Virtual Humans – Winter 23/24

Lecture 12_2 – Human Motion Synthesis

Prof. Dr.-Ing. Gerard Pons-Moll

University of Tübingen / MPI-Informatics

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

In this lecture…

- Synthesising **human object interaction.**

- Applying **fine grained control** over how interaction happens.

- Using RL to learn physics based synthesis.
  Refresher on reinforcement learning.

Synthesis beyond locomotion... Interactions.

- Interactions such as sitting on a chair, lifting a box
- There are two key challenges:
  1. Goal-oriented interactions
  2. Transiting naturally between motions

PFNN. Holden et al. Sig'19
NSM. Starke et al. SigAsia'19

1. **Goal** driven **interaction** synthesis

- PFNN: Future motion function of past motion and terrain.

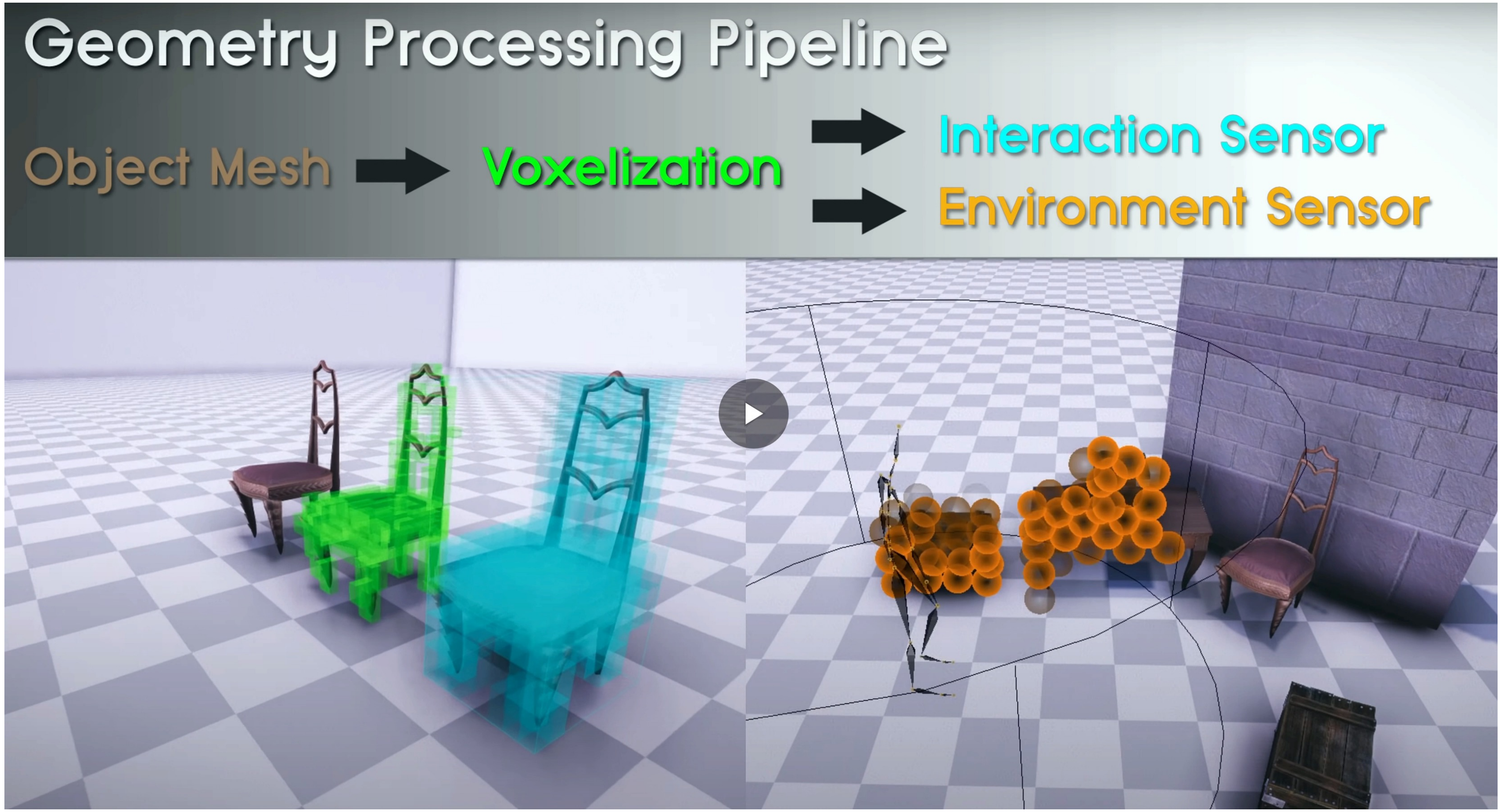
$$\overline{J_t, J'_t, T_t, T'_t, H_t, \Delta\phi} = f(\overline{J_{t-1}, J'_{t-1}, T_{t-1}, T'_{t-1}}, \overline{H_{t-1}}, \phi_t)$$

- NSM: Future motion function of past motion, goal, interaction and environment.

$$\overline{J_t, J'_t, T_t, T'_t}, G_t, I_t, E_t, \Delta\phi = f(\overline{J_{t-1}, J'_{t-1}, T_{t-1}, T'_{t-1}}, \overline{G_{t-1}}, \overline{I_{t-1}}, \overline{E_{t-1}}, \phi_t)$$

- The goal is represented as root of the object.
- How are **interaction** and **environment** encoded?

Interaction and Environment Sensors

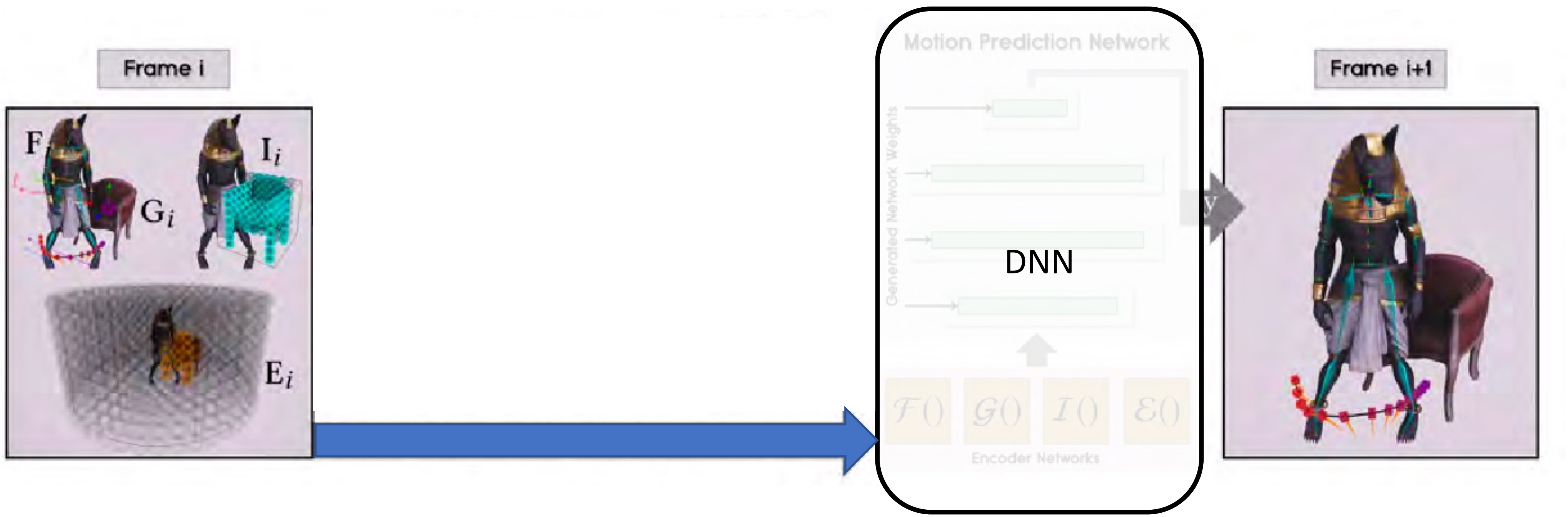


Interaction sensor $I_t \in R^{512}$: voxelized object
Environment sensor $E_t \in R^{1048}$: occupancy of the surrounding

2. Transitioning smoothly between actions/motions, e.g.: walking, lifting, sitting

NSM. Starke et al. SigAsia'19

We can simply train a NN to predict future motion, but this results in unsmooth transitions



Input: Past motion, goal, environment.

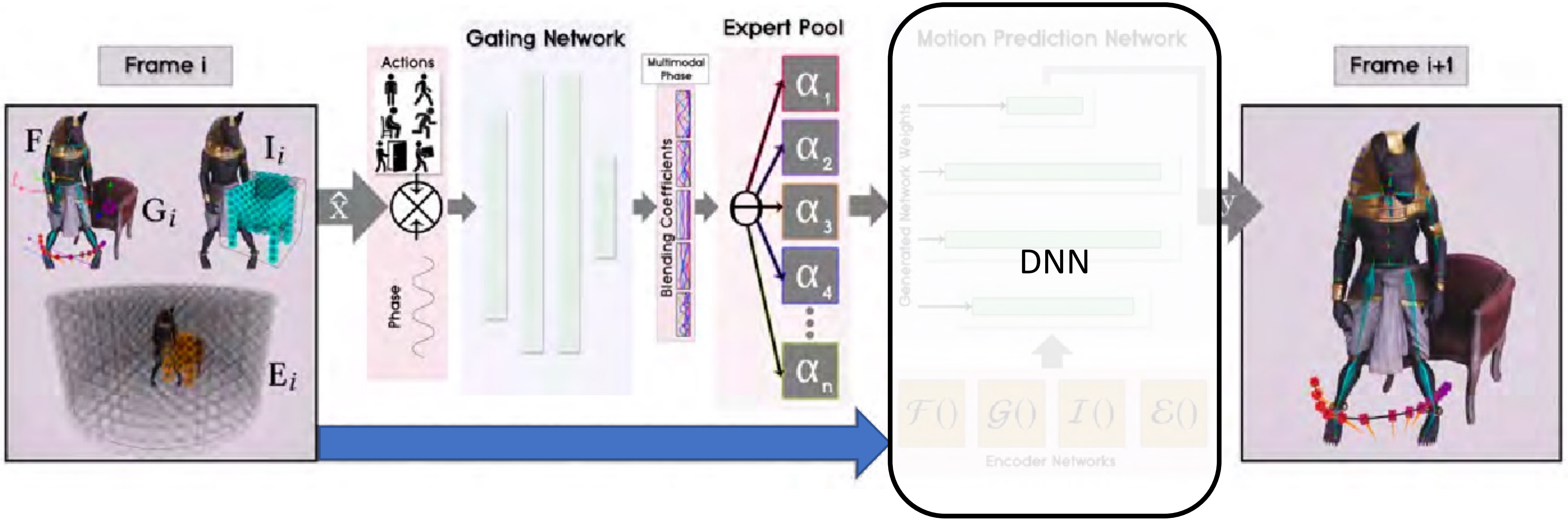
Neural Network

Output: Future Motion

2. Transitioning smoothly between actions/motions, e.g.: walking, lifting, sitting

NSM. Starke et al. SigAsia'19

Instead we use a "mixture of experts" prediction.
We predict weights for each expert. This allows each expert to master different motion.



Input: Past motion, goal, environment.

Neural Network

Output: Future Motion

We can synthesise complex interactions like sitting and lifting

Limitations of NSM

- Generated motion lacks diversity - there can be many modes.
- What if the user want to specify, for example, sit while supporting with the left hand on the armrest?

COUCH, Zhang et al. ECCV'22

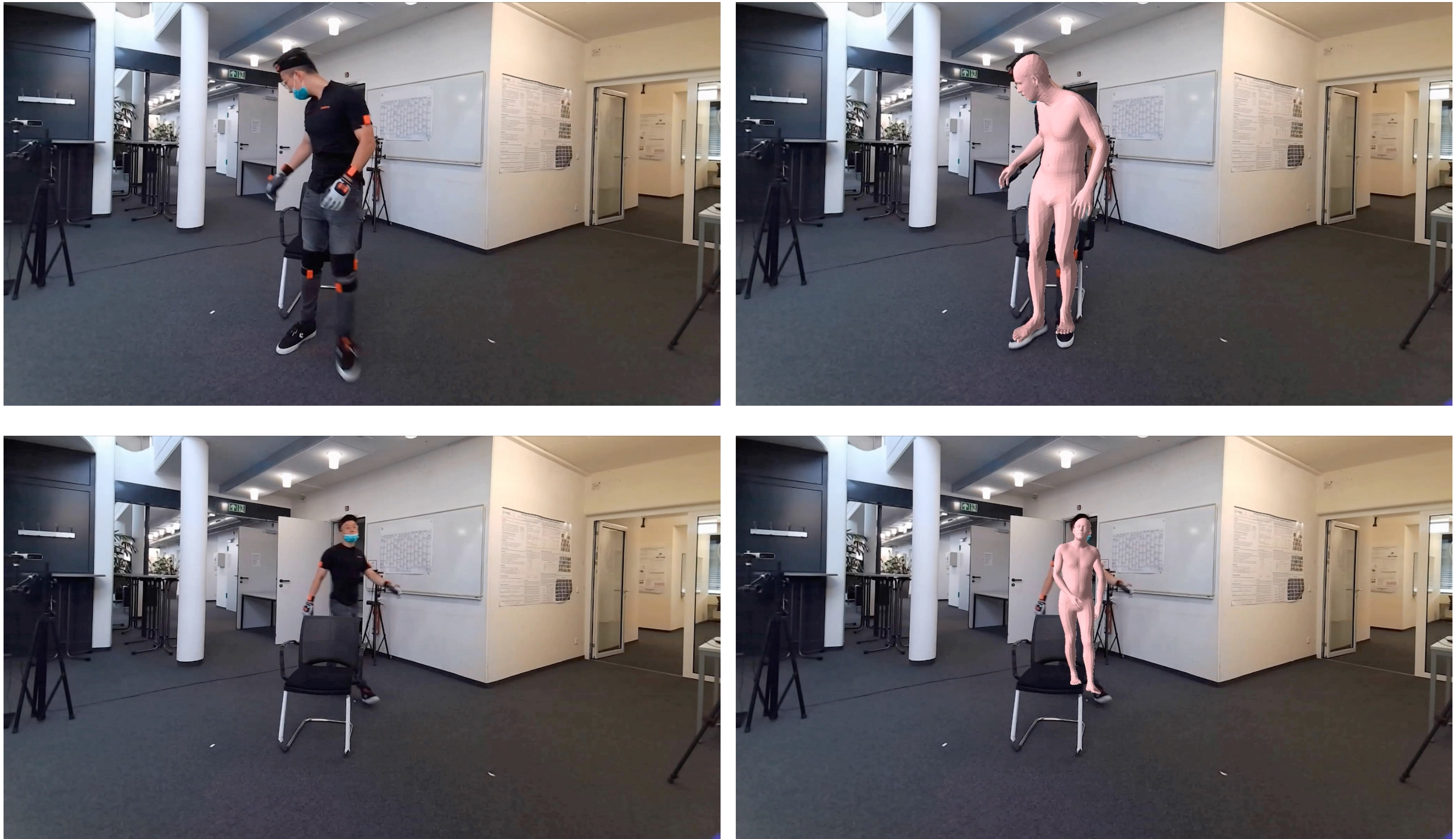
Our goal is to synthesize **controllable** human-chair interactions via **specified** or **sampled** contacts

Key Challenges

- Prior datasets do not capture:
  - ➢ real scene interactions
  - ➢ diverse and accurate contacts
- How to condition generated interaction to satisfy the contacts?
- No data!

COUCH dataset:
Robust Motion Capture with Kinects and IMUs

COUCH. Zhang et al. ECCV'22



Conditioning motion on desired contact

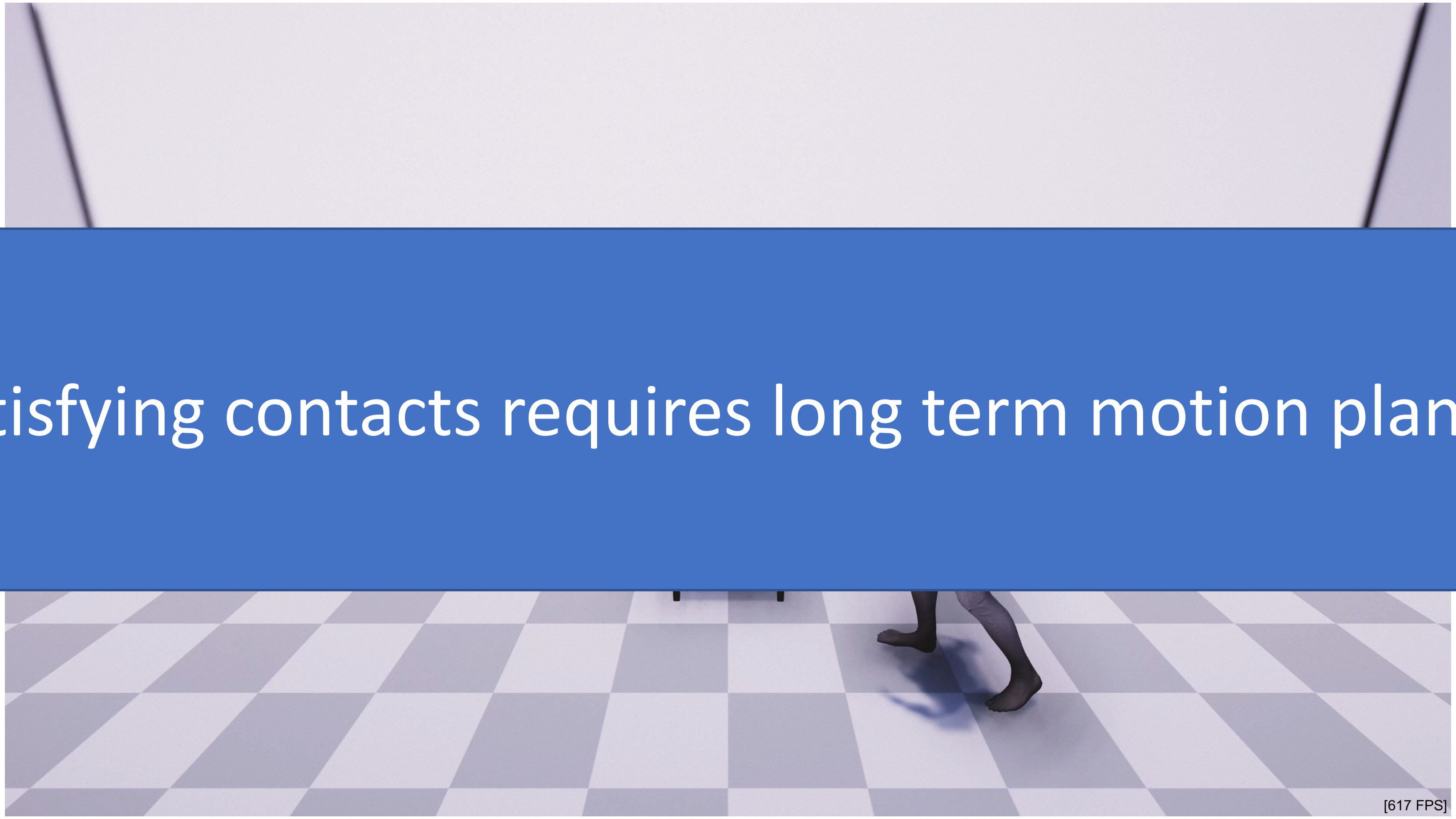
Extending "Goal driven interaction synthesis" to also add contacts, e.g. extend NSM to also take contacts.

NSM: Future motion function of past motion, goal, interaction and environment.

$$J_t, J'_t, T_t, T'_t, G_t, I_t, E_t, \Delta\phi = f(J_{t-1}, J'_{t-1}, T_{t-1}, T'_{t-1}, G_{t-1}, I_{t-1}, E_{t-1}, \phi_t)$$

$+C_t$ $+C_{t\text{-}1}$

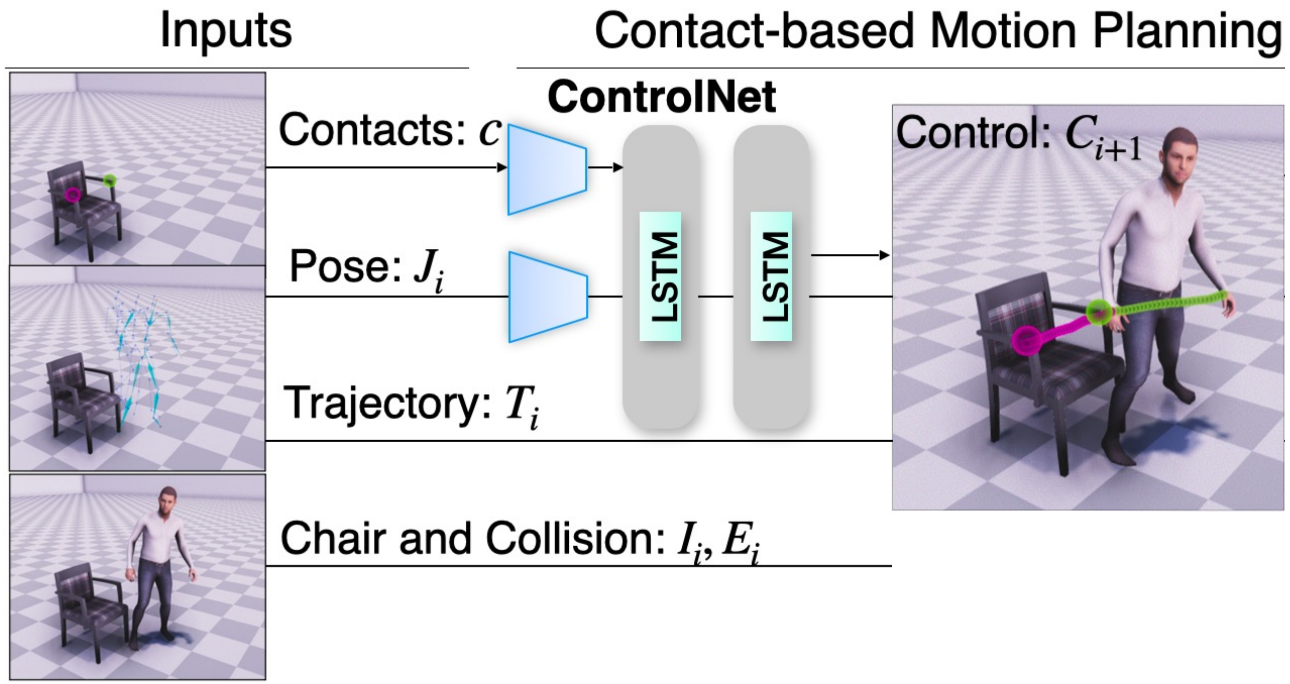
NSM + Contacts doesn't work.
Contacts are not satisfied.



Satisfying contacts requires long term motion planning.

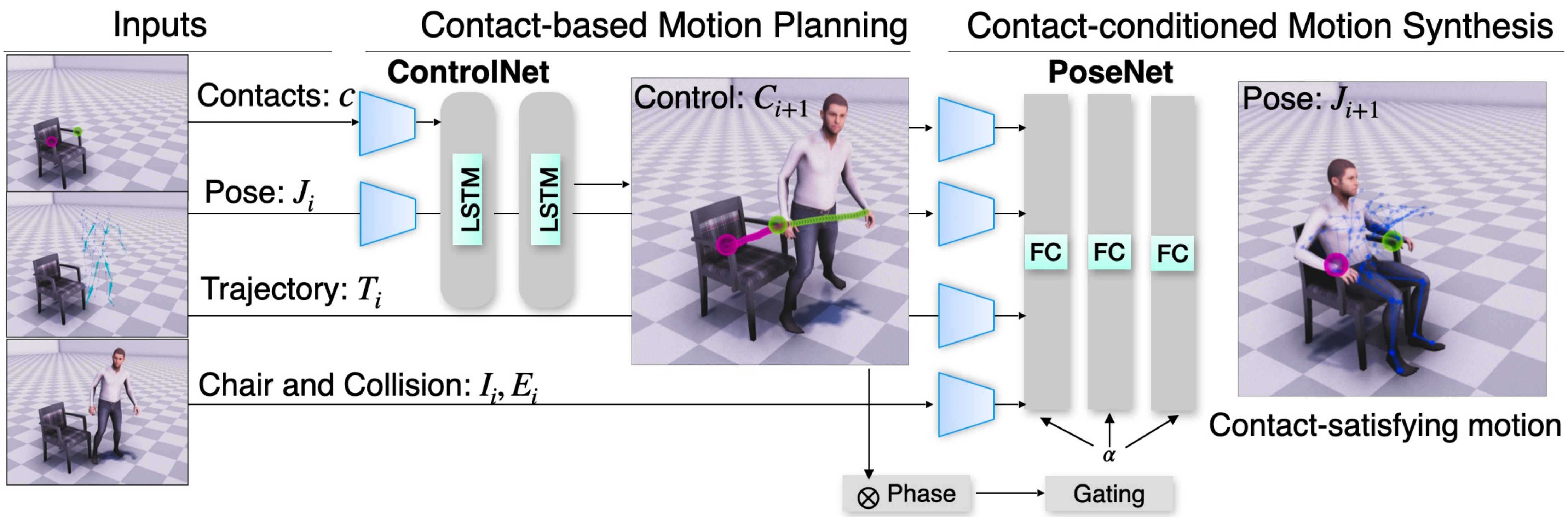
Key Idea:
Disentangle **motion planning** and **synthesis**

Given past inputs and goal, *ControlNet* predicts trajectories for root and hand joints

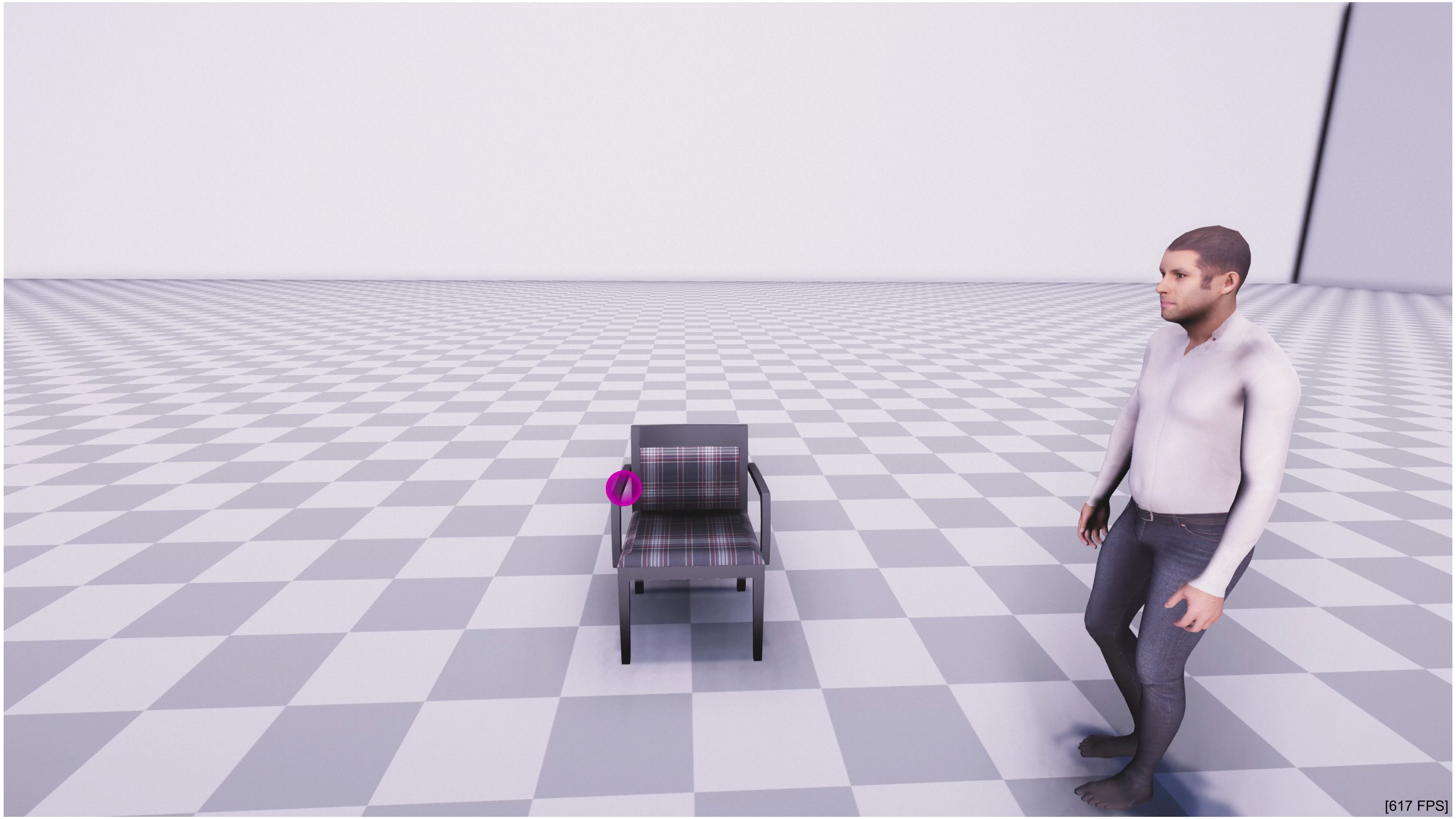


Key Idea:
Disentangle **motion planning** and **synthesis**

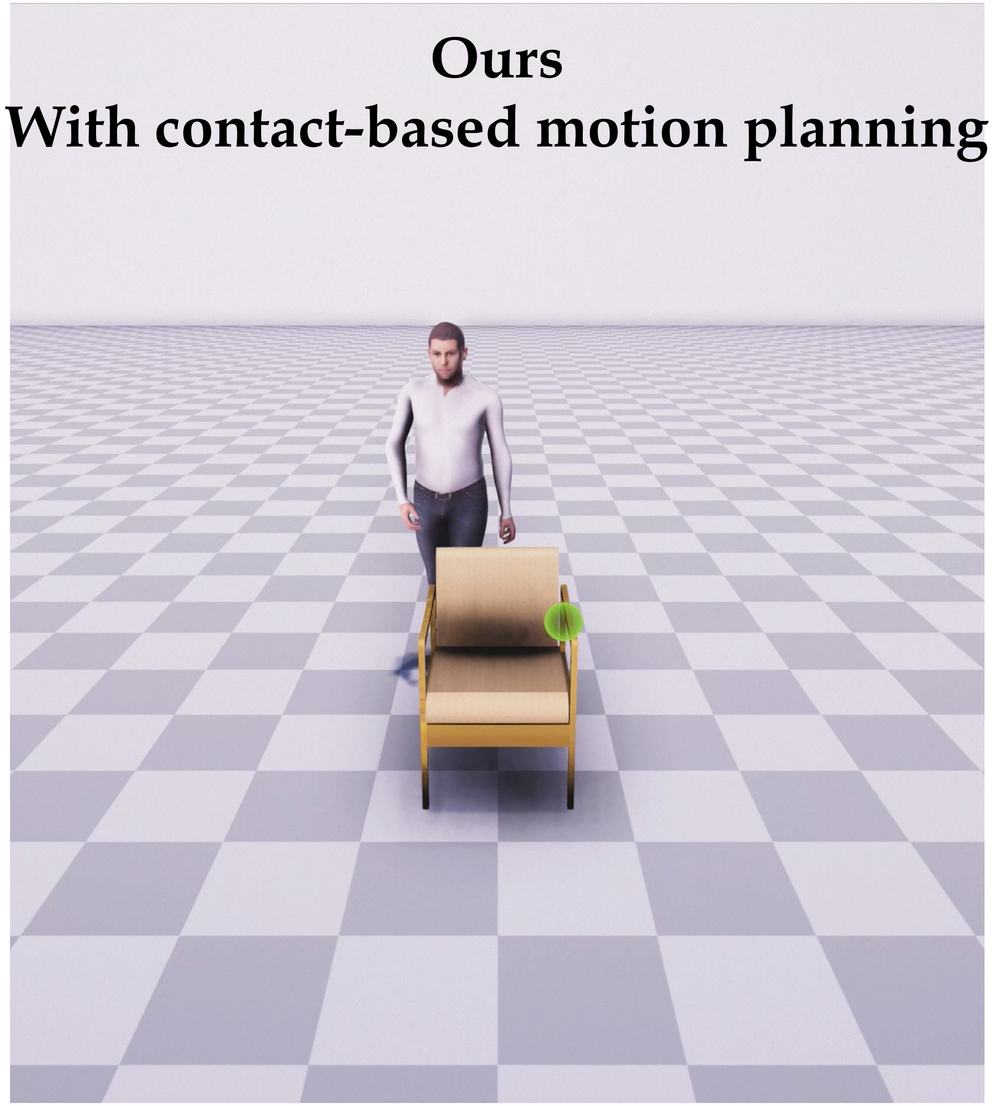
PoseNet takes controls from *ControlNet* and predicts 3D poses along the trajectories.



With 'motion planning' the contacts are satisfied



With motion planning COUCH satisfies the contacts



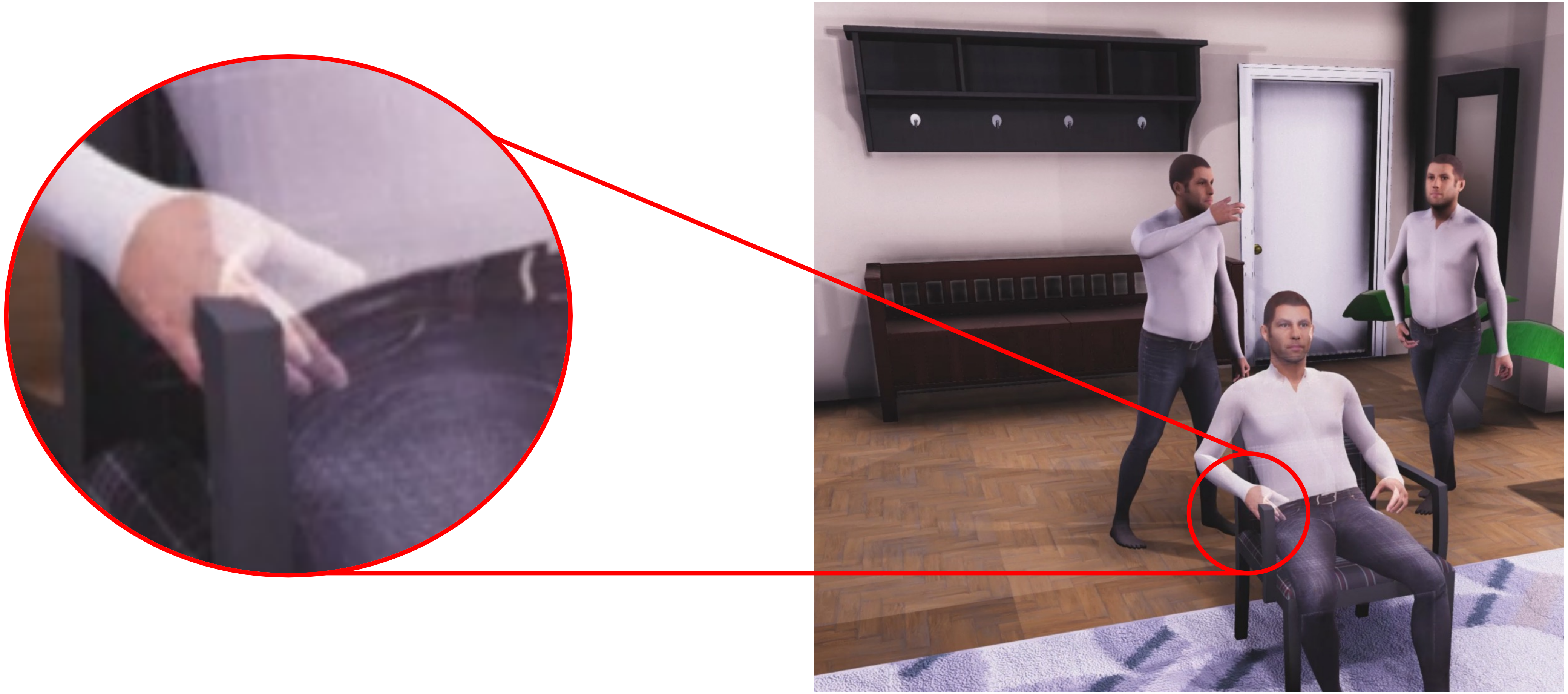
Contact with Left Hand

Limitations of COUCH

1. COUCH satisfies contacts but the hand grasp is not always natural.

2. COUCH requires a lot of data just to handle chairs.

Limitations of COUCH:
1. Contacts satisfied but grasp not always natural

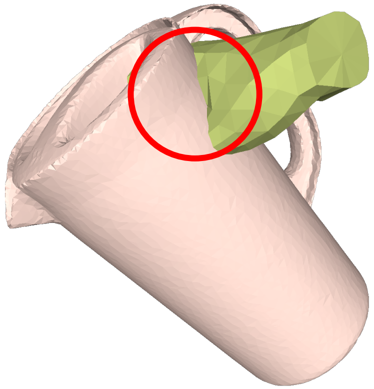


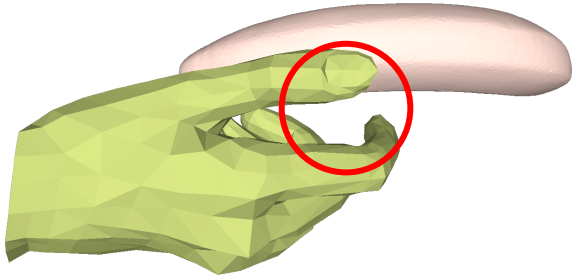
Hands are of special importance

Most of our day-to-day interactions happen through hands

Zhou et al. TOCH. ECCV 2022

Hands are are very challenging



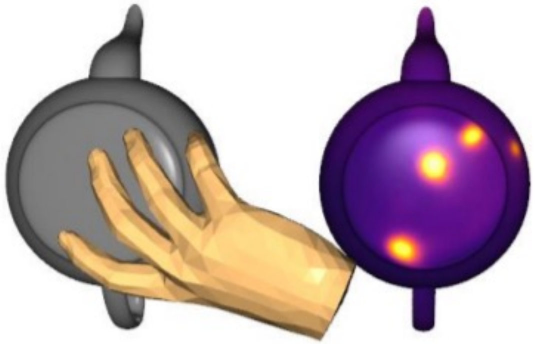


Inter-penetration

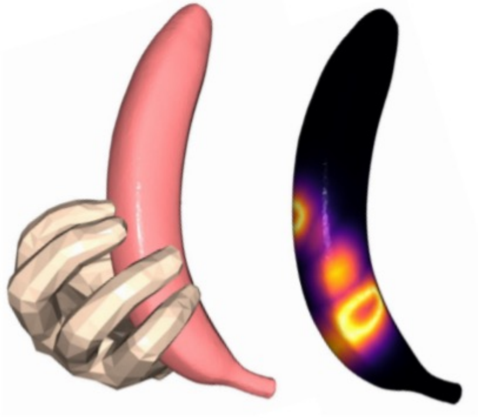
Unstable grasp

- **How to obtain correct and natural gasp?**
- **How do we even represent detailed hand-object contacts?**

Previous work represents contact with heatmap:



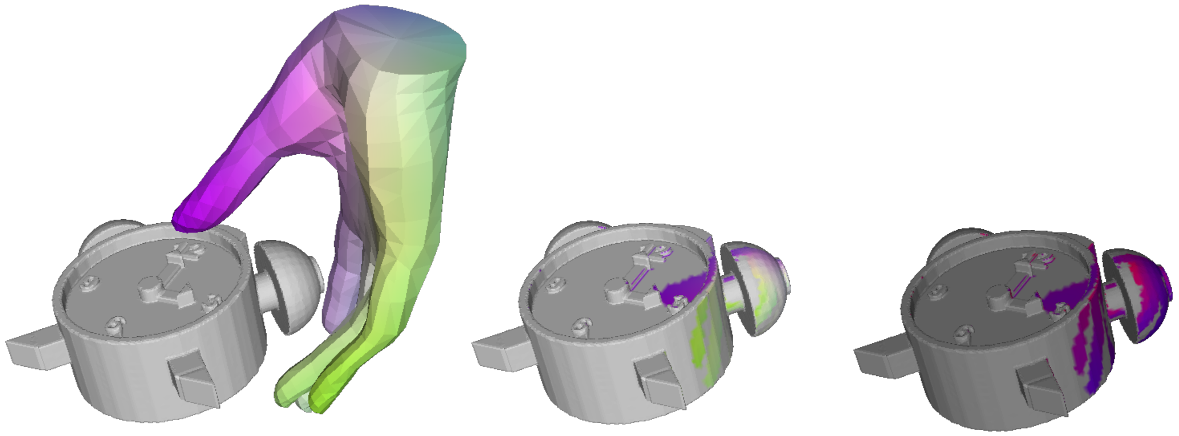
[1]



[2]

- Can only model static contact
- No correspondence between hand and object

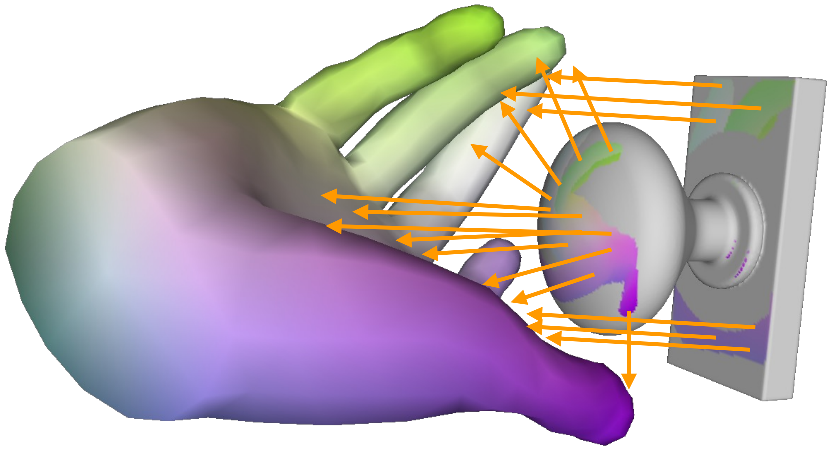
TOCH field generalizes contact to correspondence:



- Enables modeling **dynamic hand-object interaction**
- Easy hand-fitting with **dense correspondence**

[1] Grady, Patrick, et al. "ContactOpt: Optimizing Contact to Improve Grasps. “
[2] Jiang, Hanwen, et al. "Hand-Object Contact Consistency Reasoning for Human Grasps Generation."

TOCH Field

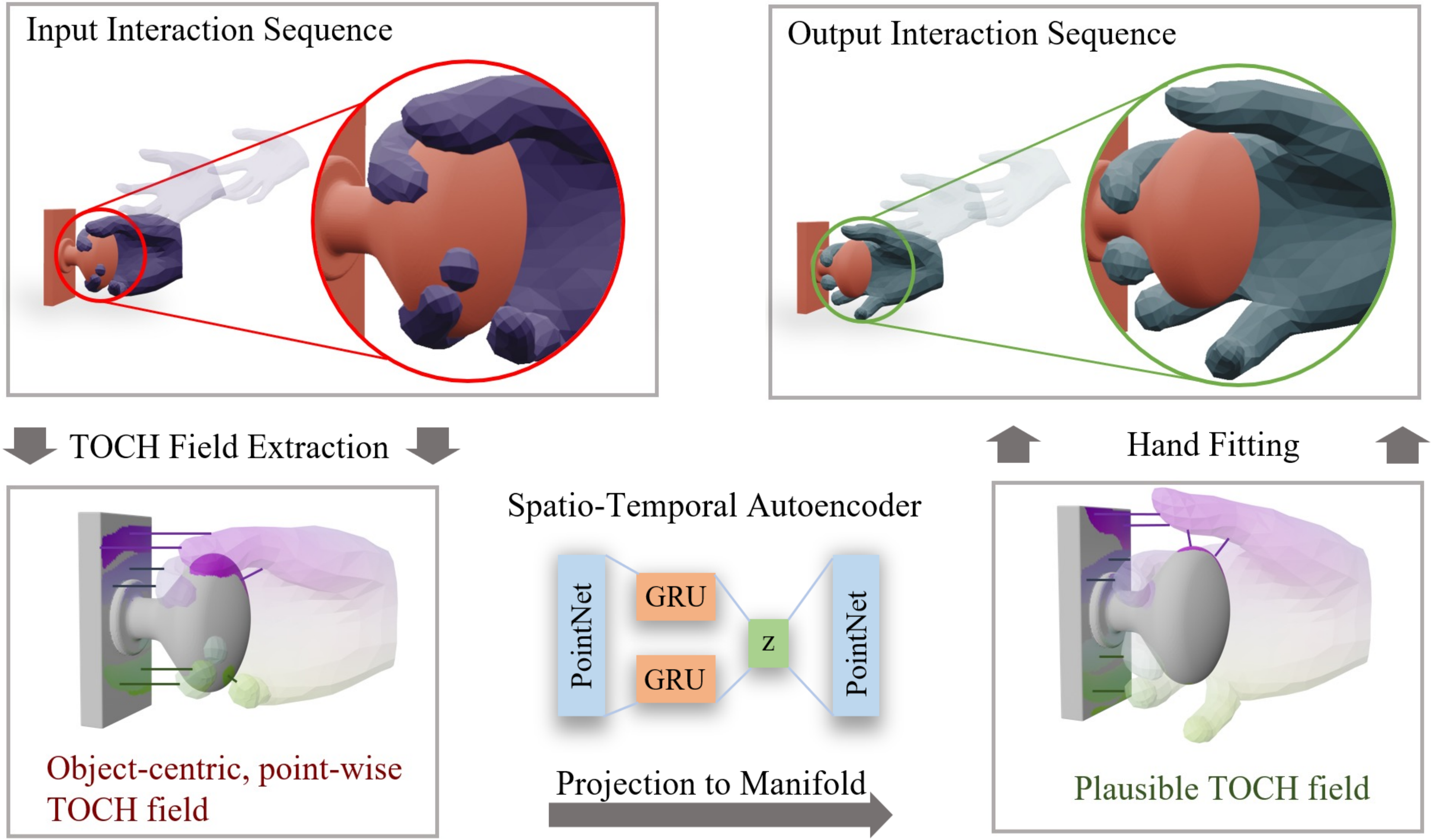


We find hand-object correspondence by ray casting

TOCH field records point-wise

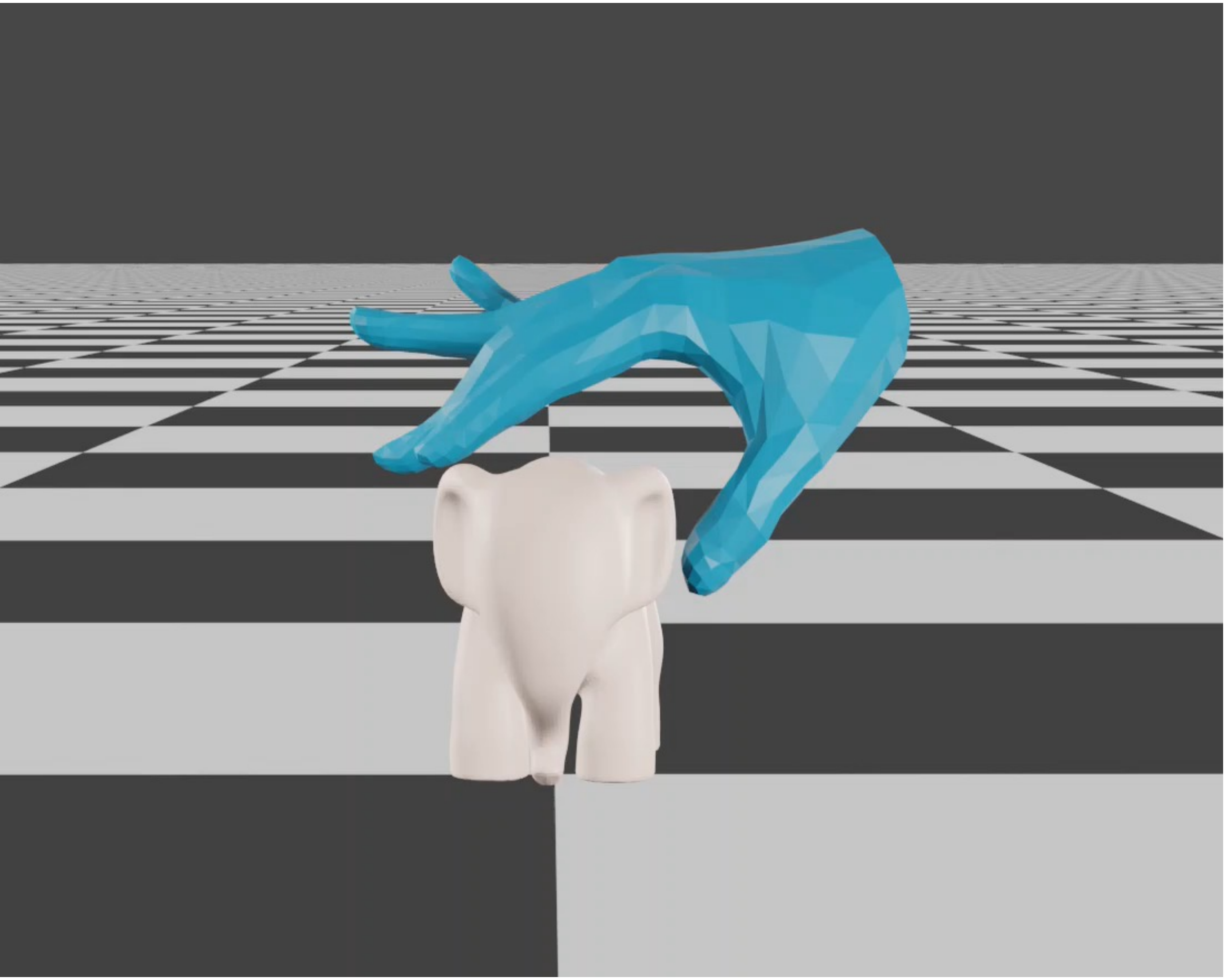
- binary correspondence indicator
- signed correspondence distance
- corresponding hand coordinates

$$\{c_i, d_i, y_i\}_{i=1}^{N}$$

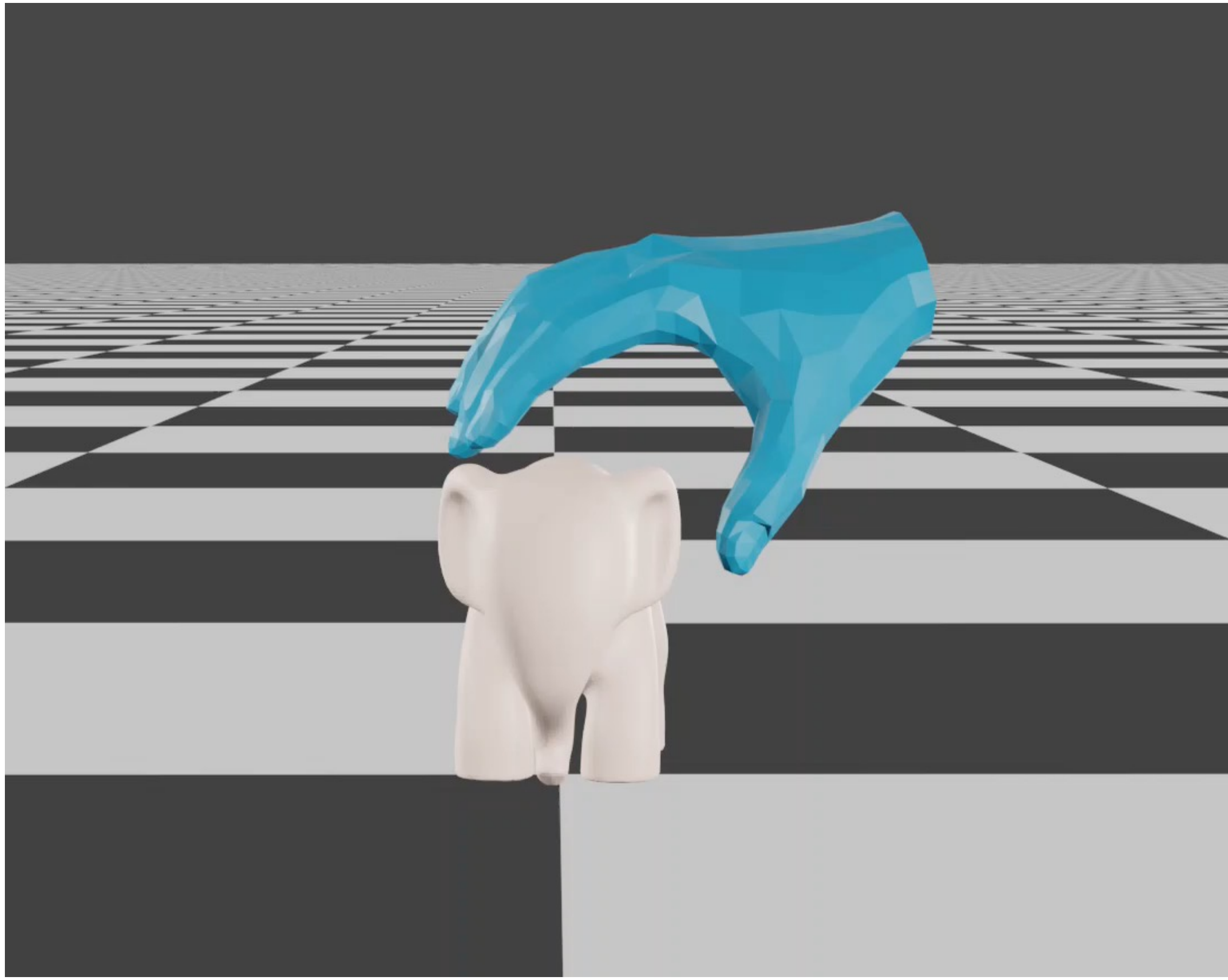

Input Interaction Sequence
Output Interaction Sequence
TOCH Field Extraction
Hand Fitting
Spatio-Temporal Autoencoder
PointNet
GRU
GRU
z
PointNet
Object-centric, point-wise TOCH field
Projection to Manifold
Plausible TOCH field

Zhou et al. TOCH. ECCV 2022

Erroneous Input



TOCH Correction

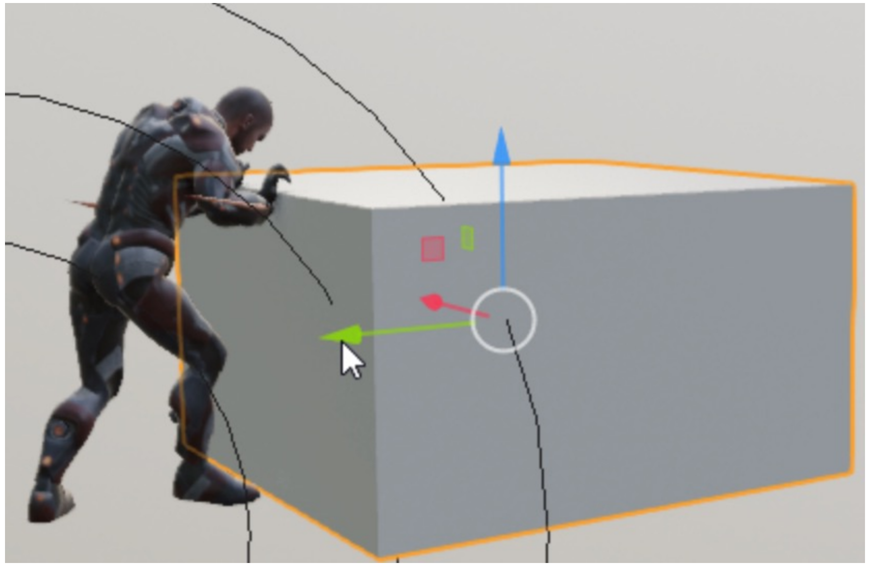


Limitations of COUCH:
2. Requires a lot of data just to handle chairs

- COUCH produces motion that satisfies contact. But it requires a lot of data just to model chairs.

- How to get data to learn **diverse interactions**? e.g. The posture required to move a heavy box is different than the light one.

- Can we learn **motion modelling using physics**?

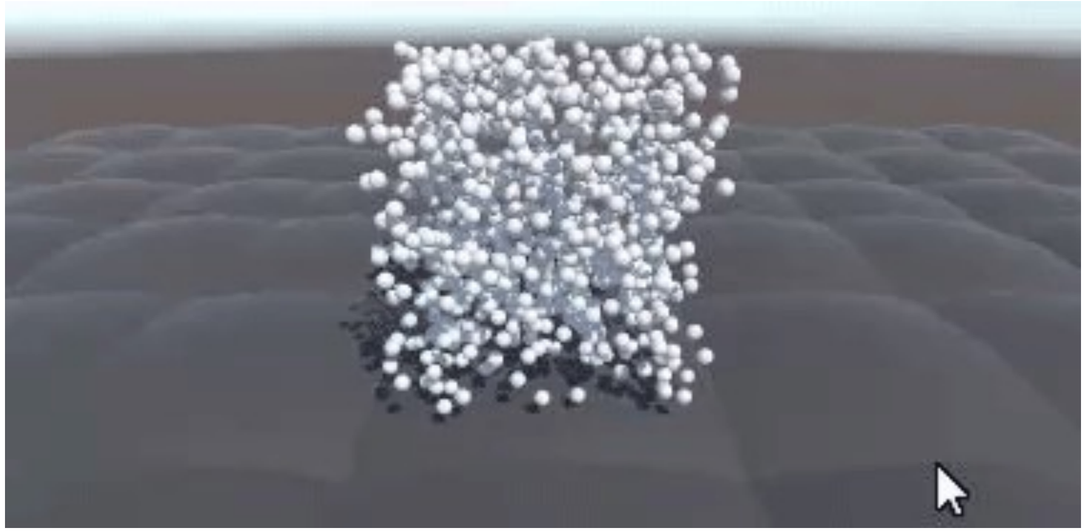


Physics based modelling

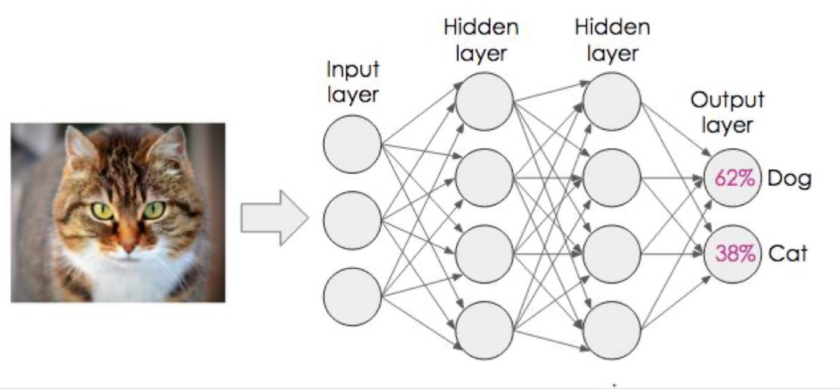
✓ Model physical properties such as forces, acceleration.

✓ Generalisable and interpretable.

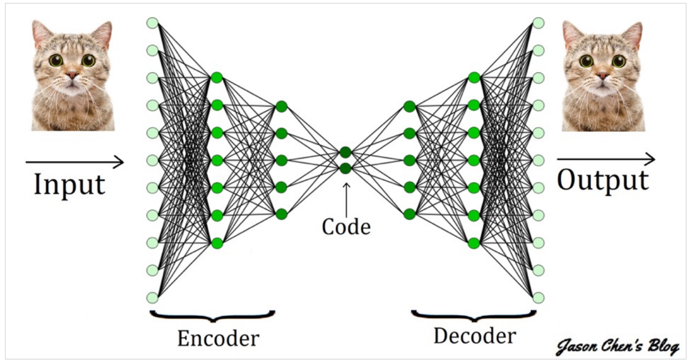
– Higher computational burden.
– Difficult to scale with supervised learning.
- **Can we use simulation?**



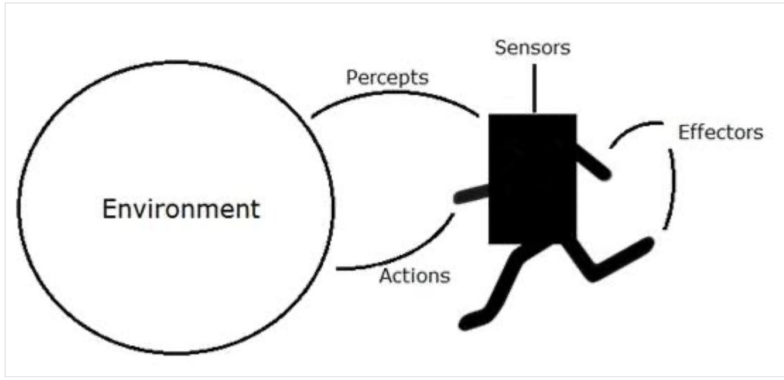
Reinforcement Learning for learning Physics



Supervised learning



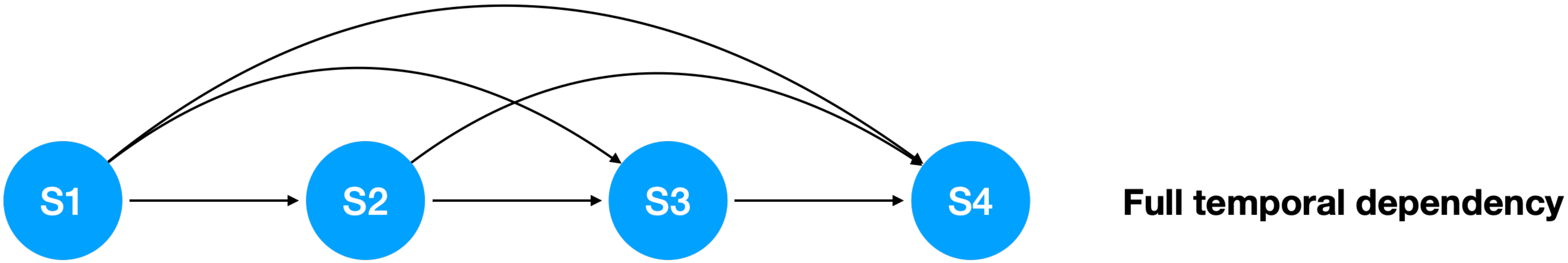
Unsupervised learning

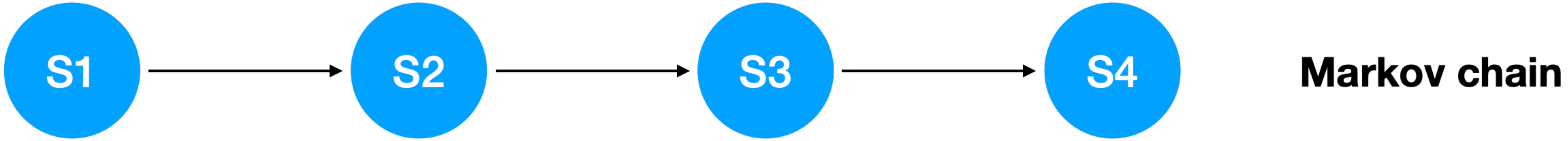


Reinforcement learning

- Unlike (un)supervised learning, RL does not need humans to prepare the training data.
- The agent collects data via exploring the environment, learns how to achieve the goal, and exploits such knowledge to make decisions.

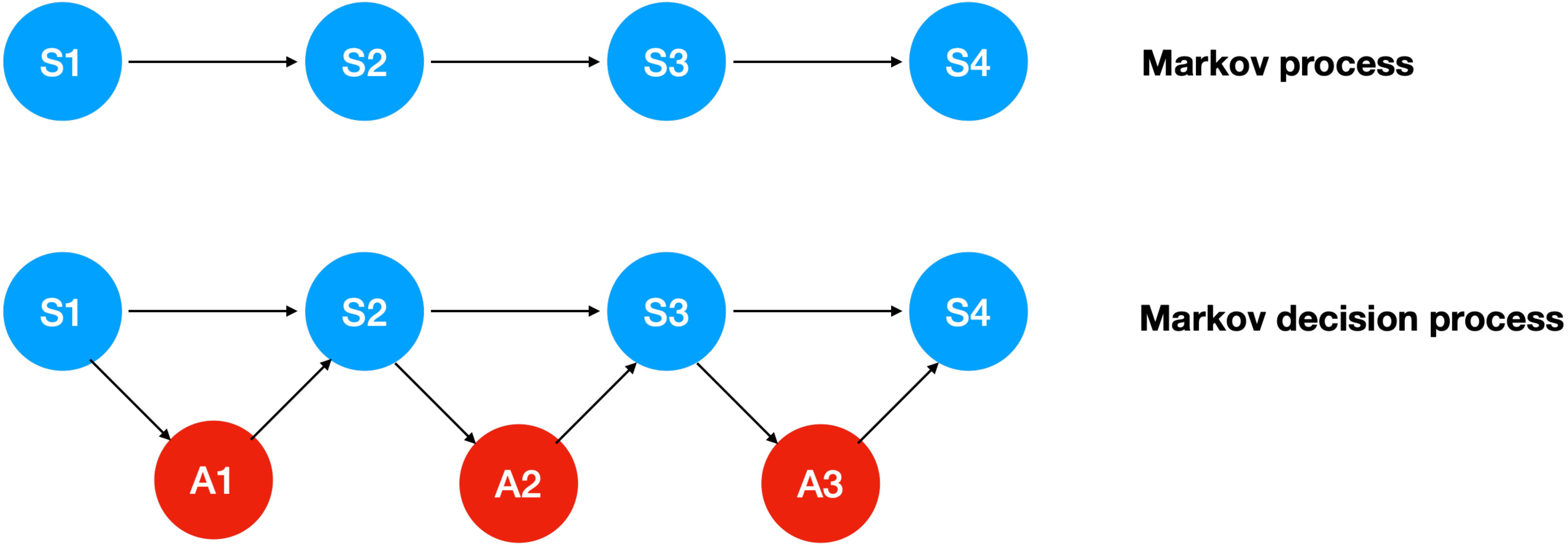
RL fundamentals: Markov process





In the Markov process, the current state is only dependent on its previous state.

RL fundamentals: Markov decision process



- The Markov decision process (MDP) is a math framework for modelling decision
- The agent takes an action according to the current state.
- The current action and the current state will cause the next state.

RL fundamentals: key concepts

MDP trajectory

$$\tau = \{(s_t, a_t)\}_{t=0}^{\infty}$$

$$\pi : S \to A$$

Policy

$$f : S \times A \to S$$

Model

Formulations of MDP:

- A MDP trajectory is a sequence of tuples of (State, Action).
- The policy is a function mapping from the state to the action.
- The (world) **model** is a function mapping from (State, Action) to State.

RL fundamentals: Policy

$$\pi : S \rightarrow A$$

Policy

$$a_t = \mu(s_t)$$

Deterministic policy

$$a_t \sim \pi(\cdot|s_t)$$

Stochastic policy

- The **policy** indicates how the agents makes a decision.
- The policy can be deterministic or stochastic.
- With a deterministic world model, the MDP trajectory is determined by the policy.

RL fundamentals: Reward, Return

The **reward** $r_t = r(s_t, a_t)$

- The reward describes what the agent should learn.
- The reward function is designed by us, and is not necessarily differentiable.
- At each time, a reward is assigned to the agent to evaluate the action taken.

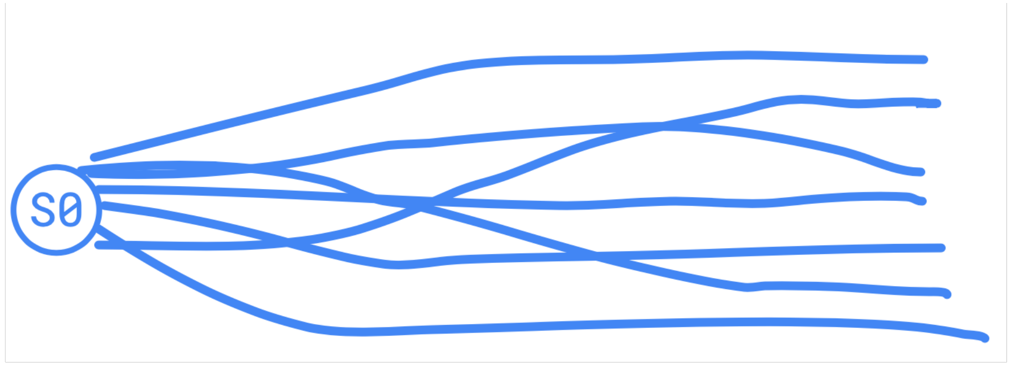
The **return** $R(\tau) = \sum_{t=0}^{\infty} \gamma^t r_t$

- The return accumulates all rewards.
- The return indicates the quality of the entire trajectory.

RL fundamentals: Value Function

$$V(s) = \mathbb{E}_{\tau \sim \pi}[R(\tau) | s_0 = s]$$

$$V^*(s) = \max_{\pi} \mathbb{E}_{\tau \sim \pi}[R(\tau) | s_0 = s]$$



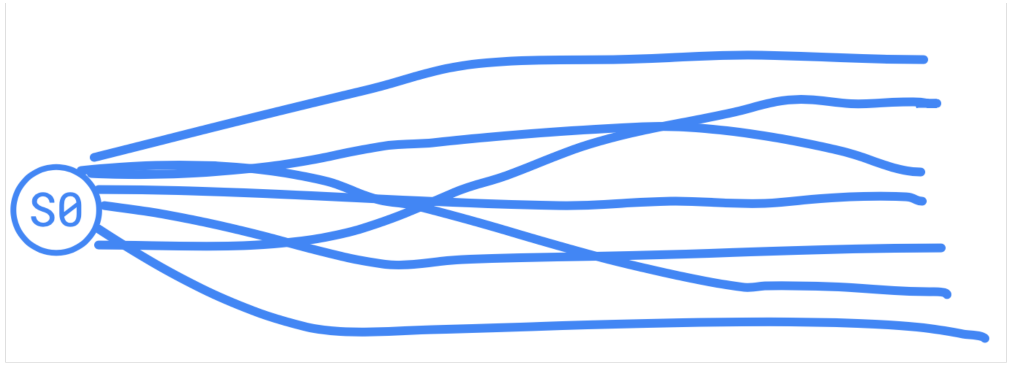
A set of trajectories starting from S0, with the same policy.

- The **value function** is the *expected return* over all trajectories provided by a policy.
- An optimal policy maximizes the **value function** at each state.
- We normally don't know this value function, but can approximate it by empirical average.

RL fundamentals: Q-function

$$Q(s,a) = \mathbb{E}_{\tau \sim \pi}[R(\tau) | s_0 = s, a_0 = a]$$

$$Q^*(s,a) = \max_{\pi} \mathbb{E}_{\tau \sim \pi}[R(\tau) | s_0 = s, a_0 = a]$$

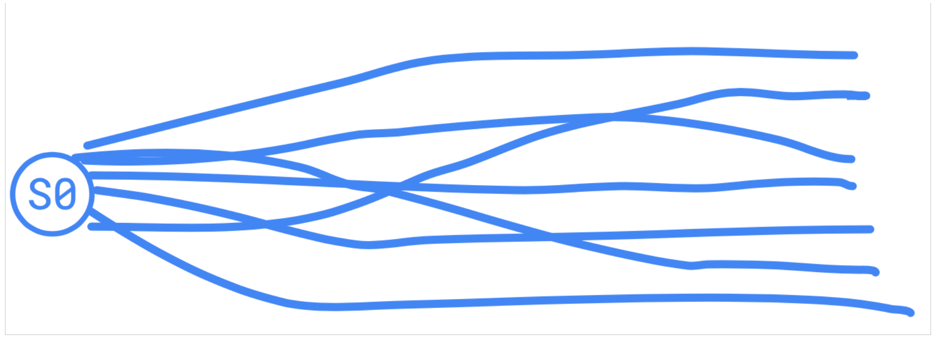


We can calculate a specific Q-function for every trajectory.

- The **Q-function** evaluates the quality of an action based on a state.
- The **value function** is the expected **Q-function** w.r.t. all actions.
- Given a MDP trajectory, we can approximate the **Q-function** by accumulating the rewards.

RL fundamentals: Advantage function

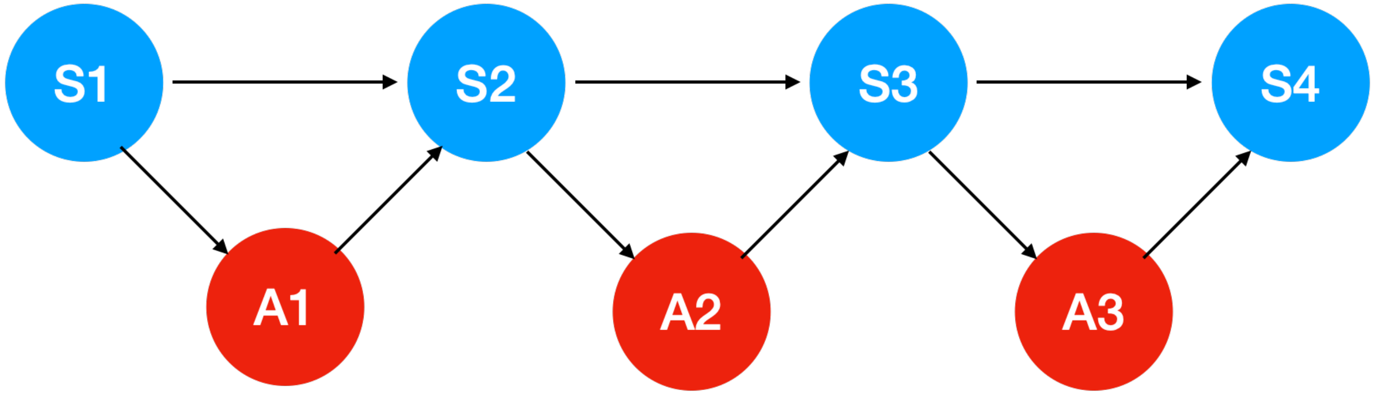
$$A(s, a) = Q(s, a) - V(s)$$



With an optimal policy, no trajectory has more advantages than others.

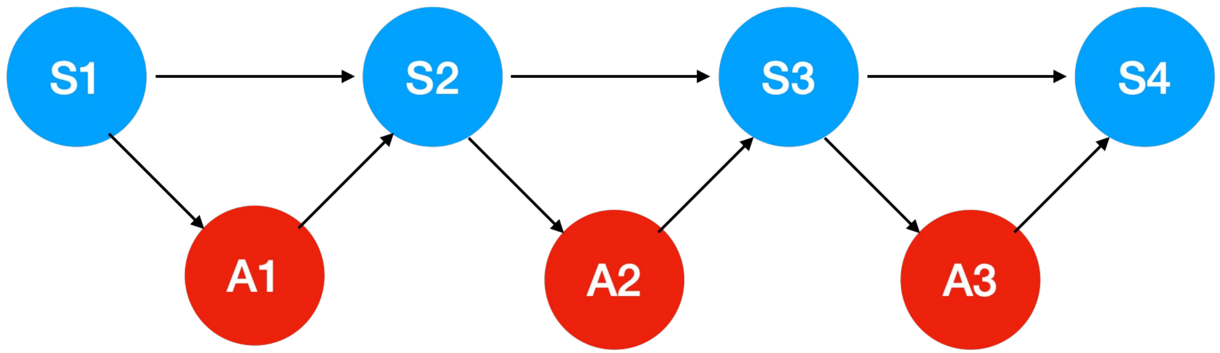
- The **advantage function** indicates how much an action is better than expected.
- If a policy is optimal, all actions based on the state are equally optimal.
- In this case, the expected advantage function w.r.t. actions is zero.

RL fundamentals: Summary



- Based on a **policy**, an agent in an environment produces a time sequence of (**state**, **action**)-s.
- At each step, the agent is assigned with a **reward**.
- We can accumulate the rewards to get the **return**.
- The expected return based on a **state** is the **value function.**
- The expected return based on a **state** and an action is the **Q-function**.
- Their difference is the **advantage function**.

RL fundamentals: Policy Optimization



MDP factorisation

$$p(\tau|\pi_\phi) = p(s_0) \prod_{t=0}^{\infty} \underbrace{p(s_{t+1}|s_t, a_t)}_{\text{Model}} \underbrace{\pi_\phi(a_t|s_t)}_{\text{Policy}}$$

Expected return

$$J(\pi_\phi) = \mathbb{E}_{\tau\sim\pi_\phi}[R(\tau)] = \int_\tau p(\tau|\pi_\phi) \underbrace{R(\tau)}_{\text{Return}} d\tau$$

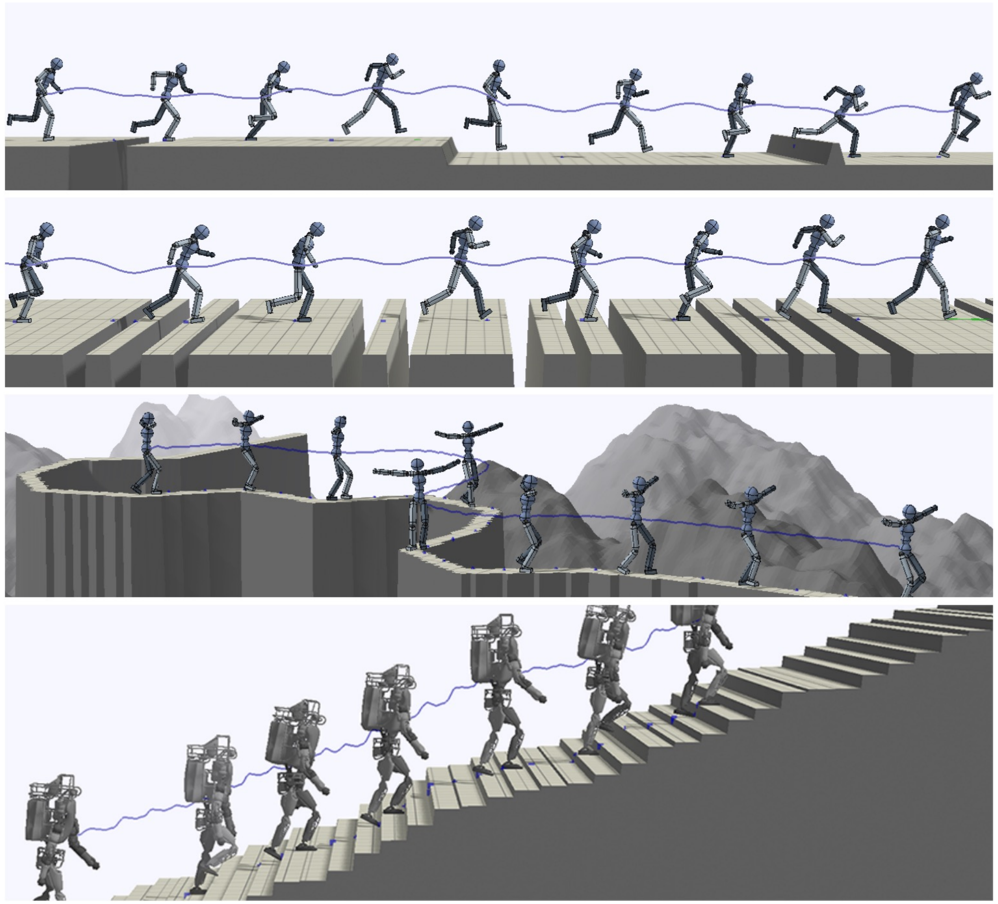
$$\phi^* = arg\max_\phi J(\pi_\phi)$$

- The **policy** can be parameterized by neural networks.
- The optimal policy is obtained by maximizing the expected return *J*.
- When optimum is reached, no actions are statistically better than expected.

Using RL for character animation

- Vast number of ways for an agent to move in a scene depending on its mass, height, skeleton etc.
- Difficult to collect all training data.

- Let an agent explore the scene with different actions.
- Physics based simulation provides signal to the agent for learning.



DeepMimic: Simplified formulation

- **State $\boldsymbol{s}$:**
  - relative rotation q between node and pelvis (root node)
  - linear velocity v
  - angular velocity ω
- **Policy, Action, Goal $(\boldsymbol{\varphi}, \boldsymbol{a}, \boldsymbol{g})$:**
  - Policy modelled by a neural network, $\varphi(s|g, H)$
  - Action is sampled from a gaussian whose centre is predicted by policy

$$a = N(\varphi(s|g, H), I)$$

  - H is the scene, encoded as terrain height at root. g is the goal location.
- **Reward:**
  - How well does agent state match the simulator state.

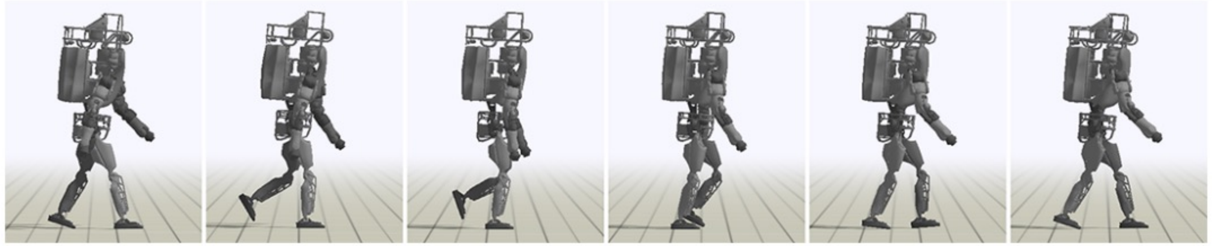
DeepMimic: Optimizing reward

$$r_t = \underbrace{\exp(-2\sum_j ||q_t^j \cdot \hat{q}_t^j||^2)}_{\text{joint rotation}} + \underbrace{\exp(-2\sum_j ||\dot{q}_t^j - \dot{\hat{q}}_t^j||^2)}_{\text{angular velocity}} + \underbrace{\exp(-2\sum_j ||p_t^j - \hat{p}_t^j||^2)}_{\text{joint position}}$$

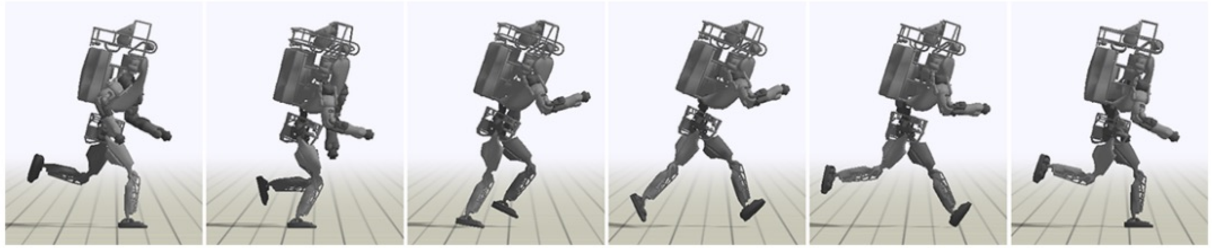
Reward penalises predicted:

- joint rotation(q_t^j)
- angular velocity($\dot{q}_t^j$)
- joint position(p_t^j)

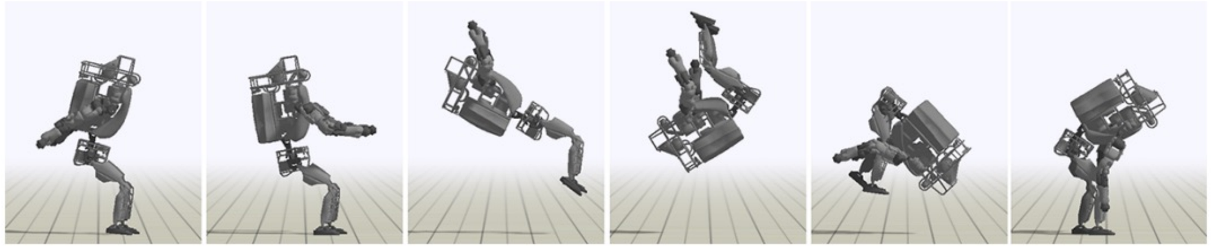
Agent can learn diverse motions using PBS



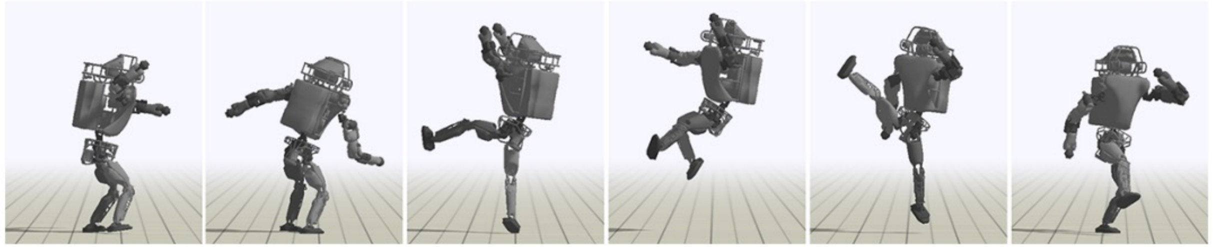
(a) Atlas: Walk



(b) Atlas: Run



(c) Atlas: Backflip



(d) Atlas: Spinkick

Takeaways

- Synthesising human-object interactions is hard and requires a large amount of supervised data (COUCH, NSM).
- Key idea is to use **conditional generation** to predict future motion as a function of **past motion, scene, goal and contacts**.

- Collecting large amounts of supervised data is hard. We can leverage simulations to learn physics behind the interactions (DeepMimic).
- Key idea is to learn a policy (modelled by NN) that predicts future **action**, with the advantage being that reward **need not be differentiable** and **we learn by doing**.