POLITECNICO DI MILANO
DEPARTMENT OF AEROSPACE ENGINEERING

POLITECNICO
MILANO 1863

Machine Learning for Mechanical Systems

AA 2024-2025

Professor: Loris Roveda

Bayesian Optimization for Tuning PID Control

Person Code	Surname	Name
10795356	Frassinella	Luca
10990413	Massini	Alessandro
10730683	Nuccio	Gabriele

June 8, 2025

Contents

1 Introduction

In modern industrial automation, robotic systems are increasingly required to operate autonomously and to adapt their behavior to varying operating conditions, while minimizing the use of time and computational or financial resources. Among the most critical challenges in this domain is the tuning of control parameters, which directly influences the robot's ability to accurately follow predefined trajectories under varying dynamic conditions.

Despite being the most widely used approach, model-based control design typically relies on accurate dynamic modeling of the robot and requires extensive data collection through dedicated experimental procedures. These requirements often make it unsuitable for fast deployment or flexible applications. To overcome these limitations, this report explores a data-driven methodology for automatic tuning of control parameters in a robotic manipulator.

The proposed method is based on a Bayesian Optimization (BO) framework, which is employed to automatically tune both low-level and high-level control parameters. Specifically, the low-level stage focuses on optimizing the equivalent link-mass parameters used for dynamic compensation, while the high-level stage is dedicated to tuning the joint PID gains. For the latter, two optimization strategies were tested: an 'all-at-once' (AAO) approach, where all gains are tuned simultaneously, and a sequential scheme that optimizes the PID parameters progressively, starting from the last joint and moving backward to the first.

Multiple cost functions and acquisition functions were evaluated to identify the most suitable configuration in terms of convergence speed and tracking performance. The experimental setup relies on a torque-controlled 7-degrees-of-freedom FRANKA Emika manipulator. To reduce complexity and ensure reproducibility, the reference motion used for tuning is a simple sinusoidal trajectory defined in the joint space.

2 Problem Description

In this section it is outlined the mathematical formulation of the problem, along with the adopted control strategy and the underlying assumptions. The goal is to provide a clear framework in which the methodology operates, specifying the structure of the control law, the parameters to be optimized, and the cost functions used for performance evaluation. The description also highlights the simplifications and hypotheses introduced to make the tuning procedure feasible within a limited experimental budget and in the absence of a complete model of the robot dynamics.

2.1 Robot Dynamics Model

The dynamics adopted to model the robot behavior is based on the classical Lagrangian formulation [1], as reported in Equation (1). This formulation provides a structured representation of manipulator dynamics, explicitly capturing the mass dependency in the inertial, Coriolis, and gravitational terms, as well as accounting for joint friction effects.

$$\mathbf{B}(\mathbf{q},\mathbf{m})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}},\mathbf{m}) + \mathbf{g}(\mathbf{q},\mathbf{m}) + \mathbf{h_f}(\dot{\mathbf{q}}) = \boldsymbol{\tau} - \mathbf{J}(\mathbf{q})^{\mathbf{T}}\mathbf{h}_{\text{ext}} \tag{1}$$

In the model, $\mathbf{q}$ is the joint position vector, with $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ denoting its first and second time derivatives, respectively, and $\mathbf{m}$ representing the robot link-mass parameters. The inertia matrix $\mathbf{B}(\mathbf{q},\mathbf{m})$, the Coriolis term $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}},\mathbf{m})$, and the gravitational vector $\mathbf{g}(\mathbf{q},\mathbf{m})$ all depend on both the configuration and mass distribution. Joint friction is modeled through $\mathbf{h}_f(\dot{\mathbf{q}})$, external forces acting on the end-effector are represented by the wrench $\mathbf{h}_{\text{ext}}$ mapped via the Jacobian $\mathbf{J}(\mathbf{q})^\top$, and the input torques applied to the joints are given by $\boldsymbol{\tau}$.

Some assumptions and simplifications are introduced to reduce computational complexity and improve convergence efficiency. First, the number of link-mass parameters $\mathbf{m}$ to be optimized is limited to four, specifically those associated with links 1, 3, 5, and the end-effector. In the first optimization stage, joint-level friction effects are neglected. Moreover, both external forces $\mathbf{h}_{\text{ext}}$ and the Coriolis term $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}},\mathbf{m})$ are assumed to be zero throughout the entire optimization process.

2.2 Control Architecture

As a control strategy, it has been implemented a PID trajectory control loop with feedback linearization and feedforward action.

The control input applied to the robot corresponds to the total joint torque, which is defined as the sum of three distinct components, each responsible each contributing to the overall control objective. The feedforward term $\boldsymbol{\tau}_{FF}$ is designed to produce the required joint accelerations based on the desired trajectory, assuming ideal model knowledge. The feedback linearization component $\boldsymbol{\tau}_{FL}$ compensates for the nonlinear effects arising from the gravitational term, effectively linearizing the system's response. Finally, the PID component $\boldsymbol{\tau}_{PID}$ corrects residual tracking errors by applying proportional, derivative, and integral feedback, thus improving precision and robustness in the presence of unmodeled dynamics, disturbances, or model mismatches.
The overall control action is thus defined as:

$$\boldsymbol{\tau} = \boldsymbol{\tau}_{FF} + \boldsymbol{\tau}_{PID} + \boldsymbol{\tau}_{FL} \tag{2}$$

with:

$$\boldsymbol{\tau}_{FF} = \mathbf{B}(\mathbf{q}, \mathbf{m})\ddot{\mathbf{q}}^d \tag{3a}$$

$$\boldsymbol{\tau}_{PID} = \mathbf{K}_p\mathbf{e}_q + \mathbf{K}_d\dot{\mathbf{e}}_q + \mathbf{K}_i \int \mathbf{e}_q \tag{3b}$$

$$\boldsymbol{\tau}_{FL} = \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{m}) + \mathbf{g}(\mathbf{q}, \mathbf{m}) \tag{3c}$$

where $\mathbf{q}^d$ is the joint position reference vector and $\mathbf{e}_q = \mathbf{q}^d - \mathbf{q}$ is the joint position error vector.

In order to achieve accurate and robust trajectory tracking, it is essential to optimize both the low-level and high-level control parameters. Specifically, the link-mass parameters, which directly affect the feedforward torque $\boldsymbol{\tau}_{FF}$ and the feedback linearization term $\boldsymbol{\tau}_{FL}$, must be accurately identified to ensure correct compensation of inertial and gravitational effects. At the same time, the PID gains play a critical role in minimizing residual tracking errors and enhancing the stability and robustness of the closed-loop system.
Moreover, based on the datasheet of the Franka Emika Panda robot [2], a realistic noise level of 0.005 Nm on the total torque was added by means of the `mvnrnd` Matlab function which aims to simulate a more realistic actuator behavior by introducing stochastic variations of the torque term.
The optimization is achieved through a two-stage optimization method aimed to decrease the number of variable to be optimize in a single problem.

As reference trajectory, a sinusoidal joint trajectory was adopted. To avoid an initial torque spike due to non-zero initial velocities, the trajectory was phase-shifted such that the joint velocities at the starting time are zero. This ensures a smoother initialization of the control input and improves safety and stability during the first iterations of the optimization, thus not requiring an anomalous increase of the applicable torque. The final motion, along with the amplitudes and frequency, is as follows:

$$\mathbf{q}_r(t) = \mathbf{q}_{0,i} - \mathbf{A}\cos(2\pi f t) + \mathbf{A}, \tag{3}$$

$$\dot{\mathbf{q}}_r(t) = 2\pi f \mathbf{A}\sin(2\pi f t), \tag{4}$$

$$\ddot{\mathbf{q}}_r(t) = (2\pi f)^2 \mathbf{A}\cos(2\pi f t), \tag{5}$$

where the oscillation amplitude is set to $A = 10°$ for all the joints, and the trajectory frequency is fixed at $f = 1$ Hz, uniformly applied to all joints. The initial joint configuration is set to:

$$\mathbf{q}_0 = \begin{bmatrix} -0.7160 & -0.5850 & 0.3504 & -1.5666 & 0.2241 & -2.1201 & -2.8398 \end{bmatrix}^\top \text{ [rad]}$$

In an initial optimization run, the performance criteria were defined solely on the basis of position and velocity tracking errors. However, it was later observed that the resulting joint torques exceeded the maximum admissible values specified in the FRANKA Emika robot datasheet. To address this issue and ensure physical feasibility, a second optimization was performed, this time including the maximum joint torque as an additional term in the cost function.

2.3 Two-Stage Optimization Method

In this paragraph, an overview of the Two-Stage Optimization Method is presented. The approach is based on the methodology proposed in literature [1]: first, the equivalent masses of joints 1, 3, 5, and 7 are optimized to improve the compensation of the robot's inertial effects in the dynamic model. Then, the PID gains are tuned to improve the tracking performance of the control system. Each stage minimizes a custom cost function that measures the deviation from the reference trajectory, incorporating constraint violations through a penalty term (L).

Stage 1: Equivalent Masses Identification

In the first stage, following the assumptions stated in the reference paper [1], external forces and joint-level friction are neglected. The controller is designed with feedback linearization and feedforward action only. When the resulting control law is substituted back into the dynamics equation, the ideal behavior is retrieved:

$$\ddot{\mathbf{q}} = \ddot{\mathbf{q}}_{\mathbf{d}} \tag{6}$$

Therefore, the robot should perfectly track the desired joint accelerations. For this reason, the objective function is designed in order to minimize the mean and maximum acceleration tracking errors over all joints:

$$J(\mathbf{m}) = \sum_{i=1}^{n} (|\ddot{\mathbf{e}}_i^{\text{mean}}| + |\ddot{\mathbf{e}}_i^{\text{max}}|) + L \tag{7}$$

where $\ddot{\mathbf{e}}_i(t) = \ddot{\mathbf{q}}_{d,i}(t) - \ddot{\mathbf{q}}_i(t)$ is the acceleration error for the i-th joint i with $n = 7$. The penalty term L is applied when the absolute joint position error exceeds an arbitrary threshold ($\bar{e} = 5°$):

$$L = 100 \cdot e^{-\frac{\xi}{T}}, \quad \text{if } |\mathbf{e}_i(\xi)| > \bar{e} \tag{8}$$

where $\xi \leq T$ is the time of first constraint violation, and T is the total simulation time.

It is worth mentioning that, in the implementation, a minimal proportional control action is retained setting $K_p = 30\,\text{Nm/rad}$ for all joints, in order to prevent numerical drift and ensure stabilization of the simulation.

Stage 2: PID Gains Optimization

Once the equivalent joint masses $\mathbf{m}$ are obtained, the second stage optimizes the full set of PID gains by minimizing the tracking error. In particular, position and velocity errors over the simulation time T are incorporated in the cost function:

$$J(\mathbf{K_p}, \mathbf{K_d}, \mathbf{K_i}) = \sum_{i=1}^{n} (|\mathbf{e}_i^{\text{mean}}| + |\mathbf{e}_i^{\text{max}}| + |\dot{\mathbf{e}}_i^{\text{mean}}| + |\dot{\mathbf{e}}_i^{\text{max}}|) + L \tag{9}$$

where $\mathbf{e}_i(t) = \mathbf{q}_{r,i}(t) - \mathbf{q}_i(t)$ and $\dot{\mathbf{e}}_i(t) = \dot{\mathbf{q}}_{r,i}(t) - \dot{\mathbf{q}}_i(t)$ are the position and velocity errors for the i-th joint. The penalty term L is defined to penalize excessive tracking errors or unstable configurations during the simulation, and is defined as:

$$L = \begin{cases} 1000 \cdot e^{-t/T}, & \text{if } |e_i(t)| > \bar{e}_{\text{safe}} = 1.5° \\ 100 \cdot e^{-t/T}, & \text{if } |e_i(t)| > \bar{e}_{\text{track}} = 0.8° \end{cases} \tag{10}$$

The first threshold, $\bar{e}_{\text{safe}} = 1.5°$, is introduced to prevent the robot from entering potentially unstable configurations that could compromise safety. If this threshold is exceeded, the simulation is immediately terminated and a large penalty is applied, which significantly impacts the surrogate model and steers the optimization away from unsafe regions of the parameter space. The second threshold, $\bar{e}_{\text{track}}$, is used to penalize tracking errors that are smaller than the safety limit but still considered unsatisfactory. The penalty is defined in a way that earlier violations are penalized more heavily leading to a better research of the global minima.

PID Gains Optimization with Torque Control

To account for the physical torque limits of the robot, as reported in Table 1, an additional term was introduced into the original cost function to represent the maximum torque exerted by the joints. This torque term was appropriately weighted to ensure that its contribution is scaled consistently with respect to the other performance criteria.

Table 1: Maximum joint torque limits for the Franka Emika Panda robot. [2]

Joint Number	1	2	3	4	5	6	7
Max Torque $\bar{\tau}$ [Nm]	±87	±87	±87	±87	±12	±12	±12

Furthermore, the penalty mechanism was updated to discard any solution in which the torque on these joints exceeds the maximum values specified in the robot's datasheet, ensuring that the resulting trajectories remain physically feasible. The resulting cost function is:

$$J(\mathbf{K_p}, \mathbf{K_d}, \mathbf{K_i}) = \sum_{i=1}^{n} \left(|\mathbf{e}_i^{\text{mean}}| + |\mathbf{e}_i^{\max}| + |\dot{\mathbf{e}}_i^{\text{mean}}| + |\dot{\mathbf{e}}_i^{\max}|\right) + 0.1 \sum_{i=1}^{n} \left(|\boldsymbol{\tau}_i^{\max}|\right) + L \tag{11}$$

where the penalty term L is now defined with a new criterion in order to prioritize the torque minimization while again penalizing earlier violations:

$$L = 100 \cdot e^{-\frac{\xi}{T}}, \quad \text{if } |\boldsymbol{\tau}_{\max}| > \bar{\tau} \tag{12}$$

Sequential PID Optimization

A sequential strategy is employed in the second stage of the optimization process, where PID gains are optimized one joint at a time, from joint 7 to joint 1. At each step, only the gains of the current joint i are optimized, while the gains of previously optimized joints ($j > i$) are kept fixed to their last optimized values. Once optimized, the gains of joint i are also frozen and used in the subsequent optimizations.

The cost function for the i-th joint accounts not only for its own tracking performance but also for that of all higher-numbered joints ($j > i$), because of the dynamic coupling between joints. The cost function is defined as:

$$J_i(K_{p_i}, K_{d_i}, K_{i_i}) = \sum_{j=i}^{n} \left(\left|\mathbf{e}_j^{\text{mean}}\right| + \left|\mathbf{e}_j^{\max}\right| + \left|\dot{\mathbf{e}}_j^{\text{mean}}\right| + \left|\dot{\mathbf{e}}_j^{\max}\right|\right) + L \tag{13}$$

where $n = 7$ is the number of joints, and L is the penalty term for torque violations or constraint violations.

Regarding the behavior of lower-numbered joints ($j < i$), two different conditions are considered during the simulation: a first optimization assume that all the lower joints perfectly follow the reference trajectory. This represents an ideal scenario that could not be reproduced in a real calibration experiment. For this reason, a second optimization is performed where all the lower joints are kept stationary (initial position, zero velocity and acceleration). This configuration is feasible and representative of the actual calibration setup.

The sequential formulation is primarily used to reduce the dimensionality of each sub-problem, now optimizing only three variables for each sub-problem and therefore allowing for a faster minimization.

2.4 Bayesian Optimization Settings

To solve all the optimization stages, MATLAB's `bayesopt` function is used. This function implements a Bayesian Optimization (BO) framework based on Gaussian Process (GP) regression, balancing exploration of the unknown parameter space and exploitation of known promising regions through the use of the acquisition function.

The adopted surrogate model is a Gaussian Process with automatic hyperparameter tuning. For the

acquisition function, `expected-improvement` is selected in the mass identification as it prioritizes regions with high potential for cost reduction, while `expected-improvement-plus` is adopted for the PID gains optimization stages, as it offers improved numerical stability and robustness in the presence of noise.

The number of seed points and maximum evaluations is adjusted for each stage, in particular 20 seed points and 100 maximum evaluations were chosen for the first stage mass optimization, 40 seed points and 100 max evaluations for the full PID tuning, 30 seed points and 50 max evaluations per joint for the sequential PID tuning. Seed points are the initial evaluations performed using randomly sampled points before the surrogate model begins guiding the optimization. These initial seed points serve to explore the design space broadly, providing the GP with a diverse dataset to fit its initial hyperparameters. This choice improves model stability and prevents premature convergence to suboptimal regions.

Finally, lower and upper bounds were imposed on the optimization variables in order to ensure exploration of a wide and physically meaningful space for each parameter. They are reported in Table 2.

	$\mathbf{m_1}$ kg	$\mathbf{m_3}$ kg	$\mathbf{m_5}$ kg	$\mathbf{m_7}$ kg	$\mathbf{K_{p_i}}$ Nm/rad	$\mathbf{K_{d_i}}$ Nms/rad	$\mathbf{K_{i_i}}$ Nm/rad/s
LB	0.5	2.0	3.25	0.375	0	0	0
UB	1.5	6.0	13.0	3.0	10000	100	300

Table 2: Lower and upper bounds for the optimization variables.

3 Results

This section presents the results obtained from the different optimization runs. The analysis includes the evolution of the cost function, the set of optimal parameters identified, and the corresponding performance metrics. Additionally, the joint trajectories of the robot are plotted and compared against the reference trajectory, highlighting the tracking errors across all joints.

3.1 Results: Mass Optimization

The first optimization phase focused on the optimization of the equivalent link-mass parameters that influence the dynamic compensation of the control law and the feedforward term of the torque.

Figure 1 and Figure 2 show the progression of the cost function value alongside the evolution of the four optimization variables over the course of the Bayesian Optimization procedure.

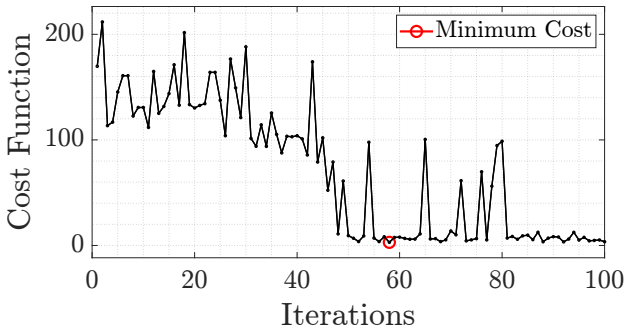

Figure 1: Evolution of the cost function value.

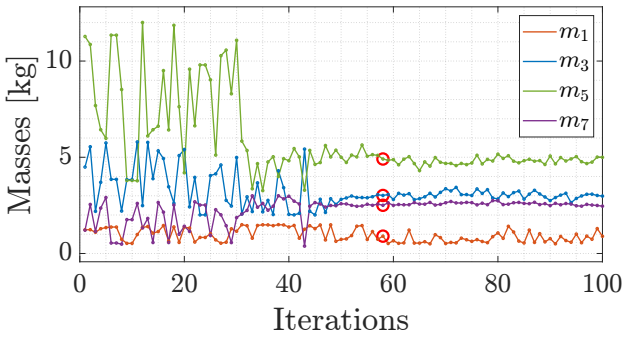

Figure 2: Evolution of the optimization variables.

The trend clearly reflects the adaptive sampling behavior of the surrogate-based framework: during the early iterations, the acquisition function favors exploration by prioritizing regions of high model uncertainty, thereby ensuring sufficient coverage of the input space. As expected, the penalty-triggering solutions occur mostly during the initial phase of the optimization, when the algorithm explores less known and potentially infeasible regions of the search space. As the optimization advances

and the surrogate model becomes more confident, the acquisition strategy shifts toward exploitation, concentrating the search around regions predicted to yield low cost values. This transition between exploration and exploitation is characteristic of Bayesian Optimization and is key to balancing global search efficiency with local refinement accuracy.

In Table 3 are presented the final results of the optimization in terms of minimum cost function and optimal equivalent link-masses. Since the optimization process is guided by a Gaussian Process, variations in initialization and surrogate model updates can lead to different outcomes across runs. Therefore, the following example is presented as a representative result rather than a unique solution.

Table 3: Minimum cost function value and optimal equivalent link masses.

J	m_1 [kg]	m_3 [kg]	m_5 [kg]	m_7 [kg]
2.9743	0.901	3.015	4.901	2.505

The cost function reaches relatively low values, indicating that the actual acceleration profile tends to follow the reference one.

3.2 Results PID Gain optimization

After the equivalent link-mass parameters have been optimized, the tuning of the PID gains is carried out. In this initial optimization, joint torque constraints were not explicitly included in the cost function, resulting in what can be considered a relatively unconstrained optimization. Figure 3 illustrates the evolution of the cost function during this second optimization phase. As observed, the cost function exhibits a trend that is qualitatively similar to the one obtained in the link-mass optimization, further highlighting the typical exploratoration-to-exploitation behavior of Bayesian Optimization.

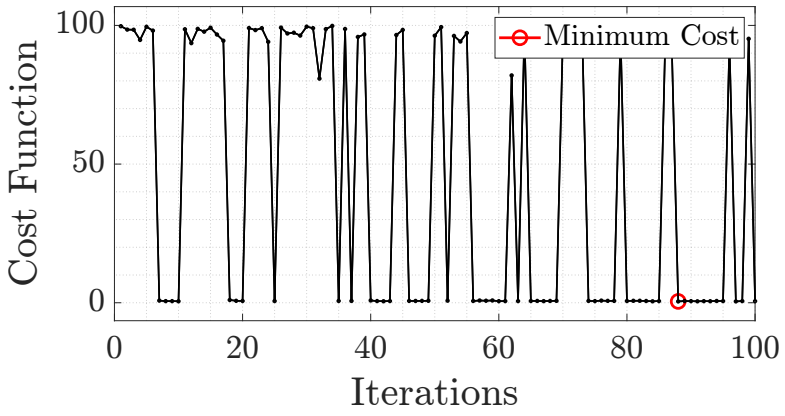

Figure 3: PID Optimization Cost Function vs Iterations.

The optimized PID gains resulting from this process are presented in Table 4. These values represent the controller parameters that achieved the lowest cost function value, indicating the best tracking performance.

Table 4: Optimized PID gains.

Joint N°	1	2	3	4	5	6	7
$K_p\,[Nm/rad]$	6667.2	5742.7	4290	6620.3	4787.7	7637.3	8130.7
$K_d\,[Nms/rad]$	74.656	66.96	29.807	9.6776	7.5722	43.116	50.817
$K_i\,[Nm/s/rad]$	21.056	162.3	152.34	231.85	27.057	81.927	193.63

Based on these optimized values, a simulation was carried out to evaluate the tracking performance of the controller, as well as to assess the control effort required by the robot to follow the trajectory imposed by the control algorithm. Figure 4 illustrates the tracking errors in terms of joint position and

velocity, while Figure 5 displays the corresponding joint torques required to execute the commanded motion.

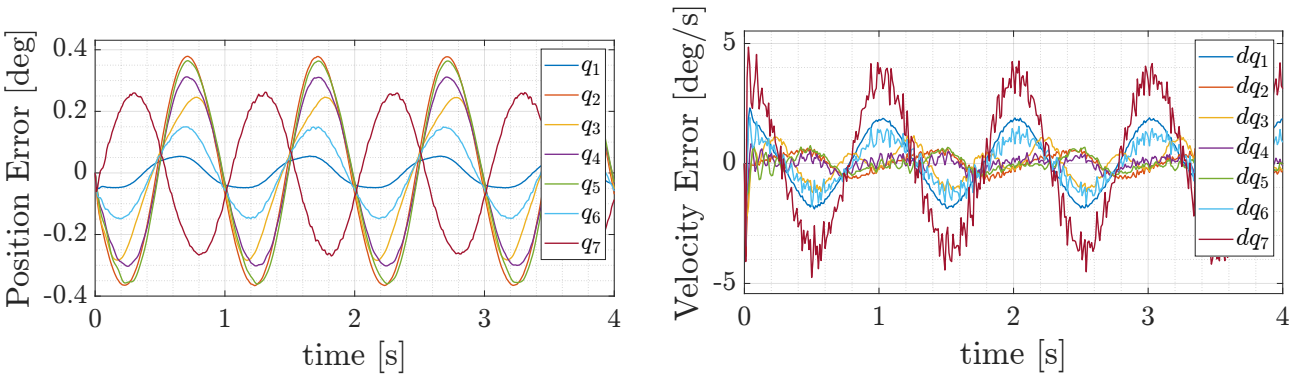

Figure 4: Position and Velocity errors.

The results clearly indicate satisfactory tracking performance, with a maximum position error of under 0.4°. On the other hand, the torque profiles reveal a less desirable behavior: joints 5, 6, and 7 exhibit excessive control effort, with peak torques exceeding the limits specified in the robot's datasheet.

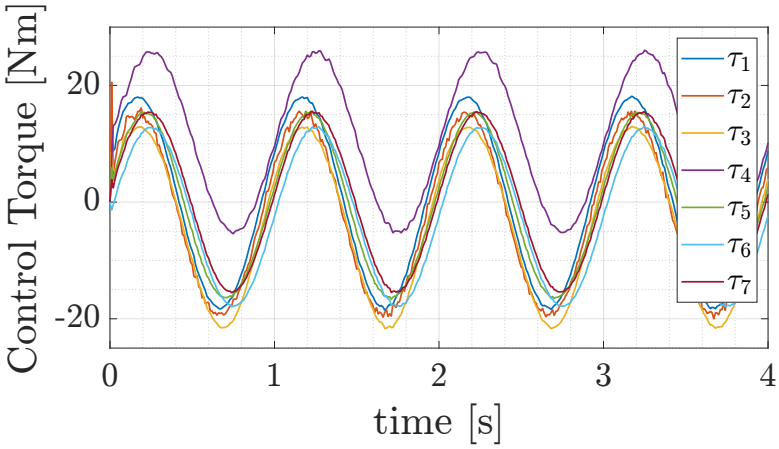

Figure 5: Joint Total Torque.

Due to the infeasibility of the previously obtained solution, the optimization was repeated using the modified cost function defined in Equation (11), which explicitly penalizes excessive joint torque. However, even with this updated formulation, the Bayesian Optimization failed to identify a solution that satisfies both the tracking accuracy and the torque constraints. In particular, while the new solution successfully respects the torque limits, it lacks the necessary control authority to accurately follow the reference trajectory, resulting in degraded tracking performance.

This issue will be further addressed by the team in Section 5, where the reference trajectory is modified, specifically by reducing the oscillation frequency of the joints, in order to enable a smoother control response with lower torque peaks, and thereby improve the overall feasibility of the solution.

3.3 Results of Sequential Optimization

To improve the efficiency of the tuning process, a sequential optimization strategy was adopted for the PID gains. As described in Section 2.3, the procedure was tested under two different configurations: an ideal case, in which the lower-numbered joints perfectly follow the reference trajectory, and a practical case, in which the lower-numbered joints are kept stationary to their initial positions.

Figure 6 shows the evolution of the cost function during the optimization of joint 1, chosen as a representative case because it is the last to be optimized and therefore influenced by the performance of all higher-numbered joints. Compared to the full PID optimization, the sequential strategy achieves

penalty-free solutions much earlier. This faster convergence is mainly due to the reduced dimensionality of the search space, and to the fact that the upper joints have already been individually optimized and can accurately follow the reference trajectory. As a result, the likelihood of encountering constraint violations or penalties during the optimization is significantly reduced.

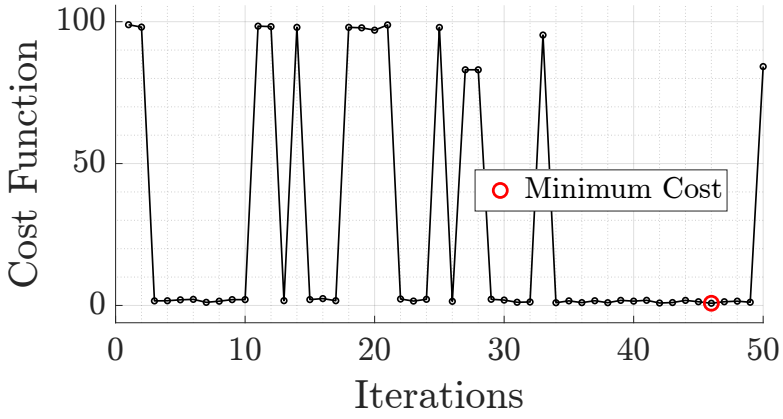

Figure 6: Sequential PID Optimization Cost Function vs Iterations (joint 1).

The final optimized PID gains obtained in the two cases are reported in Tables 5 and 6.

Table 5: Optimized PID gains – Ideal case.

Joint No	1	2	3	4	5	6	7
K_p	8582.7	2519.6	2468.4	3407.4	2127.6	6054.8	9999.8
K_d	94.3827	5.8679	21.7440	25.7222	5.7397	0.0814	47.8129
K_i	121.256	252.786	60.086	261.702	263.296	82.967	179.642

Table 6: Optimized PID gains – Practical case.

Joint No	1	2	3	4	5	6	7
K_p	2201.0	8886.1	8672.2	6395.9	6983.4	9585.7	7794.7
K_d	33.3069	62.6856	68.2907	40.1426	57.0064	80.4249	24.4819
K_i	194.663	134.658	287.416	2.571	230.981	51.363	65.412

Based on the optimized gains, the controller was tested in simulation to evaluate tracking performance and control effort. Figure 7 and Figure 8 compare the joint position and velocity errors between the two cases. As expected, the ideal case yields superior tracking performance, with reduced peak errors across all joints. In contrast, the practical case shows a degradation in tracking quality, particularly in the lower joints. This is due to the fact that, during the sequential optimization, joints are tuned in descending order, and in the practical setup, the joints not yet optimized are kept fixed at their initial positions introducing significant coupling effects in the system dynamics.

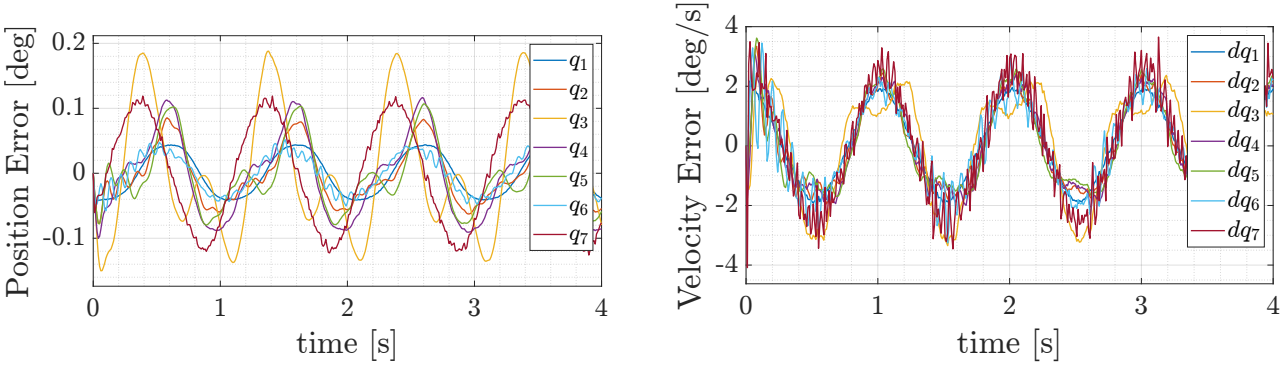

Figure 7: Position and Velocity errors for the ideal case.

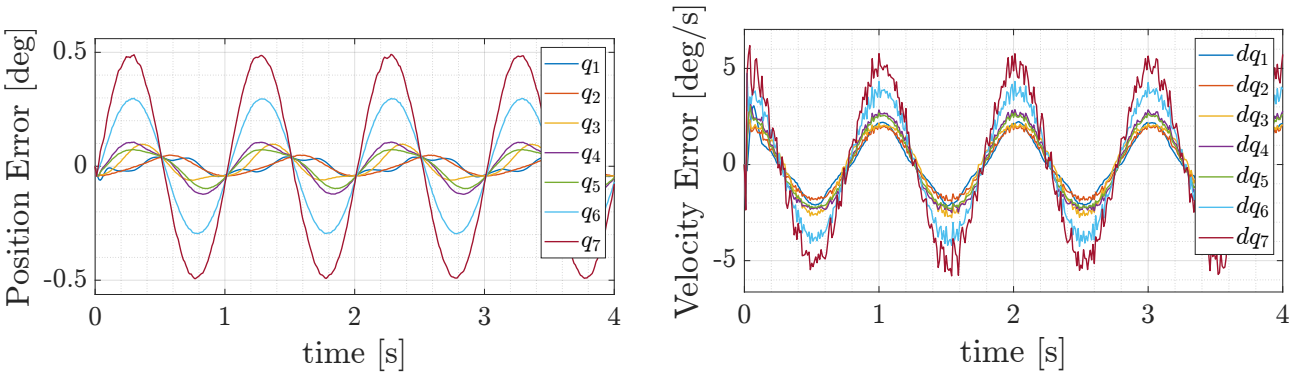

Figure 8: Position and Velocity errors for the practical case.

4 Performance Analysis

In this section some analyses will be conducted, particular attention will be given to the change in both kernel and acquisition functions. The performance of the Bayesian Optimization (BO) process heavily depends on the configuration of its internal components, such as the acquisition function and the surrogate model used to approximate the objective function. While the previous sections focused on the results obtained with a fixed BO setup, namely, using a Gaussian Process (GP) surrogate and the Expected Improvement (EI) acquisition function, this section aims to analyze how different design choices affect the quality and efficiency of the optimization.

It was analyzed the performance of alternative acquisition functions and kernel functions in the optimization of PID control gains. The analysis focuses on key metrics such as final cost function values, direction of improvement, and estimated mean and covariance of the surrogate model. This comparative study provides insights into the trade-offs involved in BO design.

4.1 Acquisition Functions

In the context of Bayesian Optimization (BO), the choice of acquisition function plays a central role in balancing exploration and exploitation. Two commonly adopted criteria are Probability of Improvement (PI) and Expected Improvement (EI) [3].

Probability of Improvement focuses solely on the likelihood of improving over the current best observation. This approach selects the next sample where the surrogate model predicts the highest probability of outperforming the incumbent minimum. While this strategy is intuitive and computationally efficient, it presents a significant limitation: it does not take into account the magnitude of the potential improvement. As a result, it may get trapped in local minima, repeatedly sampling in regions with high probability but low actual gain.

On the other hand, *Expected Improvement* addresses this limitation by weighting the probability of improvement with the expected gain. This leads to a more informed sampling policy that favors regions where either the predicted mean is low (exploitation) or the model uncertainty is high (exploration). The EI formulation therefore allows BO to explore more distant, uncertain regions of the search space, potentially escaping local optima. Suppose $f' = \min(f)$ is the min value of f observed so far, the reward function is defined as stated in Equation 14

$$u_{\mathrm{PI}}(\mathbf{x}) = \begin{cases} 1, & \text{if } f(\mathbf{x}) \leq f' \\ 0, & \text{otherwise} \end{cases} \qquad u_{\mathrm{EI}}(\mathbf{x}) = \max\left(0, f' - f(\mathbf{x})\right) \tag{14}$$

Both the two acquisition functions were adopted in the PID gains optimization. The comparison is illustrated in Figure 9, where the optimization progress under both acquisition functions is shown. The plot on the left corresponds to PI, while the one on the right corresponds to EI.

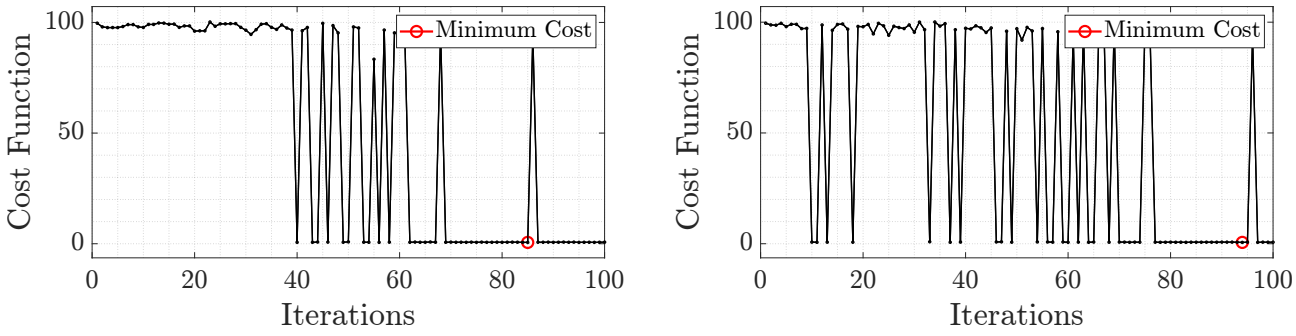

Figure 9: Probability of Improvement vs Expected Improvement.

As evident, EI results in a more diverse set of admitted solutions, characterized by multiple sharp drops in the objective function, indicating that the optimizer successfully identified several promising regions. Moreover, EI consistently exhibits a lower final cost, suggesting that it may have reached the global minimum.

In contrast, PI displays a lot of unallowable solutions which trigger the predefined penalty, with only a few good solutions. This behavior confirms that PI lacks the necessary exploratory behavior to escape local optima especially in high-dimensional or noisy cost landscapes. Moreover, it finds the first optimal solution quite later with respect to the EI acquisition function.

In summary, PI shows limited exploration capability and makes it less suitable for complex tuning tasks and lead the optimization to stuck in already explored regions. Expected Improvement, on the other side, demonstrated superior performance both in terms of convergence depth and diversity of explored regions, making it the preferred acquisition strategy in our application.

In addition to the standard `expected-improvement`, it was also tested `expected-improvement-plus` Matlab acquisition function. The latter is specifically designed for stochastic or noisy objective functions, where repeated evaluations at the same point may yield different results. In such cases, the acquisition function accounts for uncertainty in the observations by introducing a more conservative selection policy.

This choice is particularly relevant in this specific application, where it was artificially introduced a torque measurement noise to model the limited precision of the actuation system. As a result, the optimization process becomes non-deterministic, justifying the use of the noise-aware expected-improvement-plus strategy.

Preliminary tests confirmed that this acquisition function improves robustness and provides a more realistic optimization behavior in the presence of noise, especially during the PID tuning stage. The injected noise simulates real-world actuation uncertainties, making the optimization more representative of physical conditions.

4.2 Kernel Functions

The Gaussian Process (GP) in is in general defined by a mean function and by a kernel function. In particular, the kernel (or covariance function) $k(\mathbf{x}, \mathbf{x}')$ controls the shape of the function at specific points based on the distance between actual data. The choice of kernel has a direct impact on the exploration-exploitation trade-off handled by the acquisition function.

Two widely used kernel functions in Bayesian Optimization are the *Squared Exponential* and the *Matérn 3/2* kernels, which differ in the level of smoothness they impose on the modeled function.

The Squared Exponential (SE) kernel assumes a very smooth, infinitely differentiable function. Its expression is given by:

$$k_{\mathrm{SE}}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(-\frac{1}{2\ell^2}\|\mathbf{x} - \mathbf{x}'\|^2\right) \tag{15}$$

where: σ_f^2 is the signal variance (amplitude) and $\ell > 0$ is the length-scale hyperparameter,

The Matérn 3/2 kernel is the generalization of the Radial Basis function. Compared to the SE kernel, the Matérn 3/2 kernel produces functions that are once differentiable, offering a better fit for non-smooth or noisy data. It is defined as:

$$k_{\mathrm{M32}}(x_i, x_j) = \sigma_f^2 \left(1 + \frac{\sqrt{3}\, d(x_i, x_j)}{\ell}\right) \exp\left(-\frac{\sqrt{3}\, d(x_i, x_j)}{\ell}\right) \tag{16}$$

where: d is the distance between input point.

To further investigate the influence of the Gaussian Process kernel on the behavior of the surrogate model, an a posteriori analysis was conducted focusing on two gains of the first joint controller: the proportional and derivative gains (Kp1 and Kd1).

After completing the BO procedure for PID tuning, the full set of evaluated points and their corresponding objective function values was extracted and filtered by removing all the points that triggered a penalty during optimization. This filtering step allows the surrogate model to provide a more accurate estimation of both the predicted mean and variance. This dataset was used to fit two separate surrogate models a posteriori, one using the Squared Exponential (SE) kernel and the other the Matérn 3/2 kernel via `fitrgp` Matlab function. The goal of this analysis was to visualize and compare how each kernel generalizes the cost landscape and quantifies uncertainty in unexplored regions. By fixing all other parameters and varying only Kp1 and Kd1 (after standardization), contour plots of both the predicted cost function and the associated standard deviation were plotted over the parameter space.

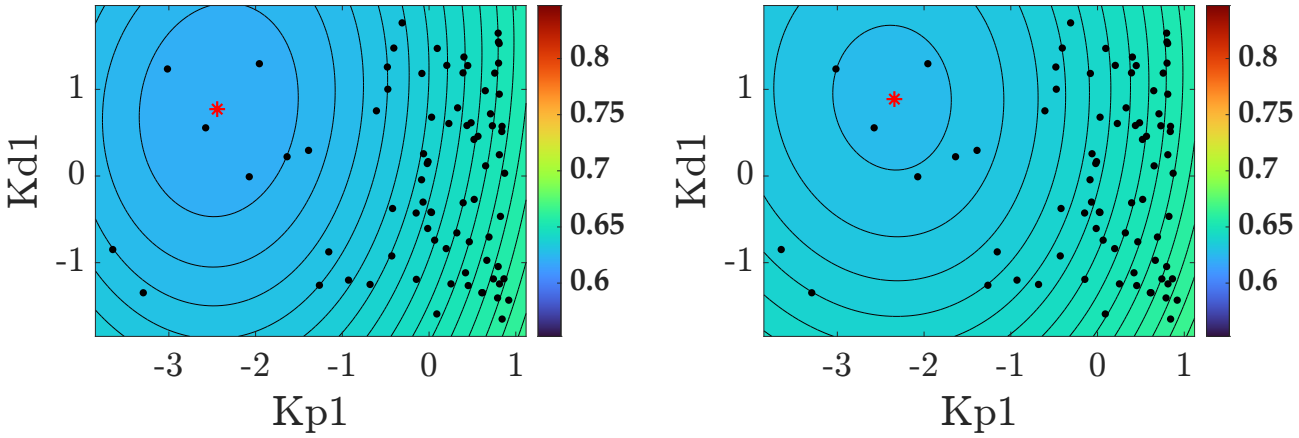

Figure 10: GP Estimated Mean - Squared Exponential vs Matern 3/2

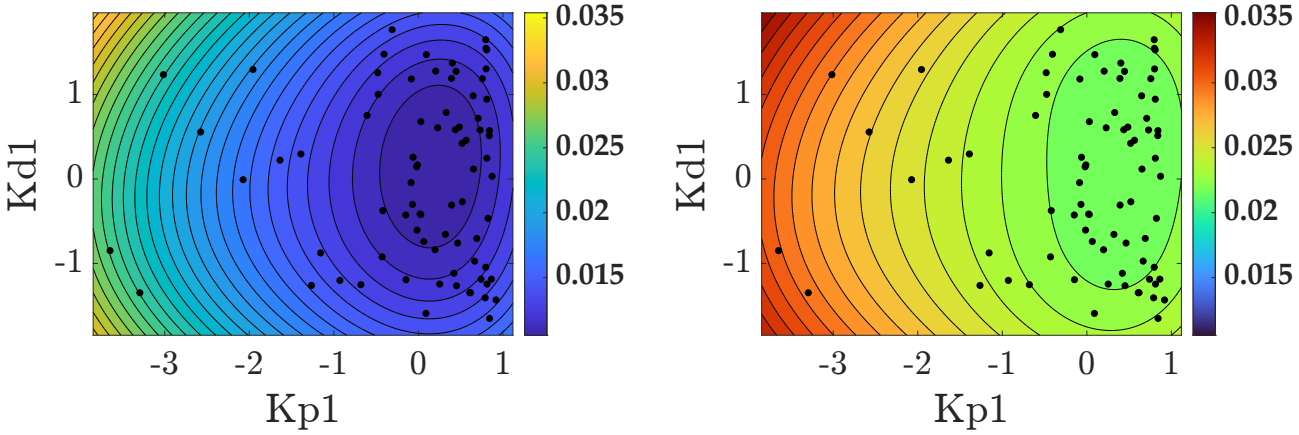

Figure 11: GP Standard Deviation - Squared Exponential vs Matern 3/2

Figure 10 illustrates the predicted mean value of the Gaussian Processes while Figure 11 the corresponding standard deviations (bottom row) for both kernels.

Both kernels produce consistent predictions for the objective function in the explored input region (defined by Kp1 and Kd1). However, the SE kernel yields a slightly smoother landscape due to its assumption of infinite differentiability, which tends to produce more optimistic minima. In contrast, the Matérn 3/2 kernel, which assumes lower smoothness, results in more aggressive contours around the predicted optimum. Furthermore, the global minima found with Squared Exponential kernel is a little below the one computed with Matern 3/2 kernel function.

Regarding the uncertainty, both kernels exhibit higher variance in low-sampled regions, particularly near the edges of the input domain. The SE kernel appears more confident (lower variance) near the predicted minimum, likely due to its stronger prior assumptions, whereas the Matérn 3/2 kernel provides a more conservative estimate with slightly higher uncertainty across the domain. This aligns with the theoretical behavior of the kernels: SE tends to overfit, while Matérn is more robust to noise and sparse sampling and so it could be preferred for exploration in a Bayesian Optimization framework.

5 Validation of the model

To validate the performance of the optimized controller and the underlying model developed in the previous sections, the optimized PID gains have been tested following a new reference sinusoidal joint trajectory. This latter, adopted from the literature [1], differs from the original in both frequency and amplitude. Specifically, each joint is now assigned a distinct oscillation frequency and amplitude, providing a more challenging and realistic validation scenario.
This validation step is designed to assess the performance of the optimized PID controllers (both AAO and sequential) in a realistic test scenario inspired by the literature, verifying their ability to ensure acceptable tracking errors and feasible torque actions under altered trajectory conditions.
In Table 7 the updated frequencies and amplitudes values are reported.

Table 7: Frequencies and amplitudes of the validation reference trajectory for each joint

Joint	1	2	3	4	5	6	7
Frequency [Hz]	0.05	0.075	0.075	0.10	0.10	0.10	0.10
Amplitude [°]	30	10	10	10	10	10	10

To faithfully reproduce the optimization setup presented in the reference literature, the simulation time was increased to 20 seconds. This extended time horizon not only aligns with the original experimental conditions, but also provides a clearer observation of the system's behavior under lower-frequency excitation. Given the reduced oscillation frequencies of the reference trajectory, a longer simulation window allows multiple full cycles to be captured, thereby making the oscillatory dynamics more evident and facilitating a more accurate evaluation of tracking performance and torque profiles. The validation step has been performed by using both the PID gains of the Sequential optimization (Table 5) and the gains of the AAO optimization (Table 4).

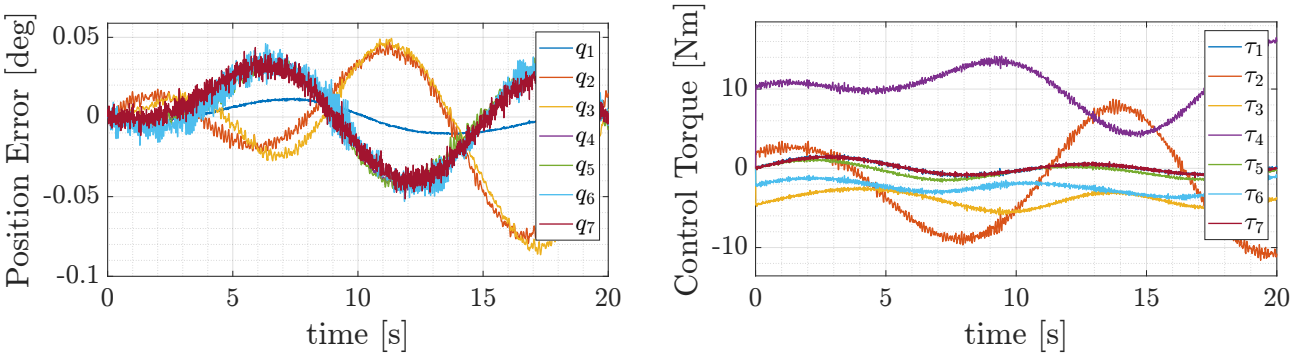

Figure 12: Maximum Tracking Error and Total Torque for Sequential Optimization.

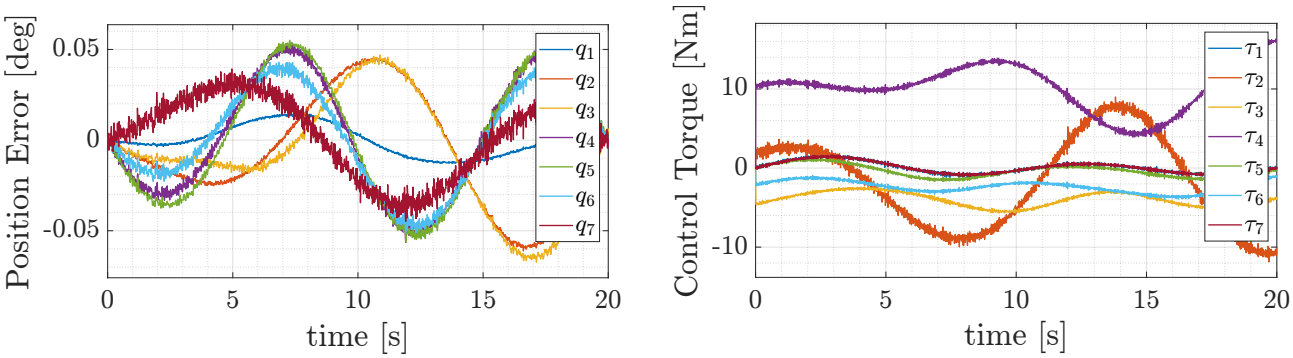

Figure 13: Maximum Tracking Error and Total Torque for AAO Optimization.

The results obtained with the modified reference trajectory confirm the validity of the proposed model and control strategy, for both AAO and Sequential approaches. In particular, the lower frequency excitation leads to a smoother and less aggressive joint motion, which directly reflects in the resulting torque profiles. Figure 12 and 13 on the left show the joints tracking error for both cases. Compared to the results obtained with the original, higher-frequency trajectory, the tracking errors are noticeably reduced across all joints. This improvement is expected, as lower-frequency motions are inherently easier to follow for a feedback controller with limited bandwidth.
Compared to the original trajectory used during optimization, the torque signals now exhibit significantly reduced peaks and smoother transitions. Most importantly, the torques required by the PID controllers remain always below the maximum admissible values defined in the Franka Emika Panda robot datasheet, for both AAO and Sequential cases. This outcome supports the accuracy of the dynamic model and demonstrates that the optimized parameters maintain safe and stable behavior even under different operating conditions. The successful generalization to a new trajectory further validates the robustness of the tuning procedure.

6 Closing comments and future improvements

In this work, a two-stage Bayesian Optimization method for the auto-tuning of a robot manipulator's PID gains was presented. The first stage focused on identifying the equivalent link masses to improve the accuracy of dynamic compensation, while the second stage optimized the PID gains for each joint with respect to trajectory tracking performance. For the second stage, both full and sequential strategies were tested. In each case, a Gaussian Process was adopted as the surrogate model, and `expected-improvement-plus` was used as the acquisition function.
The proposed methodology demonstrated good performance in tracking the prescribed reference trajectory, providing satisfactory results in terms of joint position and velocity errors, despite some difficulty in simultaneously satisfying both tracking accuracy and torque constraints. In particular, the sequential PID optimization proved to achieve faster convergence by reducing the dimensionality of the search space.

Future improvements could address multiple aspects of the current methodology, both on the modeling side and the optimization framework. One possible enhancement would be to include joint friction in the feedback linearization control law, as it would allow a more realistic compensation of physical dynamics. Even though it would introduce additional complexity and may compromise numerical stability, it would be a valuable extension.

Another improvement would be to explicitly model the Coriolis term to improve the accuracy of the dynamic model, particularly under faster joint movements. However, the additional computational cost associated with evaluating and inverting this term at each time step would have to be faced, especially if real-time applicability is considered.

Another possible improvement would be to test the generalizability of the tuned controller using more diverse reference trajectories. In particular, Cartesian trajectories like those used in the original literature via inverse kinematics would better reflect real-world manipulation tasks.

As for the cost function side, new performance metrics such as the time derivative of joint torque ($\dot{\boldsymbol{\tau}}$) could be introduced. Penalizing high-frequency torque changes might help avoid chattering and reduce mechanical stress, which is especially relevant for real-world deployment.

Finally, further gains in optimization performance could be achieved by fine-tuning the kernel hyperparameters of the Gaussian Process model. This could be done using tools like MATLAB's `fitrgp` to better adapt the surrogate model to the structure of the objective function landscape.

In conclusion, the results obtained by this work suggest that data-driven optimization strategies can be effectively integrated into modern robotic calibration pipelines, reducing the need for manual tuning and improving reliability in complex and real-world tasks.

References

[1] Loris Roveda, Marco Forgione, Dario Piga, "Two-Stage Robot COntroller Auto-Tuning Methodology for Trajectory Tracking Application," 2020.

[2] "FRANKA EMIKA ROBOT'S INSTRUCTION HANDBOOK," 2021.

[3] "Machine Learning for Mechanical Systems Course Notes, 2025."