Training a Real Robot Arm

To Scoop Some Beans

Isaac Hirsch, Matthew LeBar, and Chad Schmerling

The Project Idea

Goal:

- Use a low-cost robot arm to dig a hole in a tub filled with uncooked beans
- Make use of a leader arm to train a digging policy using imitation learning
- Use a GoPro camera to provide the policy with feedback

Considered alternative goals: teaching arm to disconnect itself, teaching arm to twist fingers, etc.

Resources:

- Koch robot follower arm
- Koch robot leader arm
- GoPro Hero Black 7
- Plastic bin
- Uncooked pinto beans

Environment Setup

- Our environment was set up in the TTIC DuckieTown Robotics Lab (courtesy of Professor Matthew “Matt” Walter)
- We clamped the follower arm down next to the bin, which was velcroed to the table to keep it stable
- The leader arm was clamped down nearby for easy observation. We 3d printed a support to set it at a similar height
- We set up the GoPro right next to the bin to provide visual information for the model, and marked it with tape so we could keep its position consistent

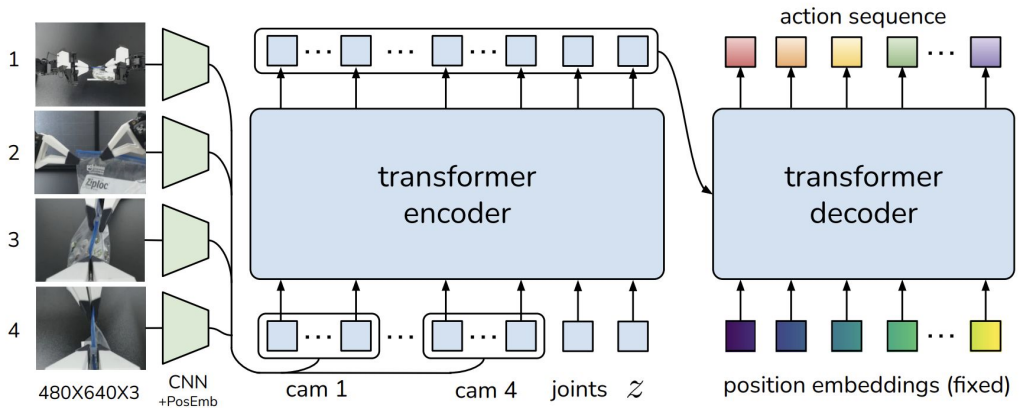
Data collection

- To train and evaluate our arm's performance, we used Hugging Face's LeRobot repo, which provided calibration, training, and testing scripts
- Our idea was to start with imitation learning to provide a foundation, then perform reinforcement learning from there
- However, just completing imitation learning was quite challenging

- We used the leader arm to teleoperate the follower arm for 29 episodes, each 20 seconds long
- The motor overheated and we struggled to find a way to combine multiple series of episodes
- The operator used the camera feed to try and scoop out as many beans from the center of the bin as possible

Model

- We used the *Action Chunking with Transformers* (ACT) algorithm built into the LeRobot repository for imitation learning
- ACT uses a transformer based VAE to learn a policy



ACT source: https://arxiv.org/pdf/2304.13705

Training

- AdamW
  - lr = 1e-5
  - batchsize = 16
  - l2 weight decay = 0.0001
- Gradient Clipping
- Checkpointed every 100 steps
- Evaluated IRL every 200 steps
- Trained for 13500 steps

Algorithm 1 ACT Training

Given: Demo dataset $\mathcal{D}$, chunk size k, weight β.
Let a_t, o_t represent action and observation at timestep t, $\bar{o}_t$ represent o_t without image observations.
Initialize encoder $q_\phi(z|a_{t:t+k}, \bar{o}_t)$
Initialize decoder $\pi_\theta(\hat{a}_{t:t+k}|o_t, z)$
for iteration $n = 1, 2, \ldots$ **do**
    Sample o_t, $a_{t:t+k}$ from $\mathcal{D}$
    Sample z from $q_\phi(z|a_{t:t+k}, \bar{o}_t)$
    Predict $\hat{a}_{t:t+k}$ from $\pi_\theta(\hat{a}_{t:t+k}|o_t, z)$
    $\mathcal{L}_{reconst} = MSE(\hat{a}_{t:t+k}, a_{t:t+k})$
    $\mathcal{L}_{reg} = D_{KL}(q_\phi(z|a_{t:t+k}, \bar{o}_t) \parallel \mathcal{N}(0, I))$
    Update θ, ϕ with ADAM and $\mathcal{L} = \mathcal{L}_{reconst} + \beta\mathcal{L}_{reg}$

ACT source: https://arxiv.org/pdf/2304.13705

Obstacles and Solutions

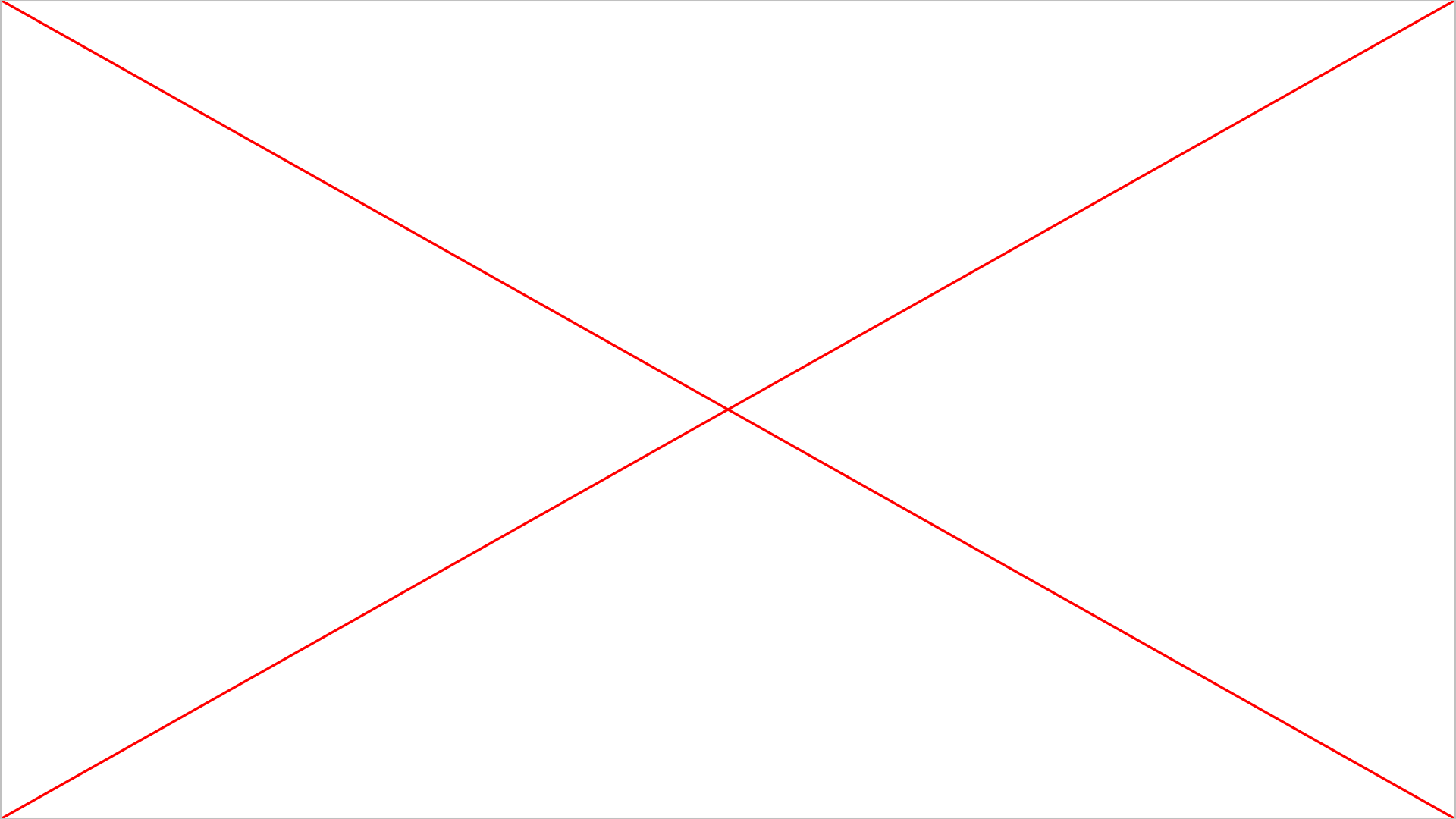
- ARM mechanical issues:
  - replacing a circuit board
  - needing to frequently power cycle robot arms
  - 3D Printing replacement parts
- GoPro required multiple specialized connectors and a backdoor exploit to work as a webcam
- LeRobot repo is extremely intricate, tailoring it to work for our particular project was often quite involved

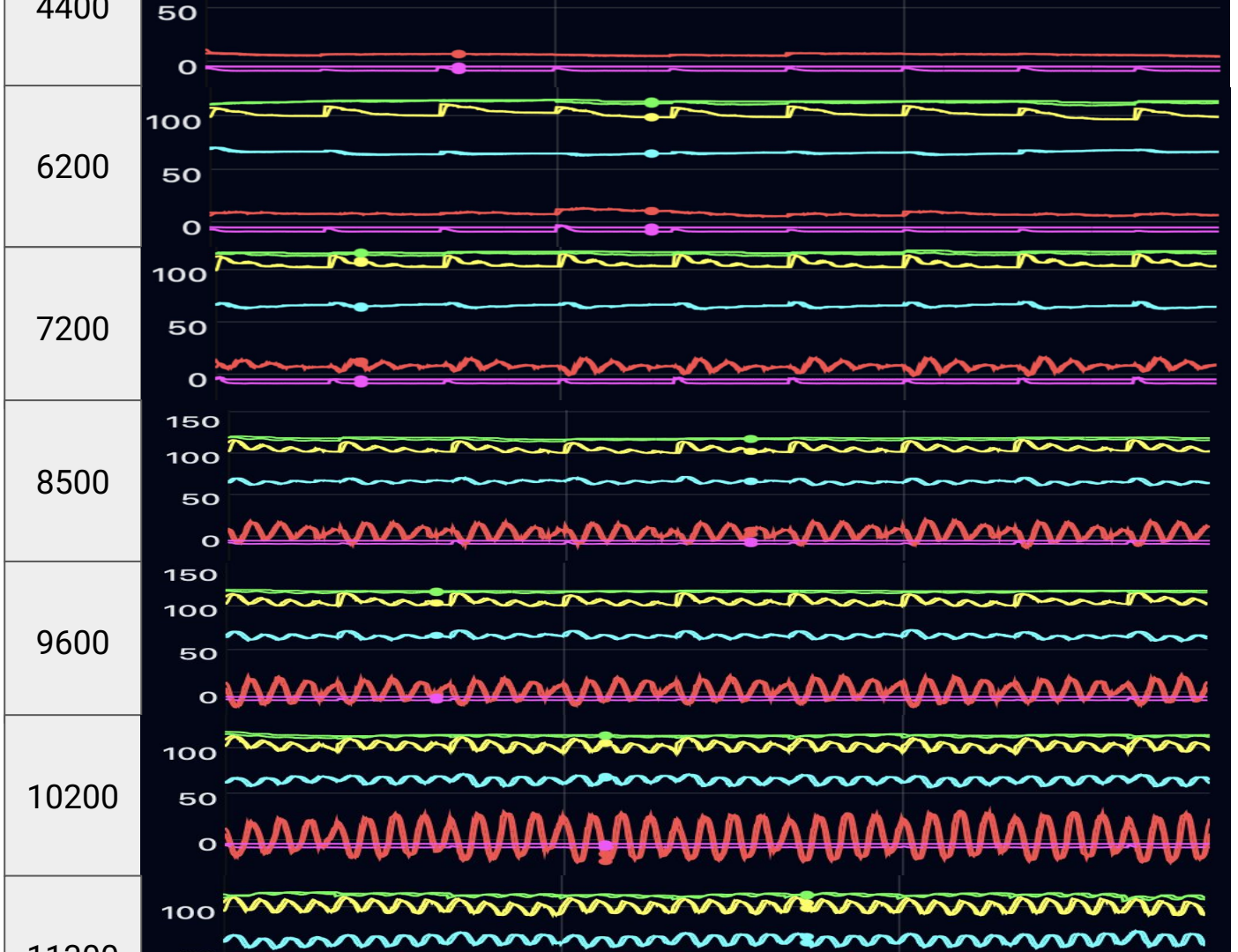
Evaluation 1

- We started off with a dumbed-down ACT model to expedite training.
  - There were *still* 11M parameters to train
- It was unable to learn

Evaluation 2

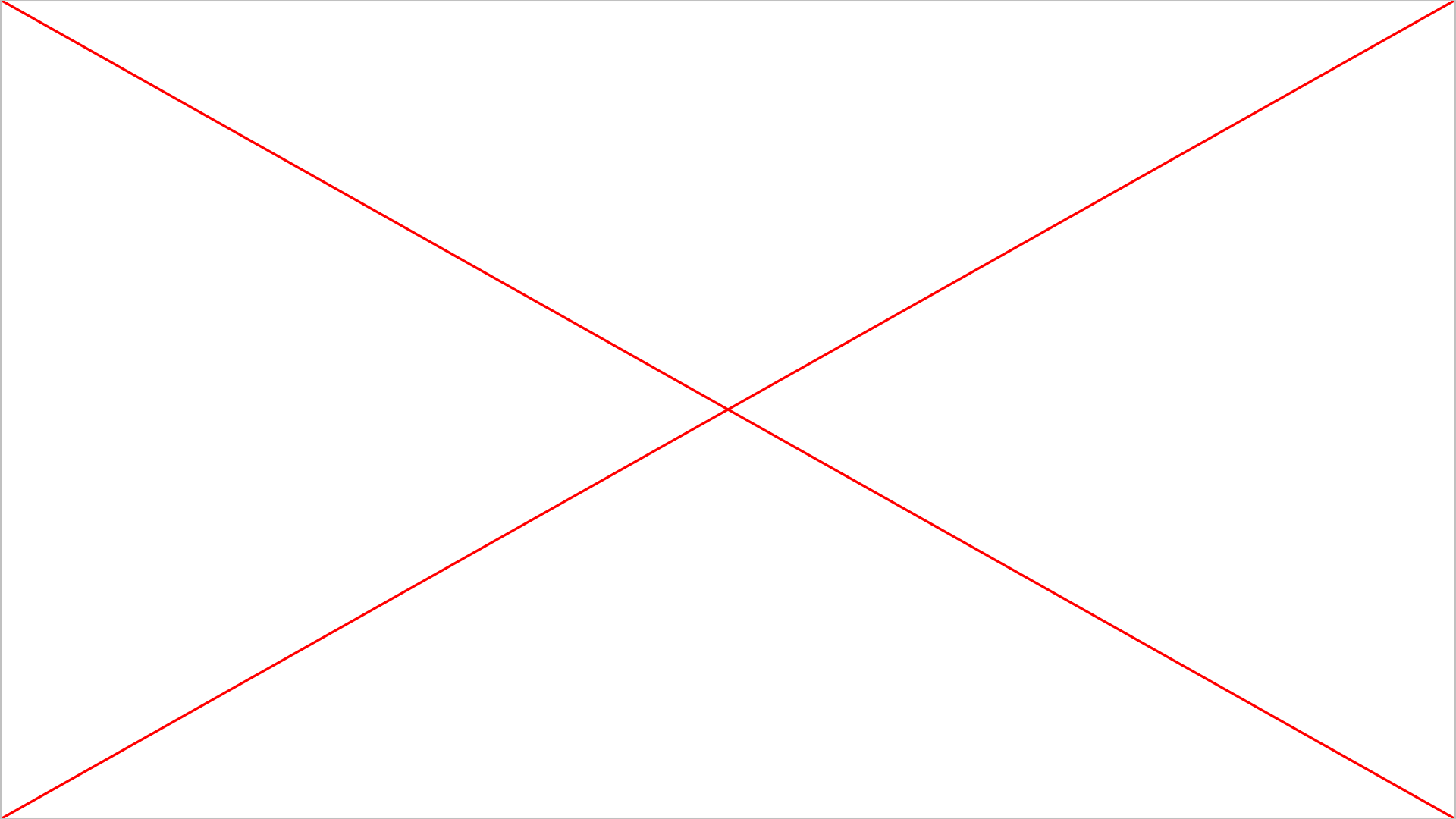
- We then returned to the author’s sets
  - There were now 45M parameters to train
- It learned much better



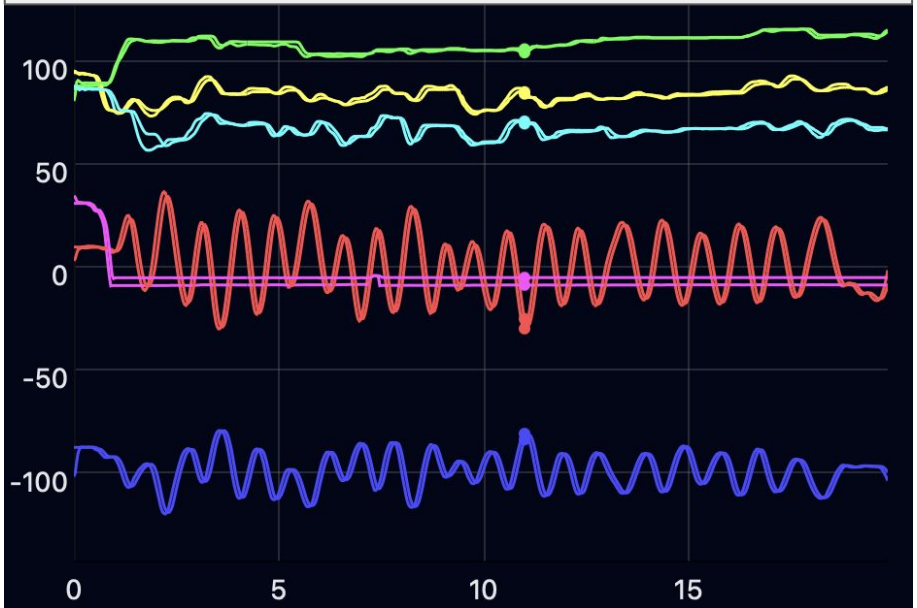

4400
50
0
6200
100
50
0
7200
100
50
0
8500
150
100
50
0
9600
150
100
50
0
10200
100
50
0
100



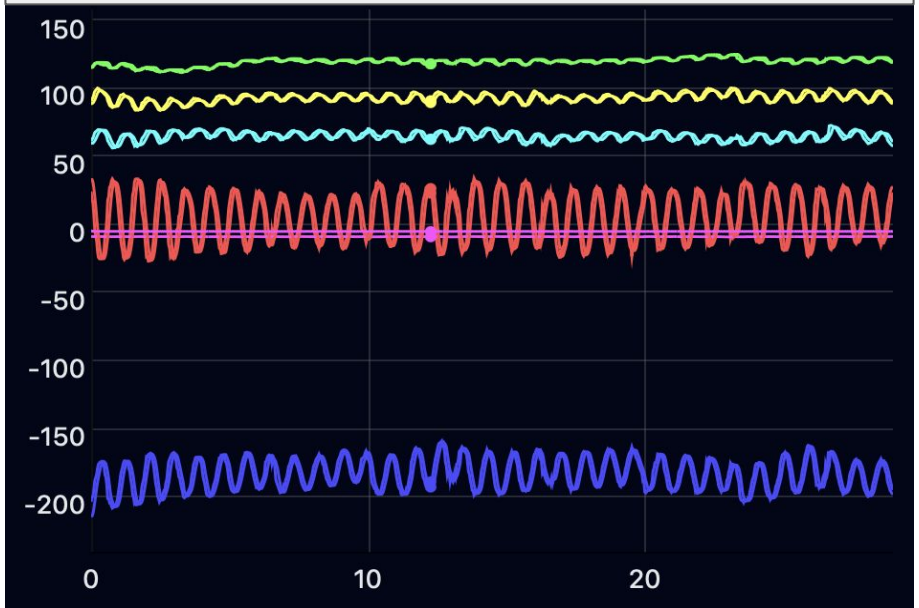




Goal



Ours (13,500)



Next steps

- Improvements for shuffling
  - Train for longer to get better results for imitation learning
  - Obtain more samples of imitation learning
  - Use reinforcement learning to bolster shuffling technique
- Future Directions
  - Use imitation learning as a base to start reinforcement learning