Excessive Disturbance Rejection Control of Autonomous Underwater Vehicle using Reinforcement Learning

Tianming Wang, Wenjie Lu, Dikai Liu
Centre for Autonomous Systems, University of Technology Sydney, Australia
tianming.wang@student.uts.edu.au, {wenjie.lu; dikai.liu}@uts.edu.au

Abstract

Current and wave disturbance can severely impact the operation stability of autonomous underwater vehicles (AUV), especially in shallow and turbulent water. Also, this kind of disturbance usually cannot be directly measured or is too expensive to measure. Traditional disturbance rejection control approaches are proven to be not sufficiently powerful to deal with such underwater disturbance. To address this problem, we have proposed a novel reinforcement learning (RL) method, which takes into consideration a certain period of states and actions history as observation, and chooses control action based on these inputs. Furthermore, model-based and model-free reinforcement learning approaches are combined in our framework, in order to achieve both sample efficiency and optimal performance under external disturbance. We empirically demonstrated on pose stabilization task using simulated AUV model that our model-based approach can realize good control performance when there is no disturbance, and that our hybrid algorithm can accelerate model-free learning and avoid damaging and risky exploratory actions at the inital stage of training.

1 Introduction

Disturbances and uncertainties widely exist in all industrial systems and bring adverse effects on performance and even stability of control systems [Xie and Guo, 2000; Gao, 2014; Li *et al.*, 2014]. Also, the external disturbance usually cannot be directly measured or is too expensive to measure. Not surprisingly, disturbance and uncertainty rejection is a key objective in control system design. In underwater environments, the effect of disturbance on the robot systems become more significant. Most inspection tasks for bridges and off-shore infrastructures require the robot operated in shallow water environments. In such applications, the underwater vehicle is often subject to large external disturbances caused by water flow and current. Woolfrey et al. [2016] considered a situation that an underwater vehicle-manipulator system operating in shallow and turbulent water where, the wave disturbances affect the efficacy of control and the accuracy of the manipulator end-effector. Thus, this paper mainly focuses on excessive disturbance rejection control for underwater robot.

In the early development of disturbance rejection control, feedback control strategy is used to suppress the unknown disturbance. Examples of feedback controllers include robust control [Skogestad and Postlethwaite, 2007], adaptive control [Åström and Wittenmark, 2013], optimal control [Bertsekas *et al.*, 1995], sliding mode control (SMC) [Edwards and Spurgeon, 1998], H-infinity control [Doyle *et al.*, 1989], etc. It should be pointed out that these advanced control schemes, rejects disturbances merely through the action of feedback regulation part and does not deal with the disturbances directly by controller design. It has been assumed that the system deals with a bounded disturbance which should be small enough [Ghafarirad *et al.*, 2014], when meeting strong disturbances, these methods may lead to some limitations.

In order to improve the disturbance rejection performance of the robot system, a feedforward compensation part for the disturbances is introduced to the controller besides a conventional feedback part [Yang *et al.*, 2010]. However, usually, it is hard or even impossible to measure the disturbances of underwater current. A feasible solution is to develop disturbance estimation technique [Zeinali and Notash, 2010; Ghafarirad *et al.*, 2012; Yang *et al.*, 2011a; Chen and Guo, 2004]. The basic idea is to estimate the disturbance (or the influence of the disturbance) from measurable variables, and then, a control action can be taken, based on the disturbance estimate, to compensate for the influence of the disturbance [Chen *et al.*, 2016]. In this setting, the disturbances do not only refer to that from the external environment of a control system but also uncertainties of the controlled system including unmodeled dynamics and parameter perturbations [Gao, 2014; Li *et al.*, 2014; Guo and Cao, 2014]. Various disturbance estimation and attenuation methods have been proposed and practiced by many researchers and engineers, such as disturbance observer (DOB)

[Ohishi *et al.*, 1987; Chen *et al.*, 2000; Umeno *et al.*, 1993; Umeno and Hori, 1991], unknown input observer (UIO) in disturbance accommodation control (DAC) [Johnson, 1968; 1971], and extended state observer (ESO) [Han, 1995; Gao *et al.*, 2001]. The disturbance observer-based control (DOBC) obtains promising robustness and disturbance rejection performance without sacrificing the nominal control performance. However, it fails to consider the constraints of the states and the controls [Gao and Cai, 2016]. And how to deal with the possible constraints in the design of the control system is an open problem.

Model predictive control (MPC) [Camacho and Alba, 2013] is well known for its constraint handling capacity. The method can achieve approximately optimal control performance even under practical constraints [Gao and Cai, 2016]. This is because MPC considers a period of time instead of only the current moment. It employs an explicit prediction model of the plant to optimize future plant behaviour [Maeder and Morari, 2010]. At each time step, an open loop optimal control sequence is obtained by means of solving an optimization problem. The first element of this sequence is applied to the plant, the rest is discarded. This optimization procedure is repeated at every time step. However, in order to realize optimal control performance under disturbance, MPC requires an accurate model of the robot system with disturbance, which is quite difficult to obtain. Thus, researchers have developed a compound control scheme consisting of a feedforward compensation part based on DOB and a feedback regulation part based on MPC (DOB-MPC) [Maeder and Morari, 2010; Yang *et al.*, 2010; 2011b; Liu *et al.*, 2012; Yang *et al.*, 2014; Dirscherl *et al.*, 2015; Gao and Cai, 2016] to realize better performance than normal DOBC or MPC. However, the MPC technique generally requires the solution of an optimization problem at every sampling instant [Liu *et al.*, 2012]. This poses an obstacle on the real-time implementation due to the heavy computational burden.

It seems that traditional control approaches are not sufficiently powerful to deal with such underwater disturbance. While reinforcement learning has shown its advantages in many control problems. Compared with the optimized control sequence provided by MPC at each time step, RL can give a single control policy after training without subsequent changes, and this policy can choose the action to take based on the state observed. Also, model-free RL does not require any model knowledge in advance to train this policy. Thus, RL may be a potentially better solution for the underwater disturbance rejection control. When using RL to deal with external disturbance, if the algorithm only observes the current robot state, and chooses action based it, this problem will not be a Markov Decision Process (MDP). The reason is that the state transition does not only depend on the current state and action, but also related to the disturbance value. Thus we cannot define this problem as a one-step MDP, a framework of multi-step MDP will be necessary.

Deep reinforcement learning algorithms based on Q-learning [Mnih *et al.*, 2015; Oh *et al.*, 2016; Gu *et al.*, 2016b], policy gradients [Schulman *et al.*, 2015a; Gu *et al.*, 2016a], and actor-critic methods [Lillicrap *et al.*, 2015; Mnih *et al.*, 2016; Schulman *et al.*, 2015b] have been shown to learn complex skills in high-dimensional state and action spaces, including robotic locomotion, autonomous driving, playing video game, and navigation. However, the high sample complexity of purely model-free algorithms has made them difficult to deploy in real world, where sample collection is limited by the constraints of real-time operation. Model-based reinforcement learning algorithms are generally known to outperform model-free methods in terms of sample efficiency [Deisenroth *et al.*, 2013], various model-based approaches have been proposed [Deisenroth and Rasmussen, 2011; Kuvayev and Sutton, 1996; Forbes and Andre, 2002; Hester and Stone, 2017; Jong and Stone, 2007; Sutton, 1991], and in practice have been applied successfully to control both simulated and real-world robotic systems, such as inverted pendulums [Deisenroth and Rasmussen, 2011], manipulators [Brauer, 2012], and legged robots [Schmidt and Lipson, 2009]. Derner et al. [2018] proposed to use symbolic regression to construct a symbolic model of robot, then used value iteration to optimize a policy based on the symbolic model found. An essential problem of model-based RL is the difficulty to scale to high-dimensional state/action spaces, Chatzilygeroudis and Mouret /shortcitechatzilygeroudis2017using tried to address this problem through using prior information about the system that is modeled to learn the residual model.

Although such model-based algorithms are significantly more sample efficient and more flexible than task-specific policies learned with model-free methods, their asymptotic performance is usually worse than model-free learners due to model bias. Model-free algorithms are not limited by the accuracy of the model, and therefore can achieve better final performance, though at the expense of much higher sample complexity [Deisenroth *et al.*, 2013; Kober *et al.*, 2013]. To address this issue, researchers tried to combine model-based methods and model-free learners, so that the algorithms can quickly achieve moderately proficient behavior, and then slowly achieve near-optimal behavior. Kumar et al. [2018] and Koryakovskiy et al. [2018] both proposed to use model-free reinforcement learning to learn a compensatory control signal on top of a model-based controller. The model-based controller can speed up learning and avoid damaging and risky exploratory actions, and model-free learner can enhance the control performance by compensating the model-plant mismatch. However, the model-based controllers usually need to do some real-time calculations, which is much slower than the forward propagation of a neural network policy. Nagabandi et al. [2018] also used model-based controller, but they used supervised learning to train a policy to mimic the model-based controller, and then used this imita-

tion policy as an initialization for the model-free learner.

This paper proposed a novel reinforcement learning algorithm for current disturbance rejection control of autonomous underwater vehicles. In order to deal with the excessive current disturbance in shallow and turbulent water, the trained policy will take a certain period of history of states and actions as current observation, and choose action based this history. In addition, model-based approach and model-free learner are combined to improve sample efficiency and ensure optimal performance under disturbance. Given a rough model of the robot, the iterative Linear Quadratic Regulator (iLQR) is used to generate trajectories without disturbance, then supervised learning is used to train an imitation policy to mimic these trajectories, using current robot state as observation. Afterwards, we use Deep Deterministic Policy Gradient (DDPG) algorithm to train a model-free policy along with the model-based policy, using history as observation. The final control signal is a combination of these two policies.

2 Preliminaries

2.1 Trajectory Optimization

Trajectory optimization is the process of finding a state-control sequence which locally minimizes a given cost function [Tassa *et al.*, 2014]. Differential Dynamic Programming (DDP) is a second-order shooting method [Mayne, 1966] which under mild assumptions admits quadratic convergence for any system with smooth dynamics [Jacobson and Mayne, 1970]. Classic DDP requires second-order derivatives of the dynamics, which are usually the most expensive part of the computation. If only the first-order terms are kept, one obtains a Gauss-Newton approximation known as iterative Linear Quadratic Regulator (iLQR) [Li and Todorov, 2004; Todorov and Li, 2005], which is similar to Riccati iterations, but accounts for the regularization and line-search required to handle the nonlinearity.

We consider a system with discrete-time dynamics, but a similar derivation holds for the continuous case [Mayne, 1966]. The dynamics is modeled by the generic function f

$$s_{t+1} = f(s_t, a_t), \tag{1}$$

which describes the evolution from time t to $t+1$ of the state $s \in \mathbb{R}^n$, given the action $a \in \mathbb{R}^m$. A trajectory $\{S,A\}$ is a sequence of states $S = \{s_0, s_1, \cdots, s_N\}$, and corresponding controls $A = \{a_0, a_1, \cdots, a_{N-1}\}$ satisfying (1).

The total cost denoted by J is a sum of running costs l and final cost l_f, incurred when starting from initial state s_0 and applying the control sequence A until the horizon N is reached:

$$J(s_0,A) = \sum_{t=0}^{N-1} l(s_t,a_t) + l_f(s_N). \tag{2}$$

Indirect methods, like iLQR, represent the trajectory implicitly using only the controls A. The states S are recovered by integration of (1) from the initial state s_0. The solution of the optimal control problem is the minimizing control sequence

$$A^\star = \arg\min_A J(s_0,A). \tag{3}$$

2.2 Reinforcement Learning

Reinforcement learning is a trial-and-error method which does not require an explicitly given model, and can naturally adapt to uncertainties in the real system [Sutton and Barto, 1998]. In reinforcement learning, the goal is to learn a policy that chooses actions $a_t \in A$ at each time step t in response to the current state $s_t \in S$, such that the total expected sum of discounted rewards is maximized over all time. At each time step, the system transitions from s_t to s_{t+1} in response to the chosen action a_t and the transition dynamics function $f : S \times A \to S$, collecting a reward r_t according to the reward function $r(s_t,a_t)$. The discounted sum of future rewards is then defined as $\sum_{t'=t}^{\infty} \gamma^{t'-t} r_{t'} | s_t, a_t$, where $\gamma \in [0,1]$ is a discount factor that prioritizes near-term rewards over distant rewards [Nagabandi *et al.*, 2018].

3 Problem Formulation

As shown in Figure 1, our underwater robot consists of a 6-degree of freedom (DOF) underwater vehicle and a 3-DOF manipulator, we only consider the 6-DOF body in this work. Due to the hardware design, the roll and pitch of the underwater vehicle are hardly affected by the external current disturbance. Thus, in order to simplify this problem, we only keep the 3-DOF for vehicle's position and the 1-DOF for yaw angle. The state space of the robot s consists of the vehicle position and yaw angle q, as well as the corresponding velocities $\dot{q}$. The action space a includes the control forces and torques of the vehicle τ_c. Also, the control limits need to be taken in consideration.

The dynamics function is given by (1), we detail it for our

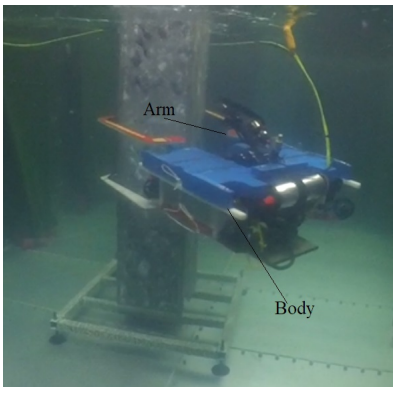

Figure 1: Real Autonomous Underwater Vehicle

robot system of the form:

$$M\ddot{q} + C + F_e = \tau_c + \tau_d, \tag{4}$$

$$\begin{bmatrix} q \\ \dot{q} \end{bmatrix} = \begin{bmatrix} q \\ \dot{q} \end{bmatrix} + \begin{bmatrix} \dot{q} \\ \ddot{q} \end{bmatrix} dt \tag{5}$$

where M is the inertia matrix (including added mass), C is the vector of Coriolis and centripetal terms (including added mass), F_e is vector of external forces, including gravity, buoyancy, fluid acceleration and drag force, τ_d is the disturbances forces, $\ddot{q}$ represent accelerations of the vehicle.

The underwater disturbance τ_d mainly comes from current and wave, which are time-varying signals. However, different from random disturbance, current and wave disturbance has a strong time-series pattern, which could be either periodic or nonperiodic. It means this pattern can be learned for future disturbance prediction and thus for better control. In our case, we assume that the disturbance is close to the robot control limits, but is constrained within a reasonable range, ensuring the controller is able to converge. And in this paper, we only consider the periodic disturbance.

4 Model-Free Reinforcement Learning

We first try using purely model-free reinforcement learning algorithm to solve this problem. In order to deal with the unknown periodic external disturbance, a certain period of history states and actions $h_t = \{s_{t-H}, a_{t-H}, \cdots, s_{t-1}, a_{t-1}, s_t\}$ need to be taken into consideration as current observation when choosing action, H represents the length of the history. Before using this observation history to train a policy, we first need to verify the existence of this transition model $s_{t+1} = f_h(h_t, a_t)$.

4.1 Dynamic Model Learning

We parameterize the learned dynamics function $\hat{f}_{h\theta}(h_t, a_t)$ as a neural network, where the parameter θ represents the weights of the network. A straightforward parameterization for $\hat{f}_{h\theta}(h_t, a_t)$ would take the current history h_t and action a_t as input, and output the predicted next state $\hat{s}_{t+1}$. However, this function will be difficult to learn when the current states s_t and s_{t+1} are too similar and the action has little effect on the output; this difficulty becomes more pronounced as the time between states Δt becomes smaller and the state differences do not indicate the underlying dynamics well [Nagabandi *et al.*, 2018]. We overcome this issue by instead learning a dynamics function that predicts the change in state s_t over one time step duration Δt. Thus, the predicted next state is as follows: $\hat{s}_{t+1} = s_t + \hat{f}_{h\theta}(h_t, a_t)$.

Collecting Training Data: We collect training data by sampling starting configurations $s_0 \sim p(s_0)$, setting a certain pattern of disturbance (amplitude, period, phase), executing random actions at each time step, and recording the resulting trajectories $\tau = (s_0, a_0, \cdots, s_{T1}, a_{T1}, s_T)$ of length T.

Data Preprocessing: We slice the trajectories $\{\tau\}$ into training data inputs (h_t, a_t) and corresponding output labels $s_{t+1} - s_t$. The useful training data should begin at $t = T - H$, since the agent starts to observe the full length of history at this time. We then subtract the mean of the data and divide by the standard deviation of the data to ensure the loss function weights the different parts of the state (e.g., positions and velocities) equally. The training data is then stored in the dataset D.

Training Dynamic Model: We train the dynamic model $\hat{f}_{h\theta}(h_t, a_t)$ by minimizing the error

$$\varepsilon(\theta) = \frac{1}{|D|} \sum_{(s_t, h_t, a_t, s_{t+1}) \in D} \frac{1}{2} ||(s_{t+1} - s_t) - \hat{f}_{h\theta}(h_t, a_t)||, \tag{6}$$

using stochastic gradient descent. While training on the training dataset D, we also calculate the loss in (6) on a evaluation dataset D_{eval}, composed of trajectories not stored in the training dataset.

Through several experiments using inverted pendulum model, we found that the error between the learned model and actual model is always less than 2%, showing that the history model exists to some extent.

4.2 Model-Free Reinforcement Learning

The existence of the dynamic model under disturbance ensures that the model-free reinforcement learning algorithm is able to learn a satisfactory policy $\pi_\phi(a|h)$ for disturbance rejection control. Specifically, we use Deep Deterministic Policy Gradient (DDPG) [Lillicrap *et al.*, 2015], which is an actor-critic, model-free algorithm based on the deterministic policy gradient that robustly solves challenging problems across a variety of domains with continuous action spaces, though our method could also be combined with other model-free reinforcement learning algorithms.

In our implementation, DDPG is used to train a neural network policy that chooses action based on a certain period of history of states and actions. During training, we set different pattern of disturbance in each episode, in order to let the algorithm learn the capability of disturbance prediction and rejection, no matter what kind of disturbance pattern occurs. Furthermore, the history space has a certain length H, in each episode, when the history space is not full, the algorithm need to randomly choose action, and add current state and action into the history space. When the history space is full, the algorithm will choose action based on the current deterministic policy, then update the history space (delete the oldest state-action pair and add the latest one), and the experience (history, action, reward, next history) for each step would be save to a replay memory. The training begins when the replay memory is full, a batch of experience would be grabbed from the replay memory and used to train the actor and critic network at each step. Thus, DDPG is also an off-policy reinforcement learning algorithm.

5 Model-Based Reinforcement Learning

We now present our model-based reinforcement learning algorithm. The prerequisite for this model-based approach is the availability of the dynamic model f. We assume the robot model in still water is given.

5.1 Model-Based Control

The model-based control is implemented using iLQR. The dynamics function is given by (4) and (5). The running cost function l and final cost function l_f are defined as:

$$\begin{aligned} c &= l(s_t, a_t) \\ &= (s_t - s_{goal})^T Q(s_t - s_{goal}) + (a_t - a_{goal})^T R(a_t - a_{goal}), \end{aligned} \tag{7}$$

$$\begin{aligned} c_f &= l_f(s_t) \\ &= (s_t - s_{goal})^T Q(s_t - s_{goal}), \end{aligned} \tag{8}$$

where s_{goal} and a_{goal} are the goal state and control, Q is the quadratic state cost matrix, R is the quadratic control cost matrix. We then optimize the sequence of actions $A = \{a_0, a_1, \cdots, a_{N-1}\}$ over a whole trajectory with length N, using the given dynamics model to predict future states:

$$\begin{aligned} A^\star &= \arg\min_A J(s_0, A), \\ s_{t+1} &= f(s_t, a_t). \end{aligned} \tag{9}$$

Also, the inequality constraints on the control need to be take into consideration when optimize the control sequence [Tassa *et al.*, 2014] . We consider inequality constraints of the form:

$$\underline{b} \leq u \leq \overline{b} \tag{10}$$

with element-wise inequality and $\underline{b}$, $\overline{b}$ the respective lower and upper bounds. Tassa et al. [2014] has proposed an algorithm called box quadratic programming (QP), which accommodates box inequality constraints on the controls, without significantly sacrificing convergence quality or computational effort.

5.2 Training Imitation Policy

Trajectory optimizers are normally computationally expensive, since they need to solve an optimization problem every time they meet a new initial state s_0, which make them not suitable for real-time operation. However, a neural network policy can perform the control signals faster, the action selection only consumes the time for one forward propagation of the neural network. Thus, we then need to train a neural network policy to mimic our model-based controller.

We first gather example trajectories with the iLQR controller, which uses the given dynamics functions f and the cost function l and l_f. We collect the trajectories into a dataset $D^\star$, and we then train a neural network policy $\pi_\psi(a|s)$ to match these expert trajectories in $D^\star$. This policy's parameters are trained using the behavioral cloning objective [Nagabandi *et al.*, 2018]

$$\min_\psi \sum_{(s_t, a_t) \in D^\star} ||a_t - \pi_\psi(s_t)||_2^2 \tag{11}$$

which we optimize using stochastic gradient descent.

6 Hybrid Reinforcement Learning

Purely model-free reinforcement learning algorithms are normally sample inefficient, requiring a very large number of samples to achieve good performance; purely model-based approaches usually lag behind the model-free algorithms in final performance due to the model inaccuracy. To achieve both the optimal control performance and the data efficiency, we can combine the benefits of model-based and model-free learning. Also, the given robot model only considers the situation in still water, without the explicit modeling of the current disturbance, so the model-based learning cannot keep the near-optimal performance under the external disturbance, this is another reason why we need an additional model-free learner. We propose a simple but highly effective method for combining our model-based approach with model-free methods by using the trained model-based policy as a priori, then training the model-free learner as a compensation for the output of the model-based policy. The final control signal will be the combination of the output of both the model-based policy and the model-free policy.

Some researchers [Nagabandi *et al.*, 2018] proposed to use the model-based policy as the initialization for the model-free reinforcement learning algorithm. However, in order to deal with the disturbance, the model-free reinforcement learning algorithm needs to use the history as the policy input, leading to different dimension of input space for model-based and model-free policies. This is the reason why we need to combine these two policies rather than using the model-based policy as the initialization for the model-free learner.

7 Experimental Results

7.1 Task Description

Our research addressed the control problems of an AUV subject to excessive external disturbance. We omitted the DOF in roll and pitch of the robot, since the robot is designed to be sufficiently stable in roll and pitch even under strong disturbance. Thus, the robot has a 8-dimensional state space and a 4-dimensional action space. In each episode of the experiment, the robot starts at a random pose, and it is controlled to reach a given pose and keep stable thereafter. The current disturbance is exerted on the x and y axes in the inertial frame.

The disturbance considered in the experiments is in the form of sinusoidal wave. We vary its amplitude, frequency and phase in each episode during training and evaluation, in order to prove that our algorithms are able to adapt to different disturbance patterns, rather than a fixed disturbance pattern.

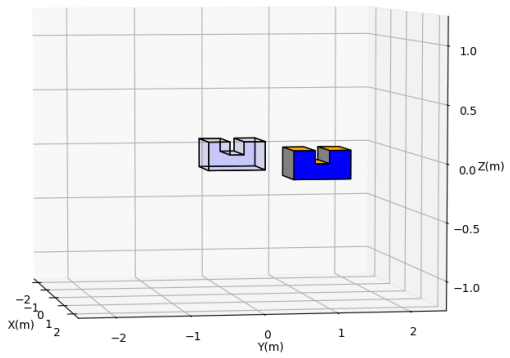

Figure 2: Simulated Autonomous Underwater Vehicle (solid: current pose, transparent: target pose)

7.2 Model-Free Reinforcement Learning

We'd like to enable the purely model-free reinforcement learning algorithm to handle the disturbance through taking the history as policy inputs. Different strength of the disturbance and different length of the history would affect the disturbance rejection capability. We first evaluate various situations for model-free reinforcement learning using empirical evaluations.

The external disturbance given in simulation were all sinusoidal waves with period ranging from 4s to 8s and phase ranging from 0 to 2π rad. We provided four different amplitude range of the disturbance, which were 50%-100%, 80%-120%, 100%-120% and 100%-150% of the robot control limits. Three choices of history length were given: 0s, 1s and 2s. We can see that larger disturbance amplitude leads to lower cumulative reward and slower convergence speed, this result accords with the common sense. We also learn from the robot trajectory data (omit here for brevity) that, the robot can always keep relatively stable if the disturbance does not exceed the control limits. Once the disturbance is larger than the control limits, the control stability will decrease significantly.

For the length of history, we could tell from the figure that, using shorter history length gives a better convergence performance. However, we believe that there should be an extremum, otherwise no history will be the best choice. This part of knowledge still requires more investigation.

7.3 Hybrid Reinforcement Learning

We now compare the purely model-free reinforcement learning algorithm with our hybrid approach. When there is neither disturbance nor history, the hybrid approach is apparently better than the model-free learner. The hybrid approach starts with a higher initial reward (-2000 vs. -9000), and is nearly 4 times faster (200 steps vs. 800 steps) to converge to an optimal value. If we take the disturbance and history into consideration, which is the exactly the problem we need to solve, we found that the hybrid approach still outperforms the model-free method, but the advantage is not that obvious.

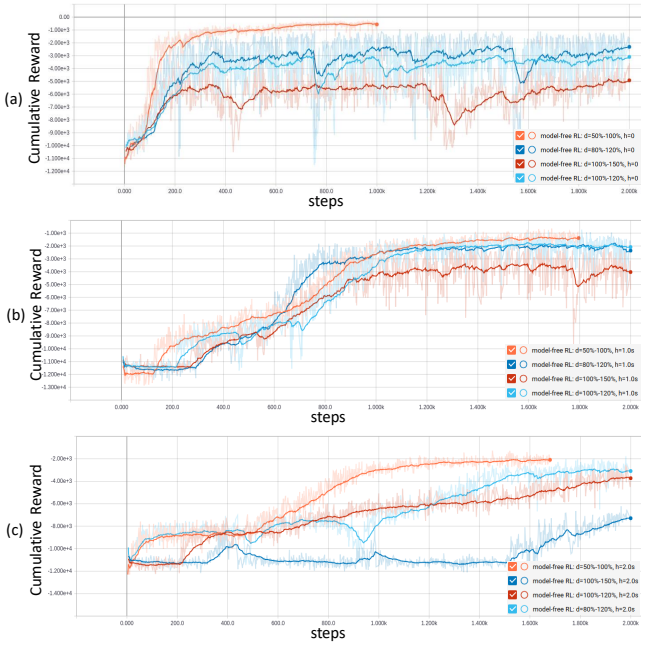

Figure 3: Model-free RL with different disturbance amplitude: (a) no history; (b) 1s history; (c) 2s history

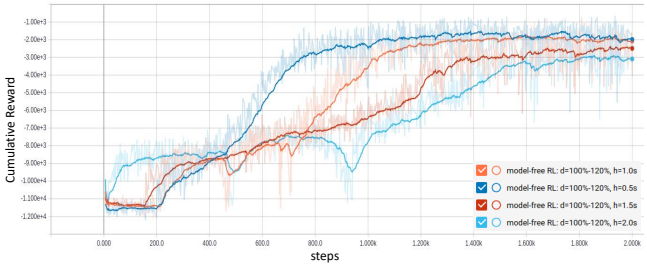

Figure 4: Model-free RL with different history length

This situation is even worse when taking the disturbance into consideration but not using the history.

This phenomenon might due to a design parameter for the hybrid RL algorithm. During the initial training steps, we only use model-based policy for action selection, and use the sampled data ro train the model-free learner. Then, after a certain number of steps, we combine the model-based policy and model-free policy together for action selection. The purpose of this setting is to avoid the initial random exploration of the model-free RL algorithm, and add the model-free policy to the running policy when it has satisfactory performance. However, the number of steps to add the model-free policy remains a problem. Currently we use 50 steps for the scenario without history and 200 steps for the scenario with history, we believe this part of work still need more investigation.

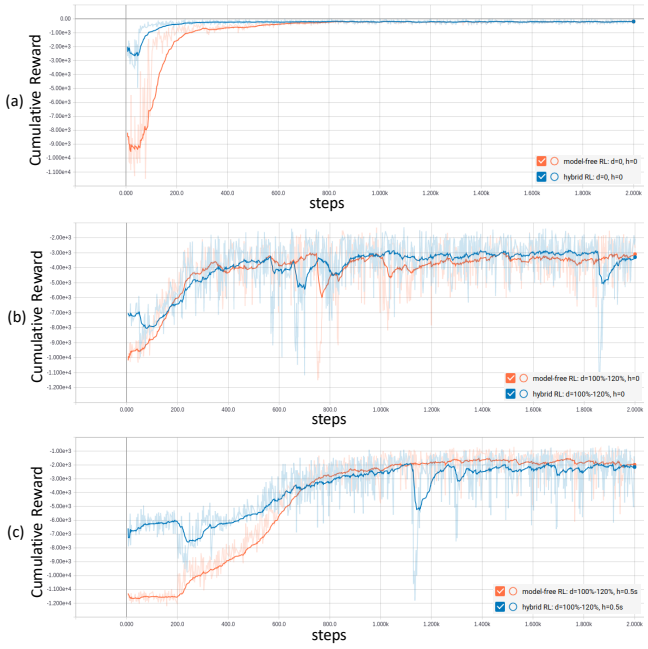

Figure 5: Comparison between model-free RL and hybrid RL: (a) no disturbance, no history; (b) has disturbance, no history; (c) has disturbance, has history

8 Conclusion

In this paper, we presented a new disturbance rejection control method, which used reinforcement learning and took a certain period of states and actions history as policy input. The results are convincing, especially when the external disturbance is exceed the robot control limits. We also proposed a model-based approach based on iLQR algorithm, the collected optimal trajectories were used for training an supervised policy. Then, we combined these two methods to realize both sample efficiency and optimal control performance under disturbance.

While the effectiveness and simplicity of our hybrid method is promising for ease of practical application, an interesting idea for future work is to investigate a more tight combination of model-based and model-free approaches, in order to further improve sample efficiency. Another direction for future work is to make more reasonable choice of the history length. The current algorithm directly take a bunch of past states and actions as the policy inputs, while these information could be utilized more sufficiently, for example, the convolutional neural network (CNN) or long short term memory (LSTM) could be considered to deal with these history data. Finally, the whole work will be implemented on a real underwater robot. In addition, the deployment of this method on real-world robotic systems is also a potential option, where the improved sample efficiency would make it practical to use even under the constraints of real-time sample collection in the real world.

References

[Åström and Wittenmark, 2013] Karl J Åström and Björn Wittenmark. *Adaptive control*. Courier Corporation, 2013.

[Bertsekas *et al.*, 1995] Dimitri P Bertsekas, Dimitri P Bertsekas, Dimitri P Bertsekas, and Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena scientific Belmont, MA, 1995.

[Brauer, 2012] Charles Brauer. Using eureqa in a stock daytrading application. *Cypress Point Technologies, LLC*, 2012.

[Camacho and Alba, 2013] Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer Science & Business Media, 2013.

[Chen and Guo, 2004] Wen-Hua Chen and Lei Guo. Analysis of disturbance observer based control for nonlinear systems under disturbances with bounded variation. In *Proceedings of International Conference on Control*, pages 1–5, 2004.

[Chen *et al.*, 2000] Wen-Hua Chen, Donald J Ballance, Peter J Gawthrop, and John O'Reilly. A nonlinear disturbance observer for robotic manipulators. *IEEE Transactions on industrial Electronics*, 47(4):932–938, 2000.

[Chen *et al.*, 2016] Wen-Hua Chen, Jun Yang, Lei Guo, and Shihua Li. Disturbance-observer-based control and related methodsan overview. *IEEE Transactions on Industrial Electronics*, 63(2):1083–1095, 2016.

[Deisenroth and Rasmussen, 2011] Marc Deisenroth and Carl E Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pages 465–472, 2011.

[Deisenroth *et al.*, 2013] Marc Peter Deisenroth, Gerhard Neumann, Jan Peters, et al. A survey on policy search for robotics. *Foundations and Trends® in Robotics*, 2(1–2):1–142, 2013.

[Derner *et al.*, 2018] Erik Derner, Jiří Kubalík, and Robert Babuška. Data-driven construction of symbolic process models for reinforcement learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8. IEEE, 2018.

[Dirscherl *et al.*, 2015] Christian Dirscherl, CM Hackl, and Korbinian Schechner. Explicit model predictive control with disturbance observer for grid-connected voltage source power converters. In *Industrial Technology (ICIT), 2015 IEEE International Conference on*, pages 999–1006. IEEE, 2015.

[Doyle *et al.*, 1989] John C Doyle, Keith Glover, Pramod P Khargonekar, and Bruce A Francis. State-space solutions to standard h/sub 2/and h/sub infinity/control problems. *IEEE Transactions on Automatic control*, 34(8):831–847, 1989.

[Edwards and Spurgeon, 1998] Christopher Edwards and Sarah Spurgeon. *Sliding mode control: theory and applications*. Crc Press, 1998.

[Forbes and Andre, 2002] Jeffrey Forbes and David Andre. Representations for learning control policies. In *Proceedings of the ICML-2002 Workshop on Development of Representations*, pages 7–14, 2002.

[Gao and Cai, 2016] Haiyan Gao and Yuanli Cai. Nonlinear disturbance observer-based model predictive control for a generic hypersonic vehicle. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 230(1):3–12, 2016.

[Gao *et al.*, 2001] Zhiqiang Gao, Yi Huang, and Jingqing Han. An alternative paradigm for control system design. In *Decision and Control, 2001. Proceedings of the 40th IEEE Conference on*, volume 5, pages 4578–4585. IEEE, 2001.

[Gao, 2014] Zhiqiang Gao. On the centrality of disturbance rejection in automatic control. *ISA transactions*, 53(4):850–857, 2014.

[Ghafarirad *et al.*, 2012] H Ghafarirad, SM Rezaei, M Zareinejad, and M Hamdi. A robust adaptive control for micro-positioning of piezoelectric actuators with environment force estimation. *Transactions of the Institute of Measurement and Control*, 34(8):956–965, 2012.

[Ghafarirad *et al.*, 2014] Hamed Ghafarirad, Seyed Mehdi Rezaei, Mohammad Zareinejad, and Ahmed AD Sarhan. Disturbance rejection-based robust control for micropositioning of piezoelectric actuators. *Comptes Rendus Mécanique*, 342(1):32–45, 2014.

[Gu *et al.*, 2016a] Shixiang Gu, Timothy Lillicrap, Zoubin Ghahramani, Richard E Turner, and Sergey Levine. Q-prop: Sample-efficient policy gradient with an off-policy critic. *arXiv preprint arXiv:1611.02247*, 2016.

[Gu *et al.*, 2016b] Shixiang Gu, Timothy Lillicrap, Ilya Sutskever, and Sergey Levine. Continuous deep q-learning with model-based acceleration. In *International Conference on Machine Learning*, pages 2829–2838, 2016.

[Guo and Cao, 2014] Lei Guo and Songyin Cao. Anti-disturbance control theory for systems with multiple disturbances: A survey. *ISA transactions*, 53(4):846–849, 2014.

[Han, 1995] Jingqing Han. The" extended state observer" of a class of uncertain systems [j]. *Control and Decision*, 1, 1995.

[Hester and Stone, 2017] Todd Hester and Peter Stone. Intrinsically motivated model learning for developing curious robots. *Artificial Intelligence*, 247:170–186, 2017.

[Jacobson and Mayne, 1970] David H Jacobson and David Q Mayne. Differential dynamic programming. 1970.

[Johnson, 1968] C Johnson. Optimal control of the linear regulator with constant disturbances. *IEEE Transactions on Automatic Control*, 13(4):416–421, 1968.

[Johnson, 1971] Cn Johnson. Accomodation of external disturbances in linear regulator and servomechanism problems. *IEEE Transactions on automatic control*, 16(6):635–644, 1971.

[Jong and Stone, 2007] Nicholas K Jong and Peter Stone. Model-based function approximation in reinforcement learning. In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, page 95. ACM, 2007.

[Kober *et al.*, 2013] Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[Koryakovskiy *et al.*, 2018] Ivan Koryakovskiy, Manuel Kudruss, Heike Vallery, Robert Babuška, and Wouter Caarls. Model-plant mismatch compensation using reinforcement learning. *IEEE Robotics and Automation Letters*, 3(3):2471–2477, 2018.

[Kumar *et al.*, 2018] Visak CV Kumar, Sehoon Ha, and Katsu Yamane. Improving model-based balance controllers using reinforcement learning and adaptive sampling. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7541–7547. IEEE, 2018.

[Kuvayev and Sutton, 1996] Leonid Kuvayev and Richard S Sutton. Model-based reinforcement learning with an approximate, learned model. In *in Proceedings of the Ninth Yale Workshop on Adaptive and Learning Systems*. Citeseer, 1996.

[Li and Todorov, 2004] Weiwei Li and Emanuel Todorov. Iterative linear quadratic regulator design for nonlinear biological movement systems. In *ICINCO (1)*, pages 222–229, 2004.

[Li *et al.*, 2014] Shihua Li, Jun Yang, Wen-Hua Chen, and Xisong Chen. *Disturbance observer-based control: methods and applications*. CRC press, 2014.

[Lillicrap *et al.*, 2015] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[Liu *et al.*, 2012] Cunjia Liu, Wen-Hua Chen, and John Andrews. Tracking control of small-scale helicopters using explicit nonlinear mpc augmented with disturbance observers. *Control Engineering Practice*, 20(3):258–268, 2012.

[Maeder and Morari, 2010] Urban Maeder and Manfred Morari. Offset-free reference tracking with model predictive control. *Automatica*, 46(9):1469–1476, 2010.

[Mayne, 1966] David Mayne. A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems. *International Journal of Control*, 3(1):85–95, 1966.

[Mnih *et al.*, 2015] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[Mnih *et al.*, 2016] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

[Nagabandi *et al.*, 2018] Anusha Nagabandi, Gregory Kahn, Ronald S. Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *Robotics and Automation (ICRA), 2018 IEEE International Conference on*, pages 7579–7586. IEEE, 2018.

[Oh *et al.*, 2016] Junhyuk Oh, Valliappa Chockalingam, Satinder Singh, and Honglak Lee. Control of memory, active perception, and action in minecraft. *arXiv preprint arXiv:1605.09128*, 2016.

[Ohishi *et al.*, 1987] Kiyoshi Ohishi, Masato Nakao, Kouhei Ohnishi, and Kunio Miyachi. Microprocessor-controlled dc motor for load-insensitive position servo system. *IEEE Transactions on Industrial Electronics*, (1):44–49, 1987.

[Schmidt and Lipson, 2009] Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *science*, 324(5923):81–85, 2009.

[Schulman *et al.*, 2015a] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

[Schulman *et al.*, 2015b] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

[Skogestad and Postlethwaite, 2007] Sigurd Skogestad and Ian Postlethwaite. *Multivariable feedback control: analysis and design*, volume 2. Wiley New York, 2007.

[Staelens *et al.*, 2013] Nicolas Staelens, Dirk Deschrijver, Ekaterina Vladislavleva, Brecht Vermeulen, Tom Dhaene, and Piet Demeester. Constructing a no-reference h. 264/avc bitstream-based video quality metric using genetic programming-based symbolic regression. *IEEE Transactions on Circuits and Systems for Video Technology*, 23(8):1322–1333, 2013.

[Sutton and Barto, 1998] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[Sutton, 1991] Richard S Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bulletin*, 2(4):160–163, 1991.

[Tassa *et al.*, 2014] Yuval Tassa, Nicolas Mansard, and Emo Todorov. Control-limited differential dynamic programming. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 1168–1175. IEEE, 2014.

[Todorov and Li, 2005] Emanuel Todorov and Weiwei Li. A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *American Control Conference, 2005. Proceedings of the 2005*, pages 300–306. IEEE, 2005.

[Umeno and Hori, 1991] Takaji Umeno and Yoichi Hori. Robust speed control of dc servomotors using modern two degrees-of-freedom controller design. *IEEE Transactions on Industrial Electronics*, 38(5):363–368, 1991.

[Umeno *et al.*, 1993] Takaji Umeno, Tomoaki Kaneko, and Yoichi Hori. Robust servosystem design with two degrees of freedom and its application to novel motion control of robot manipulators. *IEEE Transactions on Industrial Electronics*, 40(5):473–485, 1993.

[Woolfrey *et al.*, 2016] Jonathan Woolfrey, Dikai Liu, and Marc Carmichael. Kinematic control of an autonomous underwater vehicle-manipulator system (auvms) using autoregressive prediction of vehicle motion and model predictive control. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 4591–4596. IEEE, 2016.

[Xie and Guo, 2000] Liang-Liang Xie and Lei Guo. How much uncertainty can be dealt with by feedback? *IEEE Transactions on Automatic Control*, 45(12):2203–2217, 2000.

[Yang *et al.*, 2010] Jun Yang, Shihua Li, Xisong Chen, and Qi Li. Disturbance rejection of ball mill grinding circuits using dob and mpc. *Powder Technology*, 198(2):219–228, 2010.

[Yang *et al.*, 2011a] Jun Yang, W-H Chen, and Shihua Li. Non-linear disturbance observer-based robust control for systems with mismatched disturbances/uncertainties. *IET control theory & applications*, 5(18):2053–2062, 2011.

[Yang *et al.*, 2011b] Jun Yang, Shihua Li, Xisong Chen, and Qi Li. Disturbance rejection of dead-time processes using disturbance observer and model predictive control.

Chemical engineering research and design, 89(2):125–135, 2011.

[Yang *et al.*, 2014] Jun Yang, Zhenhua Zhao, Shihua Li, and Wei Xing Zheng. Nonlinear disturbance observer enhanced predictive control for airbreathing hypersonic vehicles. In *Control Conference (CCC), 2014 33rd Chinese*, pages 3668–3673. IEEE, 2014.

[Zeinali and Notash, 2010] Meysar Zeinali and Leila Notash. Adaptive sliding mode control with uncertainty estimator for robot manipulators. *Mechanism and Machine Theory*, 45(1):80–90, 2010.