# Grounded Action Transformation for Robot Learning in Simulation

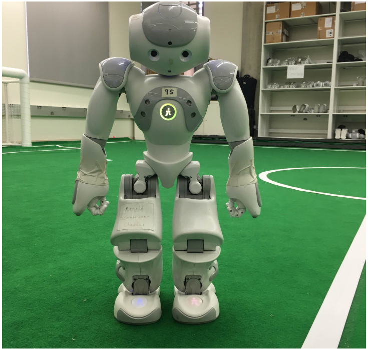
Josiah Hanna and Peter Stone

# Reinforcement Learning for Physical Robots

Learning on physical robots:

- Not data-efficient.
- Requires supervision.
- Manual resets.
- Robots break.
- Wear and tear make learning non-stationary.

Not an exhaustive list...



# Reinforcement Learning in Simulation

Learning in simulation:

- Thousands of trials in parallel.
- No supervision and automatic resets.
- Robots never break or wear out.

# Reinforcement Learning in Simulation

Learning in simulation:

- Thousands of trials in parallel.
- No supervision and automatic resets.
- Robots never break or wear out.

Policies learned in simulation often fail in the real world.

# Notation

Environment $E = \langle \mathcal{S}, \mathcal{A}, c, P \rangle$

- Robot in state $s \in \mathcal{S}$ chooses action $a \in \mathcal{A}$ according to policy $\pi$.
  - Parameterized $\pi_{\boldsymbol{\theta}}$ denoted $\boldsymbol{\theta}$
- Environment, $E$, responds with a new state $S_{t+1} \sim P(\cdot|s, a)$.
- Cost function $c$ defines a scalar cost for each $(s, a)$.
- Goal is to find $\boldsymbol{\theta}$ which minimizes:

$$J(\boldsymbol{\theta}) := \mathbb{E}_{S_1, A_1, \ldots, S_L, A_L} \left[ \sum_{t=1}^{L} c(S_t, A_t) \right]$$

# Learning in Simulation

Simulator $E_{\texttt{sim}} = \langle \mathcal{S}, \mathcal{A}, c, P_{\texttt{sim}} \rangle$.

- Identical to $E$ but different dynamics (transition function).

# Learning in Simulation

Simulator $E_{\mathtt{sim}} = \langle \mathcal{S}, \mathcal{A}, c, P_{\mathtt{sim}} \rangle$.

- Identical to $E$ but different dynamics (transition function).

$$J_{\mathtt{sim}}(\boldsymbol{\theta}') > J_{\mathtt{sim}}(\boldsymbol{\theta}_0) \nRightarrow J(\boldsymbol{\theta}') > J(\boldsymbol{\theta}_0)$$

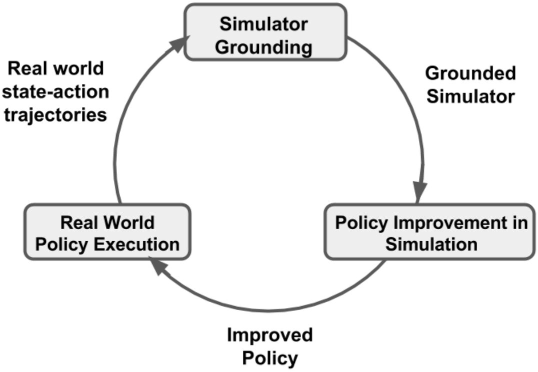
Goal: Learn $\boldsymbol{\theta}$ in simulation that also works on physical robot.

# Grounded Simulation Learning

*Grounded Simulation Learning (GSL) is a framework for robot learning in simulation by modifying the simulator with real world data so that policies learned in simulation work in the real world [**?**].*

1. Execute $\boldsymbol{\theta}_0$ on physical robot.
2. Ground simulator so $\boldsymbol{\theta}_0$ produces similar trajectories in simulation.
3. Optimize $J_{\texttt{sim}}(\boldsymbol{\theta})$ to find better $\boldsymbol{\theta}'$.
4. Test $\boldsymbol{\theta}'$ on the physical robot.
5. $\boldsymbol{\theta}_0 := \boldsymbol{\theta}'$ and repeat.

# Grounded Simulation Learning



## Grounding the Simulator

Assume $P_{\texttt{sim}}$ is parameterized by $\phi$.

$d$: Any measure of similarity between state transition distributions

Robot executes $\boldsymbol{\theta}_0$ and records dataset $\mathcal{D}$ of $(S_t, A_t, S_{t+1})$ transitions.

$$\phi^\star = \operatorname*{argmin}_{\phi} \sum_{(S_t, A_t, S_{t+1}) \in \mathcal{D}} d\left(P(\cdot|S_t, A_t), P_\phi(\cdot|S_t, A_t)\right)$$

# Grounding the Simulator

Assume $P_{\texttt{sim}}$ is parameterized by $\phi$.

$d$: Any measure of similarity between state transition distributions

Robot executes $\boldsymbol{\theta}_0$ and records dataset $\mathcal{D}$ of $(S_t, A_t, S_{t+1})$ transitions.

$$\phi^\star = \operatorname*{argmin}_{\phi} \sum_{(S_t, A_t, S_{t+1}) \in \mathcal{D}} d\left(P(\cdot|S_t, A_t), P_\phi(\cdot|S_t, A_t)\right)$$

$$\boxed{\text{How to define } \phi?}$$

# Advantages of GSL

1. No random-access simulation modification required.
2. Leaves underlying policy optimization unchanged.
3. Efficient simulator modification.

# Guided Grounded Simulation Learning

Farchy et al. presented a GSL algorithm and demonstrated a 26.7% improvement in walk speed on a Nao.

Two limitations of existing approach:

1. Modification relied on assumption that desired joint positions achieved instantaneously in simulation.
2. Used expert knowledge to select which components of $\boldsymbol{\theta}$ could be learned.

# Grounded Action Transformations

Goal: Eliminate simulator-dependent assumption of earlier work.

$$\phi^\star = \operatorname*{argmin}_{\phi} \sum_{(S_t, A_t, S_{t+1}) \in \mathcal{D}} d\left(P(\cdot|S_t, A_t), P_\phi(\cdot|S_t, A_t)\right)$$

Replace robot's action $\mathbf{a}_t$ with an action that produces a more "realistic" transition.
Learn this action as a function $g_\phi(\mathbf{s}_t, \mathbf{a}_t)$.

# Grounded Action Transformation

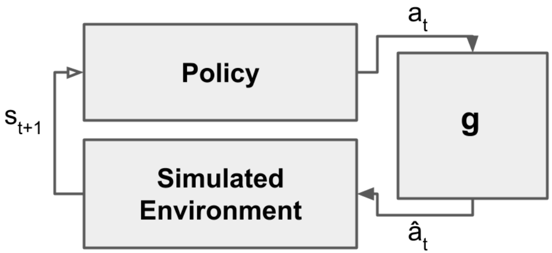



Figure : Modifiable simulator induced by GAT.

# Grounded Action Transformation

$\mathcal{X}$: the set of robot joint configurations.

Learn two functions:

- Robot's dynamics: $f : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{X}$
- Simulator's inverse dynamics: $f_{\texttt{sim}}^{-1} : \mathcal{S} \times \mathcal{X} \rightarrow \mathcal{A}$.

Replace robot's action $\mathbf{a}_t$ with $\hat{\mathbf{a}}_t := f_{\texttt{sim}}^{-1}(\mathbf{s}_t, f(\mathbf{s}_t, \mathbf{a}_t))$.

# Grounded Action Transformations

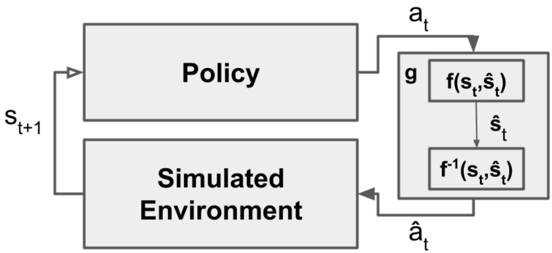



Figure : Modifiable simulator induced by GAT.

# GAT Implementation

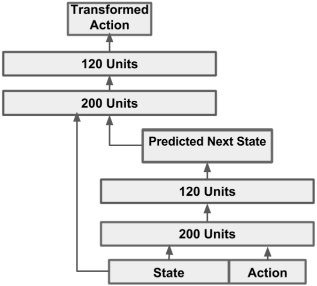
$f$ and $f_{\texttt{sim}}^{-1}$ learned with supervised learning.

- Record sequence $S_t, A_t, \ldots$ on robot and in simulation.
- Supervised learning of $g$:
  - $f_{\texttt{sim}}^{-1} : (S_t, A_t) \rightarrow X_{t+1}$
  - $f : (S_t, X_{t+1}) \rightarrow A_t$

Smooth modified actions:

$$g(\mathbf{s}_t, \mathbf{a}_t) := \alpha f_{\texttt{sim}}^{-1}(\mathbf{s}_t, f(\mathbf{s}_t, \mathbf{a}_t)) + (1 - \alpha)\mathbf{a}_t$$

# Supervised Implementation




- Forward model trained with 15 real world trajectories of 2000 time-steps.
- Inverse model trained with 50 simulated trajectories of 1000 time-steps.

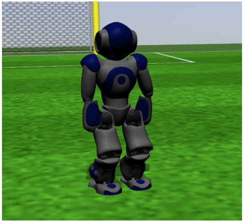
# Empirical Results

Applied GAT to learning fast bipedal walks for the Nao robot.

- Task: Walk forward towards a target.
- $\boldsymbol{\theta}_0$: University of New South Wales Walk Engine.
- Simulator: SimSpark Robocup3D Simulator and OSRF Gazebo Simulator.
- Policy optimization with CMA-ES stochastic search method.

# Empirical Results

(a) Softbank Nao



(b) Gazebo Nao

(c) SimSpark Nao

# Empirical Results

# Empirical Results

Simulation to Nao:

| Method | Velocity (cm/s) | % Improve |
|---|---|---|
| Initial policy | 19.52 | 0.0 |
| SimSpark, first iteration | 26.27 | 34.58 |
| SimSpark, second iteration | 27.97 | 43.27 |
| Gazebo, first iteration | 26.89 | 37.76 |

SimSpark to Gazebo:

| Method | % Improve | Failures | Best Gen. |
|---|---|---|---|
| No Ground | 11.094 | 7 | 1.33 |
| Noise-Envelope | 18.93 | 5 | 6.6 |
| GAT | **22.48** | **1** | 2.67 |

# Conclusion

Contributions:

1. Introduced Grounded Action Transformations algorithm for simulation transfer.
2. Improved walk speed of Nao robot by over 40 % compared to state-of-the-art walk engine.

Future Work:

- Extending to other robotics tasks and platforms.
- When does grounding actions work and when does it not?
- Reformulating learning $g$:
  - $f$ and $f_{\texttt{sim}}^{-1}$ minimize one-step error but we actually care about error over sequences of states and actions.

**Thanks for your attention!**
**Questions?**

Alon Farchy, Samuel Barrett, Patrick MacAlpine, and Peter Stone.
Humanoid robots learning to walk faster: From the real world to simulation and back.
In *Twelth International Conference on Autonomous Agents and Multiagent Systems*, 2013.