Learning Robot Motion through User Intervention and Policy Search

Aleksandar Jevtić, Adrià Colomé, Guillem Alenyà and Carme Torras

Institut de Robòtica i Informàtica industrial, CSIC-UPC, Barcelona, Spain

ABSTRACT

Reinforcement Learning (RL) through policy search is well suited for robot learning of manipulation tasks but requires a high number of policy updates to produce a satisfactory robot behavior. We propose a method that exploits **user intervention as a strategy for policy search updates**, which significantly **reduces the number of necessary updates** and allows **users to tailor robot behavior** according to their needs. We developed an interface for **gesture recognition and hand motion following** that allows a user to select and modify a desired manipulation task segment and speed up the learning process through a series of interventions. We implemented the method on a **single-arm and dual-arm robot** and applied it to a manipulation task on the laboratory tabletop. We compared our method with the conventional policy search strategy in simulation; the results show that the number of necessary policy search updates was significantly reduced by user intervention.

CONTACT

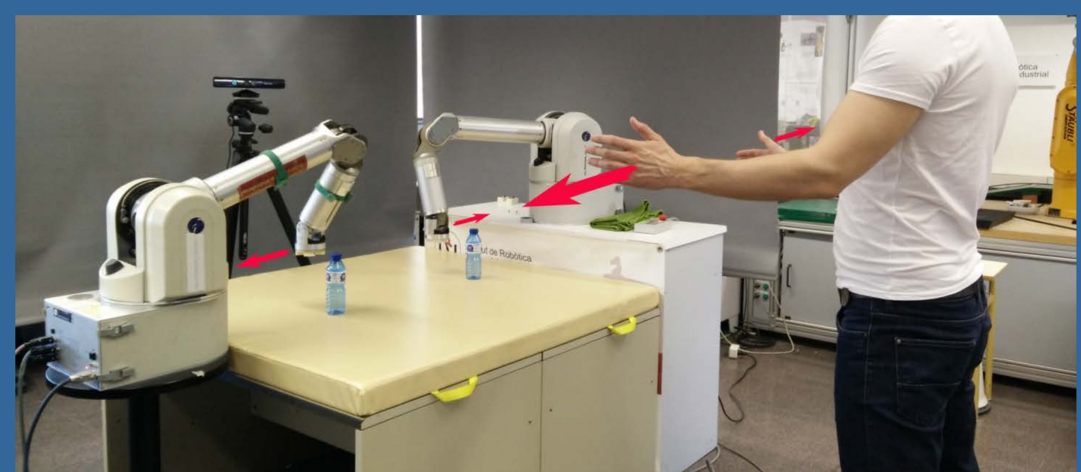
Aleksandar Jevtić
Institut de Robòtica i Informàtica industrial, CSIC-UPC
08028 Barcelona, Spain
Email: ajevtic@iri.upc.edu
Website: www.iri.upc.edu

PROPOSED INTERACTIVE LEARNING FRAMEWORK

Robot executes initial trajectory → **User modifies initial trajectory** → **Robot stores new trajectory and reward value** → **Apply policy search algorithm**

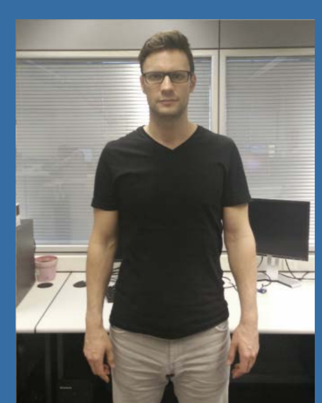
N user interventions

Experimental setup with two obstacles and presentation of the initial robots' trajectories



User intervention using hand gestures to guide the robots around the obstacles

Static gesture recognition



Hand motion following

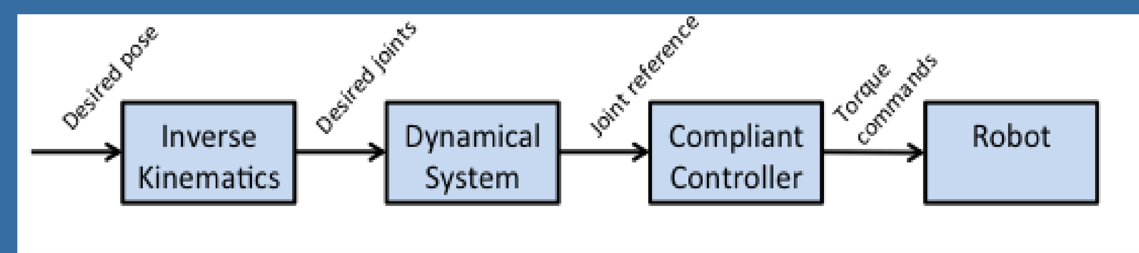
New robot position is obtained as follows:

$$\mathbf{p}_{goal} = \mathbf{p}_{cut} + (\mathbf{p}_{hand} - \mathbf{p}_0), p \in \mathbb{R}^2$$

$\mathbf{p}_{cut}$ – starting point of a robot's end-effector during the user's intervention.

ALGORITHMS

Robot control



User intervention algorithm and reinforcement learning

Algorithm 1 User intervention

```
set state to "EXECUTE";
go to first trajectory point;
while not end of trajectory do
    go to next trajectory point;
    if hand raised then
        set state to "FOLLOW";
        store trajectory cut point;
        while in "FOLLOW" state do
            follow user's hands;
            if hand raised then
                set state to "EXECUTE";
            end if
        end while
        find closest trajectory connect point;
    end if
end while
```

Algorithm 2 Reinforcement learning of new trajectories

1: Input: User-generated trajectories $\boldsymbol{\tau}_k = \{x_k^1, ..., x_k^{N_t}\}$, for $\forall k$; cut and connect points indexes I_k^{cut}, I_k^{conn}, for $\forall k$; rewards of user-generated trajectories R_k, for $\forall k$

2: Find common unmodified part of the initial trajectory for all rollouts: $I^{cut} = \min_{k=1..N_{rollouts}} (I_k^{cut})$, $I^{conn} = \max_{k=1..N_{rollouts}} (I_k^{conn})$

3: Extract segment of trajectory that will be modified by RL, for all rollouts: $\mathbf{T}_k = \{x_k^{I^{cut}}, ..., x_k^{I^{conn}}\}$, for $\forall k$

4: Apply time alignment to $\mathbf{T}_k$, for $\forall k$, using trajectory from the first rollout as reference; resample to have same number of points in all trajectories

5: Apply REPS to compute relative importance weight for each rollout: $w_k = \text{REPS}(\mathbf{R})$

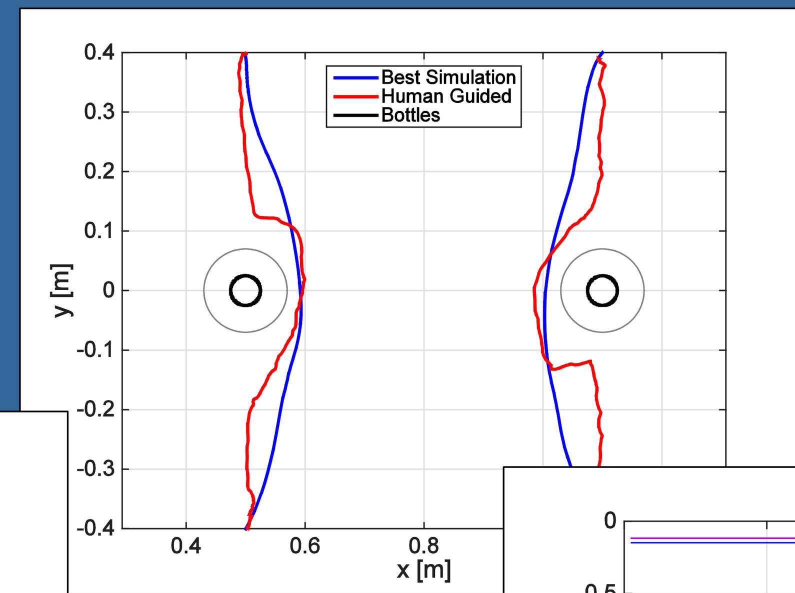
6: Obtain new trajectory by a weighted sum of the time-aligned, resampled rollouts:

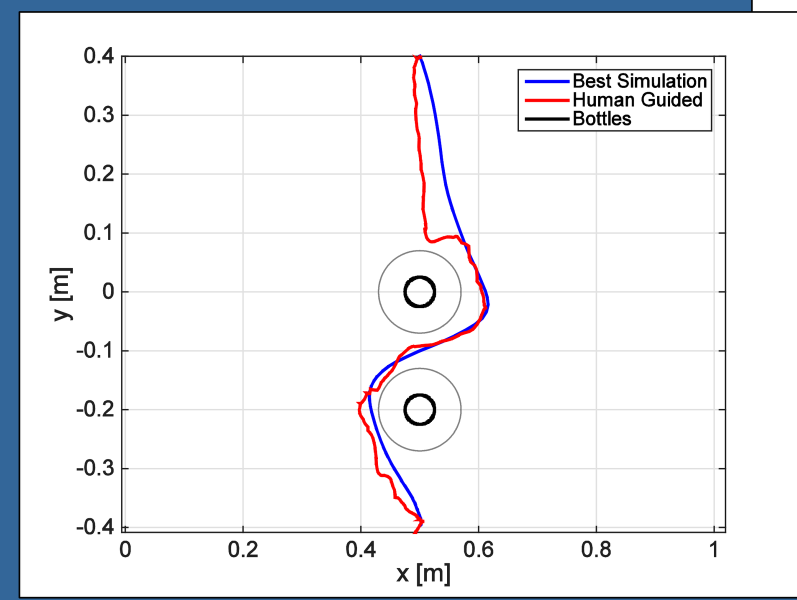
$$x_{\text{new}}^t = \sum_{k=1}^{N_{rollouts}} w_k x_k^t, \forall t = I^{cut}..I^{conn}$$

7: Merge unmodified trajectory segments from step 2, $\{x^1, ..., x^{I^{cut}-1}\}$, $\{x^{I^{conn}+1}, ..., x^{N_t}\}$ same for $\forall k$, with modified trajectory segment from step 6 to obtain the RL-modified trajectory:

$$\boldsymbol{\tau}_{\text{new}} = \{x^1, ..., x^{I^{cut}-1}, x_{\text{new}}^{I^{cut}}, ..., x_{\text{new}}^{I^{conn}}, x^{I^{conn}+1}, ..., x^{N_t}\}$$

RESULTS

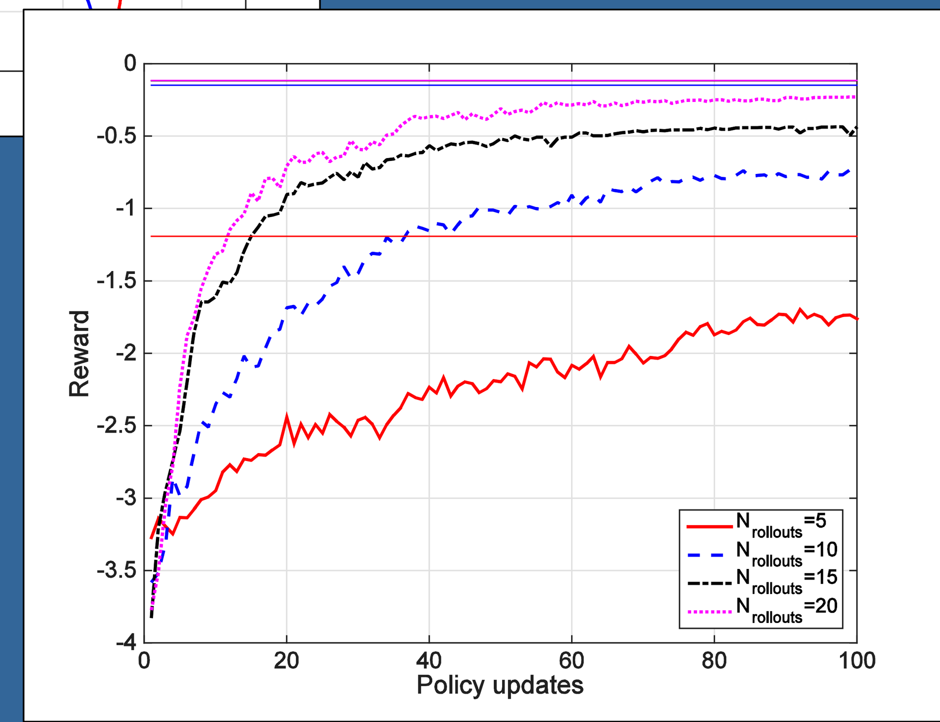




Reward function

$$R = -\alpha \cdot D_{total} - B$$

B is the number of knocked-down obstacles, D_{total} trajectory length, and α is a relative weight.



REPS algorithm uses Kullback-Liebler (KL) divergence as an indicator of difference between two probability distributions p, q over a random variable x:

$$KL(p||q) = \int p(x) \log \frac{p(x)}{q(x)} dx$$

Given the previous policy $q(\Theta)$, the new policy $\pi(\Theta)$ is obtained by adding a KL-Divergence bound ϵ between the new and old policies to the optimization of the expected reward:

$$\pi^* = argmax_\pi \int \pi(\theta) R(\theta) d\theta \; s.t. \epsilon \geq KL(\pi(\theta)||q(\theta))$$

$$1 = \int \pi(\theta) d\theta$$

where Θ are the trajectory parameters, $R(\Theta)$ is their associated reward, and $\pi(\Theta)$ is the probability, according to the policy π, of having such parameters.

REFERENCES

1. M. Deisenroth, G. Neumann and J. Peters, A survey on policy search for robotics, Foundations and Trends in Robotics, vol. 2, no. 1-2, pp. 1-142, 2013.
2. A. Jevtić, G. Doisy, Y. Parmet and Y. Edan, Comparison of interaction modalities for mobile indoor robot guidance: Direct physical interaction, person following, and pointing control, IEEE Transactions on Human-Machine Systems, vol. 45, no. 6, pp. 653-663, 2015.
3. A. Colomé, A. Planells and C. Torras, A friction-model-based framework for reinforcement learning of robotic tasks in non-rigid environments, 2015 IEEE International Conference on Robotics and Automation, pp. 5649-5654, 2015.
4. J. Peters, K. Mülling and Y. Altün, Relative entropy policy search, Twenty-Fourth National Conference on Articial Intelligence, pp. 182-189, 2011.