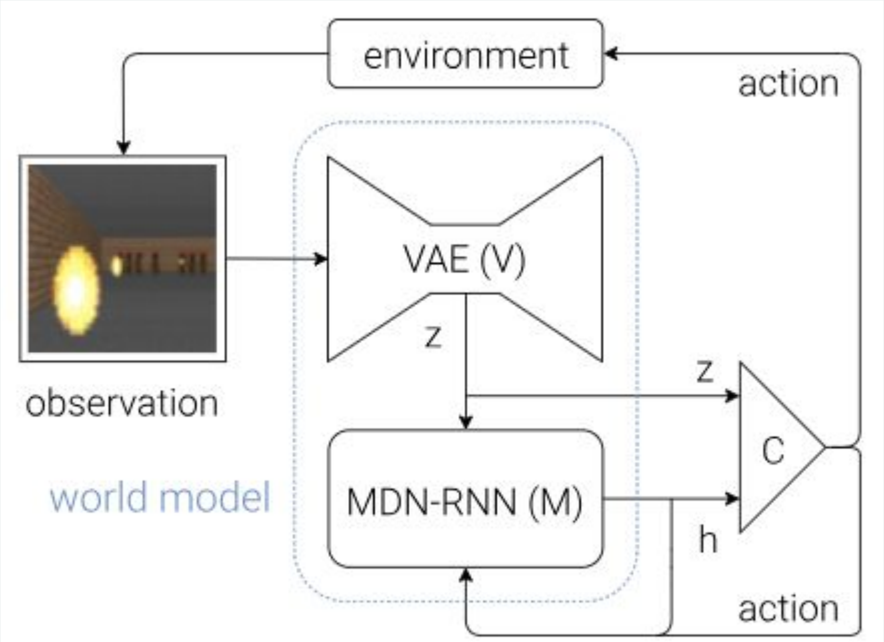
World Models



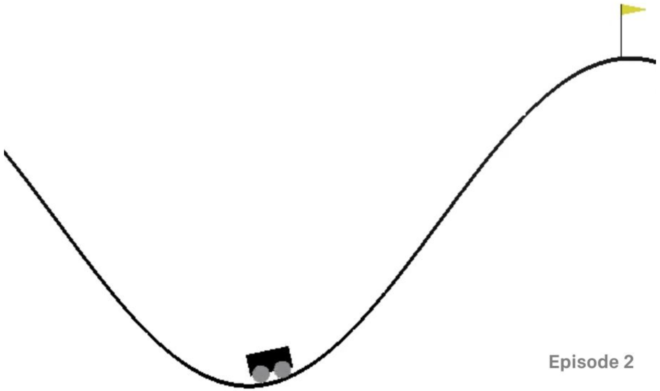
Authors: David Ha, Jürgen Schmidhuber

Andre Gou and Krishna Dusad

Project

- MVP
  - Reproduce this on MountainCar-v0 using the image as input
- Stretch
  - Experiment with learning strategies for Controller other than ES
  - Experiment with different Controllers
  - Reproduce results on racecar environment
- Super stretch
  - Try to see if we can learn a policy using the model and use it for warm start?
  - Try training V together with an M that predicts rewards, the VAE may learn to focus on task-relevant areas of the image, but the tradeoff here is that we may not be able to reuse the VAE effectively for new tasks without retraining
  - Try incorporating an external memory module if we want our agent to learn to explore more complicated worlds

Mountain Car

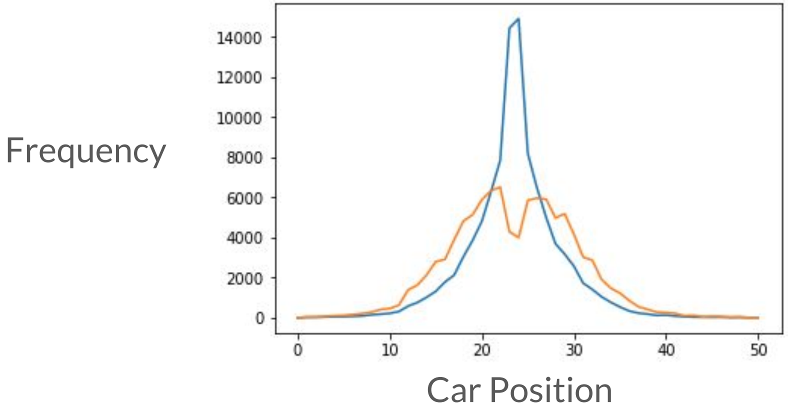


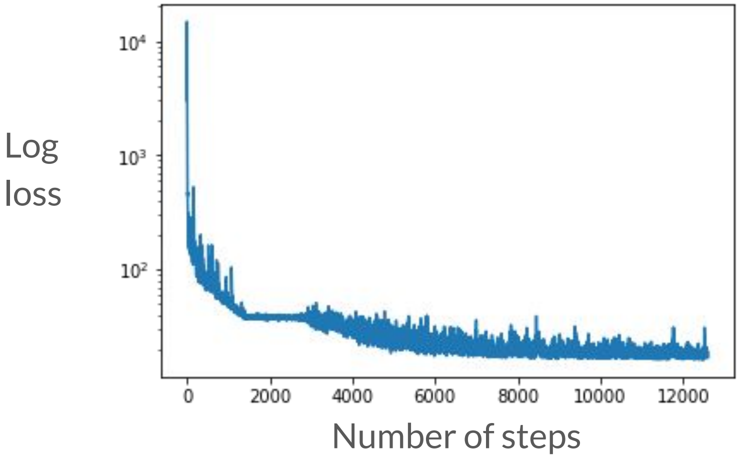
- Observations -- Car position, car velocity
- Observations (modified) -- image (64x64)
- Action -- L or R force by engine (discrete, 1D)
- (Continuous version has continuous action space)
- Reward -- 100 for reaching the target of the hill on the right hand side, minus the squared sum of actions from start to goal.

Results

VAE

- Dense network with 2 FCs for encoder, and 2 FCs for decoder.
- Hidden dimension size = 32
- Resampled data to increase coverage at the edges



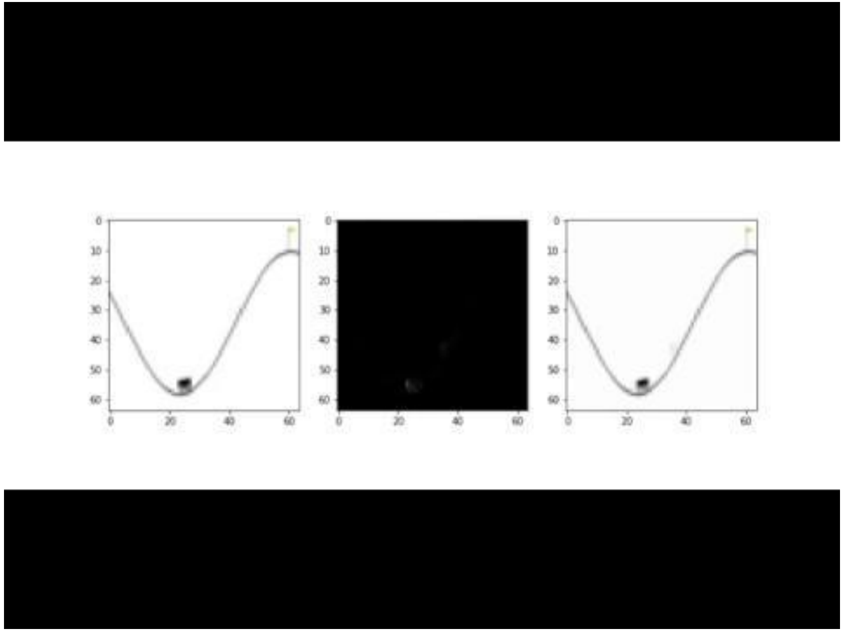


VAE

- Dense network with 2 FCs for encoder, and 2 FCs for decoder.
- Hidden dimension size = 32
- Resampled data to increase coverage at the edges

Video from sample rollout

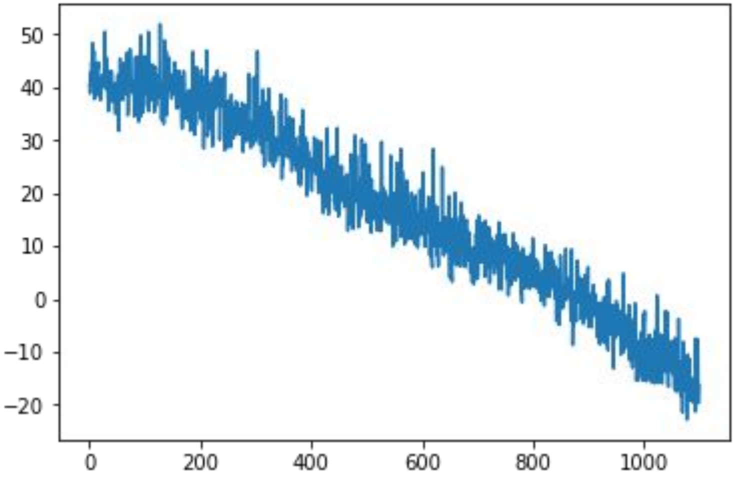
Input Frame Difference Reconstruction



M

1. LSTM with 2 layers and a hidden size of 256
2. Linear layer to predict 5 gaussians and log pis (mixing probabilities) from LSTM's output

Loss



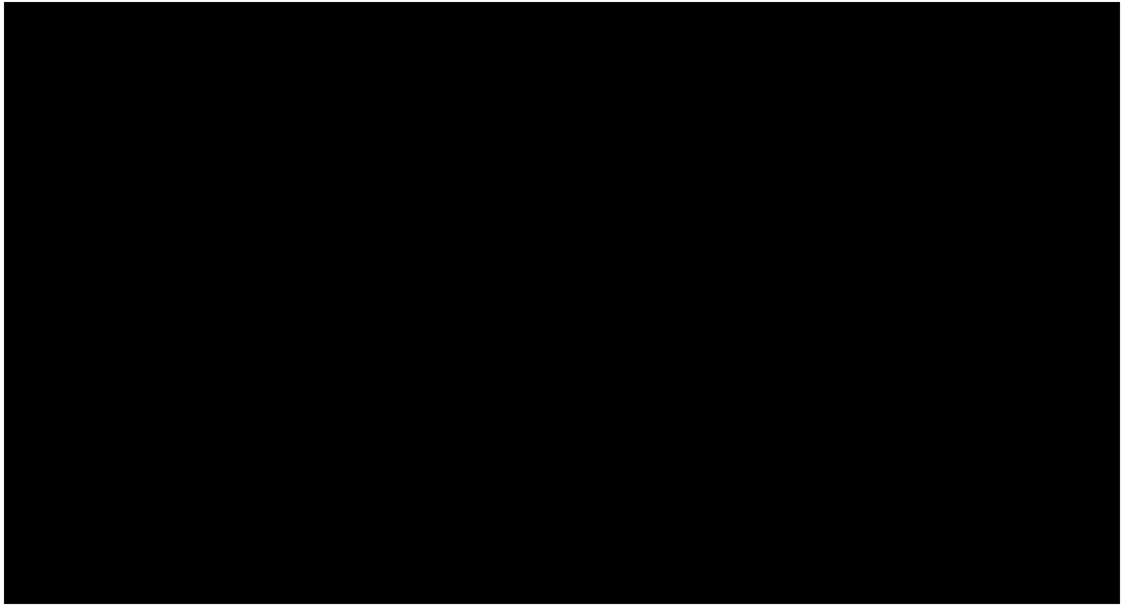
Number of steps

M

1. LSTM with 2 layers and a hidden size of 256
2. Linear layer to predict 5 gaussians and log pis (mixing probabilities) from LSTM's output

Video from sample rollout

Current Latent (Decoded)	Next Latent (Decoded)	Difference	Predicted Next Latent (Decoded)



M - Dream Sequence

Sequence generated by starting at a random state, using the vae to encode the state and then using the MDN-RNN to sample from a distribution of next possible latent spaces, for n steps and then decoding the samples using the vae

Problems

1. Unbalanced distribution of training data
   a. Solved by sampling 10x as much data then sampling more of the edges
2. Pretty bad reward function: constant negative reward
   a. Simple linear controller trained with CMA-ES using original states (velocity and position)
      i. Learned a policy of not moving to conserve fuel
   b. Changed reward -> +10 every time you hit a new max height
      i. Quickly learned an excellent policy
3. OpenAI Gym can’t output image without rendering environment
   a. SLOW!
   b. Training CMA-ES requires massive parallelization
      i. For 64 agents that each run 10 times for k generations
      ii. Convergence required k to be in the thousands...

Project -- Next Steps

- Get CMA-ES Controller working faster
  - Either find a way to easily running parallel agents
  - Modify MDNRNN to predict reward as well and train solely on the 'dreams'
- Better resampling method
- Reproduce results on racecar environment
- Experiment with learning strategies for Controller other than ES (try random sampling-shooting?)
- Super stretch
  - Try to see if we can learn a policy using the model and use it for warm start?
  - Try training V together with an M that predicts rewards, the VAE may learn to focus on task-relevant areas of the image, but the tradeoff here is that we may not be able to reuse the VAE effectively for new tasks without retraining