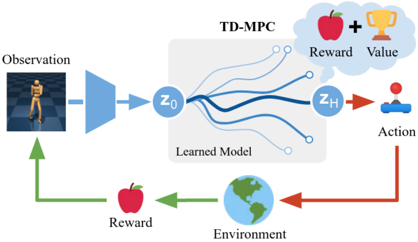
Evolution + deep RL

TD-MPC

Olivier Sigaud

Sorbonne Université
http://people.isir.upmc.fr/sigaud

Outline



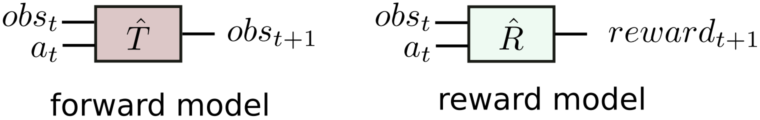
- A SoTA method for robot learning (sample efficient, time efficient, performs well)
- A flexible, multi-component algorithm: MBRL + Evo + MFRL + rep. learning
- Lots of inner synergies between components
- An older instance is POPLIN
- Roadmap:
  - We start with the model-based MPC component
  - Then we add the temporal difference component and outline synergies

Hansen, N., Wang, X., and Su, H. (2022) Temporal difference learning for model predictive control. *arXiv preprint arXiv:2203.04955*

Wang, T. and Ba, J. (2019) Exploring model-based planning with policy networks. *arXiv preprint arXiv:1906.08649*

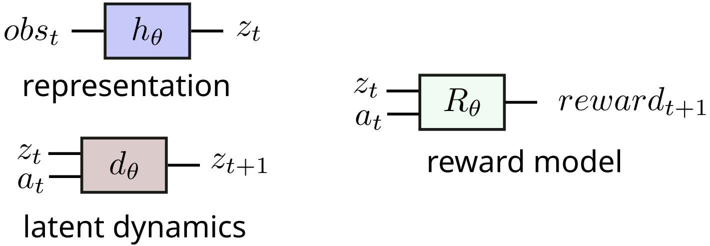
Model predictive control on learned models

Standard Model-Based RL



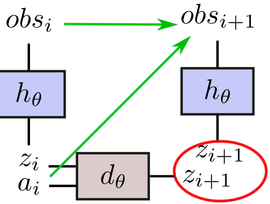
- The models can be used to predict a trajectory and its return "in imagination"
- If T is stochastic, irreducible aleatoric uncertainty
- Do not predict too far away in time (typically, 5 steps)
- When obs_t are images, need for representation learning → latent state z

Learning models for image-based MPC



- The dynamics is learnt in a latent space from $z_i = h_\theta(obs_i)$
- Then all other learning components get the latent state as input
- Latent dynamics is learnt with the consistency loss

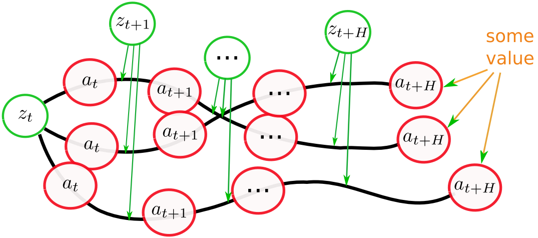
Consistency loss



$$loss = ||d_\theta(z_i, a_i) - h_\theta(obs_{i+1})||^2$$

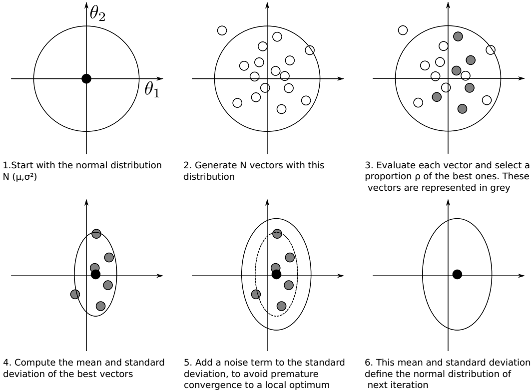
- Two ways to predict the next latent state should give consistent results
- Now that we have a model, how can we generate efficient actions?

Generating efficient sequences of actions



- With the latent dynamics, one can evaluate trajectories "in imagination", without sampling in the environment
- The next latent state is predicted through the latent dynamics model
- Trajectory values are estimated at the reached horizon from some measure
- (e.g. sum of immediate rewards)
- The CEM is used to optimize sequences of actions
- Initial sequences of actions are drawn in some way (e.g. randomly)

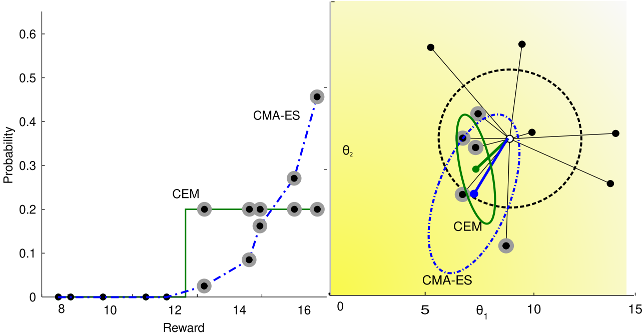
Reminder: cross-entropy method



- Here, an example where $\boldsymbol{\theta}$ is 2D

Marin, and Sigaud, O. (2012) Towards fast and adaptive optimal control policies for robots: A direct policy search approach, *Proceedings conference Robotica*, pp. 21-26

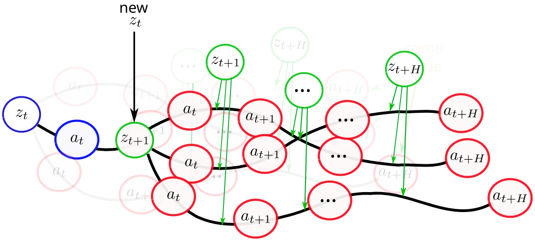
CMA-ES vs CEM



- The stronger the yellow, the higher the return
- CMA-ES uses many additional tricks
- CEM is more used in RL problems

Hansen, N. & Auger, A. (2011) CMA-ES: evolution strategies and covariance matrix adaptation. In *Proceedings of the 13th annual conference companion on Genetic and evolutionary computation* (pp. 991–1010)

Model Predictive Control (MPC)

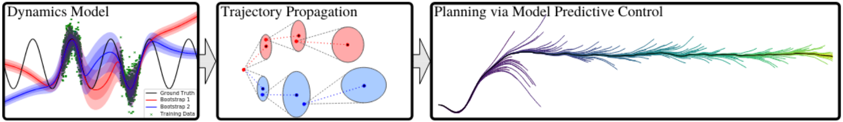


- Some sequence of actions is selected based on the value
- The first action is played (or the few first actions): receding horizon
- MPC is run again from the new current state
- And so on until the end of the episode

Kouvaritakis, B. and Cannon, M. (2016) Model predictive control. *Switzerland: Springer International Publishing*, 38:13–56

Kwon, W. H. and Han, S. (2005) *Receding Horizon Control: Model Predictive Control for State Models*. Springer

Weaknesses of MPC

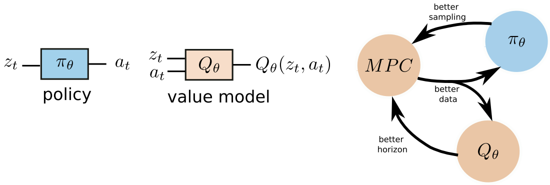


- PETS presents itself as the first paper combining CEM and MPC
  - But PETS uses ensembling over models to measure uncertainty
  - Makes it possible to distinguish aleatoric and epistemic uncertainty, and perform active learning
- The combination of MPC + CEM to optimize sequences of actions:
  - Suffers from long inference time (evaluating many sequences)
  - Particularly true if starting from random actions
  - Does not predict beyond the MPC horizon
- The TD part improves this

Chua, K., Calandra, R., McAllister, R., and Levine, S. (2018) Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31

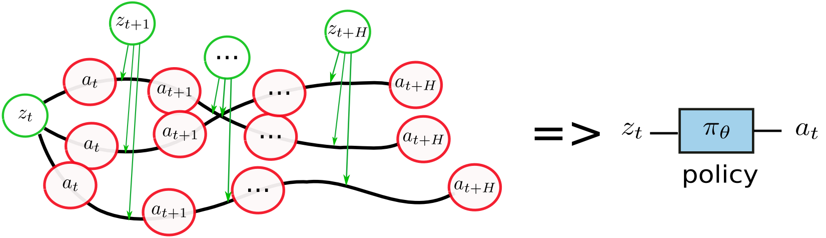
Adding temporal difference learning

Adding Temporal Differences: synergies



- TD-MPC adds a temporal difference component to MPC, with three synergistic mechanisms:
  1. The MPC trajectories are used to improve policy anc critic learning
  2. The policy is used to warm-start the MPC process with good actions
  3. The action value function $Q_\theta(z, a)$ is used to evaluate MPC trajectories beyond the horizon
- Resulting advantages:
  - Main point: A policy network triggers actions $\rightarrow$ much faster inference
  - The combination is less myopic than a single action policy
  - The combination is less myopic than a limited horizon MPC
  - The MPC part is faster and performs better

TD from CEM trajectories

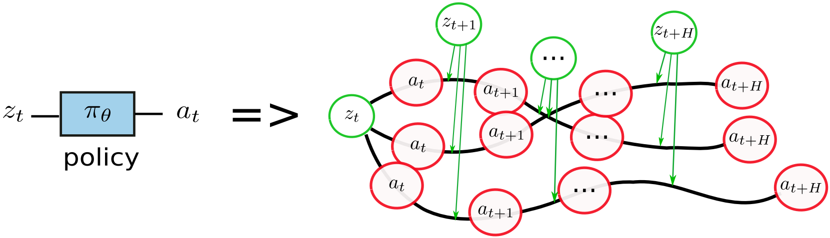


- Several approaches:
  - Simple behavioral cloning of CEM trajectories
  - Intermediate: RWR or AWR on CEM trajectories
  - Pure TD learning from a buffer of CEM trajectories
- CEM trajectories help with more efficient policy learning samples
- Less myopic than standard TD learning

Peters, J. and Schaal, S. (2007) Reinforcement learning by reward-weighted regression for operational space control. In Ghahramani, Z., editor, *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, volume 227 of *ACM International Conference Proceeding Series*, pages 745–750. ACM

Peng, X. B., Kumar, A., Zhang, G., and Levine, S. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019

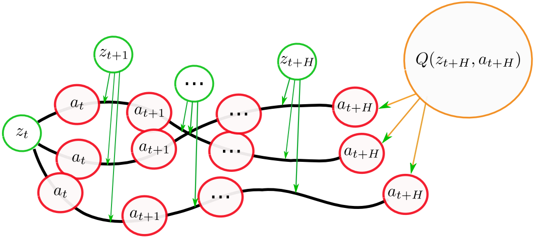
Warm-starting MPC



- The TD policy suggests appropriate actions
- More efficient than random sampling
- In PHIHP, we take a mix between random actions and policy actions

El Asri, Z., Sigaud, O., and Thome, N. (2024) Physics-informed model and hybrid planning for efficient Dyna-style reinforcement learning. *arXiv preprint arXiv:2407.02217*

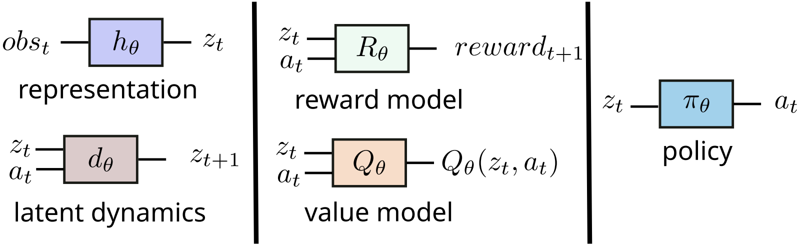
Watching beyond the horizon



- The value of trajectories is estimated from the action value model at the horizon
- The action value model summarizes the return of the rest of the episode
- A kind of improved n-step return (bias against variance)
- Less myopic than PETS-like approaches

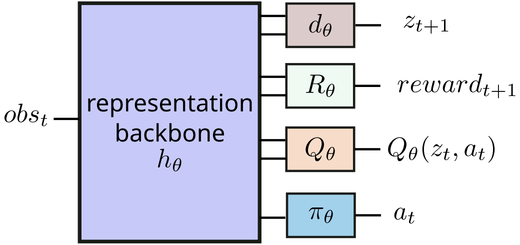
Putting everything together

Full TD-MPC models: Losses



- Samples from a replay buffer $\{obs_i, a_i, r_{i+1}, obs_{i+1}\}$
- Get $z_i = h_\theta(obs_i)$
- Losses:
  - Latent state consistency: $||d_\theta(z_i, a_i) - h_\theta(obs_{i+1})||^2$
  - Reward: $||R_\theta(z_i, a_i) - r_{i+1}||^2$
  - Value: $||r_{i+1} + \gamma Q_\theta(z_{i+1}, \pi_\theta(z_{i+1})) - Q_\theta(z_i, a_i)||^2$ (DDPG-like)
- TOLD model: Task-Oriented Latent Dynamics

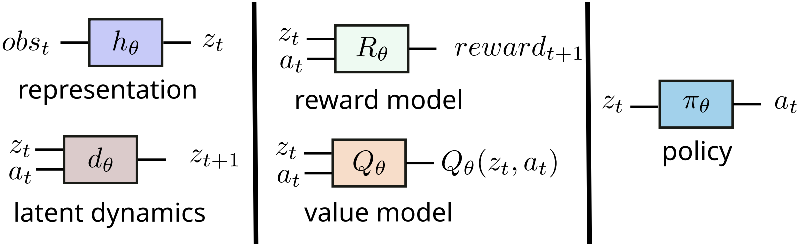
A clearer view: representation learning backbone



$$L(\theta) = c_1||R_\theta(z_i, a_i) - r_{i+1}||^2 + c_2||r_{i+1} + \gamma Q_\theta(z_{i+1}, \pi_\theta(z_{i+1})) - Q_\theta(z_i, a_i)||^2 + c_3||d_\theta(z_i, a_i) - h_\theta(obs_{i+1})||^2 \quad (1)$$

- All gradients naturally backpropagate into the representation backbone
- c_1, c_2 and c_3 are additional hyper-parameters

Other option: independent modules

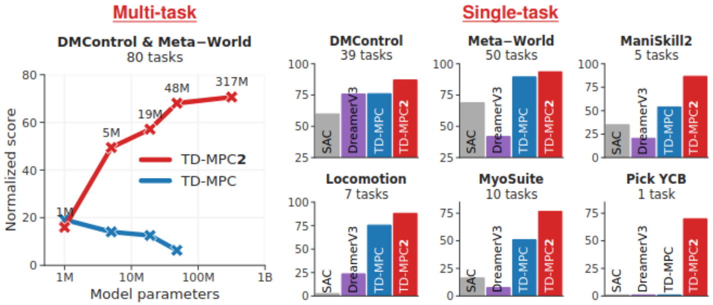


- Learn $h_\theta(obs_i)$ just with the consistency loss
- Freeze $h_\theta(obs_i)$ when backpropagating other losses
- Particularly obvious when learning from states (no h_θ backbone)
- Comparing both options is missing

Implementation details

- Uses LayerNorm in the TD part (useful recipe in many recent RL algorithms)
- For image-based experiments, h_θ is a 4-layer CNN with kernel sizes (7, 5, 3, 3), stride (2, 2, 2, 2), and 32 filters per layer.
- Might use more modern vision modules: DINOv2, ViT, ResNet, SigLip...
- Uses Prioritized Experience replay. Removed in TD-MPC2.
- Many hyper-parameters...
- More details in the paper

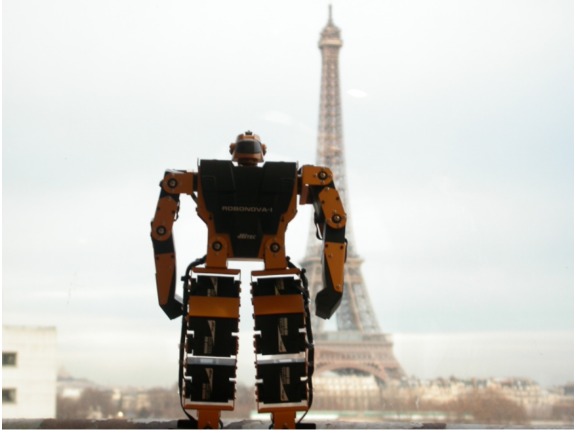
TD-MPC2



- Applied to large NNs with LayerNorm and SimNorm (5M vs 1M params)
- Uses SAC instead of TD3, no prioritized exp. replay, ensemble of Q functions...
- Conditioned on a task embedding

Hansen, N., Su, H., and Wang, X. (2023) TD-MPC2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*

Any question?



Send mail to: `Olivier.Sigaud@isir.upmc.fr`

Chua, K., Calandra, R., McAllister, R., and Levine, S. (2018).
Deep reinforcement learning in a handful of trials using probabilistic dynamics models.
Advances in neural information processing systems, 31.

El Asri, Z., Sigaud, O., and Thome, N. (2024).
Physics-informed model and hybrid planning for efficient dyna-style reinforcement learning.
arXiv preprint arXiv:2407.02217.

Hansen, N. and Auger, A. (2011).
Cma-es: evolution strategies and covariance matrix adaptation.
In *Proceedings of the 13th annual conference companion on Genetic and evolutionary computation*, pages 991–1010.

Hansen, N., Su, H., and Wang, X. (2023).
TD-MPC2: Scalable, robust world models for continuous control.
arXiv preprint arXiv:2310.16828.

Hansen, N., Wang, X., and Su, H. (2022).
Temporal difference learning for model predictive control.
arXiv preprint arXiv:2203.04955.

Kouvaritakis, B. and Cannon, M. (2016).
Model predictive control.
Switzerland: Springer International Publishing, 38:13–56.

Kwon, W. H. and Han, S. (2005).
Receding Horizon Control: Model Predictive Control for State Models.
Springer.

Marin, D. and Sigaud, O. (2012).
Towards fast and adaptive optimal control policies for robots: A direct policy search approach.
In *Proceedings Robotica*, pages 21–26, Guimaraes, Portugal.

Peng, X. B., Kumar, A., Zhang, G., and Levine, S. (2019).

Advantage-weighted regression: Simple and scalable off-policy reinforcement learning.
arXiv preprint arXiv:1910.00177.

Peters, J. and Schaal, S. (2007).
Reinforcement learning by reward-weighted regression for operational space control.
In Ghahramani, Z., editor, *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, volume 227 of *ACM International Conference Proceeding Series*, pages 745–750. ACM.

Wang, T. and Ba, J. (2019).
Exploring model-based planning with policy networks.
arXiv preprint arXiv:1906.08649.