

PID Tuning using Cross-Entropy Deep Learning: a Lyapunov Stability Analysis

Hector Kohler^{*} Benoit Clement^{*,**,*} Thomas Chaffre^{*,**}
Gilles Le Chenadec^{*}

^{*} *Lab-STICC UMR CNRS 6285, ENSTA Bretagne, Brest, France*
(e-mail: name.surname@ensta-bretagne.fr).

^{**} *Centre for Maritime Engineering, Flinders University, Australia*
(e-mail: name.surname@flinders.edu.au).

^{***} *CROSSING IRL CNRS 2010, Adelaide, Australia.*

Abstract: Underwater Unmanned Vehicles (UUVs) have to constantly compensate for the external disturbing forces acting on their body. Adaptive Control theory is commonly used there to grant the control law some flexibility in its response to process variation. Today, learning-based (LB) adaptive methods are leading the field where model-based control structures are combined with deep model-free learning algorithms. This work proposes experiments and metrics to empirically study the stability of such a controller. We perform this stability analysis on a LB adaptive control system whose adaptive parameters are determined using a Cross-Entropy Deep Learning method.

Keywords: Underwater Vehicle, Adaptive Control, Deep Learning, Lyapunov Stability.

1. INTRODUCTION

Operating in a constantly disturbed environment, Unmanned Underwater Vehicles (UUVs) must compensate for wave and current-induced forces acting on their body. In this context, a common practice is to exploit a Proportional-Integral-Differential (PID) control law, whose parameters are obtained generally using optimal model-based control theory and are then kept constant during operation. However, their performances decay under intensive process variation and uncertainties such as those found in the underwater environment, where the vehicle state measurements are limited and the uncertainty on the external disturbance is high. The performance of PID regulators can be improved by accommodating such changes using online tuning techniques, such as adaptive control theory O'Dwyer (2006); Chaffre et al. (2021, 2022).

Adaptive control methods are widely used in the context of dynamical systems and provide what seems to be an ideal framework for automatic tuning of regulators Åström and Wittenmark (2013); Parks (1981). New tuning techniques are emerging from the development of data-driven theory Brunton and Kutz (2019), with Genetic algorithm also suggested to tune PID regulators Jayachitra and Vinodha (2014). Deep Reinforcement Learning techniques (DRL) in particular, have shown great performance as optimization methods. They exploit the strong representation abilities offered by artificial neural networks (ANNs) to build non-linear mapping functions between sensor feedback and control inputs/controller parameters, directly in closed-loop systems. Such methods are denoted as learning-based (LB). Despite their increased performance, their usage in UUV applications, where process observability is low, is still limited due to the stability analysis being hardly practicable to conduct when using ANNs.

In this work, we propose an adaptive LB controller where the PID gains are automatically adjusted by an ANN for the task of minimizing setpoint tracking error (i.e. position and orientation) of UUVs under Lyapunov stability constraints. The ANN is trained with a Cross-Entropy Method (CEM) Rubinstein (1997) which, unlike gradient-based optimization methods, is a direct policy search technique. Moreover, the considered application allows us to apply the classic Lyapunov stability analysis.

This paper is organized as follows: Section 2 presents some related usage of Deep Learning methods for the control of UUVs. In Section 3, we present the considered robotic platform with its modeling and the simulation framework we used to train our controller. Our strategy for stability analysis is presented in Section 4. In Section 5 we describe our learning-based (LB) PID tuning approach. A complete description of our usage of Deep Learning is provided in Section 6 with the complete hyperparameters choice. The experiments are described in Section 7 with an analysis of the results, leading to some open questions.

2. RELATED WORK

Numerous aspects of PI-PID tuning are discussed in O'Dwyer (2006). The most common method consists of auto-tuning where an experiment is performed in open or closed-loop to estimate the process model by using recursive least squares. The idea is that if a second-order model can be generated, it can then be used to make optimal pole-placement. However, the underwater context does not allow such procedures as open-loop experiments are too dangerous for the vehicles and the limited observability of the system/environment does not allow satisfying system identification. Recently, Deep Learning methods have been successively used in the field of PID tuning.

Among the various existing techniques, Deep Reinforcement Learning (DRL) Sutton and Barto (2018) has shown great performance in the adaptive control of UUVs Wang et al. (2018), Knudsen et al. (2019). These methods are used to iteratively build an estimate of an optimal policy function, mapping the process states to the control parameters. The DRL-based methods oblige to frame the control process as a Markov Decision Process (MDP). The associated Bellman's equations are then estimated using ANNs, usually involving Gradient-based learning. In contrast, we proposed in this paper to use CEM to optimize the ANN, removing the need for the MDP formulation.

3. AUV MODELING AND SIMULATION

3.1 AUV model

A mathematical model of a ROV platform can be derived using the general equations of motion for a marine craft, which can be written in the vectorial form according to Fossen (2011) as:

$$\begin{cases} \dot{\eta} = J_{\Theta}(\eta)\nu \\ M\dot{\nu} + C(\nu)\nu + D(\nu)\nu + g(\eta) = \delta + \delta_{cable} \end{cases} \quad (1)$$

where η and ν are the position and velocity vectors respectively, δ is the control force vector and δ_{cable} is the vector describing the umbilical forces from the cable attached to the ROV. The RexROV2 is an ROV-type platform provided in UUV Simulator. It is propelled by 6 thrusters (complete details on its equation of motions are provided in Berg (2012), McCue (2016), Yang et al. (2015)). The control vector u is obtained by transforming the actuator force vector: $\delta = T(\alpha)Ku$, where $T(\alpha) \in \mathbb{R}^{n \times r}$ is the thrust allocation matrix; K is the thrust coefficient matrix; δ is the control force vector in n degrees of freedom (DoF) and $u \in \mathbb{R}^r$ is the actuator input vector. The package Manhães et al. (2016) allows us to design a vector of thruster contribution for every DoFs.

3.2 ROS packages

We used the ROS-based package called *UUV Simulator* proposed in Manhães et al. (2016), to train our controller. This package provides a Gazebo-based simulation of underwater environments and the possibility to use the existing RexROV2 platform described in Berg (2012) as a test vehicle. In addition, it is also possible to simulate several disturbances including sea currents. They are simulated as a uniform force in the Gazebo world and represented by a linear velocity v_c (in $m.s^{-1}$), a horizontal h_c , and a vertical angle j_c (in radians).

4. OUR LEARNING-BASED ADAPTIVE CONTROL FRAMEWORK

In this paper, we address the problem of setpoint regulation where the UUV's objective is to minimize its tracking error in all 6 DoFs. The tracking error e is defined as the error between, the current UUV's position and orientation, and a fixed desired position and orientation as:

$$e = \eta_d - \eta. \quad (2)$$

The position and orientation of the UUV is:

$$\eta = (x, y, z, roll, pitch, yaw), \quad (3)$$

and the desired setpoint:

$$\eta_d = (0, 0, 0, 0, 0, 0). \quad (4)$$

In our LB adaptive control framework, a parametrization such as an artificial neural network, of the PID's gains is learned (in simulation or in the real world). In the LB control framework's testing phase, at every state measurement, a control step is performed. In our case, during a control step, the state of the control process is used as input of a trained neural network. The outputs of the network parametrize the PID gains (see section 5) used to compute the control input: the UUV's thrusters. The PID control law is defined as the following:

$$u = B^{-1} [J^T(\eta) (k_p e + k_i \int_0^t e(\tau) d\tau - k_d \dot{\eta}) + g(\eta)], \quad (5)$$

where $u \in \mathbb{R}^6$ are the control inputs; the PID gain matrices $k_p, k_i, k_d \in \mathbb{R}^{6 \times 6}$ are the output of a neural network; $\eta \in \mathbb{R}^6$ is the current UUV's state, i.e. position and orientation (Eq. 3); $\dot{\eta} \in \mathbb{R}^6$ is the temporal derivative of the state vector; $g(\eta)$ is the sum of external forces acting on the UUV's body (in our case, gravity); $e \in \mathbb{R}^6$ is the tracking error (Eq. 2); B^{-1} is the fixed known thrusters allocation matrix mapping the control input u into a combination of thruster power inputs resulting to the desired movement and $J^T(\eta)$ is a transformation matrix.

The principal advantage of the considered PID regulator (Eq. 5) is that global stability and convergence analysis of the system has been well formalized by Fossen Fossen (1994). Lyapunov stability theory Liberzon (2005) is straightforward to apply with such a control law. Following Fossen (1994), there exists a Lyapunov function for the considered UUV such that:

$$V(x) = \frac{1}{2} x^T \begin{bmatrix} M_{\eta}^{-1} & \alpha I & 0 \\ \alpha I & k_p & k_i \\ 0 & k_i & \alpha k_i \end{bmatrix} x, \quad (6)$$

where $\alpha \in \mathbb{R}$ is a small positive constant and the PID gains are $k_p, k_i, k_d \in \mathbb{R}^{6 \times 6}$; M_{η}^{-1} is related to the UUV's mass and can be computed from the current η ; x is the control loop's state (not to be mistaken with the UUV's state): $x = [p, \eta, \int_0^t e(\tau) d\tau]^T \in \mathbb{R}^{18}$ and $p = M_{\eta} \dot{\eta}^T \in \mathbb{R}^6$ is the generalized momentum depending on the UUV's mass and velocity. The control loop is stable at state x if:

$$V(x) > 0 \text{ and } \dot{V}(x) < 0. \quad (7)$$

Following Lyapunov stability theory Liberzon (2005), there exists theoretical constraints on the gain matrices k_p, k_i, k_d and the small constant α such that local stability is guaranteed when the initial conditions of the systems are closed to $x = 0$. According to the proposed Lyapunov function (Eq. 6), the vehicle stability and convergence to steady-state are guaranteed (Eq. 7) if the following constraints are satisfied:

$$\begin{cases} k_d > M_{\eta}, \\ k_i > 0, \\ k_p > k_d + \frac{2}{\alpha} k_i, \\ \frac{1}{2}(1 - \alpha)k_d - \alpha M_{\eta} + \frac{\alpha}{2} \sum_{i=1}^6 (\eta_i - \eta_{id}) \frac{\partial M_{\eta}}{\partial \eta_i} > 0, \\ \alpha > 0, \end{cases} \quad (8)$$

We found that, when limiting the parameter space to value satisfying (Eq. 8), the resulting space is so small that the benefits of adaptive control are merely preserved. Therefore, we propose not to take into account the stability constraints in the parameters optimization. Our objective is then to assess to what extent the resulting solution, obtained from an ANN, can still hold some stability components. In order to facilitate the stability analysis, we aim at reducing the dimension of the space depicted in Eq. (8). First, we can transform (8) into equalities as follows:

$$\begin{cases} k_d = M_\eta + M_1, \\ k_i = 0 + M_2, \\ k_p = k_d + \frac{2}{\alpha} k_i + M_3, \\ \alpha = \left\| \frac{-k_d}{(-k_d - 2M_\eta + \sum_{i=1}^6 (\eta_i - \eta_{id}) \frac{\partial M_\eta}{\partial \eta_i})} \right\|_{max} + \epsilon, \end{cases} \quad (9)$$

where M_1 , M_2 and M_3 are three 6×6 positive matrices and ϵ is a small positive constant. With this transformation (9), the fulfillment of the Lyapunov stability constraints (8) can now be verified by only assessing the value of $[M_1, M_2, M_3, \epsilon] \in \mathbb{R}^{3 \times 6 \times 6 + 1, > 0}$. In order to further reduce this dimension space, we apply a diagonal transformation on the matrices M_i : $M_i = P \Lambda_i P^{-1}$, where $\Lambda_i \in \mathbb{R}^6$ are positive vectors and $P \in \mathbb{R}^{6 \times 6}$ is a positive invertible matrix chosen randomly beforehand. Thanks to this transformation, we can now assess the value of $[M_1, M_2, M_3, \epsilon] \in \mathbb{R}^{3 \times 6 \times 6 + 1, > 0}$ (and K_p, K_i, K_d with Eq.9) by only accessing:

$$[\Lambda_1, \Lambda_2, \Lambda_3, \epsilon] \in \mathbb{R}^{19, > 0}. \quad (10)$$

5. PID TUNING USING DEEP LEARNING

In order to take into account the process uncertainties, we propose to learn a stochastic predictive model π_ω (parameterized by ω) that maps the system state vector x into the controller parameters of the PID law:

$$\begin{cases} \pi : \Omega_x \subset \mathbb{R}^{18} & \mapsto \Theta \subset \mathbb{R}^{109} \\ x = \left[p, \eta, \int_0^t e(\tau) d\tau \right]^T & \mapsto [k_p, k_i, k_d, \alpha] \end{cases} \quad (11)$$

This mapping is composed of two stages. The first stage is a trainable neural network whose input is the state vector x , aiming at predicting the vector $[\Lambda_1, \Lambda_2, \Lambda_3, \epsilon] \in \mathbb{R}^{19}$. To this end, we model each variable $n \in \{1, \dots, 19\}$ of this vector as a random variable with an independent Normal distribution as:

$$\mathcal{N}_n(\mu_n, \sigma_n^2) = (2\pi\sigma_n^2)^{-1/2} \exp\left\{-\frac{1}{2\sigma_n^2}(x - \mu_n)^2\right\}, \quad (12)$$

The goal of this network is to output the 19 pairs of (μ, σ) representing the action distributions. We have empirically set up an architecture composed of 2 hidden layers of 32 hidden nodes each, and with the Sigmoid activation function applied to each layer. This results in a total of $(18 + 1) \times 32 + (32 + 1) \times 32 + 2 \times ((32 + 1) \times 19) = 2918$ parameters (ω) to learn from data. In a second stage of the mapping π , the PID parameters $[\Lambda_1, \Lambda_2, \Lambda_3, \epsilon]$ are obtained by sampling from the resulting Gaussian distributions $\mathcal{N}(\mu, \sigma)$. The Eq. 9 allows to compute the final PID parameters and using Eq. 5 the PID control inputs are derived. With this setting, we can more easily assess the Lyapunov stability of the system. The overall control strategy is illustrated in Figure 1.

6. LEARNING WITH THE CROSS ENTROPY METHOD

We propose to learn/optimize the weights of the neural network with the CEM. The CEM is a direct search method (i.e no gradient is computed). It is an Estimation of Distribution Algorithm (EDA) inspired by Natural Evolution Strategies Wierstra et al. (2008). With CEM, during one iteration k , N sets of weights $S_{i=1 \dots N}^k$ are sampled from a Normal distribution directly in the space of weights \mathbb{R}^{2918} . At each iteration k , N evaluations are made to determine the current best weights $S_{best=1 \dots N \times \rho}^k$ with respect to a given cost function. In our case, we used the following classic control performance (based on multi steps within an episode and connected to the tracking error (Eq.2) index as a cost function to minimize:

$$J = \sum_{steps} \frac{1}{6} \sum_{i=1}^6 (\eta_{d,i} - \eta_i)^2. \quad (13)$$

The mean and covariance of the Normal distribution are then updated as the mean and the covariance of the $N \times \rho$ best sets of weights obtained at iteration $k-1$. Noise σ_{noise}^2 is added to the covariance of the best weights to avoid local optima. We randomly sample the next iteration weights $S^k = \mathcal{N}(\text{mean}(S_{best}^{k-1}), \text{Cov}(S_{best}^{k-1}) + \sigma_{noise}^2)$.

7. EXPERIMENTS AND ANALYSIS

7.1 Experimental settings

Simulation: We control a simulated UUV in Gazebo as described in Section 3. The simulation is running in real-time and the frequency of the control loop is 20 Hz. The outputs of the ANNs, are used to compute the PID control law that is ultimately applied to the 6 thrusters. The complete training loop of the ANN is given in Figure 1.

Training setup: We used the following CEM hyperparameters that have been chosen through a grid search: population size $N = 25$, proportion to keep $\rho = 0.2$ and added noise $\sigma_{noise}^2 = 0.1$. Each training episode is composed of 200 timesteps. One epoch is defined as performing one episode using each of the N sets of parameters candidate. The training consists in performing 200 epochs, thus a total of $200 \times 200 \times 25 = 10^6$ timesteps.

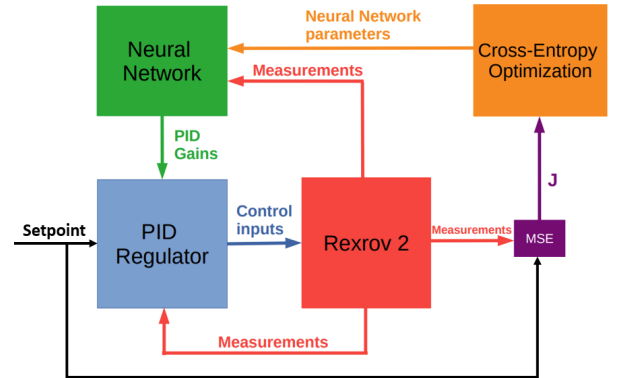


Fig. 1. Block diagram of the proposed method for PID tuning loop using Cross-Entropy Deep Learning.

Stability metrics: Thanks to the proposed Lyapunov function (6) and mapping (10), we can directly assess the state and parameters stability. We propose to measure the controller's stability as the respective percentage of steps satisfying the state and parameters constraints.

Evaluation setup: We propose three evaluation scenarios based on induced disturbance: none, Gaussian noise in sensor measurements, and Gaussian noise in control inputs with current disturbances. The length of the episode is now increased to 2000 timesteps. The initial state of the UUV is changed at the beginning of each episode. We compare our LB adaptive controller to its Lyapunov-based counterpart which consist in a PID controller whose parameters are set to: $M_1, M_2, M_3 = \text{diag}(0.5 - 10^{-5}, 0.5 - 10^{-5}, 0.5 - 10^{-5}, 0.5 - 10^{-5}, 0.5 - 10^{-5}, 0.5 - 10^{-5}) + 10^{-5} \times \mathbf{J}_{6 \times 6}$ (which satisfies the parameters stability constraints in Eq.(9)). The resulting controller remains adaptive (as the gains are a function of the vehicle state \mathbf{x}) and will be denoted as naive PID henceforth. The only difference between these controllers is the value of the parameters used to derive the PID law (5), making the comparison fair.

7.2 Control performance

The control performance is measured as the MSE on the setpoint. The evolution of this performance is illustrated in the following figures: when facing no disturbance in figure 2; when facing Gaussian noise on the vehicle position and orientation feedback in figure 3 and when facing Gaussian noise on the control inputs and sea current disturbance in figure 4.

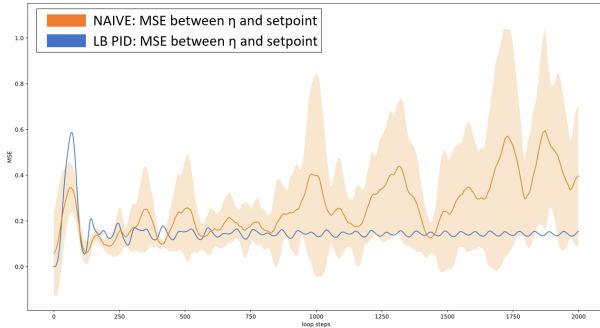


Fig. 2. Control performance without disturbance.

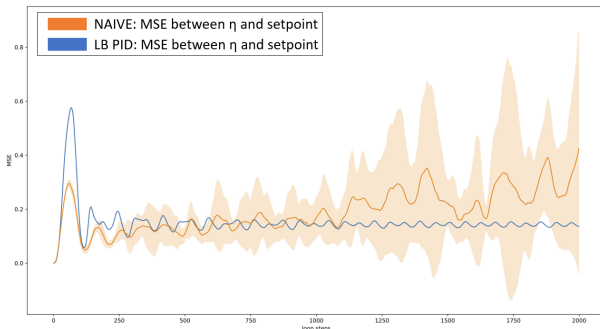


Fig. 3. Control performance with noisy position and orientation measurements.

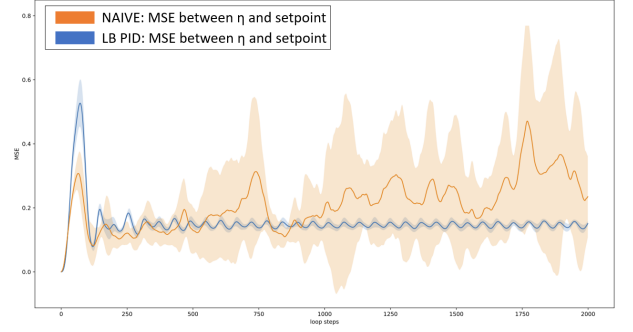


Fig. 4. Control performance with noisy control inputs and sea current disturbance.

We can see that the naive PID (in blue) results overall in better performances in terms of setpoint tracking compared to the LB PID (in orange). Nevertheless, we can see that the relative performance of the LB PID with respect to the naive PID's performance improves with increased uncertainty. In Figures 3 and 4, we can see that the performance of the LB PID matches the naive PID for approximately 500 timesteps, while without disturbance, its performance drops notably earlier as shown in figure 2. The evolution of the vehicle's state is represented in Figures 5-10. We can observe a difference in regulation dynamics depending on the DoF. For instance, we can see that the depth of the UUV is not successfully regulated by none of the controllers. Due to the short latency between episodes and the position of the UUV's CoG, the UUV slightly sinks at the beginning of the episode, altering its depth and yaw angle (z, yaw). This explains the vertical drift in their associated errors observed in the figures. In addition, as seen in Figures 5-10, the LB PID tends to regulate effectively more DoF compare to the naive PID (which fails at regulating (z, yaw), see Figures 5, 7, 9).

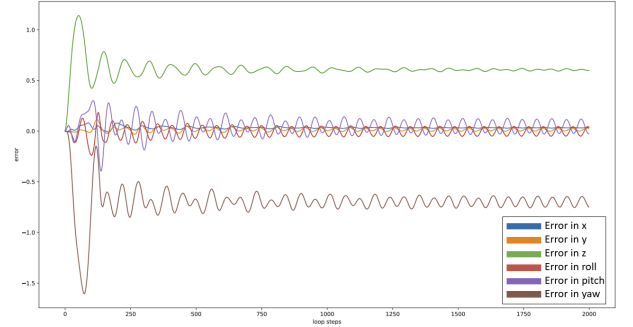


Fig. 5. The position and orientation errors of the naive PID during a non-disturbed episode.

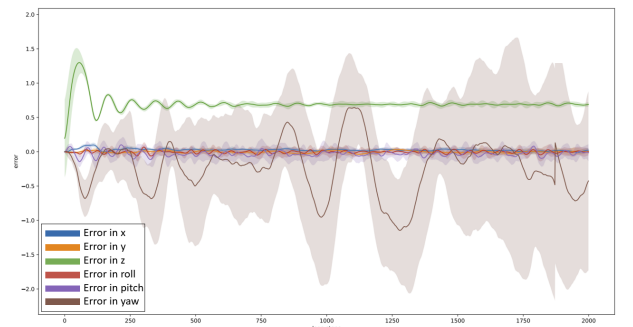


Fig. 6. The position and orientation errors of the LB PID during a non-disturbed episode.

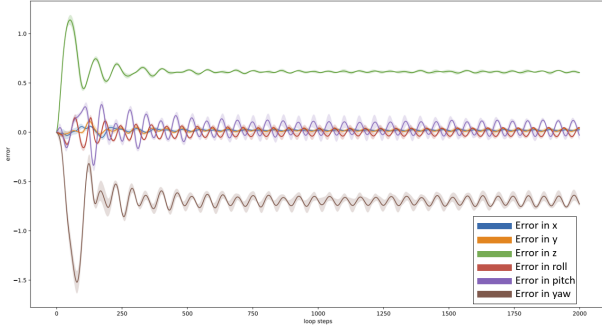


Fig. 7. The position and orientation errors of the naive PID during an episode with measurements noise.

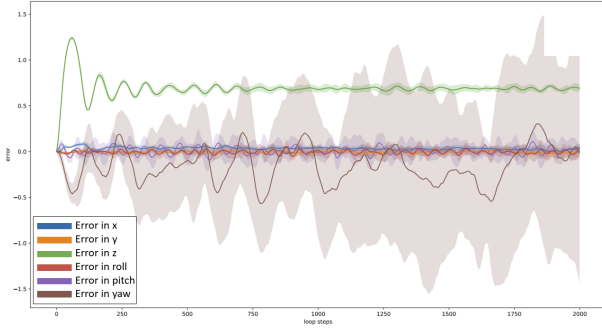


Fig. 8. The position and orientation errors of the LB PID during an episode with measurements noise.

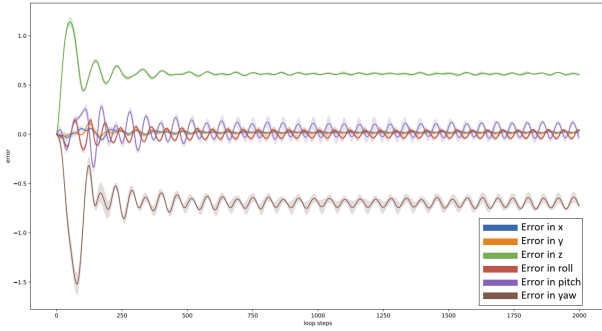


Fig. 9. Position and orientation errors of the naive PID with noisy control inputs and sea current disturbance.

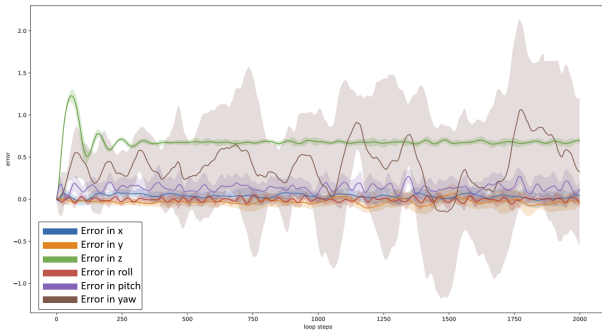


Fig. 10. The position and orientation errors of the LB PID with noisy control inputs and sea current disturbance.

7.3 Stability performance

The evolution of the stability metrics is illustrated in Figures 11-16. We can see in Figure 11 that both controllers reach a similar level of state stability. The naive PID exceeds the stability performance of the LB PID in both state and parameters stability. We believe that this is thanks to the stochastic nature of the adaptation (see Eq.(12)). Note that similar state stability does not guarantee similar control performances. Indeed, as mentioned before, both controllers reach the same level of state stability, however, we have seen that the MSE of the LB PID increases over time, so its control performance decreases but the control loop remains stable. Finally, we can see a mismatch between state and parameters stability: despite reaching a percentage of state stability as good as the naive PID, the gains obtained from the LB PID are not satisfying the constraints (8). This suggests that the Lyapunov function holds a limited conservatism. In other words, the Lyapunov-based space of parameters seems extremely small compared to the space of parameters obtained from the ANN. Future work could focus on incorporating the parameter constraints in the learning optimization scheme to analyze the difference in the resulting spaces.

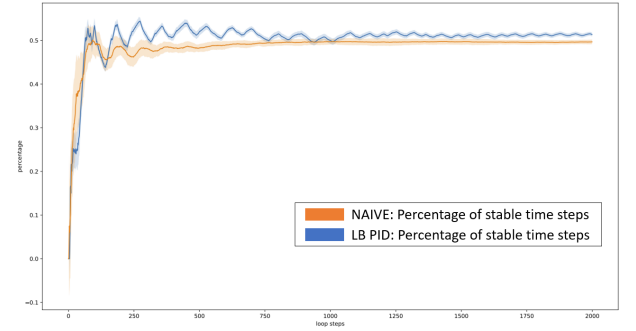


Fig. 11. Evolution of the state stability during a non-disturbed simulation

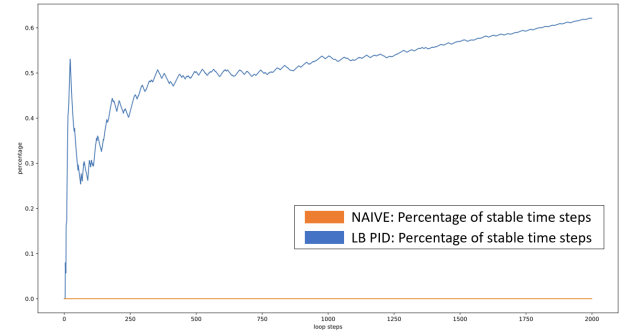


Fig. 12. Evolution of the parameters stability during a non-disturbed simulation

8. CONCLUSION

We proposed the use of CEM for PID tuning to adapt to process variation. Despite not considering stability components in the CEM procedure, the proposed LB PID is able to match the state stability obtained with the Lyapunov-based PID controller. Our result also suggests that the LB PID is better at compensating process variation. Learning-based adaptive control might be a key ingredient toward the safe deployment of fully autonomous UUVs.

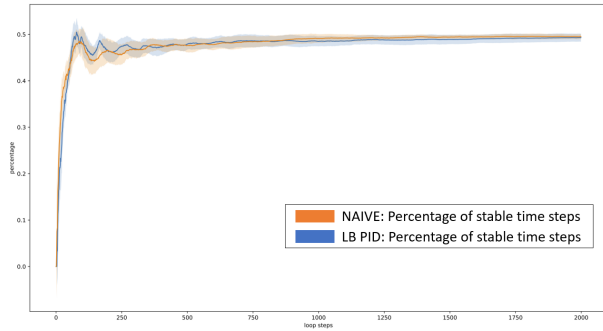


Fig. 13. Evolution of the state stability during a simulation with measurements noise

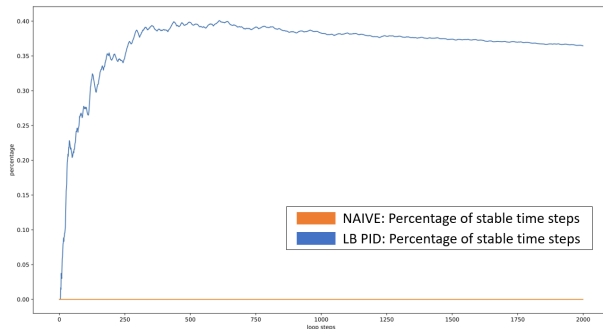


Fig. 14. Evolution of the parameters stability during a simulation with measurements noise

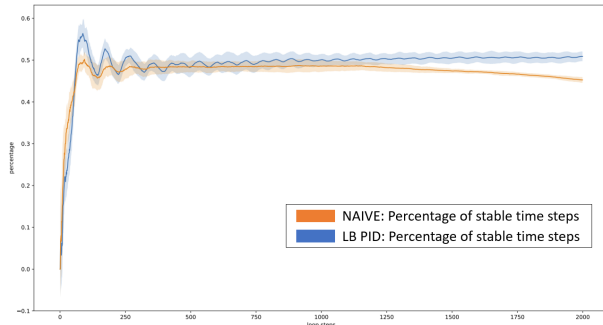


Fig. 15. Evolution of the state stability during a simulation with current and thrusters noise

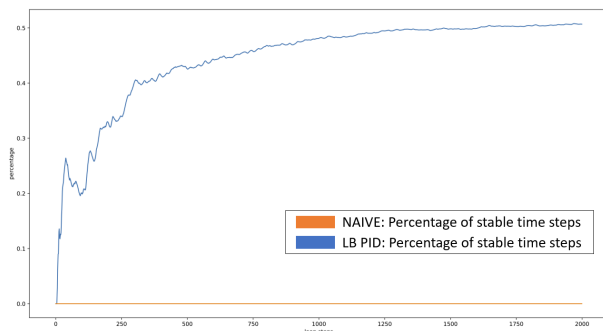


Fig. 16. Evolution simulation of the parameters stability during a simulation with current and thrusters noise

REFERENCES

- Åström, K.J. and Wittenmark, B. (2013). *Adaptive Control*. Courier Corporation.
- Berg, V. (2012). *Development and Commissioning of a DP system for ROV SF 30k*. Ph.D. thesis, NTNU.
- Brunton, S.L. and Kutz, J.N. (2019). *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press.
- Chaffre, T., Chenadec, G.L., Sammut, K., Chauveau, E., and Clement, B. (2021). Direct adaptive pole-placement controller using deep reinforcement learning: Application to auv control. 13th IFAC CAMS.
- Chaffre, T., Moras, J., Chan-Hon-Tong, A., Marzat, J., Sammut, K., Le Chenadec, G., and Clement, B. (2022). Learning-based vs model-free adaptive control of a mav under wind gust. *Informatics in Control, Automation and Robotics*, 362–385.
- Fossen, T. (1994). *Nonlinear Modelling And Control Of Underwater Vehicles*. Ph.D. thesis, NUST.
- Fossen, T.I. (2011). *Handbook of Marine Craft Hydrodynamics and Motion Control*. Wiley.
- Jayachitra, A. and Vinodha, R. (2014). Genetic algorithm based pid controller tuning approach for continuous stirred tank reactor. *Adv. Artif. Intell.*
- Knudsen, K.B., Nielsen, M.C., and Schjøberg, I. (2019). Deep learning for station keeping of auvs. In *MTS/IEEE OCEANS*.
- Liberzon, D. (2005). Review of liapunov functions and stability in control theory. *Automatica*, 41.
- Manhães, M., Scherer, S., Voss, M., Douat, L., and Rauschenbach, T. (2016). UUV simulator: A gazebo-based package for underwater intervention and multi-robot simulation. In *MTS/IEEE OCEANS*. Monterey.
- McCue, L. (2016). *Handbook of marine craft hydrodynamics and motion control [bookshelf]*. *IEEE Control Systems Magazine*.
- O'Dwyer, A. (2006). *Handbook of PI and PID controller tuning rules*. Imperial College Press.
- Parks, P. (1981). Stability and convergence of adaptive controllers-continuous systems. In *IEE Proceedings D-Control Theory and Applications*. IET.
- Rubinstein, R.Y. (1997). Optimization of computer simulation models with rare events. *European Journal of Operational Research*, 99, 89–112.
- Sutton, R.S. and Barto, A.G. (2018). *Reinforcement learning an introduction - Second edition*. MIT Press.
- Wang, C., Wei, L., Wang, Z., Song, M., and Mahmoudian, N. (2018). Reinforcement learning-based adaptive trajectory planning for auvs in under-ice environments. In *MTS/IEEE OCEANS*.
- Wierstra, D., Schaul, T., Peters, J., and Schmidhuber, J. (2008). Natural evolution strategies. *IEEE Congress on Evolutionary Computation*.
- Yang, R., Clement, B., Mansour, A., Li, M., and Wu, N. (2015). Modeling of a complex-shaped underwater vehicle for robust control scheme. *Journal of Intelligent and Robotic Systems*.