Casting Sim2Real as
Meta-Reinforcement Learning

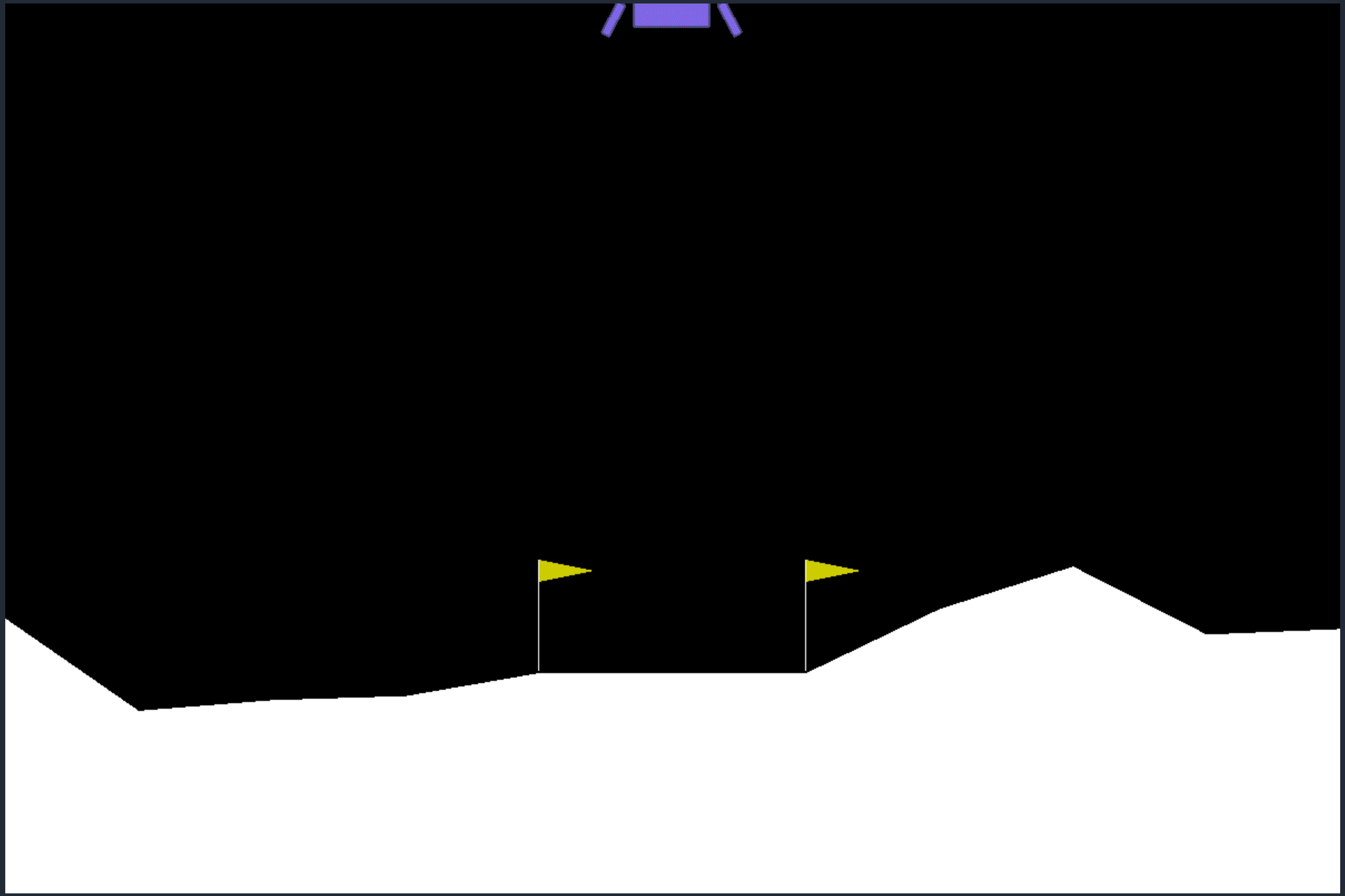
Agenda

- Motivation
- What‘s new
- PEARL
- PEARL2
- Experiments

Motivation

1. Sim2Real describes the problem of transferring a policy learned in simulation to the real world
2. Simulations are great because they are low risk and low cost
3. Ultimate goal is to train agent in simulation and adapt the policy in real world in a sample efficient way
4. Investigate meta-reinforcement learning on distribution of simulated tasks for sample efficient adoption in new tasks

Our Environement

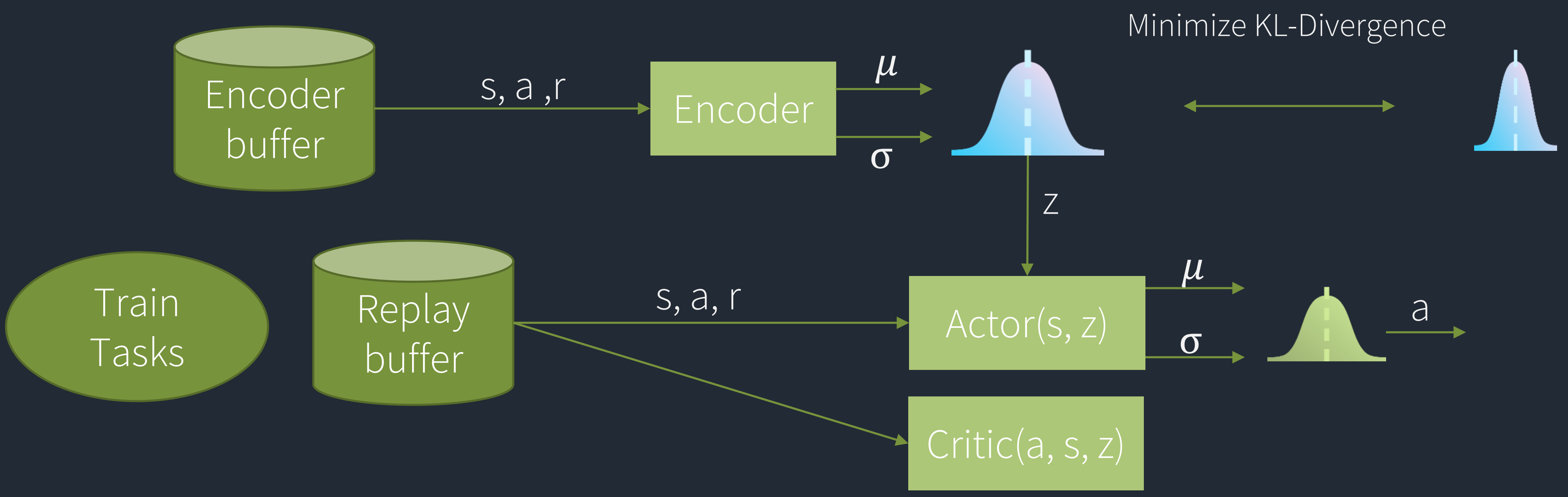


What‘s new

- Implemented PEARL
- Implemented variation of PEARL
- Out of distribution experiments
- Random wind dynamics
- Many experiments

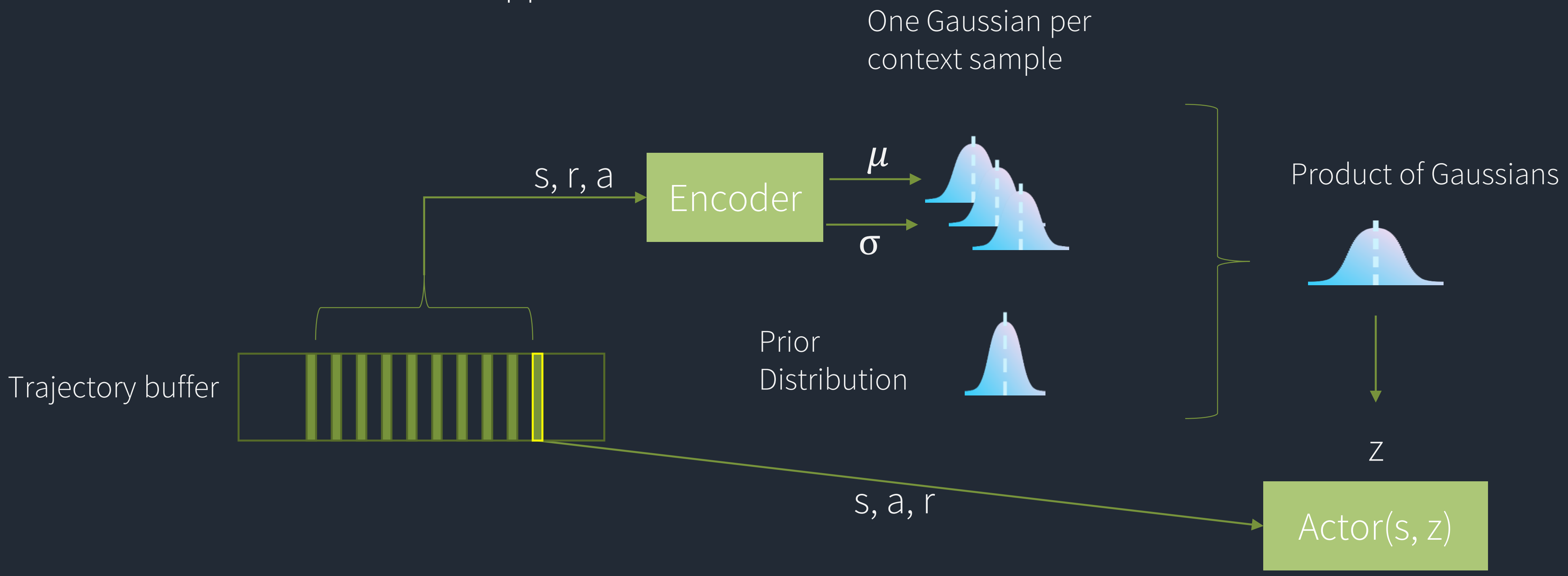
PEARL

- Off policy meta reinforcement learning
- Based on SAC
- Probabalistic context model to condition actor and critic



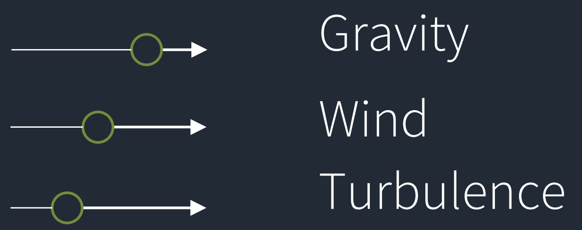
PEARL2

- Only use context from current trajectory
- Infer posterior distribution during trajectory execution
- More realistic approach

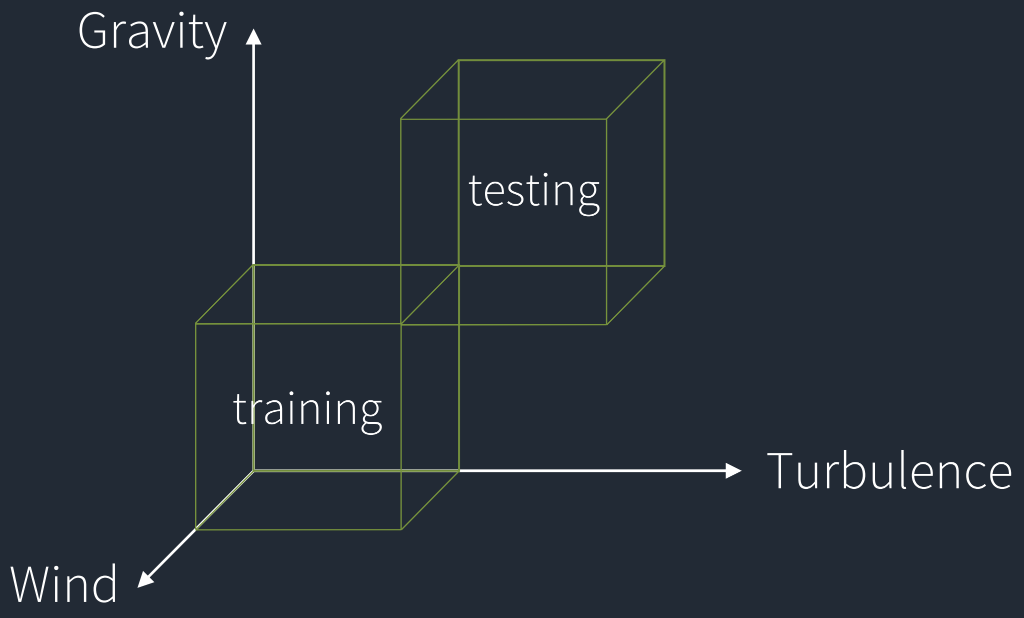


Experiments

- Domain randomization
  - Random training parameters
  - Validation on grid

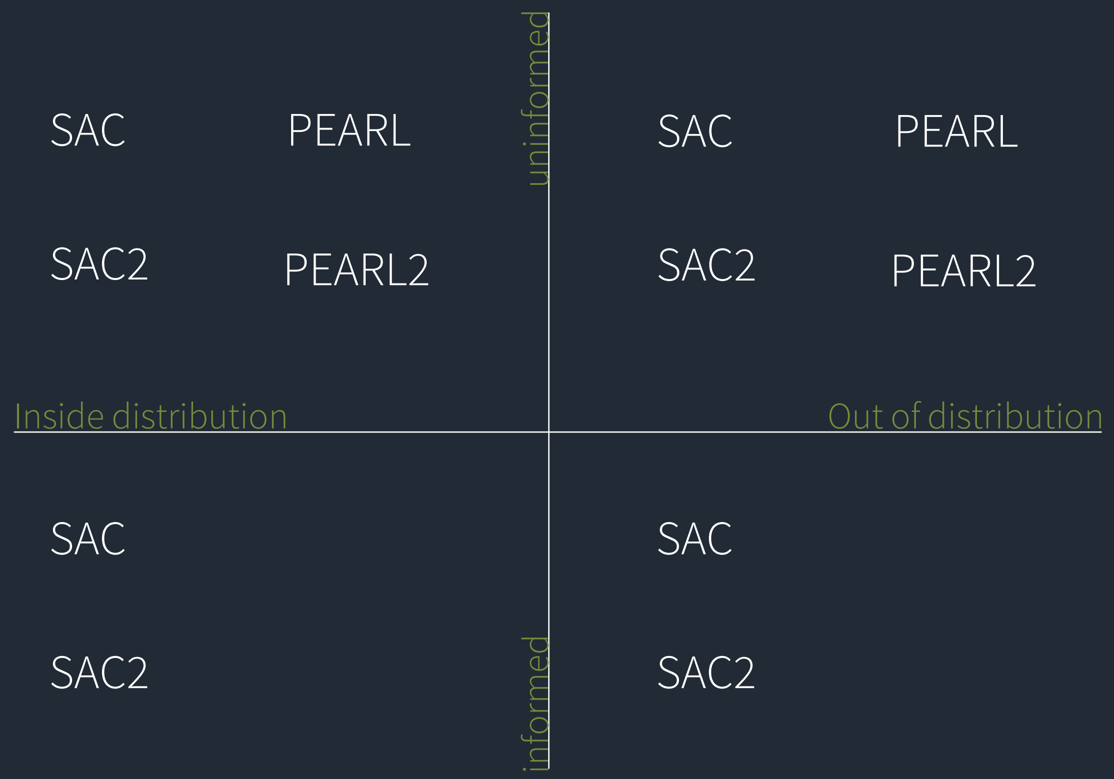


- Out of distribution
  - Random training parameters
  - Validation on grid

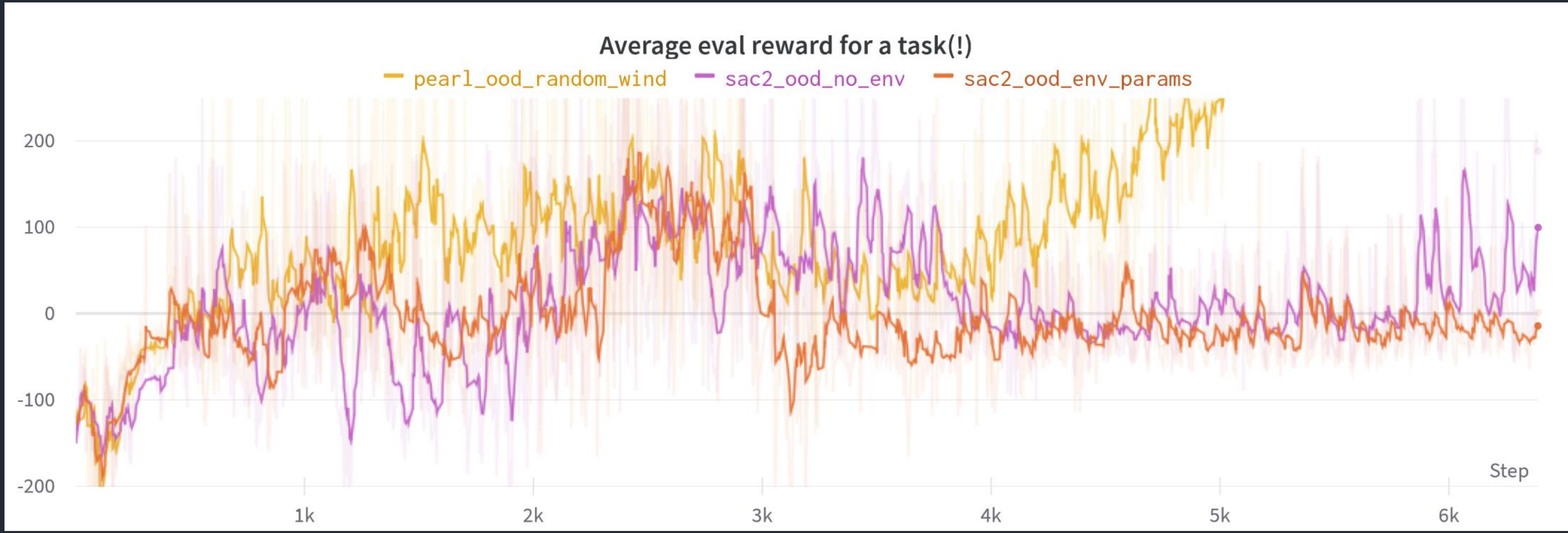


Experiments

- Comparing models
  - SAC
  - SAC2
  - PEARL
  - PEARL2
- Different modes
  - Inside distribution
  - Out of distribution
  - Passing parameters
  - Not passing parameters
  - Fixed wind
  - Random wind

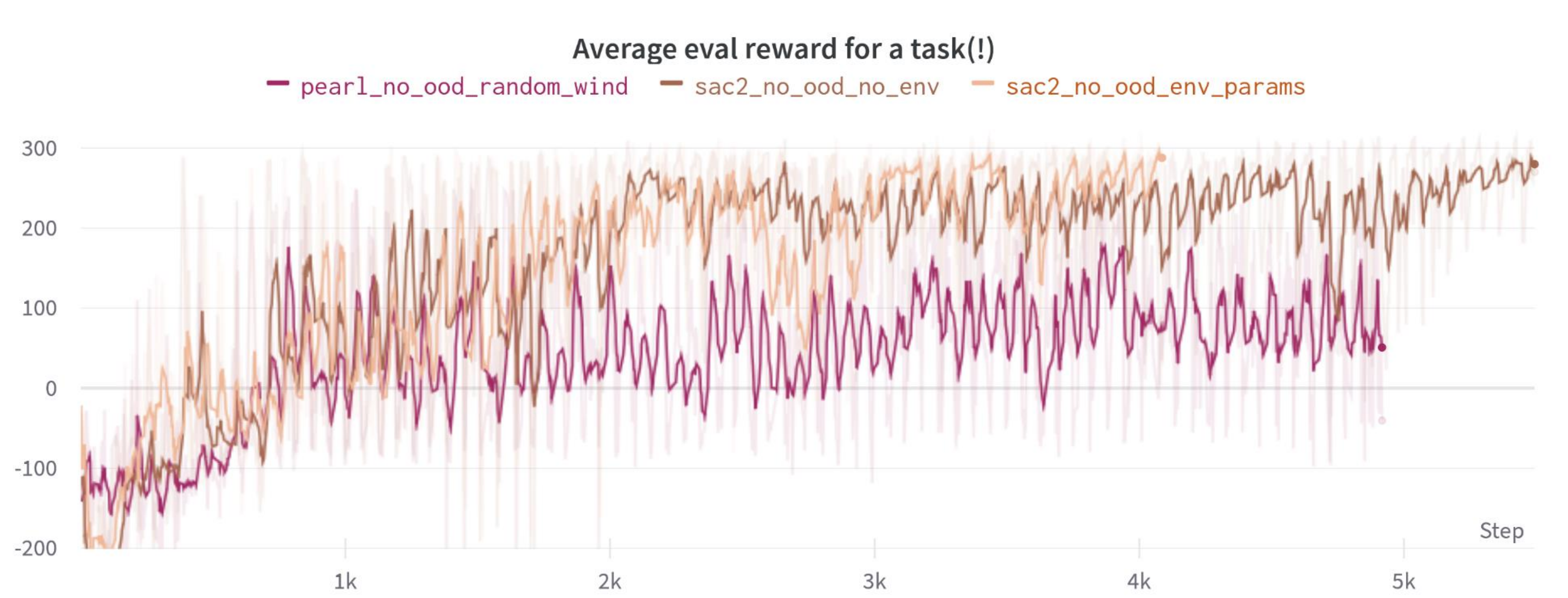


Experiments - OOD



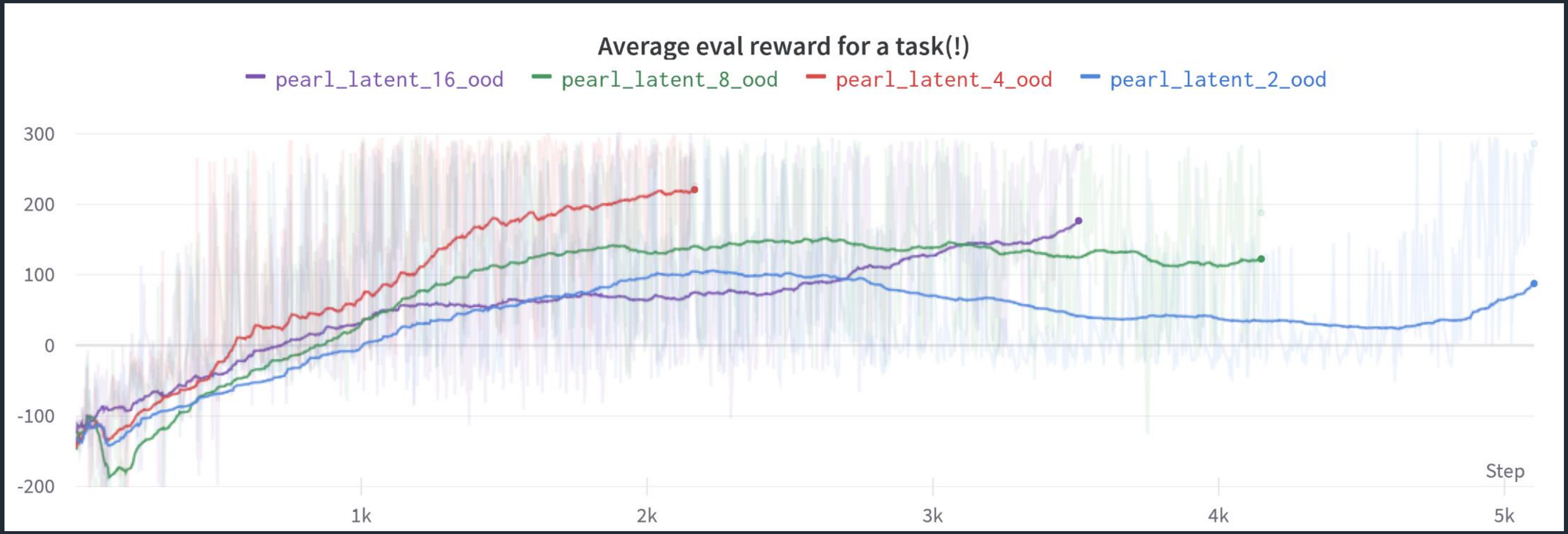
- PEARL outperforms SAC & SAC2 in OOD case
- PEARL solved all 27 evaluation tasks after 5k steps
- (in that set of experiments PEARL has 5 latent dimensions)

Experiments - ID



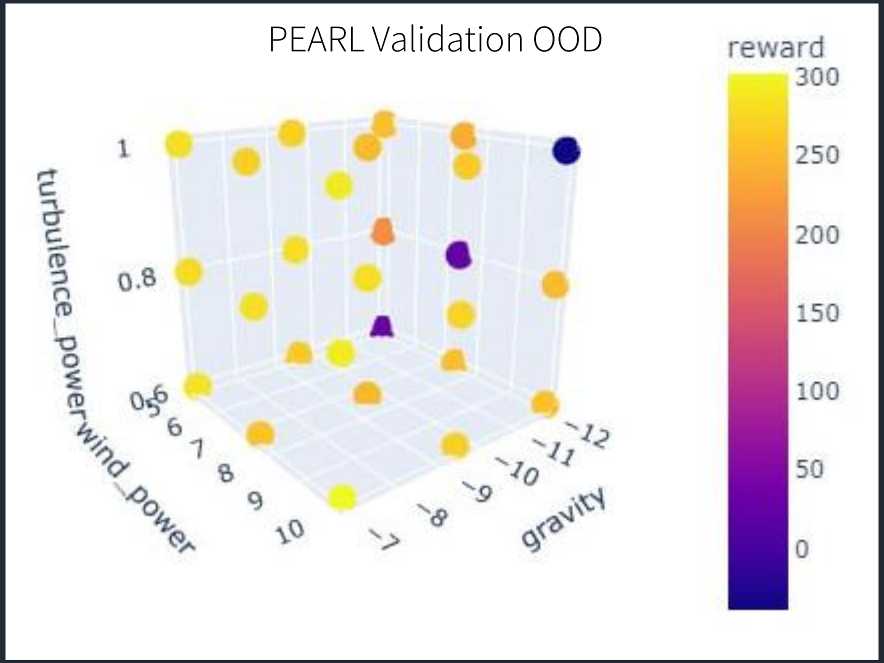
- SAC & SAC2 outperform for PEARL inside distribution tests
- Latent size of 5

Experiments – PEARL latent size



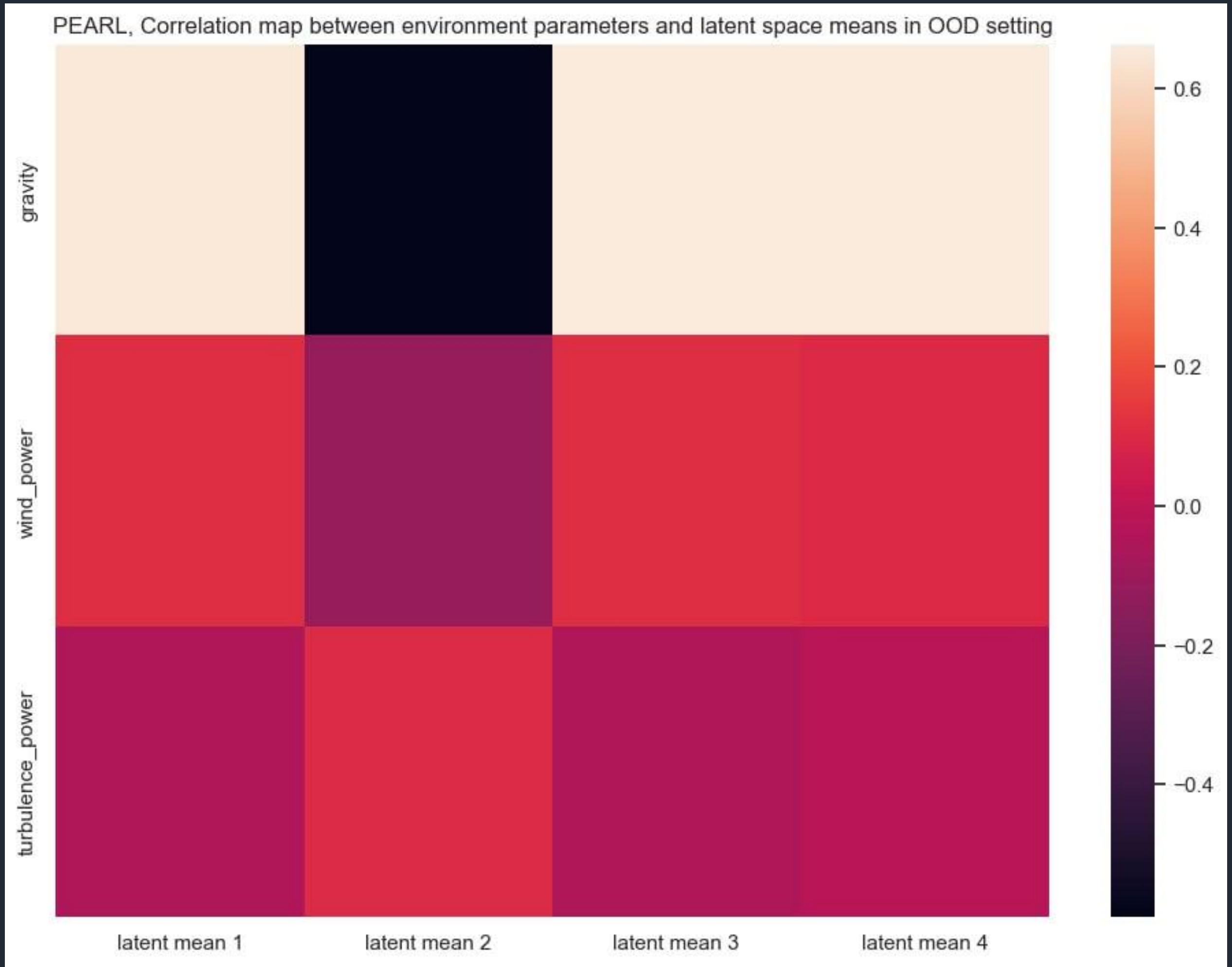
- PEARL performs best for latent size of 4
- More than 4 overfits, less than 4 underfits

Experiments – Validation Hypercube



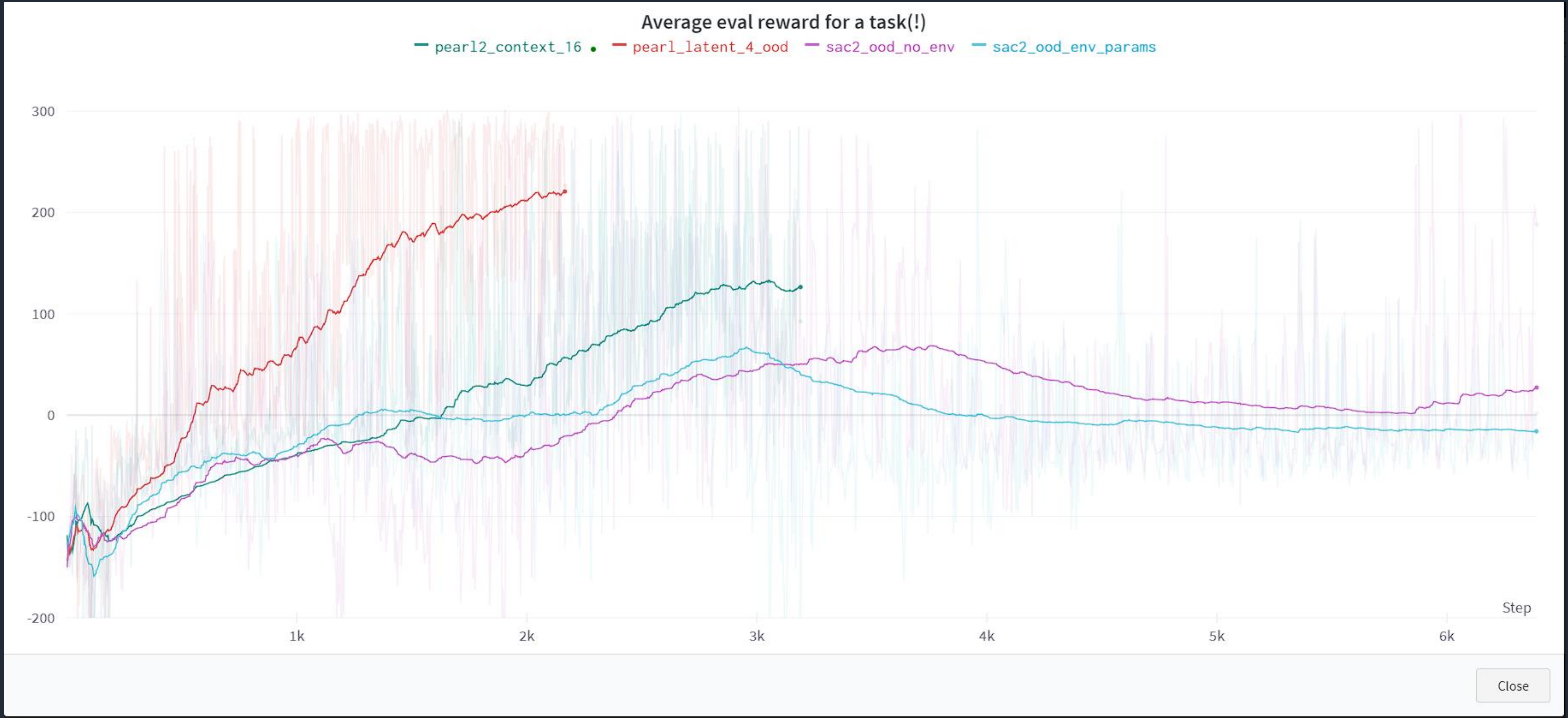
- In OOD validation PEARL performs worst for high gravity setting
- Gravity has strongest impact on reward from all three parameters
- PEARL was trained on low gravity

Experiments – Latent Correlation Map



- How does PEARL encode environment parameters?
- Explored for the best model with 4 latens variables
- Correlation map shows:
  - Latent variable 1 & 3 have same correlations
  - Latent variable 2 is orthogonal
  - variable 4 slightly different to 1 & 2

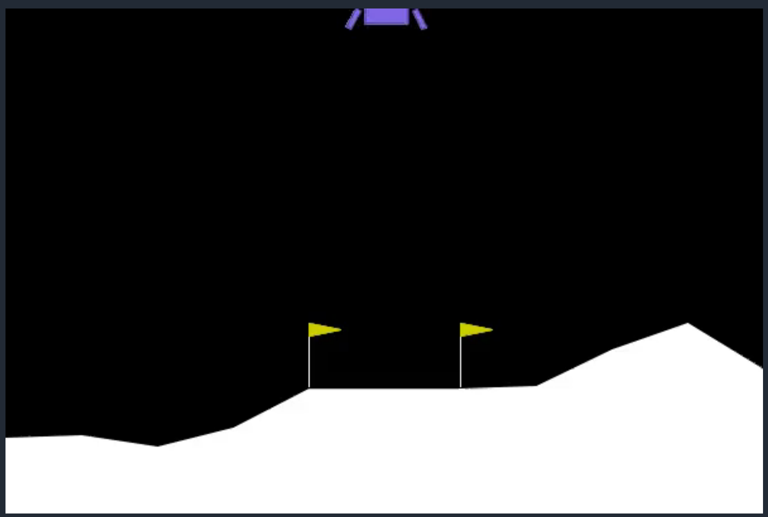
Perfomance of PEARL2



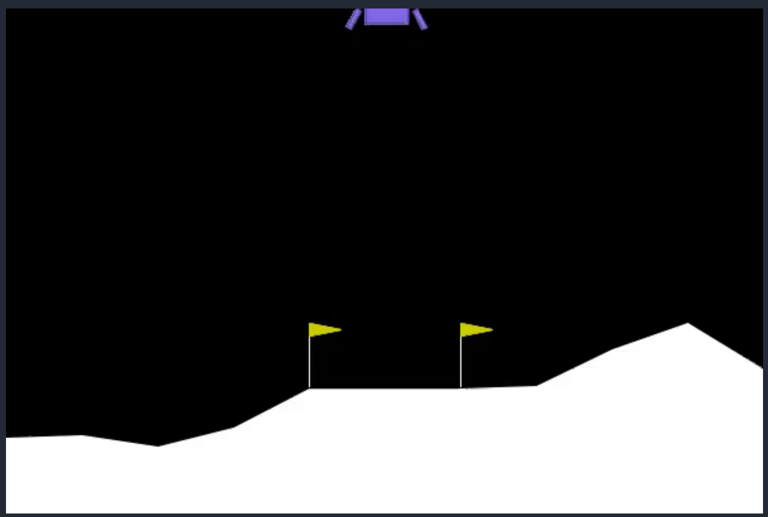
- PEARL2 was trained in hurry with small batches, still outperforms SAC

Demonstration – the hardest case

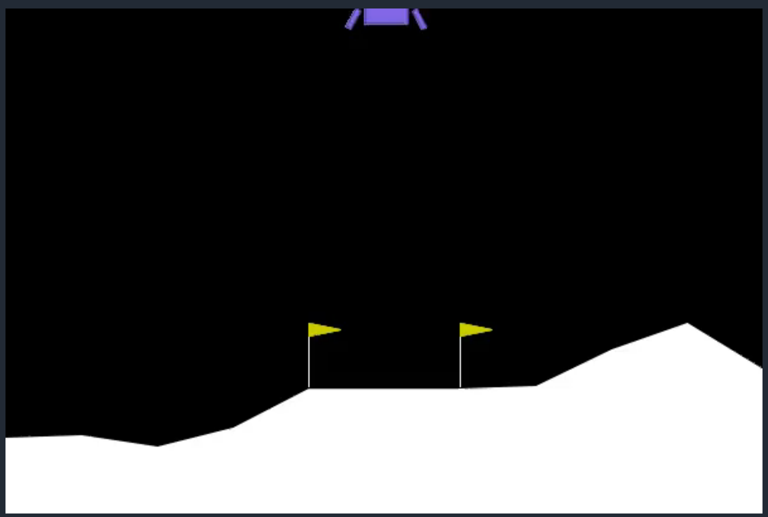
PEARL



SAC2



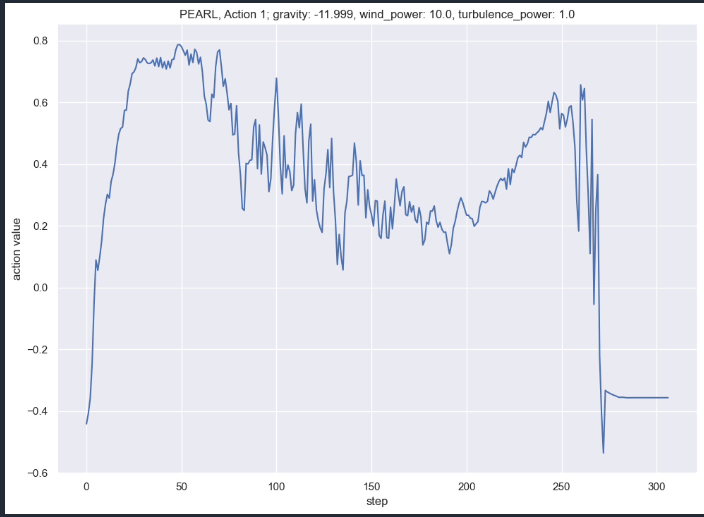
PEARL2

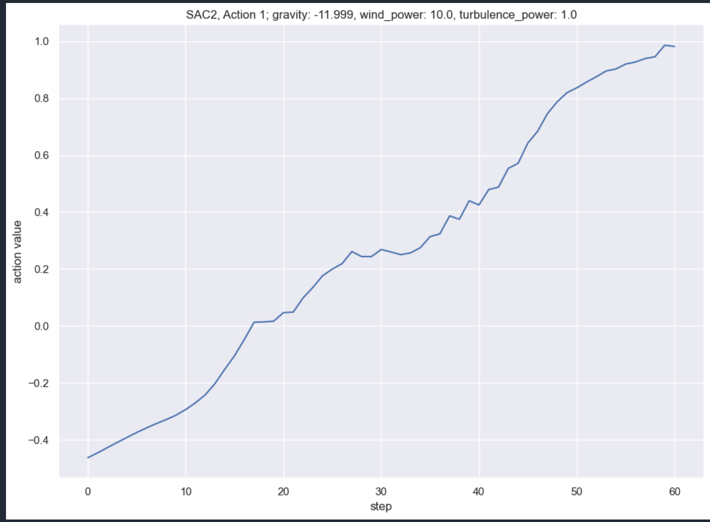


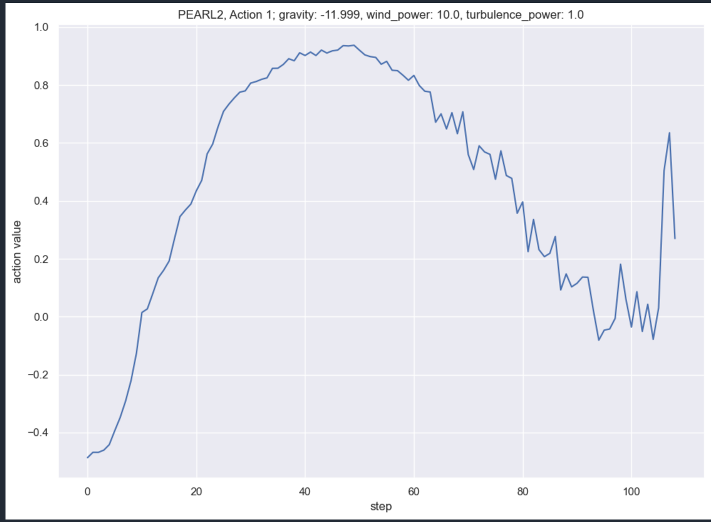
Action comparisons

PEARL | SAC2 | PEARL2

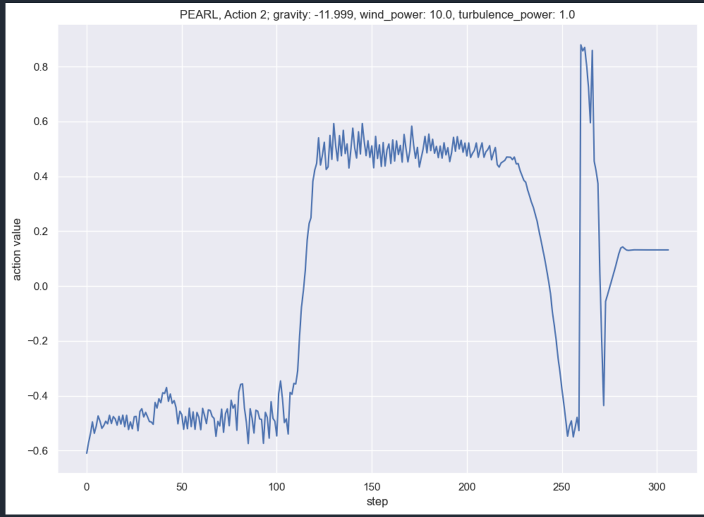
main

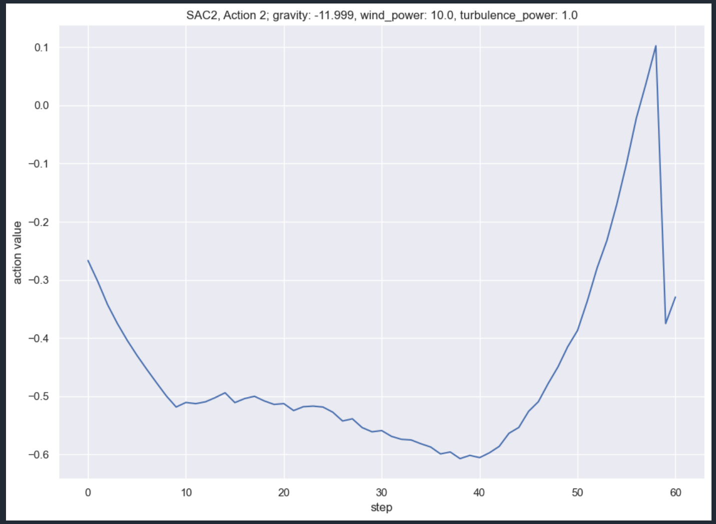


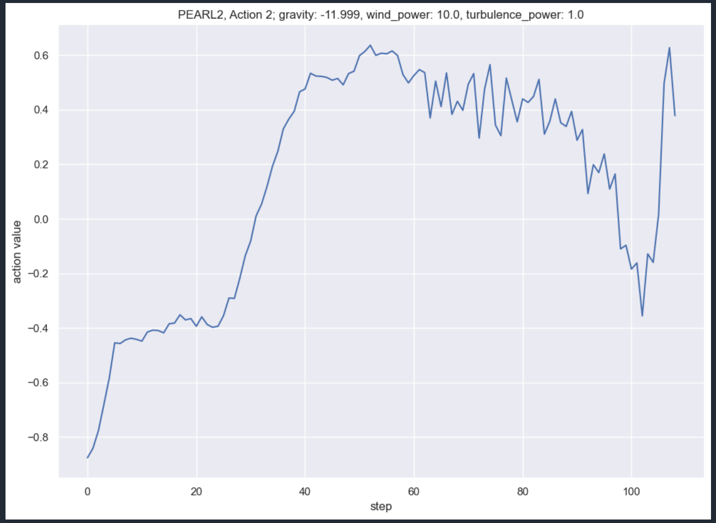




side

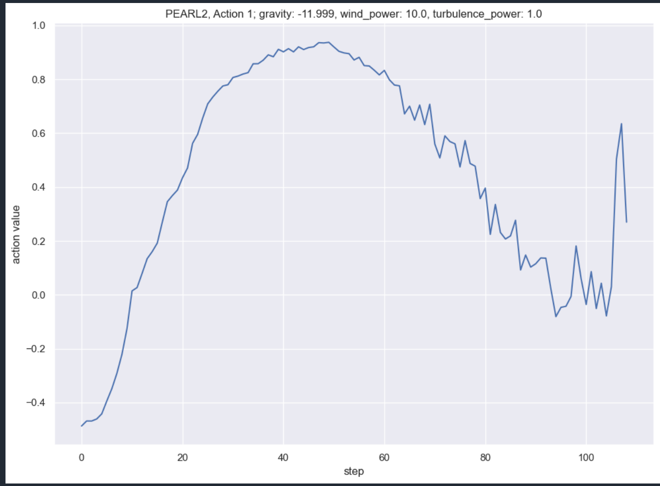




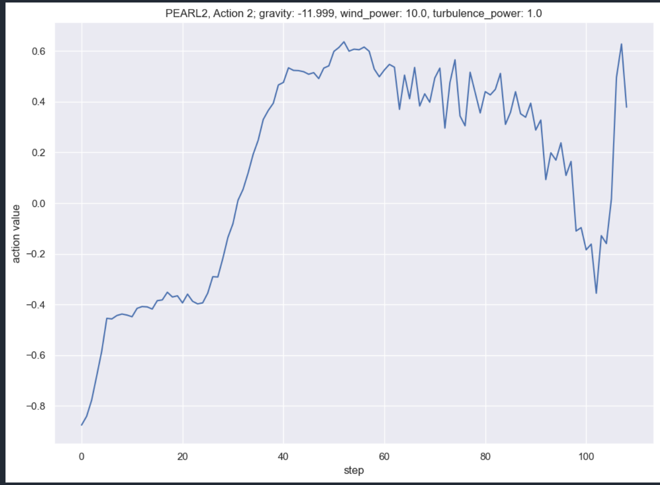


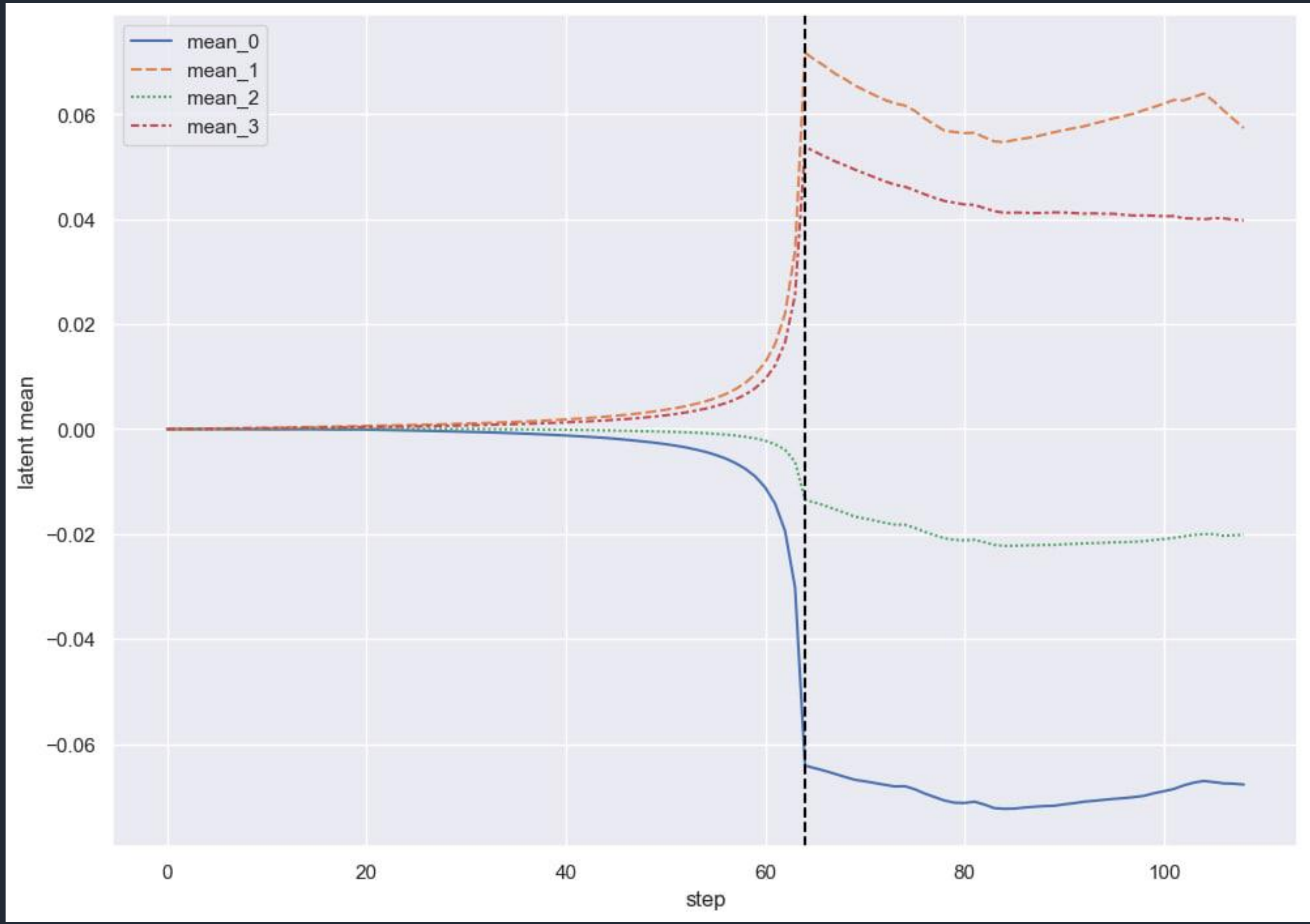
Dynamics of latent variables in PEARL2

main



side

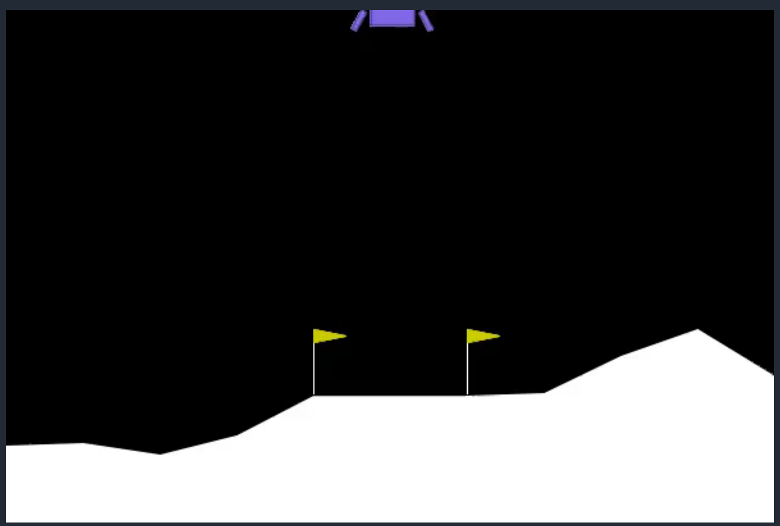


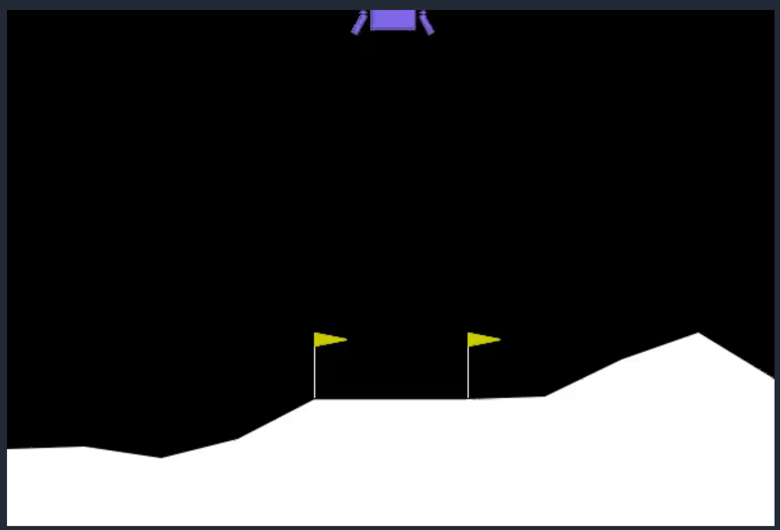


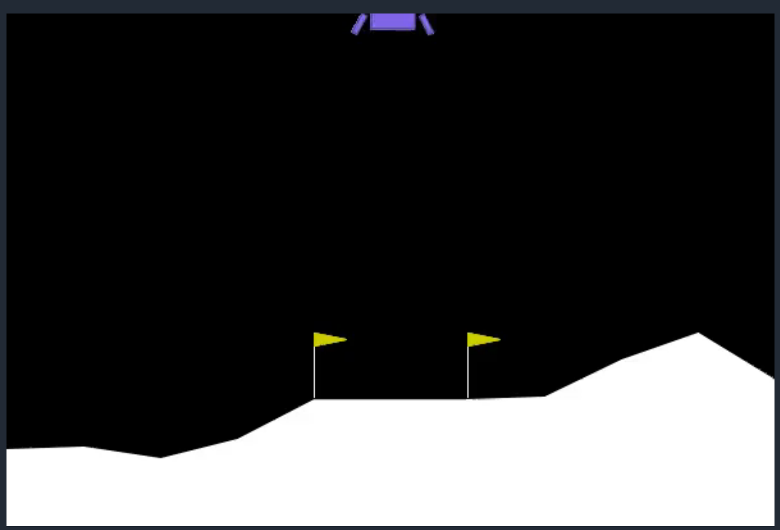
After latent variables become "saturated“, actions begin to become more "flicking“, implicitly indicating **uncertainty** about the environment.

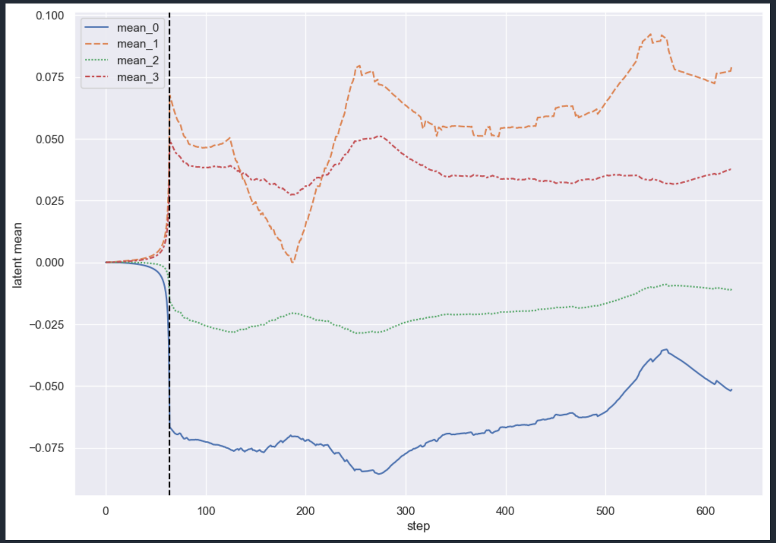
PEARL 2 with different wind trajectories

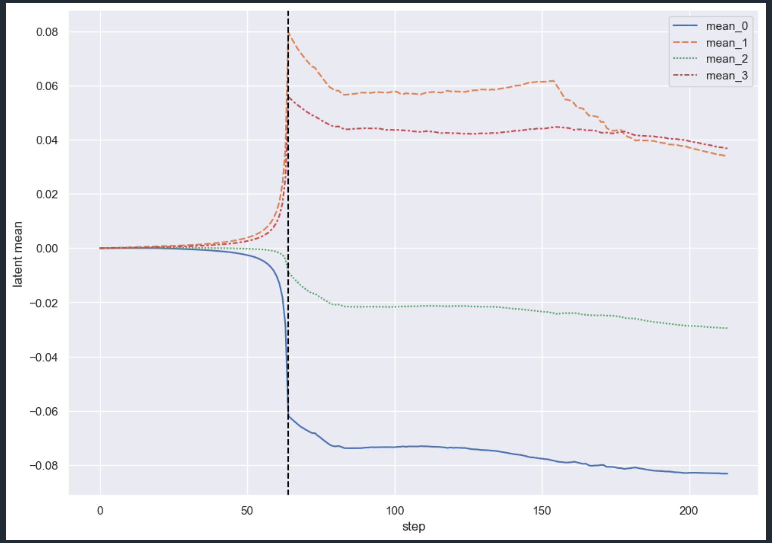
(but same environmental parameters)

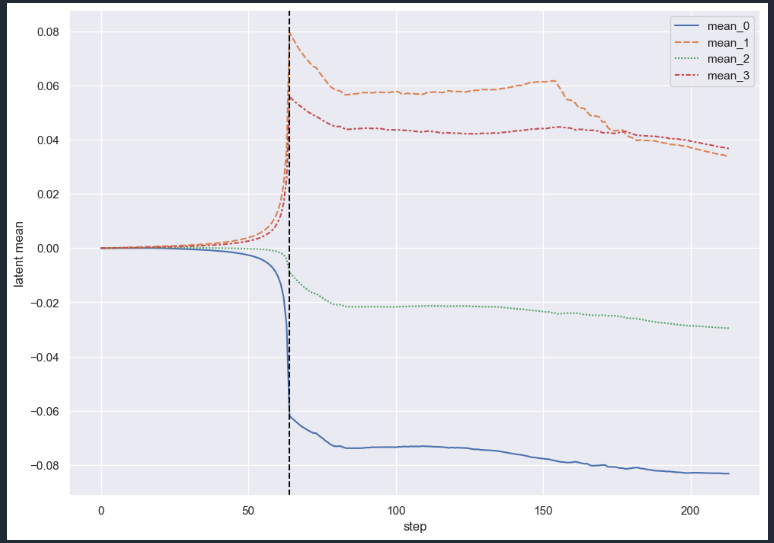












Engineering results to show off:

- The training process (>= 500 epochs) was completed at least 551 times
- =~ 300 hours of compute time (4 cores, 16 gb RAM)
- ~3000 lines of code, debugged with pain and tears, including config files with total 200 options

Possible continuations

- Smarter way to encode latent dimensions, possibly enforcing orthogonality between hidden features (as in modern GANs)
- Incorporate uncertainty into latent variables in more formal way
- Combined with our online latent variable generation approach, can be used to make models more safe
    - For example, high wind -> uncertainty about dynamics -> switch from learned approach to backoff classical control