Safe and Efficient Model-free Adaptive Control via Bayesian Optimization

Christopher König, Matteo Turchetta, John Lygeros, Alisa Rupenyan, Andreas Krause

Abstract—Adaptive control approaches yield high-performance controllers when a precise system model or suitable parametrizations of the controller are available. Existing data-driven approaches for adaptive control mostly augment standard model-based methods with additional information about uncertainties in the dynamics or about disturbances. In this work, we propose a purely data-driven, model-free approach for adaptive control. Tuning low-level controllers based solely on system data raises concerns on the underlying algorithm safety and computational performance. Thus, our approach builds on GOOSE, an algorithm for safe and sample-efficient Bayesian optimization. We introduce several computational and algorithmic modifications in GOOSE that enable its practical use on a rotational motion system. We numerically demonstrate for several types of disturbances that our approach is sample efficient, outperforms constrained Bayesian optimization in terms of safety, and achieves the performance optima computed by grid evaluation. We further demonstrate the proposed adaptive control approach experimentally on a rotational motion system.

I. INTRODUCTION

Adaptive control approaches are a desirable alternative to robust controllers in high-performance applications that deal with disturbances and uncertainties in the plant dynamics. Learning uncertainties in the dynamics and adapting have been explored with classical control mechanisms such as Model Reference Adaptive Control (MRAC) [1], [2]. Gaussian processes (GP) have been also used to model the output of a nonlinear system in a dual controller [3], while coupling the states and inputs of the system in the covariance function of the GP model. Learning the dynamics in an $L_1$-adaptive control approach has been demonstrated in [4], [5].

Instead of modeling or learning the dynamics, the system can be represented by its performance, directly measured from data. Then, the low-level controller parameters can be optimized to fulfill the desired performance criteria. This has been demonstrated for motion systems in [6]–[10]. Such model-free approaches, however, have not been brought to continuous adaptive control, largely because of difficulties in continuously maintaining stability and safety in the presence of disturbances and system uncertainties, and because of the associated computational complexity. Recently, a sample-efficient extension for safe exploration in Bayesian optimization has been proposed [11]. In this paper, we further optimize this algorithm to develop a model-free adaptive control method for motion systems.

Contribution. In this work, we make the following contributions: (1) we extend the GOOSE algorithm for policy search to adaptive control problems; that is, to problems where constant tuning is required due to changes in environmental conditions. (2) We reduce GOOSE’s complexity so that it can be effectively used for policy optimization beyond simulations. (3) We show the effectiveness of our approach in extensive evaluations on a real and simulated rotational axis drive, a crucial component in many industrial machines.

A table of symbols can be found in Section IX.

II. RELATED WORK

Bayesian optimization. Bayesian optimization (BO) [12] denotes a class of sample-efficient, black-box optimization algorithms that have been used to address a wide range of problems, see [13] for a review. In particular, BO has been successful in learning high-performance controllers for a variety of systems. For instance, [14] learns the parameters of a discrete event controller for a bipedal robot with BO, while [15], trades off real-world and simulated control experiments via BO. In [16], variational autoencoders are combined with BO to learn to control an hexapod, while [17] uses multi-objective BO to learn robust controllers for a pendulum.

Safety-aware BO. Optimization under unknown constraints naturally models the problem of learning in safety-critical conditions, where a priori unknown safety constraints must not be violated. In [18] and [19] safety-aware variants of standard BO algorithms are presented. In [20]–[22], BO is used as a subroutine to solve the unconstrained optimization of the augmented Lagrangian of the original problem. While these methods return feasible solutions, they may perform unsafe evaluations. In contrast, the SAFE OPT algorithm [23] guarantees safety at all times. It has been used to safely tune a quadrotor controller for position tracking [6], [24]. In [7], it has been integrated with particle swarm optimization (PSO) to learn high-dimensional controllers. Unfortunately, SAFE OPT may not be sample-efficient due to its exploration strategy. To address this, many solutions have been proposed. For example, [25] does not actively expand the safe set, which may compromise the optimality of the algorithm but works well for the application considered. Alternatively, STAGEOPT [26] first expands the safe set and, subsequently, optimizes over it. Unfortunately, it cannot provide good solutions if stopped prematurely. GOOSE [11] addresses this problem by using a separate optimization oracle and expanding the safe set in a goal-oriented fashion only when necessary to evaluate the inputs suggested by the oracle.

This project has been funded by the Swiss Innovation Agency (Innosuisse), grant Nr. 46716, and by the Swiss National Science Foundation under NCCR Automation.

1 Inspire AG, Zurich, Switzerland
2 Learning & Adaptive Systems group, ETH Zurich, Switzerland
3 Automatic Control Laboratory, ETH Zurich, Switzerland
* The authors contributed equally.
III. SYSTEM AND PROBLEM STATEMENT

In this section, we present the system of interest, its control scheme, and the mathematical model we use for our numerical evaluations. Finally, we introduce the safety-critical adaptive control problem we aim to solve.

A. System and controller

The system of interest is a rotational axis drive, a positioning mechanism driven by a synchronous 3 phase permanent magnet AC motor equipped with encoders for position and speed tracking (see Fig. 1). Such systems are routinely used as components in the semiconductor industry, in biomedical engineering, and in photonics and solar technologies.

We model the system as a combination of linear and nonlinear blocks, where the linear block is modeled as a damped single mass system, following [9]:

\[ G(s) := [G_p(s), G_v(s)]^T = \begin{bmatrix} \frac{1}{ms^2 + bs} & \frac{1}{ms + b} \end{bmatrix}^T, \quad (1) \]

where \( G_p(s) \) and \( G_v(s) \) are the transfer functions respectively from torque to angular position and to angular velocity, \( m \) is the moment of inertia, and \( b \) is the rotational damping coefficient due to the friction. The values of \( m \) and \( b \) are obtained via least squares fitting and shown in Fig. 1.

Next, we introduce the model for the nonlinear part of the dynamics \( f_c \), which we subtract from the total torque signal, see Fig. 2. The nonlinear coggling effects due to interactions between the permanent magnets of the rotor and the stator slots [27] are modelled using a Fourier truncated expansion between the permanent magnets of the rotor and the stator see Fig. 2. The nonlinear cogging effects due to interactions of the rotor and the stator are obtained via least squares fitting and shown in Fig. 1.

\[ c := [\sin(x), \cos(x)] \rightarrow \text{parameters} \]

\[ \text{minimization between the modelled cogging torque signals} \]

The nonlinear coggling effects due to interactions between the permanent magnets of the rotor and the stator slots [27] are modelled using a Fourier truncated expansion as \( f_c(p) = c_1 + c_2p + \sum_{k=1}^n c_{2k+2} \sin\left(\frac{2\pi}{c_0} p + c_{2k+3}\right) \)

where \( p \) is the position, \( c_1 \) is the average thrust torque, \( c_2 \) is the gradient of the curve, \( c_3 \) is the largest dominant period described by the angular distance of a pair of magnets, \( n \) is the number of modelled frequencies, and, \( c_{2k+2} \) and \( c_{2k+3} \) are respectively the amplitudes and the phase shifts of the sinusoidal functions, for \( k = 1, 2, \ldots, n \). The parameters \( c := [c_1, \ldots, c_{2n+3}] \) are estimated using least squares error minimization between the modelled cogging torque signals and the measured torque signal at constant velocity to cancel the effects from linear dynamics. The estimates of the parameters are shown in Fig. 1. To model the noise of the system, zero mean white noise with 6.09e-3 Nm variance is added to the torque input signal of the plant.

The system is controlled by a three-level cascade controller shown in Fig. 2. The outermost loop controls the position with a PI-controller \( C_p(s) = K_p \), and the middle loop controls the velocity with a PI-controller \( C_v(s) = K_v(1 + \frac{1}{s^2}) \). The innermost loop controls the current of the rotational drive. It is well-tuned and treated as a part of the plant \( G(s) \).

Feedforward structures are used to accelerate the response of the system. Their gains are well-tuned and not modified during the re-tuning procedure. However, in our experiments, we perturb them to demonstrate that our method can adapt to new operating conditions by adjusting the tunable parameters of the controller.

![Fig. 2: Scheme of the proposed BO-based adaptive control.](image)

B. Adaptive control approach

In this section, we present the adaptive control problem we aim to solve. In particular, our goal is to tune the parameters of the cascade controller introduced in Section III-A to maximize the tracking accuracy of the system, following [9]. Let \( \mathcal{X} \) denote the space of admissible controller parameters, \( x = [K_p, K_v, T_s] \), and let \( f : \mathcal{X} \rightarrow \mathbb{R} \) be the objective measuring the corresponding tracking accuracy. In particular, we define \( f \) as the position tracking error averaged over the trajectory induced by the controller \( \hat{f}(x) = \frac{1}{N} \sum_{i=1}^N |p^\text{err}_i(x)| \), where \( p^\text{err}_i \) is the deviation from the reference position at sampling time \( i \). Crucially, \( f \) does not admit a closed-form expression even when the system dynamics are known. However, for a given controller \( x \in \mathcal{X} \), the corresponding tracking accuracy \( f(x) \) can be obtained experimentally. Notice that \( f \) can be extended to include many performance metrics, to minimise oscillations, or to reduce settling time, as in [8], [9].

In practice, we cannot experiment with arbitrary controllers while optimizing \( f \) due to safety and performance concerns. Thus, we introduce two constraints that must be satisfied at all times. The first one is a safety constraint \( q_1(x) \) defined as the maximum of the fast Fourier transform (FFT) of the torque measurement in a fixed frequency window. The second one is a tracking performance constraint \( q_2(x) = |p^\text{err}(x)|_\infty \) defined as the infinity norm of the position tracking error. Finally, in reality, the system may be subject to sudden or slow disturbances such as a change of load or a drift in the dynamics due to components wear. Therefore, our goal is to automatically tune the controller parameters of our system to maximize tracking accuracy for varying operating conditions that we cannot control, while never violating safety and quality constraints along the way.

IV. BACKGROUND

**Gaussian processes.** A Gaussian process (GP) [28] is a distribution over the space of functions commonly used in non-parametric Bayesian regression. It is fully described by a mean function \( \mu : \mathcal{X} \rightarrow \mathbb{R} \), which, w.l.o.g, we set to zero for all inputs \( \mu(x) = 0 \), \( \forall x \in \mathcal{X} \), and a kernel function

<table>
<thead>
<tr>
<th>Param.</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>( m )</td>
<td>0.0191</td>
<td>kg/m²</td>
</tr>
<tr>
<td>( b )</td>
<td>30.08</td>
<td>kg/m²/s</td>
</tr>
<tr>
<td>( c_1 )</td>
<td>1.78e-3</td>
<td>N/m</td>
</tr>
<tr>
<td>( c_2 )</td>
<td>0.0295</td>
<td>N/m/rad</td>
</tr>
<tr>
<td>( c_3 )</td>
<td>0.372</td>
<td>rad</td>
</tr>
<tr>
<td>( c_4 )</td>
<td>8.99e-3</td>
<td>Nm</td>
</tr>
<tr>
<td>( c_5 )</td>
<td>0.11</td>
<td>rad</td>
</tr>
</tbody>
</table>

![Param. Value Unit](image)
\textbf{k : }\mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}. \text{ Given the data set } \mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}, \text{ where } y_i = f(x_i) + \varepsilon_i \text{ and } \varepsilon_i \sim \mathcal{N}(0, \sigma^2) \text{ is zero-mean i.i.d. Gaussian noise, the posterior belief over the function } f \text{ has the following mean, variance and covariance:}

\begin{align*}
\mu_t(x) &= k_t^\top(x)(K_t + \sigma^2I)^{-1}y_t, \\
\kappa_t(x, x') &= k_t(x, x') - k_t^\top(x)(K_t + \sigma^2I)^{-1}k_t(x'), \\
\sigma_t(x) &= k_t(x, x), 
\end{align*}

where \(k_t(x) = (k(x_1, \ldots, x_k(x_i), x'_j))\) is the positive definite kernel matrix \([k(x, x')]_{x, x' \in \mathcal{D}}\), and \(I \in \mathbb{R}^{k \times k}\). 

\textbf{Multi-task BO.} In multi-task BO, the objective depends on the extended input \((x, \tau) \in \mathcal{X} \times \mathcal{T}\), where \(x\) is the variable we optimize over and \(\tau\) is a task parameter set by the environment that influences the objective. To cope with this new dimension, multi-task BO adopts kernels of the form \(k_{\text{multi}}((x, \tau), (x', \tau')) = k_\tau(\tau, \tau') \otimes k(x, x')\), where \(\otimes\) denotes the Kronecker product. This kernel decouples the correlations in objective values along the input dimensions, captured by \(k\), from those across tasks, captured by \(k_\tau\) [29].

\textbf{GOOSE.} GOOSE extends any standard BO algorithm to provide high-probability safety guarantees in presence of a priori unknown safety constraints. It builds a Bayesian model of the constraints from noisy evaluations based on GP regression. It uses this model to build estimates of two sets: the pessimistic safe set, which contains inputs that are safe, i.e., satisfy the constraints, with high probability and the optimistic safe set that contains inputs that could potentially be safe. At each round, GOOSE communicates the optimistic safe set to the BO algorithm, which returns the input it would evaluate within this set, denoted as \(x^*\). If \(x^*\) is also in the pessimistic safe set, GOOSE evaluates the corresponding objective. Otherwise, it evaluates the constraints at a sequence of provably safe inputs, whose choice is based on a heuristic priority function, that allow us to conclude that \(x^*\) either satisfies or violates the constraints with high probability. In the first case, the corresponding objective value is observed. In the second case, \(x^*\) is removed from the optimistic safe set and the BO algorithm is queried for a new suggestion. Compared to [23], [26] GOOSE achieves a higher sample efficiency [11] while compared to [18]–[22], it guarantees safety at all times with high probability, under regularity assumptions.

\textbf{GOOSE assumptions.} To infer constraint and objective values of inputs before evaluating them, GOOSE assumes these functions belong to a class of well-behaved functions, i.e., functions with a bounded norm in some reproducing kernel Hilbert space (RKHS) [30]. Based on this assumption, we can build well-calibrated confidence intervals over them. Here, we present these intervals for the safety constraint, \(q\) (the construction for \(f\) is analogous). Let \(\mu_t^q(x)\) and \(\sigma_t^q(x)\) denote the posterior mean and standard deviation of our belief over \(q(x)\) computed according to Eqs. (2) and (3). We recursively define these monotonically increasing/decreasing lower/upper bounds for \(q(x)\):

\[l_t^q(x) = \max(l_{t-1}^q(x), \mu_{t-1}^q(x) - \beta_{t-1}^p \sigma_{t-1}^q(x)) \quad \text{and} \quad u_t^q(x) = \min(u_{t-1}^q(x), \mu_{t-1}^q(x) + \beta_{t-1}^p \sigma_{t-1}^q(x))\].

The authors of [31], [32] show that, for functions with bounded RKHS norm, an appropriate choice of \(\beta^p\) implies that \(l_t^q(x) \leq q(x) \leq u_t^q(x)\) for all \(t \in \mathbb{R}^+\) and \(x \in \mathcal{X}\). However, in practice, a constant value of 3 is sufficient to achieve safety [6], [11], [24].

To start collecting data safely, GOOSE requires knowledge of a set of inputs that are known a priori to be safe, denoted as \(S_0\). In our problem, it is easy to identify such set by designing conservative controllers for simplistic first principle models of the system under control.

\textbf{V. GOOSE for adaptive control}

In adaptive control, quickly finding a safe, locally optimal controller in response to modified external conditions is crucial. In its original formulation, GOOSE is not suitable to this problem for several reasons: (i) it assumes knowledge of the Lipschitz constant of the constraint, which is unknown in practice, (ii) it relies on a fine discretization of the domain, which is prohibitive in large domains, (iii) it explicitly computes the optimistic safe set, which is expensive, (iv) it does not account for external changes it cannot control. In this section, we address these problems step by step and we present the resulting algorithm.

\begin{algorithm}
\caption{GOOSE for adaptive control}
\begin{algorithmic}
\State \textbf{1 Input:} Safe seed \(S_0\), \(f \sim \mathcal{GP}(\mu^f, k^f; \theta^f)\), \(q \sim \mathcal{GP}(\mu^q, k^q; \theta^q)\), \(\tau = \tau_0\);
\State \textbf{2 Grid resolution:} \(\Delta x\) s.t. \(k^q(x, x + \Delta x)(\sigma_0^q)^{-2} = 0.95\)
\While\ (machine is running) \Do
\State \(S_t \leftarrow \{x \in \mathcal{X} : u_t^q(x, \tau) \leq \kappa\}\);
\State \(L_t \leftarrow \{x \in S_t : \exists \tau \notin S_t, \text{ with } d(x, \tau) \leq \Delta x\} \setminus \{x \in S_t : u_t^q(x, \tau) \geq \epsilon\};
\State \(W_t \leftarrow \{x \in L_t : u_t^q(x, \tau) - l_t^q(x, \tau) \geq \epsilon\};
\State \(x^* \leftarrow \mathcal{P}(S_t, W_t);\)
\State \(q(x^*, \tau) \geq \epsilon_0 \Rightarrow\) \text{ evaluate } \(f(x^*), q(x^*)\), update \(\tau\);
\Else\ \While\ (\exists x \in W_t, \ s.t. \ g_t^q(x, x^*) > 0) \Do
\State \(x^*_w \leftarrow \arg\min_{x \in W_t} d(x^*, x) \text{ s.t. } g_t^q(x, x^*) \neq 0;\)
\State \(\text{Evaluate } f(x^*_w), q(x^*_w),\) update \(\tau, S_t, L_t, W_t;\)
\Else \EndWhile \EndWhile \EndWhile \EndWhile
\State \text{set system to } x^*(\tau), \text{ update } \tau
\EndWhile
\end{algorithmic}
\end{algorithm}

\textbf{Task parameter.} In general, the dynamics of a controlled system may vary due to external changes. For example, our rotational axis drive may be subject to different loads or the system components may wear due to extended use. As dynamics change, so do the optimal controllers. In this case, we must adapt to new regimes imposed by the environment. To this end, we extend GOOSE to the multi-task setting presented in Section IV by introducing a task parameter \(\tau\) that captures the exogenous conditions that influence the system’s dynamics, and by using the kernel introduced in Section IV. To guarantee safety, we assume that the initial safe seed \(S_0\) contains at least one safe controller for each possible task.
Algorithm 2: PSO

1. **Input:** Safe set $S_i$, Boundary $W_i$, # particles $m$;
2. $p_i \sim \mathcal{U}(S_i)$, $v_i \sim \pm \Delta x$, $i = 1, \ldots, m$;
3. for $j \leftarrow 0$ to $J$ do
4. for $i \leftarrow 0$ to $m$ do
5. $\text{cond} \leftarrow u_i^q(p_i^j, \tau) \leq \kappa \lor \exists x \in W_i : g_i^q(x, p_i^j) > 0$;
6. $\text{cond} \leftarrow \text{cond} \land l_i^t(p_i^j, \tau) < z_i^j - 1$;
7. if $\text{cond}$ then $z_i^j \leftarrow l_i^t(p_i^j, \tau)$ else $z_i^j \leftarrow z_i^j - 1$;
8. $\bar{v}_i^{j+1} \leftarrow \alpha_i v_i^j + r_1(z_i^j - p_i^j) + r_2\bar{v}_i^j$, $\forall i$;
9. $p_i^{j+1} \leftarrow p_i^j + \bar{v}_i^j$, $\forall i$;
10. return $\bar{v}_i^j$.

**Lipschitz constant.** GoOSE uses the Lipschitz constant of the safety constraint $L_q$, which is not known in practice, to compute the pessimistic safe set and an optimistic upper bound on constraint values. For the pessimistic safe set, we adopt the solution suggested in [6] and use the upper bound of the confidence interval to compute it (see L. 3 of Algorithm 1). While pessimism is crucial for safety, optimism is necessary for exploration. To this end, GoOSE computes an optimistic upper bound for the constraint value at input $z$ based on the lower bound of the confidence interval at another input $x$ as $l_i^p(x, \tau) + L_qd(x, z)$, where $d(\cdot, \cdot)$ is the metric that defines the Lipschitz continuity of $q$. Here, we approximate this bound with $l_i^p(x, \tau) + \|\mu_i^q(x, \tau)\|_\infty d(x, z)$, where $\mu_i^q(x, \tau)$ is the mean of the posterior belief over the gradient of the constraint induced by our belief over the uncertain points on its boundary $W_i$ (L. 5), which are used to determine whether controllers belong to the optimistic safe set. Based on these, a new suggestion $x^*$ is computed (L. 6). If its lower bound is close to the best observation for the current task $x^t(\tau) = \arg\min_{(x', \tau') \in \mathcal{D}, \tau' = \tau} f(x')$, we stop (L. 7). Otherwise, if the suggestion is safe, we evaluate it and possibly update the task parameter (L. 8). Finally, if we are not sure it is safe, we evaluate all the expanders $x^*_w$ in increasing order of distance from the suggestion $x^*$, until either $x^*$ is in the pessimistic safe set and can be evaluated or there are no expanders for $x^*$ and a new query to PSO is made (L. 11 and 12). During this inner loop the task parameter is constantly updated.

**VI. NUMERICAL RESULTS**

We first apply Algorithm 1 to tune the controller in Fig. 2, simulating the system model in Section III in stationary conditions. Later, we use our method for adaptive control of instantaneous and slow-varying changes of the plant. In
Section IX, we present an ablation study that investigates the impact of the task parameter on these problems.

The optimization ranges of the controller parameters are set to $K_p \in [5, 50]$, $K_v \in [0.1, 0.11]$ for the position and velocity gains, and $T_i \in [1, 10]$ for the time constant. For each task, GoOSE returns the controller corresponding to the best observation, $x^1(t) = (K_p^1, K_v^1, T_i^1)$. The cost $f(x)$ is provided in [deg 10^{-3}] units. We denote the ‘true’ optimal controller computed via grid search as $x_{\text{grid}}^1$. Due to noise, not even this controller can achieve zero tracking error, see the table in Fig. 3. We use a zero mean prior and squared exponential kernel with automatic relevance determination, with length scales for each dimension $[l_{K_p}, l_{K_v}, l_{T_i}, l_{m}, l_{b}] = [30, 0.03, 3, 0.5, 0.3, 5]$, identical for $f$, $q_1$ and $q_2$ for the numerical simulations and the experiments on the system. The likelihood variance is adjusted separately for each GP model.

Control for stationary conditions. In this section, we compare GoOSE to CBO for controller tuning [10] in terms of the cost and the safety and performance constraints introduced in Section III when tuning the plant under stationary conditions. We benchmark these methods against exhaustive grid-based evaluation using a grid with $5 \times 11 \times 10$ points.

For each algorithm, we run 10 independent experiments, which vary due to noise injected in the simulation. Each experiment starts from the safe controller $[K_p, K_v, T_i] = [15, 0.05, 3]$. The table in Fig. 3 shows the median number of iterations needed to minimize the cost. Fig. 3 (left) shows the convergence of both algorithms for each repetition. While both algorithms converge to the optimum, GoOSE prevents constraint violations for all iterations (see Fig. 3). In contrast, CBO violates the constraints in 27.8% of the iterations to convergence, reaching far in the unsafe range beyond the safety limit of acceptable vibrations in the system (Fig. 3, center), showing that additional safety-related measures are required. While the constraint violations incurred by CBO can be limited in stationary conditions by restricting the optimization range, this is not possible for adaptive control.

Adaptive control for instantaneous changes. We show how our method adapts to instantaneous change in the load of the system. We modify the moment of inertia $m$ and estimate this from the system data. We inform the algorithm of the operating conditions through a task parameter, $\tau_m = \log_{10}(\frac{1}{N} \sum_{i} \left|\text{FFT}(v - v_{\text{ff}})\right|)$, which is calculated from the velocity measurement $v$, using the feed-forward signal $v_{\text{ff}}$ from position to velocity. This data-driven task parameter exploits the differences in the levels of noise in the velocity signal corresponding to different values of the moment of inertia $m$. As the velocity also depends on the controller parameters, configurations in a range of $\pm 0.15$ of the current $\tau_m$ value are treated as the same task configuration.

The algorithm is initialized with $[K_p, K_v, T_i] = [15, 0.05, 3]$ as safe seed for all tasks. The moment of inertia of the plant $m$ is switched every 100 iterations. The table in Fig. 4 summarizes 10 repetitions of the experiment with a stopping criterion set to $\epsilon_{\text{tol}} = 0.002$. Fig. 4 (top left) shows the sampled cost and the task parameters for one run, and the running minimum for each task. We clearly see that GoOSE quickly finds a high-performing controller for the initial operating conditions ($m = 0.0191\text{kgm}^2$). Subsequently, it adapts $K_p$ and $K_v$ in few iterations to the new regime ($m = 0.0382\text{kgm}^2$) due to the capability of the GP model to generalize across tasks (Fig. 4, top table). Finally, when the system goes back to the initial regime, GoOSE immediately finds a high-performing controller and manages to marginally improve over it. In particular, the best cost $f(x_1^1(\tau_m))$ for each condition coincides with the optimum obtained by grid evaluation $f(x_{\text{grid}}^1(\tau_m))$. Moreover, Fig. 4 (top center left and right) shows that the constraints are never violated.

Adaptive control for gradual changes. We show how our method adapts to slow changes in the dynamics. In particular, we let the rotation damping coefficient increase linearly with time: $b(t) = b_0(1 + \frac{t}{1000})$, where $b_0 = 30.08\text{kgm}^2/s$. In this case, time is the task parameter and, therefore, we use the temporal kernel $k(t, t') = (1 - \epsilon_t)(t-t')^2$ [35] as task kernel.
(b) Safety constraint $q_1$

Fig. 5: Cost (left), safety constraint (center), performance constraint (right) and summary (table) for the real-world adaptive control experiments with sudden change of the feed-forward gain $K_f$ (upper) and rotational damping $b$ (lower). The thin lines show the values for each experiment. The thick one shows the best cost found for the current task. GoOSE quickly finds optimal controllers for all the regimes (left) without violating the constraints (center-right) despite the location of the optimum keeps changing (table).

The convergence of the optimization accelerates with increasing data. Fig. 5 (top left) shows that convergence is not reached for the nominal $K_f$ during the first 50 iterations, whereas the last configuration with $\tau_K = 1.1$ requires only 16 until convergence, showing that learning is efficient, even if the optimum shifts w.r.t. the configuration. Constraint violations are completely prevented for all tasks parameters, as shown in Fig. 5 (top center and right).

Adaptive control for external parameter change. We now validate experimentally the change introduced to the controller by modifying the friction $b$ in the system, which is related to the non-linearity in the dynamics of the plant. We estimate the change in $b$ by the average torque measurement of the system and provide it as task parameter $\tau_b$, as shown in Fig. 5 (bottom). We start with optimizing the nominal controller parameters for 100 iterations, followed by an increase of $b$ (and of $\tau_b$ accordingly), which is achieved by wrapping elastic bands around the rotational axis and fixing them at the frame of the system. In the last 100 iterations we switch back to the nominal condition. The stopping criterion is set to $\epsilon_{\text{tol}} = 0.001$. Since $\tau_b$ is not fixed and is influenced by noise, all configurations with $\tau_b$ in a range of $\pm 0.1$ of the current $\tau_b$ value are treated as the same task configuration. The cost increases after the first intervention, then reaches close to nominal values in 10 iterations, and adapts almost instantaneously at the next $\tau_b$ switch (Fig. 5, bottom left). The constraints are never violated (Fig. 5, bottom center, right).

VIII. Conclusion

We present a model-free approach to safe adaptive control. To this end, we introduce several modifications to GoOSE, a safe Bayesian optimization method, to enable its practical use on a rotational motion system. We demonstrate numerically and experimentally that our approach is sample efficient, safe, and achieves the optimal performance for different types of disturbances encountered in practice. Our approach can be further extended by including multiple performance metrics in the optimization objective, or gradually tightening the constraints, once sufficient system data is available.
IX. APPENDIX

Here, we study the impact of tracking the task parameter and adapting to its changes on the performance and safety of the algorithm. To this end, we repeat the numerical experiments for instantaneous and gradual changes presented in Section VI and the experiment for internal parameter changes on the real system presented in Section VII without explicitly modeling the task parameter in our algorithm, and compare the results with those obtained modeling it.

In general, the change in systems dynamics that are induced by a change in external conditions and that are captured by the task parameter greatly impacts the performance and safety of the system when a given controller is applied. As a result, for a given controller $x$, the algorithm is likely to observe widely different cost and constraints values depending on the task. Without the additional degree of freedom provided by the explicit modelling of the task parameter, the GP model must do its best to make sense out of widely different observations corresponding to the same controller that are too large to be explained by the observation noise. Therefore, we expect this degradation in the quality of the model to induce poor predictive distributions and, therefore, to lead to undesirable and unpredictable behavior of the algorithm.

Numerical results. We start by showing the results obtained for the numerical experiments in Fig. 6. The top panel shows the performance of the algorithm for the instantaneous change of the moment of inertia in the system, and the bottom panel demonstrates the performance during a gradual linear change in the rotational damping. In both cases, the algorithm does not model these changes.

The first experiment replicates exactly the one we presented in the top of Fig. 4. Initially, $m$ is 0.0191kgm$^2$, which induces an average value of $\tau_m$ over the duration of the task of 1.83, then doubles to 0.0382kgm$^2$, which induces a lower average value $\tau_m = 1.5$ as the higher moment of inertia reduces the effect of the noise, and switches back to 0.0191kgm$^2$, which, due to the model mismatch, induces an even higher average value of $\tau_m = 2$. Notice that, even though the task parameter $\tau_m$ is not provided to the algorithm, we still plot it to show where the tasks switch. During the initial phase of the optimization, the algorithm only experiences one task and, therefore, it is not affected by the fact that it is not modeling the task explicitly. In this phase, an optimum is found after the initial 100 iterations. In the second phase, when $m$ increases, the control task becomes effectively easier. Therefore, the algorithm still manages to control the system and to slightly improve the performance. In this phase, aggressive controllers are classified as safe. This is because they do not induce dangerous vibrations in the system due to the increased moment of inertia. However, switching back to the first, lower value of $m$ creates problems. This is due to two main facts: (i) controllers that were previously deemed safe due to the high value of $m$ are not anymore. However, since $\tau_m$ is not modelled, the algorithm is not aware of this. (ii) the GP is conditioned on highly conflicting observations induced by the different tasks that cannot explain through measurement noise. As a consequence, the confidence intervals it provides deteriorate. This results in multiple violations of both constraints.

The second numerical experiment replicates the one shown at the bottom of Fig. 4. Although the task parameter is no longer tracked, the 30 iterations window for $x^\dagger$ in Algorithm 1 is kept in the algorithm as an adaptive component. However, this adaptive component alone is not sufficient since the GPs are not able to distinguish between old, less accurate data and new, more accurate data without keeping track of the task parameter. Therefore, as the rotation damping coefficient $b$ increases, the system becomes less robust towards aggressive controllers and the algorithm starts using unsafe parameter settings that used to be safe in previous iterations (see figure Fig. 6 bottom).

Experimental result. In comparison to the two numerical experiments of figure Fig. 6, where we see that the algorithm fails in maintaining safety when we don’t inform the GPs about a system relevant change, in this experiment we show that the performance can also deteriorate, without loosing stability. The internal change of $K_{hi}$ has a marginal influence on the set of stable controller parameters. However, the table in Fig. 5 shows that it greatly influences the location of the optimal parameters within this set. Without tracking a task parameter representing the change in $K_{hi}$, the algorithm optimizes the objective function for the initial value of $K_{hi}$. Since there is no controller in later stages that outperforms the optimum of the first stage, the algorithm quickly reaches the termination criterion and applies the optimal controller found in the the first stage to the system. This results in sub-optimal costs as can be seen by comparing Figs. 5 and 6 (left plots and tables). Furthermore, similar to the experiments of Fig. 6, some parameter settings used by the algorithm after switching the initial $K_{hi}$ value violate the safety or the performance constraint, since similar settings were accepted as safe in prior configurations/tasks.

**LIST OF SYMBOLS**

The next list describes several symbols that we used throughout the paper and their explanation.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D$</td>
<td>Data set</td>
</tr>
<tr>
<td>$\mathcal{X}$</td>
<td>Space of admissible controllers</td>
</tr>
<tr>
<td>$g^\dagger(x,z)$</td>
<td>Optimistic noisy expansion operator</td>
</tr>
<tr>
<td>$K_p$</td>
<td>Position proportional gain</td>
</tr>
<tr>
<td>$K^\dagger_p$</td>
<td>Position gain in $x^\dagger$</td>
</tr>
<tr>
<td>$K_v$</td>
<td>Velocity proportional gain</td>
</tr>
<tr>
<td>$K^\dagger_v$</td>
<td>Velocity gain in $x^\dagger$</td>
</tr>
<tr>
<td>$\mu$</td>
<td>GP mean</td>
</tr>
<tr>
<td>$\xi$</td>
<td>Overall particles best position</td>
</tr>
<tr>
<td>$x^*$</td>
<td>Recommendation of PSO</td>
</tr>
<tr>
<td>$\tau$</td>
<td>Task parameter</td>
</tr>
<tr>
<td>$T_i$</td>
<td>Velocity time constant</td>
</tr>
<tr>
<td>$T^\dagger_i$</td>
<td>Time constant in $x^\dagger$</td>
</tr>
<tr>
<td>$x^\dagger(\tau)$</td>
<td>Feasible controller corresponding to best observed cost for task $\tau$</td>
</tr>
</tbody>
</table>
(a) Minimum and observed cost $f$. 
(b) Safety constraint $q_1$. 
(c) Tracking error constraint $q_2$. 

Fig. 6: Cost (left), safety constraint (center) and performance constraint (right) for the simulated optimization experiments with sudden change of the moment of inertia $m$ (upper) and a slow change of the rotational damping $b$ (lower) without tracking of the task parameter. The tables summarize the optimization experiments for 10 repetitions. Without task parameter tracking GoOSE results in unsafe behavior for both kind of disturbances.

<table>
<thead>
<tr>
<th>$\tau_m$</th>
<th>1.83</th>
<th>1.5</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(x^A_{\text{grid}}(\tau_m))$</td>
<td>1.27</td>
<td>0.97</td>
<td>1.27</td>
</tr>
<tr>
<td>$f(x^B_{\text{grid}}(\tau_m))$</td>
<td>1.26</td>
<td>1.03</td>
<td>1.39</td>
</tr>
<tr>
<td>$K^\tau$</td>
<td>50</td>
<td>46.73</td>
<td>46.73</td>
</tr>
<tr>
<td>$\sigma_1$</td>
<td>0.095</td>
<td>0.11</td>
<td>0.11</td>
</tr>
<tr>
<td>$T^1$</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\tau$</th>
<th>30</th>
<th>300</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(x^A_{\text{grid}}(\tau))$</td>
<td>1.28</td>
<td>1.41</td>
</tr>
<tr>
<td>$f(x^B_{\text{grid}}(\tau))$</td>
<td>1.30</td>
<td>1.40</td>
</tr>
<tr>
<td>$K^\tau$</td>
<td>46.7</td>
<td>50</td>
</tr>
<tr>
<td>$\sigma_1$</td>
<td>0.095</td>
<td>0.086</td>
</tr>
<tr>
<td>$T^1$</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\tau_{Kf}$</th>
<th>1.0</th>
<th>0.95</th>
<th>1.05</th>
<th>0.9</th>
<th>1.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(x^A)$</td>
<td>1.62</td>
<td>2.08</td>
<td>2.01</td>
<td>3</td>
<td>2.89</td>
</tr>
<tr>
<td>$K^\tau$</td>
<td>25</td>
<td>25</td>
<td>25</td>
<td>25</td>
<td>25</td>
</tr>
<tr>
<td>$\sigma_1$</td>
<td>0.07</td>
<td>0.07</td>
<td>0.07</td>
<td>0.07</td>
<td>0.07</td>
</tr>
<tr>
<td>$T^1$</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

(a) Minimum and observed cost $f$. 
(b) Safety constraint $q_1$. 
(c) Tracking error constraint $q_2$. 

Fig. 7: Cost (left), safety constraint (center), performance constraint (right) and summary (table) for the real-world adaptive control experiments with sudden change of the feed-forward gain $K_f$, without considering the task parameter. The table summarizes the performance of the algorithm on the real system for 10 different experiments. Without keeping track of the task parameter, the cost for each different $K_f$ after the first is higher, compared to task parameter tracking. While safety is mostly maintained, the performance constraint is violated.

$x^A_{\text{grid}}(\tau)$ Best feasible controller computed with grid search for task $\tau$ 

$f$ Objective (Average tracking error) 

$k$ GP kernel 

$L$ Safe set boundary 

$L_{qf}$ Lipschitz constant 

$l_k(x)$ Lower bound of confidence interval 

$q_1$ Safety constraint (FFT of the torque) 

$q_2$ Performance constraint (Max tracing error) 

$S$ Safe set 

$S^c$ Optimistic safe set 

$S_0$ Initial safe seed 

$u_k(x)$ Upper bound of confidence interval 

$W$ Uncertain points on safe set boundary 

$x^*_{u_i}$ Expander 

$s_i$ $i^{th}$ particle best position

REFERENCES


