Task Planning for Unmanned Vehicle Delivery

Team 24
ZHU HAOYU, WANG GUANDING, ZENG CONG

Our project focuses on the path planning of unmanned vehicles (UVs) for parcels delivery across different communities within an urban environment. We employed the REINFORCE algorithm to train our network and explored two types of baselines to reduce the variance in policy gradient estimation.

1. Code explanation

Overall, our newly added code primarily includes two training functions and one evaluation function. The train_exp function adopts an Exponential Moving Average (EMA) baseline, while the train_greedy function employs a greedy rollout baseline. In the eval function, we test the trained model in newly generated random environments.

Detail:

- train :

  In each iteration, we sample from 128 different maps (BATCH_SIZE), obtaining the total rewards and action probabilities of 128 trajectories for training. Specifically, we use the cumulative return Rt of each trajectory rather than the action reward rt as the training signal. Every 10 iterations (per epoch), we evaluate the current policy on 100 fixed test environments using the greedy strategy. If the evaluation performance surpasses all previous results, we save the corresponding policy. We trained for a total of 2000 epochs, and experimental results indicate that the model converges after approximately 800 epochs.

  As is well known, the standard REINFORCE algorithm directly computes the gradient of the expected return, which often results in high variance. To address this, we employ two baseline methods and train using the advantage $R_t - b_t$. The first method is the Exponential Moving Average (EMA) baseline, defined as follows:

$$b_t = \begin{cases} \bar{R}_t, & t = 0 \\ \beta\, b_{t-1} + (1-\beta)\, \bar{R}_t, & t \geq 1 \end{cases}$$

  The second method uses the reward obtained by the current policy under the greedy rollout as the baseline. Experimental results show that the model trained with the second baseline performs slightly better, while the first provides greater training stability.

- evaluation:

  The eval function loads the best model trained with the greedy baseline method for testing and visualizes the trajectory of the robot. It also provides functionality to evaluate the model's performance across multiple environments using two strategies: greedy rollout and multi-sample best-of-K selection.

2. Existing Libraries

- pandas
  Used to save training data to Excel, including the average reward and loss for each iteration, as well as the test reward and test time for each epoch.
- mathplotlib
  Used to plot the reward, loss, test reward curves.

3. Reflections/Lessons Learned

- Why REINFORCE
  For the task planning problem, there is no definitive optimal solution at each step; thus, we adopted a stochastic policies based algorithm. Initially, we used the PPO algorithm, but we find that the reward plateaued at around 100, far below our expected performance of approximately 600. We hypothesize that, for a task planning problem emphasizing long-term rewards (with a discount factor of 1) and a clearly defined reward function—maximizing the number of high-reward nodes visited within a limited distance—the PPO approach, which treats each action as an independent learning sample, may lead to unstable or wrong gradient updates. Consequently, inspired by prior studies, we switched to the REINFORCE algorithm, which treats each full trajectory as a single learning signal—better aligned with our task objective.

- Writing code suit for GPU
  In our initial implementation, training was surprisingly slow, about 5 minutes per epoch, same speed as to CPU. We later discovered that we were not using the GPU's parallel computing capabilities effectively. Instead of running GPU computation after each individual map sample, we should process all sampled data(128 map) in a single batch to fully utilize GPU parallelism.

4. Result

Figure 1 shows the reward curves of the two baselines during training. Figure 2 illustrates the robot's visiting routes across different points, demonstrating that the robot has learned how to plan its path to achieve higher rewards.

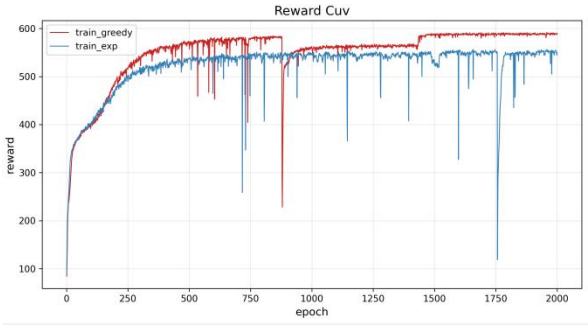

Figure. 1.

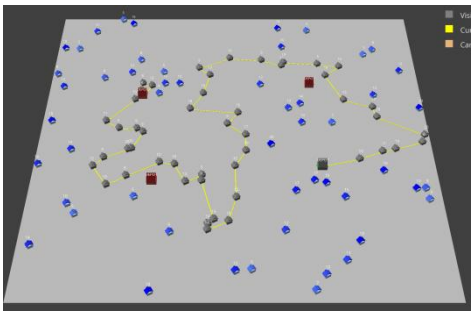

Figure. 2.