Learn to Calibration: RL model for Camera Calibration

April 14, 2020

1 Defination of POMDP

MDP time step: $t \in 1 : T$, frame of each observation in each MDP time step: $k \in 1 : K$.
Action: the parameter A_t that determines trajectory executed at each MDP time step $x_t^{1:K}$, which further determined the dataset acquired at each step D_t.
States: the total dataset gathered so far: $S_t = \bigcup_{i=0}^{t-1} D_i$.
Observation:
the optimal intrinsics estimated with the total observation gathered so far: $Y_t = \theta_t^*$. We can also add items such as observability and coverage that are meaningful to guide the actions
Alternative: Only observability and coverage, the params are only computed at the end of the sequence.
Transition model: deterministic, as A_t can determine D_t, which can be represented by a map h: $D_t = h(A_t)$, so:

$$S_{t+1} = \bigcup_{i=0}^{t} D_i = (\bigcup_{i=0}^{t-1} D_i) \cup D_t = S_t \cup h(A_t)$$

Observation model: the estimation model, with maximum likelihood:

$$Y_t = \theta_t^* = argmin_\theta Loss(\theta, \bigcup_{i=0}^{t} D_i) = argmax_\theta L(\theta, S_t)$$

The whole POMDP model is illustrated in figure 1.
POMDP rewards: the improvement of coverage, observability, the decrease of distance from the true value, reprojection error, the length of the trajectory.

$$R_t = a_1 \delta C_t + a_2 \delta O_t - a_3 \delta d_t - a_4 \delta E_t - a_5 l_t$$

2 RL for POMDP

RL states: belief state $b(S_t)$, which can be uniquely determined by the previous actions and observations $A_{1:t}, Y_{1:t}$. here we use RNN to encode the relationship between

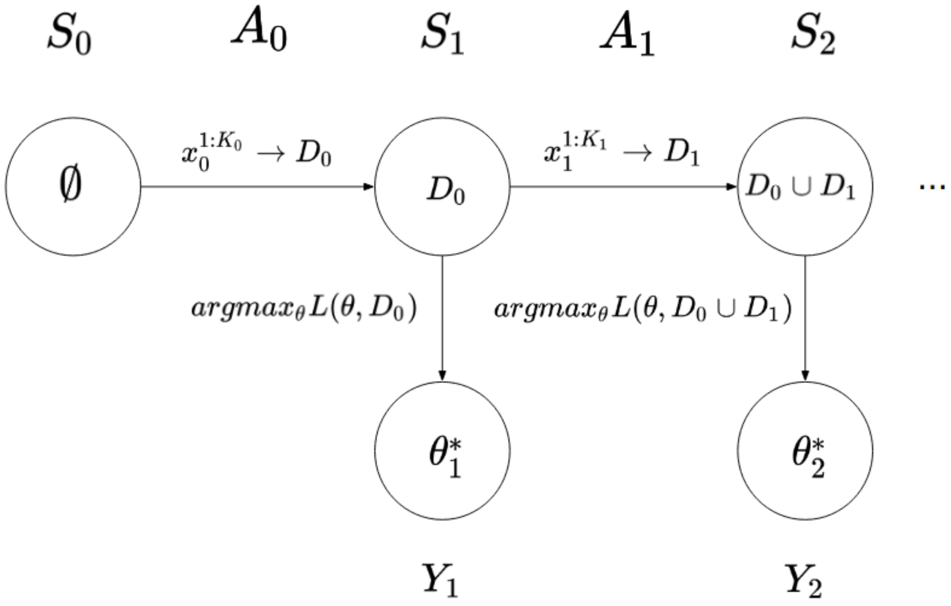
S_0
A_0
S_1
A_1
S_2
$\emptyset$
$x_0^{1:K_0} \rightarrow D_0$
D_0
$x_1^{1:K_1} \rightarrow D_1$
$D_0 \cup D_1$
$\cdots$
$argmax_\theta L(\theta, D_0)$
$argmax_\theta L(\theta, D_0 \cup D_1)$
θ_1^*
θ_2^*
Y_1
Y_2

Figure 1: Summary of POMDP model of the problem

the current belief state and history of observations and actions. Which is shared among value network and policy network
RL policy and value: The policy and value network has a shared RNN part, which encode the belief state of current time step. Then two separated fully connected layers with shared input from RNN encodes output action and value function of the state respectively. If we want to output the Q-function, then the current action is also input to the value layer. The entire architecture is shown in figure 2. Here a_t and y_t are action and observation of the POMDP model, which are the trajectory parameters and optimal intrinsic estimated respectively. The hidden state h_t is the unknown encoding of the state S_t. $Q(h_t, a_t)$ is the Q-function and μ is the deterministic policy.

RL agent: In this problem, we choose the policy to be deterministic, because the dynamics of the problem are relatively stable (even given random action can have high possibility to reach the goal state) and deterministic policies can achieve high performance faster than stochastic policies in such problem. Therefore we plan to use DDPG and TD3 to learn good deterministic policies. As for DDPG, the policy are updated by gradient ascent:

$$\delta\alpha = \nabla_\alpha \mu_\alpha(y_{1:t}, a_{1:t-1}) \nabla_a Q_\beta(y_{1:t}, a_{1:t}, a)|_{a=\mu_\alpha(y_{1:t}, a_{1:t-1})}$$

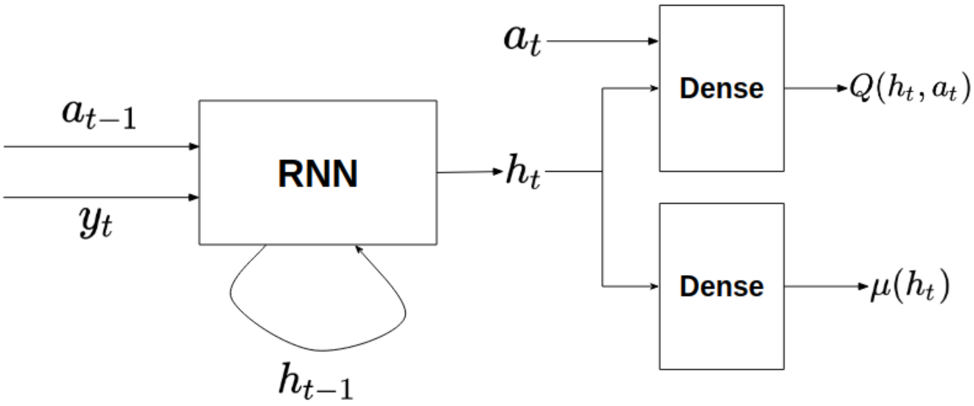

Figure 2: RL RNN framework for POMDP

The value network parameters are updated by minimize mean square TD error:

$$\delta\beta = \nabla_\beta[r_t + \gamma Q_{\beta'}(y_{1:t+1}, a_{1:t}, \mu_{\alpha'}(y_{1:t+1}, a_{1:t})) - Q_\beta(y_{1:t}, a_{1:t-1}, a_t)]^2$$

Here α' and β' are exponential moving average of α and β respectively, which serve as the target policy and value network parameters. TD3 improves DDPG in several aspects so that it can provide more stability and better performance, which will be tested later in our problem.