Deep Reinforcement Learning

Lecture 4: Deep Model-Based RL

Juan Diego Cardenas Cartagena
j.d.cardenas.cartagena@rug.nl

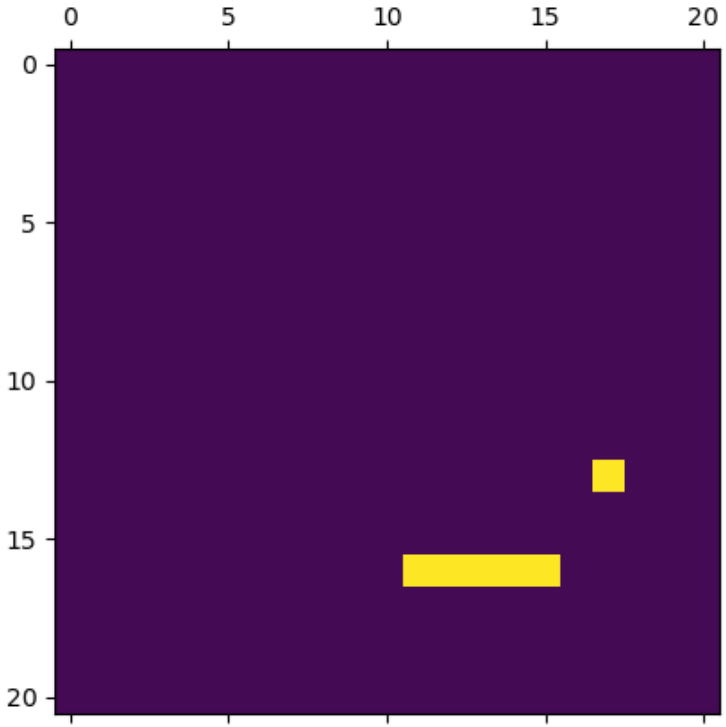
university of groningen

Announcements

Homework

Submit your results from the **catch environment**

Deadline: Tuesday, 16 May.



Catch environment

Announcements

University Elections

The University elections will take place from 8 to 12 May 2023 for the University Council, the Faculty Councils and the Service Department Councils.

Today's Agenda

- Learning objectives
- Recap and Motivation
- Introduction
- Planning
- Architectures
- Deep Model-based RL
- State of the Art
- Closing Remarks

Learning objectives

Target Questions

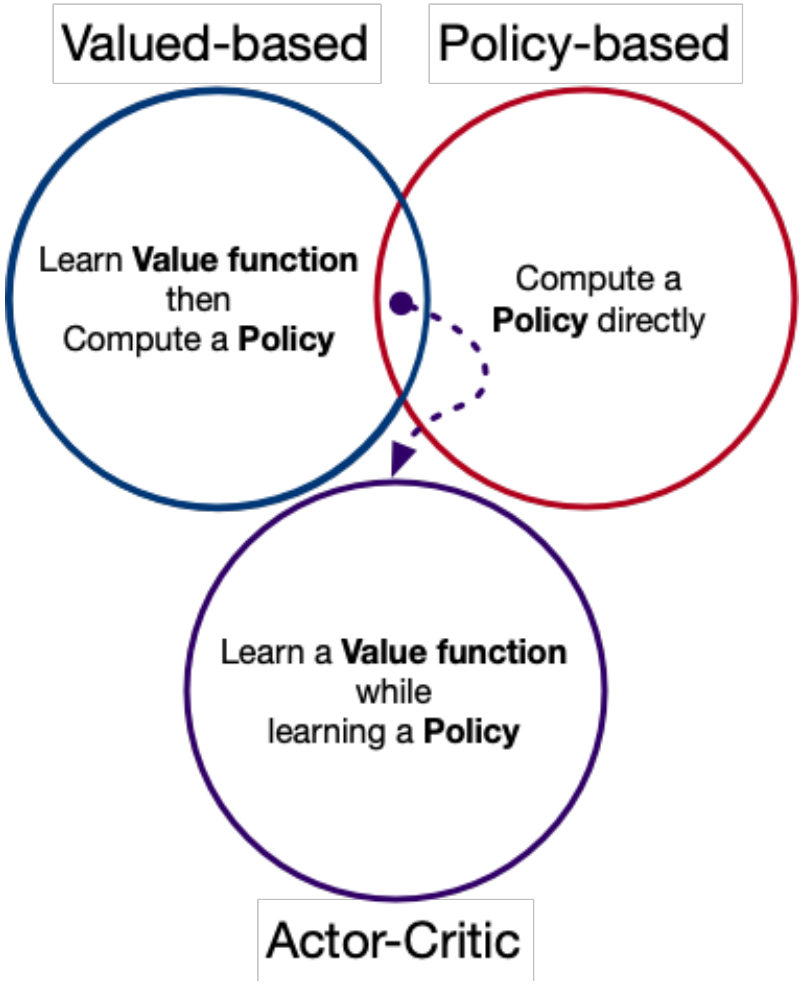
- How do RL agents work with **models**?
- What is the difference between **Model-free** and **Model-based** RL?
- What are the **approaches** for Model-based RL?
- How is **Deep Learning** related to Model-based RL?

Recommended reading

- A. Plaat, **Chapter 5: Model-based Reinforcement Learning**, in Deep Reinforcement Learning, a textbook. 2022, pp. 119-145. doi: 10.1007/978-981-19-0638-1.
- R. S. Sutton and A. G. Barto, **Chapter 8: Planning and Learning with Tabular Methods**, in Reinforcement learning: an introduction, Second edition. Cambridge, Massachusetts: The MIT Press, 2018, pp. 160-191.

Recap

Model-free RL



Model-free RL architectures

Dynamic Programming

Problem Setting:

- Consider a system with the form

$$s_{t+1} = f(s_t, a_t, \omega_t),\ t = 0, 1, ..., T-1,$$

  where t is a discrete-time index, s_t, a_t and ω_t are the state, action, and random noise at time t, respectively. Moreover, f is a transition function that describes how the states update in time.

- The control system is subject to the cost (or reward) $g_t(s_t, a_t, \omega_t)$, which accumulates over time. The total cost for an T-horizon in expectation is given by

$$J_0(s_0) = \mathbb{E}_\omega \left[g_T(s_T) + \sum_{t=0}^{T-1} g_t(s_t, a_t, \omega_t) \right] \quad (1)$$

where $g_T(s_T)$ is a terminal cost.

Dynamic Programming

Principle of Optimality [1]

Let $\pi^\star = \pi_0^\star, \pi_1^\star, ..., \pi_{T-1}^\star$ be an optimal policy which minimizes the total cost in (1).

Consider the sub-problem whereby the system is at s_i at time i and aims to minimize the cost-to-go from time i to T

$$J_i(s_i) = \mathbb{E}_\omega \left[g_T(s_T) + \sum_{t=i}^{T-1} g_t(s_t, a_t, \omega_t) \right],$$

Then the truncated policy $\pi_i^\star, \pi_{i+1}^\star, ..., \pi_{T-1}^\star$ is optimal for this subproblem

Dynamic Programming

Proposition: The Dynamic Programming Algorithm [1]

For every initial state s_0, the optimal cost $J^\star(s_0)$ of (1) is equal to $J_0(s_0)$, given by the last step of the following algorithm, which proceeds backwards in time from period $T-1$ to 0:

$$J_T(s_T) = g_T(s_T),$$

$$J_t(s_t) = \min_{a_t \in \mathcal{A}(s_t)} \mathbb{E}_\omega \left[g_t(s_t, a_t, \omega_t) + J_{t+1}(f(s_t, a_t, \omega_t)) \right] \quad (2)$$

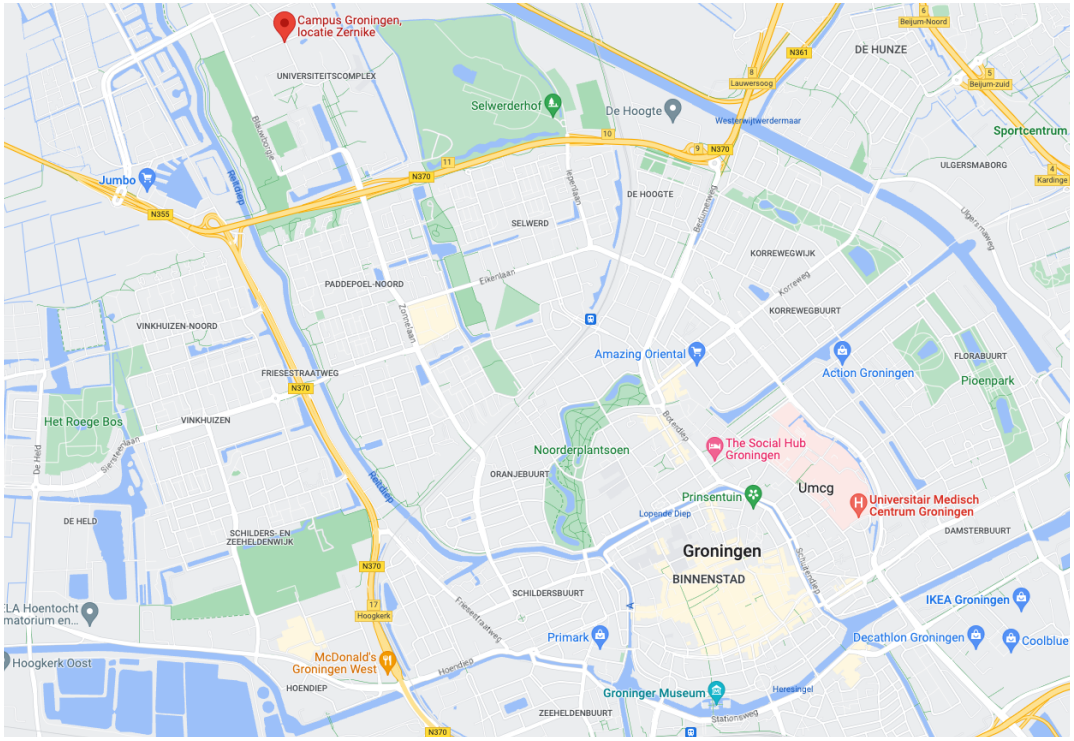
$t = 0, 1, ..., T-1$. Furthermore, if $a_t^\star = \pi_t^\star(s_t)$ minimizes the right side of (2) for each s_t and t, the policy $\pi^\star = \pi_0^\star, \pi_1^\star, ..., \pi_{T-1}^\star$ is optimal.

Sketch of the proof: show via induction that $J_t^\star(s_t) = J_t(s_t)$ by using the principle of optimality.

Motivation

Planning

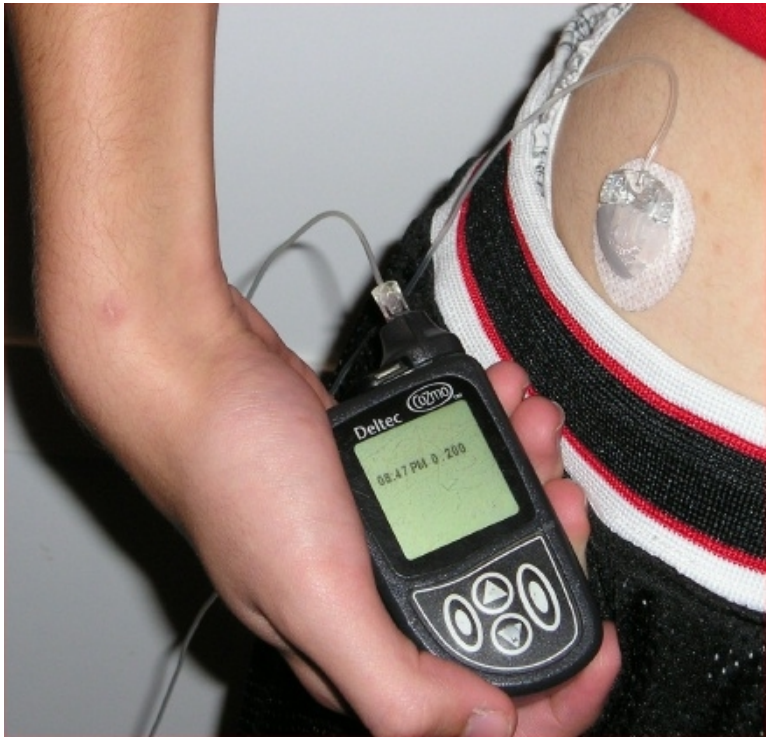
Arriving to a new city case, based on [2] Do you remember the first time when you came to the Zernike campus? How did you find the **most comfortable** or **shortest** path from your home to the campus?



Map of Groningen, taken from maps.google.com

The Cost of a Sample

Context Let us suppose that Lupita is a type I **Diabetes** patient. She uses an **insulin pump** to regulate her metabolism.



Insulin pump, taken from commons.wikimedia.org

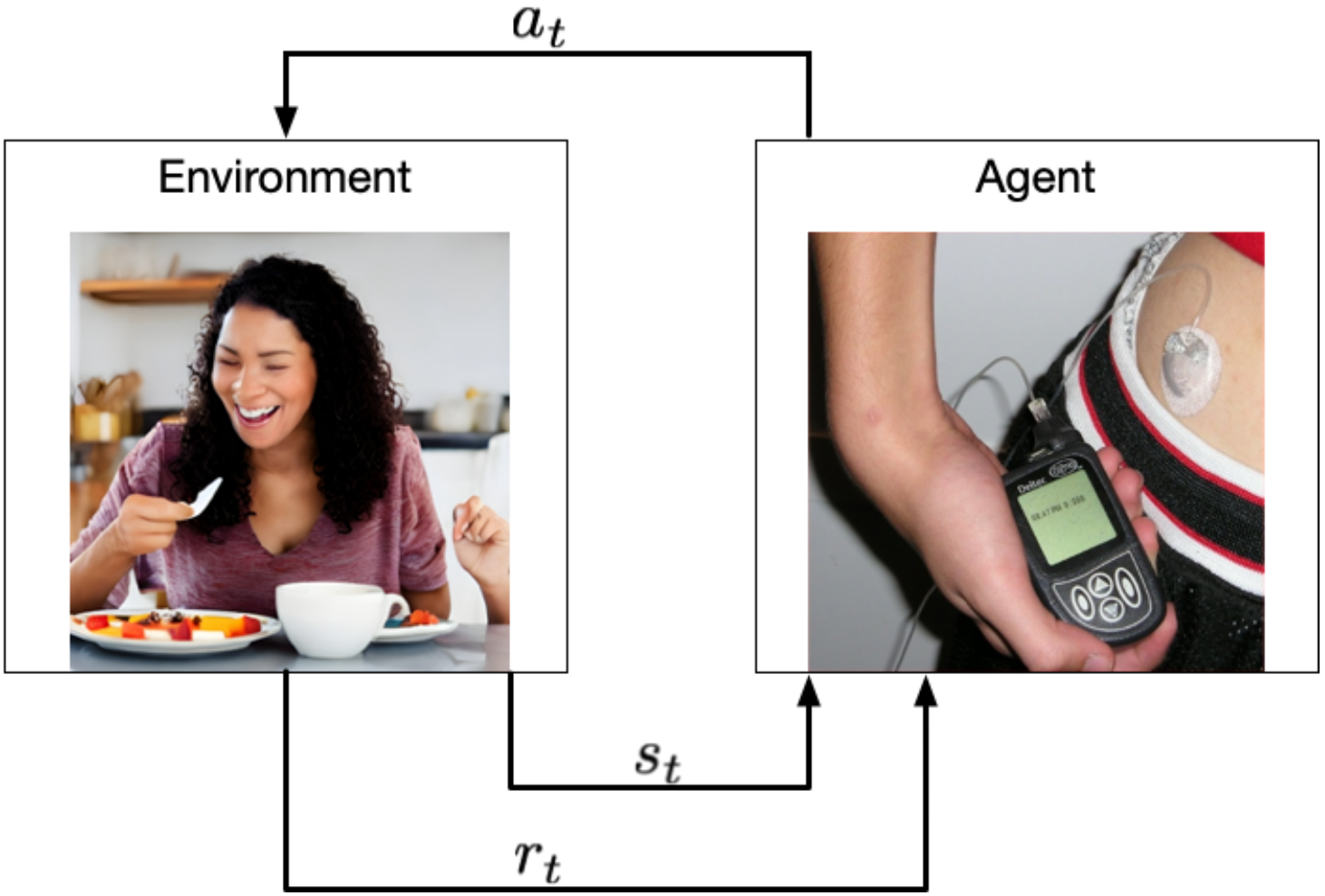
The Cost of a Sample

Artificial Pancreas Case, based on [3, 4]

As part of the R&D and control design team of Healthy Pancreas B.V., you should provide a **feasibility report of an RL** approach for the control system in a new insulin pump.

Disclaimer: Implementing RL algorithms in the artificial pancreas is an open research question.

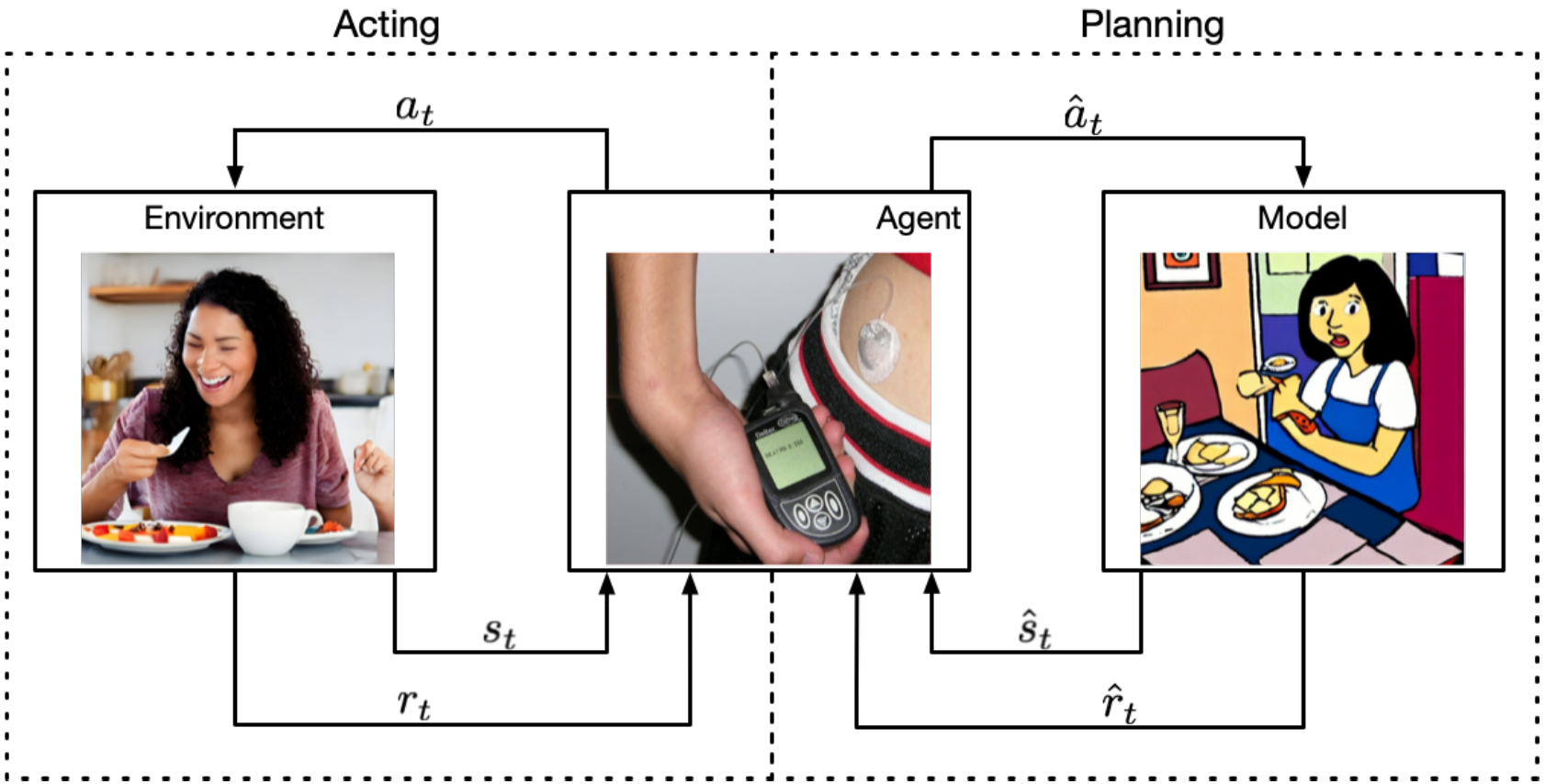
The Cost of a Sample



Agent-Environment interaction for the Artificial Pancreas example

Introduction

Introduction to Model-based RL



(Simplified) Agent-Environment-Model interaction for the Artificial Pancreas example

Introduction to Model-based RL

- While the Model-free approach aims to learn either a value function, a policy, or both, directly from the environment, this agent-environment interaction to converge in (sub-)optimal policy becomes a sample-intensive task.

- The Model-based approach uses a model of the environment to generate artificial state transitions with the respective reward and utilize them to train the agent. Those models may enable the possibility of transfer learning and acquiring knowledge about non-visited states.

Introduction to Model-based RL

- Let $\mathcal{T} = (\mathcal{P}, \Re)$ be the true model of the environment, where $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ and $\Re : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ are transition and reward functions.

- A model-based RL is composed by an agent with policy $\pi : \mathcal{A} \times \mathcal{S} \to [0, 1]$ and approximate model $\hat{\mathcal{T}}_\phi = (\hat{\mathcal{P}}_\phi, \hat{\Re}_\phi)$ where $\hat{\mathcal{P}}_\phi : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ and $\hat{\Re}_\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ approximate the transition and reward of the environment.

- The model can come from first principles to machine learning techniques. This lecture focuses on deep neural networks.

Introduction to Model-based RL

Advantages w.r.t. Model-free RL

- The RL agent becomes more sample efficient as the model can generate more samples per interaction with the environment.
- The model generates trajectories for future states so that the agent can perform planning and execute actions based on long-horizon.
- We can use expert knowledge to build the models, e.g., physics, biology, or chemistry.
- The model-based approach can take advantage of control theory to perform convergence and stability analysis, e.g. Lyapunov stability.

Introduction to Model-based RL

Disadvantages w.r.t. Model-free RL

- The policy is highly dependant on the quality of the model.
- Building (or learning) a model is not straightforward for complex systems.
- The model introduces bias in the policy.
- Computing complex models might consume more resources than interacting directly with the environment.
- Planning far in the future with the model can result in misleading policies due to error accumulation between the environment and model.
- The agent may become unstable due to compute policies under irrelevant or impossible states provided by the model.

Introduction to Model-based RL

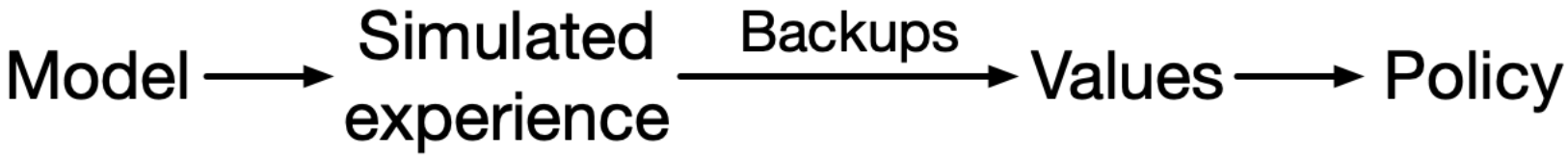
Deadly Triad and Curse of Dimensionality [5]

- As other RL architectures that involve function approximators like deep NN, model-based RL may suffer from the deadly triad. That is, potential issues of divergence and instability.
- The deadly triad appears when the algorithm combines these three elements: function approximators, bootstrapping, and off-policy training.
- Also, model-based RL suffers from the curse of dimensionality in high-dimensional tasks, i.e. data becomes sparse and variance increases.

Planning

Workflow

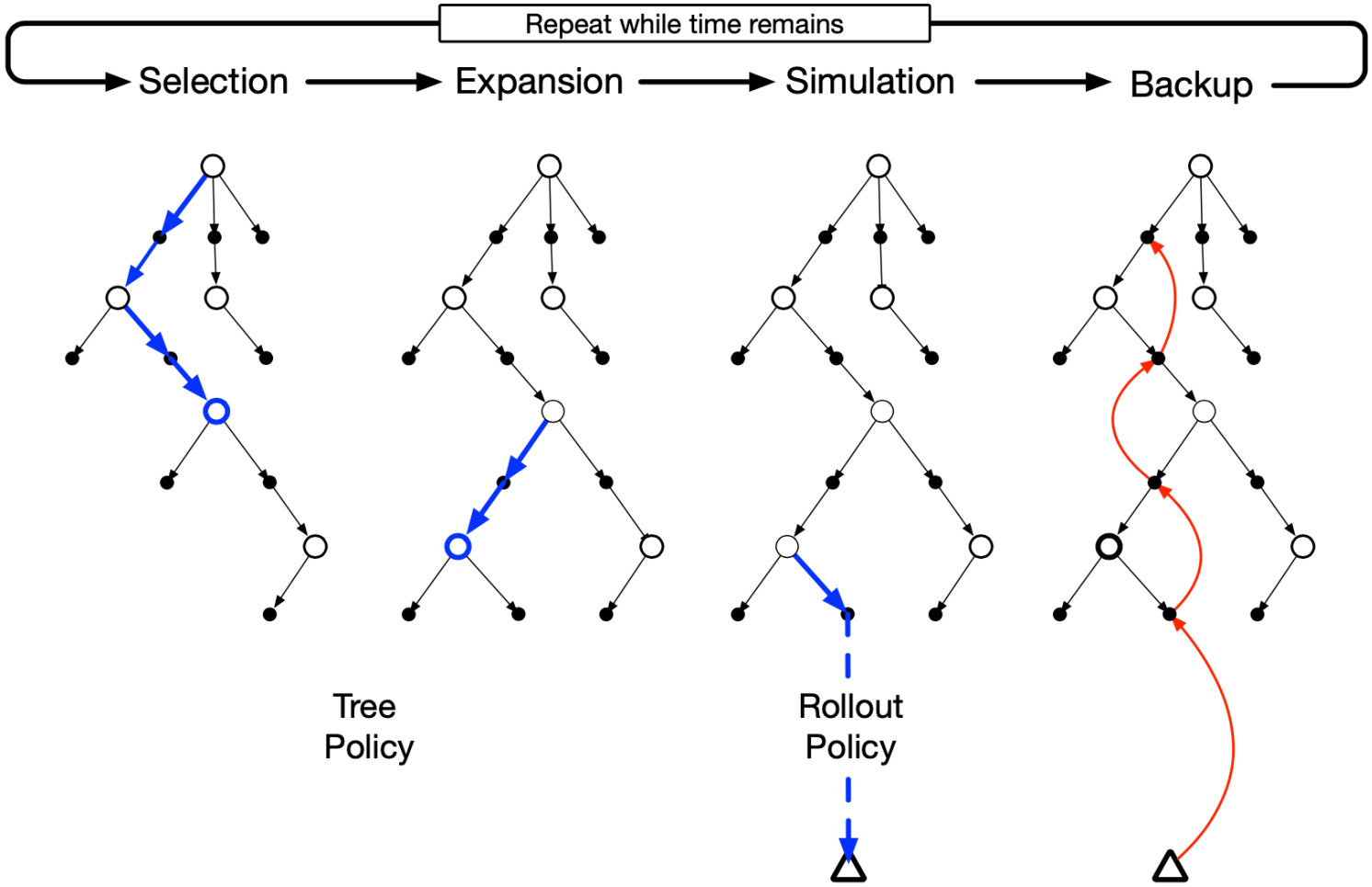
Use the model $\hat{\mathcal{T}}_\phi$ to select a sequence of actions, generate a simulated experience, and back up reward values to train the policy.



Planning workflow, taken from [5]

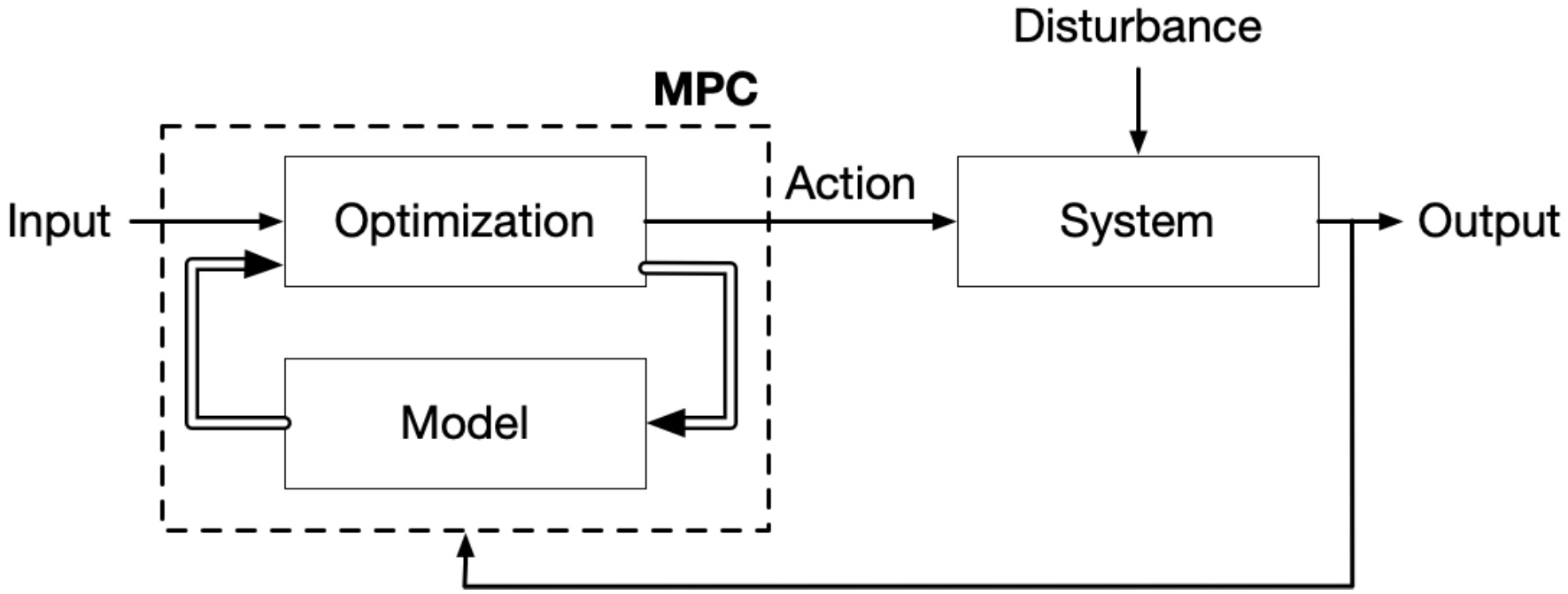
Quick question: Do you remember any algorithm that fits the planning steps?

Monte Carlo Tree Search (MCTS)



Steps in the MCTS, Taken from [5]

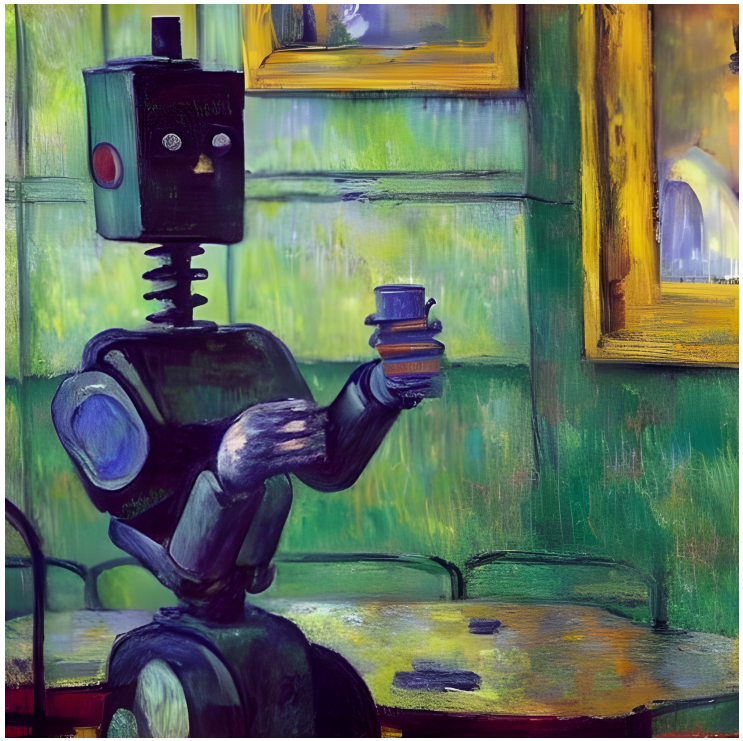
Model Predictive Control (MPC)



Simplified diagram of an MPC-based control loop, adapted from [6]

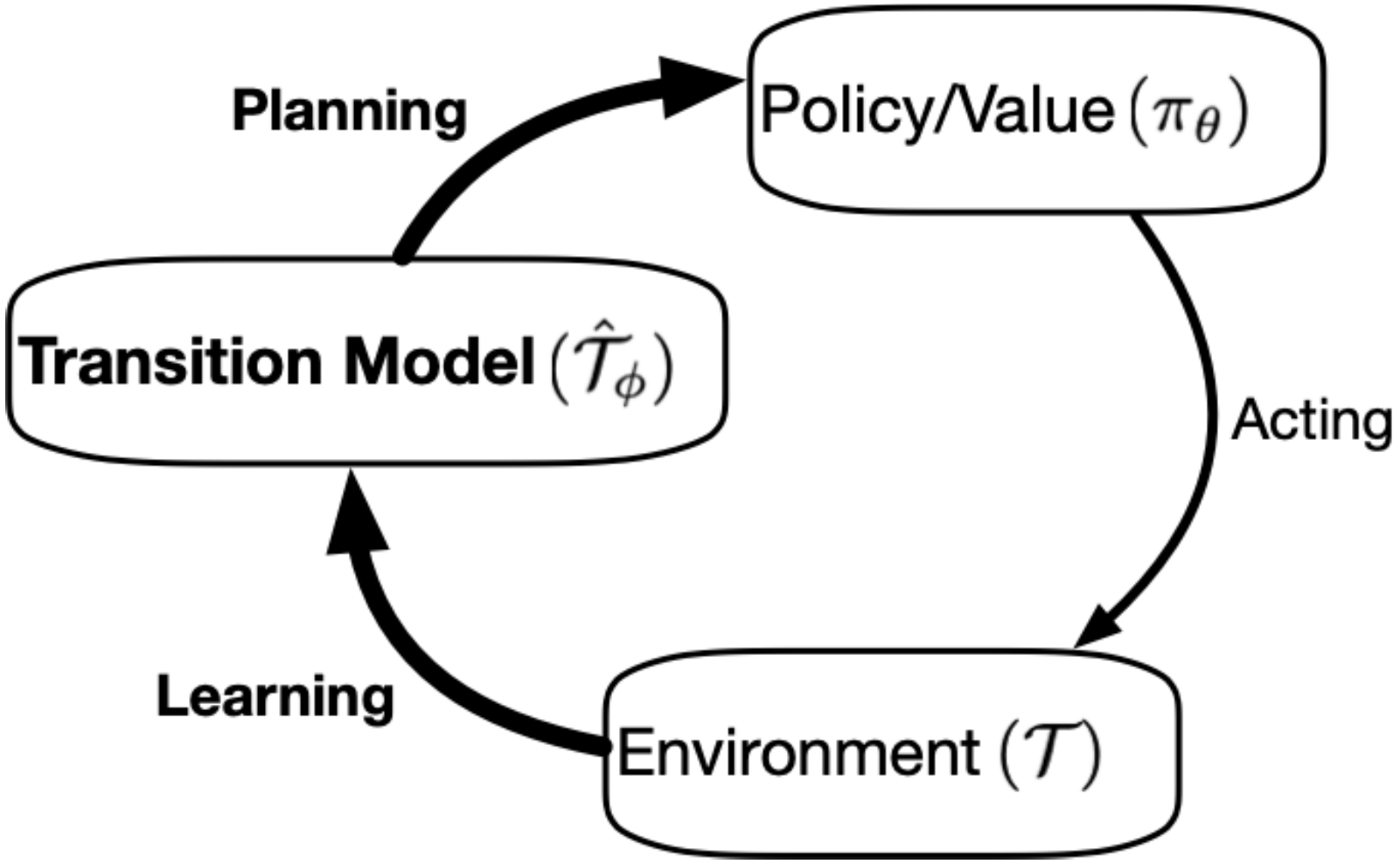
Break

See you at:



Architectures for Model-based RL

Explicit Planning on Given/Learned Transitions



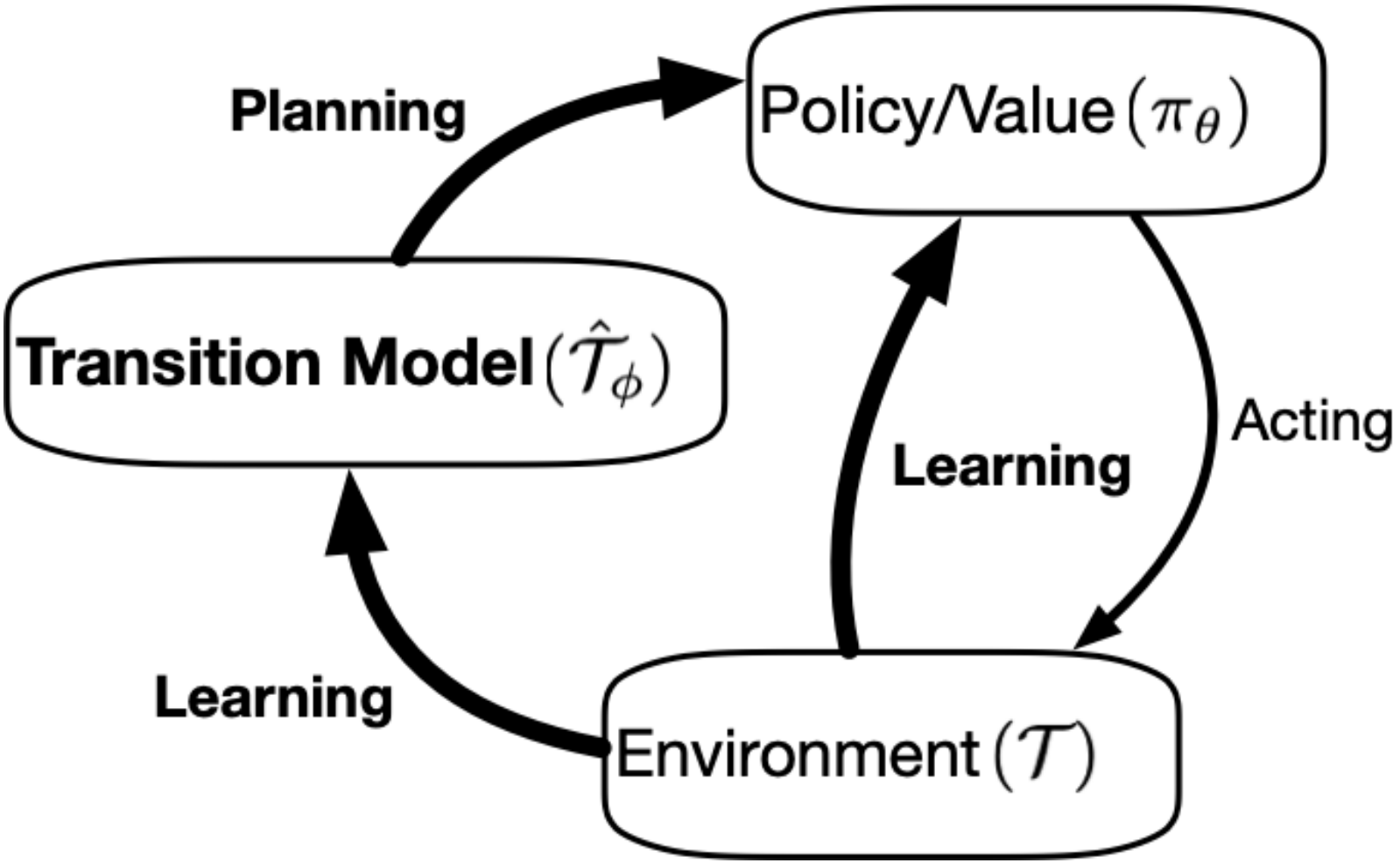
Given/learned transitions architecture, adapted from [7]

Explicit Planning on Given/Learned Transitions

Algorithm Explicit Planning/Learned Transitions, from [7]

repeat
  Sample env $\mathcal{T}$ to generate data $D = (s, a, r', s')$
  Use D to learn $\hat{\mathcal{T}}_\phi$
  Use $\hat{\mathcal{T}}_\phi$ to update π by planning
until π converges

Hybrid Model-free/Model-based RL - Dyna



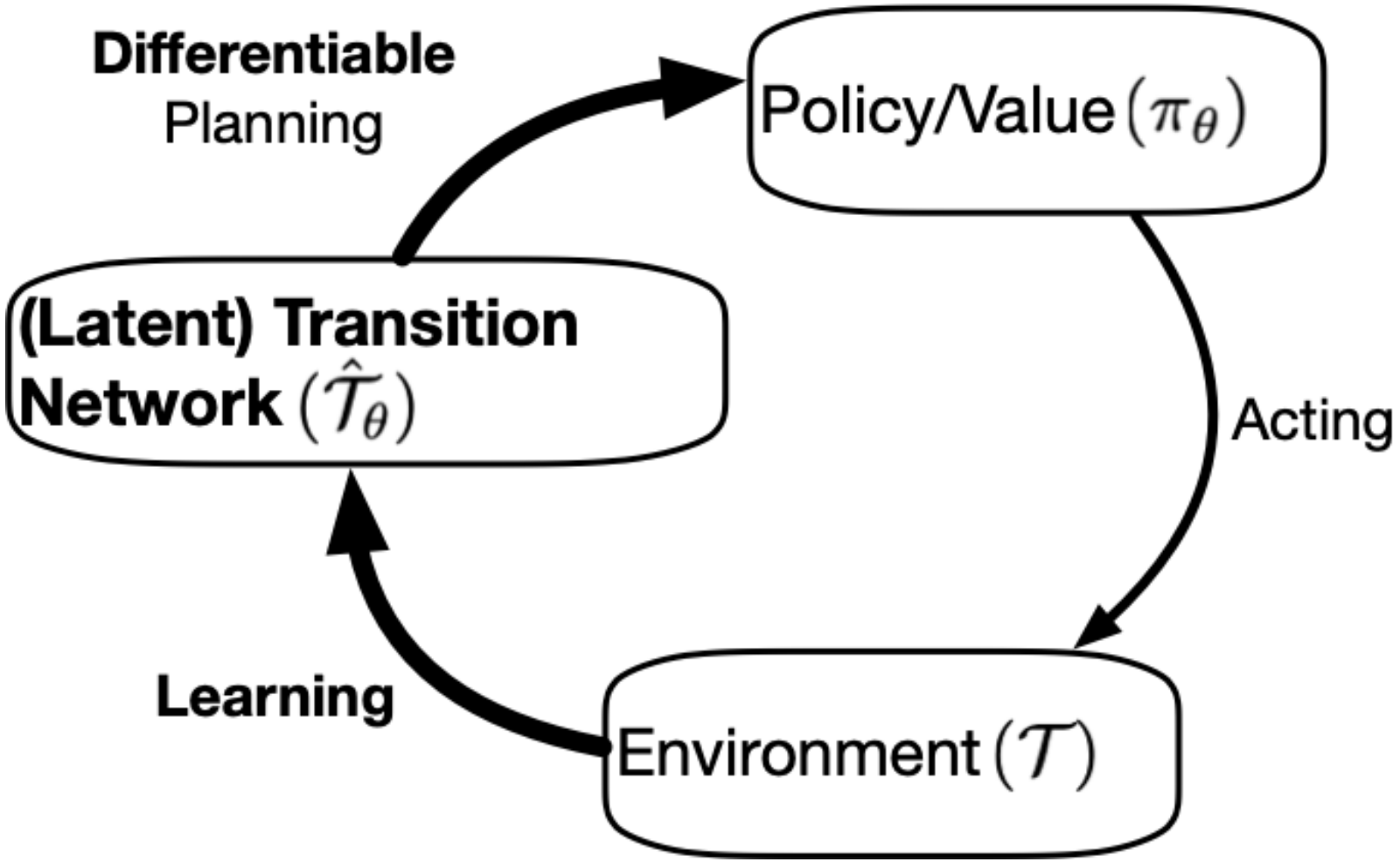
Hybrid model-free/model-based architecture, adapted from [7]

Hybrid Model-free/Model-based RL - Dyna

Algorithm Dyna-Q algorithm, from [5]

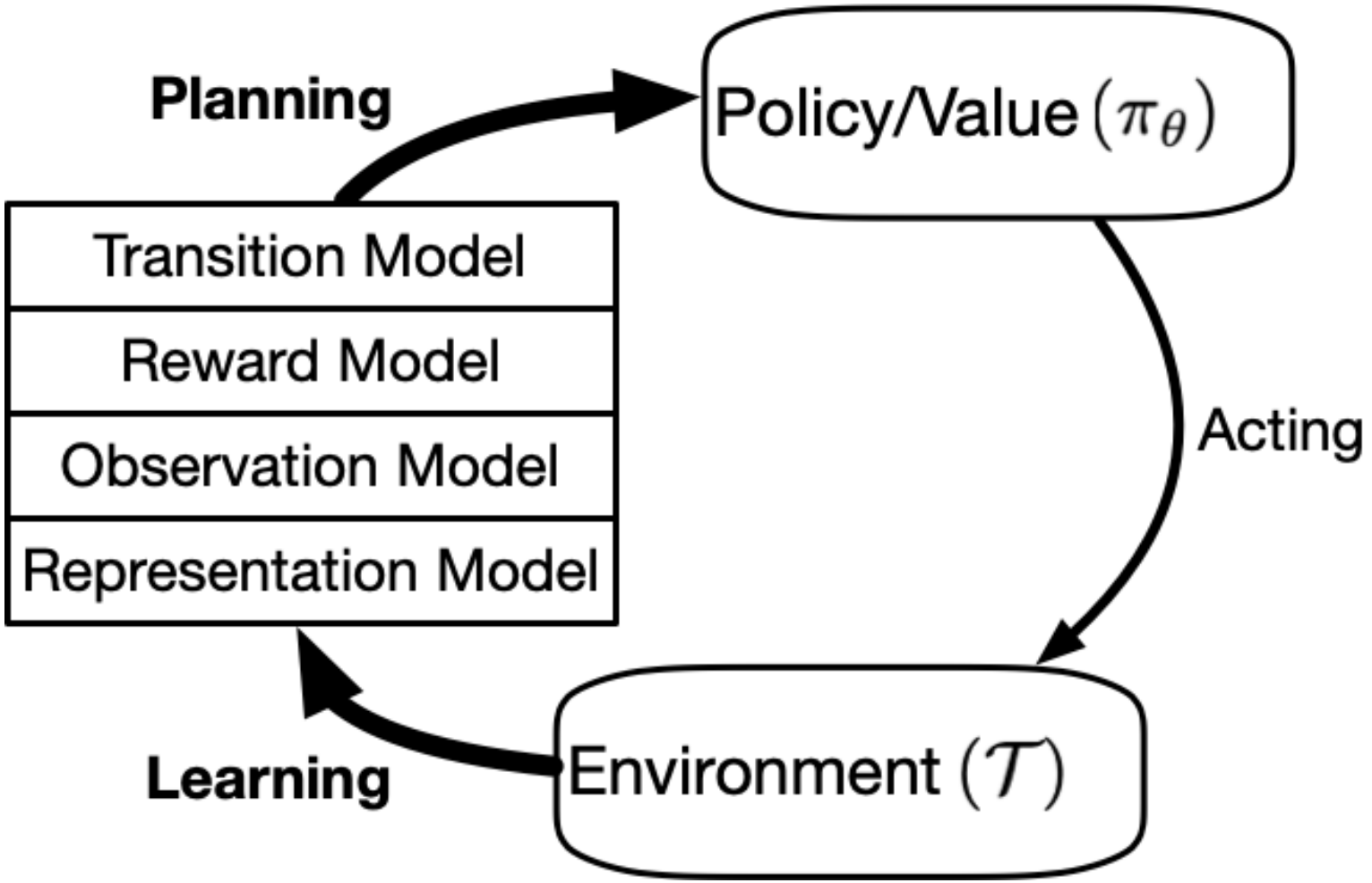
Require: Initialize $Q(s, a)$, $\hat{\mathcal{T}}_\phi(s, a)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}$
  while true **do**
    $S \leftarrow$ current (nonterminal) state
    $A \leftarrow \epsilon$-greedy(s, Q)
    Take action A in env $\mathcal{T}$; observe resultant R' and S'
    Update Q-values with experience from the environment
    Learn $\hat{\mathcal{T}}_\phi$ using new information (S, A, R', S')
    repeat
      $S \leftarrow$ random previously observed state
      $A \leftarrow$ random action previously taken in S
      $R', S' \leftarrow \hat{\mathcal{T}}_\phi(S, A)$
      Update Q-values with simulated experience
    until n times
  end while

End-to-end Learning of Planning and Transitions



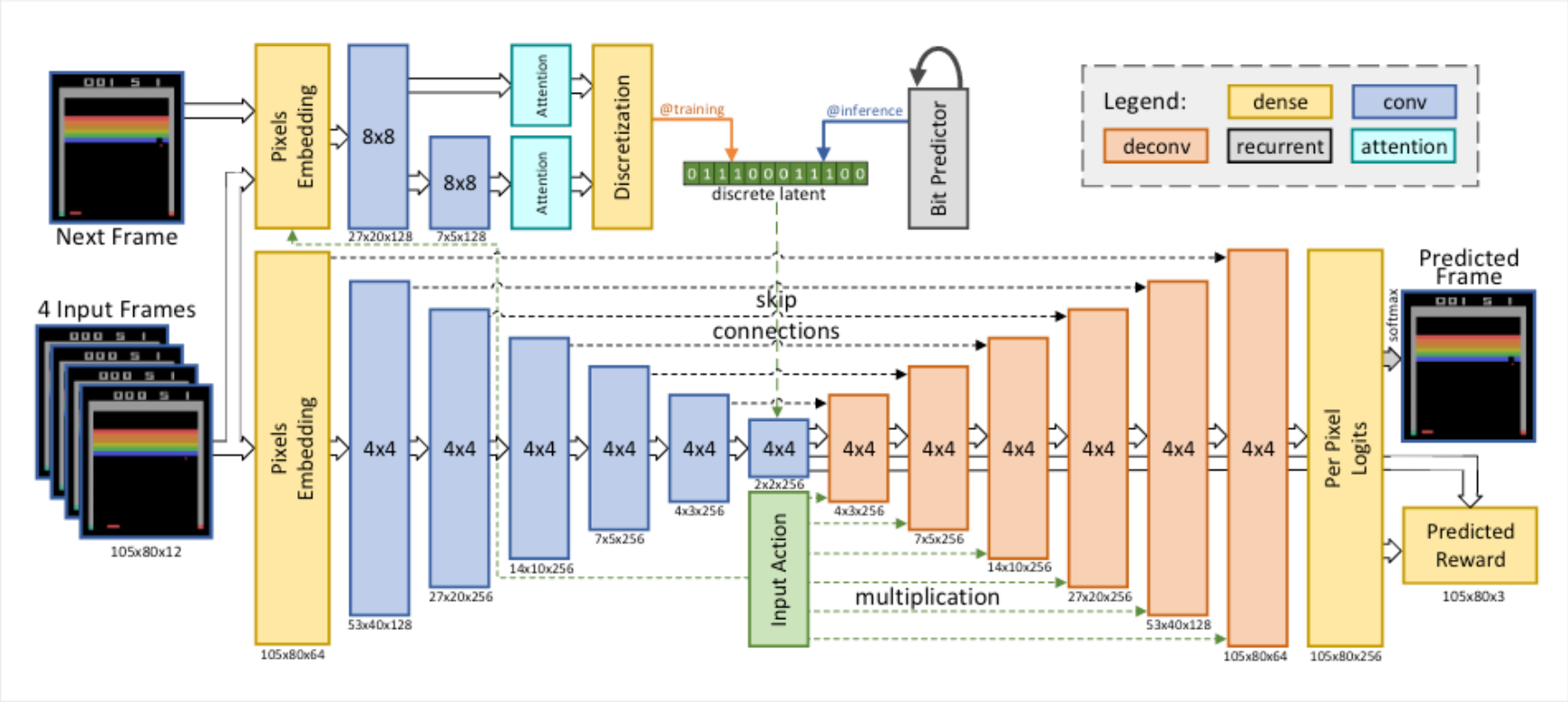
End-to-end transitions architecture, adapted from [7]

Latent Models



Architecture with latent model, taken from [7]

Latent Models in Atari case



Example of a model with discrete latent space, taken from [8]

Latent Models in Atari case

Algorithm 1: Pseudocode for SimPLe

Initialize policy π
Initialize model parameters θ of env'
Initialize empty set $\mathbf{D}$
while not done **do**
  ▷ collect observations from real env.
  $\mathbf{D} \leftarrow \mathbf{D} \cup \text{COLLECT}(env, \pi)$
  ▷ update model using collected data.
  $\theta \leftarrow \text{TRAIN\_SUPERVISED}(env', \mathbf{D})$
  ▷ update policy using world model.
  $\pi \leftarrow \text{TRAIN\_RL}(\pi, env')$
end while

Pseudocode for SimPLe, taken from [8]

Value Iteration Networks

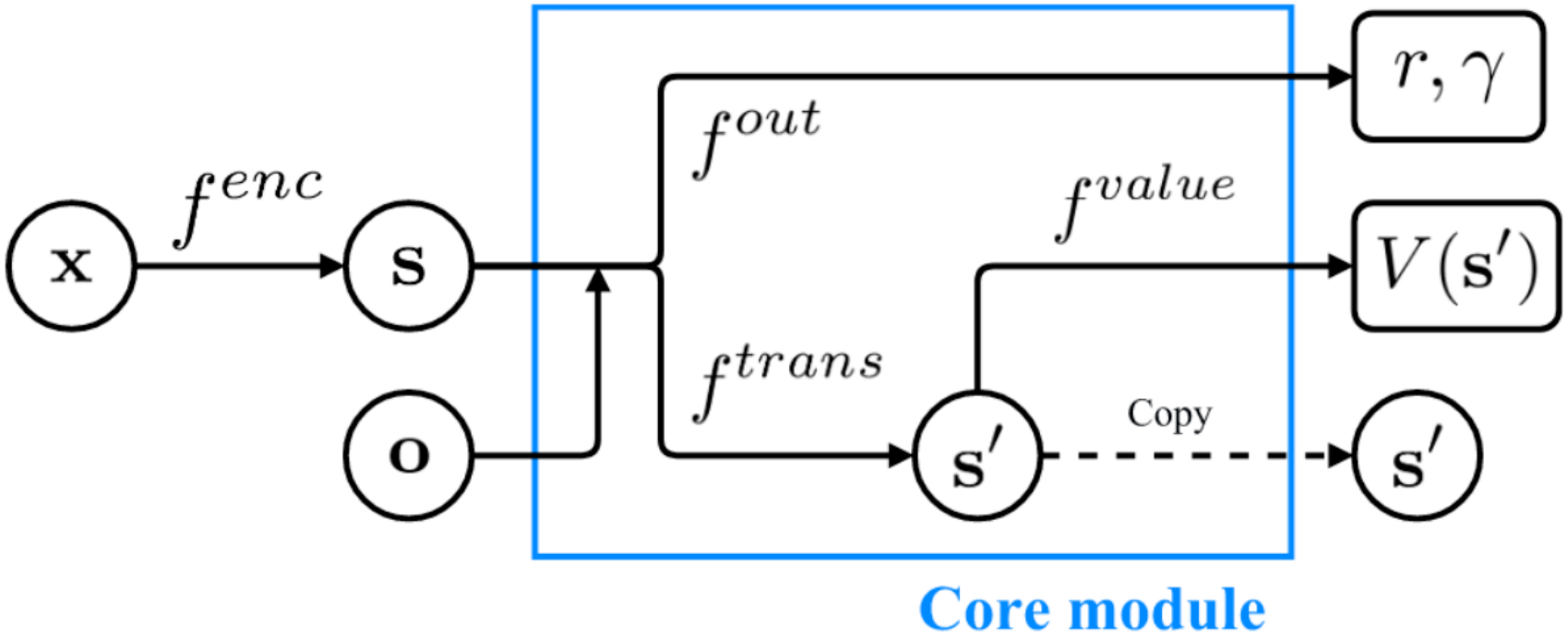
Research Question:

Is it possible to plan without predicting future observations?

Architecture setup [9]:

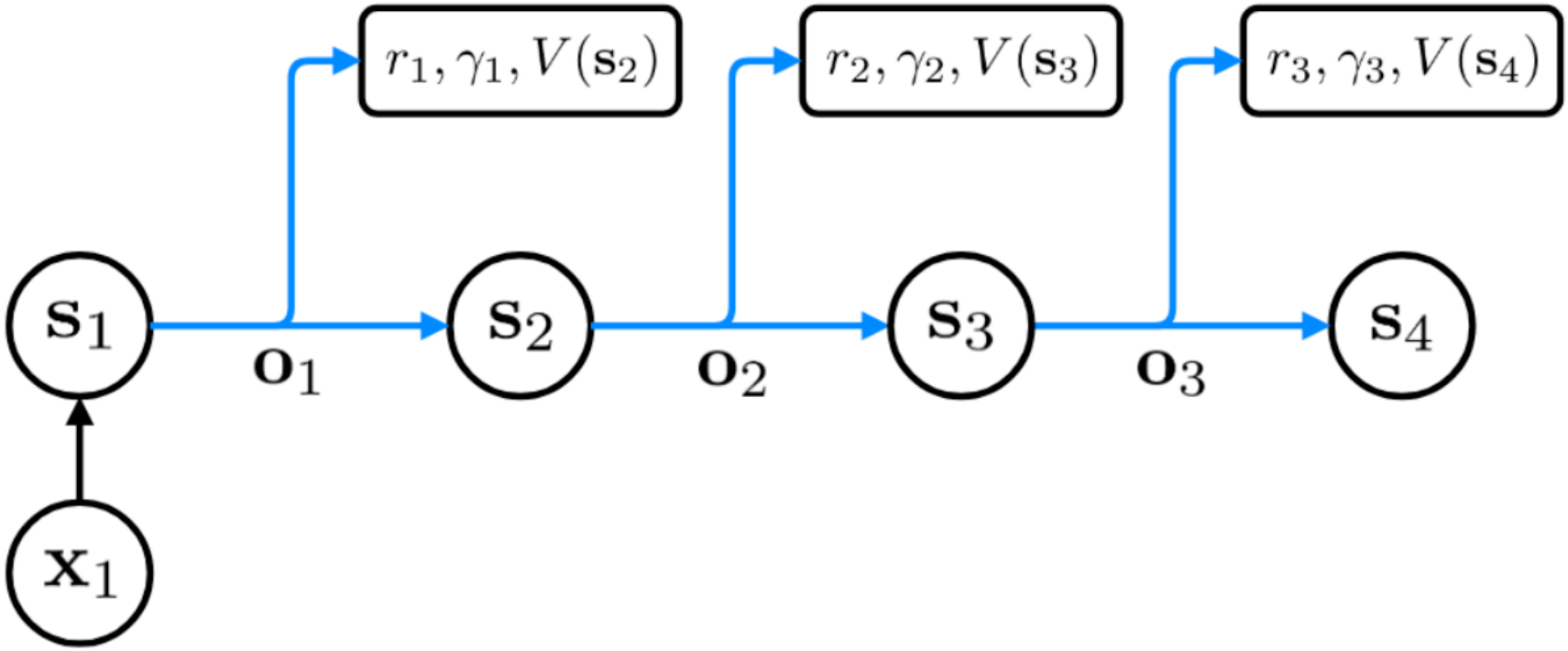
- Encoding $f_\theta^{enc} : x \mapsto s$
- Value $f_\theta^{value} : s \mapsto V_\theta(s)$
- Outcome $f_\theta^{out} : s, o \mapsto r, \gamma$
- Transition $f_\theta^{trans} : s, o \mapsto s'$
- Core $f_\theta^{core} : s, o \mapsto r, \gamma, V_\theta(s), s'$

Value Iteration Networks



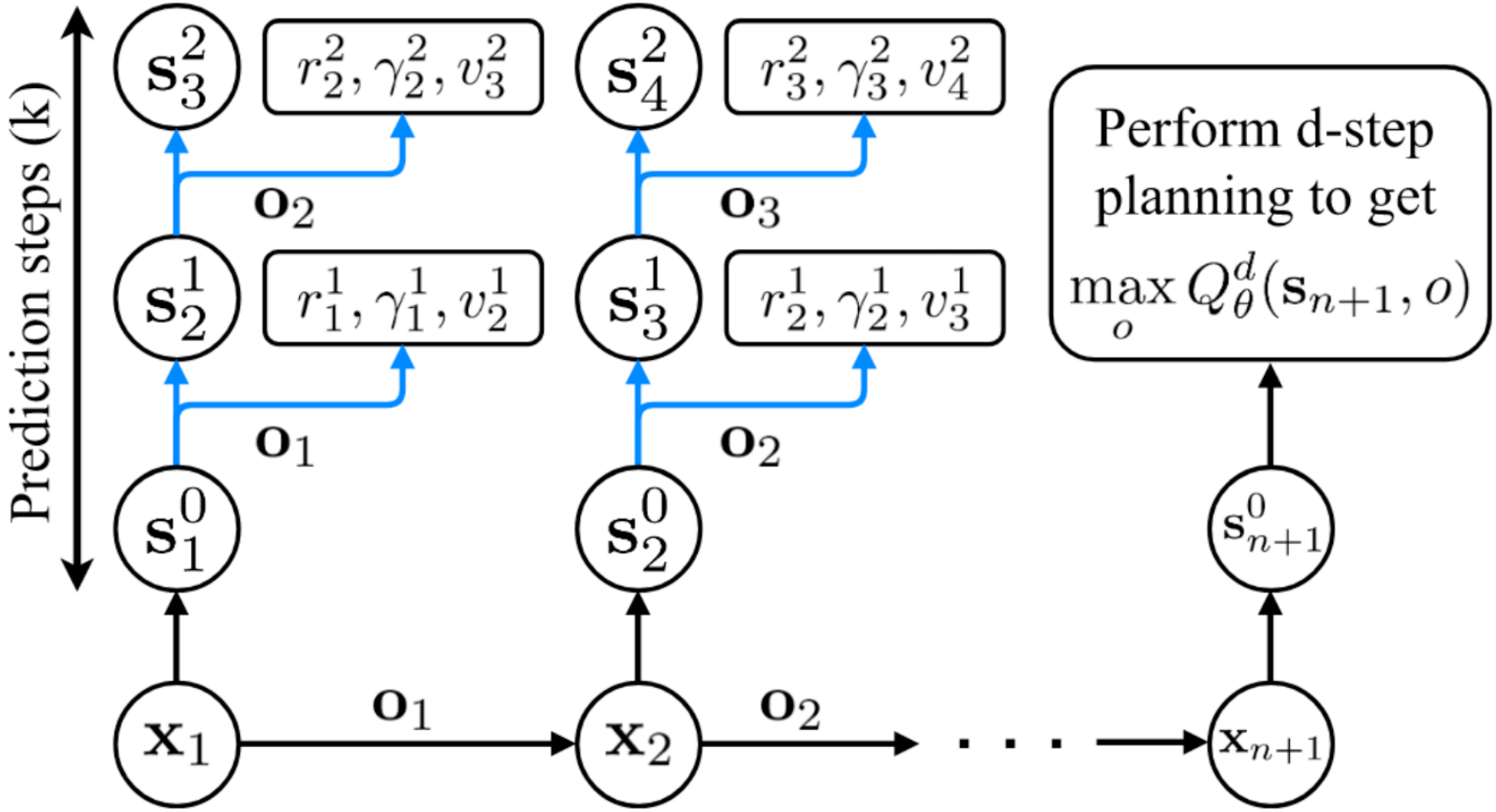
One-step rollout in Value Prediction Networks, taken from [9]

Value Iteration Networks



Multi-step rollout in Value Prediction Networks, taken from [9]

Value Iteration Networks



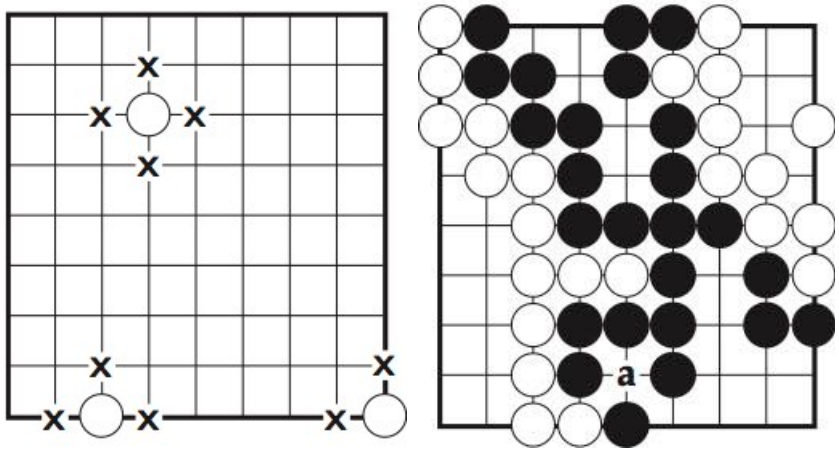
Learning process, taken from [9]

State of the Art

AlphaGo

Quick introduction to Go:

- Objective: "The main object of the game is to use your stones to **form territories** by surrounding vacant areas of the board. It is also possible to **capture** your opponent's stones by completely surrounding them." [10]
- (Simplified) Rules:



Go game. Taken from [10]

Setup for AlphaGo

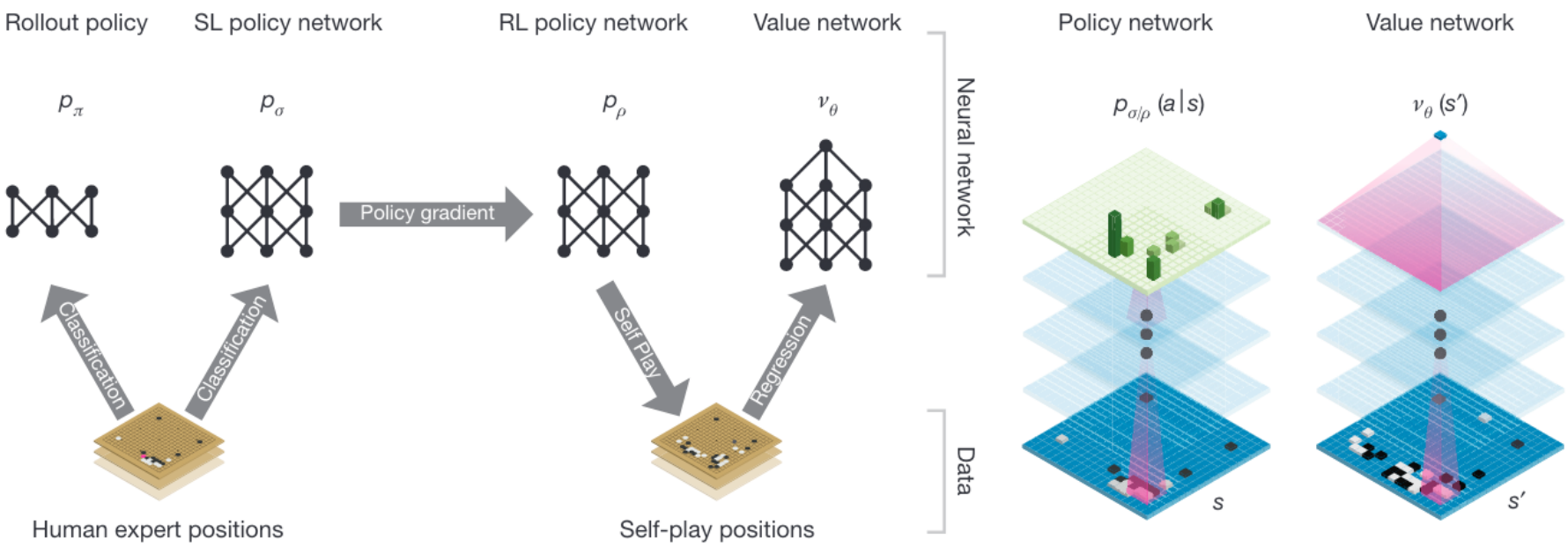
Motivation:

- The game of Go has $\sim 3^{19\times 19}$ states on a 19×19 board with black, white, and empty
- Previous work only achieved an amateur level.

Problem Setting:

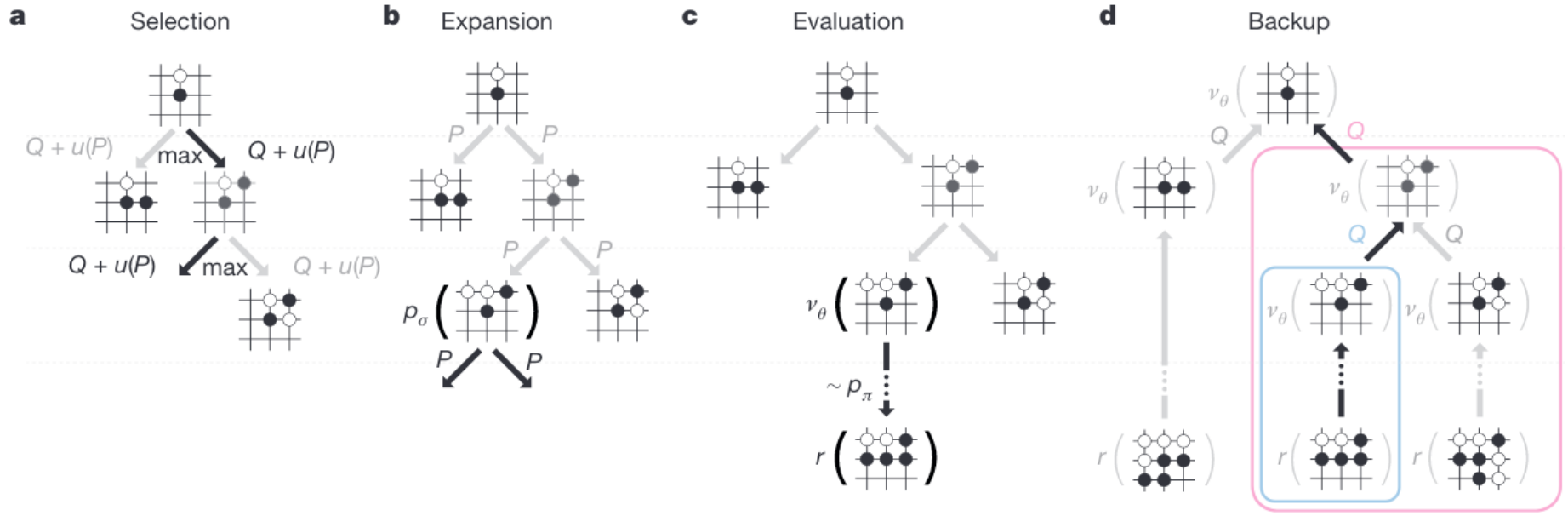
- $\mathcal{S}$ is composed of legal positions on the board
- $\mathcal{A}(s)$ is given by the legal moves on the state s
- The transition function $f(s, a)$ follows the rules of the game
- The reward is given at the end of the match by $z_t = \pm r(s_T)$
- Where $r(s_T)$ is the score of the player at the terminal state s_T

Training overview in AlphaGo



Neural network training pipeline and architecture, taken from [11]

Planning in AlphaGo



Monte Carlo tree search in AlphaGo, taken from [11]

Open Research Questions for Model-based RL

The following questions are active research among the RL community [7]:

- What is an **acceptable difference** between the model and the environment to achieve reasonable performance?
- How can we provide **robustness** and **safety guarantees** to model-based RL in complex systems?
- Is model-based RL more **explainable** and **trustworthy** than the model-free ones?
- What are the requirements to implement a model-based RL architecture in a **multi-agent** task?

Closing Remarks

Take-home Messages

- The main motivation behind model-based RL is to get more samples efficiently.
- There are three main approaches for model-based RL: explicit planning on **given** or **learned** transitions, and **End-to-end** learning of planning and transitions.
- The usefulness of model-based RL is found in the **generalization** and **planning** capabilities of the model and RL architecture.

Model-based RL in the Society You may watch the

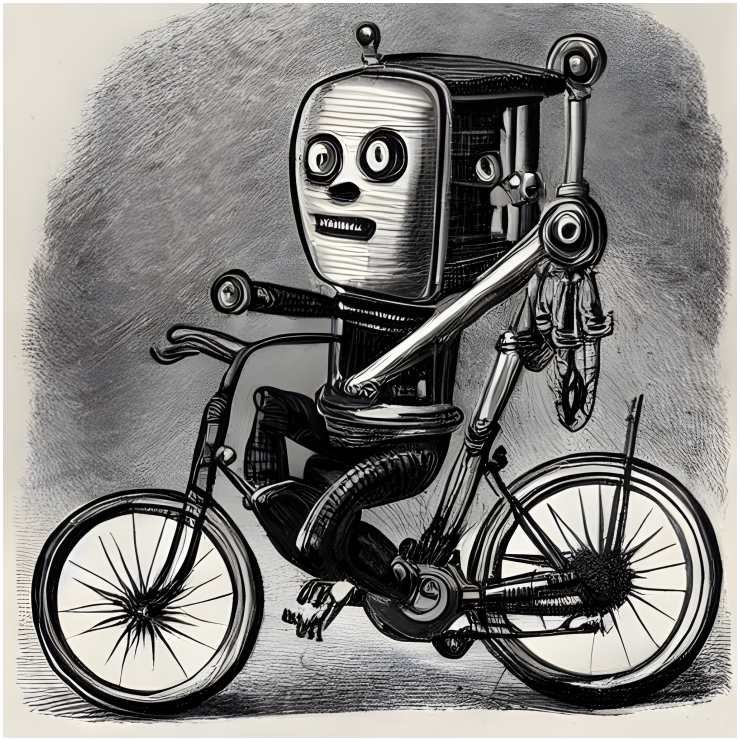
DeepMind's documentary about AlphaGo at the following link: youtube.com/...

Poster of AlphaGo Movie, taken from @alphagomovie in Twitter

Next Lecture

Is it easier to learn how to drive a moped if you already know how to drive a bike? How can we replicate this learning process in RL?

Meta RL



References

[1] D. P. Bertsekas, Dynamic programming and optimal control, 3rd ed., ser. Athena scientific optimization and computation series. Belmont, Mass: Athena Scientific, 2005, vol. 1, oCLC: 835987011.

[2] A. Plaat, Deep Reinforcement Learning, a textbook, 2022, arXiv:2201.02135 [cs]. [Online]. Available: http://arxiv.org/abs/2201.02135

[3] M. Tejedor, A. Z. Woldaregay, and F. Godtliebsen, "Reinforcement learning application in diabetes blood glucose control: A systematic review," Artificial Intelligence in Medicine, vol. 104, p. 101836, Apr. 2020.

[4] I. Fox and J. Wiens, "Reinforcement Learning for Blood Glucose Control: Challenges and Opportunities."

[5] R. S. Sutton and A. G. Barto, Reinforcement learning: an introduction, second edition ed., ser. Adaptive computation and machine learning series. Cambridge, Massachusetts: The MIT Press, 2018

[6] M. Schwenzer, M. Ay, T. Bergs, and D. Abel, “Review on model predictive control: an engineering perspective,” The International Journal of Advanced Manufacturing Technology, vol. 117, no. 5-6, pp. 1327–1349, Nov. 2021. [Online]. Available: https://link.springer.com/10.1007/s00170-021-07682-3

[7] A. Plaat, W. Kosters, and M. Preuss, “Deep Model-Based Reinforcement Learning for High-Dimensional Problems, a Survey,” Dec. 2020, arXiv:2008.05598 [cs]. [Online]. Available: http://arxiv.org/abs/2008.05598

[8] L. Kaiser, M. Babaeizadeh, P. Milos, B. Osinski, R. H. Campbell, K. Czechowski, D. Erhan, C. Finn, P. Kozakowski, S. Levine, A. Mohiuddin, R. Sepassi, G. Tucker, and H. Michalewski, “Model-Based Reinforcement Learning for Atari,” Feb. 2020, arXiv:1903.00374 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1903.00374

[9] J. Oh, S. Singh, and H. Lee, “Value Prediction Network,” Nov. 2017, arXiv:1707.03497 [cs]. [Online]. Available: http://arxiv.org/abs/1707.03497

[10] The British Go Association, “How to play,” 2017. [Online]. Available: https://www.britgo.org/intro/intro2.html

[11] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, “Mastering the game of Go with deep neural networks and tree search,” Nature, vol. 529, no. 7587, pp. 484–489, Jan. 2016. [Online]. Available: http://www.nature.com/articles/nature16961

See you next week