Predictive Linear Online Tracking for Unknown Targets

Anastasios Tsiamis [*1] **Aren Karapetyan** [*1] **Yueshan Li** [2] **Efe C. Balta** [1 3] **John Lygeros** [1]

Abstract

In this paper, we study the problem of online tracking in linear control systems, where the objective is to follow a moving target. Unlike classical tracking control, the target is unknown, non-stationary, and its state is revealed sequentially, thus, fitting the framework of online non-stochastic control. We consider the case of quadratic costs and propose a new algorithm, called predictive linear online tracking (PLOT). The algorithm uses recursive least squares with exponential forgetting to learn a time-varying dynamic model of the target. The learned model is used in the optimal policy under the framework of receding horizon control. We show the dynamic regret of PLOT scales with $\mathcal{O}(\sqrt{TV_T})$, where V_T is the total variation of the target dynamics and T is the time horizon. Unlike prior work, our theoretical results hold for non-stationary targets. We implement PLOT on a real quadrotor and provide open-source software, thus, showcasing one of the first successful applications of online control methods on real hardware.

1. Introduction

Target tracking is a fundamental control task for autonomous agents, allowing them to be used in a variety of applications including environmental monitoring (Aucone et al., 2023), agriculture (Daponte et al., 2019), and air shows (Schoellig et al., 2014) to name a few. Typically, an autonomous agent is given a reference trajectory, which should be tracked with as little error as possible. In the case of linear systems with quadratic costs the problem, also known as Linear Quadratic Tracking (LQT), can be formulated as follows. Let the autonomous agent be governed by the linear dynamics

$$x_{t+1} = Ax_t + Bu_t, \tag{1}$$

where $x_t \in \mathbb{R}^n$ is the state, and $u_t \in \mathbb{R}^m$ is the control input, while $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ denote the *known* system dynamics. Given a target trajectory $r_t \in \mathbb{R}^{n\times 1}$, $t \geq 0$ the goal is finding the optimal control input of the finite-horizon problem

$$\min_{u_{0:T-1}} \sum_{t=0}^{T-1} \left(\|x_t - r_t\|_Q^2 + \|u_t\|_R^2\right) + \|x_T - r_T\|_{Q_T}^2, \tag{2}$$

subject to the dynamics (1), for some total time (horizon) T. The state, input, and terminal penalties, $Q \in \mathbb{R}^{n\times n}$, $R \in \mathbb{R}^{m\times m}$, $Q_T \in \mathbb{R}^{n\times n}$, respectively, are design choices.

In classical target tracking, the target state r_t is known a priori and is precomputed. However, in many cases, the target might be time-varying and *unknown*. Such scenarios arise, for example, in the case of adversarial target tracking, wild animal tracking, moving obstacle avoidance, pedestrian tracking, etc. In such cases, the target trajectory is generated by an external unknown dynamical system, also known as "exosystem" (Nikiforov & Gerasimov, 2022). In this work, we assume that the target is generated by time-varying, autoregressive (AR) dynamics. Let

$$z_t = \begin{bmatrix} r_t^\top & \cdots & r_{t-p+1}^\top \end{bmatrix}^\top$$

contain past target states, with p the memory or past horizon of the dynamics. Then, the target at the next time step is given by the AR model

$$r_{t+1} = S_{t+1} z_t, \tag{3}$$

where $S_t \in \mathbb{R}^{n\times np}$ is an *unkonwn* matrix.

Adaptive control has a long history of dealing with dynamical uncertainty (Annaswamy & Fradkov, 2021), where the goal is to simultaneously control the system and adapt to the uncertainty. One of the most widely-used algorithms in the adaptive control literature with ubiquitous applications has been the celebrated Recursive Least Squares (RLS) algorithm with forgetting factors (Åström et al., 1977). While adaptive control has been extensively studied in the stochastic or time-invariant regime (Guo & Ljung, 1995;

[*]Equal contribution [1]Automatic Control Laboratory, ETH Zürich, Zürich, Switzerland [2]Institute for Dynamic Systems and Control, ETH Zürich, Zürich, Switzerland [3]Control and Automation Group, inspire AG, Zürich, Switzerland. Correspondence to: Anastasios Tsiamis <atsiamis@ethz.ch>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

Ljung & Gunnarsson, 1990), results for non-stochastic, time-varying systems are relatively scarce. By studying the problem through the modern lens of online learning and non-stochastic control (Hazan & Singh, 2022), we can provide guarantees for new classes of time-varying systems.

Most works on adaptive control follow one of the two possible paradigms: i) *direct* control and ii) *indirect* control, mirroring model-free and model-based Reinforcement Learning. In the former case, adaptation occurs directly at the control policy; in the latter case, a model of the uncertainty is kept at all times and the control policy is updated indirectly based on the current model. A notable benefit of indirect architectures is that they decouple control from learning, making it easier to incorporate changes in the control objective or robustness specifications. In the case of non-stochastic control, learning the optimal feedforward law (Agarwal et al., 2019a; Foster & Simchowitz, 2020) fits the direct paradigm. Prior works that use predictions (Li et al., 2019; Zhang et al., 2021b) are closer to the indirect paradigm but they either do not provide a prediction method or the regret guarantees are suboptimal in the case of targets with dynamic structure.

1.1. Contribution

In this work, we propose a tracking control algorithm, called Predictive Linear Online Tracking (PLOT). It adapts to unknown targets online using sequentially measured target data, based on the indirect paradigm. PLOT uses a modified version of the RLS algorithm with forgetting factors (Yuan & Lamperski, 2020) to learn the target dynamics and predict future target states. Then, it computes a receding horizon control input using the predicted target states in a certainty equivalent fashion. To characterize closed-loop performance, we use the notion of dynamic regret (Zinkevich, 2003). In our setting, dynamic regret compares the incurred closed-loop cost of any online algorithm against the cost incurred by the optimal control actions in hindsight, that is, the actions generated by the optimal non-causal policy that has full knowledge about future target states. Our contributions are the following.

Dynamic regret for online tracking. We prove that the dynamic regret of PLOT is upper-bounded by $\mathcal{O}(\sqrt{TV_T})$, where V_T is the total variation of the target dynamics S_t (path length). When the target dynamics are static (zero variation), the regret becomes logarithmic and we recover prior results (Foster & Simchowitz, 2020). While prior dynamic regret bounds exist for linear quadratic control, they either focus on direct online control (Baby & Wang, 2022), where we learn over a directly parameterized disturbance affine feedback policy, or they assume no structure for the targets (Li et al., 2019; Karapetyan et al., 2023). Instead, we focus on indirect online control, where we learn an internal model representation for the unknown target. By setting up multiple step ahead predictors, we have more flexible and frequent feedback, thus avoiding extra logarithmic factors due to delayed learning feedback as in (Baby & Wang, 2022; Foster & Simchowitz, 2020). Using the total variation of the target *dynamics* S_t to characterize dynamic regret is a new point of view, improving prior bounds that use the variation of the target state r_t itself (Li et al., 2019).

Prediction of time-varying partially observed systems. We employ the RLS algorithm to predict target states multiple time steps into the future. Our result is of independent interest since it applies to time-varying AR systems or systems with exogenous inputs (ARX), which is a common model in system identification. We obtain dynamic regret guarantees for prediction by adapting the analysis of Yuan & Lamperski (2020) to the setting of learning with delayed feedback (Joulani et al., 2013). This result can be seen as a generalization of time-invariant Kalman filtering to the non-stochastic, time-varying, multi-step ahead case.

Experimental demonstration. While the theory of online non-stochastic control has matured over the past years (Hazan & Singh, 2022), its practice is still lagging, with only a few notable experimental demonstrations on hardware (Suo et al., 2021; Snyder et al., 2023). In this work, we perform extensive simulations showing how the regret analysis can be used as a tool to tune the controller hyperparameters, provide comparisons of several online control methods in simulations, and implement PLOT on a hardware setup. To facilitate benchmarking and future developments, we make our code open-source and implement it on Crazyflie drones with a reconfigurable software architecture built on the Robot Operating System (ROS). To the best of our knowledge, our work is one of the first to demonstrate the use of online non-stochastic control with guarantees on a real quadrotor.

1.2. Organization and Notation

The rest of the paper is organized as follows. Section 2 states the assumptions and defines the control objective. The proposed algorithm, PLOT, is presented in Section 3, and its dynamic regret is analyzed in Section 4. The experiment results in simulation and on hardware are presented in Section 5. Section 6 provides concluding remarks and future directions. Additional background material, detailed proofs, and further implementation and experimental details can be found in the Appendix.

For a matrix M, $\|M\|$ denotes the ℓ_2 induced operator norm, while $\|M\|_F$ denotes the Frobenius norm. For positive definite $P \succ 0$, we define the weighted Frobenius norm as $\|M\|_{F,P} = \sqrt{\mathrm{tr}(MPM^\top)}$, where $\mathrm{tr}(\cdot)$ denotes the trace. For a vector $x \in \mathbb{R}^n$, $\|x\|$ denotes the Euclidean norm

and $\|x\|_P = \sqrt{x^T P x}$ denotes the weighted P norm for $P \succ 0$. We use $x_{1:t} := \{x_1, ..., x_t\}$ as a sequence of variables spanning from time step 1 to t. Given $n \in \mathbb{N}^+$, the identity matrix of dimension n is defined by $\mathbf{I}_n$.

2. Problem Statement

Consider the LQT problem (2). Let the target state r_t be sequentially revealed. The following events happen in sequence at every time step t. i) The current state of the system x_t and the target r_t are received and the current tracking cost is incurred; ii) The controller applies an action u_t; iii) The system and the target evolve according to (1), (3) respectively. We assume the initial value z_{-1} to be known.

Many fundamental targets can be captured by dynamics (3). For example, a target moving on a circle or a straight line with a constant speed can be captured by linear time-invariant dynamics–see Appendix C.1. By considering time-varying dynamics, we can model more complex target trajectories, e.g., combinations of lines, circles, switching patterns, etc. Our results can be generalized directly to targets driven by exogenous variables (ARX) $r_{t+1} = S_{t+1} z_t + v_{t+1}$–it is sufficient to redefine $\tilde{z}_t = \begin{bmatrix} z_t^\top & 1 \end{bmatrix}^\top$ and $\tilde{S}_t \triangleq \begin{bmatrix} S_t & v_t \end{bmatrix}$. We can thus capture even richer targets, including other control systems. In fact, *any* arbitrary bounded adversarial target can be captured by such a model; it is sufficient to take $S_{t+1} = 0$ and $v_{t+1} = r_{t+1}$. We note that the above representation is not unique–see also Appendix C.1.

We remark that the LQT problem is equivalent to the Linear Quadratic Regulator (LQR) with disturbances–see Appendix B. Hence, the techniques applied here could also be applied to noisy system dynamics (1), where we attach a dynamic structure to the disturbances.

By applying the one step ahead model (3) multiple times, we obtain k-step ahead recursions of the form

$$r_{t+k} = S_{t+k|t} z_t, \tag{4}$$

where $k \in \mathbb{N}$ is the number of future steps. The $k-$step ahead matrix $S_{t+k|t}$ is a nonlinear function (multinomial) of $S_{t+k}, \dots, S_{t+1}$. Expressions for the multi-step ahead recursions (4) can be found in Appendix C.2. By definition $S_{t+1|t} \equiv S_{t+1}$. The notation $S_{t+k|t}$ indicates that given all information z_t at time t, the product $S_{t+k|t} z_t$ acts as a $k-$step ahead prediction of r_{t+k} at time t.

To ensure that the online tracking problem is well-defined, we consider the following boundedness assumptions on the target dynamics and trajectory.

Assumption 2.1 (Bounded Signals)**.** The target state is bounded. For some $D_r \geq 0$, we have $\|r_t\| \leq D_r$, for $t = 1, ..., T$.

Assumption 2.2 (Stability)**.** Let $\mathcal{S} \triangleq \{S \in \mathbb{R}^{n \times pn} \| S \| \leq M\}$, for some $M \geq 0$. The $k-$step ahead dynamics are uniformly bounded $\|S_{t+k|t}\| \in \mathcal{S}$, for all $t = 0, \dots, T$, $k \leq T - t$.

Such boundedness conditions reflect typical assumptions in online learning (Yuan & Lamperski, 2020). The second condition allows general S_k as long as the multi-step ahead matrices do not blow up.

2.1. Control Objective

Our goal is to design an online controller that adapts to the unknown target dynamics. To evaluate the online performance, we use dynamic regret, which compares the incurred cost of the online controller to the *optimal* controller in hindsight that knows all future target states in advance (called simply the *optimal controller*). For a target state realization $\mathbf{r} := r_{0:T}$, we denote the cumulative cost by

$$J_T(u_{0:T-1}; \mathbf{r}) \triangleq \sum_{t=0}^{T-1} \left(\|x_t - r_t\|_Q^2 + \|u_t\|_R^2 \right) + \|x_T - r_T\|_{Q_T}^2.$$

The optimal controller minimizes the cumulative cost, given knowledge of the whole target realization

$$u^*_{0:T-1} = \operatorname{argmin}_{u_{0:T-1}} J_T(u_{0:T-1}; \mathbf{r}).$$

On the contrary, for an online policy π, the input u_t is a function of the current state x_t and the target states only up to time step t, so that

$$u_t^\pi = \pi_t(x_t; r_{1:t}), \quad \forall t = 0, ..., T-1,$$

where the notation u^π denotes the control input under the policy π. In other words, the optimal controller u_t^* is *non-causal* while the online controller u_t^π is *causal*.

The dynamic regret is given by the cumulative difference between the cost achieved by the online causal policy and the cost achieved by the non-causal optimal controller

$$\mathcal{R}(\pi) = J_T(u^\pi_{1:T-1}; \mathbf{r}) - J_T(u^*_{1:T-1}; \mathbf{r}), \tag{5}$$

defined for a target state realization $\mathbf{r}$. We can now summarize the main objective of the paper.

Problem 1 (Dynamic Regret)**.** Design an online controller π that adapts to the unknown target states r_t, and characterize the dynamic regret $\mathcal{R}(\pi)$.

To make sure that the control problem is well-defined, we introduce the following assumption which is standard in the control literature (Zhang et al., 2021b). It guarantees that the LQT controller will be (internally) stable.

Assumption 2.3 (Well-posed LQT)**.** The pair (A, B) is stabilizable and Q, R are symmetric positive definite.

We assume the terminal cost matrix Q_T is chosen as the solution X of the Discrete Algebraic Riccati Equation.

Assumption 2.4 (Terminal Cost)**.** We select $Q_T = X$, where X is the unique solution to Riccati equation

$$X = Q + A^\top XA - A^\top XB(R + B^\top XB)^{-1}B^\top XA. \quad (6)$$

Assumption 2.4 is only introduced to streamline the presentation and it is not restrictive. If we select a different terminal cost, then the effect on the optimal total cost (2) will be negligible (will only differ by a constant).

2.2. LQT Optimal Controller

Having full access to future target states, the optimal controller admits the following analytical solution (Foster & Simchowitz, 2020; Goel & Hassibi, 2022)

$$u_t^*(x_t) = -K(x_t - r_t) - \underbrace{\sum_{i=t}^{T-1} K_{i-t}(Ar_i - r_{i+1})}_{q_t(r_{t:T})}, \quad (7)$$

where the feedback and feedforward matrices are given by

$$K = (R + B^\top XB)^{-1}B^\top XA, \quad (8)$$

$$K_t = (R + B^\top XB)^{-1}B^\top (A - BK)^{\top,t}X, \; t \geq 0, \quad (9)$$

with X defined in (6). Since the dynamics (1) are known, all gain matrices K, K_t, $t \geq 0$ are known. The feedback term $-K(x_t - r_t)$ can be computed since the error $x_t - r_t$ is measured before choosing the action. The only unknown is the feedforward term $q_t(r_{t:T})$. Note that the *functional form* of the feedforward term is known. The missing piece of information is the future target states. Following the paradigm of indirect control, we will use this structural observation to decompose the problem of online tracking into one of online prediction and control design.

A notable property of LQT is that, under Assumption 2.3, matrix $(A - BK)$ has all eigenvalues inside the unit circle. Therefore, there exist $c_0 > 0$ and $\rho \in (0, 1)$ such that

$$\|K_k\| \leq c_0\rho^k, \text{ for all } k \geq 0. \quad (10)$$

Upper bounds for c_0, ρ as well as a relaxed version of Assumption 2.3 can be found in Appendix B. Interestingly, the above property implies that future target states get discounted when considering the current action; we only need to know accurately the imminent target states.

3. Predictive Linear Online Tracking

Our proposed online tracking algorithm can be decoupled into two steps at every time step t: i) predicting the future target states for up to $W-$steps into the future, for some prediction horizon W; ii) computing the current online control action based on the certainty equivalence principle and the receding horizon control framework.

Target Prediction. For the prediction step, we employ the RLS algorithm with forgetting factors. At every time step t, the RLS algorithm provides predictions $r_{t+1|t}, \ldots, r_{t+W|t}$ of the future target states $r_{t+1}, \ldots, r_{t+W}$, for some horizon $W > 0$. For every step ahead prediction, we keep W separate predictors $\hat{S}_{t+1|t}$,..., $\hat{S}_{t+W|t}$ without exploiting the shared structure. This parameterization leads to a higher-dimensional but convex problem. In contrast, learning over matrices $S_{t+W}, \ldots, S_{t+1}$ leads to non-convex optimization problems since the multi-step ahead predictors are products of entries in $S_{t+W}, \ldots, S_{t+1}$–see also Remark 3.1.

When predicting the target state $r_{t+k|t}$ for k steps ahead, the true value r_{t+k} is revealed after k steps into the future. To deal with this *delayed feedback* issue we follow the approach of Joulani et al. (2013). Within each $k-$step ahead predictor, we maintain k copies $\{\hat{S}_{t+k-j|t-j}\}_{j=0}^{k-1}$--we refer to them as learners–which are updated independently at non-overlapping time steps. Hence, the estimate $\hat{S}_{t+k|t}$ is updated based only on pairs of $(r_{t-ik}, z_{t-(i+1)k})$, for $i = 0, 1, 2, \ldots$. At each time step t, only one of the k independent learners is invoked, which aims to minimize the following prediction error $\forall t > 1$:

$$\min_{S \in \mathcal{S}} \sum_{i=0}^{\lfloor (t+1)/k \rfloor - 1} \gamma^i f_{t-ik,k}(S), \quad (11)$$
$$f_{\tau,k}(S) \triangleq \|r_\tau - Sz_{\tau-k}\|^2$$

where $\gamma \in (0, 1]$ is the forgetting factor that attributes higher weights to more recent data points. Recall that $\mathcal{S}$ is the set in which the dynamics lie–see Assumption 2.2. Note that each independent learner is updated based on at most $\lceil T/k \rceil$ data points. Instead of solving (11), we opt for a recursive implementation with projections adapted from Yuan & Lamperski (2020) and can be found in Algorithm 1.

Algorithm 1 RLS for k-step-ahead prediction

Require: Forgetting factor $\gamma \in (0, 1)$, Regularizer $\varepsilon > 0$.
Initialize k learners $\hat{S}_{j+k|j} \in \mathcal{S}$, $j = -1, ..., k-2$.
Initialize $P_{j|j-k} = \varepsilon \mathbf{I}$ with $j = -1, ..., k-2$.
for $t = k-1, \ldots, T$ **do**
  Predict $r_{t|t-k} = \hat{S}_{t|t-k} z_{t-k}$.
  Receive true target state r_t; Incur loss $f_{t,k}(\hat{S}_{t|t-k})$
  Update $P_{t|t-k} = \gamma P_{t-k|t-2k} + z_{t-k} z_{t-k}^T$;
  $\hat{S}^*_{t+k|t} = \hat{S}_{t|t-k} + (r_t - \hat{S}_{t|t-k} z_{t-k}) z_{t-k}^\top P_{t|t-k}^{-1}$.
  Project $\hat{S}_{t+k|t} = \Pi_{\mathcal{S}}^{P_{t|t-k}}(\hat{S}^*_{t+k|t})$,
  where $\Pi_{\mathcal{S}}^{P_{t|t-k}}(Y) \triangleq \arg\min_{S \in \mathcal{S}} \|S - Y\|_{F, P_{t|t-k}}$.
end for

Since we need k learners for every $k-$step ahead predictor, we have a total number of $W(W+1)/2$ learners. However, only W of them are actively updated at every time step (one active learner per predictor). An visualization of the

learning architecture for $k = 1, 2, 3$ can be found in Figure 7 in Appendix D.2.

Different from direct approaches (Foster & Simchowitz, 2020; Baby & Wang, 2022), where the learning feedback is delayed by W steps, here the learning feedback delay adapts to the $k-$step ahead horizon offering more flexibility. For example, the feedback for the one-step ahead predictor is always available without waiting W steps. A downside is that we update more predictors.

Remark 3.1 (Improper Learning versus Single Learner). To obtain the multi-step ahead predictions $\hat{r}_{t+k|t}$, $k = 1, \ldots, W$, we keep W *separate* predictors $\hat{S}_{t+1|t}$,..., $\hat{S}_{t+W|t}$ without exploiting the shared structure, i.e., we follow improper learning. By following this choice, we convexify the problem, potentially at the expense of higher sample complexity. An alternative, "naive" approach is to estimate the one-step ahead predictor $\hat{S}_{t+1|t}$, using a single learner. Then, to predict k steps ahead, e.g. $\hat{r}_{t+k|t}$, we could propagate the target states through $\hat{S}_{t+1}$, for k time steps. However, propagating through multiple time steps can lead to instability. For example, if $z_t = r_t$ (namely the AR memory length $p = 1$), we will have $\hat{r}_{t+k|t} = \hat{S}^k_{t+1} r_t$. Due to the exponent k, this could lead to a rapid accumulation of errors. Enforcing the boundedness constraint of Assumption 2.2 directly on $\hat{S}^k_{t+1}$ would not work either since it leads to a non-convex problem. A detailed comparison of PLOT and this "naive" approach in simulation can be found in Appendix F.4

Receding Horizon Control. In the receding horizon control framework, instead of looking over the total time T as in (2), we only look over a window of length W and solve the following optimal control problem

$$\begin{aligned} \min_{u_{t:t+W-1}} \sum_{k=0}^{W-1} & \left(\|x_{t+k} - r_{t+k|t}\|_Q^2 + \|u_{t+k}\|_R^2\right) \\ & + \|x_{t+W} - r_{t+W|t}\|_P^2 \\ \text{s.t.} \quad & x_{t+k+1} = Ax_{t+k} + Bu_{t+k},\ k \le W-1, \end{aligned} \quad , \tag{12}$$

where $r_{t+k|t}$ are the predictions of the RLS algorithm. We replace the true target states of (2) with their prediction in (12) according to the certainty equivalence principle. The receding horizon control policy that minimizes (12) can be computed in closed form as

$$u_t^\pi(x_t) = -K(x_t - r_t) - \sum_{i=t}^{t+W-1} K_{i-t}(Ar_{i|t} - r_{i+1|t}). \tag{13}$$

The functional form of the receding horizon control is similar to the one of the optimal controller in (7). The main difference is that the feedforward term is applied to the predicted disturbances, instead of the actual ones.

Algorithm 2 PLOT: Predictive Linear Online Tracking

Require: Horizon W, forgetting factor $\gamma \in (0, 1)$
Compute $X, K, \{K_0, ..., K_{W-1}\}$ as in (6), (8), (9).
Initialize RLS predictors according to the Algorithm 1, for $k = 1, ..., W$, respectively.
for t = 0, ..., $T - 1$ **do**
  Observe system state x_t, target state r_t.
  for $k = 1, ..., W$ **do**
    if $t + k \le T$ **then**
      Update the k-step-ahead predictor and predict $r_{t+k|t}$ according to the Algorithm 1.
    else
      Set $r_{t+k-1|t} = 0$.
    end if
  end for
  Compute u_t^π as in (13)
end for

The main part of our proposed algorithm, Predictive Linear Online Tracking (PLOT), is described in Algorithm 2. We initialize W RLS predictors (Algorithm 1) to estimate the k-step-ahead dynamics $S_{t+k|t}$ for $k = 1, ..., W$ independently. At each time step, the algorithm receives the latest target and state measurements, updates the RLS predictors, and makes new predictions. Then, it computes the receding horizon control input given the predictions and the state information. The whole procedure is iterated over all time steps up to the total time T. Note that if the prediction time $k + t$ exceeds the total time T, then we set the predicted target to zero; as seen by (7) only the target states up to time T affect the control problem.

4. Dynamic Regret and Tuning

To characterize the performance of the PLOT algorithm, we provide dynamic regret bounds in terms of the total variation of the target dynamics, which is defined as

$$V_T \triangleq \sum_{t=1}^{T} \|S_t - S_{t-1}\|_F. \tag{14}$$

For arbitrary $\gamma \in (0, 1)$, we have the following guarantees.

Theorem 4.1 (Dynamic Regret)**.** *Select a prediction horizon W and a forgetting factor $\gamma \in (0, 1)$. Let ρ be the decay rate of the LQT gains as in (10) and let $\tilde{W} = \min\{(1-\rho)^{-1}, W\}$. The dynamic regret of the PLOT policy, as given by Algorithm 2, is upper bounded by*

$$\begin{aligned} \mathcal{R}(\pi) \le\ & \alpha_1 \rho^{2W}(1-\rho)^{-2}T + \alpha_2 \tilde{W}^4 V_T (1-\gamma)^{-1} \\ & - \alpha_3 \tilde{W}^2 (T+1)\log\gamma - \alpha_4 \tilde{W}^3 \log(1-\gamma) + \alpha_5 \tilde{W}^3, \end{aligned}$$

where $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ (given in (28)) are positive constants related to system-specific constants, the state dimension n, and the target memory p.

The first term in the regret bound captures the truncation effect, i.e., the fact that we only use W-step (instead of T) ahead predictions in (12). The other terms capture the effect of prediction errors on the control performance. Note that there is a tradeoff between large and small prediction horizons. Smaller prediction horizons lead to better prediction performance but worse truncation error and, conversely, larger prediction horizons improve the truncation term but degrade the prediction performance. Nonetheless, if we increase the prediction horizon W past the threshold of $(1-\rho)^{-1}$, then the regret guarantees stop degrading, thus improving prior work (Foster & Simchowitz, 2020; Baby & Wang, 2022). Essentially, $\tilde{W}$ can be thought of as the "effective" prediction horizon; beyond $\tilde{W}$ any prediction errors have negligible effect. This is a consequence of deploying multiple step-ahead predictors of varying delays and the exponentially decaying LQT gains,

Assume that γ is close to 1. Then $-\log\gamma \approx 1-\gamma$ and the dominant regret terms are the second and third terms, $V(1-\gamma)^{-1}$ and $T(1-\gamma)$ respectively. By balancing these two terms, we obtain the following interpretable rate, similar to Yuan & Lamperski (2020).

Corollary 4.2 (Tuning)**.** *Select prediction horizon* $W = -\frac{\log T}{2\log\rho}$ *and forgetting factor* $\gamma = 1-\sqrt{\frac{\max\{V_T,\log^2 T/T\}}{4MT}}$*, where* ρ *is the decay rate of the LQT gains as in* (10)*. Then, the dynamic regret of the PLOT policy is upper-bounded by*

$$\mathcal{R}(\pi) = \max\{\mathcal{O}(\sqrt{TV_T}), \mathcal{O}(\log T)\}.$$

To deal with the truncation term, it is sufficient to choose a prediction horizon W, which grows logarithmically with the total time T, as also noted in Zhang et al. (2021b). The constant $4M$ in the denominator guarantees that the forgetting factor is positive. To tune the forgetting factor and obtain the rate, we require knowledge of the path length V_T, which is not always available. Following the procedure of Baby & Wang (2022), we can overcome this limitation by initiating multiple learners at various time steps and by running a follow-the-leading-history (FLH) meta-algorithm on top, which can also improve the dynamic regret bound to $T^{1/3}V_T^{2/3}$. We do not explore this possibility here, since we focus on studying the performance of the RLS algorithm with forgetting factors. When the path length is close to 0, i.e. the target dynamics are almost static, we obtain logarithmic regret guarantees. In this case, the optimal policy can be rewritten as a static affine control law, with respect to the past target states. The regret, in this case, is the static regret with respect to the best (static) affine policy, recovering the result of Foster & Simchowitz (2020).

Remark 4.3 (Path Length and Complexity). In certain works (Li et al., 2019; Karapetyan et al., 2023), the dynamic regret is given with respect to the total variation of the target states themselves, that is, $L_T = \sum_{t=2}^{T-1}\|r_t - r_{t-1}\|$. In contrast, here, we capture learning complexity by the total variation of the target dynamics V_T. We argue that the former notion of complexity can be suboptimal in the case of targets with dynamic structure, while ours is superior in this setting. For example, consider a target that is moving on a circle with constant velocity–see Example C.2. Then, the former path length is linear $L_T = \mathcal{O}(T)$ while the latter is zero $V_T = 0$. In control applications, many targets of interest have dynamic structures, e.g. tracking other drones, tracking moving objects, etc. Nonetheless, if the targets are fully unstructured and arbitrary, PLOT might not perform as well since it tries to fit a dynamic model to arbitrary data–see Appendix F.3.2. Still, the regret order cannot be worse than the one in prior work if we are learning over ARX models–see discussion in Appendix C.1.

The full proof of Theorem 4.1 and Corollary 4.2 can be found in Appendix E. In the following, we provide a sketch of the proof. First, similar to Foster & Simchowitz (2020), we invoke the "performance difference lemma" (Kakade, 2003) to turn the control problem into a prediction one [1].

Lemma 4.4 (Performance Difference Lemma (Foster & Simchowitz, 2020))**.** *For any policy* π *such that* $u_t^\pi = -K(x_t - r_t) + \hat{q}_t$*, where* K *is given by* (8)*, and the optimal action defined in* (7)*, the dynamic regret can be written as*

$$\mathcal{R}(\pi) = \sum_{t=0}^{T-1}\|\hat{q}_t - q_t(r_{t:W})\|^2_{(R+B^TXB)}.$$

The above fundamental result shows that the control problem can be cast as a prediction problem, where we try to predict the affine part of the optimal control law. Second, we bound the prediction error by analyzing the dynamic regret of the RLS algorithm following the steps of Yuan & Lamperski (2020). Define the prediction regret for $k-$step ahead prediction as

$$\mathcal{R}^{(k)}_{\text{pred}} \triangleq \sum_{t=k-1}^{T}\|\hat{S}_{t|t-k}z_{t-k} - r_t\|^2.$$

Since the sequence $r_{0:T}$ satisfies (3), there is no subtracted term in the regret; matrices $S_{t|t-k}$ achieve zero error. In the Appendix, we prove regret bounds for the prediction problem that hold for non-realizable sequences as well.

Theorem 4.5 (Regret for AR system prediction)**.** *Define the total variation of the* k*-step ahead dynamics as* $V_T^k \triangleq \sum_{t=k}^{T-k}\|S_{t+k|t} - S_{t|t-k}\|$*. The dynamic regret of the RLS algorithm (Algorithm 1) for* k*-steps ahead prediction is upper bounded by*

$$\mathcal{R}^{(k)}_{\text{pred}} \le \frac{\beta_1}{1-\gamma}V_T^k + \beta_2 T\log\frac{1}{\gamma} + k\beta_3\log\frac{1}{1-\gamma} + k\beta_4,$$

[1] In our case, the optimal controller is the true minimizer of the cost, hence, the regret in this case becomes the advantage function of (Foster & Simchowitz, 2020).

where $\beta_1, \beta_2, \beta_3, \beta_4$ are given in (23) and are positive constants related to system-specific constants, the state dimension n, and the target memory p.

Again, we can select the forgetting factor similar to Corollary 4.2, to obtain more specific regret bounds. The dominant dimensional dependence is hidden in the coefficient β_2 and is of the order of np^2 which is worse by p compared to the optimal linear regression rate of np. This is not a limitation of the RLS algorithm but an artifact of the norm of z_t scaling with $\sqrt{p}$. We can overcome this limitation by imposing "fading memory" constraints on the dynamics. If we rewrite the dynamics as $S_t z_{t-1} = S_t^{[1]} r_{t-1} + \cdots + S_t^{[p]} r_{t-p}$, then fading memory constraints would force $S_t^{[i]}$ to decay exponentially with i.

To conclude the proof, we combine the previous steps while also accounting for the truncation effect. The only remaining step is to upper bound the k-steps ahead path lengths V_T^k in terms of the path length of the one-step-ahead path length V_T as defined in (14). In Lemma C.3, we prove that $V_T^k \leq \sqrt{p} k^2 M^2 V_T$. This is why the factor $\tilde{W}^4$ that multiplies the path length in the final regret bound in Theorem 4.1 appears with an exponent of 4 instead of just 2.

5. Simulations and Experimental Validation

In this section, we demonstrate the performance of PLOT both in simulation and in hardware experiments. We study the setting of tracking adversarial unknown targets with a quadrotor, which is of particular interest given the number of potential applications, such as hunting adversarial drones in airports (Dressel & Kochenderfer, 2019) or artistic choreography (Schoellig et al., 2014), to name a few. The code for both the simulation and hardware experiments can be accessed from `https://gitlab.nccr-automation.ch/akarapetyan/plot`.

We consider the Crazyflie 2.1 quadrotor (Bitcraze, 2023b), a versatile, open-source, nano-sized quadrotor developed by Bitcraze (Bitcraze, 2023b), whose dynamics can be modeled with a linear time-invariant system linearized around a hovering point (Beuchat, 2019). For such a model, we define the state $x := [\vec{p}\,;\, \dot{\vec{p}}\,;\, \vec{\psi}]$, where $\vec{p} := \begin{bmatrix} x & y & z \end{bmatrix}^\top$ is the position vector in the inertial frame, $\vec{\psi} := \begin{bmatrix} \gamma & \beta & \alpha \end{bmatrix}^\top$ is the attitude vector in the inertial frame with γ, β and α for the roll, pitch and yaw angle, respectively. The action $u := [f\,;\, \vec{\omega}]$ includes the total thrust f and the angular rate $\vec{\omega} := \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^\top$ in the body frame. For the detailed derivation of the linear dynamic model, see Appendix F.1 and (Beuchat, 2019). For all examples, we take the sampling time to be $T_s = 0.1$ seconds and the LQR cost matrices fixed at $Q := diag(80, 80, 80, 10, 10, 10, 0.01, 0.01, 0.1)$, and $R := diag(0.7, 2.5, 2.5, 2.5)$.

5.1. Simulation Results

Here, we consider the linearized dynamics of the Crazyflie quadrotor, derived in Appendix F.1.We show how PLOT successfully tracks both a static reference target with $V_T = 0$ path length and a dynamic target with $V_T = \mathcal{O}(\sqrt{T})$. In addition, we also demonstrate how the proven regret guarantees can be used as a guideline for tuning the algorithm parameters such as the prediction horizon length W and the forgetting factor γ. We also provide a performance comparison to other online control algorithms from the literature for tracking a non-stationary target.

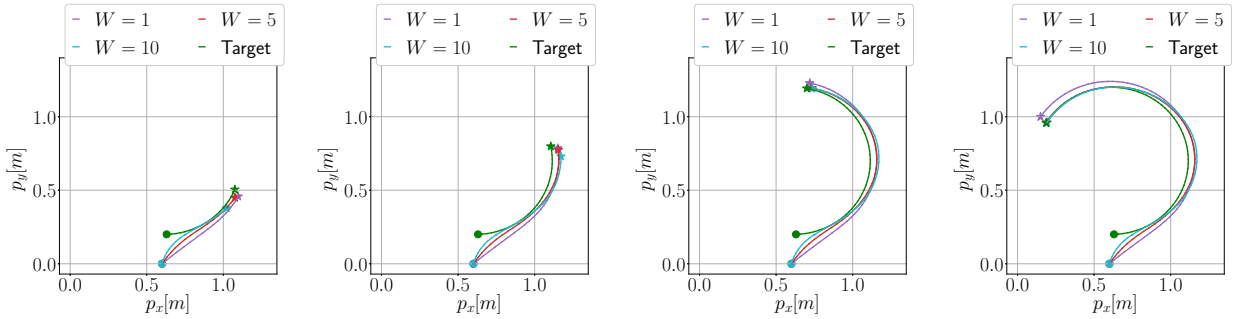
5.1.1. Target with a $V_T = 0$ path length

Given the dynamic regret analysis and the main result in Theorem 4.1, the smaller the path length V_T the easier the control task is for PLOT. To show its tracking performance for such a target, as well as to visualize the effect of the prediction horizon W on the control performance, we consider the circular target detailed in Example C.2 that has $V_T = 0$ path length. With this target revealed and measured online, as described in Section 2, we run PLOT repeatedly for various horizon lengths. The trajectory plots of the online target and the quadrotor are shown in Figure 1 for the first $T = 2, 3, 5$ and $T = 7$ seconds. The regret plots for the considered multiple horizon lengths are shown in Figure 2 for a simulation of $T = 200$ seconds. Firstly, both figures show that for too small W-s, PLOT exhibits a poorer tracking and regret performance due to the horizon truncation error. As expected from Corollary 4.2, given the nature of the reference target, PLOT with longer prediction horizons performs better by learning the reference dynamics online, achieving sublinear regret as verified by Figure 2. Additionally, the regret is increased for larger W-s only up to a certain saturated level. This behavior, as discussed in Section 4, is thanks to the multi-predictor setup of PLOT, making its regret scale with $\tilde{W}$, as opposed to linearly with W. See Appendix F.2.1 for further details.

5.1.2. Target with a $V_T = \mathcal{O}(\sqrt{T})$ Path Length

For more challenging targets, one can make use of Corollary 4.2 to tune for the best forgetting factor γ if the path length V_T is known. Here, we consider the case when only the order of the path length of a given reference is known, and, using Corollary 4.2, fix $\gamma_a = 1 - c_\gamma T^{-a}$, where $c_\gamma \in \mathbb{R}_+$ is a constant independent of V_T and T, while $a \in \mathbb{R}_+$ is such that $\gamma_a \in (0, 1]$ and is tuned based on V_T and T.

We fix a spiraling reference target with $V_T = \mathcal{O}(\sqrt{T})$, with exact dynamics detailed in Appendix F.2.2. This order of V_T results in the optimal value of $a = 0.25$ and a forgetting factor $\gamma_{0.25} = 0.78$ from Corollary 4.2, with $c_\gamma = 1.50$ obtained experimentally.



(a) $T = 2$ seconds (b) $T = 3$ seconds (c) $T = 5$ seconds (d) $T = 7$ seconds

Figure 1. Trajectory plots of a circular target with a $V_T = 0$ path length and the PLOT Algorithm for varying prediction horizon lengths, simulated for $T = 2, 3, 5$ and $T = 7$ seconds.

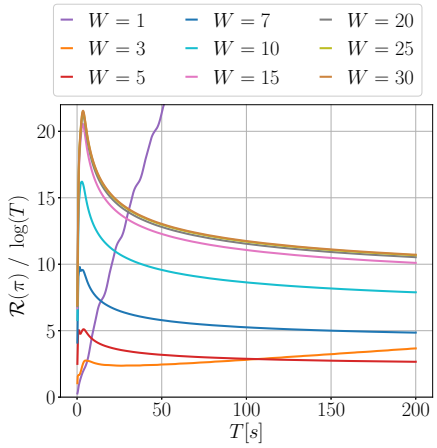


Figure 2. Log-normalized regret of the PLOT Algorithm with a range of prediction horizon lengths, simulated over a horizon of $T = 200$ seconds.

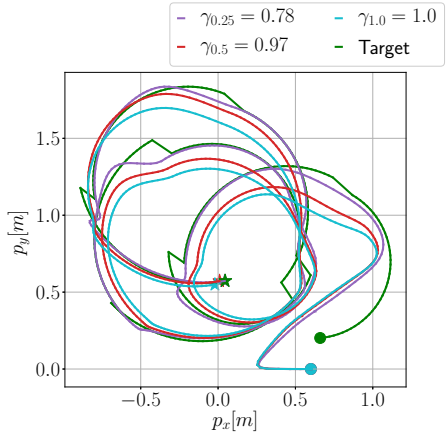


Figure 3. Trajectory plot of a spiral with a $V_T = \mathcal{O}(\sqrt{T})$ tracked with PLOT with a $W = 5$ and a range of values for γ.

PLOT's performance with three difference forgetting factors is shown in the trajectory plot in Figure 3 for $T = 40$ seconds, showing a better tracking performance for the regret-optimal $\gamma_{0.25} = 0.78$. To show that c_γ is independent of T, we fix, c_γ to the tuned value of 1.50 and perform experiments with varying horizon lengths from $T = 150$ to $T = 300$ seconds. For each T, PLOT is run with 4 different forgetting factors with $a = 0.1$, 0.25, 0.5 and 1 (corresponding to $\gamma = 1$), and the regret at the end of each experiment is shown in Figure 4. The results confirm that $\gamma_{0.25} = 1 - 1.5T^{-0.25}$ consistently outperforms other forgetting factors in terms of regret, as expected from Corollary 4.2. Further details on this example are provided in Appendix F.2.2.

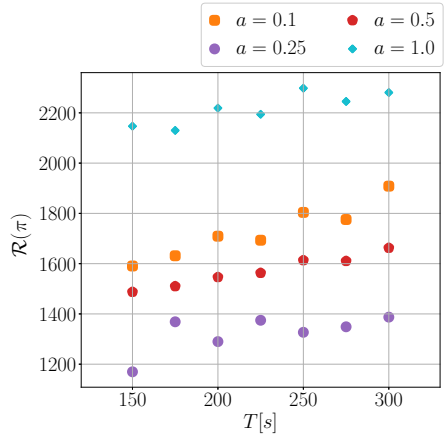


Figure 4. Regret of PLOT with varying $\gamma_a = 1 - c_\gamma T^{-a}$; $\gamma_{0.25}$ is regret-optimal based on Corollary 4.2.

5.1.3. Comparison with Other Online Control Methods

Next, we compare the dynamic regret of PLOT to other controllers. Namely, the disturbance-action policy (DAP) with memory proposed in (Agarwal et al., 2019a), the Riccatitron algorithm from (Foster & Simchowitz, 2020), the SS-OGD algorithm by (Karapetyan et al., 2023), the Follow

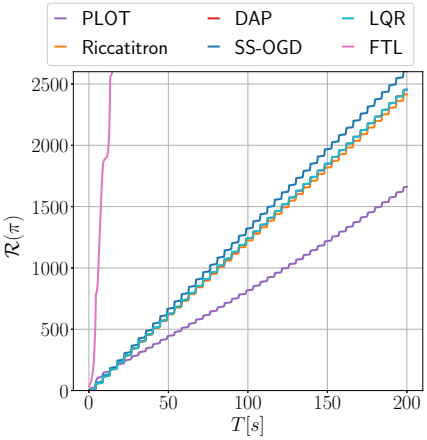


Figure 5. Dynamic Regret of online control algorithms applied to the online tracking problem.

the Leader (FTL) algorithm (Abbasi-Yadkori et al., 2014), as well as the Naive LQR controller that applies only an optimal state error feedback input without any affine term. The reference target is chosen to follow the dynamics

$$r_{t+1} = S_{t+1|t} = \begin{bmatrix} 1 & 0 & T_s & 0 \\ 0 & 1 & 0 & T_s \\ 0 & 0 & s_t \cos\theta_t & -s_t \sin\theta_t \\ 0 & 0 & s_t \sin\theta_t & s_t \cos\theta_t \end{bmatrix} r_t,$$

where $s_0 = 1, \theta_0 = 0.06$, $T_s = 0.1s$ and $s_k = -s_{k-1}$, $\theta_k = -0.99 \times \theta_{k-1}$ for every $k = \sqrt{T}$ given some T that we simulate $T = 200$ seconds.

Figure 5 shows the accumulated regret of all algorithms. For the given challenging example, PLOT outperforms the others. This is mainly because, unlike all the others, PLOT implements a dynamic approach with a forgetting factor adapting to the fast-changing reference on time. Moreover, it deploys an indirect approach of learning the dynamics of the reference and then using it in the control, as opposed to a direct approach of learning the affine control term, implemented by all the others apart from Naive LQR. For the given dynamic target the direct and static approaches produce a much smaller affine term compared to PLOT, leading to a performance very close to that of Naive LQR. The indirect approach of PLOT, however, may perform worse in cases where the reference targets lack dynamics and are mutually independent at each time step, as discussed in Remark 4.3. Further details on this, as well as the considered algorithms' implementation are provided in Appendix F.3.

5.2. Experimental Validation on Quadrotors

We validate the proposed algorithm on the Crazyflie 2.1 quadrotor. Due to the non-linear dynamics of the quadrotor and the noisy state measurements, the hardware experiments have tight requirements for the stability and robustness of the control algorithm. In the hardware experiments, we define the states and actions to be the same as in the simulations. The control inputs are sent to the quadrotor through a radio communicator from a centralized computer that receives state measurements through a local area network connection to the motion capture system, and runs the online controller. The virtual reference trajectory is generated online after the drone has successfully taken off and is at a predefined hovering position. The controller receives the target at a rate of 10 Hz during flight.

We implement PLOT for the linearized model of the drones derived in Appendix F with a fixed prediction horizon of $W = 5$ and a forgetting factor of $\gamma = 0.8$. Figure 6 shows the trajectory plots of PLOT and Naive LQR for a horizon of $T = 40$ seconds and an "infinity"-shaped reference target.. The Naive LQR controller exhibits a delayed tracking behavior as expected, while PLOT achieves a smaller tracking error. Further details on the practical implementation of the algorithm are provided in Appendix G.

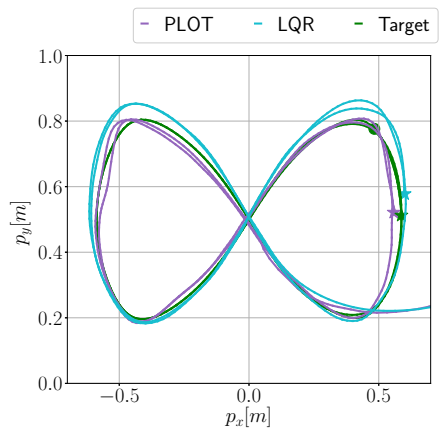


Figure 6. Trajectory plot of an "infinity"-shaped reference tracked by the drone with the PLOT and Naive LQR controllers.

6. Conclusion

We studied online tracking of unknown targets for linear quadratic problems and provided an algorithm with dynamic regret guarantees. By exploiting dynamic structure, we can obtain sharper guarantees compared to prior work that assumes no structure. It is open whether our regret analysis can be extended to *non-realizable* targets, that is, targets that do not obey (1) exactly for matrices that satisfy Assumption 2.2. While the guarantees for prediction still hold, it is open whether the guarantees for the controller are retained. A benefit of our implicit control paradigm is that we can decompose tracking into prediction and certainty equivalent control. This makes the approach promising for Model Predictive Control tracking applications. Another open question is whether the guarantees can be extended to output tracking or non-quadratic costs.

Acknowledgements

This work has been supported by the Swiss National Science Foundation under NCCR Automation (grant agreement 51NF40_180545), and by the European Research Council under the ERC Advanced grant agreement 787845 (OCAL).

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Abbasi-Yadkori, Y., Bartlett, P., and Kanade, V. Tracking adversarial targets. In *International Conference on Machine Learning*, pp. 369–377. PMLR, 2014.

Agarwal, N., Bullins, B., Hazan, E., Kakade, S., and Singh, K. Online control with adversarial disturbances. In *International Conference on Machine Learning*, pp. 111–119. PMLR, 2019a.

Agarwal, N., Hazan, E., and Singh, K. Logarithmic regret for online control. *Advances in Neural Information Processing Systems*, 32, 2019b.

Anava, O., Hazan, E., Mannor, S., and Shamir, O. Online learning for time series prediction. In *Conference on learning theory*, pp. 172–184. PMLR, 2013.

Anderson, B. and Moore, J. *Optimal Filtering*. Dover Publications, 2005.

Annaswamy, A. M. and Fradkov, A. L. A historical perspective of adaptive control and learning. *Annual Reviews in Control*, 52:18–41, 2021.

Åström, K. J., Borisson, U., Ljung, L., and Wittenmark, B. Theory and applications of self-tuning regulators. *Automatica*, 13(5):457–476, 1977.

Aucone, E., Kirchgeorg, S., Valentini, A., Pellissier, L., Deiner, K., and Mintchev, S. Drone-assisted collection of environmental DNA from tree branches for biodiversity monitoring. *Science Robotics*, 8(74):eadd5762, 2023.

Azoury, K. S. and Warmuth, M. K. Relative loss bounds for on-line density estimation with the exponential family of distributions. *Machine learning*, 43:211–246, 2001.

Baby, D. and Wang, Y.-X. Optimal dynamic regret in LQR control. *Advances in Neural Information Processing Systems*, 35:24879–24892, 2022.

Balta, E. C., Iannelli, A., Smith, R. S., and Lygeros, J. Regret analysis of online gradient descent-based iterative learning control with model mismatch. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pp. 1479–1484. IEEE, 2022.

Beuchat, P. N. N-rotor vehicles: modelling, control, and estimation. 2019.

Bitcraze. Crazyradio, 2023a. URL https://www.bitcraze.io/products/crazyradio-2-0/.

Bitcraze. Crazyflie 2.1, 2023b. URL https://www.bitcraze.io/products/crazyflie-2-1/.

Bristow, D. A., Tharayil, M., and Alleyne, A. G. A survey of iterative learning control. *IEEE control systems magazine*, 26(3):96–114, 2006.

Cervantes, I. and Alvarez-Ramirez, J. On the PID tracking control of robot manipulators. *Systems & control letters*, 42(1):37–46, 2001.

Cesa-Bianchi, N. and Lugosi, G. *Prediction, learning, and games*. Cambridge university press, 2006.

Chen, X. and Hazan, E. Black-box control for linear dynamical systems. In *Conference on Learning Theory*, pp. 1114–1143. PMLR, 2021.

Daponte, P., De Vito, L., Glielmo, L., Iannelli, L., Liuzza, D., Picariello, F., and Silano, G. A review on the use of drones for precision agriculture. 275(1):012022, 2019.

Ding, D., Yuan, J., and Jovanović, M. R. Discounted online Newton method for time-varying time series prediction. In *2021 American Control Conference (ACC)*, pp. 1547–1552. IEEE, 2021.

Dressel, L. and Kochenderfer, M. J. Hunting drones with other drones: Tracking a moving radio target. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 1905–1912. IEEE, 2019.

Foster, D. and Simchowitz, M. Logarithmic regret for adversarial online control. In *International Conference on Machine Learning*, pp. 3211–3221. PMLR, 2020.

Ghai, U., Lee, H., Singh, K., Zhang, C., and Zhang, Y. No-regret prediction in marginally stable systems. In *Conference on Learning Theory*, pp. 1714–1757. PMLR, 2020.

Goel, G. and Hassibi, B. The power of linear controllers in LQR control. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pp. 6652–6657. IEEE, 2022.

Goel, G. and Hassibi, B. Regret-optimal estimation and control. *IEEE Transactions on Automatic Control*, 68(5): 3041–3053, 2023.

Gradu, P., Hazan, E., and Minasyan, E. Adaptive regret for control of time-varying dynamics. In *Learning for Dynamics and Control Conference*, pp. 560–572. PMLR, 2023.

Guo, L. and Ljung, L. Performance analysis of general tracking algorithms. *IEEE Transactions on Automatic Control*, 40(8):1388–1402, 1995.

Hazan, E. and Seshadhri, C. Efficient learning algorithms for changing environments. In *Proceedings of the 26th annual international conference on machine learning*, pp. 393–400, 2009.

Hazan, E. and Singh, K. Introduction to online nonstochastic control. *arXiv preprint arXiv:2211.09619*, 2022.

Heirung, T. A. N., Ydstie, B. E., and Foss, B. Dual adaptive model predictive control. *Automatica*, 80:340–348, 2017.

Joulani, P., Gyorgy, A., and Szepesvári, C. Online learning under delayed feedback. In *International Conference on Machine Learning*, pp. 1453–1461. PMLR, 2013.

Kakade, S. M. On the sample complexity of reinforcement learning. 2003.

Karapetyan, A., Bolliger, D., Tsiamis, A., Balta, E. C., and Lygeros, J. Online linear quadratic tracking with regret guarantees. *IEEE Control Systems Letters (L-CSS)*, 2023.

Köhler, J., Müller, M. A., and Allgöwer, F. A nonlinear tracking model predictive control scheme for dynamic target signals. *Automatica*, 118:109030, 2020.

Kozdoba, M., Marecek, J., Tchrakian, T., and Mannor, S. On-line learning of linear dynamical systems: Exponential forgetting in Kalman filters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 4098–4105, 2019.

Li, Y., Chen, X., and Li, N. Online optimal control with linear dynamics and predictions: Algorithms and regret analysis. *Advances in Neural Information Processing Systems*, 32, 2019.

Li, Y., Das, S., and Li, N. Online optimal control with affine constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 8527–8537, 2021.

Ljung, L. and Gunnarsson, S. Adaptation and tracking in system identification—a survey. *Automatica*, 26(1):7–21, 1990.

Maeder, U., Borrelli, F., and Morari, M. Linear offset-free model predictive control. *Automatica*, 45(10):2214–2222, 2009.

Mania, H., Tu, S., and Recht, B. Certainty equivalence is efficient for linear quadratic control. *Advances in Neural Information Processing Systems*, 32, 2019.

Minasyan, E., Gradu, P., Simchowitz, M., and Hazan, E. Online control of unknown time-varying dynamical systems. *Advances in Neural Information Processing Systems*, 34: 15934–15945, 2021.

Modares, H. and Lewis, F. L. Linear quadratic tracking control of partially-unknown continuous-time systems using reinforcement learning. *IEEE Transactions on Automatic control*, 59(11):3051–3056, 2014.

Nikiforov, V. and Gerasimov, D. *Adaptive regulation: reference tracking and disturbance rejection*, volume 491. Springer Nature, 2022.

Niknejad, N., Yaghmaie, F. A., and Modares, H. Online reference tracking for linear systems with unknown dynamics and unknown disturbances. *Transactions on Machine Learning Research*, 2023.

Nonhoff, M., Köhler, J., and Müller, M. A. Online convex optimization for constrained control of linear systems using a reference governor. *IFAC-PapersOnLine*, 56(2): 2570–2575, 2023.

Owens, D. H. and Hätönen, J. Iterative learning control—an optimization paradigm. *Annual reviews in control*, 29(1): 57–70, 2005.

Pan, Y., Li, X., and Yu, H. Efficient PID tracking control of robotic manipulators driven by compliant actuators. *IEEE Transactions on Control Systems Technology*, 27 (2):915–922, 2018.

Pannocchia, G. and Bemporad, A. Combined design of disturbance model and observer for offset-free model predictive control. *IEEE Transactions on Automatic Control*, 52(6):1048–1053, 2007.

Parsi, A., Iannelli, A., and Smith, R. S. An explicit dual control approach for constrained reference tracking of uncertain linear systems. *IEEE Transactions on Automatic Control*, 2022.

Peterson, B. and Narendra, K. Bounded error adaptive control. *IEEE Transactions on Automatic Control*, 27(6): 1161–1168, 1982.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A. Y., et al. ROS: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, pp. 5. Kobe, Japan, 2009.

Schoellig, A., Siegel, H., Augugliaro, F., and D'Andrea, R. *So You Think You Can Dance? Rhythmic Flight Performances with Quadrocopters*, pp. 73–105. 01 2014. ISBN 978-3-319-03903-9. doi: 10.1007/978-3-319-03904-6_4.

Simchowitz, M. Making non-stochastic control (almost) as easy as stochastic. *Advances in Neural Information Processing Systems*, 33:18318–18329, 2020.

Simchowitz, M. and Foster, D. Naive exploration is optimal for online LQR. In *International Conference on Machine Learning*, pp. 8937–8948. PMLR, 2020.

Snyder, D., Booker, M., Simon, N., Xia, W., Suo, D., Hazan, E., and Majumdar, A. Online learning for obstacle avoidance. In *Conference on Robot Learning*, pp. 2926–2954. PMLR, 2023.

Soloperto, R., Köhler, J., Müller, M. A., and Allgöwer, F. Dual adaptive MPC for output tracking of linear systems. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pp. 1377–1382. IEEE, 2019.

Suo, D., Agarwal, N., Xia, W., Chen, X., Ghai, U., Yu, A., Gradu, P., Singh, K., Zhang, C., Minasyan, E., et al. Machine learning for mechanical ventilation control. *arXiv preprint arXiv:2102.06779*, 2021.

Tsiamis, A. and Pappas, G. J. Online learning of the Kalman filter with logarithmic regret. *IEEE Transactions on Automatic Control*, 68(5):2774–2789, 2022.

Vamvoudakis, K. G. Optimal trajectory output tracking control with a Q-learning algorithm. In *2016 American Control Conference (ACC)*, pp. 5752–5757. IEEE, 2016.

Yu, C., Shi, G., Chung, S.-J., Yue, Y., and Wierman, A. The power of predictions in online control. *Advances in Neural Information Processing Systems*, 33:1994–2004, 2020.

Yuan, J. and Lamperski, A. Trading-off static and dynamic regret in online least-squares and beyond. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 6712–6719, 2020.

Zhang, K., Hu, B., and Basar, T. Policy optimization for $\mathcal{H}_2$ linear control with $\mathcal{H}_\infty$ robustness guarantee: Implicit regularization and global convergence. *SIAM Journal on Control and Optimization*, 59(6):4081–4109, 2021a.

Zhang, R., Li, Y., and Li, N. On the regret analysis of online LQR control with predictions. In *2021 American Control Conference (ACC)*, pp. 697–703. IEEE, 2021b.

Zhang, Z., Cutkosky, A., and Paschalidis, I. Adversarial tracking control via strongly adaptive online learning with memory. In *International Conference on Artificial Intelligence and Statistics*, pp. 8458–8492. PMLR, 2022.

Zhao, P., Wang, Y.-X., and Zhou, Z.-H. Non-stationary online learning with memory and non-stochastic control. In *International Conference on Artificial Intelligence and Statistics*, pp. 2101–2133. PMLR, 2022.

Zhou, H., Song, Y., and Tzoumas, V. Safe non-stochastic control of control-affine systems: An online convex optimization approach. *IEEE Robotics and Automation Letters*, 2023.

Ziemann, I. and Sandberg, H. Regret lower bounds for learning Linear Quadratic Gaussian systems. *arXiv preprint arXiv:2201.01680*, 2022.

Zinkevich, M. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th international conference on machine learning (icml-03)*, pp. 928–936, 2003.

Organization

In Appendix B we discuss properties of LQT control and the equivalence to the problem of LQR with disturbances. In Appendix C, we discuss issues regarding the dynamic representation of the targets. We present analytical expressions for the multi-step ahead predictors. We also discuss how the dynamic structure of the targets can lead to improved learning performance. Finally, we provide an expression for the total variation norm of multi-step ahead predictors in terms of the total variation of the system matrices S_t. The proof of Theorem 4.5 can be found in Appendix D, where we also provide more details for the prediction problem. The proof of Theorem 4.1 and Corollary 4.2 can be found in Appendix E. Simulations visualizing the performance of PLOT, its hyperparemeter tuning based on the regret analysis, as well as comparison with other benchmark algorithms are provided in Appendix F. Appendix G provides the technical details of the implementation of PLOT on the Crazyflie quadrotors and its tracking performance on those. The code for the simulation and hardware experiments is provided in `https://gitlab.nccr-automation.ch/akarapetyan/plot`.

A. Additional Related Work

Online and non-stochastic control. Online and non-stochastic control techniques have been applied to systems with known (Agarwal et al., 2019a) as well as unknown system dynamics (Simchowitz, 2020). In this paper, we focus on the case of known *system* dynamics, where the main purpose of the online controller is to react to (non-stochastic) *unknown* disturbances/targets.[2] In this setting, Agarwal et al. (2019a) achieved $\mathcal{O}(\sqrt{T})$ regret for general convex stage cost functions under a disturbance affine feedback policy. In the case of quadratic costs, and more generally strongly convex stage costs[3], the guarantees were improved to logarithmic (Foster & Simchowitz, 2020; Simchowitz, 2020; Agarwal et al., 2019b). The aforementioned works consider static problems in the sense that they compete against the best stationary policy. The dynamic setting was studied in (Zhao et al., 2022), where dynamic regret guarantees of the order of $\mathcal{O}(\sqrt{TV_T})$ for general convex costs were derived for the first time, using gradient-based (first order) base learners. In (Baby & Wang, 2022) dynamic regret guarantees for the quadratic cost case (LQR) were derived using a variant of the Follow the Leading History (FLH) algorithm. It was proved that the dynamic regret with respect to any dynamic disturbance affine policy is upper bounded by $T^{1/3}V_T^{2/3}$, where V_T is the total variation of the parameters of the policy. Learning disturbance affine feedback policies follows the direct online control paradigm since we adapt the policy parameters directly.

In this paper, we study implicit online control architectures, where we decouple online prediction of disturbances/targets from control design. Several works have studied the effect of predictions on receding horizon control (Yu et al., 2020; Li et al., 2019; Zhang et al., 2021b). However, either no method of prediction is provided or the dynamic regret guarantees are suboptimal in our setting. In particular, the regret guarantees scale with the total variation of the disturbance/target itself $\sum_{t=0}^{T-1} \|r_{t+1} - r_t\|$ (Li et al., 2019), which is appropriate for unstructured or slowly-varying targets but is suboptimal in the case of targets with dynamic structure; there exist targets for which the total variation of their state r_t is linear, while the total variation of the target dynamics S_t is zero–see Appendix C.1.

Abbasi-Yadkori et al. (2014) introduced the problem of non-stochastic control for target tracking, where the target can be adversarial. Logarithmic regret guarantees are provided for learning affine control policies with constant affine terms. However, policies with constant affine terms can only track static points as targets; they are not rich enough to track trajectories that are not points and, in general, time-varying targets. Recently, Niknejad et al. (2023) studied online learning for target tracking in the case of unknown system dynamics and unknown time-invariant target dynamics. Our setting is different since we have known dynamics for the system, time-varying dynamics for the target, and we use dynamic regret. Their static regret guarantee of $T^{2/3}$ is conservative for our setting, where we can obtain logarithmic regret in the static case, however, it applies to general convex costs and unknown dynamics. In (Karapetyan et al., 2023), dynamic regret guarantees are provided for target tracking, under a first order method which treats the targets as slowly varying or steady-state. Similar to (Li et al., 2019), the dynamic regret depends on the total variation of the target state which can be suboptimal in the case of target with dynamics. Zhang et al. (2022) proved $\tilde{\mathcal{O}}(\sqrt{|\mathcal{I}|})$ (for any interval $\mathcal{I}$) adaptive regret guarantees for tracking in the case of convex costs. Their setting is different since no dynamic structure is assumed for the targets, the comparator class is controllers which are constant over the interval of interest, while we focus on square losses and dynamic regret.

[2]Note that as shown in many works, see (Foster & Simchowitz, 2020; Karapetyan et al., 2023), the problem of tracking is equivalent under certain conditions to the problem of disturbance rejection. Hence, for the purpose of this literature review, we can use disturbance/target interchangeably.

[3]Certain additional conditions are required.

Although related, our setting is different from (Gradu et al., 2023) and (Minasyan et al., 2021) where the system itself is linear time varying, there is no target, and adaptive regret is considered as a performance metric. Instead, here, the system's dynamics are time invariant, the target's dynamics are unknown and time varying, and we consider dynamic regret. Safe online non-stochastic control subject to constraints has also been studied before (Zhou et al., 2023; Nonhoff et al., 2023; Li et al., 2021). Dealing with constraints is a more challenging problem in general. In many cases, the analysis is restricted to the static case (Li et al., 2021), requires relaxing the notion of policy regret (Zhou et al., 2023) or the path length scales with the target state (Nonhoff et al., 2023), which is not always optimal in our setting as we discuss above.

Finally, we note that when the *system* dynamics are *unknown*, it is no longer possible to achieve logarithmic regret in the general case, even under stochastic disturbances (Simchowitz & Foster, 2020; Ziemann & Sandberg, 2022). It is possible to achieve $\tilde{\mathcal{O}}(\sqrt{T})$ regret (Chen & Hazan, 2021; Simchowitz, 2020; Mania et al., 2019) instead.

Prediction and online least squares. Online linear regression and the least squares method, in particular, have been studied extensively before (Azoury & Warmuth, 2001; Cesa-Bianchi & Lugosi, 2006). Anava et al. (2013) provided logarithmic regret guarantees for online prediction of auto-regressive (AR) and auto-regressive with moving average (ARMA) systems using the online Newton step (ONS) method. The result holds for exp-concave prediction loss functions, which includes the square prediction loss. Similar guarantees were proved for partially observed stochastic linear systems in the case of Kalman filtering (Ghai et al., 2020; Tsiamis & Pappas, 2022). Under first-order methods, the regret guarantees become $\mathcal{O}(\sqrt{T})$ (Anava et al., 2013; Kozdoba et al., 2019), however, first-order methods are also known to adapt to changes (Zinkevich, 2003; Hazan & Seshadhri, 2009). On the contrary, the standard versions of second-order methods like RLS or ONS are not adequate for time-varying settings, where adaptation is needed. Another challenge is that in the case of multi-step ahead prediction, these methods should be adapted to deal with delayed learning feedback.

To solve the former issue, one way is to employ multiple learners along with a meta-algorithm that chooses the best expert online, also known as the Follow the Leading History (FLH) algorithm (Hazan & Seshadhri, 2009; Baby & Wang, 2022). Another option is to employ exponential forgetting (Yuan & Lamperski, 2020; Ding et al., 2021). In this paper, we follow the second approach since it has been standard practice in the adaptive control literature (Guo & Ljung, 1995; Åström et al., 1977). Moreover, it has a simple implementation for real-time applications, like drone control.

To solve the latter issue, we employ the standard technique of Joulani et al. (2013) that decomposes the problem of learning with delays of duration k into k non-overlapping and non-delayed online learning instances.

Control Literature. Common feedback control policies, such as the proportional-integral-derivative (PID) controllers find extensive use for tracking problems in practice (Pan et al., 2018; Cervantes & Alvarez-Ramirez, 2001). For example, integral control has been used to track constant unknown offsets. However, they typically require fine-tuning and their tracking performance for unknown time-varying trajectories is not guaranteed in general. Tracking of known reference trajectories with uncertain repetitive dynamics is studied in the context of iterative learning control (Bristow et al., 2006; Owens & Hätönen, 2005) with online extensions studying dynamic regret (Balta et al., 2022). However, the methods do not extend to non-repetitive setting of LQT with unknown time-varying references.

Predictive controllers such as model predictive control (MPC) have also been used to track constant references, see, for example, offset-free MPC (Pannocchia & Bemporad, 2007; Maeder et al., 2009). Various tracking MPC methods are proposed for time-varying references, e.g., (Köhler et al., 2020). However, unless we have an accurate prediction model for the reference trajectory, the guarantees do not extend to more general sequentially revealed unknown trajectories. Dual MPC methods with active exploration are proposed for tracking applications and uncertain dynamics (Heirung et al., 2017; Soloperto et al., 2019; Parsi et al., 2022). However, transient performance, regret guarantees, or the case of unknown time-varying references are not studied.

The LQT problem with unknown targets has been studied before from the point of view of control theory and adaptive control (Vamvoudakis, 2016; Modares & Lewis, 2014; Peterson & Narendra, 1982). Typically, the goal is to prove asymptotic convergence or boundedness of tracking error by appealing to Lyapunov stability theory. Most results assume time-invariant target dynamics. Tracking time-varying targets is possible using adaptive control techniques, under the assumption that the parameters of the target evolve like a random walk (Guo & Ljung, 1995). However, this assumption excludes adversarial targets. By using regret as a metric, we can obtain non-asymptotic guarantees that also capture adversarial, non-stochastic behaviors.

Finally, an alternative approach is to employ robust control techniques to account for worst-case disturbances/tracking

errors. Notable approaches include the celebrated $\mathcal{H}_\infty$ control approach or mixed $\mathcal{H}_2 - \mathcal{H}_\infty$ control (Zhang et al., 2021a). Recently, robust controllers inspired by the notion of regret were designed (Goel & Hassibi, 2023).

B. Linear Quadratic Control Properties

In this section, we revisit some properties of the LQT controller, like (internal) stability. We also discuss how to relax Assumption 2.3. Finally, we discuss that LQT is equivalent to LQR control with disturbances.

B.1. LQT properties

Let us first recall the notions of stabilizability and detectability. A pair of system and input matrices (A, B) is *stabilizable* if and only if there exists a linear feedback gain $K_0 \in \mathbb{R}^{m\times n}$ such that $(A - BK_0)$ has all eigenvalues strictly inside the unit circle.

Let us now introduce the following proposition, which shows that the closed loop matrix $A - BK$ under the feedback gain K defined in (8) is stable, that is, all of its eigenvalues are inside the unit circle. It also shows that the feedforward gains $K_t, \forall t \leq T$ defined in (9) decay exponentially fast as t increases. Under the optimal control law (7), we do not have stability in the classical sense; since the target state can be arbitrary the tracking error might not remain close to the origin. However, we have internal stability since all signals remain bounded as long as r_t is bounded. A more general case with time-variant costs of the proposition is proven in Corollary 1 of (Zhang et al., 2021b).

Proposition B.1 (Stability (Zhang et al., 2021b))**.** *Let Assumption 2.3 be in effect. Recall the definition of the feedback and feedforward gains in* (8)*,* (9)*. For all $t \in \mathbb{N}$, the closed loop matrix $(A - BK)$ satisfies*

$$\|(A - BK)^t\| \leq \sqrt{\frac{\lambda_{max}(X)}{\lambda_{min}(X)}}\rho^t,\ \rho = \sqrt{1 - \frac{\lambda_{min}(Q)}{\lambda_{max}(X)}} \in (0, 1)$$

where $\lambda_{min}(\cdot)$ and $\lambda_{\max}(\cdot)$ denote the minimum and maximum eigenvalue respectively, X is defined in (6)*. As a result, the coefficient matrices K_t defined in* (9) *satisfy*

$$\|K_t\| \leq c_0\rho^t$$

for $c_0 = \frac{\|B\|\lambda_{max}(X)}{\lambda_{min}(R)}\sqrt{\frac{\lambda_{max}(X)}{\lambda_{min}(X)}}$.

Assumption 2.3 enables us to quantify the constants c_0, ρ. In fact, we can relax Assumption 2.3 and replace it with the following assumption.

Assumption 2.3′. The pair (A, B) is stabilizable, the pair (Q, A) is detectable, Q is positive semi-definite, and R is symmetric and positive definite.

Note that a pair (Q, A) is detectable if and only if $(A^\top, Q^\top)$ is stabilizable. Under the above assumption, we still retain stability of $(A - BK)$ (Anderson & Moore, 2005). The only difference is that we do not have an explicit characterization of ρ anymore as in Proposition B.1.

B.2. LQT as LQR with disturbances

We can recast the LQT problem into an LQR problem with adversarial disturbances, see, for example, (Karapetyan et al., 2023). By redefining the tracking error as the system state $e_t = x_t - r_t$ and encoding the time-varying target states as the disturbance $w_t = Ar_t - r_{t+1}$, the resulting LQR problem is given as,

$$\begin{aligned} \min_{u_{0:T-1}} \sum_{t=0}^{T-1} \left(\|e_t\|_Q^2 + \|u_t\|_R^2\right) + \|e_T\|_X^2 \\ \text{s.t.} \quad e_{t+1} = Ae_t + Bu_t + w_t,\ \forall t = 0, ..., T-1. \end{aligned} \tag{15}$$

Therefore, we can treat the online LQT problem as an online, non-stochastic LQR problem with adversarial disturbances. The converse is also true if we set $r_t = w_{t-1} - Ar_{t-1}$ and initialize $r_0 = 0$.

C. Autoregressive Systems

In this section, we present certain properties of autoregressive systems

$$r_{t+1} = S_{t+1} z_t,$$

where $z_t = \begin{bmatrix} r_t^\top & r_{t-1}^\top & \cdots & r_{t-p}^\top \end{bmatrix}^\top$, for some past horizon $p > 0$. Note that autoregressive systems can also be described by state-space equations, viewing z_t as a non-minimal state representation. Define the extended matrices

$$\mathcal{A}_{t+1} \triangleq \begin{bmatrix} & & S_{t+1} & & \\ \hline \mathbf{I}_n & 0 & \cdots & 0 & 0 \\ 0 & \mathbf{I}_n & \cdots & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & \mathbf{I}_n & 0 \end{bmatrix}, \mathcal{B} \triangleq \begin{bmatrix} \mathbf{I}_n \\ \hline 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{16}$$

Then r_t can be though of as the output of the following state-space system

$$\begin{aligned} z_{t+1} &= \mathcal{A}_{t+1} z_t \\ r_t &= \mathcal{B}^\top z_t \end{aligned} \tag{17}$$

Our results extend directly to auto-regressive dynamics with additional exogenous variables (ARX models)

$$r_{t+1} = S_t z_t + v_t.$$

It is sufficient to extend the regressor vector to contain 1 as the last element

$$\tilde{z}_t = \begin{bmatrix} z_t^\top & 1 \end{bmatrix}^\top.$$

We can then extend S_t accordingly

$$\tilde{S}_t = \begin{bmatrix} S_t & v_t \end{bmatrix}.$$

C.1. Representation and learning complexity

Auto-regressive dynamics can cover many commonly encountered trajectories. Two elementary examples include constant velocity and circular targets.

Example C.1 (Constant Velocity Target)**.** *Let z_t, y_t denote positions in a 2D horizontal plane with $\dot{z}_t, \dot{y}_t$ the respective velocities and the target state as $r_t = [z_t, y_t, \dot{z}_t, \dot{y}_t]^T$. Let T_s be the sampling time for discretizing the target dynamics. Then, a target with constant velocity can be represented by*

$$r_{t+1} = S r_t, \quad S = \begin{bmatrix} 1 & 0 & T_s & 0 \\ 0 & 1 & 0 & T_s \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Note that there might be multiple representations. We could also use the second order representation

$$r_{t+1} = S^{[1]} r_t + S^{[2]} r_{t-1}, \quad S^{[1]} = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, S^{[2]} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

using the fact that $y_t - y_{t-1} = y_{t+1} - y_t$ (and similarly for z_t) under constant velocity.

Example C.2 (Circular Target with Constant Speed)**.** *Let z_t, y_t denote positions in a 2D horizontal plane with $\dot{z}_t, \dot{y}_t$ the respective velocities and the target state as $r_t = [z_t, y_t, \dot{z}_t, \dot{y}_t]^T$. Let T_s be the sampling time for discretizing the target dynamics. Then $\forall k \in \mathbb{N}^+, t = 1, ..., T-k$, the circular target with constant speed can be represented by*

$$r_{t+1} = S r_t, \quad S = \begin{bmatrix} 1 & 0 & T_s & 0 \\ 0 & 1 & 0 & T_s \\ 0 & 0 & \cos\theta & -\sin\theta \\ 0 & 0 & \sin\theta & \cos\theta \end{bmatrix},$$

where we used Euler discretization.

By allowing time-varying dynamics, we can also capture switching patterns, e.g. waypoint tracking, switching orientation, etc. In the case of ARX models $r_{t+1} = S_t z_t + v_t$, we can capture targets that behave like control systems themselves, e.g., this representation could be used for controlling a drone in order to track another drone.

In many previous works (Karapetyan et al., 2023; Li et al., 2019; Nonhoff et al., 2023), learning complexity is captured by the total variation (path length) of the target itself

$$L_T = \sum_{t=0}^{T-1} \|r_{t+1} - r_t\|.$$

In contrast, here, we capture learning complexity by the total variation of the target dynamics

$$V_T = \sum_{t=0}^{T-1} \|S_{t+1} - S_t\|_F.$$

In the case of ARX models we just replace S_t with $\tilde{S}_t$. As shown in (Li et al., 2019, Th. 3), in the case of unstructured targets, e.g. randomly generated r_t, the former notion of complexity is optimal. However, we argue that in the case of dynamic structure, the former notion of complexity might be suboptimal. Consider the circular target or the linear target example. The total variation of the target state is linear with T since r_t is constantly changing: $L_T = \Omega(T)$. However, the total variation of the target dynamics is zero $V_T = 0$. Using the PLOT algorithm would give us logarithmic regret in this case, while previous methods would give us linear regret.

We stress that every target can be represented trivially by ARX models. We can just set $S_t = 0$, $v_t = r_t$. In this case, the total variation of $\tilde{S}_t = \begin{bmatrix} S_t & v_t \end{bmatrix}$ is equal to L_T, as in prior work. However, the representation is non-unique in general. In many cases, the underlying dynamic structure will imply that a lower complexity representation exists, e.g., see Examples C.1, C.2. If the targets have no dynamic structure, then, our current setting still captures this case; the regret of PLOT will just be larger, i.e., of the same order as L_T. Hence, our dynamic regret bounds supersede the ones in prior work.

C.2. Multi-step ahead dynamics expressions

Given the state representation (17), we can now represent the multi-step ahead recursions in a compact way. Let

$$\Phi_{t+i|t} \triangleq \mathcal{A}_{t+i} \cdots \mathcal{A}_{t+1},\ \Phi_{t|t} = \mathbf{I}_{np}$$

be the transition matrix of system (17). Then, we obtain $k-$step ahead recursions of the form

$$r_{t+k} = S_{t+k|t} z_t,$$

where the matrices $S_{t+k|t}$ are given by

$$S_{t+k|t} = \mathcal{B}^\top \Phi_{t+i|t} \tag{18}$$

By definition, if $S_{0,\dots,T}$ satisfies Assumption 2.2, then $\|\mathcal{B}^\top \Phi_{t+i|t}\| \leq M$.

C.3. Perturbation Analysis

Let $S_0, \dots, S_T$ be any sequence that satisfies Assumption 2.2. We will now show that we can upper-bound the total variation norm of the k-step ahead matrices $V_T^k \triangleq \sum_{t=k}^{T-k} \|S_{t+k|t} - S_{t|t-k}\|$ in terms of the total variation norm of the one-step ahead matrices $V_T \triangleq \sum_{t=1}^{T} \|S_t - S_{t-1}\|_F$.

Lemma C.3. *Recall that M is the upper bound on matrices $S_{t+k|t}$ and p is the past horizon (memory) of the auto-regressive dynamics. Let $\tilde{M} = \max\{M, 1\}$. The following inequality is true*

$$V_T^k \leq \sqrt{p} k^2 \tilde{M}^2 V_T.$$

Proof. By adding and subtracting terms $S_{t+j|t+j-k}$, $j = 1, \dots, k-1$ and the triangle inequality, we obtain

$$\|S_{t+k|t} - S_{t|t-k}\| \leq \sum_{j=t}^{t+k-1} \|S_{j+1|j+1-k} - S_{j|j-k}\|.$$

Hence, we get

$$V_T^k \leq k \sum_{j=k-1}^{T-1} \|S_{j+1|j+1-k} - S_{j|j-k}\|. \tag{19}$$

Let us now analyze

$$\Delta_{k,t} \triangleq S_{t+k+1|t+1} - S_{t+k|t} = \mathcal{B}^\top(\Phi_{t+k+1|t+1} - \Phi_{t+k|t}).$$

Adding and subtracting $\Phi_{t+k|t+j-1}\Phi_{t+j|t+1}$, for $j = k, k-1, \ldots, 2$ we obtain

$$\begin{aligned}\Delta_{k,t} = \mathcal{B}^\top(\mathcal{A}_{t+k+1} - \mathcal{A}_{t+k})\Phi_{t+k|t+1} + \mathcal{B}^\top\Phi_{t+k|t+k-1}(\mathcal{A}_{t+k} - \mathcal{A}_{t+k-1})\Phi_{t+k-1|t+1} + \ldots \\ + \mathcal{B}^\top\Phi_{t+k|t+1}(\mathcal{A}_{t+2} - \mathcal{A}_{t+1}).\end{aligned}$$

By the triangle inequality

$$\begin{aligned}\|\Delta_{k,t}\| \leq \|\mathcal{B}^\top\|\|\mathcal{A}_{t+k+1} - \mathcal{A}_{t+k}\|\|\Phi_{t+k|t+1}\| + \|\mathcal{B}^\top\Phi_{t+k|t+k-1}\|\|\mathcal{A}_{t+k} - \mathcal{A}_{t+k-1}\|\|\Phi_{t+k-1|t+1}\| + \ldots \\ + \|\mathcal{B}^\top\Phi_{t+k|t+1}\|\|\mathcal{A}_{t+2} - \mathcal{A}_{t+1}\|.\end{aligned}$$

By definition, all terms $\mathcal{B}^\top\Phi_{t+k|t+j-1}$, $j = k, k-1, \ldots, 2$ are bounded by $\max\{M, 1\}$. Meanwhile, the error terms

$$\mathcal{A}_{t+j} - \mathcal{A}_{t+j-1} = \begin{bmatrix} S_{t+j} - S_{t+j-1} \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

can be bounded by $\|S_{t+j} - S_{t+j-1}\|$, for $j = k, \ldots, 2$. Finally, we need to bound the norm of the transition matrices. Observe that by definition

$$\Phi_{t+j|t} = \begin{bmatrix} S_{t+j|t} \\ S_{t+j-1|t} \\ \vdots \\ S_{t+j-p+1|t} \end{bmatrix}, \text{ if } j \geq p, \; \Phi_{t+j|t} = \left[\begin{array}{cc} S_{t+j|t} & \\ S_{t+j-1|t} & \\ \vdots & \\ S_{t+1|t} & \\ \hline \mathbf{I}_{n(p-j)} & \mathbf{0}_{n(p-j)\times nj} \end{array}\right], \text{ if } j < p,$$

where $\mathbf{0}_{q_1\times q_2}$ denotes the zero matrix of dimensions $q_1 \times q_2$. As a result $\|\Phi_{t+j|t}\| \leq \sqrt{p}\tilde{M}$ Putting everything together, we obtain

$$\|\Delta_{k,t}\| \leq \sqrt{p}\tilde{M}^2 \sum_{j=2}^{k} \|S_{t+j} - S_{t+j-1}\|.$$

The results follow from the above inequality and (19). □

D. Proofs for Prediction

In this section, we prove Theorem 4.5. We provide regret upper bounds for prediction in terms of the total variation norm of the dynamics V_T. Note that the prediction guarantees hold for any sequence $r_{0:T}$ that satisfies Assumptions 2.1, 2.2. First, we show the result for one-step ahead prediction $k = 1$. Then, we generalize to $k > 0$.

D.1. Regret guarantees for one-step ahead prediction

Consider the one-step ahead prediction problem. For simplicity, define $\hat{S}_{t|t-1} = \hat{S}_t$ and $P_{t|t-1} = P_t$. Then Algorithm 1 is equivalent to Algorithm 3. Let r_t, for $t \leq T$ be any arbitrary target sequence satisfying Assumption 2.1. The target sequence may not necessarily satisfy (3). Let $S_0, \ldots, S_T$ be any "comparator" sequence that satisfies Assumption 2.2. Let $\hat{S}_t$ be the sequence generated by the RLS algorithm. Then the dynamic regret of the one-step ahead predictor versus the sequence $S_0, \ldots, S_T$ is defined as

$$\mathcal{R}_{\text{pred}}^{(1)}(S_{0:T}) \triangleq \frac{1}{2}\sum_{t=0}^{T} \|\hat{S}_t z_{t-1} - r_t\|^2 - \frac{1}{2}\sum_{t=0}^{T} \|S_t z_{t-1} - r_t\|^2. \tag{20}$$

Algorithm 3 Recursive Least Squares, for $k = 1$

Require: forgetting factor $\gamma \in (0, 1)$, initial regularization ε
Initialize: $P_{-1} = \varepsilon I_{np}$, $\hat{S}_0 \in \mathcal{S}$;
for $t = 0, \dots, T$ **do**
Predict $\hat{r}_t = \hat{S}_t z_{t-1}$ and incur loss $\|r_t - \hat{S}_t z_{t-1}\|^2$
Update $P_t = \gamma P_{t-1} + z_{t-1} z_{t-1}^\top$;
Update $\hat{S}_{t+1} = \Pi_{\mathcal{S}}^{P_t}(\hat{S}_t + (r_t - \hat{S}_t z_{t-1}) z_{t-1}^\top P_t^{-1})$,
where $\Pi_{\mathcal{S}}^{P_t}(Y) \triangleq \arg\min_{S \in \mathcal{S}} \|S - Y\|_{F,P_t}$;
end for

Note that if the sequence r_t satisfies the autoregressive dynamics (3) and we choose the comparator sequence to coincide with the true dynamics then the regret reduces to

$$\mathcal{R}_{\text{pred}}^{(1)}(S_{0:T}) \triangleq \frac{1}{2} \sum_{t=0}^{T} \|\hat{S}_t z_{t-1} - r_t\|^2.$$

Note that we define the projection operator $\Pi_{\mathcal{S}}^{P_t}(Y)$ as

$$\Pi_{\mathcal{S}}^{P_t}(Y) \triangleq \arg\min_{S \in \mathcal{S}} \|S - Y\|_{F,P_t}, \tag{21}$$

where P_t should be symmetric positive definite. Recall that the weighted Frobenius norm is given by

$$\|S\|_{F,P_t}^2 = \text{tr}(S P_t S^\top).$$

To bound the regret of RLS, we adapt the proof of (Yuan & Lamperski, 2020) while keeping track of all quantities of interest, e.g. the system dimension, logarithmic terms etc., and while working with matrices instead of vectors.

Theorem D.1. *Let r_t, for $t \leq T$ be any target trajectory that satisfies Assumption 2.1. Let $S_0, \dots, S_T$ be any sequence that satisfies Assumption 2.2. Define the path length as $V_T = \sum_{t=0}^{T-1} \|S_{t+1} - S_t\|_F$. The regret of the RLS algorithm (Algorithm 3) for the one-step ahead prediction problem is upper bounded by*

$$\mathcal{R}_{\text{pred}}^{(1)}(S_{0:T}) \leq \frac{\beta_1}{1-\gamma} V_T + \beta_2 T \log \frac{1}{\gamma} + \beta_3 \log \frac{1}{1-\gamma} + \beta_4 \tag{22}$$

with the constants

$$\begin{aligned} \beta_1 = 2\sqrt{n} M \left(\frac{\varepsilon + p D_r^2}{1-\gamma} \right), \ \beta_2 = np(1 + \sqrt{p} M)^2 D_r^2, \\ \beta_3 = \beta_2, \ \beta_4 = 2\gamma \varepsilon n M^2 + \beta_2 \log \frac{\varepsilon + p D_r^2}{\varepsilon}. \end{aligned} \tag{23}$$

Proof. Using the law of cosines identity $\|b - a\|^2 - \|b - c\|^2 = -\|a - c\|^2 - 2(a - c)^\top (b - a)$, we obtain

$$\begin{aligned} \mathcal{R}_{\text{pred}}(S_{0:T}) &= \sum_{t=0}^{T} -\frac{1}{2} z_{t-1}^\top (\hat{S}_t - S_t)^\top (\hat{S}_t - S_t) z_{t-1} - (r_t - \hat{S}_t z_{t-1})^\top (\hat{S}_t - S_t) z_{t-1} \\ &= \sum_{t=0}^{T} -\frac{1}{2} \text{tr}((\hat{S}_t - S_t) z_{t-1} z_{t-1}^\top (\hat{S}_t - S_t)^\top) - \text{tr}((\hat{S}_t - S_t) z_{t-1} (r_t - \hat{S}_t z_{t-1})^\top), \end{aligned}$$

where we used the identity $a^\top b = \text{tr}(a^\top b) = \text{tr}(b a^\top)$. Invoking Lemma D.2 and Lemma D.3, we have

$$\mathcal{R}_{\text{pred}}(S_{0:T}) \leq 2\sqrt{p} M (\varepsilon + \frac{p D_r^2}{1-\gamma}) V_T + \frac{1}{2} \max_{t \leq T} \|r_t - \hat{S}_t z_{t-1}\|^2 \sum_{t=0}^{T} z_{t-1}^\top P_t^{-1} z_{t-1} + 2\gamma \varepsilon n M^2.$$

The result follows from Lemma D.5 and the fact that $\|r_t\| \leq D_r$, $\|\hat{S}_t z_{t-1}\| \leq M \|z_{t-1}\| \leq M \sqrt{p} D_r$ since z_t a concatenation of p vectors. As a result, we can bound the error $\|r_t - \hat{S}_t z_{t-1}\|^2$ by $(1 + \sqrt{p} M)^2 D_r^2$ for all $t \leq T$. To simplify the final bound we use $T + 1 \leq 2T$. □

The following technical lemmas are auxiliary results towards proving Theorem D.1. Lemmas D.2, D.3 control the growth of the first order (gradient) term in the regret. Lemma D.4 upper bounds matrix P_t while Lemma D.5 contains a standard elliptical potential bound, tailored to the forgetting factor case.

Lemma D.2 (Gradient Inner Product)**.** *Consider the conditions of Theorem D.1. Let $\hat{S}_t$ be the RLS estimate at time t and S_t, S_{t+1} be any arbitrary matrices that satisfy the constraints, i.e., $S_t,\ S_{t+1} \in \mathcal{S}$. We have*

$$
\begin{aligned}
&-2\mathrm{tr}(z_{t-1}(r_t - \hat{S}_t z_{t-1})^\top(\hat{S}_t - S_t)) \\
&\leq \|\hat{S}_t - S_t\|_{F,P_t}^2 - \|\hat{S}_{t+1} - S_{t+1}\|_{F,P_t}^2 + \|r_t - \hat{S}_t z_{t-1}\|^2 z_{t-1}^\top P_t^{-1} z_{t-1} + 4\sqrt{p}M(\varepsilon + \frac{pD_r^2}{1-\gamma})\|S_{t+1} - S_t\|_F
\end{aligned}
$$

Proof. By the non-expansiveness of the projection operator, we have

$$
\begin{aligned}
\|\hat{S}_{t+1} - S_t\|_{F,P_t}^2 &\leq \|\hat{S}_t + (r_t - \hat{S}_t z_{t-1})z_{t-1}^\top P_t^{-1} - S_t\|_{F,P_t}^2 \\
&= \|\hat{S}_t - S_t\|_{F,P_t}^2 + \|(r_t - \hat{S}_t z_{t-1})z_{t-1}^\top P_t^{-1}\|_{F,P_t}^2 + 2\mathrm{tr}(z_{t-1}(r_t - \hat{S}_t z_{t-1})^\top(\hat{S}_t - S_t)) \\
&= \|\hat{S}_t - S_t\|_{F,P_t}^2 + \mathrm{tr}\left((r_t - \hat{S}_t z_{t-1})z_{t-1}^\top P_t^{-1} z_{t-1}(r_t - \hat{S}_t z_{t-1})^\top\right) \\
&\quad + 2\mathrm{tr}(z_{t-1}(r_t - \hat{S}_t z_{t-1})^\top(\hat{S}_t - S_t)) \\
&= \|\hat{S}_t - S_t\|_{F,P_t}^2 + \|r_t - \hat{S}_t z_{t-1}\|^2 z_{t-1}^\top P_t^{-1} z_{t-1} + 2\mathrm{tr}(z_{t-1}(r_t - \hat{S}_t z_{t-1})^\top(\hat{S}_t - S_t))
\end{aligned}
$$

Meanwhile, adding and subtracting S_{t+1} in the norm in the left-hand side of the above inequality we obtain

$$
\begin{aligned}
\|\hat{S}_{t+1} - S_t\|_{F,P_t}^2 &= \|\hat{S}_{t+1} - S_{t+1}\|_{F,P_t}^2 + \|S_{t+1} - S_t\|_{F,P_t}^2 + 2\mathrm{tr}((\hat{S}_{t+1} - S_{t+1})P_t(S_{t+1} - S_t)^\top) \\
&\geq \|\hat{S}_{t+1} - S_{t+1}\|_{F,P_t}^2 + \|S_{t+1} - S_t\|_{F,P_t}^2 - 2\|\hat{S}_{t+1} - S_{t+1}\|_{F,P_t}\|S_{t+1} - S_t\|_{F,P_t} \\
&\geq \|\hat{S}_{t+1} - S_{t+1}\|_{F,P_t}^2 - 2\|\hat{S}_{t+1} - S_{t+1}\|_{F,P_t}\|S_{t+1} - S_t\|_{F,P_t} \\
&\geq \|\hat{S}_{t+1} - S_{t+1}\|_{F,P_t}^2 - 4\sqrt{n}M(\varepsilon + \frac{pD_r^2}{1-\gamma})\|S_{t+1} - S_t\|_F,
\end{aligned}
$$

where we used $\|\hat{S}_{t+1} - S_{t+1}\|_F \leq \sqrt{\mathrm{rank}(\hat{S}_{t+1} - S_{t+1})}\|\hat{S}_{t+1} - S_{t+1}\|$, $\mathrm{rank}(\hat{S}_{t+1} - S_{t+1}) \leq n$, $\|\hat{S}_{t+1} - S_{t+1}\| \leq 2M$ and Lemma D.4. Combining the above two inequalities gives us the result. □

Lemma D.3 (Telescoping Series)**.** *Consider the conditions of Theorem D.1.*

$$
\sum_{t=0}^{T} \|\hat{S}_t - S_t\|_{F,P_t}^2 - \|\hat{S}_{t+1} - S_{t+1}\|_{F,P_t}^2 \leq \sum_{t=0}^{T} \mathrm{tr}((\hat{S}_t - S_t)z_{t-1}z_{t-1}^\top(\hat{S}_t - S_t)^\top) + \gamma\varepsilon\,\mathrm{tr}((\hat{S}_0 - S_0)(\hat{S}_0 - S_0)^\top).
$$

Proof. Notice that

$$
\begin{aligned}
\|\hat{S}_{t+1} - S_{t+1}\|_{F,P_{t+1}}^2 - \|\hat{S}_{t+1} - S_{t+1}\|_{F,P_t}^2 &= \mathrm{tr}((\hat{S}_{t+1} - S_{t+1})(P_{t+1} - P_t)(\hat{S}_{t+1} - S_{t+1})^\top) \\
&\leq \mathrm{tr}((\hat{S}_{t+1} - S_{t+1})z_t z_t^\top(\hat{S}_{t+1} - S_{t+1})^\top),
\end{aligned}
$$

since $P_{t+1} = \gamma P_t + z_t z_t^\top$, $P_t \succ 0$, and $\gamma \leq 1$. For $t = 0$, we have

$$
\|\hat{S}_0 - S_0\|_{F,P_0}^2 = \mathrm{tr}((\hat{S}_0 - S_0)(z_{-1}z_{-1}^\top + \gamma\varepsilon I)(\hat{S}_0 - S_0)^\top)
$$

The result follows by dropping the last negative term $\|\hat{S}_{T+1} - S_{T+1}\|_{F,P_T}^2$. □

Lemma D.4 (Design matrix bound)**.** *Consider the conditions of Theorem D.1 with $P_t = \gamma P_{t-1} + z_{t-1}z_{t-1}^\top$ and $P_{-1} = \varepsilon I$. We have*

$$
\|P_t\| \leq \varepsilon + \frac{pD_r^2}{1-\gamma}.
$$

Proof. Since $\|z_t\| \leq \sqrt{p}\max_{t-p\leq r\leq t}\|r_k\| \leq \sqrt{p}D_r$, we have recursively that

$$\|P_t\| \leq \gamma\|P_{t-1}\| + pD_r^2 = \frac{1-\gamma^t}{1-\gamma}pD_r^2 + \gamma^{t+1}\varepsilon \leq \frac{1}{1-\gamma}pD_r^2 + \varepsilon$$

□

Lemma D.5 (Forgetting potential lemma)**.** *Consider the conditions of Theorem D.1 with $P_t = \gamma P_{t-1} + z_{t-1}z_{t-1}^\top$ and $P_{-1} = \varepsilon I$. The following upper bound is true*

$$\sum_{t=0}^{T} z_{t-1}^\top P_t^{-1} z_{t-1} \leq np\log(\frac{\varepsilon + pD_r^2}{\varepsilon(1-\gamma)}) + np(T+1)\log\frac{1}{\gamma}.$$

Proof. Note that $\gamma P_{t-1} = P_t - z_{t-1}z_{t-1}^\top = P_t^{1/2}(\mathbf{I}_{np} - P_t^{-1/2}z_{t-1}z_{t-1}^\top P_t^{-1/2})P_t^{1/2}$. Using the identity $\det(I+DC) = \det(I+CD)$, we obtain

$$\det(\gamma P_{t-1}) = \det(P_t)(1 - z_{t-1}^\top P_t^{-1} z_{t-1}),$$

which, in turn, implies

$$z_{t-1}^\top P_t^{-1} z_{t-1} = 1 - \frac{\det(\gamma P_{t-1})}{\det(P_t)} \leq -\log\frac{\det(\gamma P_{t-1})}{\det(P_t)}.$$

The inequality follows from the fact that $\gamma P_{t-1} \preceq P_t$ and the elementary inequality $x - 1 \geq \log x$, for $0 < x \leq 1$. By the properties of the determinant, we also have $\det(\gamma P_{t-1}) = \gamma^{np}\det(P_{t-1})$. Hence we have

$$z_{t-1}^\top P_t^{-1} z_{t-1} \leq \log\det(P_t) - \log\det(P_{t-1}) + np\log 1/\gamma.$$

Summing all inequalities and by telescoping

$$\sum_{t=0}^{T} z_t^\top P_t^{-1} z_t \leq \log\det P_T - \log\det(\varepsilon I_{np}) + np(T+1)\log 1/\gamma.$$

The final bound follows from the upper bound on P_T given in Lemma D.4. □

D.2. Regret guarantees for multi-step ahead prediction

Consider now the case of $k-$steps ahead prediction. Recall that for any fixed $k = 1, \ldots, W$, the $k-$step ahead predictor maintains k learners that are updated at non-overlapping intervals. An example for $k = 1, 2, 3$ can be found in Figure 7.

Time t / Steps ahead	t=2	t=3	t=4	t=5	t=6
k=1	Learner 1,0 $\hat{S}_{3\|2} = \hat{S}_{2\|1} + \ldots$	Learner 1,0 $\hat{S}_{4\|3} = \hat{S}_{3\|2} + \ldots$	Learner 1,0 $\hat{S}_{5\|4} = \hat{S}_{4\|3} + \ldots$	Learner 1,0 $\hat{S}_{6\|5} = \hat{S}_{5\|4} + \ldots$	Learner 1,0 $\hat{S}_{7\|6} = \hat{S}_{6\|5} + \ldots$
k=2	Learner 2,1 $\hat{S}_{4\|2} = \hat{S}_{2\|0} + \ldots$	Learner 2,0 $\hat{S}_{5\|3} = \hat{S}_{3\|1} + \ldots$	Learner 2,1 $\hat{S}_{6\|4} = \hat{S}_{4\|2} + \ldots$	Learner 2,0 $\hat{S}_{7\|5} = \hat{S}_{5\|3} + \ldots$	Learner 2,1 $\hat{S}_{8\|6} = \hat{S}_{6\|4} + \ldots$
k=3	Learner 3,0 $\hat{S}_{5\|2} = \hat{S}_{2\|-1} + \ldots$	Learner 3,1 $\hat{S}_{6\|3} = \hat{S}_{3\|0} + \ldots$	Learner 3,2 $\hat{S}_{7\|4} = \hat{S}_{4\|1} + \ldots$	Learner 3,3 $\hat{S}_{8\|5} = \hat{S}_{5\|2} + \ldots$	Learner 3,1 $\hat{S}_{9\|6} = \hat{S}_{6\|3} + \ldots$

$(t+1) \mod k = 0$ $(t+1) \mod k = 1$ $(t+1) \mod k = 2$

Figure 7. The updates of the $k = 1, 2, 3$ steps ahead predictors for $t = 2, \ldots, 6$. Projections are omitted for simplicity. Each predictor consists of k independent learners that are updated at non-overlapping time steps. For example, for $k = 2$, there are two learners and the update for $\hat{S}_{t+2|t}$ is decoupled from the one of $\hat{S}_{t+1|t-1}$.

Let $S_{k-1|-1}, S_{k|0}, \ldots, S_{T|T-k}$ be any comparator sequence that satisfies Assumption 2.2. In this case, the regret of $k-$step ahead prediction is defined as

$$\mathcal{R}_{\text{pred}}^{(k)} \triangleq \sum_{t=k-1}^{T} \|\hat{S}_{t|t-k} z_{t-k} - r_t\|^2 - \sum_{t=k-1}^{T} \|S_{t|t-k} z_{t-k} - r_t\|^2.$$

Once again if we choose the sequence to coincide with the true dynamics, we get that

$$\mathcal{R}_{\text{pred}}^{(k)} \triangleq \sum_{t=k-1}^{T} \|\hat{S}_{t|t-k} z_{t-k} - r_t\|^2.$$

Recall that we have k learners updated at non-overlapping time steps. Let i, for $i = 0, \ldots, k-1$, denote the index for the learner. We will decompose the problem into k independent one-step ahead prediction problems and invoke Theorem D.1.

Every learner i is invoked to predict $r_{t+k|t}$ when $((t+1) \mod k) = i$. Learner 0 is invoked to predict $r_{k-1|-1}$, $r_{2k-1|k-1}$, $\ldots$, etc. Similarly, learner i is invoked to predict $r_{k+i-1|i-1}$, $r_{2k+i-1|k+i-1}$, $\ldots$, etc. We will redefine the time axis to reduce the problem to one-step ahead prediction. To streamline the presentation define

$$\begin{aligned}
\hat{S}_{\tau+1}^{i} &\triangleq \hat{S}_{(\tau+1)k+i-1|\tau k+i-1} \\
S_{\tau+1}^{i} &\triangleq S_{(\tau+1)k+i-1|\tau k+i-1} \\
z_{\tau-1}^{i} &= z_{\tau k+i-1},\ r_{\tau}^{i} = r_{(\tau+1)k+i-1}
\end{aligned}$$

and denote

$$\begin{aligned}
N_{k,i} &= \lfloor \frac{T+1}{k} \rfloor, \text{ if } i \leq (T+1) \mod k \\
&= \lfloor \frac{T+1}{k} \rfloor - 1, \text{ otherwise.}
\end{aligned}$$

Note that learner i is invoked starting at $\tau = 0$ up to $N_{i,k}$ times.

Denote the regret of learner i by

$$\mathcal{R}_{\text{pred}}^{(k,i)}(S_{0:N_{k,i}}^{i}) \triangleq \frac{1}{2} \sum_{\tau=0}^{N_{k,i}} \|\hat{S}_{\tau+1}^{i} z_{\tau}^{i} - r_{\tau+1}^{i}\|^2 - \frac{1}{2} \sum_{t=0}^{N_{k,i}} \|S_{\tau+1}^{i} z_{\tau}^{i} - r_{\tau+1}^{i}\|^2. \tag{24}$$

We can now prove Theorem 4.5.

D.3. Proof of Theorem 4.5

Based on the above notation, the regret can be decomposed into k terms

$$\mathcal{R}_{\text{pred}}^{(k)} = \sum_{i=0}^{k-1} \mathcal{R}_{\text{pred}}^{(k,i)}(S_{0:N_{k,i}}^{i}).$$

As a consequence of redefining the time axis, the prediction regret of every individual learner i can be bounded using Theorem D.1. Let $V_T^{k,i} = \sum_{\tau=0}^{N_{k,i}-1} \|S_{\tau+1}^{i} - S_{\tau}^{i}\|$. Then we obtain

$$\mathcal{R}_{\text{pred}}^{(k,i)}(S_{0:N_{k,i}}^{i}) \leq \frac{\beta_1}{1-\gamma} V_T^{k,i} + \beta_2 (\frac{T+1}{k}) \log \frac{1}{\gamma} + \beta_3 \log \frac{1}{1-\gamma} + \beta_4. \tag{25}$$

Notice that $V_T^k = \sum_{i=0}^{k-1} V_T^{k,i}$. Hence, summing up we obtain

$$\mathcal{R}_{\text{pred}}^{(k)} \leq \frac{\beta_1}{1-\gamma} V_T^k + \beta_2 (T+1) \log \frac{1}{\gamma} + k\beta_3 \log \frac{1}{1-\gamma} + k\beta_4. \qquad \square$$

E. Regret of the PLOT algorithm

Let $\Sigma = B^\top X B + R$. Consider again the optimal non-causal policy

$$u_t^*(x_t) = -K(x_t - r_t) - \underbrace{\sum_{i=t}^{T-1} K_{i-t}(Ar_i - r_{i+1})}_{q_t(r_{t:T})},$$

where the optimal feedforward terms are given by

$$q_t(r_{t:T}) = \sum_{i=t}^{T-1} K_{i-t}(Ar_i - r_{i+1}).$$

Consider also the causal suboptimal policy of the PLOT algorithm

$$u_t^\pi(x_t) = -K(x_t - r_t) - \underbrace{\sum_{i=t}^{\min\{t+W-1,T\}} K_{i-t}(Ar_{i|t} - r_{i+1|t})}_{\hat{q}_t(r_{t:t+W|t})},$$

where we define the truncated feedforward terms that use the reference predictions

$$\hat{q}_t(r_{t:t+W|t}) = \sum_{i=t}^{\min\{t+W-1,T\}} K_{i-t}(Ar_{i|t} - r_{i+1|t}).$$

Finally, define the residual feedforward terms $\delta_t(r_{t+W:T}) \triangleq q_t(r_{t:T}) - \hat{q}_t(r_{t:t+W})$. With these definitions in hand, we can now analyze the regret of PLOT.

By invoking the performance difference lemma (Lemma 4.4), we obtain that the regret is equal to

$$\begin{aligned}\mathcal{R}(\pi) &= \sum_{t=0}^{T-1} \|\hat{q}_t(r_{t:t+W|t}) - q_t(r_{t:T})\|_\Sigma^2 \\ &= \sum_{t=0}^{T-1} \|\hat{q}_t(r_{t:t+W|t} - r_{t:t+W}) - \delta_t(r_{t+W:T})\|_\Sigma^2.\end{aligned}$$

Invoking Cauchy-Schwartz for the two summands, we can now decompose the regret into two terms

$$\mathcal{R}(\pi) \le \underbrace{2\sum_{t=0}^{T-1} \|\hat{q}_t(r_{t:t+W|t} - r_{t:t+W})\|_\Sigma^2}_{\text{prediction error}} + \underbrace{2\sum_{t=0}^{T} \|\delta_t(r_{t+W:T})\|_\Sigma^2}_{\text{truncation error}}, \tag{26}$$

where the first term captures the effect of the prediction error, while the second term captures the effect of truncation. To bound the latter we invoke the following lemma.

Lemma E.1 (Truncation term)**.** *The truncation term of the regret satisfies*

$$2\sum_{t=0}^{T} \|\delta_t(r_{t+W:T})\|_\Sigma^2 = 2\sum_{t=0}^{T-1-W} \|\sum_{i=t+W}^{T-1} K_{i-t}(Ar_i - r_{i+1})\|_\Sigma^2 \le \alpha_1 \frac{\rho^{2W}}{(1-\rho)^2} T,$$

where

$$\alpha_1 = 2c_0^2(\|A\| + 1)^2 D_r^2 \|\Sigma\|. \tag{27}$$

Proof. The first equality follows by the definition of $\delta_t(r_{t+W:T})$ and the fact that $\delta_t(r_{t+W:T}) = 0$, for $t + W > T - 1$. By Proposition B.1, we have $\|K_t\| \leq c_0\rho^t$. Hence

$$
\begin{aligned}
&2\sum_{t=0}^{T-1-W} \|\sum_{i=t+W}^{T-1} K_{i-t}(Ar_i - r_{i+1})\|_\Sigma^2 \leq 2c_0^2(\|A\|+1)^2\|\Sigma\|D_r^2 \sum_{t=0}^{T-1-W}(\sum_{i=t+W}^{T-1}\rho^{i-t})^2 \\
&= \alpha_1\rho^{2W}\sum_{t=0}^{T-1-W}(\sum_{i=0}^{T-1-W-t}\rho^i)^2 \leq \alpha_1\rho^{2W}\sum_{t=0}^{T-1-W}\frac{(1-\rho^{T-W-t})^2}{(1-\rho)^2} \\
&\leq \alpha_1\frac{\rho^{2W}}{(1-\rho)^2}T
\end{aligned}
$$

□

What remains to show is that the prediction error term is upper bounded in terms of the prediction regret of the RLS algorithm.

Lemma E.2 (Prediction term)**.** *The prediction term of the regret is upper bounded by*

$$
2\sum_{t=0}^{T-1}\|\hat{q}_t(r_{t:t+W|t} - r_{t:t+W})\|_\Sigma^2 \leq c_1\sum_{i=1}^{W}\rho^{i-1}\mathcal{R}_{pred}^{(i)},
$$

where

$$
c_1 = 2c_0^2\|\Sigma\|\tilde{W},\ \tilde{W} = \min\{(1-\rho)^{-1}, W\}.
$$

Proof. Let $t \leq T - W + 1$ for simplicity. The case $t > T - W + 1$ is similar. Then,

$$
\hat{q}_t(r_{t:t+W|t}) = \sum_{i=0}^{W-1} K_i(Ar_{t+i|t} - r_{t+i+1|t}).
$$

By regrouping the terms, and since $r_t = r_{t|t}$ we obtain

$$
\hat{q}_t(r_{t:t+W|t}) = K_0Ar_t + \sum_{i=1}^{W-1}(K_iA - K_{i-1})r_{t+i|t} - K_{W-1}r_{t+W|t}.
$$

As a result

$$
\hat{q}_t(r_{t:t+W|t} - r_{t:t+W}) = \sum_{i=1}^{W-1} L_i(r_{t+i|t} - r_{t+i}) - K_{W-1}(r_{t+W|t} - r_{t+W})
$$

where $L_i \triangleq K_iA - K_{i-1}$. Using the properties of the LQT controller and (6)

$$
\begin{aligned}
L_i = K_iA - K_{i-1} &= \Sigma^{-1}B^\top(A-BK)^{\top,i}XA - \Sigma^{-1}B^\top(A-BK)^{\top,i-1}X \\
&= \Sigma^{-1}B^\top(A-BK)^{\top,i-1}((A-BK)^\top XA - X) \\
&= \Sigma^{-1}B^\top(A-BK)^{\top,i-1}(A^\top XA - K^\top B^\top XA - X) \\
&= \Sigma^{-1}B^\top(A-BK)^{\top,i-1}(-Q + K^\top B^\top XA - K^\top B^\top XA) \\
&= -\Sigma^{-1}B^\top(A-BK)^{\top,i-1}Q.
\end{aligned}
$$

Hence, by Proposition B.1 and the fact that $Q \preceq X$, we also get

$$
\|L_i\| \leq c_0\rho^{i-1}.
$$

The difference between the truncated feedforward terms now becomes

$$
\begin{aligned}
\|\hat{q}_t(r_{t:t+W|t} - r_{t:t+W})\|_\Sigma^2 &\leq c_0^2\|\Sigma\|\left(\sum_{i=1}^{W}\rho^{i-1}\|r_{t+i} - r_{t+i|t}\|\right)^2 = c_0^2\|\Sigma\|\left(\sum_{i=1}^{W}\rho^{(i-1)/2}\rho^{(i-1)/2}\|r_{t+i} - r_{t+i|t}\|\right)^2 \\
&\overset{i)}{\leq} c_0^2\|\Sigma\|\sum_{i=1}^{W}\rho^{i-1}\sum_{i=1}^{W}\rho^{i-1}\|r_{t+i} - r_{t+i|t}\|^2 \leq c_0^2\|\Sigma\|\tilde{W}\sum_{i=1}^{W}\rho^{i-1}\|r_{t+i} - r_{t+i|t}\|^2,
\end{aligned}
$$

where $i)$ follows by Cauchy-Schwartz. The result for $t > T - W + 1$ is similar. To obtain the final bound we just need to sum over t. □

Before we proceed to the proof of Theorem 4.1, let us recall the following standard result.

Lemma E.3 (Geometric Series)**.** *Let* $0 \leq \rho < 1$. *Then the following hold*

$$S_1(\rho) \triangleq \sum_{i=1}^{N} i\rho^{i-1} = \frac{-(N+1)\rho^N(1-\rho) + (1-\rho^{N+1})}{(1-\rho)^2} \leq \max\left\{\frac{1}{(1-\rho)^2}, N^2\right\}$$

$$S_2(\rho) \triangleq \sum_{i=1}^{N} i^2\rho^{i-1} = \frac{1+\rho-(N+1)^2\rho^N + (2N^2+2N-1)\rho^{N+1} - N^2\rho^{N+2}}{(1-\rho)^3} \leq \max\left\{\frac{2}{(1-\rho)^3}, N^3\right\}$$

Proof. The bounds $\sum_{i=1}^{N} i^k\rho^{i-1} \leq N^{k+1}$ are immediate since $0 \leq \rho < 1$. The closed-form expressions of the sums are standard, but we repeat the proof here for completeness. For $S_1(\rho)$, the expression follows from

$$S_1(\rho) = \frac{d}{d\rho}(\sum_{i=0}^{N} \rho^i) = \frac{d}{d\rho}\left(\frac{1-\rho^{N+1}}{1-\rho}\right).$$

To show the upper bound notice that

$$-(N+1)\rho^N(1-\rho) + (1-\rho^{N+1}) = 1 - N\rho^N(1-\rho) - \rho^N \leq 1.$$

For S_2 we use the identity

$$S_2(\rho) = \rho\sum_{i=1}^{N} i(i-1)\rho^{i-2} + S_1(\rho) = \rho\frac{d}{d\rho}S_1(\rho) + S_1(\rho).$$

To show the upper bound notice that

$$\begin{aligned}&1+\rho-(N+1)^2\rho^N + (2N^2+2N-1)\rho^{N+1} - N^2\rho^{N+2}\\ &= 1+\rho-N^2(1-\rho)^2 - 2N\rho^N(1-\rho) - \rho^N - \rho^{N+1} \leq 1+\rho \leq 2.\end{aligned}$$

□

E.1. Proof of Theorem 4.1

By (26), Lemma E.1, and Lemma E.2, we obtain that

$$\mathcal{R}(\pi) \leq \alpha_1\frac{\rho^{2W}}{(1-\rho)^2}T + c_1\sum_{i=1}^{W}\rho^{i-1}\mathcal{R}_{\text{pred}}^{(i)}.$$

Further, by Theorem 4.5

$$\begin{aligned}\mathcal{R}(\pi) \leq\ &\alpha_1\frac{\rho^{2W}}{(1-\rho)^2}T + c_1\beta_1\frac{1}{1-\gamma}\sum_{i=1}^{W}\rho^{i-1}V_T^i + c_1\beta_2(T+1)\log\frac{1}{\gamma}\sum_{i=1}^{W}\rho^{i-1}\\ &+ c_1\left(\beta_3\log\frac{1}{1-\gamma} + \beta_4\right)\sum_{i=1}^{W} i\rho^{i-1}.\end{aligned}$$

By the perturbation bound in Lemma C.3, we can upper bound $V_T^i \leq \sqrt{p}M^2 i^2 V_T$. Hence, by invoking Lemma E.3 we finally obtain

$$\begin{aligned}\mathcal{R}(\pi) &\leq \alpha_1 \frac{\rho^{2W}}{(1-\rho)^2} T + 2c_1\beta_1\sqrt{p}M^2 \frac{1}{1-\gamma}\tilde{W}^3 V_T + c_1\beta_2\tilde{W}(T+1)\log\frac{1}{\gamma} \\ &+ c_1\left(\beta_3\tilde{W}^2\log\frac{1}{1-\gamma} + \tilde{W}^2\beta_4\right) \\ &= \alpha_1\frac{\rho^{2W}}{(1-\rho)^2}T + \alpha_2\tilde{W}^4 V_T(1-\gamma)^{-1} - \alpha_3\tilde{W}^2(T+1)\log\gamma \\ &- \alpha_4\tilde{W}^3\log(1-\gamma) + \alpha_5\tilde{W}^3.\end{aligned}$$

The coefficients are given by

$$\begin{aligned}\alpha_1 &= 2c_0^2(\|A\|+1)^2 D_r^2\|\Sigma\| \\ \alpha_2 &= 4c_0^2\|\Sigma\|\beta_1\sqrt{p}M^2 \\ \alpha_3 &= 2c_0^2\|\Sigma\|\beta_2 \\ \alpha_4 &= 2c_0^2\|\Sigma\|\beta_3 \\ \alpha_5 &= 2c_0^2\|\Sigma\|\beta_4,\end{aligned} \tag{28}$$

where $\beta_1, \beta_2, \beta_3, \beta_4$ are given in (23) and $\|\Sigma\| = \|B^\top XB + R\|$. □

E.2. Proof of Corollary 4.2

The fifth term is constant since $\tilde{W}$ is bounded. Taking $W = -\frac{logT}{2\log\rho}$ implies that $\rho^{2W}T = 1$, hence the truncation term is constant. Note that under the given choice for γ, we have $(1-\gamma)^{-1} = \mathcal{O}(T^2)$, since $1-\gamma \geq \log T/T^2$. Thus, the fourth term is at most logarithmic with T: $\log(1-\gamma)^{-1} = \mathcal{O}(\log T)$.

The second term satisfies

$$\frac{V_T}{1-\gamma} = \sqrt{4M}\frac{V_T\sqrt{T}}{\max\{\sqrt{V_T}, \log T/\sqrt{T}\}} = \sqrt{4M}\min\{\sqrt{V_T T}, \log T\}$$

Finally, for the third term, we invoke the elementary inequality

$$\log\gamma \geq 1 - \frac{1}{\gamma},$$

which is equivalent to

$$-(T+1)\log\gamma \leq (T+1)\frac{1-\gamma}{\gamma}.$$

Note that the maximum possible value of V_T is $2MT$ since matrices S_t are bounded. Hence the forgetting factor $\gamma = 1 - \sqrt{\frac{\max\{V_T, \log^2 T/T\}}{4MT}} \geq 1 - \frac{1}{\sqrt{2}}\sqrt{\frac{V_T}{2MT}} \geq 1 - \frac{1}{\sqrt{2}}$ can be lower bounded. As a result, we obtain

$$-(T+1)\log\gamma \leq (T+1)\frac{1-\gamma}{\gamma} \leq \frac{\sqrt{2}}{\sqrt{2}-1}(T+1)\sqrt{\frac{\max\{V_T, \log^2 T/T\}}{4MT}} = \mathcal{O}(\max\{\sqrt{TV_T}, \log T\}).$$ □

F. Simulations

F.1. The Quadrotor Model

To demonstrate the performance of PLOT on a practical use-case we consider the linearized dynamics of the Crazyflie quadrotor (Bitcraze, 2023b), derived in (Beuchat, 2019). The mini quadrotor can be modeled with a continuous-time nonlinear model $g : \mathbb{R}^{9\times 1} \times \mathbb{R}^{4\times 1} \to \mathbb{R}^{9\times 1}$

$$\dot{x} = g(x, u),$$

where the state x and the input u are defined as

$$x := \begin{bmatrix} p \\ \dot{p} \\ \psi \end{bmatrix}, \qquad u := \begin{bmatrix} f \\ \omega \end{bmatrix},$$

with $p := [p_x; p_y; p_z]^\top$ defined as the position vector in the inertial frame, $\psi := [\gamma, \beta, \alpha]^\top$ as the attitude vector in the same frame and γ, β and α denote the attitude angles, roll, pitch and yaw, respectively. The action $u := [f; \omega]^\top$ comprises of the total thrust f, as well as the angular rate $\omega := [\omega_x, \omega_y, \omega_z]^\top$ in the body frame. We provide a brief derivation of the linearized model for the quadrotor of interest and refer the readers to (Beuchat, 2019) for a detailed derivation.

Given the low speeds of the quadrotor, we neglect the aerodynamic drag forces resulting in the following equations of motion for translation

$$\ddot{p} = \begin{bmatrix} \ddot{p}_x \\ \ddot{p}_y \\ \ddot{p}_z \end{bmatrix} = \frac{1}{m}\left(R_{IB} \begin{bmatrix} 0 \\ 0 \\ f \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} \right), \tag{29}$$

where m is the mass of the quadrotor, g is the gravitational acceleration constant, and R_{IB} is the rotation matrix from the body frame to the inertial frame. In particular, given the attitude angles, R_{IB} is given by

$$R_{IB} = \begin{bmatrix} c_\alpha c_\beta & (-s_\alpha c_\gamma + c_\alpha s_\beta s_\gamma) & (s_\alpha s_\gamma + c_\alpha s_\beta c_\gamma) \\ s_\alpha c_\beta & (c_\alpha c_\gamma + s_\alpha s_\beta s_\gamma) & (-c_\alpha s_\gamma + s_\alpha s_\beta c_\gamma) \\ -s_\beta & c_\beta s_\gamma & c_\beta c_\gamma \end{bmatrix},$$

where $s_\theta := \sin(\theta)$ and $c_\theta := \cos(\theta)$ for a given angle θ. The equations of motion for rotation can similarly be derived as

$$\dot{\psi} = \begin{bmatrix} 1 & \sin\gamma\tan\beta & \cos\gamma\tan\beta \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma\sec\beta & \cos\gamma\sec\beta \end{bmatrix} \omega. \tag{30}$$

Using the equations (29) and (30), the linearized dynamic for the quadrotor can then be attained as

$$\dot{\delta x} = \underbrace{\frac{\partial g}{\partial x^\top}\Big|_{\substack{x=x_h \\ u=u_h}}}_{A_c} \delta x \; + \; \underbrace{\frac{\partial g}{\partial u^\top}\Big|_{\substack{x=x_h \\ u=u_h}}}_{B_c} \delta u,$$

where $x_h = \mathbf{0} \in \mathbb{R}^{9\times 1}$ and $u = [mg; 0; 0; 0]^\top$ are the hovering position, steady state, and input, and $\delta x := x - x_h,\ \delta u := u - u_h$. Calculating the respective Jacobians, and discretizing the resulting dynamics with a sampling time T_s, the following dynamics are derived

$$x_{t+1} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & T_s & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & T_s & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & T_s & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & gT_s & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & -gT_s & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{A} x_t \; + \; \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \frac{T_s}{m} & 0 & 0 & 0 \\ 0 & T_s & 0 & 0 \\ 0 & 0 & T_s & 0 \\ 0 & 0 & 0 & T_s \end{bmatrix}}_{B} u_t, \tag{31}$$

where $x_t = \delta x_t$, and with a slight abuse of notation we take $u_t := \delta u_t$ and consider u_h as a feedforward input term required for hovering.

Throughout this section, we will consider the linearised drone dynamics (31) for PLOT and other online control algorithms. We fix the sampling time to $T_s = 0.1$ seconds and the quadratic cost matrices to $Q := diag(80, 80, 80, 10, 10, 10, 0.01, 0.01, 0.1)$, and $R := diag(0.7, 2.5, 2.5, 2.5)$ to match the ones tuned for the experiments on the hardware.

For the PLOT policy, we take $p = 1$ for all experiments, and the projection step 8 in the RLS Algorithm 1 is executed with a quadratic program solver, solving the following problem

$$\hat{S}_{t+k|t} = \arg\min_{S} \|S - Y\|_{F,P_{t|t-k}} \\ \text{s.t. } \|\text{vec}(S)\|_\infty \leq M,$$

where $\text{vec}(S)$ is a vector formed by vertically stacking the columns of S. Note that, the $\|\cdot\|_\infty$ constraint is a conservative approximation of the original problem, resulting in a safe, and conservative estimate of the original projection $\Pi_{\mathcal{S}}^{P_{t|t-k}}$.

In this section, we first analyze PLOT, showing the effect of the prediction horizon, W on the tracking performance, and how this is reflected in the dynamic regret bound. We also show how the regret bound can be used to adjust the forgetting factor, γ, to achieve better tracking given the order of the reference target path length. Next, we compare the proposed method with state-of-the-art online control algorithms for the non-stochastic adversarial control setting with both static and dynamic regret bounds. We show how for certain dynamic references the proposed indirect method of PLOT provides better tracking and lower dynamic regret, and how for others it can perform worse than direct methods.

F.2. PLOT: Regret for hyperparameter tuning

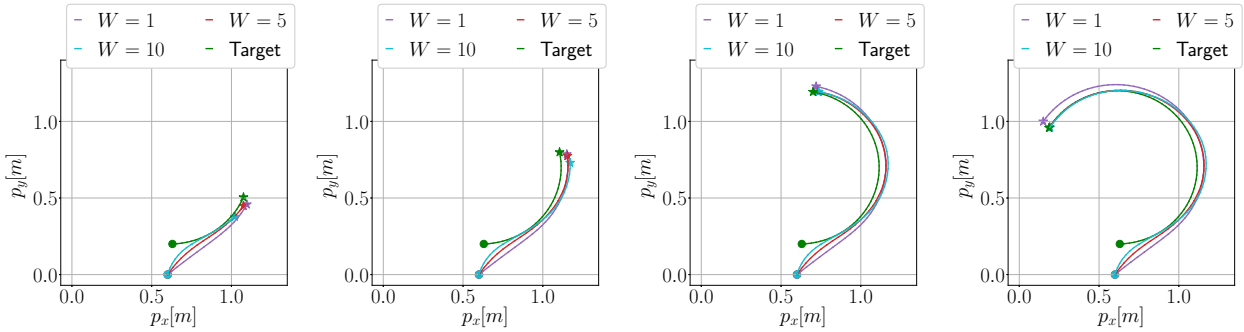
The dynamic regret guarantees derived in Theorem 4.1 and Corollary 4.2 can be used to tune the hyperparameters of the proposed method, such as prediction horizon length, W, or the forgetting factor γ. Though the exact bounds are often over-conservative in practice, the bound order still provides an intuition of the effect of the parameters on tracking performance.

F.2.1. The prediction horizon W

In this setting, we aim to demonstrate the effect of the prediction horizon length, W, on the tracking performance of PLOT. We consider a static target with a $V_T = 0$ path length. A simple target respecting this condition is the circular target, as introduced in Example C.2 with $\theta = 0.06$ radians. With this target revealed and measured online, as described in Section 2, we run the PLOT algorithm repeatedly for varying numbers of horizon lengths. For all, we set the same initial state of $p = [0.6, 0.0, 0.4]^\top$, $M = 10$, and forgetting factor $\gamma = 0.8$. We define the augmented matrix $\tilde{S}_t \triangleq \begin{bmatrix} S_t & v_t \end{bmatrix}$ for $t = 1, \ldots, T-1$ as in Section 2, and for all $k = 1, \ldots, W$, the k learners are initialized as follows: $\hat{S}_{j+k|j} = \mathbf{I}_6, v_{j+k|j} = \mathbf{0}_6$ and $P_{j|j-k} = 10^{-4} \times \mathbf{I}_7$, for all $j = -1, \ldots, k-2$. Here, $\mathbf{1}_d$ denotes the d-dimensional vector of all ones.

The circular target, as well as the trajectory followed by the quadrotor model (31) under the PLOT algorithm is shown in Figure 8 at $T = 2, 3, 5$ and $T = 7$ seconds. Figure 9 shows one complete round of a circle with $T = 10.7$ seconds. The corresponding regret plots (including more prediction horizons) are shown in Figure 10 for a longer simulation of $T = 200$ seconds.

Several observations can be made from these experiments. Firstly, a prediction horizon length not long enough, e.g. $W = 1$ or $W = 3$ results in a poor performance, reflected in both Figures in terms of tracking and regret. While higher horizon lengths achieve efficient tracking by learning the reference dynamics on the go, achieving sublinear regret. This behavior matches the intuition Corollary 4.2 provides, as a long enough W is required to achieve logarithmic regret in T. Secondly, an increase in regret of PLOT for larger W-s can be observed in Figure 10, as "predicted" by the second term of the bound in Theorem 4.1. This increase in regret, however, is saturated at a certain value as can be observed from the figure; in this example, the increase stops after around $W = 15$. This behavior also is captured by the bound in Theorem 4.1, as, crucially, the second term is weighted by increasing powers of ρ, which reduces regret exponentially counteracting the linear increase with W. This shows that the PLOT does not suffer, at least in terms of regret, due to higher prediction horizons.



(a) $T = 2$ seconds (b) $T = 3$ seconds (c) $T = 5$ seconds (d) $T = 7$ seconds

Figure 8. Trajectory plots of a circular target with a $V_T = 0$ path length and the PLOT Algorithm for varying prediction horizon lengths, simulated for $T = 2, 3, 5$ and $T = 7$ seconds.

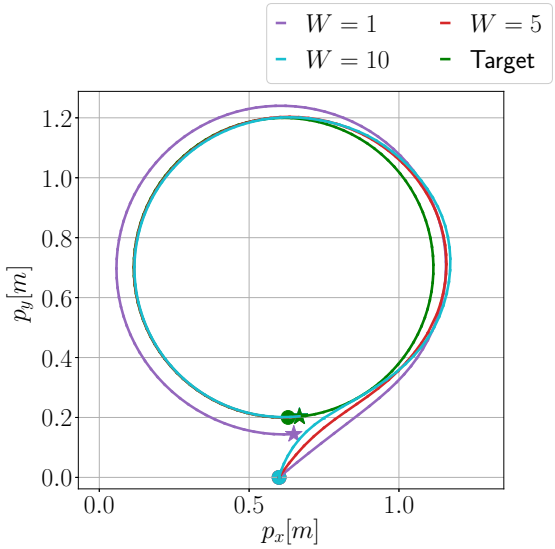


Figure 9. Trajectory plot of a circular target with a $V_T = 0$ path length and the PLOT Algorithm for varying prediction horizon lengths, simulated for $T = 10.7$ seconds.

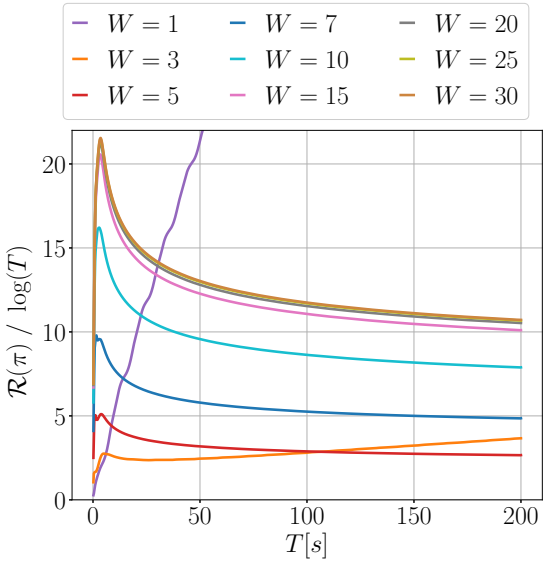


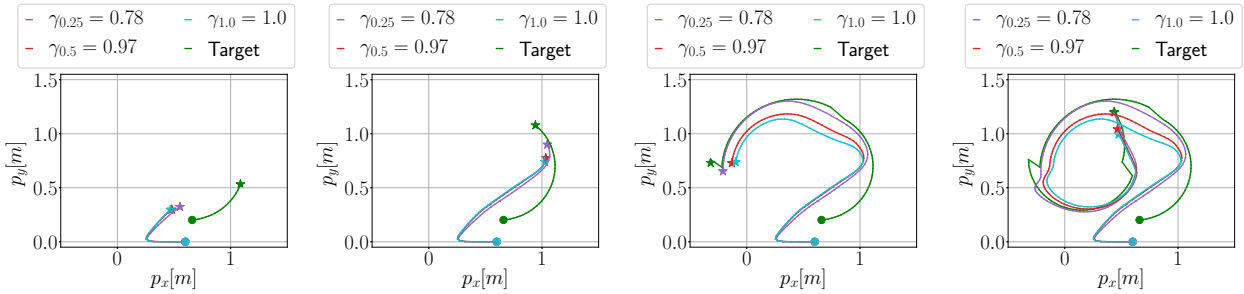
Figure 10. Log-normalized regret of the PLOT Algorithm with a range of prediction horizon lengths, simulated over a horizon of $T = 200$ seconds.

F.2.2. The forgetting factor γ

As suggested by Corollary 4.2, the desired regret rate can be achieved by tuning the forgetting factor, γ, given the path length V_T, and the control horizon, T. In practice, the exact constants appearing in the expression of optimal γ are not known. However, one can still tune the gamma based on the order of the path length and the prediction horizon. In particular, we fix $\gamma_a = 1 - c_\gamma T^{-a}$, where $c_\gamma \in \mathbb{R}_+$ is constant independent of V_T and T, while $a \in \mathbb{R}_+$ is such that $\gamma_a \in (0, 1]$ and is tuned based on V_T and T.

To show this on an example, we first fix a dynamic reference trajectory with a path length $V_T = \mathcal{O}(\sqrt{T})$ and perform 5 different runs of PLOT for a range of values for $\gamma = \gamma_a$, with $a = 0.10, 0.20, 0.25, 0.40, 0.50$ and 1.0. We fix $M = 1$ and $W = 5$. Note that for this path length the optimal value for a, suggested by Corollary 4.2 is $a = 0.25$. The dynamics for the reference are generated by starting with the circle static dynamics in the previous example and increasing the radius of the circle every $\sqrt{T}$ time steps by a factor sampled uniformly from 0.7 and 1.5. In addition, a shift of $[-0.1, 0.1]^\top$ is applied to $[p_x, p_y]^\top$ at the same time step. As in the previous example, we define the augmented matrix $\tilde{S}_t \triangleq \begin{bmatrix} S_t & v_t \end{bmatrix}$ for $t = 1, \dots, T-1$ as in Section 2. To highlight the effect of the forgetting factor, for all $k = 1, \dots, W$, the k learners are initialized as follows: $\hat{S}_{j+k|j} = \mathbf{0}_{6\times 6}, v_{j+k|j} = \mathbf{0}_6$ and $P_{j|j-k} = \mathbf{I}_7$, for all $j = -1, \dots, k-2$.

The dynamic target trajectory together with the system trajectory under the PLOT controller with different forgetting factors



(a) $T = 2$ seconds (b) $T = 4$ seconds (c) $T = 9$ seconds (d) $T = 15$ seconds

Figure 11. Trajectory plot of a spiral with a $V_T = \mathcal{O}(\sqrt{T})$ tracked with PLOT with $W = 5$ and a range of values for γ, simulated for $T = 2, 4, 9$ and $T = 15$ seconds.

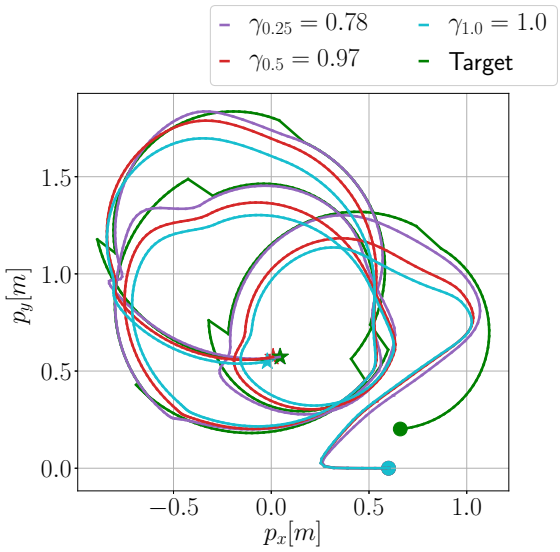


Figure 12. Trajectory plot of a spiral with a $V_T = \mathcal{O}(\sqrt{T})$ tracked with PLOT with a $W = 5$ and a range of values for γ.

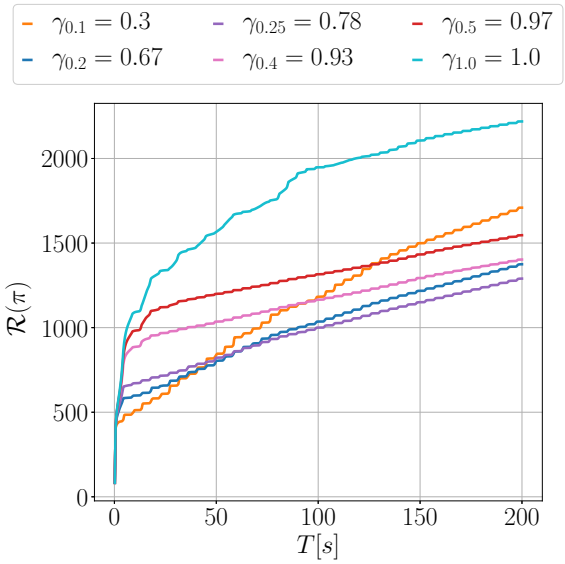


Figure 13. Regret of PLOT for the spiral target with $V_T = \sqrt{T}$ for a range of γ-s. $\gamma = 0.78$ is chosen as per Corollary 4.2.

is plotted in Figure 11 for $T = 2, 4, 9$ and $T = 15$ seconds. A longer trajectory for $T = 40$ seconds is provided in Figure 12 with the corresponding regret plot in Figure 13 for $T = 200$ seconds and with more values of γ. It can be observed from the figures that $\gamma_{0.25}$ tuned optimally to scale with $1 - T^{-0.25}$ achieves the lowest regret at the end of the horizon compared to other a-s. Intuitively, forgetting helps to adapt quickly to the fast-changing dynamics of the reference and lose the influence of the random initialization of the predictions exponentially fast. This can be observed on the plot, by noticing that the regret for $\gamma = 1.0$, corresponding to PLOT without any forgetting is the highest, as expected from the theoretical results. The random initialization of learners, as well as the fast-changing dynamics of the reference, make the no-forgetting tracker maintain a larger tracking error. At the other extreme, forgetting too much, as in this case with $\gamma = 0.3$ hinders the learner from estimating the dynamics, enough to cause higher regret.

To show that the scaling constant is independent of horizon T, we fix, c_γ to a tuned value of 1.5 and perform experiments with varying horizon lengths from $T = 150$ to $T = 300$ seconds. For each T, PLOT is run 4 times, each time with a different γ_a. The regret at the end of each horizon is calculated, and the results are visualized in Figure 14. As expected from Corollary 4.2, PLOT with forgetting factor $\gamma_{0.25}$ performs the best for such a reference, across different horizon lengths.

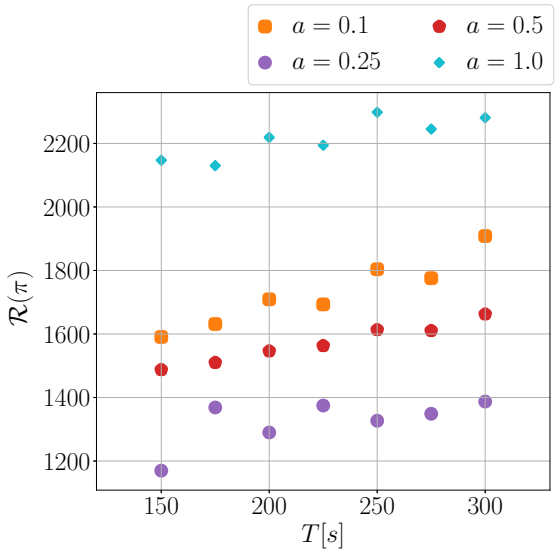


Figure 14. Regret of PLOT with varying $\gamma_a = 1 - c_\gamma T^{-a}$.

F.3. Comparison with Benchmarks

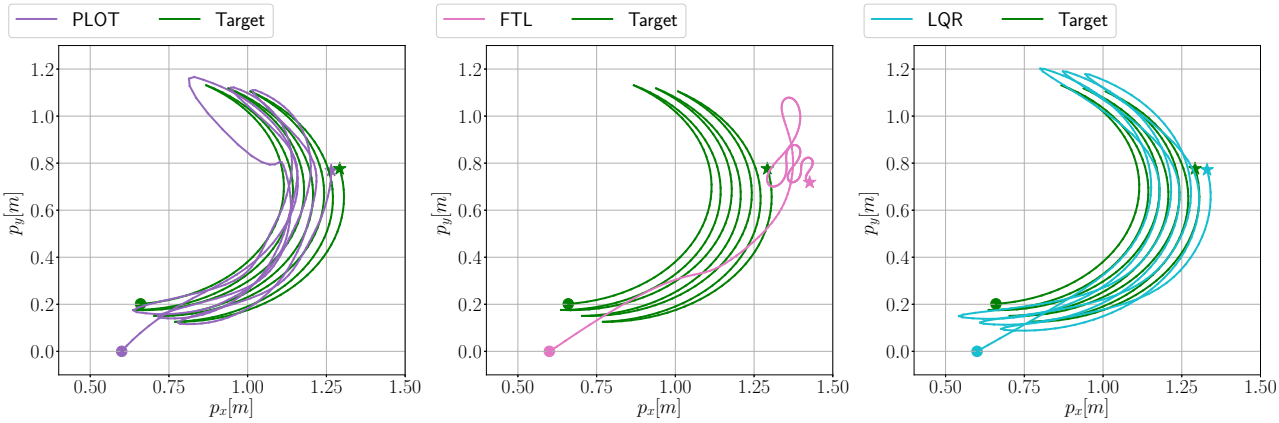
In this section, we consider a highly dynamic reference target and compare PLOT to state-of-the-art online learning algorithms that can also be applied to the online tracking problem in the linear quadratic setting. In particular, we consider the following dynamics for the target with $p = 1$

$$r_{t+1} = S_{t+1|t} = \begin{bmatrix} 1 & 0 & T_s & 0 \\ 0 & 1 & 0 & T_s \\ 0 & 0 & s_t \cos\theta_t & -s_t \sin\theta_t \\ 0 & 0 & s_t \sin\theta_t & s_t \cos\theta_t \end{bmatrix} r_t, \tag{32}$$

where $s_0 = 1, \theta_0 = 0.06$, $T_s = 0.1s$ and $s_k = -s_{k-1}$, $\theta_k = -0.99 \times \theta_{k-1}$ for every $k = \sqrt{T}$ given some T. We simulate $T = 200$ seconds or 2000 time steps.

We compare PLOT to 5 algorithms, namely, the Follow the Leader (FTL) algorithm (Abbasi-Yadkori et al., 2014), Disturbance Action Policy controller (Agarwal et al., 2019a), Riccatitron (Foster & Simchowitz, 2020), SS-OGD (Karapetyan et al., 2023), and Naive LQR that only performs a static feedback on the currently observed tracking error. Apart from the latter, we tune each controller's hyperparameters to achieve the best performance for the given reference, by performing a grid search over the hyperparameters. All controllers start from the same initial state and follow the same target revealed at the same point in time. We provide the details for each below in Appendix F.3.1.

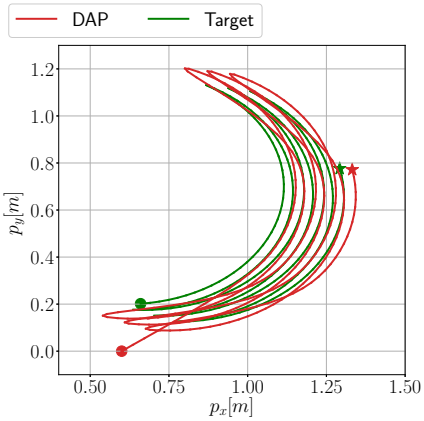
The trajectories of the target and the system under the considered algorithms are plotted in Figure 15 for the first 30 seconds of the simulation. The corresponding dynamic regret plot for the full $T = 200$ seconds horizon is shown in Figure 16. Given the dynamic nature of the reference with a sublinear path length, PLOT outperforms all of the other considered benchmarks. There are two main possible reasons for this. Firstly, our approach is aimed at dynamic targets, i.e. unlike the others, it incorporates a forgetting factor that equips it with adaptive capabilities and enjoys dynamic regret guarantees. This allows it to deal better with such time-varying references to which it is hard or impossible to fit a time-invariant model better compared to the other static methods. Secondly, for such targets, the indirect approach of PLOT, i.e., learning the target dynamics and then incorporating these in the control action, is more advantageous than a direct approach deployed by all the other benchmarks. While the affine term of the optimal control action changes drastically throughout the horizon due to the sign and direction change of the target, the dynamics of the exosystem do not change as drastically. Thus, PLOT, learning the dynamics of the exosystem with forgetting is able to provide a better tracking performance, while DAP, Riccatitron, and SS-OGD learn a much smaller affine term for the controller. This results in their performance being indistinguishable from that of the naive LQR. The FTL algorithm aiming for an average best performance instead learns a control action that keeps



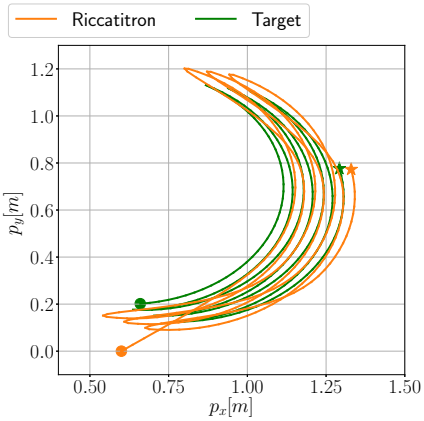
(a) PLOT (ours)

(b) FTL (Abbasi-Yadkori et al., 2014)

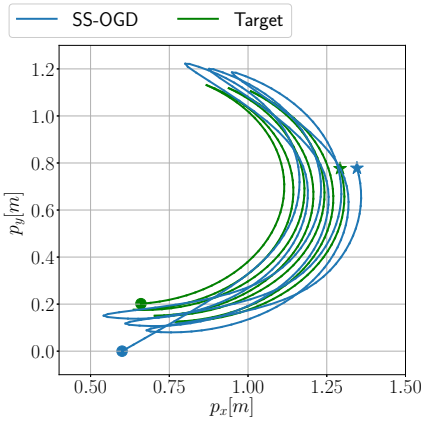
(c) Naive LQR



(d) DAP (Agarwal et al., 2019a)



(e) Riccatitron (Foster & Simchowitz, 2020)



(f) SS-OGD (Karapetyan et al., 2023)

Figure 15. Trajectory plots of a reference target with a $V_T = \mathcal{O}(\sqrt{T})$ following (32) tracked by PLOT and other online control algorithms for a $T = 30$ seconds long horizon.

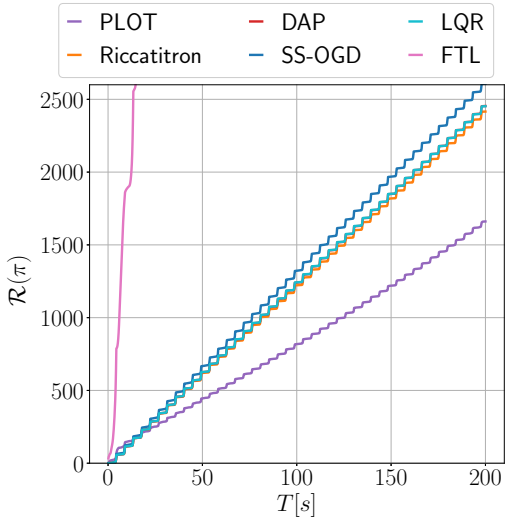


Figure 16. Dynamic Regret of Online Control algorithms applied to the online tracking problem of the unknown target following (32).

it roughly in the center of the evolving spiral target ending up with a dynamic regret of an order higher compared to the others.

Table 1 compares the average computational time for each iteration of the algorithm in milliseconds, the size of the memory that needs to be reserved and updated for the online execution of the algorithm, and the accumulated regret at the end of the given target trajectory. The average computational time is obtained by averaging the time it takes to compute the control action at each timestep over the entire horizon of 2000 timesteps. The memory for each controller shows the size of the past state and control variables that need to be stored in the memory. For example for the naive LQR controller, this is 0, while for SS-OGD it is 4, as at each timestep it requires the past control input.

Naive LQR, performing only a single matrix multiplication at each timestep is unsurprisingly the fastest, requiring no extra memory. The computational time of FTL and SS-OGD is of the same order, however, for this benchmark, both incur a higher cost and therefore a higher regret. When PLOT is implemented by approximating step 8 of Algorithm 1 with an unweighted Frobenious norm projection, it can be implemented by a thresholding operation on singular values, requiring no quadratic programs to be solved. In this case, PLOT, Riccatitron, and DAP have similar computation times but are all an order higher compared to the Naive LQR. While compared to it, Riccatitron and DAP attain almost the same regret, PLOT outperforms both, requires less memory than Riccatitron, and is around a millisecond faster than DAP. When the full weighted projection is carried out in Algorithm 1, step 8, by approximately solving it with a quadratic program consuming 9.3 milliseconds, PLOT attains the lowest regret.

Table 1. Comparison of Online Control Algorithms for Tracking the Unknown Target with Dynamics (32)

Algorithm	Comp. Time [ms]	Memory	Regret
PLOT (Ours)	10.7*	54	**1661**
PLOT (Ours, w/ unweighted projection)	2.10	54	1685
Riccatitron (Foster & Simchowitz, 2020)	2.30	90	2414
DAP (Agarwal et al., 2019a)	3.50	45	2454
FTL (Abbasi-Yadkori et al., 2014)	0.40	**0**	40004
SS-OGD (Karapetyan et al., 2023)	0.33	4	2608
Naive LQR	**0.25**	**0**	2453

* - 9.3 ms spent on the weighted projection with a QP solver.

F.3.1. Benchmark Implementation Details

PLOT: The prediction horizon is set to $W = 6$, the forgetting factor to $\gamma = 0.9$, and $M = 10$. The the augmented matrix is defined as $\tilde{S}_t \triangleq \begin{bmatrix} S_t & v_t \end{bmatrix}$ for $t = 1, \ldots, T-1$ as in Section 2, and for all $k = 1, \ldots, W$, the k learners are initialized as

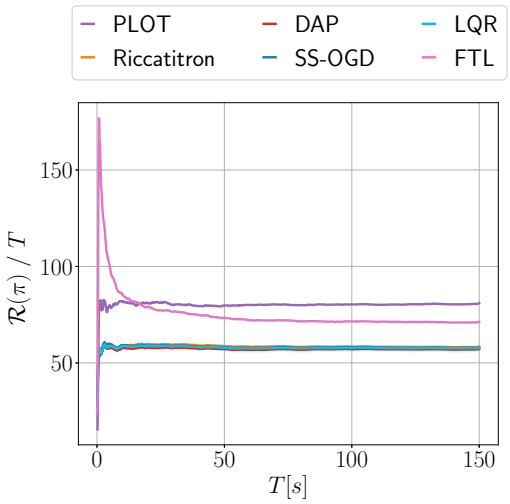


Figure 17. Average Dynamic Regret of Online Control algorithms applied to the online tracking problem of the unknown target following (33).

follows: $\hat{S}_{j+k|j} = \mathbf{I}_6, v_{j+k|j} = \mathbf{0}_6$ and $P_{j|j-k} = 10^{-4} \times \mathbf{I}_7$, for all $j = -1, \dots, k-2$.

Riccatitron: Following the notation of (Simchowitz, 2020), the horizon of the learners is set to $h = 5$ and the disturbance action policy length to $m = 5$ decided upon by a search over the best parameter. The latter choice increases the memory requirement of Riccatitron in Table 1 to achieve a good tracking performance. We disable the projection, but note that even with projection enabled the results are almost identical. The Online Newton Step (ONS) update is chosen, with The learning rate set to $\eta_{ons} = 0.2$ and $\varepsilon_{ons} = 1$.

DAP: As per the notation of (Agarwal et al., 2019a), and to match the memory allocation for PLOT, we fix the horizon length for the disturbance action policy to $H = 5$. The projection is performed as detailed in (Agarwal et al., 2019a), and the learning rate is set to $\frac{0.1}{\sqrt{T}}$.

FTL: The algorithm is implemented as detailed in (Abbasi-Yadkori et al., 2014) and contains no hyperparameters to be tuned.

SS-OGD: The affine control term is initialized with a vector of zeros, and the learning rate is set to $\alpha = 0.001 * \mathbf{I}_4$, following the notation of (Karapetyan et al., 2023).

Naive LQR: The naive LQR controller performs state feedback on the observed error

$$u_t^{LQR}(x_t) = -K(x_t - r_t), \qquad \forall t = 0, \dots T-1,$$

with K defined in (8).

F.3.2. Random Targets

The dependence of the dynamic regret of PLOT on V_T implies that for targets having a dynamic structure, including the one in the preceding example, PLOT's tracking performance is better compared to targets that lack dynamics. While the latter case is still covered by the structure imposed on the targets, as discussed in Section C, the regret can scale linearly with T in the worst case when r_t at each time step is independent from the others. In this case, the indirect approach of PLOT may result in a poorer performance compared to direct approaches like DAP, Riccatitron, or SS-OGD when γ, W, and M are not optimally tuned for the specific realization.

To showcase this marginal example, we consider random targets sampled from the uniform distribution

$$r_t \sim \mathcal{U}\left(\begin{bmatrix}-1\\-1\end{bmatrix}, \begin{bmatrix}1\\1\end{bmatrix}\right). \tag{33}$$

We compare PLOT with the same benchmark controllers whose tunable parameters are the same as in Section F.3.1. For PLOT we take $W = 2$, $\gamma = 0.02$, and $M = 0.7$, leaving the other parameters unchanged from Section F.3.1.

We perform 10 independent realizations of the random targets over a horizon of length $T = 150$ seconds and compute the averaged regret over these runs for each of the six considered algorithms, shown in Figure 17. The random reference updates combined with the low forgetting factor γ result in $\hat{S}_t$-s with large singular values, and, consequently, large inputs, while the optimal action in this case is staying close to the origin. While the performance can be improved significantly by lowering M and increasing γ, bringing the averaged regret to around 25, this example shows that for a completely random agent with no dynamics a direct approach may perform better than an indirect one like PLOT.

F.4. A Naive-RLS Controller

The multi-predictor setup of PLOT is crucial for its stability and regret performance. To showcase this we consider also a simpler controller that we refer to as Naive-RLS in Algorithm 4. In contrast to PLOT, it obtains W-step ahead dynamics predictions by recursively applying only the one step ahead estimate $\hat{S}_{t+1|t}$, resulting in $r_{t+1} = \hat{S}_{t+1|t} r_t,\ r_{t+2|t} = \hat{S}^2_{t+1|t} r_t,\ \ldots,\ r_{t+W|t} = \hat{S}^W_{t+1|t} r_t$, for $p = 1$. As detailed in Section 3, this can result in unstable predictions in particular when S_t is time-varying. For $p > 1$, we define $z_{t|t} = z_t$, $r_{t|t} = r_t$, and, for $k = 1, \ldots, W-1$

$$z_{t+k|t} = \begin{cases} \begin{bmatrix} r^\top_{t+k|t} & \cdots & r_{t|t} & r_{t-1} \cdots & r^\top_{t-p+k+1} \end{bmatrix}^\top & \text{if } p > k, \\ \begin{bmatrix} r^\top_{t+k|t} & \cdots & r^\top_{t-p+k+1|t} \end{bmatrix}^\top & \text{if } p \leq k. \end{cases}$$

Algorithm 4 Naive-RLS

Require: Horizon W, forgetting factor $\gamma \in (0, 1)$
Compute $X, K, \{K_0, ..., K_{W-1}\}$ as in (6), (8), (9).
Initialize 1 RLS learner according to the Algorithm 1, for $k = 1$.
for t = 0, ..., $T - 1$ **do**
  Observe system state x_t, target state r_t.
  for $k = 1, ..., W$ **do**
    if $t + k \leq T$ **then**
      Update the 1-step-ahead predictor and obtain $\hat{S}_{t+1|t}$ according to the Algorithm 1.
      Predict $r_{t+k|t} = \hat{S}_{t+1|t} z_{t+k-1|t}$
    else
      Set $r_{t+k-1|t} = 0$.
    end if
  end for
  Compute u^π_t as in (13)
end for

To demonstrate the failure of Naive-RLS for agents with time-varying dynamics, we consider the static (circular) and dynamic references of Sections F.2.1 and F.2.2, respectively. Figure 18 depicts the former case, showing the comparable performance of Naive-RLS to PLOT, since the static S remains stable even when exponentiated. In the latter case with a dynamic S_t, shown in Figure 19, Naive-RLS becomes unstable due to exploding matrices $\hat{S}_{t+k|t}$. The implementation details of both controllers are provided below.

PLOT: The prediction horizon is set to $W = 10$, the forgetting factor to $\gamma = 0.8$, and $M = 1$. The the augmented matrix is defined as $\tilde{S}_t \triangleq \begin{bmatrix} S_t & v_t \end{bmatrix}$ for $t = 1, \ldots, T-1$ as in Section 2, and for all $k = 1, \ldots, W$, the k learners are initialized as follows: $\hat{S}_{j+k|j} = \mathbf{I}_6, v_{j+k|j} = \mathbf{0}_6$ and $P_{j|j-k} = 10^{-4} \times \mathbf{I}_7$, for all $j = -1, \ldots, k-2$.

Naive-RLS: The prediction horizon is set to $W = 10$, the forgetting factor to $\gamma = 0.8$, and $M = 1$. The the augmented matrix is defined as $\tilde{S}_t \triangleq \begin{bmatrix} S_t & v_t \end{bmatrix}$ for $t = 1, \ldots, T-1$ as in Section 2, and for all $k = 1, \ldots, W$, the k learners are initialized as follows: $\hat{S}_{j+k|j} = \mathbf{I}_6, v_{j+k|j} = \mathbf{0}_6$ and $P_{j|j-k} = 10^{-4} \times \mathbf{I}_7$, for all $j = -1, \ldots, k-2$.

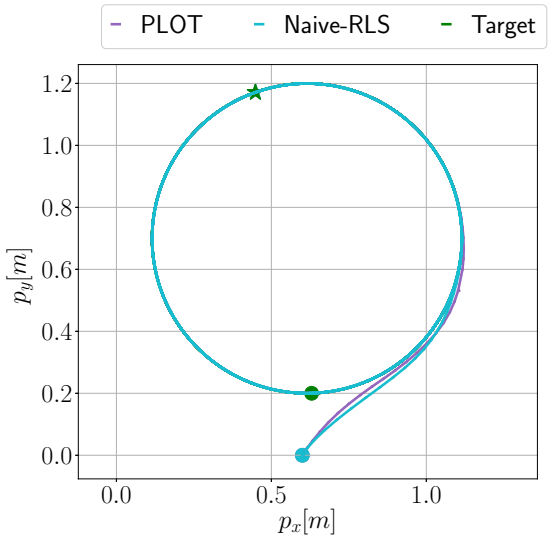


Figure 18. Naive-RLS shows comparable performance to PLOT on the static, circular target from Example C.2.

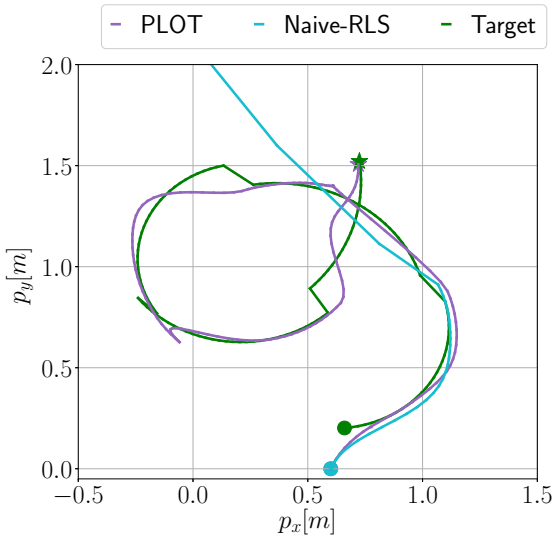


Figure 19. Naive-RLS becomes unstable while PLOT manages tracks the dynamic target of Section F.2.2.

G. Implementation on Quadrotors

The online tracking setting is of particular interest when applied to the control of quadrotors to track either a virtual target or another object of interest. We extend the simulation setup of Appendix F to a practical one by applying PLOT to track a virtual sequentially revealed target using a Crazyflie 2.1 Quadrotor (Bitcraze, 2023b) depicted in Figure 20. The controllers use the linearised model of the quadrotor and the derived in Appendix F.1. Below, we detail our experimental setup, as well as the results of the online tracking experiments.

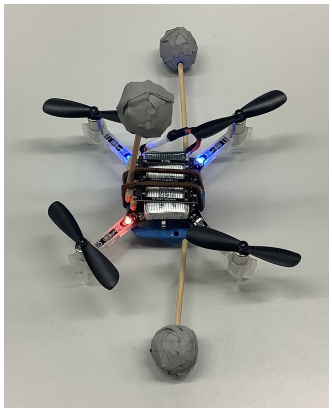


Figure 20. The Crazyflie 2.1 Quadrotor with three visual markers.

Figure 21. The D-FALL Laboratory of ETH Zürich, consisting of a dedicated Flying Arena and 5 (3 shown) mounted VICON Cameras for state estimation.

G.1. Experimental Setup

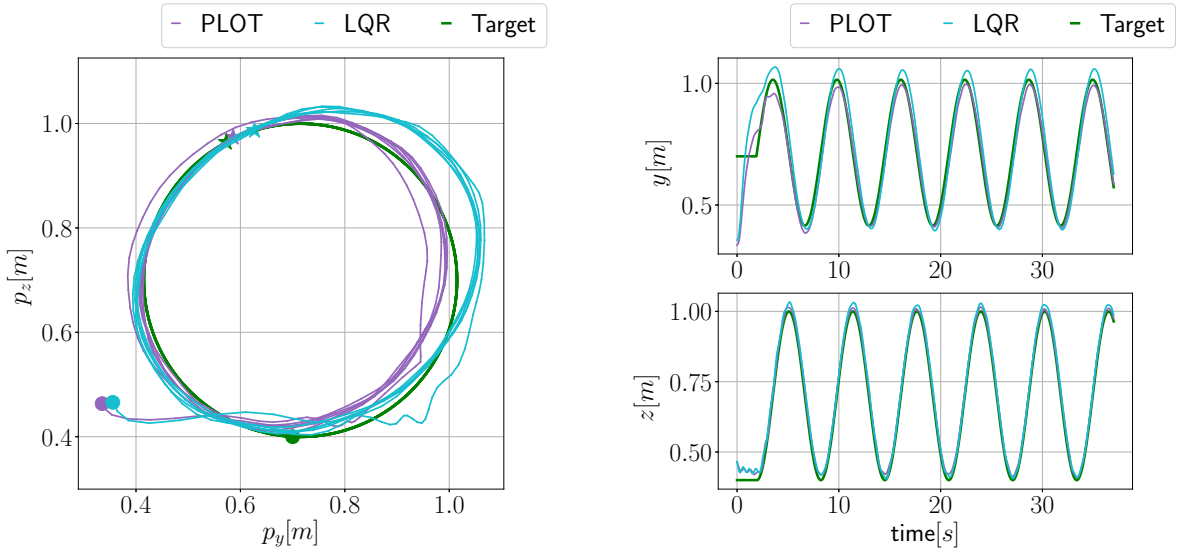
The Crazyflie quadrotors are equipped with a 3-axis accelerometer and gyroscope for onboard angular-rate control and 3 retro-reflective markers, as shown in Figure 20. These markers are detected by 5 VICON cameras mounted around the flying arena, shown in Figure 21, and the VICON system provides a spatial position estimate of the drone in its precalibrated reference frame of reference.

We set up the control architecture with the Robotic Operating System (ROS) (Quigley et al., 2009), which allows the controller to receive the state of the drone from the VICON mocap system in real-time. The control action is computed on a local computer and communicated with the drone by sending variable-sized packets through a 2.4Ghz USB dongle (Bitcraze, 2023a). The virtual reference trajectory is generated online after the drone has successfully taken off and is at a pre-defined random initial hovering position at a height of 0.4 meters. The controller receives the target state through the ROS network as it flies at a predetermined rate of 10 Hz.

G.2. Online Tracking Experiments

We run PLOT on the described setup to show its tracking performance. The cost matrices are taken to be the same as in Appendix F.1, the prediction horizon is fixed to $W = 5$ for the "infinity"-shape and $W = 3$ for the circle, the forgetting factor to $\gamma = 0.8$ and no projection is performed. The initialization of PLOT is the same as in Appendix F.3.1. Two virtual reference target shapes, a circle and an "infinity"-shape are generated online and published to the ROS network. The quadrotors start in the flying arena in Figure 21 and follow the target as soon as state information is received. Figures 22 and 23 below show the trajectory and state plots of PLOT and Naive LQR implemented for a horizon of $T = 40$ seconds for the circle and "infinity"-shape reference trajectories, respectively. Naive LQR is implemented as described in Appendix F.3.1.

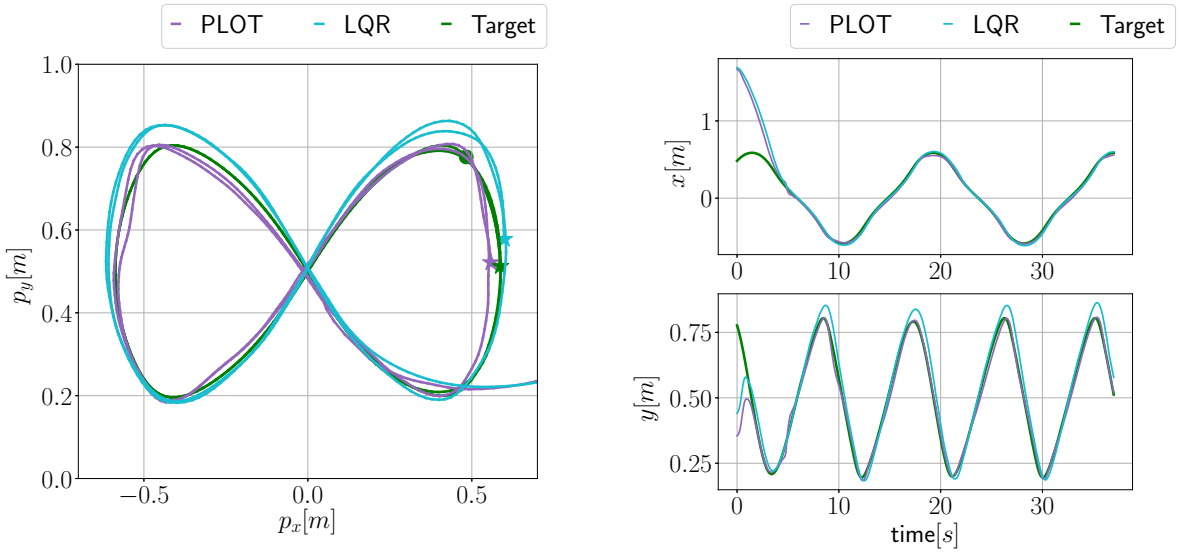
As in the simulations, without any affine term, the naive LQR controller exhibits a delayed tracking behavior as expected. The constant offset can be noted clearly for both shapes. In comparison, the PLOT algorithm experiences a smaller tracking error. As opposed to the linear simulations in Appendix F, the nonlinear hardware implementation has additional errors, especially for the circular example, due to imperfect mass value or linearization error, among other reasons. The VICON system is also known to introduce a drift through its localization algorithm when some of the cameras are malfunctioning.



(a) Trajectory plot.

(b) State plot.

Figure 22. Circular reference tracked by Crazyflie Drones with the PLOT and Naive LQR controllers.



(a) Trajectory plot.

(b) State plot.

Figure 23. “Infinity”-shaped reference tracked by Crazyflie Drones with the PLOT and Naive LQR controllers.