# Model Predictive Control for robots adapting their task space motion online

Nicolas Torres Alberto, Antun Skuric, Lucas Joseph, Vincent Padois, David Daney

# Model Predictive Control for robots adapting their task space motion online

Nicolas Torres Alberto[1,2], Antun Skuric[2], Lucas Joseph[2], Vincent Padois[2] and David Daney[2]

***Abstract*— Robots require the ability to autonomously and continuously react to unexpected online changes in the task definition and in the environment, especially those cohabited with humans. To react to these changes, the task, from the current state up to the finish, must instantly be re-considered. This implies a prohibitive re-computation cost.**

**This paper proposes a modular control architecture based on Model Predictive Control, that offers a good compromise between optimally achieving the task and the required computation time, by only re-considering the near future. This framework offers a generic way to formulate task-related objectives and constraints that dissociates the planning from the execution, which depends mainly on the robot dynamics. The proposal exploits a linear formalization of the MPC in SE(3) to implement this architecture in a high-frequency closed-loop controller, achieving task re-planning at the control rate.**

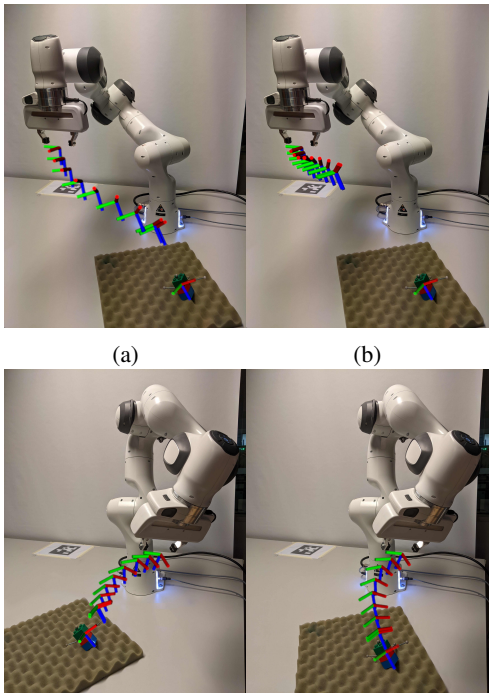
**The pertinence of the proposed control architecture is demonstrated using experiments with the Franka Emika robots in scenarios where the task to be achieved is modified on the fly.**

## I. Introduction

Collaborative robotics has increasingly gained interest over the last decades. From an application point of view, it crystallises early expectations [1], [2] around the improvement of human working conditions [3], [4]. From a research stand point, it raises several essential questions, for example regarding the prediction of human motion [5] or the underlying cognitive principles behind the concept of collaboration [6]. Among these research questions, the more general one of controlling robots in dynamic environments[1] is a recurring yet unsolved problem. A large amount of literature has been dedicated to this problem but the most recent works in the domain of collaborative robotics have focused on human avoidance [7], [8]. Another essential feature when considering dynamic environments is the general ability to autonomously and continuously adapt to unplanned task updates. The work in this paper proposes a control approach to address this problem.

Assuming there exist means to update online the state of the robot with respect to its environment and its task goal (i.e. perception), the most obvious way to endow robots with adaptation capabilities is based on online motion re-planning towards the goal. Actually, even though finding a feasible path to the target and updating it continuously is mandatory to get a chance to realize the task at some point, re-planning the temporal aspects of the motion from start to finish is actually debatable. Indeed, the whole planned trajectory is unlikely to be achieved when facing unplanned changes in the task. Moreover, despite the advances in the domain of motion planning [9] and optimal trajectory computation [10], online re-planing of both the path and the time profile over the whole motion remains an intractable problem. It is all the more true when the planning approach includes the computation of a control input trajectory. Re-planning over a receding horizon appears as a good alternative [11] to maintain the control problem computationally feasible online while still aiming at an optimal achievement of the task. This is a compromise solution between approaches where an optimal trajectory is computed once offline and approaches where adaptation to changes in the environment are performed without full adaptation of that trajectory [12].

A second implication of the evolution of robots in dynamic environments is related to the impossibility to verify a priori that all constraints, either task-related or linked to the intrinsic



(a) (b) (c) (d)

Fig. 1: A linear Model Predictive Controller is used to compute a position and orientation trajectory towards a target (plastic brick) in a receding horizon. (a) and (b) highlight the effects of changing the task space constraints on the fly (in this case max velocities) on the covered distance for one horizon. (c) and (d) illustrate the continuous adaptation of the trajectory when the target location is modified online.

[1]Stellantis, Centre Technique Vélizy, France
nicolas.torres@stellantis.fr
[2]Inria, AUCTUS Team, Talence, France
antun.skuric@inria.fr, lucas.joseph@inria.fr, vincent.padois@inria.fr, david.daney@inria.fr

This work is funded by the ANRT and Stellantis Group and performed within the framework of the OpenLab AI on the one hand and by BPI France through the LICHIE project in collaboration with Airbus on the other hand.

[1]as opposed to static ones where robots are physically separated from anything that could provoke a change in either the task or the surrounding objects during execution

robot capabilities, will be met. In other words, on-line re-planning definitely provides some level of guarantee that the generated motion will not lead the robot to violate constraints. However, it is unlikely to be the case if unplanned events occur while executing the motion. This leads to situations where emergency stops are the only way to prevent an accident. To avoid these control exceptions, controllers must be formulated as optimization problems where constraints are explicitly considered as such in the resolution [13].

Model Predictive Control (MPC) appears as a good candidate as it features both the ability to optimally re-plan the motion given the current state of the system (i.e. in closed-loop) over a receding horizon while accounting for constraints. Beyond local optimality, reasoning over a horizon provides an implicit way to render the set of considered constraints compatible which is much more likely to succeed with respect to approaches attempting to perform this compatibilisation once off-line [14], [15], [16].

Assuming that MPC can keep up with the robot control rate, one can rely on a control architecture where, given an operational space target, a joint velocity or torque control input horizon is re-computed at each control time step and fed to the robot. This is what is proposed in [17] where MPC is performed through Differential Dynamic Programming which can be solved efficiently [18] at each control step over a receding horizon. This approach couples two underlying sub-problems and solves them together: 1) the computation of an operational space trajectory towards the target and 2) the resolution of the operational to joint space mapping. Even though constraints are implemented in the cost function and cannot be formally guaranteed, this approach presents the advantage of considering joint and actuation constraints directly at the MPC problem level. As a consequence, except for the joint torque servoing performed at the actuation level, no inner control loop is required.

Yet, alternative approaches can be considered where an MPC is solved at the operational space level to incrementally generate a feasible task-space trajectory given some task-related constraints [19], [7]. The output of the MPC is then fed to a joint level controller which can, for example, solve for the joint torques at each control instant using a constrained optimization approach [13]. This approach does not allow to account for the joint level limits of the robot at the MPC level. However, it provides a modular architecture closing the loop on the current operational state of the robot, independently from the nature of the controller provided with a given robot. The later approach is retained in this work.

The formulation of both control levels, task-space MPC and constrained inverse dynamics, is described in Section II. Section III provides a description of the experiments performed with a 7 DoF manipulator to validate the pertinence of the proposed control approach. These results, exemplified in Fig. 1, illustrate several interesting characteristics of the approach ranging from: 1) constraints compliance; 2) the ability to manage instantaneously switching targets; 3) the ability to continuously adapt the trajectory towards a continuously moving target; 4) the effect of the modification of velocity constraints in the MPC on the generated motion horizon at each control step. An accompanying video offers a compact summary of all these results. Finally, Section IV discusses several aspects of the obtained results and potential improvements. Section V concludes the paper by summarizing the contributions and describing potential future research work on this topic.

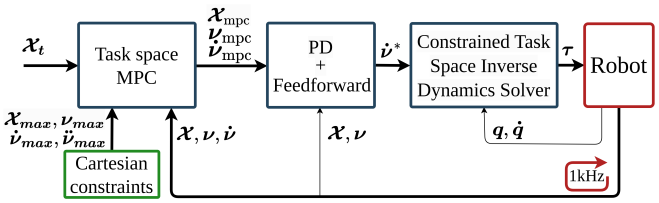



Fig. 2: The proposed modular control architecture is based on a cascade loop composed of: 1) a task-space Model Predictive Controller (MPC) for online re-planning; 2) a constrained Task Space Inverse Dynamics solver to follow the planned trajectory.

## II. Control Architecture

The control architecture drafted in Section I and summarily described on Figure 2 consists of two main components. The first one is a linear MPC, formulated at the operational space level. It generates an optimal trajectory in $\mathbb{SE}(3)$ over a receding horizon, given a reference target and constraints on the admissible motion, both provided on the fly. These constraints can be both related to the real joint space motion capabilities of the robot and to restrictions relative to the performed task. The second one computes the robot control input leading to the execution of the planned trajectory while respecting the joint-level constraints using an inverse dynamics solver. These two components are described hereafter. In all, the architecture enables the robot to re-plan a trajectory at the control frequency of the robot while accounting for pose constraints on the end-effector up to the jerk order.

### A. MPC-based motion re-planning over a receding horizon

Assuming jerk to be a control input at the task level, the evolution of the state – pose, twist and acceleration – of the end-effector of a robot can be described using a discrete time linear system (indexed with n):

$$\boldsymbol{x}_{n+1} = \boldsymbol{A}\boldsymbol{x}_n + \boldsymbol{B}\boldsymbol{u}_n \tag{1}$$

where $\boldsymbol{A}$ and $\boldsymbol{B}$ represent its state and input matrices whereas $\boldsymbol{x}$ and $\boldsymbol{u}$ are the state and input vectors.

Discretizing time in h time steps of duration T starting at time $t_n$ (total duration hT) yields the propagated state space equation:

$$\boldsymbol{X}_{0\ldots\text{h}} = \begin{bmatrix} \boldsymbol{x}_{k|n} \\ \boldsymbol{x}_{k+1|n} \\ \vdots \\ \boldsymbol{x}_{k+\text{h}|n} \end{bmatrix} \quad \boldsymbol{U}_{0\ldots\text{h}-1} = \begin{bmatrix} \boldsymbol{u}_{k|n} \\ \boldsymbol{u}_{k+1|n} \\ \vdots \\ \boldsymbol{u}_{k+\text{h}-1|n} \end{bmatrix} \tag{2}$$

$$\overline{\boldsymbol{X}} = \boldsymbol{X}_{1\ldots\text{h}} \quad \underline{\boldsymbol{X}} = \boldsymbol{X}_{0\ldots\text{h}-1} \quad \underline{\boldsymbol{U}} = \boldsymbol{U}_{0\ldots\text{h}-1}$$

where the discretized time steps inside the horizon are defined as $t_k = t_n + k\text{T}$ for $k = 0, 1, \ldots, \text{h}$. This allows extending the definition in (1) for the horizon:

$$\overline{\boldsymbol{X}} = \boldsymbol{A}'\underline{\boldsymbol{X}} + \boldsymbol{B}'\underline{\boldsymbol{U}} \tag{3}$$

where $\boldsymbol{A}', \boldsymbol{B}'$ are matrices constructed from on $\boldsymbol{A}, \boldsymbol{B}$ via the recursive propagation of (1) over the time horizon. $\overline{\boldsymbol{X}}$ and $\underline{\boldsymbol{U}}$ respectively correspond to the future states and control inputs of the system.

From these propagated linear dynamics, one can formulate a linear optimal control problem over a receding horizon as a Quadratic Program (QP) with the following cost function:

$$\min_{\boldsymbol{u}} \left\| \overline{\boldsymbol{X}}_t - \overline{\boldsymbol{X}}(\underline{\boldsymbol{U}}) \right\|_{\boldsymbol{P}}^2 + \left\| \underline{\boldsymbol{U}} \right\|_{\boldsymbol{R}}^2 \tag{4}$$

subject to equality (3) and polyhedral constraints:

$$\boldsymbol{x}_m \leq \boldsymbol{C_x} \boldsymbol{x}_k \leq \boldsymbol{x}_M \qquad \boldsymbol{u}_m \leq \boldsymbol{C_u} \boldsymbol{u}_k \leq \boldsymbol{u}_M \tag{5}$$

where $\overline{\boldsymbol{X}}_t$ is a vector of the target state $\boldsymbol{x}_t$ over the horizon. This is an important detail and one of the original features of the proposed approach: no pre-interpolation or a priori trajectory generation is required. The goal at each discrete step of the horizon is to reach the final target with a null velocity and a greater importance is given to the last step of the horizon to favour convergence towards the target. The resulting motion is solely shaped by the considered constraints on the admissible task space motions. This implicitly yields a time optimal trajectory over the horizon. $\boldsymbol{P}, \boldsymbol{R}$ respectively represent symmetric positive definite weighting matrices for the input and state throughout the horizon. Finally, $\boldsymbol{C_x}, \boldsymbol{C_u}$ allow formulating linear constraints with respect to the state and input, while $\boldsymbol{x}_m, \boldsymbol{x}_M$ and $\boldsymbol{u}_m, \boldsymbol{u}_M$ respectively designate the state and input bounds. This formulation at the jerk level can be almost equivalently reformulated at the acceleration level through the inclusion of a discretized constraint on the derivative of the acceleration. This is the choice made in this work and throughout the remainder of this paper.

Equation (4) under constraints (3) and (5) constitutes the general linear MPC formulation as a QP. It progressively provides a trajectory to track for the end-effector as the optimization window evolves. The linearity of this MPC formulation is a mandatory feature for the corresponding optimization problem to be solved at each control time step. However, writing this problem in a linear fashion, given that the Cartesian pose of the end-effector $\mathcal{X}$ belongs in $\mathbb{SE}(3)$ is far from simple. The next sub-section describes the proposed formulation.

### *B. Pose integration over a horizon*

To understand how (1) can express the pose dynamics in linear form, consider the first-order integration of a pose in $\mathbb{SE}(3)$ subject to an infinitesimal body twist increment for a single time step [20]:

$$\mathcal{X}_{k+1} = \mathcal{X}_k e^{\boldsymbol{\nu}_k \mathrm{T}} \tag{6}$$

where $e^{\boldsymbol{\nu}_k T}$ is the homogeneous matrix equivalent to the application of $\boldsymbol{\nu}_k$ for a duration of T. It is computed using the matrix exponential, that surjectively maps the Lie algebra $\mathfrak{se}(3)$ onto the Lie group $\mathbb{SE}(3)$. Its inverse operation is known as the logarithmic map. This relation can be exploited to represent $\mathcal{X}$ in a vector form. In fact, given some initial pose $\mathcal{X}_0$, there exists a function $\boldsymbol{\psi}$ (referred to as the *lift function*) that maps $\mathcal{X}_k$ to a vector $\boldsymbol{\xi}_k$ (further explained in [21] [20] [22]):

$$\begin{aligned} \boldsymbol{\psi} &: \mathbb{SE}(3) \to \mathfrak{se}(3) & \log(\mathcal{X}_0^{-1}\mathcal{X}_k) &\to \boldsymbol{\xi}_k \\ \boldsymbol{\psi}^{-1} &: \mathfrak{se}(3) \to \mathbb{SE}(3) & \mathcal{X}_0\, e^{\boldsymbol{\xi}_k} &\to \mathcal{X}_k \end{aligned} \tag{7}$$

The lift function maps elements from the $\mathbb{SE}(3)$ manifold $\mathcal{M}$ to its tangent space at $\mathcal{X}_0$: $\boldsymbol{\psi} : \mathcal{M} \to \mathcal{T}_{\mathcal{X}_0}\mathcal{M}$. Notice that the pose delta $\Delta\mathcal{X}_k \in \mathbb{SE}(3)$ (or equivalent homogeneous matrix) needed to move from $\mathcal{X}_0$ to $\mathcal{X}_k$ can be computed with $\Delta\mathcal{X}_k = e^{\boldsymbol{\psi}(\mathcal{X}_k)}$. The function $\boldsymbol{\psi}^{-1}$ is referred to as the *retract function*.

Using the lift and retract functions, the geodesic path between $\mathcal{X}_0$ and $\mathcal{X}_t$ can be interpolated for any pose in between with a parameter $\alpha \in [0, 1]$:

$$\mathcal{X}(\alpha) = \boldsymbol{\psi}^{-1}(\alpha\, \boldsymbol{\psi}(\mathcal{X}_t)) = \mathcal{X}_0 e^{\alpha\, \boldsymbol{\psi}(\mathcal{X}_t)} \tag{8}$$

It is straightforward to notice that $\mathcal{X}(0) = \mathcal{X}_0$ and $\mathcal{X}(1) = \mathcal{X}_t$.

This geodesic path is obtained by minimizing the *Log-Euclidean* ([23], Table 1) distance metric between the lifted pose $\boldsymbol{\xi}_k$ and the lifted target pose $\boldsymbol{\xi}_t = \boldsymbol{\psi}(\mathcal{X}_t)$:

$$e_k = \left\| \boldsymbol{\xi}_t - \boldsymbol{\xi}_k \right\|_2^2 \tag{9}$$

The target related cost in (4) corresponds to the discretized version of (9) over the MPC horizon. Hence, it leads to minimizing the geodesic distance towards the target pose over the horizon.

Furthermore, in order to link (6) with (1) (with linear MPC in mind), it needs to be linearized. Computing $\boldsymbol{\psi}(\mathcal{X}_{k+1})$ yields:

$$\boldsymbol{\psi}(\mathcal{X}_{k+1}) = \log(\Delta\mathcal{X}_k e^{\mathrm{T}\boldsymbol{\nu}_k}) \tag{10}$$

In fact, this expression has a first-order approximation in its Lie algebra (see Equations 68-74 in [20])

$$\boldsymbol{\xi}_{k+1} = \log(e^{\boldsymbol{\xi}_k} e^{\mathrm{T}\boldsymbol{\nu}_k}) \approx \boldsymbol{\xi}_k + \mathrm{T}\mathrm{J}_{\mathrm{R}}^{\log}(\boldsymbol{\xi}_k)\boldsymbol{\nu}_k \tag{11}$$

where $\mathrm{J}_{\mathrm{R}}^{\log}(\boldsymbol{\xi}_k)$ is the right-trivialized Jacobian of the logarithmic map, evaluated at $\boldsymbol{\xi}_k$. A closed-form expression can be found in [24] [20] and is implemented in [25]. It relates the additive increments in $\mathcal{T}_0\mathcal{M}$ (the tangent map at the origin) to the right-multiplied increments in $\mathbb{SE}(3)$.

Equation (11) offers the missing piece to have a concrete definition for (1) when considering acceleration $\boldsymbol{a}_k$ as the control input:

$$\boldsymbol{A}_k = \begin{bmatrix} \mathbf{I}_6 & \mathrm{T}.\mathrm{J}_{\mathrm{R}}^{\log}(\boldsymbol{\xi}_k) \\ \mathbf{0}_6 & \mathbf{I}_6 \end{bmatrix} \quad \boldsymbol{B}_k = \begin{bmatrix} \frac{\mathrm{T}^2}{2}\mathrm{J}_{\mathrm{R}}^{\log}(\boldsymbol{\xi}_k) \\ \mathrm{T}.\mathbf{I}_6 \end{bmatrix} \tag{12}$$

for which the discretized state and input vectors at each $k$, and the target state $\boldsymbol{x}_t$ are:

$$\boldsymbol{x}_k = \begin{bmatrix} \boldsymbol{\xi}_k^T & \boldsymbol{\nu}_k^T \end{bmatrix}^T \quad \boldsymbol{x}_t = \begin{bmatrix} \boldsymbol{\xi}_t^T & \mathbf{0}_6^T \end{bmatrix}^T \quad \boldsymbol{u}_k = \boldsymbol{a}_k \tag{13}$$

With the MPC being formulated in $\mathfrak{se}(3)$, the lift operator $\boldsymbol{\psi}$ is used to map the input state and the desired pose into $\mathfrak{se}(3)$ while the retract operator $\boldsymbol{\psi}^{-1}$ is used to map the computed trajectory to $\mathbb{SE}(3)$. It is also important to note that the duration of a step T in the horizon is generally longer than the robot control/re-planning period and the output of the MPC is thus interpolated using (8).

### *C. Constrained task space inverse dynamics solver*

Given the interpolated first step of the trajectory computed by the MPC, task space inverse dynamics has to be performed to compute the input joint torque for the robot. Feeding the inverse dynamics solver with the desired acceleration computed by the MPC may sound tempting. Nevertheless, the feedback provided by the closed-loop MPC is, partly due to the interpolated output, not efficient enough to properly reject tracking errors related to the inaccuracies in the model of the robot. These inaccuracies are mostly related to dry friction at the joint level, imperfectly rejected by the lower-level torque control loop in most robots. As a consequence, a PD controller including a feed-forward term in acceleration is used to compute a corrected desired acceleration :

$$\dot{\boldsymbol{\nu}}^* = \boldsymbol{K}_p \log(\mathcal{X}^{-1}\mathcal{X}_{\mathrm{mpc}}) + \boldsymbol{K}_d(\boldsymbol{\nu}_{\mathrm{mpc}} - \boldsymbol{\nu}) + \dot{\boldsymbol{\nu}}_{\mathrm{mpc}} \tag{14}$$

where $\mathcal{X}_{\mathrm{mpc}}, \boldsymbol{\nu}_{\mathrm{mpc}}, \dot{\boldsymbol{\nu}}_{\mathrm{mpc}}$ are respectively the interpolated desired pose, twist and acceleration outputted by the MPC; $\mathcal{X}, \boldsymbol{\nu}$ are the measured pose and twist of the robot and $K_p$ and $K_d$ are positive proportional and derivative gains.

Given this control acceleration, one can solve task-space inverse dynamics under constraints through a quadratic programming

formulation, similar to the one in [26]. The optimal joint torque $\boldsymbol{\tau}^{\text{opt}}$ is computed at each control step as:

$$\boldsymbol{\tau}^{\text{opt}} = \arg\min_{\boldsymbol{\tau}} \left\| \dot{\boldsymbol{\nu}}^* - \dot{\boldsymbol{\nu}}(\boldsymbol{\tau}) \right\|_2^2 + w_{\text{reg}} \left\| \boldsymbol{f}_{\text{reg}}(\boldsymbol{\tau}) \right\|_2^2 \tag{15}$$

$$\text{s.t.} \quad \boldsymbol{\tau}_m \leq \boldsymbol{\tau} \leq \boldsymbol{\tau}_M, \quad \boldsymbol{q} \in \mathcal{Q}, \quad \dot{\boldsymbol{q}} \in \dot{\mathcal{Q}} \tag{16}$$

$$\dot{\boldsymbol{\nu}}(\boldsymbol{\tau}) = \boldsymbol{J}(\boldsymbol{q})\boldsymbol{M}(\boldsymbol{q})^{-1}\boldsymbol{\tau} + \boldsymbol{b}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \tag{17}$$

$$\boldsymbol{f}_{\text{reg}}(\boldsymbol{\tau}) = \boldsymbol{\tau}^* - \boldsymbol{\tau} \tag{18}$$

The cost function is designed to find the optimal torque $\boldsymbol{\tau}^{\text{opt}}$ that minimizes the difference between the acceleration resulting from the applied torque through the system's dynamics in (17) and the control acceleration $\dot{\boldsymbol{\nu}}^*$ computed to track the desired trajectory. This optimization accounts for joint level constraints on the minimum and maximum torque $\boldsymbol{\tau}_m$ and $\boldsymbol{\tau}_M$ in (16). It also considers the admissible joint space configuration and velocity $\mathcal{Q}$ and $\dot{\mathcal{Q}}$, expressed at the current robot state $\{\boldsymbol{q}, \dot{\boldsymbol{q}}\}$ as a function of $\boldsymbol{\tau}$ using a Taylor expansion. In (17), $\boldsymbol{J}$ and $\boldsymbol{M}$ are respectively the task space Jacobian matrix associated to the end-effector and the joint space inertia matrix. Finally, $\boldsymbol{b}$ is a bias acceleration term which also incorporates the effect of gravity and the Coriolis/Centrifugal forces acting on the system and expressed at the end-effector level.

A regularization term is added to the cost function with a small weight ($w_{\text{reg}} \ll 1$) to ensure the uniqueness of the computed solution in case of a redundant robot while minimally affecting the main task of the robot.Moreover, $\boldsymbol{\tau}^*$ can be chosen in different ways mostly depending on the targeted application. Here, $\boldsymbol{\tau}^*$ is chosen to ensure convergence of the robot configuration to the neutral configuration $\boldsymbol{q}_0$. The servoing of this configuration can be expressed at the joint torque level using a proportional-derivative controller such that: $\boldsymbol{\tau}^* = \boldsymbol{K_q}(\boldsymbol{q}_0 - \boldsymbol{q}) - \boldsymbol{K_{\dot{q}}}\dot{\boldsymbol{q}}$ with $\boldsymbol{K_q}$ and $\boldsymbol{K_{\dot{q}}}$ positive proportional and derivative gains.

## III. Experiment description and results

The objective of the experiment is to validate the online adaptation of the robot to arbitrary target pose modifications. A three phases experiment is proposed, as illustrated in Fig. 3 and detailed in the following Section.

### A. Experiment description

The robot is first requested to pick-up a plastic brick placed in the workspace over an irregular surface (to induce non-trivial orientations). The pose of the brick is not known a priori and is detected via a vision system. Given this target pose information $\mathcal{X}_t$, the robot starts its motion towards the brick using the control scheme described in Fig. 2 and in Section II. Once the object is reached, the robot grasps it and releases it after a few seconds. Then, the robot goes back to its neutral position and the brick is placed at a new pose. After a few seconds, the robot starts its second motion, towards this new goal. While in motion, the brick is displaced by a human and the robots adapts its trajectory online to reach it. In the third phase, the human grasps the brick, moves it randomly around and the robot tracks it in real time.

The set-up is illustrated on Fig. 3. Perception of the plastic brick is provided by an Optitrack camera array. The experiment is carried out on a torque-controlled 7-DoF Franka Emika Panda serial robot. The control loop described in Fig. 2 runs at 1kHz. In order to keep up with the control rate, the MPC uses a 750ms control horizon (h = 5, T = 150ms), which yields an average computation time of 0.1ms (max 0.2ms) using qpOASES[27]. The same QP solver is used to solve the constrained task space inverse dynamics. The values of gains and weights for this inner QP controller are

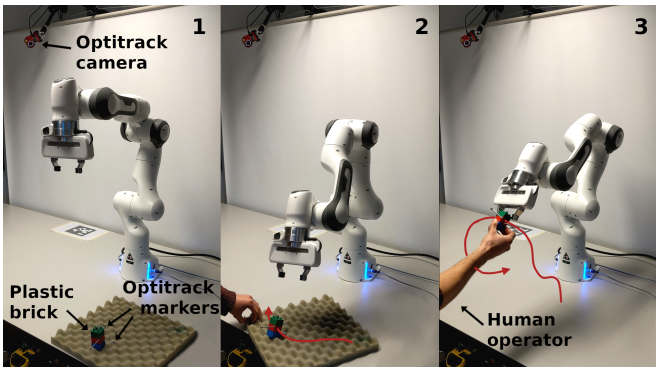



Fig. 3: The experimental setup is composed of a torque-controlled 7-DoF Franka Emika Panda serial robot and an Optitrack camera array. Reflective markers are placed on a plastic brick. The camera array is used to detect the position and orientation of the plastic brick relatively to the robot base.

summarized in Table I. They are not provided in details for the MPC controller for the sake of compactness. It is just recalled here that a larger cost is assigned to the terminal step of the horizon to favour convergence towards the target. Details can be found in the source code[2].

| Gains / Weights | $\boldsymbol{K}_p$ | $\boldsymbol{K}_d$ | $\boldsymbol{K_q}$ | $\boldsymbol{K_{\dot{q}}}$ | $w_{\text{reg}}$ |
|---|---|---|---|---|---|
| Value | 150 | 24.5 | 100 | 10 | $10^{-5}$ |
| Units | $\text{s}^{-2}$ | $\text{s}^{-1}$ | $\text{N.m.s}^{-2}$ | $\text{N.m.s}^{-1}$ | $\text{N}^{-2}\text{.m}^2$ |

TABLE I: Values of the gains and weights retained for the described experiments.

Details on the motion resulting from one run of this experiment are provided in Fig.4 for the first of the three described phases for space reasons. The two other experiments can be observed in the video attached to this paper.

### B. Analysis

Fig. 4a shows the error between the current and the target poses. The discontinuity seen at the beginning corresponds to the start of the experiment sequence, where the target pose switches from the "home pose" to the plastic brick pose. This highlights the convergence towards the requested pose without any a priori knowledge on the trajectory to follow. Note that this is different from the resulting tracking error of following the trajectory planned by the MPC with the constrained task-space inverse dynamics solver. This error, shown in Fig. 4b), is bounded to 19.7mm with a mean value of 4mm ± 1 for the linear component of the motion and $5.18 \times 10^{-2}$rad with a mean value of $2.5 \times 10^{-3}$rad ± $5.5 \times 10^{-3}$ for the angular component. The final error is not exactly zero (2.1mm and $10.4 \times 10^{-2}$rad) due to the relatively low gains of the PD controller and the absence of an integral term to reject disturbances due to dry friction at the joint level. Yet, the robot is able to grasp the targeted object at the end of the motion as can be seen in the accompanying video.

Fig. 4c shows both the planned (by the MPC) and the actual velocities; as opposed to the acceleration and jerk curves (respectively in Fig. 4d and 4e) that only show the planned trajectories. The first thing to notice is that the constraints on the planned trajectory are met by the MPC, as can be seen on the velocity

[2]https://gitlab.inria.fr/auctus-team/people/nicolas-torres/lmpcpoly-ws/

and jerk curves. It is also interesting to notice that the obtained linear velocity profile is similar, to some extent, to trapezoidal acceleration profiles, maximizing the use of some of the prescribed Cartesian limits shown in Table II. This confirms the near-optimal character of the generated trajectory.

| | $\boldsymbol{\nu}_{\max}$ | $\dot{\boldsymbol{\nu}}_{\max}$ | $\ddot{\boldsymbol{\nu}}_{\max}$ |
|---|---|---|---|
| linear | 0.2 (m/s) | 2 (m/s$^2$) | 1000 (m/s$^3$) |
| angular | 0.8 (rad/s) | 5 (rad/s$^2$) | 3000 (rad/s$^3$) |

$$-\boldsymbol{\nu}_{\max} \leq \boldsymbol{\nu} \leq \boldsymbol{\nu}_{\max}, \quad -\dot{\boldsymbol{\nu}}_{\max} \leq \dot{\boldsymbol{\nu}} \leq \dot{\boldsymbol{\nu}}_{\max}, \quad -\ddot{\boldsymbol{\nu}}_{\max} \leq \ddot{\boldsymbol{\nu}} \leq \ddot{\boldsymbol{\nu}}_{\max}$$

TABLE II: Cartesian constraints for $\boldsymbol{\nu}, \dot{\boldsymbol{\nu}}, \ddot{\boldsymbol{\nu}}$ in the MPC configuration during the experiment.

Finally, one can notice that the robot's end-effector velocity surpasses the established limits. This is discussed in Section IV.

## IV. Discussions

The modular nature of the proposed architecture offers the potential to generically consider different types of task inputs. It can for example be easily adapted to use a discretized pose path, which could come from an obstacle avoidance solver. Hence, the same architecture can accommodate a complex pathing solution while retaining its core structure. This architecture can also accommodate different types of task space to joint space solvers. This is an important feature as not all robots allow for joint torque control.

Moreover, the cost function of the MPC and the Cartesian constraints offer a generic way to design human-aware solutions considering the non-trivial cognitive aspects of human-robot collaboration. For example, representing the human motion capabilities as Cartesian constraints [28]. This is further enabled by decoupling the task-related definition and constraints from the joint-level controller. Indeed, some constraints are intrinsically task related and somewhat independent of the robot performing the task while others are strongly related to the whole robot motion. It is for example the case for safety-aware constraints [29] or the exploitation of the robot redundancy to perform secondary tasks without compromising performance. Examples of the later case are shown for safety purposes in [30], [31] and for situation awareness in [32].

Fig. 4c shows that the real end-effector's velocity loosely follows the planned velocity, going beyond the established limits in some cases. The measured velocity is being used as the initial state of the MPC at each time step, this causes the associated QP to become infeasible. This characteristic is intrinsic to any QP-based architecture and it can be considered both a problem and an advantage: for one, it allows easily detecting ill-conditioned initial states that could result in undefined behaviors (for example, to trigger an emergency stop mode); on the other hand, it demands that a proper strategy is put in-place to deal with this situation. For this article, the inner loop controller continues to follow the last horizon that was successfully computed until the MPC can compute a new one. This allows the execution to continue smoothly.

Nevertheless, this opens one fundamental question that naturally arises from this type of cascade control architectures: the compatibility of the cascading controllers. For instance, the MPC could plan a trajectory that respects the Cartesian constraints even if it is instantaneously unfeasible in a control step or may lead to an inevitable joint-level constraint violation in the near future [14]. To solve this problem, joint-level constraints [16] should be implicitly considered at the MPC level. This opens up one potential line of research on how to efficiently formulate the actual robot capacities as Cartesian constraints [33]. This type of constraints goes beyond classical box-like bounds and is a step towards more complex adaptation of the robot behaviour based on its motion capabilities.

## V. Conclusion

This paper proposes an efficient linear model predictive control approach that can deal with the online planning of SE(3) motion in task space. The main features of the proposed control approach lie in its ability to generate optimal Cartesian motion which dynamically accounts both for targets updated on-the-fly and evolving motion constraints. This receding horizon approach endows the robot with the ability to interactively adapt to task adaptation. In the context of collaborative robotics, using this controller, the safety of a human operator sharing its workspace with the robot could be accounted for through the sensor-based adaptation of maximum velocity constraints.

Future work will focus on extending linear constraints in the MPC for the Cartesian pose state, to limit the effective position and orientation space. This work also opens doors to potentially exploiting a more efficient expression of Cartesian constraints such as convex polytopes that consider the actual robot joint space capabilities in the receding horizon optimization stage. The reduction of the computation time required to solve the MPC problem is also an important matter. It will be pursued to refine the computation of the trajectory over the horizon and potentially improve the performance of the closed-loop.

## References


[1] B. J. Makinson, "Research and development prototype for machine augmentation of human strength and endurance: Hardiman i project," General Electric Company, Schenectady, NY, USA, Tech. Rep., 1971.

[2] J. E. Colgate, M. Peshkin, and S. H. Klostermeyer, "Intelligent assist devices in industrial applications: a review," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, 2003.

[3] P. Maurice, V. Padois, Y. Measson, and P. Bidaud, "Human-oriented design of collaborative robots," *International Journal of Industrial Ergonomics*, vol. 57, 2017.

[4] C. Schoose, A. Cuny-Guerrier, S. Caroly, L. Claudon, P. Wild, and A. Savescu, "Evolution of the biomechanical dimension of the professional gestures of grinders when using a collaborative robot," *International Journal of Occupational Safety and Ergonomics*, 2022.

[5] J. Mainprice, R. Hayne, and D. Berenson, "Predicting human reaching motion in collaborative tasks using inverse optimal control and iterative re-planning," in *IEEE International Conference on Robotics and Automation*, 2015.

[6] J. Y. Chen and M. J. Barnes, *Handbook Of Human Factors And Ergonomics*. John Wiley & Sons, Ltd, 2021, ch. Human-Robot Interaction.

[7] M. Eckhoff, R. J. Kirschner, E. Kern, S. Abdolshah, and S. Haddadin, "An MPC framework for planning safe and trustworthy robot motions," in *International Conference on Robotics and Automation*, 2022.

[8] K. Merckaert, B. Convens, C. ju Wu, A. Roncone, M. M. Nicotra, and B. Vanderborght, "Real-time motion control of robotic manipulators for safe human–robot coexistence," *Robotics and Computer-Integrated Manufacturing*, vol. 73, 2022.

[9] D. Coleman, I. Sucan, S. Chitta, and N. Correll, "Reducing the barrier to entry of complex robotic software: a moveit! case study," *Journal of Software Engineering for Robotics*, vol. 5, no. 1, May 2014.

[10] H. Pham and Q. C. Pham, "A new approach to time-optimal path parameterization based on reachability analysis," *IEEE Transactions on Robotics*, vol. 34, 6 2018.

[11] M. M. Ghazaei Ardakani, B. Olofsson, A. Robertsson, and R. Johansson, "Model predictive control for real-time point-to-point trajectory generation," *Transactions on Automation Science and Engineering*, vol. 16, no. 2, 2019.

[12] L. Joseph, J. K. Pickard, V. Padois, and D. Daney, "Online velocity constraint adaptation for safe and efficient human-robot workspace sharing," in *International Conference on Intelligent Robots and Systems*, Las Vegas, United States, Oct. 2020.

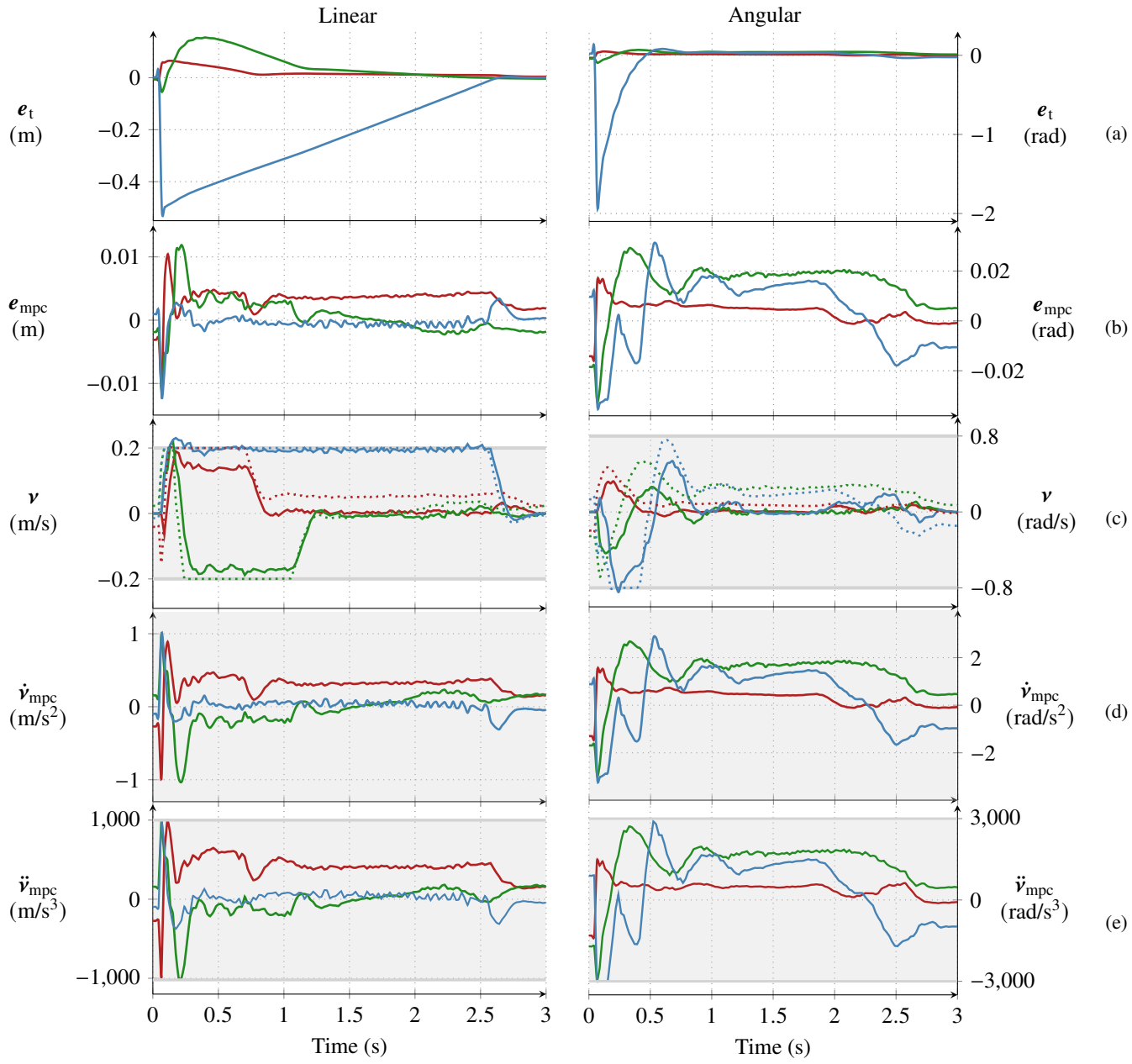



Fig. 4: Tracking results of driving the robot towards a target pose given on the fly. For each plot, the red, green and blue curves respectively correspond to the components in the x, y and z axis. For the velocity plots, the plain lines correspond to the actual robots velocity and the dotted lined to the one computed by the mpc ($\nu_{mpc}$). The shaded areas represent the bounds given to the MPC.

[13] M. Liu, Y. Tan, and V. Padois, "Generalized hierarchical control," *Autonomous Robots*, vol. 40, no. 1, 2015.

[14] S. Rubrecht, V. Padois, P. Bidaud, and M. De Broissia, "Constraints compliant control: constraints compatibility and the displaced configuration approach," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010.

[15] A. Meguenani, "Safe control of robotic manipulators in dynamic contexts," Ph.D. dissertation, Université Pierre et Marie Curie-Paris VI, 2017.

[16] A. Del Prete, "Joint position and velocity bounds in discrete-time acceleration/torque control of robot manipulators," *Robotics and Automation Letters*, vol. 3, no. 1, 2017.

[17] S. Kleff, A. Meduri, R. Budhiraja, N. Mansard, and L. Righetti, "High-Frequency Nonlinear Model Predictive Control of a Manipulator," in *IEEE International Conference on Robotics and Automation*, Xi'an, China, May 2021.

[18] C. Mastalli, R. Budhiraja, W. Merkt, G. Saurel, B. Hammoud, M. Naveau, J. Carpentier, L. Righetti, S. Vijayakumar, and N. Mansard, "Crocoddyl: An efficient and versatile framework for multi-contact optimal control," in *IEEE International Conference on Robotics and Automation*, 2020.

[19] M. Faroni, M. Beschi, and N. Pedrocchi, "An mpc framework for online motion planning in human-robot collaborative tasks," in *24th IEEE International Conference on Emerging Technologies and Factory Automation*, 2019.

[20] J. Solà, J. Deray, and D. Atchuthan, "A micro lie theory for state estimation in robotics," *CoRR*, vol. abs/1812.01537, 2018. [Online]. Available: http://arxiv.org/abs/1812.01537

[21] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration theory for fast and accurate visual-inertial navigation," *Transactions on Robotics*, 2015.

[22] N. Torres Alberto, L. Joseph, V. Padois, and D. Daney, "A linearization method based on Lie algebra for pose estimation in a time horizon," in *18th International Symposium on Advances in Robot Kinematics*, Bilbao, Spain, June 2022.

[23] S. Jayasumana, R. Hartley, M. Salzmann, H. Li, and M. Harandi, "Kernel methods on the riemannian manifold of symmetric positive definite matrices," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[24] T. D. Barfoot and P. T. Furgale, "Associating uncertainty with three-dimensional poses for use in estimation problems," *Transactions on Robotics*, vol. 30, 2014.

[25] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiraux, O. Stasse, and N. Mansard, "The Pinocchio c++ library, a fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives," in *IEEE International Symposium on System Integrations*, 2019.

[26] L. Joseph, V. Padois, and G. Morel, "Towards x-ray medical imaging with robots in the open: safety without compromising performances," in *IEEE International Conference on Robotics and Automation*, 2018.

[27] J. Ferreau, C. Kirches, A. Potschka, H. Bock, and M. Diehl, "qpoases: A parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, 12 2014.

[28] A. Skuric, V. Padois, N. Rezzoug, and D. Daney, "On-line feasible wrench polytope evaluation based on human musculoskeletal models: an iterative convex hull method," *IEEE Robotics and Automation Letters*, 2022. [Online]. Available: https://hal.inria.fr/hal-03369576

[29] L. Joseph, V. Padois, and G. Morel, "Experimental validation of an energy constraint for a safer collaboration with robots," in *International Symposium on Experimental Robotics*, Buenos Aires, Argentina, Nov. 2018.

[30] ——, "Online minimization of the projected mass of a robot for safe workspace sharing with a human," in *Workshop on "Human movement science for physical human-robot collaboration" at the IEEE International Conference on Robotics and Automation*, Montreal, Canada, May 2019.

[31] N. Mansfeld, B. Djellab, J. R. Veuthey, F. Beck, C. Ott, and S. Haddadin, "Improving the performance of biomechanically safe velocity control for redundant robots through reflected mass minimization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.

[32] B. Camblor, N. Benhabib, D. Daney, V. Padois, and J.-M. Salotti, "Task-consistent signaling motions for improved understanding in human-robot interaction and workspace sharing," in *2022 17th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2022.

[33] A. Skuric, V. Padois, and D. Daney, "On-line force capability evaluation based on efficient polytope vertex search," in *IEEE International Conference on Robotics and Automation*, Xi'an, China, May 2021.