Micro-data policy search

Learning in a handful of trials

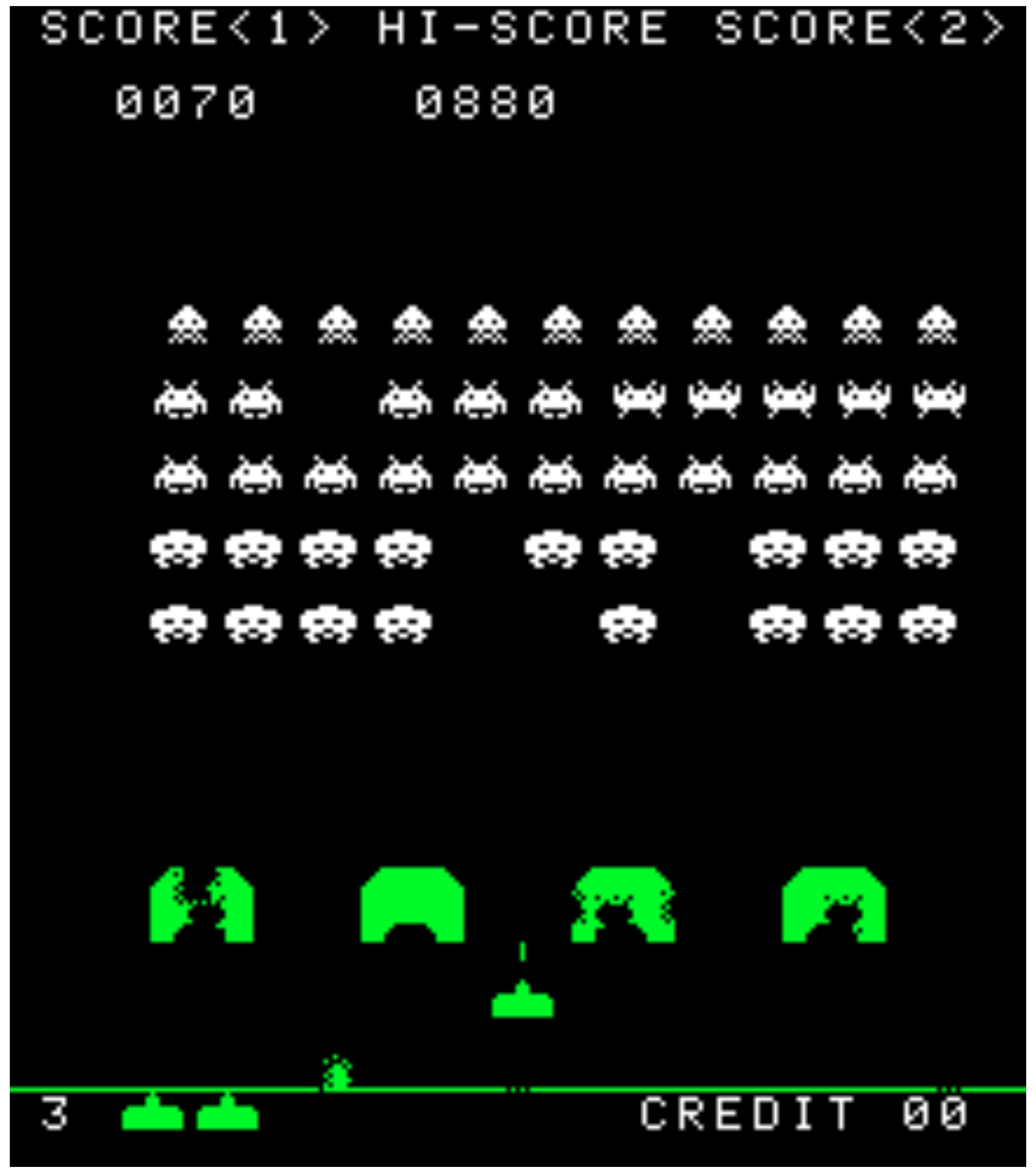
JB Mouret & Konstantinos Chatzilygeroudis

@jbmouret @jb_mouret

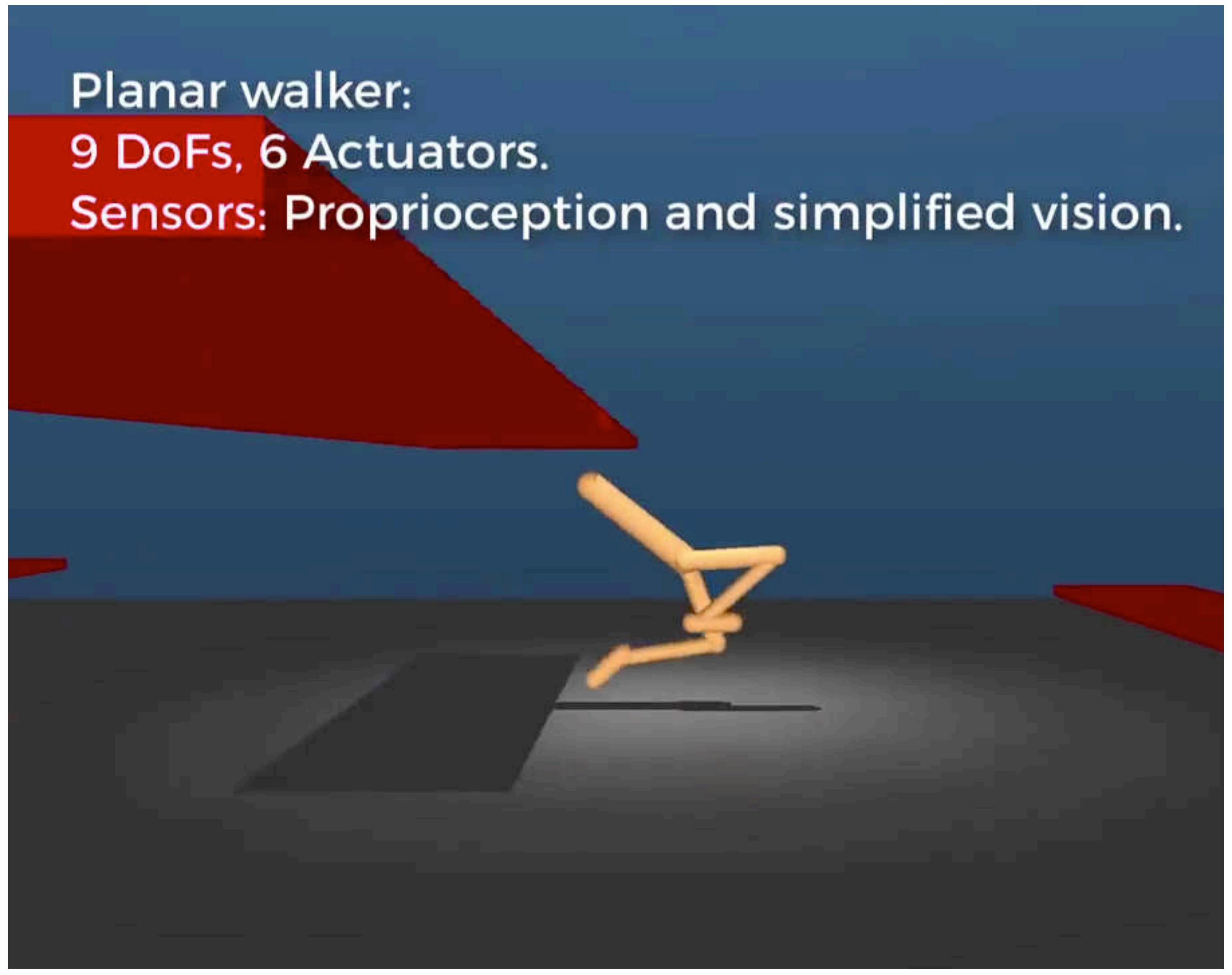
jean-baptiste.mouret@inria.fr — https://members.loria.fr/jbmouret

(Deep) Reinforcement Learning is nice but…

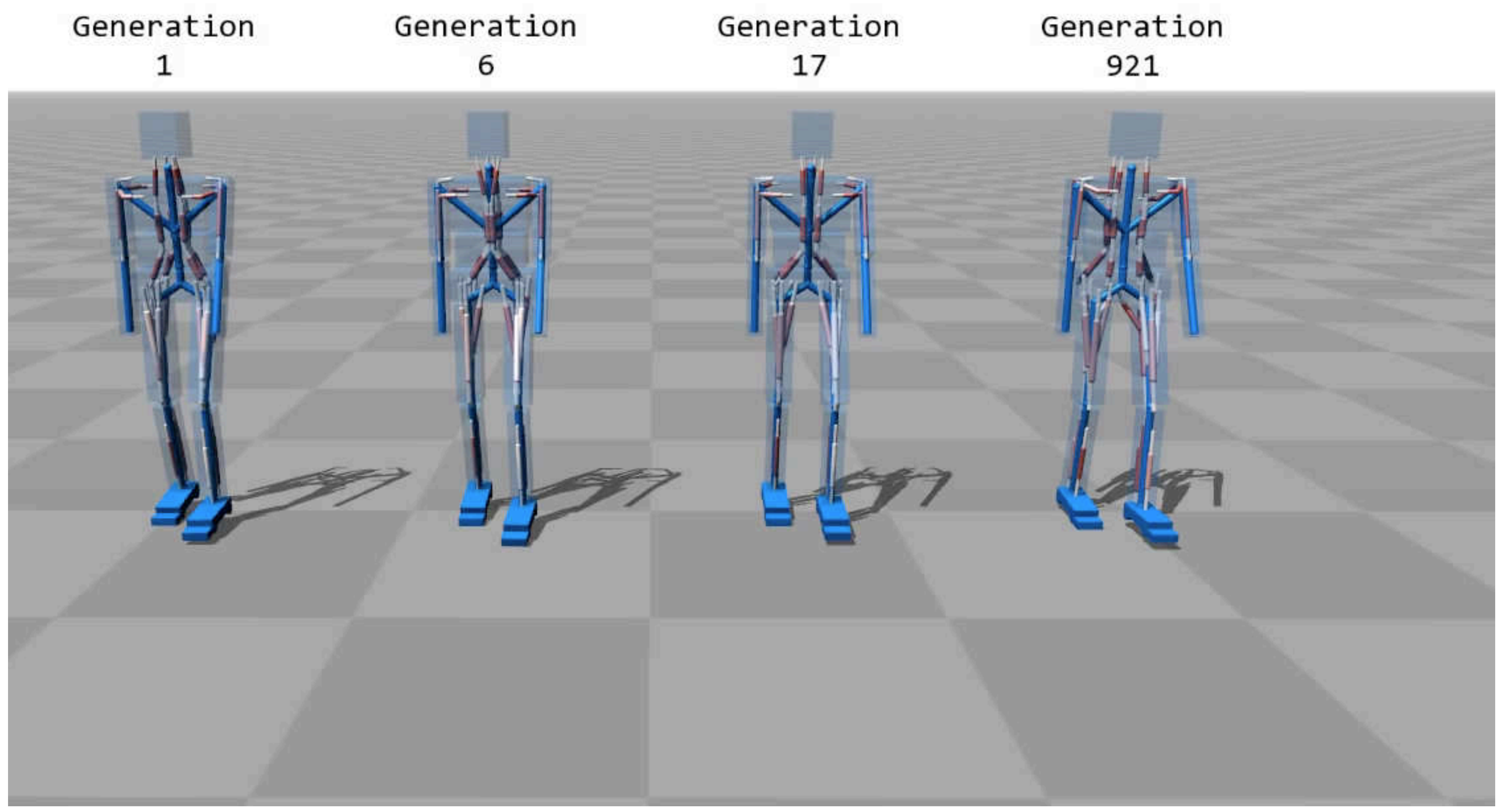
… it needs millions of time-steps to find good solutions (data-efficiency)



DQN for Atari games :
38 days (original paper)



DPPO / PPO
~10 million training steps
~ 500k seconds at 20Hz (140 hours, or 6 days)



Algorithm: CMA-ES / ~ 20000 episodes

Mnih, V., et al (2015). Human-level control through deep reinforcement learning. *nature*, *518*(7540), 529-533.
Heess N, *et al.* Emergence of locomotion behaviours in rich environments. arXiv preprint arXiv:1707.02286. 2017 Jul 7. (Google DeepMind)
Geijtenbeek, T., van de Panne, M., & van der Stappen, A. F. (2013). "*Flexible muscle-based locomotion for bipedal creatures*". ACM Transactions on Graphics (TOG), 32(6), 206.

Can we do the same on a robot?

… if we have a lot of time & money

Levine, S., Pastor, P., Krizhevsky, A., Ibarz, J., & Quillen, D. (2018). Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, *37*(4-5), 421-436.

… this is why we use simulators!

but:

- Some systems are very expensive to simulate (e.g., fluid dynamics)
- There is (always) a reality gap: policies learned in simulation do not work well on the real system

- *Why do you want to learn?*
  - you have a model (simulator) but you cannot write a controller:
    - are you sure? (Model-Predictive Control, planning, …)

    … MPC/Planning is too expensive to run onboard: use learning (e.g., Guided Policy Search)

    … non-trivial sensors (e.g., vision): maybe use deep-RL (?)

    … model too complex for MPC/planning: maybe use deepRL (?)

No learning
… but model-based control!

Unknown damage … no model!

… this is why we use simulators!

but:

- Some systems are very expensive to simulate (e.g., fluid dynamics)
- There is (always) a reality gap: policies learned in simulation do not work well on the real system

- *Why do you want to learn a policy?*
  - you have a model (simulator) but you cannot write a controller:
    - are you sure? (Model-Predictive Control, planning, …)

    … MPC/Planning is too expensive to run onboard: use learning (e.g., Guided Policy Search)

    … non-trivial sensors (e.g., vision): maybe use deep-RL (?)

    … model too complex for MPC/planning: maybe use deepRL (?)

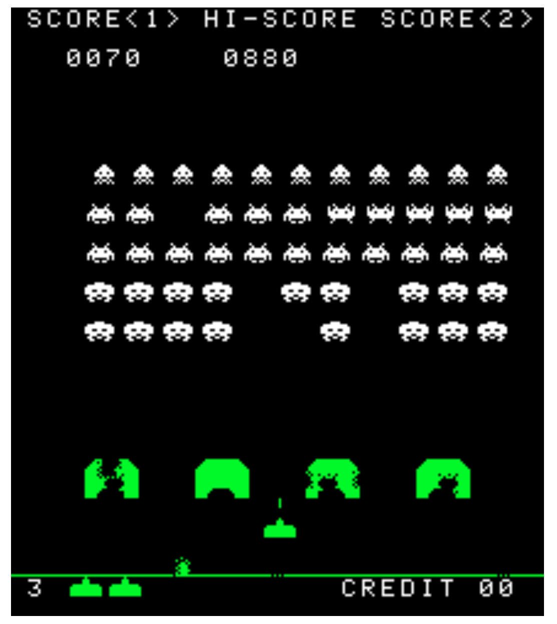
  - model/reward is (partially) unknown (e.g., damaged robot): we cannot simulate it accurately!
  - (reinforcement) learning most useful for adaptation in robotics (vs design)

- Using a simulator = having a model (possibly complex) of the system!

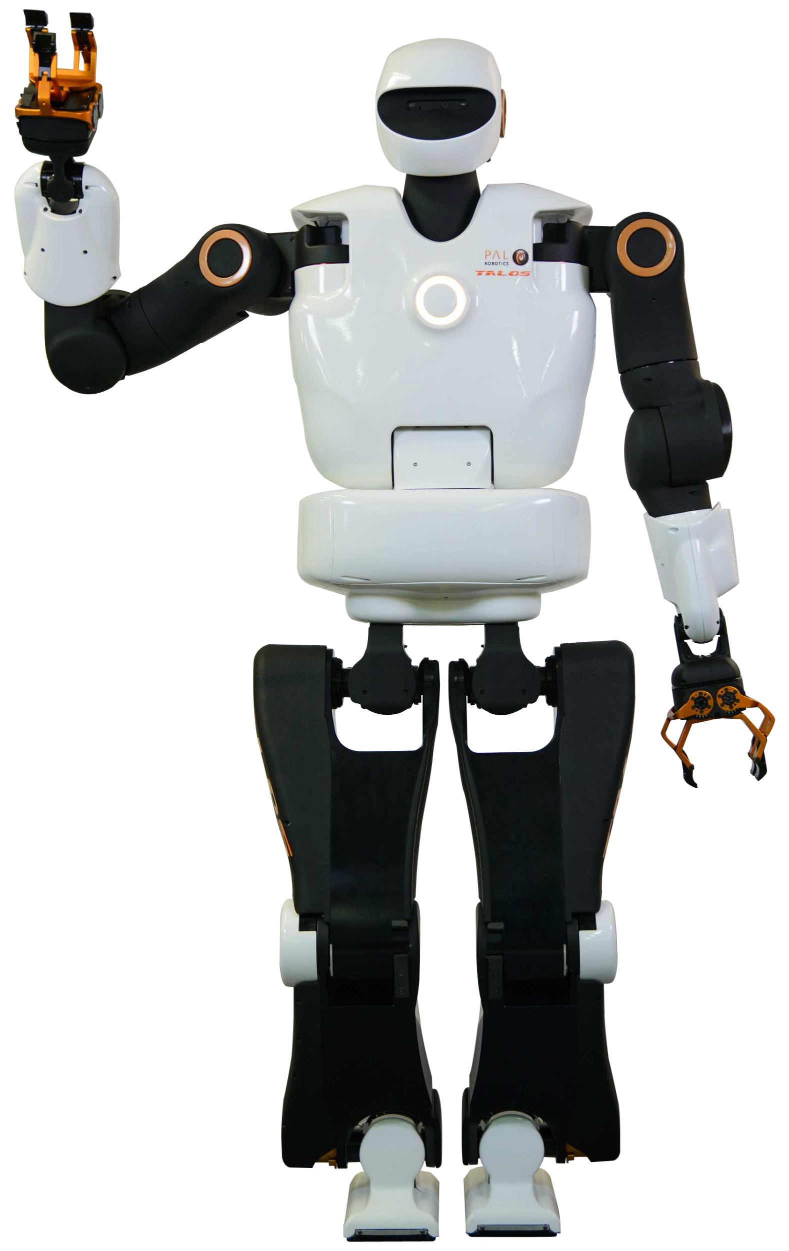
learning is most useful *when the model is unknown* *… which means no (accurate) simulator*

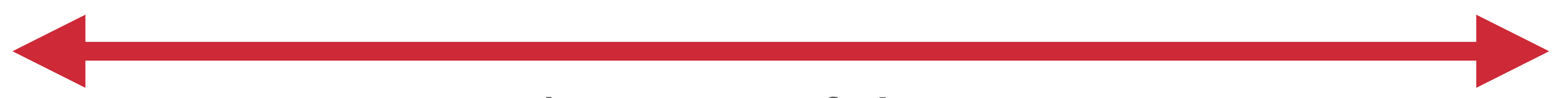
AlphaGo

4.9 million games (self-play) / 40 days



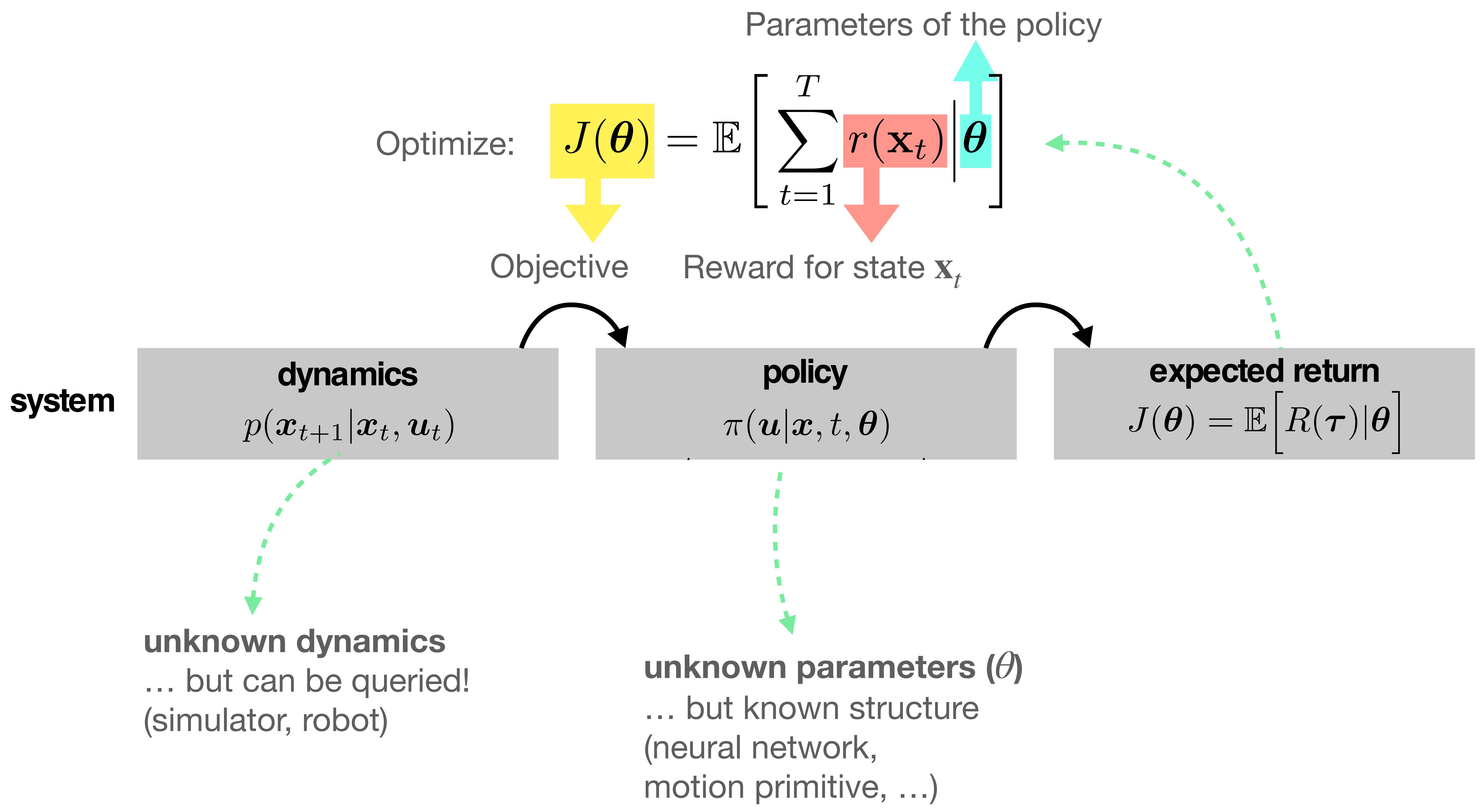
Atari games : 38 days



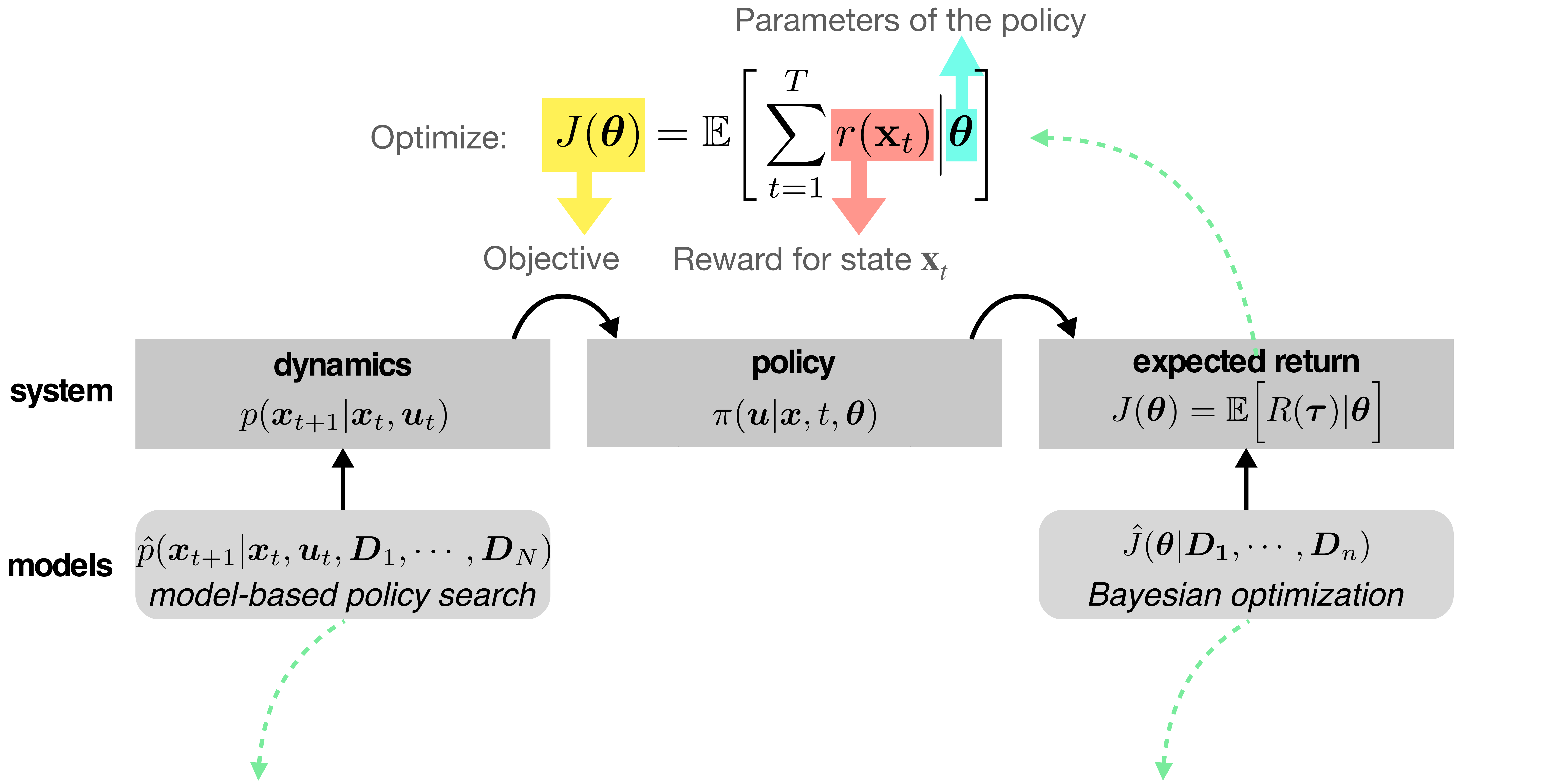


- We want to minimize the **interaction time** with the system
- We consider that pure computation time is "free"

Mouret, JB. (2016) "Micro-Data Learning: The Other End of the Spectrum." *ERCIM News*
K. Chatzilygeroudis, V. Vassiliades, F. Stulp, S. Calinon, J.-B. Mouret. (2018). A survey on policy search algorithms for learning robot controllers in a handful of trials. IEEE TRO 2020


Parameters of the policy
Optimize: $J(\boldsymbol{\theta}) = \mathbb{E}\left[\sum_{t=1}^{T} r(\mathbf{x}_t) \middle| \boldsymbol{\theta}\right]$
Objective
Reward for state $\mathbf{x}_t$
system
dynamics
$p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t)$
policy
$\pi(\boldsymbol{u}|\boldsymbol{x}, t, \boldsymbol{\theta})$
expected return
$J(\boldsymbol{\theta}) = \mathbb{E}\left[R(\boldsymbol{\tau})|\boldsymbol{\theta}\right]$
unknown dynamics
… but can be queried!
(simulator, robot)
unknown parameters (θ)
… but known structure
(neural network,
motion primitive, …)

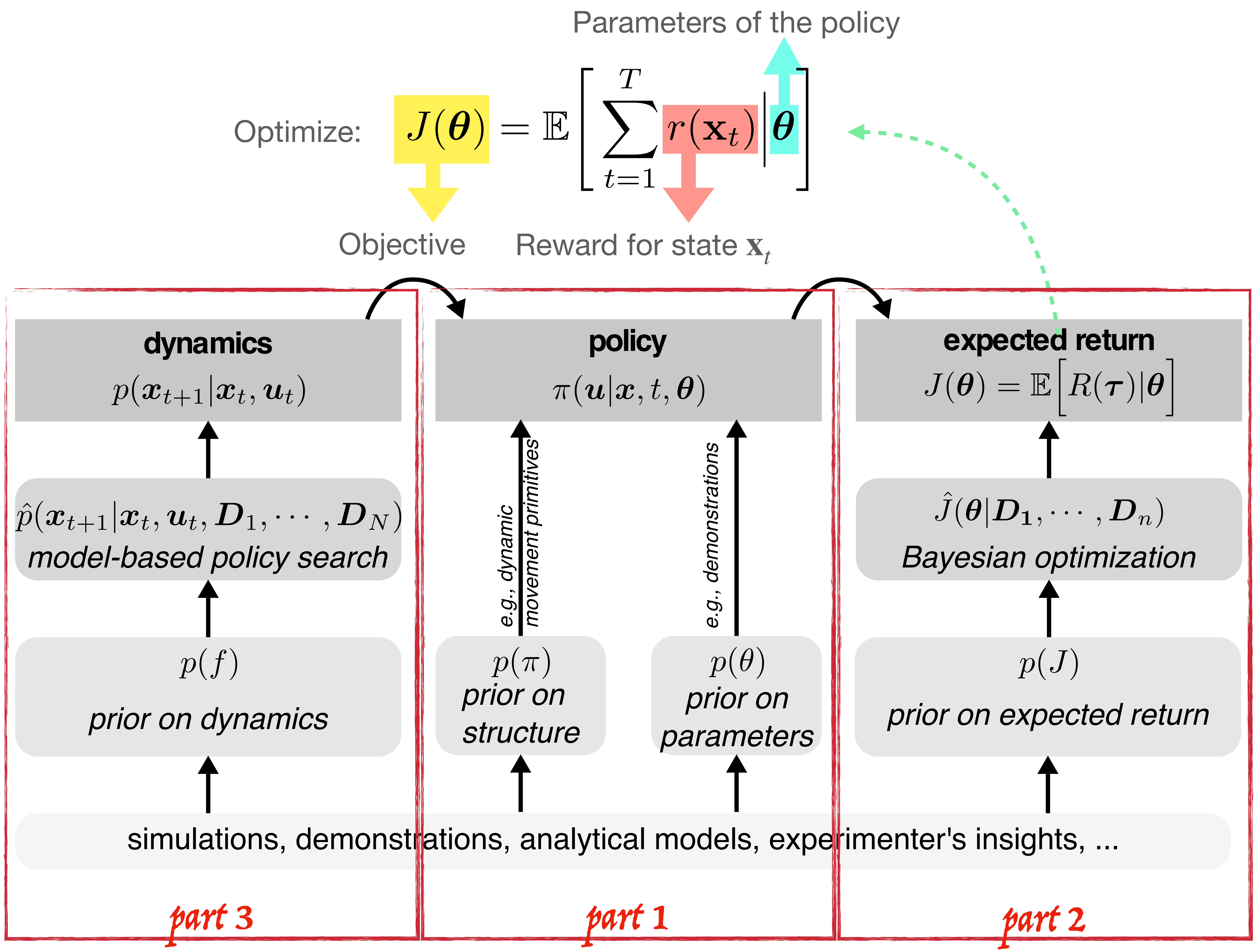
K. Chatzilygeroudis, V. Vassiliades, F. Stulp, S. Calinon, J.-B. Mouret. (2020). A survey on policy search algorithms for learning robot controllers in a handful of trials. IEEE Transactions on Robotics,


Parameters of the policy
Optimize: $J(\boldsymbol{\theta}) = \mathbb{E}\left[\sum_{t=1}^{T} r(\mathbf{x}_t) \middle| \boldsymbol{\theta}\right]$
Objective
Reward for state $\mathbf{x}_t$
system
dynamics
$p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t)$
policy
$\pi(\boldsymbol{u}|\boldsymbol{x}, t, \boldsymbol{\theta})$
expected return
$J(\boldsymbol{\theta}) = \mathbb{E}\Big[R(\boldsymbol{\tau})|\boldsymbol{\theta}\Big]$
models
$\hat{p}(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{D}_1, \cdots, \boldsymbol{D}_N)$
model-based policy search
$\hat{J}(\boldsymbol{\theta}|\boldsymbol{D}_1, \cdots, \boldsymbol{D}_n)$
Bayesian optimization

learn to predict dynamics from (x, u)
… then use the predictor like
a simulator to search for $\pi(u \,|\, x, \theta)$

learn to predict rewards from θ
… then search for the best
parameters directly

K. Chatzilygeroudis, V. Vassiliades, F. Stulp, S. Calinon, J.-B. Mouret. (2020). A survey on policy search algorithms for learning robot controllers in a handful of trials. IEEE Transactions on Robotics,


Parameters of the policy
Optimize: $J(\boldsymbol{\theta}) = \mathbb{E}\left[\sum_{t=1}^{T} r(\mathbf{x}_t) \middle| \boldsymbol{\theta}\right]$
Objective
Reward for state $\mathbf{x}_t$
system
models
priors
dynamics
$p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t)$
$\hat{p}(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{D}_1, \cdots, \boldsymbol{D}_N)$
model-based policy search
$p(f)$
prior on dynamics
policy
$\pi(\boldsymbol{u}|\boldsymbol{x}, t, \boldsymbol{\theta})$
e.g., dynamic movement primitives
e.g., demonstrations
$p(\pi)$
prior on structure
$p(\theta)$
prior on parameters
expected return
$J(\boldsymbol{\theta}) = \mathbb{E}\left[R(\boldsymbol{\tau})|\boldsymbol{\theta}\right]$
$\hat{J}(\boldsymbol{\theta}|\boldsymbol{D_1}, \cdots, \boldsymbol{D}_n)$
Bayesian optimization
$p(J)$
prior on expected return
simulations, demonstrations, analytical models, experimenter's insights, ...
part 3
part 1
part 2

K. Chatzilygeroudis, V. Vassiliades, F. Stulp, S. Calinon, J.-B. Mouret. (2020). A survey on policy search algorithms for learning robot controllers in a handful of trials. IEEE Transactions on Robotics,

Outline

1 Hand-designed policies

use well-designed policies with few parameters

use demonstrations

2 Bayesian optimization (a model of the reward)

learn to predict the reward value from θ

3 Model-based policy search

learn to predict the reward value from θ

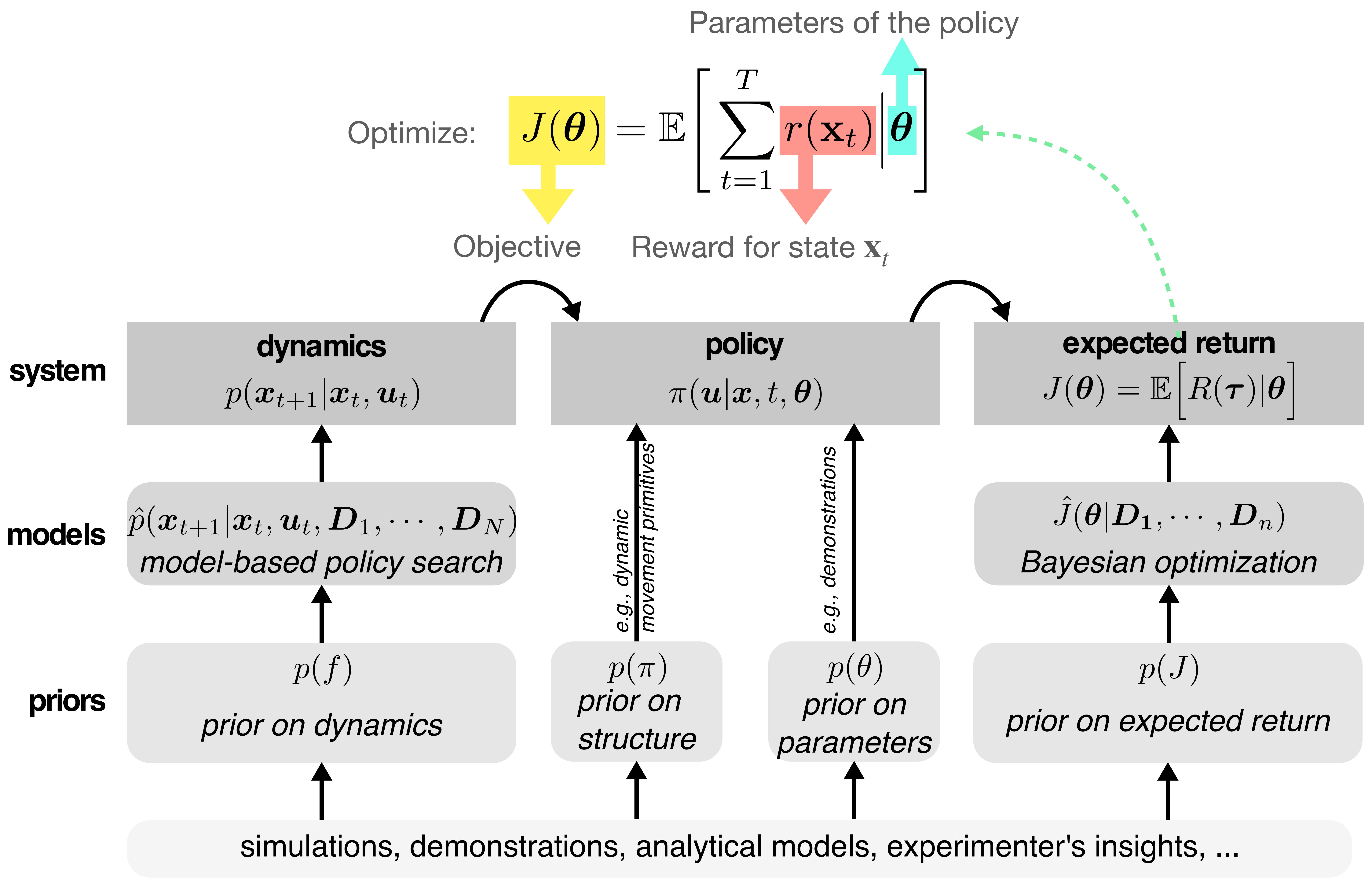
4 Keynote: Dongheui Lee (TU Munich)

Efficient Motor Skills Learning in Robotics

Micro-data policy search

Conclusion

JB Mouret & Konstantinos Chatzilygeroudis


Parameters of the policy
Optimize: $J(\boldsymbol{\theta}) = \mathbb{E}\left[\sum_{t=1}^{T} r(\mathbf{x}_t) \middle| \boldsymbol{\theta}\right]$
Objective
Reward for state $\mathbf{x}_t$
system
models
priors
dynamics
$p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t)$
policy
$\pi(\boldsymbol{u}|\boldsymbol{x}, t, \boldsymbol{\theta})$
expected return
$J(\boldsymbol{\theta}) = \mathbb{E}\left[R(\boldsymbol{\tau})|\boldsymbol{\theta}\right]$
$\hat{p}(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{D}_1, \cdots, \boldsymbol{D}_N)$
model-based policy search
e.g., dynamic movement primitives
e.g., demonstrations
$\hat{J}(\boldsymbol{\theta}|\boldsymbol{D}_1, \cdots, \boldsymbol{D}_n)$
Bayesian optimization
$p(f)$
prior on dynamics
$p(\pi)$
prior on structure
$p(\theta)$
prior on parameters
$p(J)$
prior on expected return
simulations, demonstrations, analytical models, experimenter's insights, ...

K. Chatzilygeroudis, V. Vassiliades, F. Stulp, S. Calinon, J.-B. Mouret. (2020). A survey on policy search algorithms for learning robot controllers in a handful of trials. IEEE Transactions on Robotics,

… what to use?

- **Low-DOF robots (<10), high-D policy (NN)**
  ➔ small state-action space
  ➔ model-based policy search

- **High-DOF robot, low-D policy (hand-designed)**
  ➔ Bayesian optimization

- **Complex robots, complex policies**
  ➔ Model-based policy search + prior from model (simulation, meta-learning)
  ➔ BO + MAP-Elites

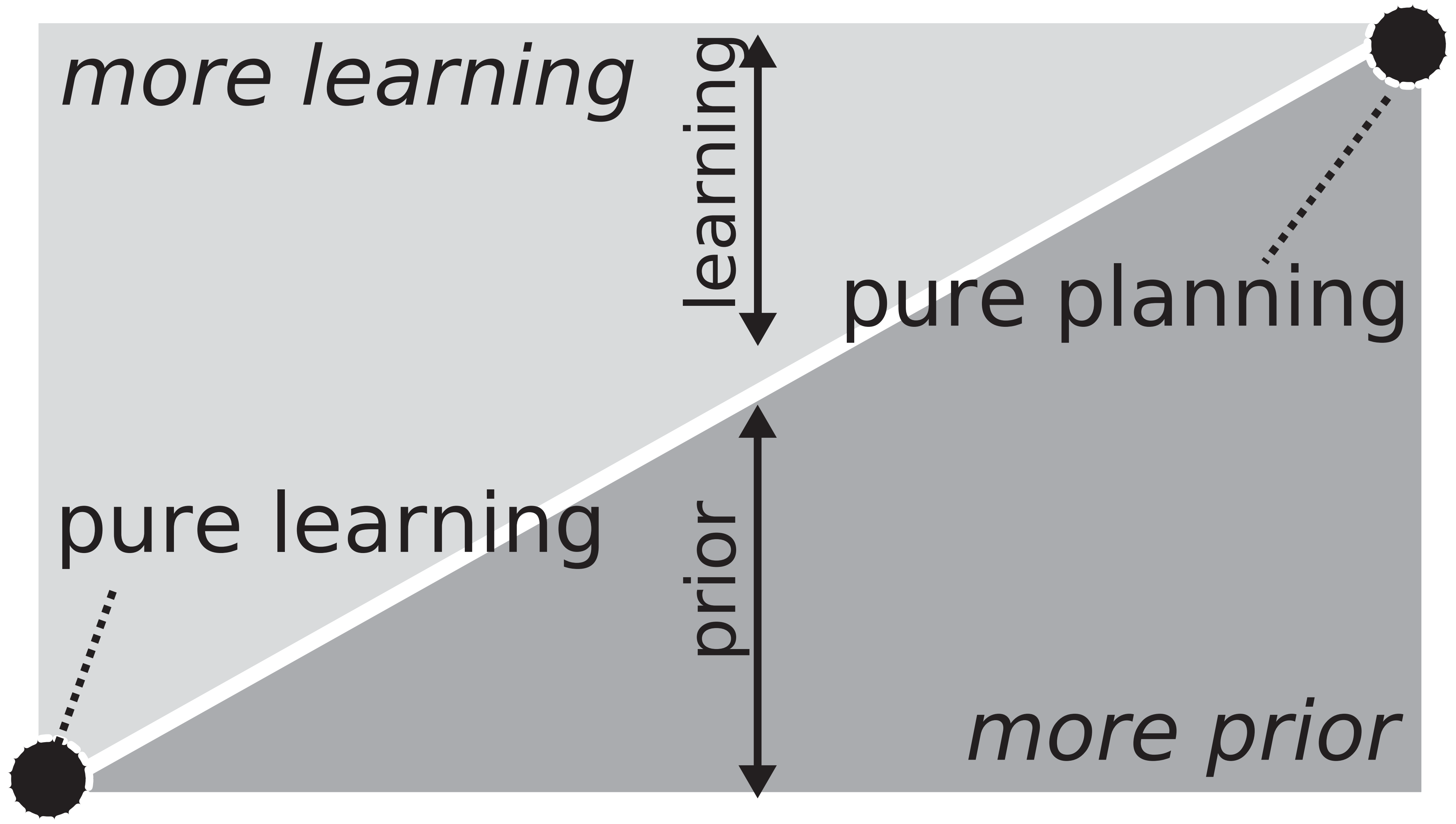
- **High-D raw inputs (e.g., images)**
  ➔ Sim2real? Use a VAE for unsupervised low-dimensional input?

- **Computation time:**
  ➔ very high with model-based approach
  ➔ low with Bayesian optimisation (+ MAP-Elites)

K. Chatzilygeroudis, V. Vassiliades, F. Stulp, S. Calinon, J.-B. Mouret. (2020). A survey on policy search algorithms for learning robot controllers in a handful of trials. IEEE Transactions on Robotics,

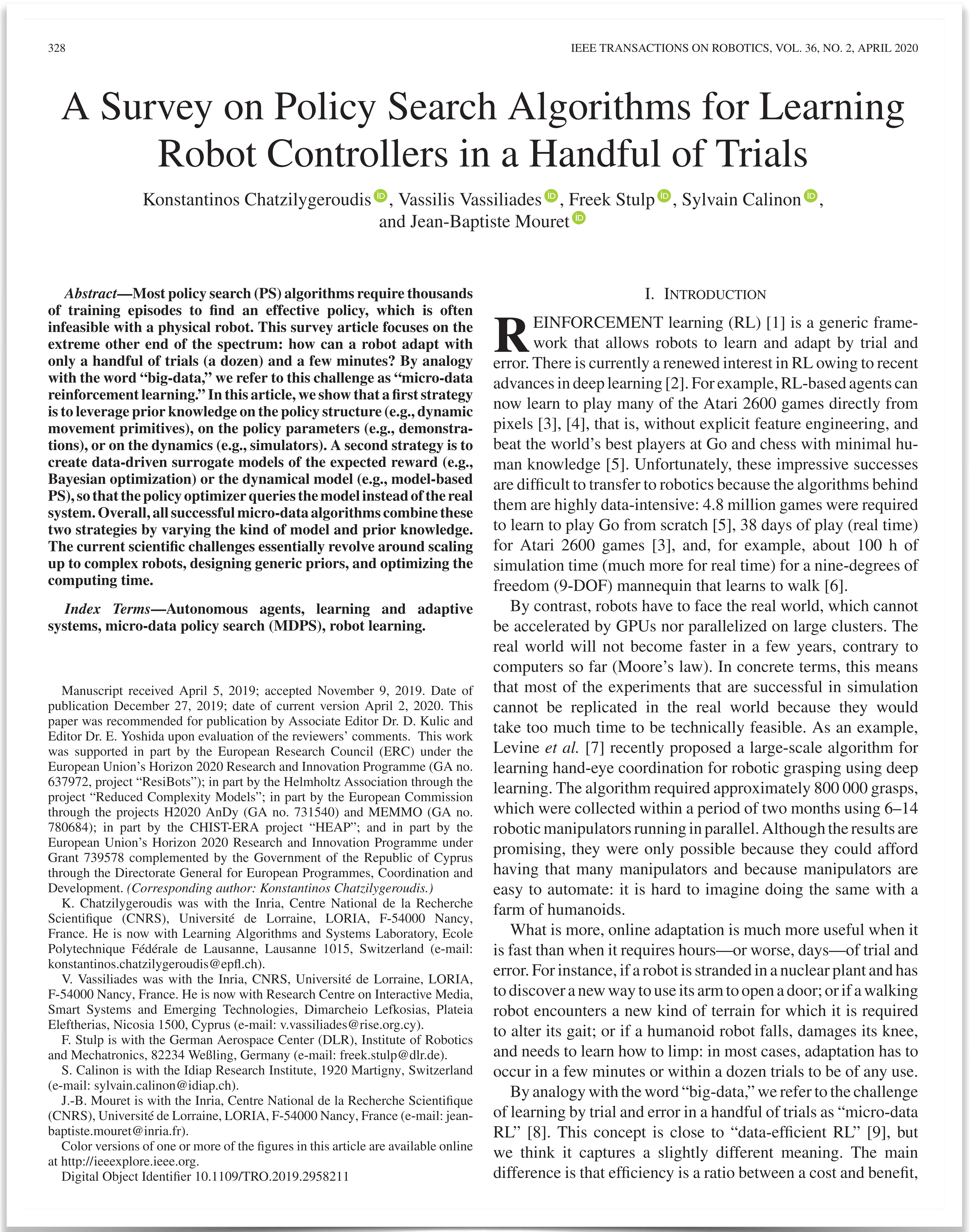
… everything is about priors

- **There are always priors:**
  - neural network topology
  - using a neural networks vs something else
  - hyper-parameters
  - simulator of a real robot
  - design of a reward function, …
- **The best priors are generic priors**
  - e.g. convolutional neural networks are a great prior!
  - priors from simulation
  - meta-learning
- **The best priors can be overcome**
- **We should think more about our prior and work on improving them**
  (instead of hiding them)



K. Chatzilygeroudis, V. Vassiliades, F. Stulp, S. Calinon, J.-B. Mouret. (2020). A survey on policy search algorithms for learning robot controllers in a handful of trials. IEEE Transactions on Robotics,

Further readings



A Survey on Policy Search Algorithms for Learning Robot Controllers in a Handful of Trials

Konstantinos Chatzilygeroudis, Vassilis Vassiliades, Freek Stulp, Sylvain Calinon, and Jean-Baptiste Mouret

Abstract—**Most policy search (PS) algorithms require thousands of training episodes to find an effective policy, which is often infeasible with a physical robot. This survey article focuses on the extreme other end of the spectrum: how can a robot adapt with only a handful of trials (a dozen) and a few minutes? By analogy with the word "big-data," we refer to this challenge as "micro-data reinforcement learning." In this article, we show that a first strategy is to leverage prior knowledge on the policy structure (e.g., dynamic movement primitives), on the policy parameters (e.g., demonstrations), or on the dynamics (e.g., simulators). A second strategy is to create data-driven surrogate models of the expected reward (e.g., Bayesian optimization) or the dynamical model (e.g., model-based PS), so that the policy optimizer queries the model instead of the real system. Overall, all successful micro-data algorithms combine these two strategies by varying the kind of model and prior knowledge. The current scientific challenges essentially revolve around scaling up to complex robots, designing generic priors, and optimizing the computing time.**

Index Terms—**Autonomous agents, learning and adaptive systems, micro-data policy search (MDPS), robot learning.**

I. Introduction

REINFORCEMENT learning (RL) [1] is a generic framework that allows robots to learn and adapt by trial and error. There is currently a renewed interest in RL owing to recent advances in deep learning [2]. For example, RL-based agents can now learn to play many of the Atari 2600 games directly from pixels [3], [4], that is, without explicit feature engineering, and beat the world's best players at Go and chess with minimal human knowledge [5]. Unfortunately, these impressive successes are difficult to transfer to robotics because the algorithms behind them are highly data-intensive: 4.8 million games were required to learn to play Go from scratch [5], 38 days of play (real time) for Atari 2600 games [3], and, for example, about 100 h of simulation time (much more for real time) for a nine-degrees of freedom (9-DOF) mannequin that learns to walk [6].

By contrast, robots have to face the real world, which cannot be accelerated by GPUs nor parallelized on large clusters. The real world will not become faster in a few years, contrary to computers so far (Moore's law). In concrete terms, this means that most of the experiments that are successful in simulation cannot be replicated in the real world because they would take too much time to be technically feasible. As an example, Levine *et al.* [7] recently proposed a large-scale algorithm for learning hand-eye coordination for robotic grasping using deep learning. The algorithm required approximately 800 000 grasps, which were collected within a period of two months using 6–14 robotic manipulators running in parallel. Although the results are promising, they were only possible because they could afford having that many manipulators and because manipulators are easy to automate: it is hard to imagine doing the same with a farm of humanoids.

What is more, online adaptation is much more useful when it is fast than when it requires hours—or worse, days—of trial and error. For instance, if a robot is stranded in a nuclear plant and has to discover a new way to use its arm to open a door; or if a walking robot encounters a new kind of terrain for which it is required to alter its gait; or if a humanoid robot falls, damages its knee, and needs to learn how to limp: in most cases, adaptation has to occur in a few minutes or within a dozen trials to be of any use.

By analogy with the word "big-data," we refer to the challenge of learning by trial and error in a handful of trials as "micro-data RL" [8]. This concept is close to "data-efficient RL" [9], but we think it captures a slightly different meaning. The main difference is that efficiency is a ratio between a cost and benefit,

Manuscript received April 5, 2019; accepted November 9, 2019. Date of publication December 27, 2019; date of current version April 2, 2020. This paper was recommended for publication by Associate Editor Dr. D. Kulic and Editor Dr. E. Yoshida upon evaluation of the reviewers' comments. This work was supported in part by the European Research Council (ERC) under the European Union's Horizon 2020 Research and Innovation Programme (GA no. 637972, project "ResiBots"); in part by the Helmholtz Association through the project "Reduced Complexity Models"; in part by the European Commission through the projects H2020 AnDy (GA no. 731540) and MEMMO (GA no. 780684); in part by the CHIST-ERA project "HEAP"; and in part by the European Union's Horizon 2020 Research and Innovation Programme under Grant 739578 complemented by the Government of the Republic of Cyprus through the Directorate General for European Programmes, Coordination and Development. *(Corresponding author: Konstantinos Chatzilygeroudis.)*

K. Chatzilygeroudis was with the Inria, Centre National de la Recherche Scientifique (CNRS), Université de Lorraine, LORIA, F-54000 Nancy, France. He is now with Learning Algorithms and Systems Laboratory, Ecole Polytechnique Fédérale de Lausanne, Lausanne 1015, Switzerland (e-mail: konstantinos.chatzilygeroudis@epfl.ch).

V. Vassiliades was with the Inria, CNRS, Université de Lorraine, LORIA, F-54000 Nancy, France. He is now with Research Centre on Interactive Media, Smart Systems and Emerging Technologies, Dimarcheio Lefkosias, Plateia Eleftherias, Nicosia 1500, Cyprus (e-mail: v.vassiliades@rise.org.cy).

F. Stulp is with the German Aerospace Center (DLR), Institute of Robotics and Mechatronics, 82234 Weßling, Germany (e-mail: freek.stulp@dlr.de).

S. Calinon is with the Idiap Research Institute, 1920 Martigny, Switzerland (e-mail: sylvain.calinon@idiap.ch).

J.-B. Mouret is with the Inria, Centre National de la Recherche Scientifique (CNRS), Université de Lorraine, LORIA, F-54000 Nancy, France (e-mail: jean-baptiste.mouret@inria.fr).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2019.2958211

K. Chatzilygeroudis, V. Vassiliades, F. Stulp, S. Calinon, J.-B. Mouret. (2020). A survey on policy search algorithms for learning robot controllers in a handful of trials. IEEE Transactions on Robotics,