Performance-driven Constrained Optimal Auto-Tuner for MPC

Albert Gassol Puigjaner, Manish Prajapat, Andrea Carron and Melanie N. Zeilinger

***Abstract*— One of the significant challenges in Model Predictive Control (MPC) is the safe tuning of its cost function parameters. In safe tuning for MPC, the goal is to find the cost function parameters that maximize the system's performance while ensuring that the performance stays consistently above a given threshold. In this context, we propose the Constrained Optimal Auto-Tuner for MPC algorithm (COAT-MPC), a method that safely explores the cost function parameters domain to reach the most performant parameters. COAT-MPC makes use of Upper Confidence Bounds (UCB) on the entire parameters' domain as the goal for each optimization iteration, and sequentially explores the parameter space towards this goal. We present an in-depth theoretical analysis of our proposed method, establishing its safety with high probability and demonstrating provable finite-time convergence. We perform comprehensive simulations and comparative analyses with a hardware platform against classical Bayesian Optimization (BO) and state-of-the-art methods. With these experiments, we demonstrate that our approach outperforms these competitive baselines in terms of fewer constraint violations and improved cumulative regret over time in the autonomous racing scenario. To the best of the authors' knowledge, this research represents the first successful implementation of safe tuning in MPC. Additionally, we open-sourced the code of the proposed method[1].**

I. Introduction

Model Predictive Control (MPC) is a prominent optimization-based control framework that can handle constraints and optimize system performance by predicting the system's future behaviour. MPC is widely used in many robotic applications such as autonomous driving [1], four-legged robots [2], and bipedal robots [3]. While MPC is a successful optimal control technique, one of the significant challenges in its implementation is the tuning of the cost function parameters. The cost function, which defines the control objectives, plays a crucial role in the performance of the MPC. However, designing a cost function that balances competing objectives is a non-trivial task that typically requires significant trial and error. Moreover, the cost function parameters often depend on the specific environment and system dynamics, making it difficult to design a single set of parameters that can perform well in all scenarios. Usually, the task of fine-tuning cost function parameters involves heuristic methods and demands expert knowledge, typically leading to a significant number of costly experimental iterations.

In almost all the applications, we tune to maximize some performance function, e.g., while tuning for racing, we optimize the lap time. Unfortunately, often these performance functions are *a-priori* unknown and need to be learned through data. Naively, to find the optimal parameters, one may try out all the parameters in a brute force approach, however, this is highly inefficient and may lead the system to halt with most of the parameters. For example, in tuning an MPC for autonomous racing, it is undesirable to use parameters that make the car move extremely slow, or even stop before finishing a lap. To this end, our goal is to develop an algorithm that automatically tunes the MPC cost function weights and converges to the optimal set of tuning parameters while ensuring that a desired performance function is above a threshold throughout the process. Thus, also eliminates inefficiencies from trial and error, and prevents undesirable effects such as slowing down or halting in the tuning process.

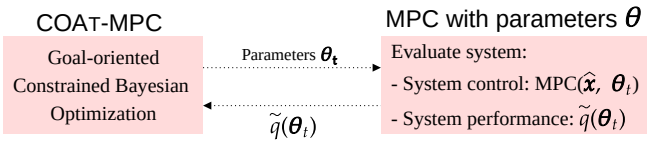


Fig. 1: COAT-MPC algorithm overview. The constrained Bayesian Optimization proposes a new set of cost function weights θ_t, which are evaluated on the system. The algorithm captures a noisy performance function sample, which is used to update the Bayesian optimization surrogate model posterior and acquire a new set of cost function weights. The process is repeated until convergence.

To tackle this, we propose a novel algorithm: COAT-MPC, Constrained Optimal Auto-Tuner for Model Predictive Control. COAT-MPC safely explores the space of parameters and builds a belief about the *a-priori* unknown performance function through data, utilizing tools from Gaussian processes [4]. COAT-MPC incorporates safe exploration ideas from [5], [6], [7] and recursively recommends sufficiently informative parameters that ensure exploration while satisfying the performance constraint. We establish convergence guarantees to the optimal tuning parameters in a finite number of samples while ensuring performance constraint satisfaction with an arbitrarily high probability. For finite time convergence, we present a sample complexity bound by extending the analysis of [7] from continuous to discrete domains. In particular, our sample complexity result removes an explicit dependence on the discretization size and thus significantly improves prior safe exploration results in discrete domains [6], [8], [9].

Finally, we demonstrate the effectiveness of COAT-MPC in the challenging application of autonomous racing. We tune a Model Predictive Contouring Control [10] formulation with the objective of optimizing the lap time while avoiding undesirable effects such as halting. Our evaluation encompasses a comprehensive analysis in both, simulation and on a 1:28 scale RC racecar [11]. We present a comparative analysis against other automatic tuning methods including classical

{agassol,manish.prajapat,
carrona,mzeilinger}@ethz.ch

[1]Code: https://github.com/albertgassol1/coat_mpc.

Bayesian optimization. The results demonstrate that our approach outperforms other methods in terms of the number of constraint violations and yields an improved cumulative regret over time. To the best of our knowledge, this paper is the first that presents a safe exploration algorithm for MPC tuning while having strong theoretical guarantees.

II. Related works

Controller tuning has been an active area of research in the field of robotics and different methods have arisen in the literature. For instance, optimization-based approaches have proven to be effective in tuning the parameters of robotic systems. These methods involve iteratively optimizing a user-defined objective function through experiments [12], [13]. At each iteration, the objective function gradients are calculated and used to update the controller parameters. Despite the efficacy of these methods, they require the availability of an analytical objective function and the computation of its gradients, which are often not available.

Recently, data-driven methods aimed at learning the relationship between system parameters and a desired metric have emerged as promising solutions for automatic tuning. For instance, methods such as the Metropolis-Hastings algorithm [14] and Policy Search methods [15], have demonstrated state-of-the-art results in model-based agile flight control.

Bayesian Optimization [16], [17] has been particularly successful due to its ability to model the objective function using a limited number of samples [18]. Bayesian Optimization is a global optimization method that can efficiently find the optimal solution of a black-box continuous function. It utilizes a surrogate probabilistic model to represent the objective function, which is updated as new data is acquired. In the context of MPC parameter tuning, BO can be used to efficiently explore the parameter space and find the optimal cost function parameters for a given platform and environment [19] by selecting a proper objective function. This can lead to improved performance while reducing the time and effort required for manual tuning. Additionally, subsequent works have introduced contextual information from the environment or system [20], as well as considering the confidence of the surrogate model to enhance the convergance rate of BO [20]. However, it is worth noting that these BO methods do not take into account constraints on the objective function.

While Bayesian Optimization is a powerful optimization method for MPC parameter tuning, it has a limitation when it comes to handling constraints. The standard BO algorithm does not take into account constraints on its surrogate model, which can lead to parameters that produce very poor performance due to the unbounded exploration of the algorithm. This is particularly problematic in the context of robotics, where safety and performance during testing are of paramount importance.

Several approaches have been developed to incorporate constraints into the BO algorithm. One such approach involves utilizing a variant of the Expected Improvement (EI) function, referred to as the Constrained Expected Improvement (EI_C) [21], [22]. This approach involves modeling the constraint function with a prior distribution and incorporating a probability of violation into the acquisition function. It has been shown to be effective in tuning MPC systems [23]. However, none of the aforementioned methods provide a guarantee of constraint satisfaction, which may result into evaluating poor performance parameters.

In the literature of Constrained Bayesian Optimization (CBO), SAFE-OPT [8], [5], [24] is introduced as an algorithm that aims to provide high-probability guarantees of constraint satisfaction. The algorithm leverages the regularity assumption on the objective function and the Lipschitz continuity to identify a set within the parameter domain where the constraints on the underlying objective function are unlikely to be violated. Even though SAFE-OPT has been proven to guarantee safety, it tends to explore the complete safe parameter region, consequently leading to sample inefficiency in relation to the optimization task.

In order to tackle the sample inefficiency issues of safe exploration, the authors of [6] propose GOOSE, a goal-oriented safe exploration algorithm for any interactive machine learning methods. GOOSE leverages the regularity assumption on the constraint function to define over- and under-approximations of the safe set. A goal within the over-approximated set is defined at each iteration with the purpose of steering the recommendations of GOOSE towards the goal while ensuring safety.

In this paper, we propose to exploit goal-oriented safe exploration to tackle the problem of tuning the cost function weights of an MPC.

III. Problem statement

We consider a non-linear dynamical system controlled using an MPC with cost function parameters $\boldsymbol{\theta} \in \mathbb{R}^{N_\theta}$, see Section IV-A for details on MPC. We define a function $q : D \to \mathbb{R}$, where $D \subseteq \mathbb{R}^{N_\theta}$ is a finite domain of cost function parameters, that measures the performance of a given set of tuning parameters. The performance function $q(\boldsymbol{\theta})$ is *a-priori* unknown and needs to be learned with data. To learn the performance function, at any iteration n, one can control the system with an MPC using any parameter $\boldsymbol{\theta}_n \in D$ and obtain a noisy observation of $q(\boldsymbol{\theta}_n)$. We examine the problem of finding the parameters that maximize q while ensuring that the performance is above a user-specified performance threshold τ in all iterations, i.e., $q(\boldsymbol{\theta}_n) \geq \tau, \forall n \geq 1$. Ideally, we do not want to execute all parameters, but only those that are essential to guarantee convergence to optimal parameters while always satisfying constraints.

Clearly, without making any assumptions, this is not possible, since we do not even know if the initial set of parameters will satisfy the performance constraint. Therefore, we make the following safe seed assumption.

Assumption 1 (Safe seed): An initial safe set of parameters $\mathcal{S}_0 \subset D$ is known, i.e., $\forall \boldsymbol{\theta} \in \mathcal{S}_0, q(\boldsymbol{\theta}) \geq \tau$.
This assumption can be readily ensured by having an MPC controller that can control the system to obtain measurements

of performance function q. E.g, in autonomous racing, parameters of an MPC controller (need not be optimized) that can drive the car to finish the lap will satisfy assumption 1.

Next, since the function q is *a-priori* unknown, for exploration, we need a mechanism such that knowing about q at certain $\boldsymbol{\theta}$ provides us with some information about the neighboring region. To this end, we make some regularity assumptions on the objective function q.

Assumption 2: The domain D is endowed with a positive definite kernel $k_q(\cdot,\cdot)$, and that q has a bounded norm in the associated Reproducing Kernel Hilbert Space (RKHS) [25], $||q||_k \leq B_q < \infty$.

This assumption allows us to use Gaussian Processes (GPs) [4] to model the objective function q, see Section IV-B for more details on GPs. It captures the property that similar parameters lead to similar performance outcomes. This assumption is common in several prior Bayesian Optimization works that use GPs to model the unknown function [20], [6], [8]. We consider that constraint q is L-Lipschitz continuous with respect to the metric d on D. This is automatically satisfied, for e.g., while using common isotropic kernels, such as the Mátern and Gaussian kernels.

Using assumptions assumptions 1 and 2, we construct a true reachable set up to user-defined precision ϵ denoted by $S^{q,\epsilon}$. This set includes all the parameters that can be reached starting from safe seed $\mathcal{S}_0$, while always being safe with ϵ margin, i.e, $q(\boldsymbol{\theta}) \geq \tau + \epsilon$ (see Section IV-C for details on how to construct this set). Thus, for the problem of safe controller tuning, the best any tuning algorithm can attain is the following,

$$\boldsymbol{\theta}^* := \underset{\boldsymbol{\theta}\in S^{q,\epsilon}}{\arg\max} \quad q(\boldsymbol{\theta}). \tag{1}$$

COAT-MPC objective. To conclude the problem statement, we want to guarantee convergence to a $\boldsymbol{\theta}^g$, which satisfies $q(\boldsymbol{\theta}^g) \geq q(\boldsymbol{\theta}^*) - \epsilon$, where $\epsilon > 0$ is a user-defined accuracy. We shall ensure constraint satisfaction for all iterations, i.e., $q(\boldsymbol{\theta}_n) \geq \tau, \forall n \geq 1$ and should complete the tuning process in a finite number of samples.

IV. Background

In this section, we first explain the underlying MPC used to control the system in Section IV-A, then we delve into Gaussian Processes in Section IV-B which is used to model the unknown function q. Finally, we introduce concepts of safe exploration that we utilize in COAT-MPC in Section IV-C.

A. Model predictive control (MPC)

In MPC, a controller optimizes the system's predicted performance by minimizing a defined cost function while ensuring that constraints are satisfied. In particular, an MPC can be formulated as follows:

$$\begin{aligned} \boldsymbol{X}^*, \boldsymbol{U}^* = \arg\min_{\boldsymbol{X},\boldsymbol{U}} \quad & \sum_{k=1}^{N} l(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{\theta}) \\ \text{s.t.} \quad & \boldsymbol{x}_0 = \hat{\boldsymbol{x}},\ \boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) \\ & \boldsymbol{x}_k \in \mathcal{X},\ \boldsymbol{u}_k \in \mathcal{U} \end{aligned} \tag{2}$$

where $\boldsymbol{x}_k \in \mathbb{R}^{N_x}$ is the system state with N_x states, $\boldsymbol{u}_k \in \mathbb{R}^{N_u}$ is the control input with N_u controls, $\boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) : \mathbb{R}^{N_x} \times \mathbb{R}^{N_u} \to \mathbb{R}^{N_x}$ denotes the system dynamics, and $l(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{\theta}) : \mathbb{R}^{N_x} \times \mathbb{R}^{N_u} \times \mathbb{R}^{N_\theta} \to \mathbb{R}$ is the cost function. The weights $\boldsymbol{\theta}$ shape the cost function and play a crucial role in the system's performance.

B. Gaussian processes

Gaussian Processes (GP) [4] are probability distributions over a class of continuous smooth functions. GPs are characterized by mean $\mu : \mathbb{R}^{N_\theta} \to \mathbb{R}$ and a kernel function $k : \mathbb{R}^{N_\theta} \times \mathbb{R}^{N_\theta} \to \mathbb{R}$, which captures the notion of similarity between data points. Without loss of generality, we normalize such that $k(\boldsymbol{\theta}, \boldsymbol{\theta}) \leq 1, \forall \boldsymbol{\theta} \in \mathbb{R}^{N_\theta}$. Given a set of n noisy samples collected at $A_n = \{\boldsymbol{\theta}_i\}_{i=1}^n$ perturbed by $\eta_n \sim Q_\eta$ i.i.d. σ-sub-Gaussian noise given by $\boldsymbol{y}_n = [q(\boldsymbol{\theta}_1) + \eta_1, \ldots, q(\boldsymbol{\theta}_n) + \eta_n]^\top$, we can compute the posterior over q in closed form using,

$$\begin{aligned} \mu_q^{(n)}(\boldsymbol{\theta}) =& {\boldsymbol{k}_q^{(n)}}^\top(\boldsymbol{\theta})(\boldsymbol{K}_q^{(n)} + \boldsymbol{I}_n\sigma_\eta^2)^{-1}\boldsymbol{y}_n \\ k_q^{(n)}(\boldsymbol{\theta}, \boldsymbol{\theta}') =& k_q^{(n)}(\boldsymbol{\theta}, \boldsymbol{\theta}') - \\ & {\boldsymbol{k}_q^{(n)}}^\top(\boldsymbol{\theta})(\boldsymbol{K}_q^{(n)} + \boldsymbol{I}_n\sigma_\eta^2)^{-1}\boldsymbol{k}_q^{(n)}(\boldsymbol{\theta}') \\ \sigma_q^{(n)}(\boldsymbol{\theta}) =& \sqrt{k_q^{(n)}(\boldsymbol{\theta}, \boldsymbol{\theta})} \end{aligned}$$

where the covariance matrix $\boldsymbol{K}_q^{(n)}$ is defined as $\boldsymbol{K}_q^{(n)}(i,j) = k_q^{(n)}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j),\ i,j \in \{1,\ldots,n\}$, and $\boldsymbol{k}_q^{(n)}(\boldsymbol{\theta}) = [k_q^{(n)}(\boldsymbol{\theta}_1, \boldsymbol{\theta}), \ldots, k_q^{(n)}(\boldsymbol{\theta}_n, \boldsymbol{\theta})]^\top$ and $\sigma_q^{(n)} : \mathbb{R}^{N_\theta} \to \mathbb{R}$ denotes the predictive variance.

Additionally, we define the maximum *information capacity* $\gamma_n := \sup_{A\subseteq D:|A|\leq n} I(\boldsymbol{y}_A; q)$ associated with the kernel k_q, where $I(\boldsymbol{y}_A; q_A)$ denotes the mutual information between q evaluated at locations in the set A and the noisy samples $\boldsymbol{y}_A$ collected at A [26]. Intuitively, it quantifies a best case scenario where we can select the measurements in the most informative manner. This definition allows us to build up on safe exploration in the following section.

C. Safe exploration

In this section, we introduce the necessary tools from prior works [6], [8] required to safely explore the domain of cost function weights efficiently. In order to safely explore the domain, we need to reason about the parameters that could eventually be included within our safe set of parameters, as well as continuously evaluate the parameters which have already been classified as safe.

Reachable set and safe set approximations. To this end, by utilizing the GP posterior (**??**), we construct interesting lower and upper confidence bounds on q at each iteration $n \geq 1$ as:

$$\begin{aligned} l_n(\boldsymbol{\theta}) &:= \max\left(l_{n-1}(\boldsymbol{\theta}), \mu_q^{(n-1)}(\boldsymbol{\theta}) - \beta_n\sigma_q^{(n-1)}(\boldsymbol{\theta})\right), \\ u_n(\boldsymbol{\theta}) &:= \min\left(u_{n-1}(\boldsymbol{\theta}), \mu_q^{(n-1)}(\boldsymbol{\theta}) + \beta_n\sigma_q^{(n-1)}(\boldsymbol{\theta})\right), \end{aligned}$$

with $l_0(\boldsymbol{\theta}) = \mu_q^{(0)}(\boldsymbol{\theta}) - \beta_1\sigma_q^0(\boldsymbol{\theta}))$ and $u_0(\boldsymbol{\theta}) = \mu_q^{(0)}(\boldsymbol{\theta}) + \beta_1\sigma_q^0(\boldsymbol{\theta}))$. Note that by construction with intersecting con-

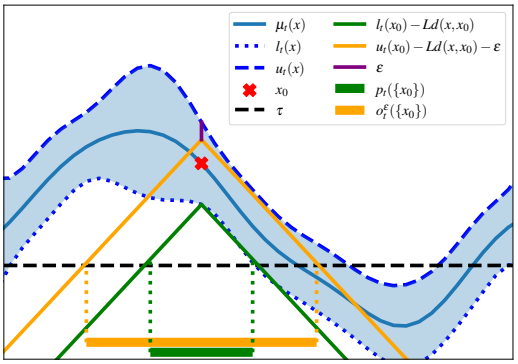


Fig. 2: Pessimistic and optimistic operators evaluated at x_0. The operators make use of the GP upper and lower confidence bounds, as well as the L-Lipschitz continuity.

fidence bounds, $l_n(\cdot)$ is non-increasing and $u_n(\cdot)$ is non-decreasing function in n, i.e.,

$$l_{n+1}(\boldsymbol{\theta}) \leq l_n(\boldsymbol{\theta}), u_{n+1}(\boldsymbol{\theta}) \geq u_n(\boldsymbol{\theta}) \; \forall \boldsymbol{\theta} \in D.$$

Using the intersecting confidence bounds and the GPs error bounds from Theorem 2 of [27], the following corollary [24] follows directly:

Corollary 1 (Theorem 2 [27]): Let assumption 2 hold. If $\sqrt{\beta_n} = B + 4\sigma\sqrt{\gamma_n + 1 + \ln(1/\delta)}$, it holds that $l_n(\boldsymbol{\theta}) \leq q(\boldsymbol{\theta}) \leq u_n(\boldsymbol{\theta}), \forall \boldsymbol{\theta} \in \mathbb{R}^{N_\theta}$ with probability at least $1-\delta$.

Throughout this work, we implicitly use $\sqrt{\beta_n}$ from Corollary 1. Similarly to GoOSE [6], we next define a one-step reachability operator exploiting the L-Lipschitz continuity of q and build a pessimistic and an optimistic constraint satisfaction operators over it using high probability confidence bounds.

$$r^\epsilon(\mathcal{S}) = \{\boldsymbol{\theta} \in D \,|\, \exists \boldsymbol{\theta}' \in \mathcal{S} : q(\boldsymbol{\theta}') - \epsilon - Ld(\boldsymbol{\theta}, \boldsymbol{\theta}') \geq \tau\}$$
$$p_n(\mathcal{S}) = \{\boldsymbol{\theta} \in D \,|\, \exists \boldsymbol{\theta}' \in \mathcal{S} : l_n(\boldsymbol{\theta}') - Ld(\boldsymbol{\theta}, \boldsymbol{\theta}') \geq \tau\}$$
$$o_n^\epsilon(\mathcal{S}) = \{\boldsymbol{\theta} \in D \,|\, \exists \boldsymbol{\theta}' \in \mathcal{S} : u_n(\boldsymbol{\theta}') - Ld(\boldsymbol{\theta}, \boldsymbol{\theta}') - \epsilon \geq \tau\}$$

A visual representation of the pessimistic and optimistic operators evaluated at a single point is depicted in Fig. 2. For notational convenience we denote $r(\mathcal{S}) := r^0(\mathcal{S})$ when referring to $\epsilon = 0$ case (analogously for the pessimistic and the optimistic operator as well). By applying these one-step safety operators recursively, we next define the pessimistic, optimistic and reachability expansion operators as:

$$\tilde{R}^\epsilon(\mathcal{S}) = \lim_{m\to\infty} R^{\epsilon,m}(\mathcal{S}) \tag{3}$$
$$\tilde{P}_n(\mathcal{S}) = \lim_{m\to\infty} P_n^m(\mathcal{S}) \tag{4}$$
$$\tilde{O}_n^\epsilon(\mathcal{S}) = \lim_{m\to\infty} O_n^{\epsilon,m}(\mathcal{S}) \tag{5}$$

where $P_n^m(\mathcal{S}) := p_n(p_n \cdots (p_n(\mathcal{S})))$ and $O_n^{\epsilon,m}(\mathcal{S}) := o_n^\epsilon(o_n^\epsilon \cdots (o_n^\epsilon(\mathcal{S})))$ are the m-step pessimistic and optimistic expansion operators. Using the expansion operators on $\mathcal{S}_{n-1}^p$ we obtain a pessimistic $\mathcal{S}_n^p = \tilde{P}_n(\mathcal{S}_{n-1}^p)$ and an optimistic $\mathcal{S}_n^{o,\epsilon} = \tilde{O}_n^\epsilon(\mathcal{S}_{n-1}^p)$ estimates of the true safe set. Analogously, $R^{\epsilon,m}(\mathcal{S}) := r^\epsilon(r^\epsilon \cdots (r^\epsilon(\mathcal{S})))$ denotes the the m-step ϵ close true reachability operator. Applying the reachability operator from Eq. (3) on the initial safe set $\mathcal{S}_0$, we obtain the $\epsilon-$close true reachability set $S^{q,\epsilon} = \tilde{R}^\epsilon(\mathcal{S}_0)$, which includes all the parameters while being at least ϵ conservative from violating the constraint.

V. COAT-MPC

In this section we present our novel algorithm COAT-MPC, for the optimization of MPC cost function parameters while respecting the performance constraint. The algorithm is presented in Algorithm 2 with its optimality guarantees deferred to Section VI.

Intuition. To guarantee safety, we construct a pessimistic and an optimistic set at each iteration, $\mathcal{S}_n^p$ and $\mathcal{S}_n^{o,\epsilon}$, using the pessimistic and optimistic expansion operators of Eqs. (4) and (5). We define a goal, $\boldsymbol{\theta}_n^g$, within the optimistic set as our target recommendation. COAT-MPC recommends $\boldsymbol{\theta}_n^g$ only if it is included in the pessimistic set, otherwise, the algorithm performs a safe expansion to learn about the safety of the goal. While GoOSE performs the safe expansion until $\boldsymbol{\theta}_n^g \in \mathcal{S}_n^p$ or $\boldsymbol{\theta}_n^g \notin \mathcal{S}_n^{o,\epsilon}$, COAT-MPC recomputes $\boldsymbol{\theta}_n^g$ after each time iteration, thus only performing one safe expansion iteration. Furthermore, COAT-MPC's safe expansion consists of sampling the closest parameter to the goal within the pessimistic set, instead of sampling at the most uncertain expander. A visualization of COAT-MPC in the 1D setting is depicted in Figure 3.

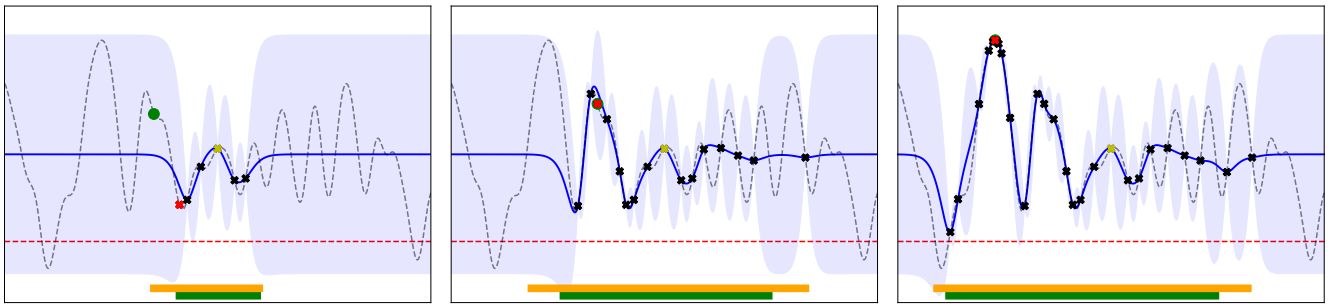
Safe Expansion. COAT-MPC's safe expansion strategy, outlined in Algorithm 1, consists of recommending the closest point to the goal, with respect to the Euclidean distance, inside the pessimistic set and that is not ϵ-accurate. Thus, COAT-MPC cautiously recommends safe parameters that are as close as possible to the goal, while maintaining exploration through the statistical confidence ϵ. Thus, if the algorithm is certain enough about the performance q of a parameter, it will not futher explore it.

Algorithm 1 Safe Expansion (SE)

1: **Input:** $\mathcal{S}_n^p, \boldsymbol{\theta}_n^g$
2: **Recommend:** $\arg\min_{\boldsymbol{\theta} \in \mathcal{S}_n^p} ||\boldsymbol{\theta}_n^g - \boldsymbol{\theta}||_2$, s.t. $w_n(\boldsymbol{\theta}) \geq \epsilon$

Phases of COAT-MPC. Following assumption 1, we initialize the algorithm with a safe parameter seed, $\mathcal{S}_0$, where the constraint $q(\boldsymbol{\theta}) \geq \tau, \forall \boldsymbol{\theta} \in \mathcal{S}_0$ is known to be satisfied. In Line 2, we initialize the pessimistic set to the safe seed $\mathcal{S}_0$ and the optimistic set to the parameters domain D. At the beginning of every iteration, we compute the goal $\boldsymbol{\theta}_n^g$ by means of the Upper Confidence Bounds (UCB) over the optimistic set (Line 4) and the width of the confidence bounds $w_{n-1}(\boldsymbol{\theta}) = u_{n-1}(\boldsymbol{\theta}) - l_{n-1}(\boldsymbol{\theta})$. We then distinguish between three different cases depending on weather the goal is in the pessimistic set and on the knowledge that we currently have about it.

1) **The goal is in the pessimistic set and it is ϵ-accurate** ($w_{n-1}(\boldsymbol{\theta}_n^g) < \epsilon$): we have reached our goal (Lines 5 and 6), since we know that the goal is safe and we have explored the space up to the statistical confidence.



(a) COAT-MPC at $n = 5$ (b) COAT-MPC at $n = 15$ (c) COAT-MPC at $n = 28$. Final iteration.

Fig. 3: COAT-MPC illustration. The algorithm learns the pessimistic (green bar) and optimistic (orange bar) sets, and safely explores the parameter space. At $n = 5$, the goal is outside of the pessimistic set, although it is inside the optimistic set. COAT-MPC expands the pessimistic set by approaching the goal. It reaches the goal at $n = 15$, and by further expanding the sets, it discovers the maximum of the function at $n = 28$. (i) The gray, dashed line represents the true function. (ii) The red, dashed line represents the constraint. (iii) The blue line represents the Gaussian Process mean, and the shaded blue area represents the confidence bounds ($\mu_n(\boldsymbol{\theta}) \pm \beta_n \sigma_n(\boldsymbol{\theta})$). (iv) The cross markers represent the samples, with yellow denoting the first sample and red the COAT-MPC recommended sample. (v) The green dot denotes the goal at each iteration.

2) **The goal is in the pessimistic set but it is not ϵ-accurate**: we recommend the goal and update the GP after getting a noisy evaluation of q to update our confidence about the goal (Lines 7 and 8).
3) **The goal is not in the pessimistic set**: we trigger Safe Expansion, Algorithm 1, and update the GP after getting a noisy evaluation of q (Lines 10 and 11).

If the algorithm has not terminated, the pessimistic and optimistic sets are updated using their expansion operators (Lines 13 and 14). This way, we safely expand the possible safe parameter candidates and safely explore the parameters domain.

Algorithm 2 COAT-MPC

1: **Input:** Safe seed $\mathcal{S}_0$, $q \sim \mathcal{GP}(\mu_0(\boldsymbol{\theta}), k_0(\boldsymbol{\theta}, \boldsymbol{\theta}^{'}))$, τ, D, Lipschitz constant L
2: $\mathcal{S}_0^p \leftarrow \mathcal{S}_0$, $\mathcal{S}_0^{o,\epsilon} \leftarrow D$
3: **for** $n = 1, \dots N_{max}$, **do**
4: $\quad \boldsymbol{\theta}_n^g \leftarrow \arg\max_{\theta \in \mathcal{S}_{n-1}^{o,\epsilon}} \mu_q^{(n-1)}(\theta) + \beta_n^{1/2} \sigma_q^{(n-1)}(\theta)$
5: $\quad$ **if** $\boldsymbol{\theta}_n^g \in \mathcal{S}_{n-1}^p$ and $w_{n-1}(\boldsymbol{\theta}_n^g) < \epsilon$ **then**
6: $\quad\quad$ Terminate
7: $\quad$ **else if** $\boldsymbol{\theta}_n^g \in \mathcal{S}_{n-1}^p$ **then**
8: $\quad\quad$ $y_n \leftarrow q(\boldsymbol{\theta}_n) + \eta_n$ and Update GP
9: $\quad$ **else**
10: $\quad\quad$ $\boldsymbol{\theta}_n \leftarrow \text{SE}(\mathcal{S}_{n-1}^p, \mathcal{S}_{n-1}^{o,\epsilon}, \boldsymbol{\theta}_n^g)$
11: $\quad\quad$ $y_n \leftarrow q(\boldsymbol{\theta}_n) + \eta_n$ and Update GP
12: $\quad$ **end if**
13: $\quad$ $\mathcal{S}_n^p \leftarrow \tilde{P}_n(\mathcal{S}_{n-1}^p)$
14: $\quad$ $\mathcal{S}_n^{o,\epsilon} \leftarrow \tilde{O}_n^{\epsilon}(\mathcal{S}_{n-1}^p)$
15: **end for**
16: **Recommend:** $\boldsymbol{\theta}_n^g$

VI. Theoretical Analysis

In this section, we present our core theoretical result, i.e., convergence to optimal tuning parameters while ensuring performance constraint in finite time with high probability. For the sample complexity result, we make the following regularity assumption which is easy to ensure by using a suitable kernel.

Assumption 3: $\beta_n \gamma_n$ grows sublinear in n, i.e., $\beta_n \gamma_n < \mathcal{O}(n)$.

Such assumptions are common in most prior works [9], [6] aimed to establish sample complexity or sublinear regret results and are not restrictive. This can be satisfied for commonly used kernels, e.g., linear kernels, squared exponential, Matern, etc., with sufficient eigen decay [28], [16] under the bounded B_q assumption 2.

Theorem 1: Let assumptions 1 to 3 holds and $n^\star$ be the largest integer such that $\frac{n^\star}{\beta_{n^\star}\gamma_{n^\star}} \leq \frac{C_1}{\epsilon^2}$ with $C_1 \coloneqq 8/\log(1+\sigma^{-2})$. The recommendation of COAT-MPC at iteration n, $\hat{\boldsymbol{\theta}}_n$, satisfies $q(\hat{\boldsymbol{\theta}}_{\boldsymbol{n}}) \geq \tau, \forall n \geq 1$ and the closed loop system (**??**) satisfies state and input constraints for all times. Furthermore, $\exists n \leq n^\star$ such that the following holds with probability at least $1 - \delta$:

$$q(\hat{\boldsymbol{\theta}}_n) \geq \max_{\theta \in \bar{R}_\epsilon(S_0)} q(\theta) - \epsilon.$$

The proof is in Appendix A. The theorem makes two statements: firstly, with high probability, *any algorithm* that samples at $u_n(\boldsymbol{\theta}) - l_n(\boldsymbol{\theta}) \geq \epsilon$ in $\mathcal{S}_n$ will explore the full safe domain. Secondly, if sampled as per UCB while respecting the uncertainty constraint, we are guaranteed to converge to the optimal solution. In contrast to the earlier sample complexity results [8], we do not have an explicit dependence on the domain size $|\bar{R}_0(S_0)|$. The value of $n^\star$ depends on the accuracy parameter ϵ, the confidence parameter δ and the maximum information capacity defined via $\gamma_n = \max_{|A| \leq n} I(q; y_A)$, where $I(q; y_A)$ is mutual information between the function f and the observations $y_A = q_A + \omega_A$ at the points in the set A. For commonly used kernels, γ_n is known to grow sublinear in n [28], which implies a finite time convergence (upper bound on $n^\star$) as per **??**. Moreover, safety is an immediate consequence of the algorithm since the recommendation of COAT-MPC is from the safe set formed using the lower confidence bound. $l(\theta') \geq l(\theta) - Ld(\theta, \theta') \geq \tau$, now if $l(\theta) \geq \tau$ using line 7 we are guaranteed to have $l(\theta') \geq \tau \implies q(\theta') \geq \tau$

implying safety at each recommendation location with high probability.

VII. Experimental Results

We present an evaluation of the COAt-MPC algorithm's performance in the context of MPC tuning, as defined in Sec. III. Our evaluation is conducted using an autonomous racing simulation and the scaled RC racecar platform shown in Figure 4. Even though the presented experimental results focus on autonomous racing applications, the proposed algorithm and theoretical analysis are widely applicable to the safe tuning of MPC cost function parameters in other applications.

Fig. 4: 1:28 scale RC racecar [11] and track used in the experiments.

A. Model Predictive Counturing Control (MPCC)

In the proposed experiments with the autonomous racing platform, we use a particular MPC formulation known as MPCC (Model Predictive Counturing Control) [10], which is specifically tailored for autonomous racing in a known track. In particular, the MPCC for autonomous racing is formulated as the following nonlinear program:

$$\begin{aligned} \boldsymbol{X}^*, \boldsymbol{U}^* = \arg\min_{\boldsymbol{X},\boldsymbol{U}} \quad & \sum_{k=1}^{N} -q_\gamma \gamma_k + \boldsymbol{e}_k^T \boldsymbol{Q} \boldsymbol{e}_k + \boldsymbol{u}_k^T \boldsymbol{R} \boldsymbol{u}_k \\ \text{s.t.} \quad & \boldsymbol{x}_0 = \hat{\boldsymbol{x}},\ \boldsymbol{x}_{k+1} = \boldsymbol{f}_d(\boldsymbol{x}_k, \boldsymbol{u}_k) \\ & \boldsymbol{x}_k \in \mathcal{X},\ \boldsymbol{u}_k \in \mathcal{U},\ \boldsymbol{h}_k(\boldsymbol{x}_k, \boldsymbol{u}_k) \in \mathcal{H} \end{aligned} \tag{6}$$

where $\boldsymbol{x}_k = [x_k, y_k, \psi_k, v_{x_k}, v_{y_k}, \dot{\psi}_k]$ is the state of the car including position, orientation, and velocities, $\boldsymbol{u}_k = [\delta_k, T_k]$ are the input of the car (steering angle and drivetrain command), γ_k is the parameter that determines the progress along the reference trajectory, $\boldsymbol{e}_k$ denotes contour and lag errors, $\boldsymbol{f}_d(\cdot,\cdot)$ is the nominal car model consisting of a Pacejka dynamic bicycle model [29], and $\boldsymbol{h}_k(\boldsymbol{x}_k, \boldsymbol{u}_k)$ are linear and nonlinear constraints on the states and inputs. The matrix $\boldsymbol{Q} = diag([Q_{contour}, Q_{lag}])$ controls the longitudinal and lateral deviation from the reference trajectory, q_γ regulates the progress of the car, and $\boldsymbol{R}$ determines the smoothness of the inputs.

The MPC formulation of Eq. (7) aims to maximize the progress while penalizing the deviation from the reference trajectory. It is worth noting that this MPC formulation does not minimize time. However, we set the lap time as the objective function of our proposed safe Bayesian Optimization method to achieve lap time minimization.

B. Experimental Setup

To quantify the performance of the MPC, we define the objective function q as the negative lap time of a single flying lap, where the car does not start from a stationary position. The negative sign is introduced to reflect the objective of achieving the fastest lap time. Additionally, we establish the performance upper bound as $\tau = \tau_{scale}\hat{q}(\mathcal{S}_0)$, where $\tau_{scale} \geq 1$ is a scaling factor that is specified by the user, and $\hat{q}(\mathcal{S}_0)$ represents a noisy evaluation of the negative lap time obtained from the initial seed of parameters. Hence, we constrain the lap times to always be lower than the initial lap time multiplied by a scaling factor larger than one.

We conduct a comparative analysis of our proposed method with several non-constrained optimization techniques, namely, GP-UCB [16], Weighted Maximum Likelihood (WML) [15], and Confidence Region Bayesian Optimization (CRBO) [30]. Additionally, we evaluate our method against constrained optimization methods, specifically, (EI_C) [21] and SAFE-OPT [8].

In our experiments, we made the decision to jointly optimize the parameters $Q_{contour}$ and Q_{lag}. To accomplish this, we uniformly discretized the parameter space into 10,000 combinations within the range of $[0,\ 1000]^2$. These combinations were then normalized to fit within the range of $[0,\ 1]^2$. Furthermore, we set the initial weights for $Q_{contour}$ and Q_{lag} to 500. For the methods that utilized a Gaussian Process to model the lap time function, such as COAt-MPC, we selected a Matérn Kernel with a smoothness parameter of $\nu = 5/2$. A unique length-scale of $l = 0.1$ was also chosen for both dimensions. Finally, we choose to evaluate our algorithm with $\beta = 5.0$.

C. Simulation results

We present a comprehensive evaluation of our proposed method in comparison to the established baseline methods over a total of 70 iterations, during which the tuning methods are permitted to sample and assess 70 distinct parameters. Note that, in this setup, the proposed method, COAt-MPC, takes less than 70 iterations due to its termination criteria. As depicted in Table I, our method surpasses the baseline techniques in terms of minimizing performance constraint violations while converging to the optimal parameters in approximately 30 iterations.

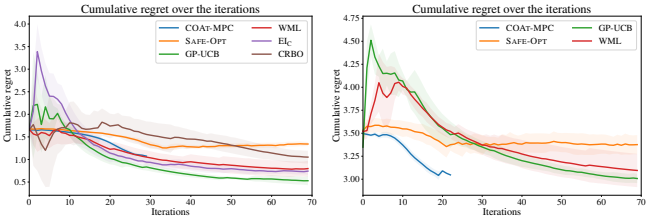


Fig. 5: Mean cumulative regret over time and standard deviation at each iteration.**Left**: Simulation results. **Right**: RC platform results.

Algorithm	Simulation			RC platform		
	#Constraint violations	Min. lap time[s]	Mean lap time ± std. deviation[s]	#Constraint violations	Min. lap time[s]	Mean lap time ± std. deviation[s]
COAt-MPC	**0**	**4.68**	5.57 ± 0.57	**0.33**	**6.49**	7.55 ± 0.50
SAFE-OPT	1.8	4.71	5.84 ± 0.82	5.33	6.52	7.88 ± 0.74
GP-UCB	5.2	**4.68**	$\mathbf{5.03 \pm 1.02}$	7.0	6.82	$\mathbf{7.51 \pm 0.77}$
WML	2.83	4.71	5.30 ± 0.80	6.66	6.95	7.60 ± 0.68
EI_C	7.0	**4.68**	5.24 ± 1.16	-	-	-
CRBO	16.4	4.71	5.55 ± 1.39	-	-	-

TABLE I: Comparison of COAt-MPC with baselines. The algorithms were run 5 times in simulation and 3 times with the RC platform. The results of this table are averaged.

Figure 5 illustrates GP estimate of the negative lap time function, in addition to the samples obtained using our method. The sampling of the baselines is depicted in Figure 6. As evidenced in the figures, COAt-MPC initiates the process by sampling in proximity to the initial parameters, aiming to expand the pessimistic and optimistic sets. Subsequently, the method cautiously samples parameters that result in an improved lap time, and converges once it is ϵ-certain that it has found the optimal parameters.

D. RC platform results

After observing the outcomes in the simulated setting, our algorithm was benchmarked against SAFE-OPT, GP-UCB, and WML, which proved to be the best options among all baseline methods with respect to constraint violations. As shown in Table I, our approach surpasses the baseline methods in terms of constraint violations and closely approximates the average mean lap time of GP-UCB, while effectively converging to the optimal parameters. Moreover, our method stands out by achieving the lowest cumulative regret over time, as depicted in Figure 5. Figure 6 illustrates the GP estimate of the negative lap time function and the samples obtained using our method and the baselines

VIII. CONCLUSIONS

We proposed COAt-MPC, a constrained Bayesian Optimization method. Our approach leverages the assumption of Lipschitz continuity in the objective function, allowing the generation of a pessimistically and optimistically safe set. We leverage the optimistic set to define a goal location at each iteration, while we restrict our recommendations to be within the pessimistic set. We presented a rigorous theoretical analysis of our method, conclusively demonstrating its ability to achieve finite-time convergence. Additionally, our comprehensive evaluation against state-of-the-art methods demonstrated that our method outperforms them in terms of the number of constraint violations, as well as in cumulative regret over time, in the context of MPC tuning for autonomous racing.

Despite its advantages, COAt-MPC has certain limitations. Primarily, it is limited in the number of parameters that it can simultaneously tune. Our approach discretizes the parameter space in order to compute the safe set and sample based on our proposed criterion at every step. This discretization process, however, triggers exponential growth in memory consumption, making the method unfeasible for a vast number of parameters. Additionally, our method is confined to a single constraint on the objective function and, as such, it cannot accommodate other system constraints.

APPENDIX

A. Proof of sample complexity bound

The predictive confidence intervals are built recursively as

$$Q_{n-1}(\boldsymbol{\theta}) \coloneqq [\mu_{n-1}(\boldsymbol{\theta}) \pm \beta_n^{1/2} \sigma_{n-1}(\boldsymbol{\theta})]. \tag{7}$$

Instead of using Q_{n-1} directly, we use their intersection $C_n(\boldsymbol{\theta}) \coloneqq C_{n-1}(\boldsymbol{\theta}) \cap Q_n(\boldsymbol{\theta})$, which ensures that the confidence intervals are monotonically contained after the recursive measurements with $C_0(\boldsymbol{\theta}) = [0, \infty]$. Based on this, we define $u_n(\boldsymbol{\theta}) \coloneqq \max_\theta C_n(\boldsymbol{\theta})$ as the upper confidence bound on $q(\boldsymbol{\theta})$ where $u_{n+1}(\boldsymbol{\theta}) \leq u_t(\boldsymbol{\theta}), \forall \boldsymbol{\theta} \in D$ and similarly, $l_n(\boldsymbol{\theta}) \coloneqq \min_{\boldsymbol{\theta}} C_n(\boldsymbol{\theta})$ as the lower confidence bound on $q(\boldsymbol{\theta})$ where $l_{n+1}(\boldsymbol{\theta}) \geq l_n(\boldsymbol{\theta}), \forall \boldsymbol{\theta} \in D$. We define width of the confidence interval $w_n(\boldsymbol{\theta}) \coloneqq u_n(\boldsymbol{\theta}) - l_n(\boldsymbol{\theta})$.

Define a pessimistic set $\mathcal{S}_n^{p,sage} \coloneqq \{\theta \in D | \exists \theta' \in D, l_n^q(\theta') - L_q d(\theta, \theta') \geq \tau\}$. This set is motivated from [7] and will be used to simplify the analysis. Note that $\mathcal{S}_n^{p,sage} = p_n(D) = \lim_{m \to \infty} P_n^m(D)$.

The following lemma establishes that our pessimistic set is always a subset of the one in [7].

Lemma 1: $\mathcal{S}_n^p \subseteq \mathcal{S}_n^{p,sage}, \forall n \geq \tau$.

Proof: Proof by contradiction. Let's assume $\exists \theta_e \in \mathcal{S}_n^p \backslash \mathcal{S}_n^{p,sage}$. Hence for some $S \subseteq D$, $\exists \theta' \in S : l_n^q(\theta_e) - L_q \|\theta' - \theta_e\| \geq \tau$. Since $S \subseteq D$, the definition of $\mathcal{S}_n^{p,sage}$ implies $\theta_e \in \mathcal{S}_n^{p,sage}$. This is a contradiction. ■

Corollary 2 (Theorem 1 [7]): Let assumptions 2 and 3 hold. Let $n^\star$ be the largest integer satisfying $\frac{n^\star}{\beta_{n^\star} \gamma_{n^\star}} \leq \frac{C}{\epsilon^2}$, with $C = 8/\log(1 + \sigma_q^{-2})$. The sampling scheme $\theta_n \in \mathcal{S}_{n-1}^p : w_{n-1}(\theta_n) \geq \epsilon$ satisfy $q(\theta_n) \geq \tau, \forall n \geq 1$ with probability at least $1-\delta$ and $\exists n \leq n^\star : \forall \theta \in \mathcal{S}_n^p, w_n(\theta) < \epsilon$.

Proof: The theorem 1 in [7] uses a sampling rule $\theta_n \in \mathcal{S}_{n-1}^{p,sage} : w_{n-1}(\theta_n) \geq \epsilon$. Our sampling rule as per Line 7 and Line 10 ensures $w_{n-1}(\theta_n) \geq \epsilon$. Using Lemma 1, $\mathcal{S}_{n-1}^p \subseteq \mathcal{S}_{n-1}^{p,sage}$, hence $\exists n \leq n^\star : \forall \theta \in \mathcal{S}_n^{p,sage}, w_n(\theta) < \epsilon \implies \forall \theta \in \mathcal{S}_n^p, w_n(\theta) < \epsilon$. $\forall \theta \in \mathcal{S}_n^p, l_n^q(\theta) \geq \tau \implies q(\theta)$ which follows by construction of the confidence bounds crefX and ensures safety. Hence proved. ■

Thus, using the result from [7], we terminate the process within $n^\star$ iterations. Next, we guarantee the optimality of

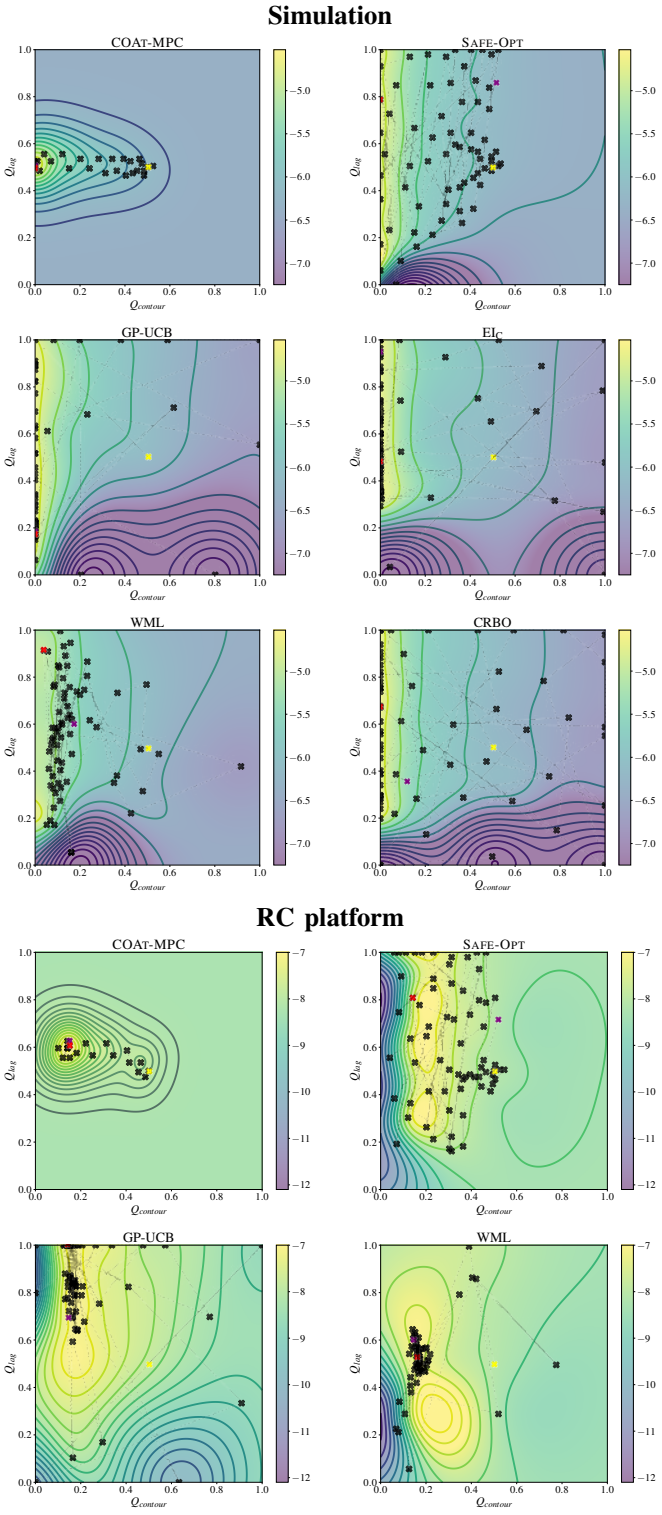


Fig. 6: The figures show the samples of the negative lap time and the GP. The dashed lines denote the sample trajectories. The yellow cross marker denotes the initial values of $Q_{contour}$ and Q_{lag}, the purple cross marker is the last sample, and the red marker is the sample that yields the best lap time.

the objective if we converge with uncertainty $\leq \epsilon$ which is guaranteed by design in the algorithm.

Lemma 2: $\exists n \leq n^\star$, $S^{q,\epsilon} \subseteq \mathcal{S}_n^p$.

Proof: It holds since $\mathcal{S}_n^p = \tilde{P}_n(\mathcal{S}_{n-1}^p) \supseteq \tilde{R}_\epsilon(\mathcal{S}_{n-1}^p) \supseteq \tilde{R}_\epsilon(\mathcal{S}_0) = S^{q,\epsilon}$. The equalities follow from definition of the sets $S^{q,\epsilon}$ and $\mathcal{S}_n^p$. The set inequality $\tilde{P}_n(\mathcal{S}_{n-1}^p) \supseteq \tilde{R}_\epsilon(\mathcal{S}_{n-1}^p)$ holds since $\exists n \leq n^\star : \forall \theta \in \mathcal{S}_n^p, w_n(\theta) < \epsilon$, which further implies $q(\theta) - \epsilon < l_n^q(\theta)$. Hence for any $\mathcal{S} \subseteq \mathcal{S}_n^p, r_n(\mathcal{S}) \subseteq p_n(\mathcal{S})$. $\forall m$ $P_n^m(\mathcal{S}_{n-1}^p) \subseteq \mathcal{S}_n^p$ due to monotonicity property. The last set inequality follows since $\mathcal{S}_0 \subseteq \mathcal{S}_{n-1}^p$. ■

Lemma 3: Let assumption 2 holds and $\theta_n^g := \arg\max_{\theta \in \mathcal{S}_{n-1}^{o,\epsilon}} \mu_{n-1}(\theta) + \beta_n^{1/2} \sigma_{n-1}(\theta)$. If $w_n(\theta_n^g) < \epsilon \implies q(\theta_n^g) \geq \max_{\theta \in \bar{R}_\epsilon(S_0)} q(\theta) - \epsilon$ with probability at least $1-\delta$.

Proof: By definition $\forall n \geq 1, \bar{R}_\epsilon(S_0) \subseteq \mathcal{S}_n^{o,\epsilon}$. Given $w_{n-1}(\theta_n^g) < \epsilon \implies l_n^q(\theta_n^g) > u_n^q(\theta_n^g) - \epsilon \geq q(\theta_n^g) - \epsilon$. Since $\bar{R}_\epsilon(S_0) \subseteq \mathcal{S}_n^{o,\epsilon}$ and $\hat{\boldsymbol{\theta}} := \arg\max_{\theta \in \bar{R}_\epsilon(S_0)} q(\theta)$:

$$\begin{aligned} u_n^q(\hat{\boldsymbol{\theta}}) &\leq u_n^q(\theta_n^g) \\ &< l_n^q(\theta_n^g) + \epsilon \\ &\leq q(\theta_n^g) + \epsilon \\ \implies q(\hat{\boldsymbol{\theta}}) &\leq q(\theta_n^g) + \epsilon \end{aligned}$$

Hence $q(\theta_n^g) \geq \max_{\theta \in \bar{R}_\epsilon(S_0)} q(\theta) - \epsilon$. ■

We now combine the results of the above lemmas in the following theorem and additionally guarantee the safety of the closed-loop system throughout the tuning process.

Theorem 1: Let assumptions 1 to 3 holds and $n^\star$ be the largest integer such that $\frac{n^\star}{\beta_{n^\star}\gamma_{n^\star}} \leq \frac{C_1}{\epsilon^2}$ with $C_1 := 8/\log(1+\sigma^{-2})$. The recommendation of COAt-MPC at iteration n, $\hat{\boldsymbol{\theta}}_n$, satisfies $q(\hat{\boldsymbol{\theta}}_n) \geq \tau, \forall n \geq 1$ and the closed loop system (**??**) satisfies state and input constraints for all times. Furthermore, $\exists n \leq n^\star$ such that the following holds with probability at least $1-\delta$:

$$q(\hat{\boldsymbol{\theta}}_n) \geq \max_{\theta \in \bar{R}_\epsilon(S_0)} q(\theta) - \epsilon.$$

Proof: Under assumption 1, MPC is feasible and hence the closed-loop system satisfies the state and input constraints at all times. We sample in Lines 8 and 11 only if $\boldsymbol{\theta}_n \in \mathcal{S}_{n-1}^p \implies \exists \boldsymbol{\theta} : l_n^q(\boldsymbol{\theta}) - L_q d(\boldsymbol{\theta}, \boldsymbol{\theta}_n) \geq \tau \implies q(\boldsymbol{\theta}) - L_q d(\boldsymbol{\theta}, \boldsymbol{\theta}_n) \geq \implies q(\boldsymbol{\theta}_n) \geq \tau$. Under assumptions 2 and 3, **??** and Lemma 1 shows a sample complexity bound on $\exists n \leq n^\star$, beyond which uncertainty in the $\mathcal{S}_n^p$ is uniformly bounded by ϵ. Once this holds, Lemma 2 shows that we explore the maximum possible exploration domain. In Lines 8 and 11, we sample only if uncertainty is $\geq \epsilon$ and we terminate if it's less than ϵ which happens by **??**. Further Lemma 3 establish the resulting optimality of the COAt-MPC algorithm. Hence proved. ■

We are using a modified GoOSE in the sense that the sampling strategy is as per SageMPC but the set definition is as per GoOSE but without dynamics. We need to do this to be able to relate the growth of the set with respect to the initial location of the agent.

Logic of proof:

$\mathcal{S}_n^p \subseteq \mathcal{S}_n^{pl,sage}$ holds directly by definition. At $n^\star$ the uncertainty is less than ε everywhere. so the algorithm

terminates. Show that $\mathcal{S}_n^{o,\epsilon} \subseteq \mathcal{S}_n^p$ holds directly since the $u_n^q(\boldsymbol{\theta}) - \epsilon \leq l_n^q(\boldsymbol{\theta})$.

REFERENCES

[1] J. L. Vázquez, M. Brühlmeier, A. Liniger, A. Rupenyan, and J. Lygeros, "Optimization-based hierarchical motion planning for autonomous racing," *IEEE/RJS International Conference on Intelligent Robots and Systems*, 2020. 1

[2] D. Kang, F. De Vincenti, N. C. Adami, and S. Coros, "Animal motions on legged robots using nonlinear model predictive control," *IEEE/RJS International Conference on Intelligent Robots and Systems*, 2022. 1

[3] S. Kuindersma, R. Deits, M. F. Fallon, A. K. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake, "Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot," *Autonomous Robots*, 2015. 1

[4] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*, 2003. 1, 3

[5] F. Berkenkamp, A. Krause, and A. Schoellig, "Bayesian optimization with safety constraints: Safe and automatic parameter tuning in robotics," *Machine Learning*, 2021. 1, 2

[6] M. Turchetta, F. Berkenkamp, and A. Krause, "Safe exploration for interactive machine learning," *Conference on Neural Information Processing Systems*, 2019. 1, 2, 3, 4, 5

[7] M. Prajapat, J. Köhler, M. Turchetta, A. Krause, and M. Zeilinger, "Safe guaranteed exploration for non-linear system," *ArXiv*, 2023. 1, 7

[8] Y. Sui, A. Gotovos, J. Burdick, and A. Krause, "Safe exploration for optimization with gaussian processes," *International Conference on Machine Learning*, 2015. 1, 2, 3, 5, 6

[9] M. Prajapat, M. Turchetta, M. Zeilinger, and A. Krause, "Near-optimal multi-agent learning for safe coverage control," *Conference on Neural Information Processing Systems*, 2022. 1, 5

[10] A. Liniger, A. Domahidi, and M. Morari, "Optimization-based autonomous racing of 1:43 scale rc cars," *Optimal Control Aplications and Methods*, 2015. 1, 6

[11] A. Carron, S. Bodmer, L. Vogel, R. Zurbrügg, D. Helm, R. Rickenbach, S. Muntwiler, J. Sieber, and M. N. Zeilinger, "Chronos and crs: Design of a miniature car-like robot and a software framework for single and multi-agent robotics and control," *IEEE International Conference on Robotics and Automation*, 2023. 1, 6

[12] K. J. Åström, "Theory and applications of adaptive control - a survey," *Automatica*, 1983. 2

[13] K. Åström, T. Hägglund, C. Hang, and W. Ho, "Automatic tuning and adaptation for pid controllers - a survey," *Control Engineering Practice*, 1993. 2

[14] A. Loquercio, A. Saviolo, and D. Scaramuzza, "Autotune: Controller tuning for high-speed flight," *IEEE Robotics and Automation Letters*, 2022. 2

[15] A. Romero, S. Govil, G. Yilmaz, Y. Song, and D. Scaramuzza, "Weighted maximum likelihood for controller tuning," *IEEE International Conference on Robotics and Automation*, 2022. 2, 6

[16] N. Srinivas, A. Krause, S. M. Kakade, and M. W. Seeger, "Gaussian process optimization in the bandit setting: No regret and experimental design," *International Conference on Machine Learning*, 2009. 2, 5, 6

[17] P. Frazier, "A tutorial on bayesian optimization," *ArXiv*, 2018. 2

[18] J. Mockus, *Bayesian Approach to Global Optimization: Theory and Applications*, ser. Mathematics and its Applications. Springer Netherlands, 2012. 2

[19] K. Chatzilygeroudis, V. Vassiliades, F. Stulp, S. Calinon, and J.-B. Mouret, "A survey on policy search algorithms for learning robot controllers in a handful of trials," *IEEE Transactions on Robotics*, 2018. 2

[20] L. P. Fröhlich, C. Küttel, E. Arcari, L. Hewing, M. N. Zeilinger, and A. Carron, "Contextual tuning of model predictive control for autonomous racing," *IEEE/RJS International Conference on Intelligent Robots and Systems*, 2022. 2, 3

[21] J. R. Gardner, M. J. Kusner, Z. E. Xu, K. Q. Weinberger, and J. P. Cunningham, "Bayesian optimization with inequality constraints," *International Conference on Machine Learning*, 2014. 2, 6

[22] J. T. Wilson, F. Hutter, and M. P. Deisenroth, "Maximizing acquisition functions for bayesian optimization," *Conference on Neural Information Processing Systems*, 2018. 2

[23] F. Sorourifar, G. Makrygirgos, A. Mesbah, and J. A. Paulson, "A data-driven automatic tuning method for mpc under uncertainty using constrained bayesian optimization," *IFAC Symposium on Advanced Control of Chemical Processes*, 2021. 2

[24] F. Berkenkamp, A. P. Schoellig, and A. Krause, "Safe controller optimization for quadrotors with gaussian processes," *IEEE International Conference on Robotics and Automation*, 2016. 2, 4

[25] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. The MIT Press, 2018. 3

[26] N. Srinivas, A. Krause, S. M. Kakade, and M. W. Seeger, "Information-theoretic regret bounds for gaussian process optimization in the bandit setting," *IEEE Transactions on Information Theory*, 2012. 3

[27] S. R. Chowdhury and A. Gopalan, "On kernelized multi-armed bandits," *International Conference on Machine Learning*, 2017. 4

[28] S. Vakili, K. Khezeli, and V. Picheny, "On information gain and regret bounds in gaussian process bandits," *IEEE International Conference on Artificial Intelligence and Statistics*, 2021. 5

[29] H. B. Pacejka, in *Tyre and Vehicle Dynamics (Second Edition)*. Oxford: Butterworth-Heinemann, 2006. 6

[30] L. P. Fröhlich, M. N. Zeilinger, and E. D. Klenske, "Cautious bayesian optimization for efficient and scalable policy search," *Learning for Dynamics Control Conference*, 2021. 6