Assignment 02: Path Tracking for a Robot Manipulator

Luca Podavini 257844 Matteo Grisenti 257855

November 2025

1 Path Tracking

OCP Formulation:

$$
\begin{aligned}
\min_{X,U,S,W} \quad & \sum_{k=0}^{N-1} \left(w_v \|\dot{q}_k\|^2 + w_a \|u_k\|^2 + w_w \|w_k\|^2 \right) \\
\text{s.t.} \quad & q_{k+1} = q_k + d_t\,\dot{q}_k, \qquad \dot{q}_{k+1} = \dot{q}_k + d_t\,u_k, && \forall k \in [0, N-1], \\
& q_{\min} \le q_k \le q_{\max}, \qquad -\dot{q}_{\max} \le \dot{q}_k \le \dot{q}_{\max}, && \forall k \in [1, N], \\
& \tau_{\min} \le \tau_k \le \tau_{\max}, && \forall k \in [1, N-1], \\
& s_{k+1} = s_k + d_t\,w_k, && \forall k \in [1, N-1], \\
& y(q_k) = p(s_k), && \forall k \in [1, N], \\
& x(0) = x_{\text{init}}, \qquad s(0) = 0, \qquad s(N) = 1.
\end{aligned}
$$

After implementing the cost and constraint functions as specified in the OCP formulation, the resulting trajectories demonstrate several key behaviors inherent to this path-tracking formulation:

- **Perfect Path Adherence**: The end-effector's position perfectly tracks the specified *'infinity'* path. This is an expected outcome, as the formulation enforces a strict equality constraint $y(q_k) = p(s_k)$, which forces the position of the end-effector $y(q)$ to match the path coordinate $p(s)$ at every step k.

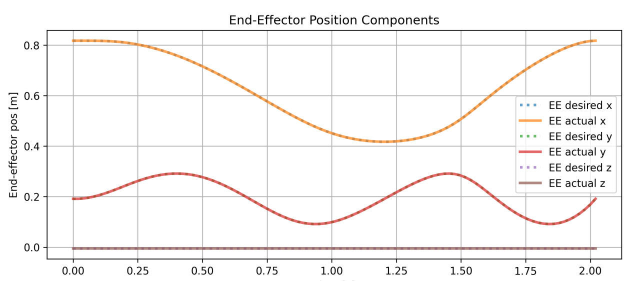

Figure 1: End-effector position components (Q1).

- **Variable Path Velocity (Slow Start, Fast Finish)**: The robot does not progress along the path at a constant speed. It begins slowly and then significantly accelerates as it approaches the final time step (Figure 2). This behavior is a direct consequence of the optimization objective, that attempts to minimize the running costs, including the path progress input (w_k).

To keep the cost low, it prefers to move slowly (low w_k) at the beginning. However, it is bound by the hard terminal constraint $s_N = 1$, meaning it must complete the entire path within the fixed N time steps. This forces the optimizer to increase the value of w_k towards the end, resulting in a final burst of speed to satisfy the constraint. This acceleration is clearly visible in the plot of the w_k trajectory.

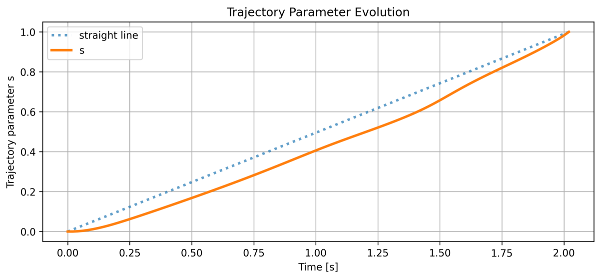

Figure 2: Trajectory parameter s (Q1).

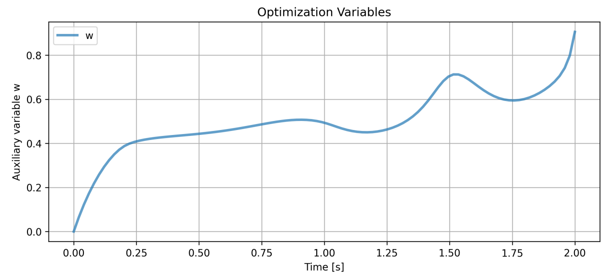

Figure 3: Path velocity w (Q1).

2 Cycling Path Tracking

OCP Formulation:

$$\min_{X,U,S,W} \quad \sum_{k=0}^{N-1} \left(w_v \|\dot{q}_k\|^2 + w_a \|u_k\|^2 + w_w \|w_k\|^2 \right) \qquad + \; w_{\text{final}} \, \|x(N) - x(0)\|^2$$

$$\begin{aligned}
\text{s.t.} \quad & q_{k+1} = q_k + d_t\,\dot{q}_k, \qquad \dot{q}_{k+1} = \dot{q}_k + d_t\,u_k, && \forall k \in [0, N-1],\\
& q_{\min} \le q_k \le q_{\max}, \qquad -\dot{q}_{\max} \le \dot{q}_k \le \dot{q}_{\max}, && \forall k \in [1, N],\\
& \tau_{\min} \le \tau_k \le \tau_{\max}, && \forall k \in [1, N-1],\\
& s_{k+1} = s_k + d_t\,w_k, && \forall k \in [1, N-1],\\
& y(q_k) = p(s_k), && \forall k \in [1, N],\\
& x(0) = x_{\text{init}}, \qquad s(0) = 0, \qquad s(N) = 1.
\end{aligned}$$

For this question, we introduced a terminal cost to encourage cyclic behavior, penalizing the distance between the initial and final robot state ($x(0)$ and $x(N)$) with a weight of $w_{final} = 1$.

- **Impact on Final State**: This new terminal cost successfully modifies the robot's behavior. The final joint position $q(N)$ and joint velocities $\dot{q}(N)$ now are significantly closer to the initial state q_0, $\dot{q}_0$ than in the previous scenario. Though not identical, as the cost $w_{final} = 1$ still needs to be balanced against the running costs.

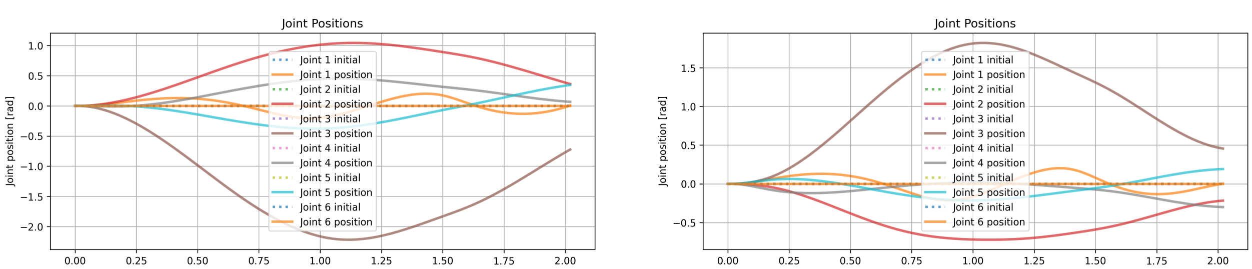

Figure 4: Joint positions comparison: without terminal cost (left) vs with terminal cost (right)

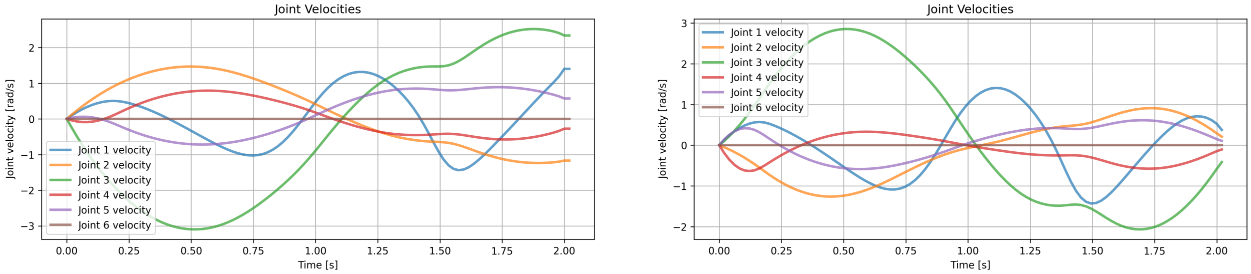

Figure 5: Joint velocities without terminal cost (left) vs with terminal cost (right)

- **Trade-off and Torque**: A primary consequence is an increase in the required joint torques throughout the trajectory. This occurs becouse the optimizer is now balancing multiple, competing objectives. The addition of the terminal cost makes the minimization of running costs (such as velocities and joint accelerations/torques) relatively less relevant.

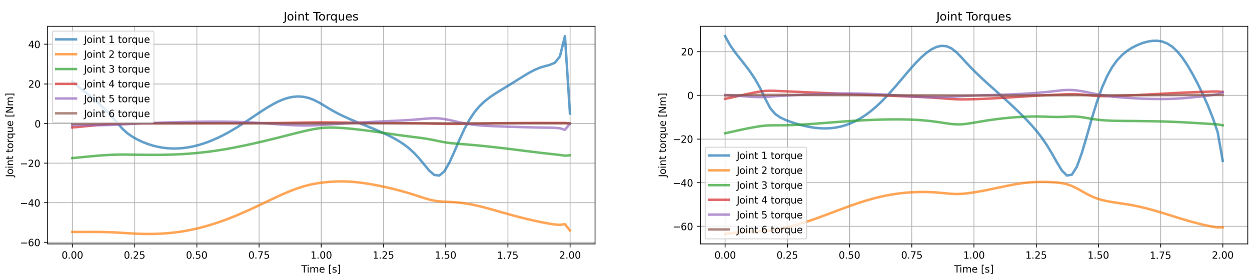

Figure 6: Joint torques without terminal cost (left) vs with terminal cost (right)

- **Solver Strategy**: The new optimizer reduces the emphasis on minimizing w_k, allowing it to focus more on completing the path and advancing faster along the reference trajectory. This behavior can be seen in the path velocity graph $w(t)$, which shows higher values at the beginning compared to the previous experiment without the cycling cost, and reaches a peak around 60% of the path. After this point, $w(t)$ decreases because the robot is already well aligned with the path.

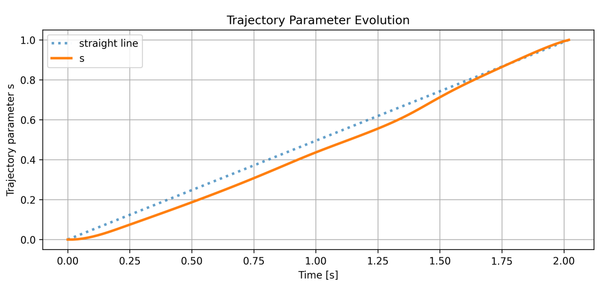

Figure 7: Path parameter s (Q2).

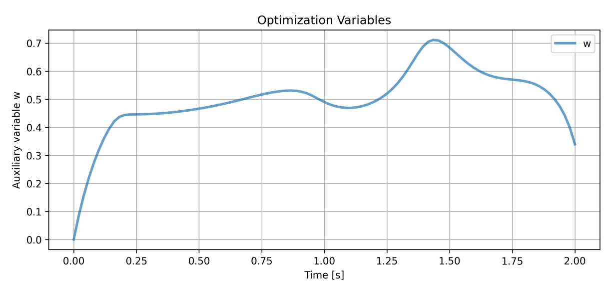

Figure 8: Path velocity w (Q2).

As a result, path tracking is faster at the beginning, leaving more steps available at the end of the horizon. During this final phase, the robot can focus on the cycling task, organizing its motion to reach a joint configuration as close as possible to the initial state $x(0)$. This is evident in Figure 9, where the end of the path is denser and contains more steps.

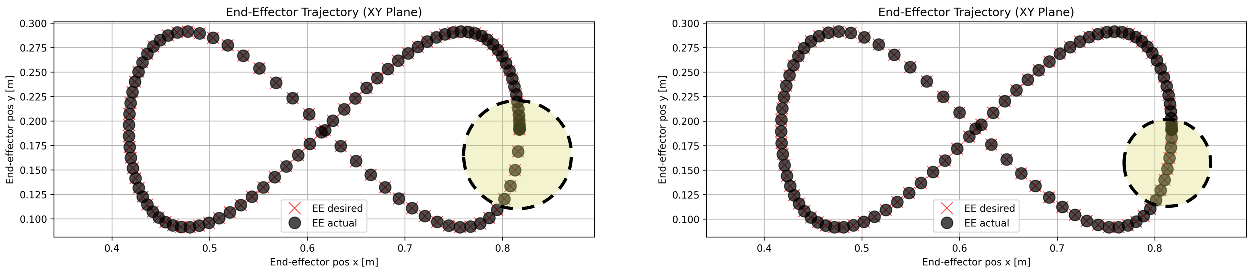

Figure 9: End-effector trajectory xy without terminal cost (left) vs with terminal cost (right)

Addition: With this set of weights, the robot is not able to fully recover the initial configuration, so a cycling behavior is not possible. We investigated a solution to enable the cycling behavior by increasing the weight of the cycling terminal cost. We found that a w_{final} of 10^3 allows the robot to reach a nearly perfect final configuration.

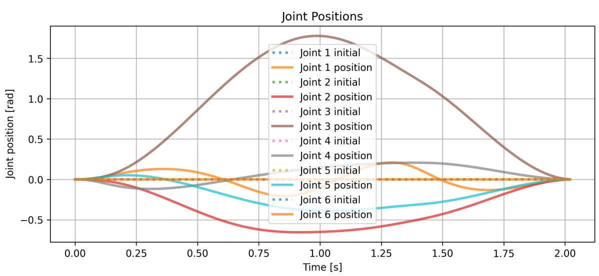

Figure 10: Joint position ($w_{\text{final}} = 10^3$).

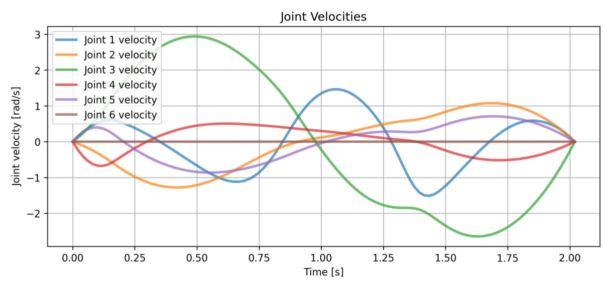

Figure 11: Joint velocities ($w_{\text{final}} = 10^3$).

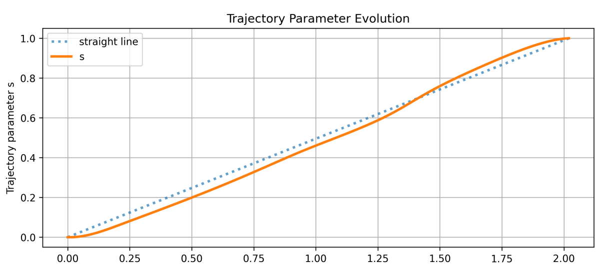

Figure 12: Path parameter s ($w_{\text{final}} = 10^3$).

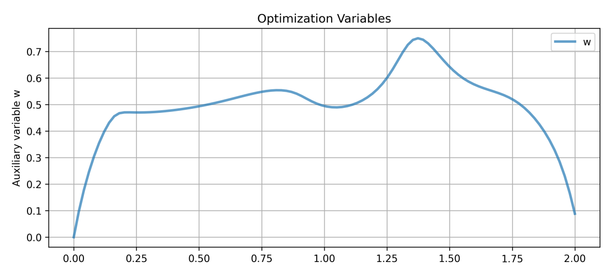

Figure 13: Path velocity w ($w_{\text{final}} = 10^3$).

3 Cycling Trajectory Tracking

In the trajectory, the path parameter S and the path velocity W are fixed. The problem requires $w(t) = 1/N$, which makes S a linear sequence ($s_k = k/N$). This means that the state X and control U sequences are the only remaining optimization variables.

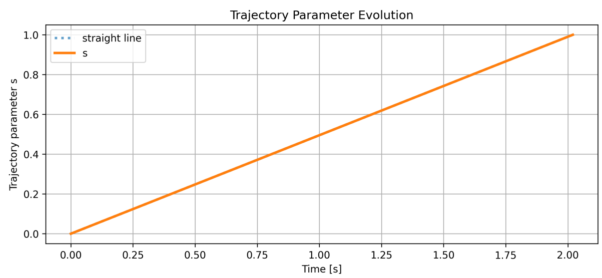

Figure 14: Fixed path parameter s_k (Q3).

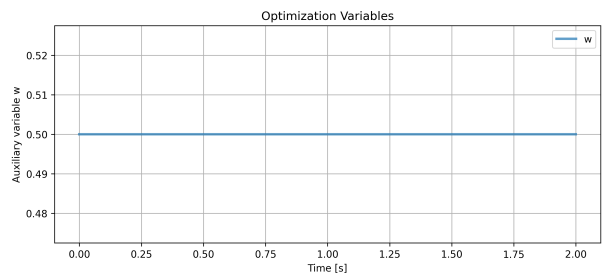

Figure 15: Path velocity w_k (Q3).

3.1 Trajectory Tracking Hard Constraint

OCP Formulation:

$$
\begin{aligned}
\min_{X,U} \quad & \sum_{k=0}^{N-1} \left(w_v \|\dot{q}_k\|^2 + w_a \|u_k\|^2 \right) \qquad + \; w_{\text{final}} \|x(N) - x(0)\|^2 \\
\text{s.t.} \quad & q_{k+1} = q_k + d_t \, \dot{q}_k, \quad \dot{q}_{k+1} = \dot{q}_k + d_t \, u_k, && \forall k \in [0, N-1] \\
& q_{\min} \le q_k \le q_{\max}, \quad \dot{q}_{\min} \le \dot{q}_k \le \dot{q}_{\max}, && \forall k \in [1, N] \\
& \tau_{\min} \le \tau_k \le \tau_{\max}, && \forall k \in [1, N-1] \\
& s_{k+1} = s_k + 1/N, && \forall k \in [1, N-1] \\
& y(q_k) = p(s_k), && \forall k \in [1, N] \\
& x(0) = x_{\text{init}}, \qquad s(0) = 0, \qquad s(N) = 1.
\end{aligned}
$$

The main difference from the previous formulation, where S and W were part of the decision process, is that in the previous model, the optimizer had the flexibility to decide *how much* to advance at each time step, allowing it to adapt the path timing based on the robot's current configuration. However, enforcing this new, rigid schedule as a **hard constraint** ($y(q_k) = p(s_k)$) was found to render the problem **unfeasible**.

By removing the optimizer's ability to modulate the path velocity, there are inevitable points where the system cannot physically achieve the imposed advancement. In other words, the robot cannot reach the target path point $p(s_k)$ at the pre-determined time k.

3.2 Trajectory Tracking Soft Constraint

This unfeasibility, which was a clear issue with the hard-constraint approach, was subsequently managed by reformulating the tracking requirement as a cost term (i.e., a soft constraint).

OCP Formulation:

$$
\begin{aligned}
\min_{X,U} \quad & \sum_{k=0}^{N-1} \left(w_v \|\dot{q}_k\|^2 + w_a \|u_k\|^2 + w_p \|y_q - p_s\|^2 \right) \qquad + w_p \|y(q_N) - p(s_N)\|^2 + w_{\text{final}} \|x(N) - x(0)\|^2 \\
\text{s.t.} \quad & q_{k+1} = q_k + d_t\, \dot{q}_k, \quad \dot{q}_{k+1} = \dot{q}_k + d_t\, u_k, \qquad \forall k \in [0, N-1] \\
& q_{\min} \le q_k \le q_{\max}, \quad \dot{q}_{\min} \le \dot{q}_k \le \dot{q}_{\max}, \qquad \forall k \in [1, N] \\
& \tau_{\min} \le \tau_k \le \tau_{\max}, \qquad \forall k \in [1, N-1] \\
& s_{k+1} = s_k + 1/N, \qquad \forall k \in [1, N-1] \\
& x(0) = x_{\text{init}} \quad s(0) = 0 \quad s(N) = 1
\end{aligned}
$$

The resulting end-effector trajectory, as expected, is not perfectly accurate. This is because a perfect solution does not exist, as indicated by the infeasibility of the corresponding hard-constraint formulation. The proposed strategy therefore represents the best compromise between the different cost terms and the overall feasibility of the problem.

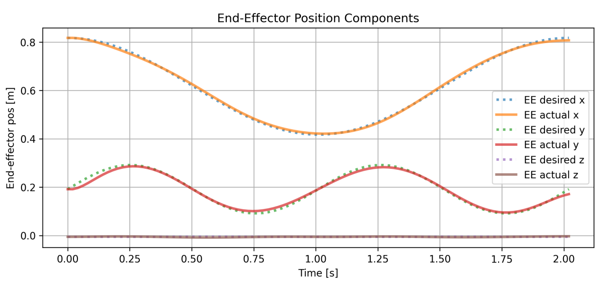

Figure 16: End effector position components (Q3).

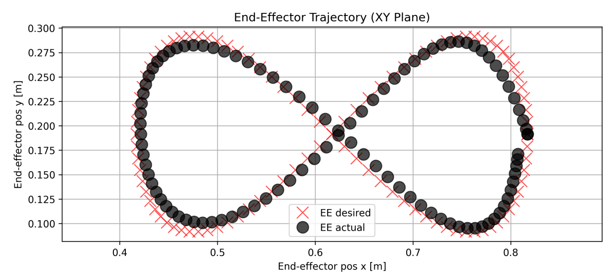

Figure 17: End effector trajectory xy (Q3).

The primary advantage of the trajectory formulation is the simplification of the OCP, obtained by fixing the variables S and W. This choice leads to a substantial reduction in computational complexity, as removing S and W from the set of decision variables makes the optimization problem significantly easier to solve.

A second advantage is a slight improvement in the execution of the cycling task.

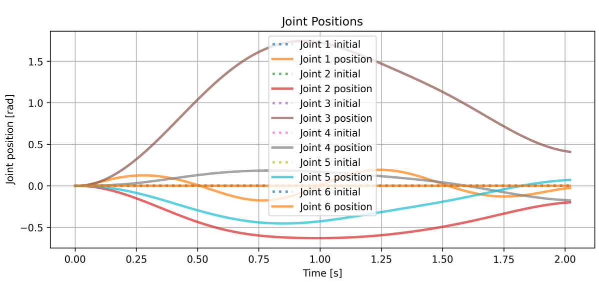

Figure 18: Joint position (Q3).

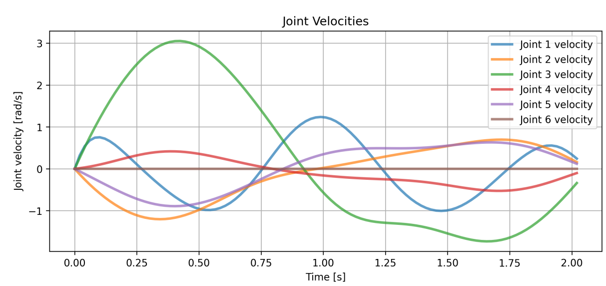

Figure 19: Joint velocities (Q3).

4 Path & Trajectory Tracking Min Time

To minimize the tracking time, we need to include the time step d_t as an optimizable variable.

4.1 Path Tracking Min Time

OCP Formulation:

$$\min_{X,U,S,W,d_t} \quad \sum_{k=0}^{N-1} \left(w_v \|\dot{q}_k\|^2 + w_a \|u_k\|^2 + w_w \|w_k\|^2 \right) d_t \; + \; w_{\text{time}} d_t^2 \qquad + \; w_{\text{final}} \, \|x(N) - x(0)\|^2$$

$$\begin{aligned}
\text{s.t.} \quad & q_{k+1} = q_k + d_t\,\dot{q}_k, \qquad \dot{q}_{k+1} = \dot{q}_k + d_t\,u_k, && \forall k \in [0, N-1],\\
& q_{\min} \le q_k \le q_{\max}, \qquad \dot{q}_{\min} \le \dot{q}_k \le \dot{q}_{\max}, && \forall k \in [1, N],\\
& \tau_{\min} \le \tau_k \le \tau_{\max}, && \forall k \in [1, N-1],\\
& d_{t\,\min} \le d_t \le d_{t\,\max}, && \forall k \in [1, N-1],\\
& s_{k+1} = s_k + d_t\,w_k, && \forall k \in [1, N-1],\\
& y(q_k) = p(s_k), && \forall k \in [1, N],\\
& x(0) = x_{\text{init}}, \qquad s(0) = 0, \qquad s(N) = 1.
\end{aligned}$$

The resulting solution is faster than in the previous case. In the earlier cycling path–tracking formulation, the time step was $d_t = 0.02\,\text{s}$, which over 100 steps corresponds to a total duration of 2 s. In the new formulation, the optimal time step is 1.19×10^{-2} s, resulting in a total duration of 1.19 s for the same number of steps.

The downside of this solution is that it places a strong emphasis on time minimization $w_{time} = 10^{-1}$ while the other weights are very lower $\sim 10^{-6}$. As a consequence, the relative importance of the other objectives is reduced. This results in a noticeable mismatch between the initial and terminal states, both in joint configuration and joint velocity, indicating a degradation of cycling-path tracking.

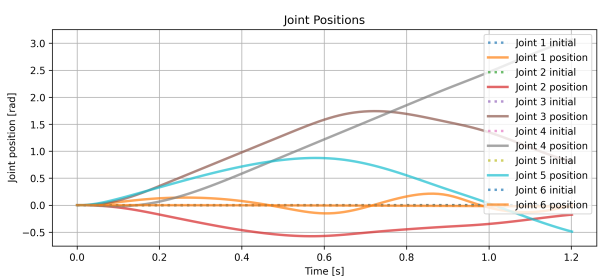

Figure 20: Joint position (Q4.1)

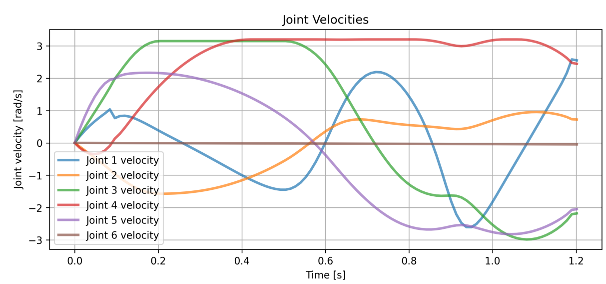

Figure 21: Joint velocity (Q4.1)

As shown in the joint velocity diagram, not only is the terminal target degraded, but the running objective is also affected. Both the joint velocity, joint torque and path velocity minimization goals appear to be partially neglected, as these quantities reach higher values compared to the original cycling path–tracking experiment. Particularly for the joint velocity where it reach its limits, as evidenced by the flat regions in the graphs (Figure 21);

Since this drowback are very significant and also dengerous, particularly the velocity and torques, the OCP need a rebalance of the weights in order to get a more resonable solution.

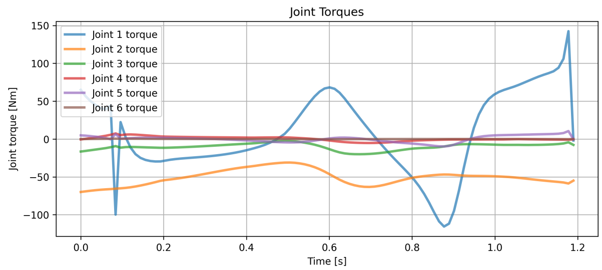

Figure 22: Joint torques (Q4.1)

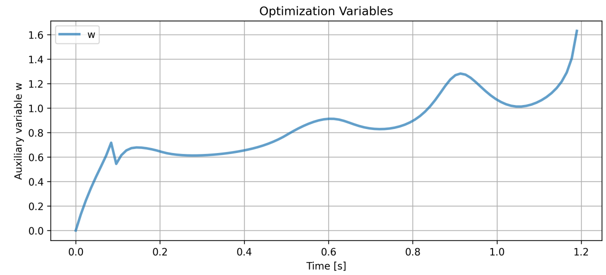

Figure 23: Path velocity w_k (Q4.1)

Note: In the initial formulation we implemented the time–step cost as a *terminal cost* rather than as a running cost. This means that the time penalty was added only once, whereas in the running–cost formulation it is added at every step (e.g., 100 times for a horizon of 100 steps). This change is equivalent to increasing w_{time} by a factor of 100. Indeed, in the initial running–cost implementation the optimizer did not reduce d_t; on the contrary, it tended to *increase* it, because its contribution to the total cost was much smaller than that of the other objectives.

4.2 Trajectory Tracking Min Time

OCP Formulation:

$$\min_{X,U} \quad \sum_{k=0}^{N-1} \left(w_v \|\dot{q}_k\|^2 + w_a \|u_k\|^2 + w_p \|y_q - p_s\|^2 \right) d_t \; + \; w_{\text{time}} d_t^2 \qquad + w_p \|y(q_N) - p(s_N)\|^2 + w_{\text{final}} \|x(N) - x(0)\|^2$$

$$\begin{aligned}
\text{s.t.} \quad & q_{k+1} = q_k + d_t\, \dot{q}_k, \quad \dot{q}_{k+1} = \dot{q}_k + d_t\, u_k, && \forall k \in [0, N-1] \\
& q_{\min} \le q_k \le q_{\max}, \quad \dot{q}_{\min} \le \dot{q}_k \le \dot{q}_{\max}, && \forall k \in [1, N] \\
& \tau_{\min} \le \tau_k \le \tau_{\max}, && \forall k \in [1, N-1] \\
& s_{k+1} = s_k + 1/N, && \forall k \in [1, N-1] \\
& x(0) = x_{\text{init}} \quad s(0) = 0 \quad s(N) = 1
\end{aligned}$$

In the trajectory-tracking implementation, this imbalance in the weighting is particularly evident. Indeed, the optimizer is able to reduce d_t down to 10^{-3} s (i.e., the lower bound), but it gives almost no importance to the tracking objective. As a consequence, the resulting trajectory is far from satisfactory: the system does not follow the desired path at all.

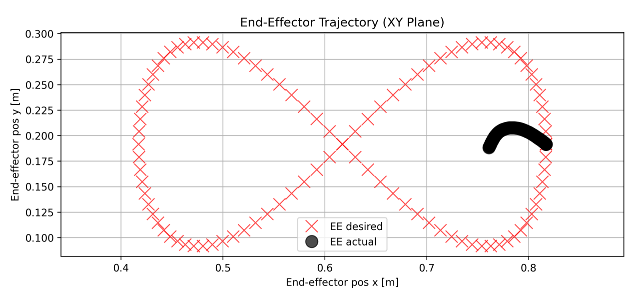

Figure 24: End effector trajectory xy (Q4.2)

We attempted to increase the weight associated with the trajectory-tracking term, to decrease the weight of the time-minimization term, and more generally to adjust the cost-function weights. However, none of these modifications produced a significant improvement in the resulting behavior.

Our hypothesis is that, since each running-cost term is multiplied by the factor d_t, the optimizer exhibits an unintended behavior: it systematically attempts to minimize d_t as much as possible in order to reduce all running-cost terms simultaneously. This results in an artificial and undesired optimization bias.