*. https://openreview.net/pdf?id=fgFBtYgJQX_.
3. Rudin, Nikita, David Hoeller, Philipp Reist, and Marco Hutter. 2021. “Learning to Walk in Minutes Using Massively Parallel Deep Reinforcement Learning.” *Https://openreview.net › Forumhttps://openreview.net › Forum*. https://openreview.net/pdf?id=wK2fDDJ5VcF.

SoftRobotics Laboratory

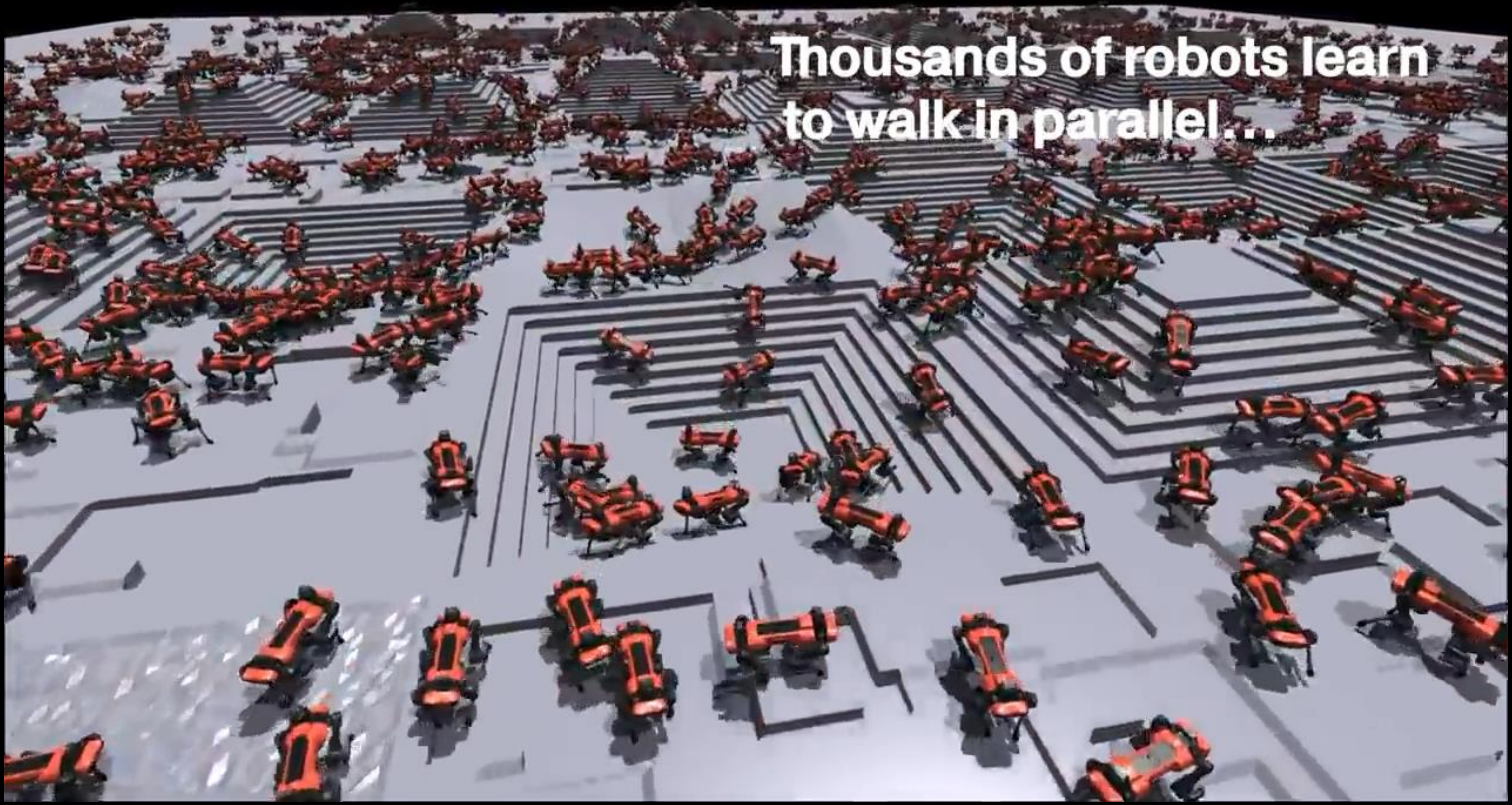
Sim2real framework for dexterous manipulation

1 hour of training at 70,000 fps ≒ 2 months of simulated time



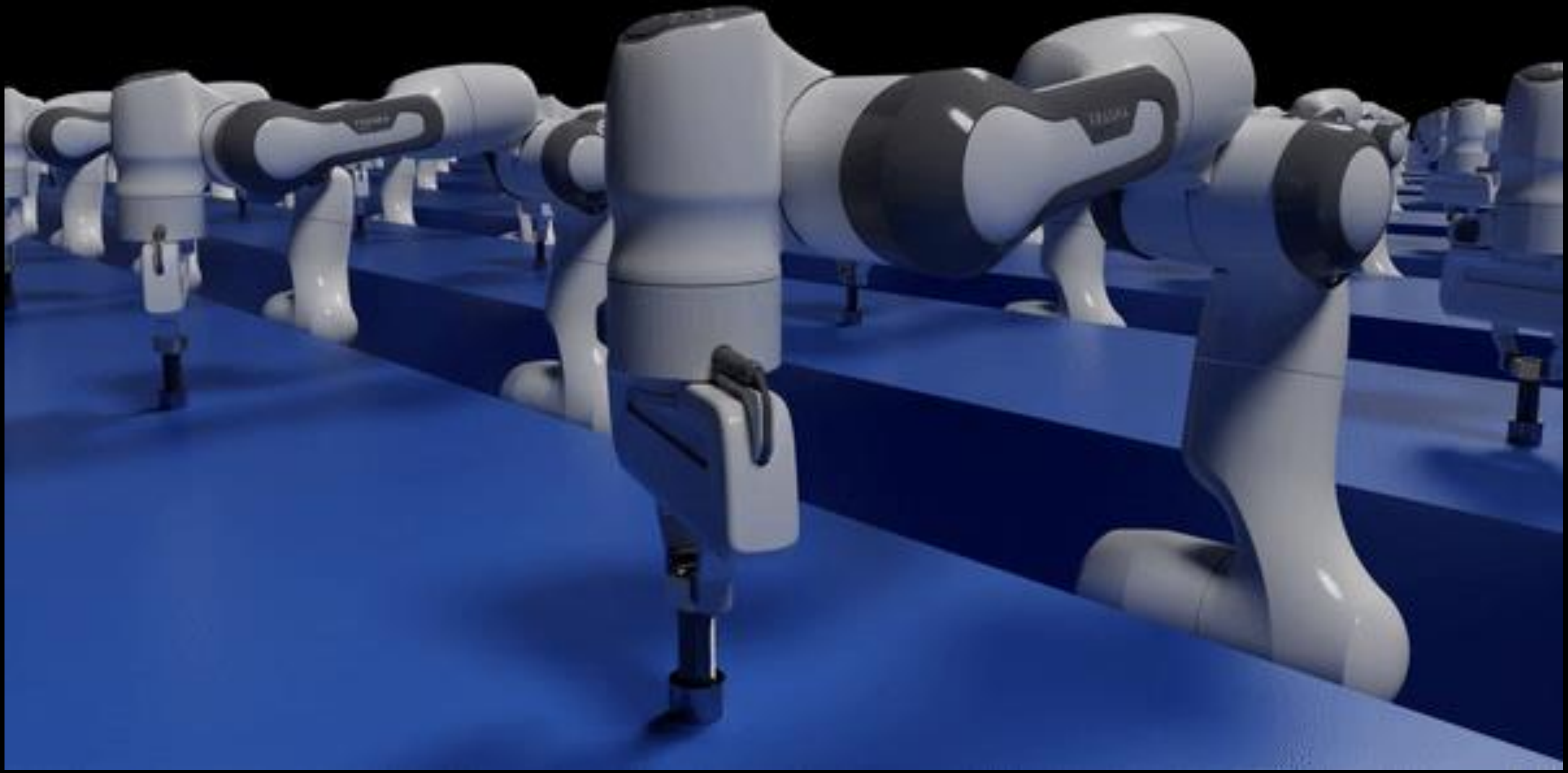


Massively parallel RL in different domains of robotics



Nikita Rudin, David Hoeller, Philipp Reist, Marco Hutter: Learning to Walk in Minutes Using Massively Parallel Deep Reinforcement Learning

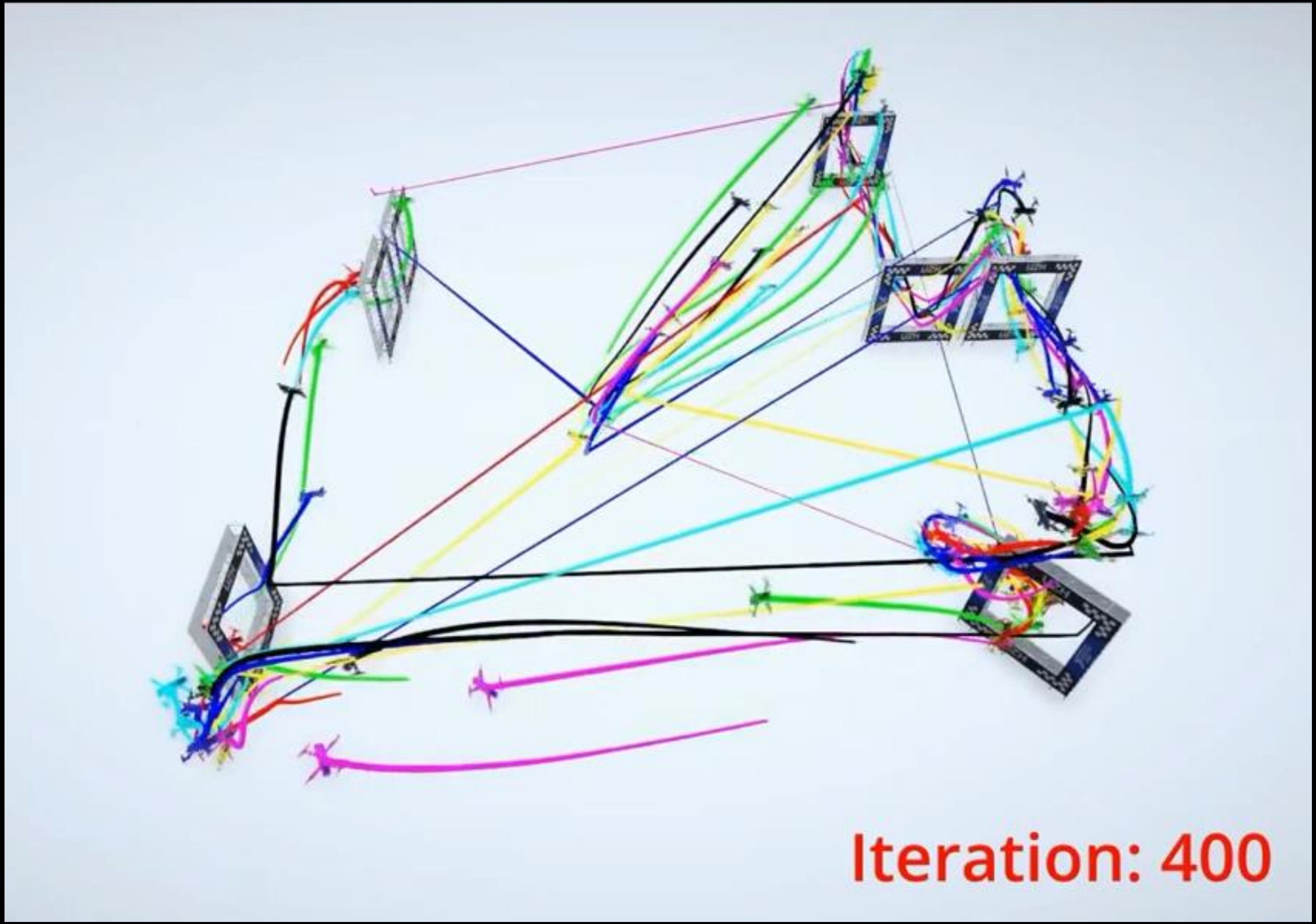
Massively parallel RL in different domains of robotics



Advancing Robotic Assembly with a Novel Simulation Approach Using NVIDIA Isaac: [Technical Blog]

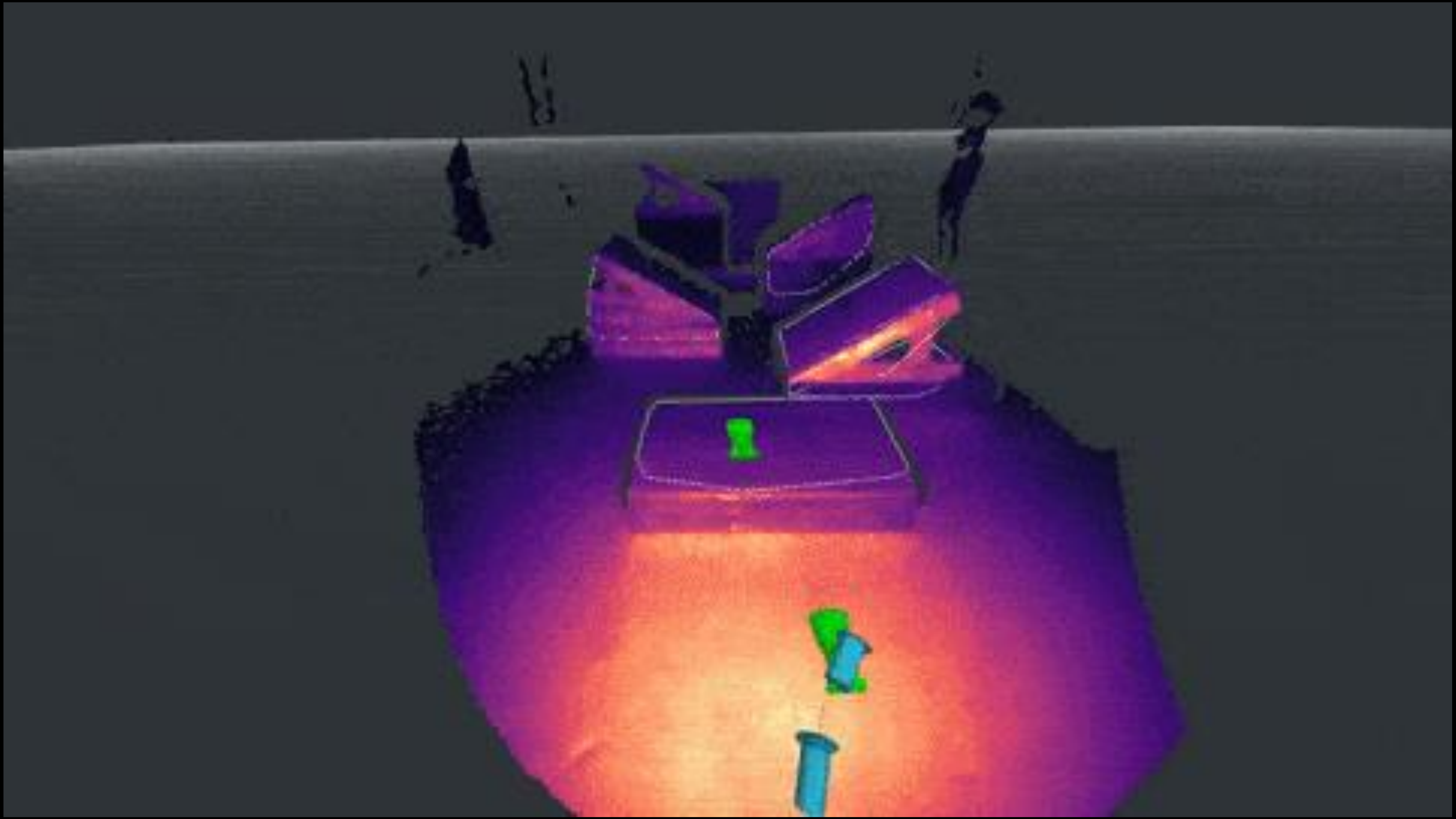
SoftRobotics Laboratory

Massively parallel RL in different domains of robotics



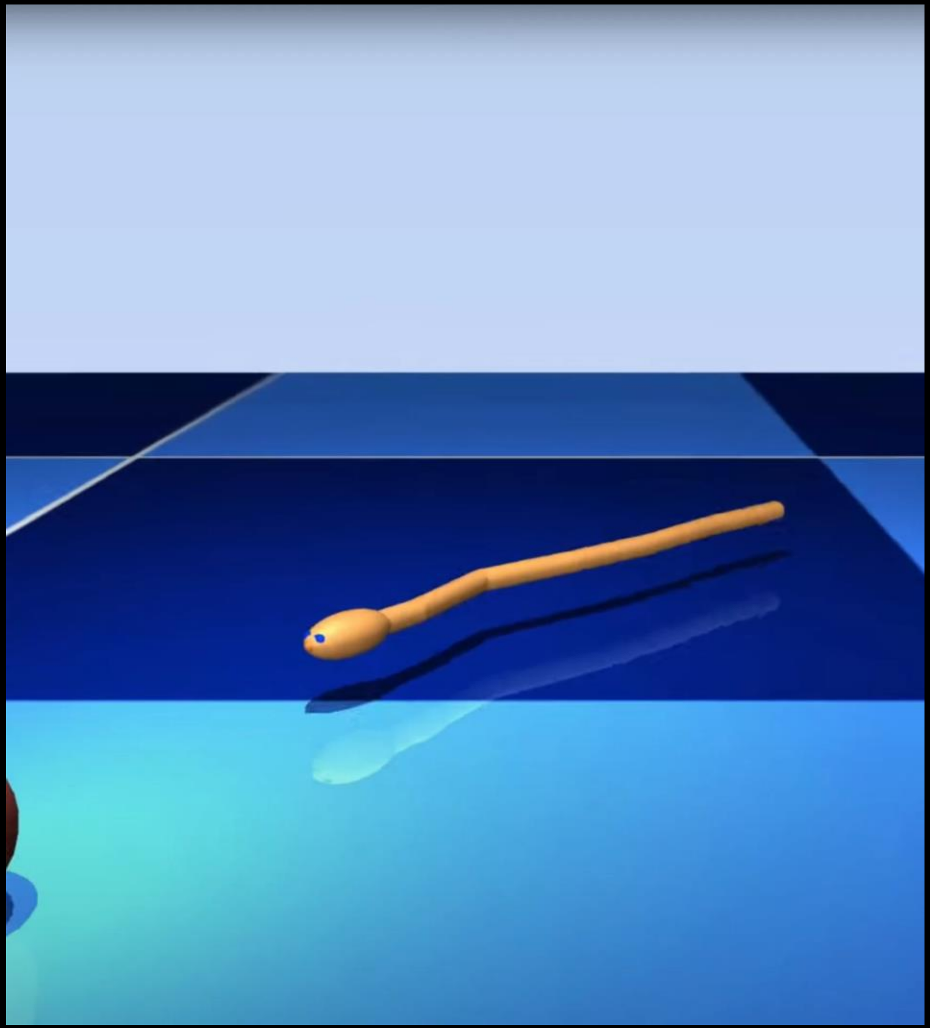
Nico Messikommer, Yunlong Song, Davide Scaramuzza: Contrastive Initial State Buffer for Reinforcement Learning

Online Rigid Body Sim for BD Atlas' MPC



Flipping the Script with Atlas: [Technical Blog]

Run model predictive control *on* the simulated MuJoCo model



MPC: model predictive control

- define a cost function to minimize, encoding the desired task
- iteratively apply optimization at every step to find the “best” control inputs

MuJoCo-MPC uses the MuJoCo simulation as the ***model*** in the MPC
→ efficient and accurate model-based control

https://github.com/google-deepmind/mujoco_mpc

Howell, Taylor, Nimrod Gileadi, Saran Tunyasuvunakool, Kevin Zakka, Tom Erez, and Yuval Tassa. 2022. “Predictive Sampling: Real-Time Behaviour Synthesis with MuJoCo.” *arXiv [Cs.RO]*. arXiv. http://arxiv.org/abs/2212.00541.

key idea when simulating robots:

simplification

Why simplify?

Efficiency

trade off some accuracy for speed

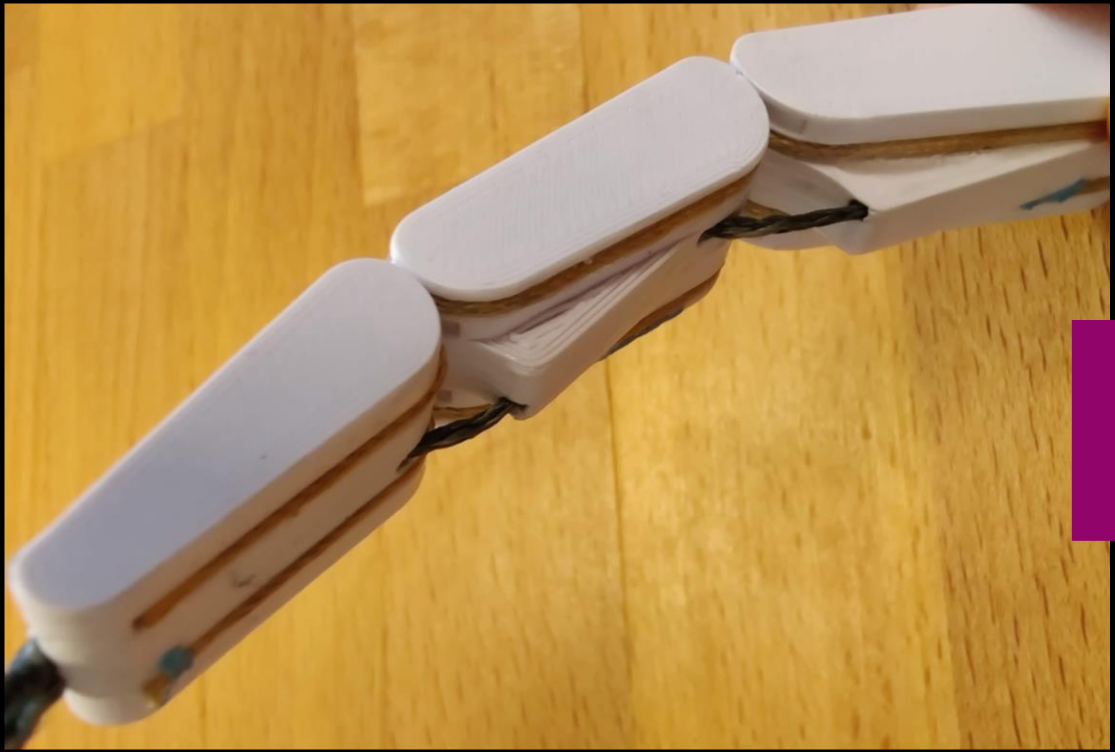
For example…
RL→ efficient simulation enables faster training
MPC→ faster exploration enables higher control frequency, longer control horizon

Modularity

If the real robot has a good low-level controller, the simulator can use high-level control inputs

For example…
The tendons of the Faive Hand are not (currently) replicated in the simulated model, and it accepts joint-level commands which are easier to learn

Simplification example #1: rolling contact joints



Real robot

Contact between cylindrical surfaces

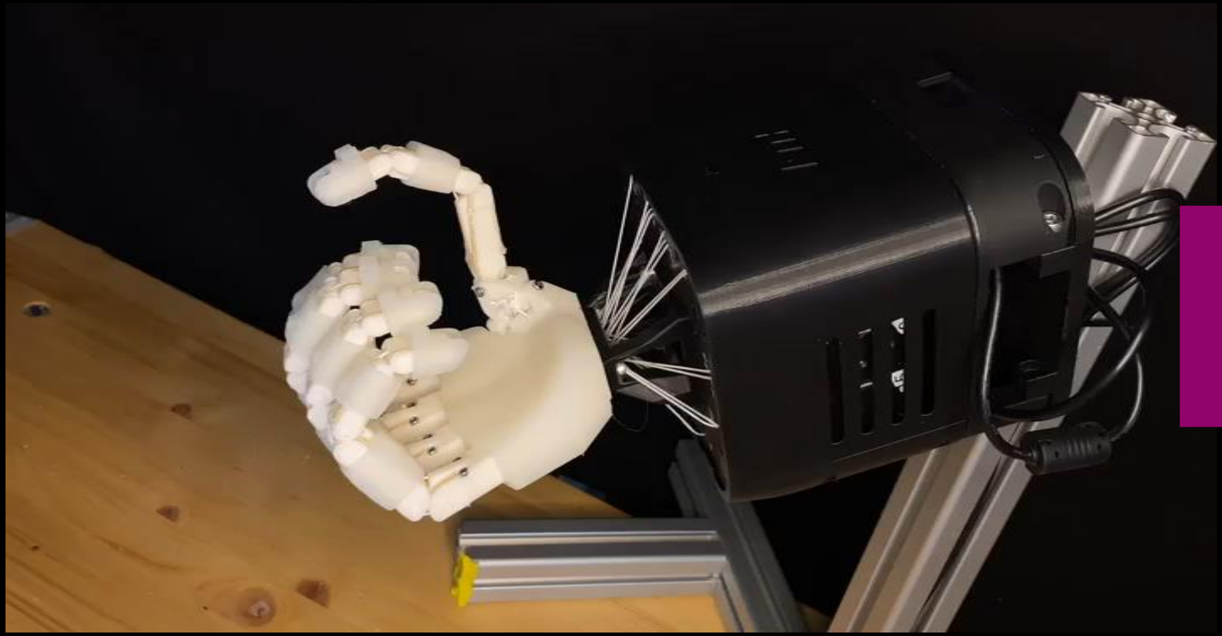
Ligaments ensure that they don't slide or move apart

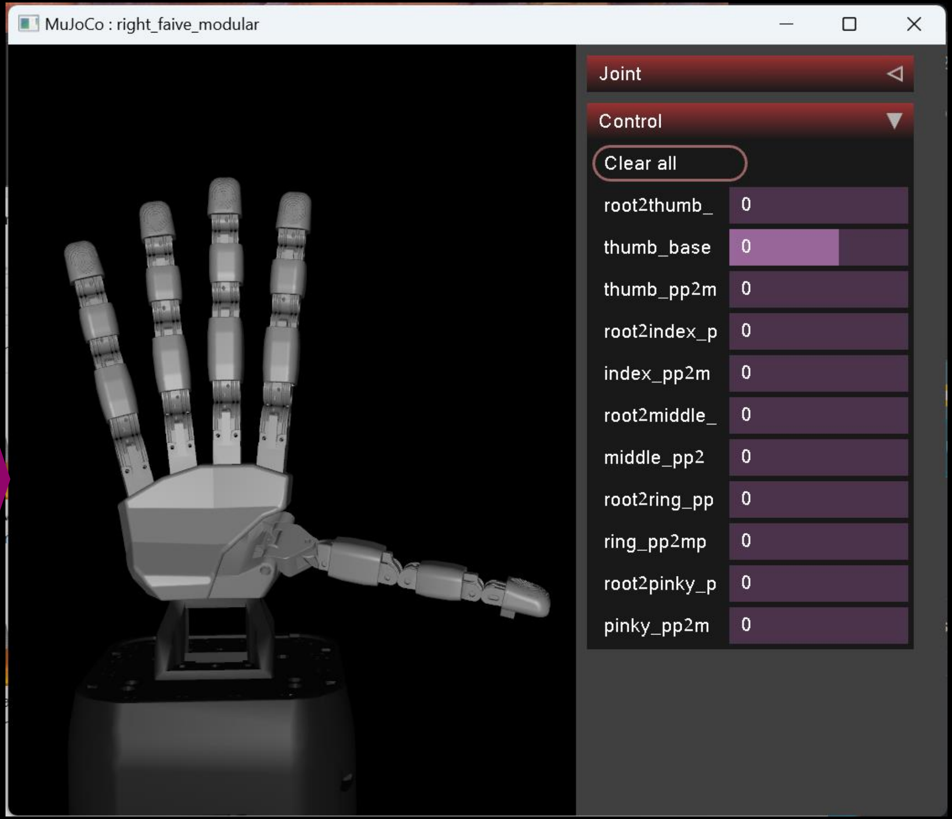
Simulated robot

Two hinge joints make up a single rolling contact joint

Constrained to roll together when the joint is actuated

Simplification example #2: tendon-driven actuation





Real robot

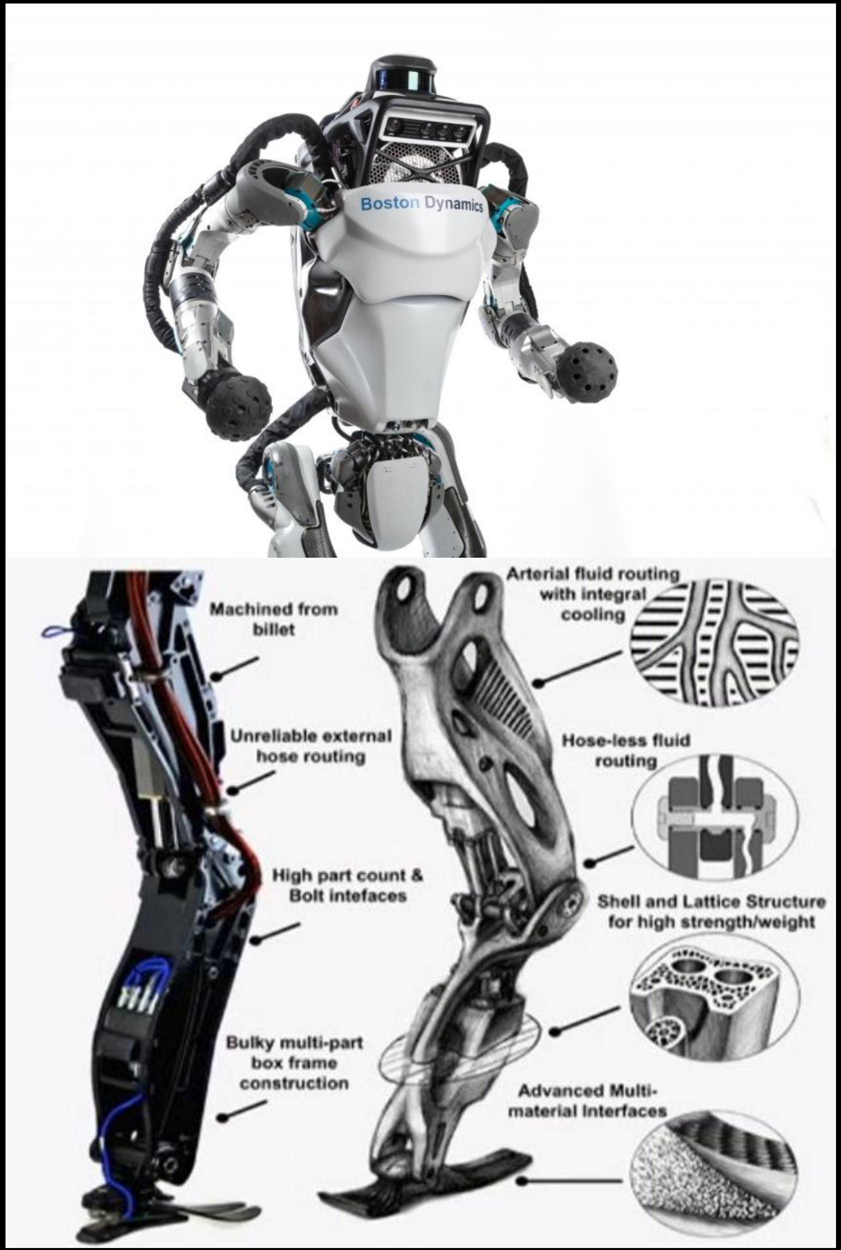
16 servo motors drive the tendons to actuate the hand

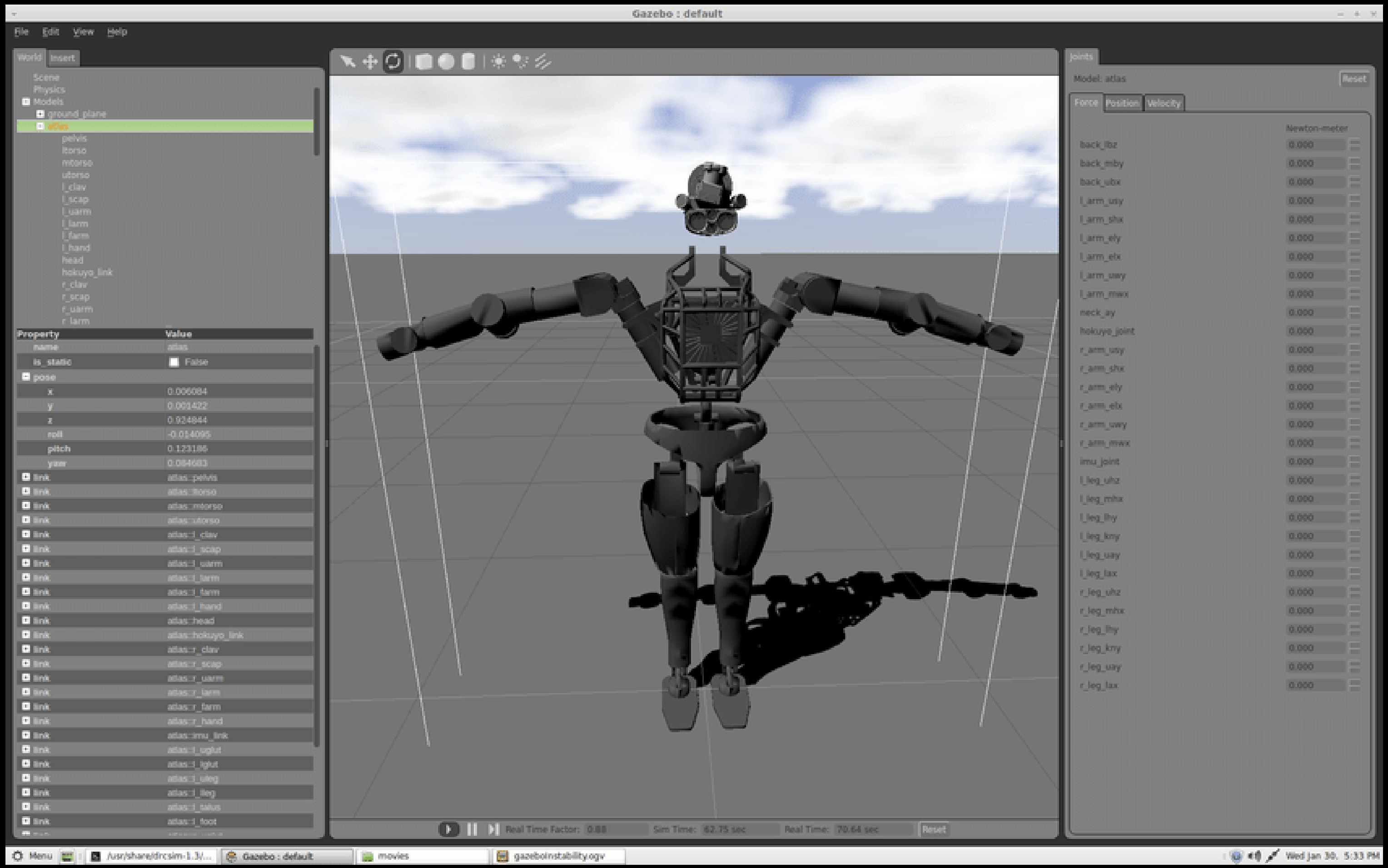
Low-level controller converts joint angles → tendon lengths → servo motor angles and sends it to Dynamixel motors

Simulated robot

The musculoskeletal system is ignored, and the robot is modelled purely as a joint axis-driven robot

Another example: Hydraulic actuators of the Atlas robot





Summary

Simulate and fail often to simplify later real-world experiments

Summary of why simulators are needed

***Safe* and *fast* environment for testing robot controllers**

Parallelized simulation for reinforcement learning (RL)

Model-based control based on simulators

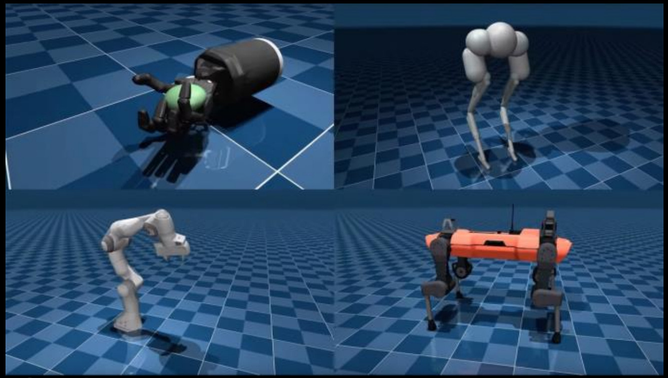
What simulators are out there? (2025 ver.)

Important questions for simulators:
Stability and accuracy?
GPU parallelization (i.e. speed)?
Open source?
Extensive documentation / large community?

"Team Google"

Google DeepMind

MuJoCo



(but not really)

"Team NVIDIA"

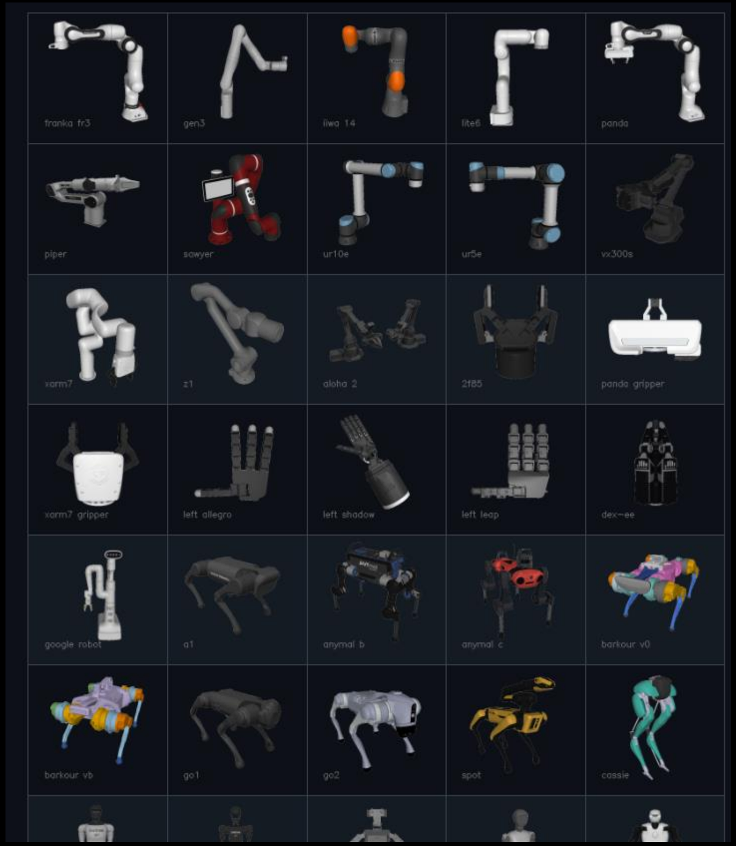
Isaac Sim / NVIDIA Omniverse

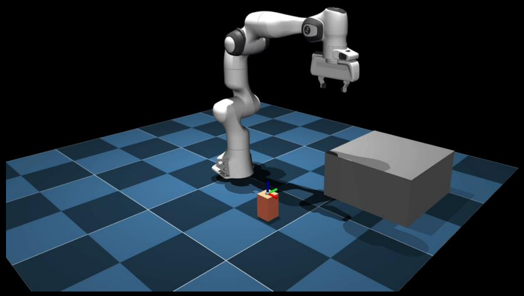
SoftRobotics Laboratory

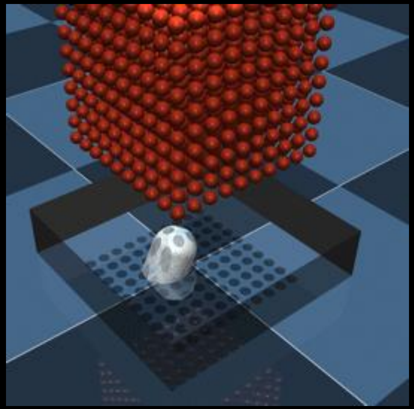
MuJoCo (<u>**Mu**</u>lti-<u>**Jo**</u>int dynamics with <u>**Co**</u>ntact)

Collection of high-quality MuJoCo assets: https://github.com/google-deepmind/mujoco_menagerie

- Originally developed by Emo Todorov et al. (just 3 people!)
- Optimization-based contact solver for efficiency, accuracy, and stability
- Acquired by Google DeepMind in 2021, open sourced 7 months later
- Supports URDF or MJCF (MuJoCo format)
- Originally CPU-only, but…
- MuJoCo XLA (MJX)
    - JAX/XLA rewrite of MuJoCo
    - runs on CPU, GPU, or TPU
    - Doesn't support [some features](#) ☹
- MuJoCo Warp (MJWarp)
    - Team up with NVIDIA
    - Now in Beta
    - GPU-optimized reimplementation to run on NVIDIA hardware
    - Should support more MuJoCo features than MJX
    - Will be integrated into MJX in future







SoftRobotics Laboratory

E. Todorov, T. Erez and Y. Tassa, "MuJoCo: A physics engine for model-based control," 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura-Algarve, Portugal, 2012, pp. 5026-5033, doi: 10.1109/IROS.2012.6386109.

NVIDIA Isaac Sim / Isaac Lab

- Isaac Sim
  - simulator using PhysX for the physics engine
  - Apache-2.0 open source (building/using it requires other NVIDIA licensed components)
- Isaac Lab:
  - Wraps Isaac Sim, to build RL / IL workflows
- Only runs on NVIDIA's GPUs
- Lab will become compatible with Newton
  - open-source GPU physics engine
  - MuJoCo-Warp solver

MuJoCo

Google DeepMind

In our hands-on part, we will implement dexterous RL on the ORCA hand using faive_lab, an extension of Isaac Lab!

E. Todorov, T. Erez and Y. Tassa, "MuJoCo: A physics engine for model-based control," 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura-Algarve, Portugal, 2012, pp. 5026-5033, doi: 10.1109/IROS.2012.6386109.