Control Engineering Practice 142 (2024) 105769

Contents lists available at ScienceDirect

Control Engineering Practice

journal homepage: www.elsevier.com/locate/conengprac

Scenario-based model predictive control with probabilistic human predictions for human–robot coexistence

Artemiy Oleinikov [a], Sergey Soltan [b], Zarema Balgabekova [a], Alberto Bemporad [c], Matteo Rubagotti [a,*]

[a] Department of Robotics and Mechatronics, Nazarbayev University, 53 Kabanbay Batyr Ave, Astana, 010000, Kazakhstan
[b] Department of Computer Science, University of Milan, Via Giovanni Celoria 18, Milan, 20133, Italy
[c] IMT School for Advanced Studies, Piazza S.Francesco 19, Lucca, 55100, Italy

ARTICLE INFO

Keywords:
Nonlinear model predictive control
Robot motion planning
Physical human–robot interaction

ABSTRACT

This paper proposes a real-time motion planning scheme for safe human–robot workspace sharing relying on scenario-based nonlinear model predictive control (NMPC), a well-known approach for solving stochastic NMPC problems. A scenario tree is generated via higher-order Markov chains to provide probabilistic predictions of the human motion. Scenario-based NMPC is then used to generate point-to-point motions of the robot manipulator based on the above-mentioned human motion predictions, accounting for safety constraints via speed and separation monitoring (SSM). This means that the robot speed is always modulated to be able to stop before a possible collision with the human occurs. After proving theoretical properties on recursive feasibility and closed-loop stability of the proposed motion planning strategy, this is tested experimentally on a Kinova Gen3 robot interacting with a human operator, showing superior performance with respect to an NMPC scheme not relying on human predictions and to a fixed-path SSM strategy.

1. Introduction and related work

Research on physical human–robot interaction (pHRI) has been driven by the shift from standard manufacturing applications, with robots and human operators separated by fences, to collaborative applications, in which robots and humans share their workspace (Ajoudani et al., 2018; Li et al., 2023). Safety is a key requirement to enable seamless interaction and collaboration between human operators and robot manipulators (Haddadin & Croft, 2016). In industrial practice, human safety in the presence of collaborative robots is ensured via ergonomic design and by generating a motion that satisfies certain criteria, defined in the ISO 10218 and ISO/TS 15066 standards. A first possibility is to stop the robot whenever a human operator is detected in its workspace; this mode of operation, known as *safety-rated monitored stop* (SRMS), is suitable in case of occasional human presence. A second possibility is to vary the allowable robot speed based on the measured distance with the human, to ensure that the robot will be able to stop before a collision occurs. This is known as *speed and separation monitoring* (SSM). A third mode of operation is *power and force limiting* (PFL), in which the robot speed is limited – regardless of the distance with the human – so as to guarantee that collisions would not be harmful. PFL can only be applied to collaborative robots, with suitable ergonomic design, and is often used in case of constant human presence in the robot workspace. These methodologies are typically implemented for a fixed robot path either stopping the robot when needed (SRMS), applying velocity scaling (SSM) or limiting the maximum robot speed at an acceptably low threshold to allow for collisions (PFL). For more details on the topic, the interested reader is referred to the survey by Chemweno, Pintelon, and Decre (2020) and the references therein.

To improve performance while maintaining safety, one can plan the robot trajectory online at the same time satisfying one of the above-mentioned safety criteria (see, e.g., Flowers and Wiens (2023) and Tonola, Faroni, Pedrocchi, and Beschi (2021)). A possible solution is to use model predictive control (MPC) with SSM constraints, to improve performance compared to fixed-path SSM approaches such as those of Kim, Kirschner, Yamada, and Okamoto (2020), Marvel (2013), Marvel and Norcross (2017) and Rosenstrauch, Pannen, and Krüger (2018). Using MPC (Grüne & Pannek, 2017), the robot motion would be re-planned online at each sampling interval by numerically solving a finite-horizon optimal control problem (FHOCP). To the best of the authors' knowledge, the first paper published on SSM-based MPC was that of Zheng, Wieber, and Aycard (2020), in which linear MPC was used to track a desired robot trajectory: the obstacle represented by

* Corresponding author.
E-mail address: rubagotti@ieee.org (M. Rubagotti).

https://doi.org/10.1016/j.conengprac.2023.105769
Received 11 April 2023; Received in revised form 20 October 2023; Accepted 24 October 2023
Available online 1 November 2023

the human operator was accounted for by using separating planes (to keep the MPC problem linear), and the SSM condition was imposed by ensuring that the planned robot speed was equal to zero at the end of the prediction horizon. This approach was validated on a 7-DoF robot in simulation via Robot Operating System (ROS), accounting for the presence of a single human arm modeled as a cylinder. A different approach was defined by Oleinikov, Kusdavletov, Shintemirov, and Rubagotti (2021), proposing a nonlinear MPC (NMPC) strategy to enforce SSM constraints as defined in Marvel (2013). The space occupancy of both human and robot was accounted for using spheres, and the whole robot trajectory until the goal configuration was planned in the NMPC problem. As also highlighted by Zheng et al. (2020), since long-term predictions of the human motion are difficult to obtain, the robot trajectory in the work by Oleinikov et al. (2021) was continuously re-planned based on the currently measured human position. This approach was validated experimentally on a real Kinova Gen3 collaborative robot, simulating the presence of the human operator via a set of prerecorded human motions. The approach of Oleinikov et al. (2021) was used by Tusseyeva, Oleinikov, Sandygulova, and Rubagotti (2022) to assess perceived safety, while a neural-network-based approximation of the NMPC strategy of Oleinikov et al. (2021) was defined by Nurbayeva, Shintemirov, and Rubagotti (2023). Another approach was proposed by Eckhoff, Kirschner, Kern, Abdolshah, and Haddadin (2022), in which a human-aware Cartesian linear MPC motion planner was coupled with a safe motion unit that only accounted for the distance between robot end-effector and human wrist (so safety was not imposed with reference to the whole human body); this unit enforced in practice a simplified SSM condition, though SSM was never explicitly mentioned in the paper. This approach was validated experimentally on a Franka Emika Panda manipulator with an actual human operator. In the cited articles by Eckhoff et al. (2022), Nurbayeva et al. (2023), Oleinikov et al. (2021), Tusseyeva et al. (2022) and Zheng et al. (2020), an explicit prediction method of the human motion was never used, typically assuming a constant human pose along the prediction horizon.

To obtain human motion predictions, different methods were defined, such as artificial neural networks (Ravichandar & Dani, 2017), Gaussian mixture models (Luo & Berenson, 2015), Markov chains, and hidden Markov models (HMMs). Papers belonging to the last two categories are briefly described in the following, as they are related to the approach presented in this paper. The capability of HMMs to represent stochastic transitions between different motion patterns makes them a reasonable candidate for predicting and capturing complex factors such as human intentions (Ding et al., 2011). HMMs were used by Ding et al. (2011) to compute the region of the workspace possibly occupied by a human in future time instants. In the work of Liu and Wang (2017), assembly tasks were modeled as a sequence of human motions that were predicted based on a motion transition probability matrix generated via HMM. A trajectory clustering algorithm was introduced by Sung, Feldman, and Rus (2012) for the extraction of motion patterns that were then used for HMM training to predict the motion of a target for an interception task. Growing HMMs were used for human intention inference by Vasquez, Fraichard, and Laugier (2009) and Elfring, Van De Molengraft, and Steinbuch (2014). In the paper of Rohrmuller, Althoff, Wollherr, and Buss (2008), Markov chains were used to model and predict the dynamics of moving humans for a "soft" risk mapping in populated environments. Finally, Zanchettin, Casalino, Piroddi, and Rocco (2019) proposed to model human activities during manufacturing and assembly tasks as higher-order Markov chains (HOMCs), developing a prediction algorithm to estimate the time before the occurrence of operations requiring human–robot collaboration.

Contribution

The first contribution of this paper is the definition of an HOMC-based human prediction method, explicitly tailored towards its application in conjunction with NMPC, which generates a number of representative poses of the human body associated with the discrete states of a Markov chain. In most collaborative assembly tasks, the positions most often assumed by a human operator are related to the locations of parts or tools (Zanchettin et al., 2019); this justifies the use of discrete states associated with specific human poses. However, knowing only the present discrete state (e.g., corresponding to the operator inserting a part at a given location) would not provide enough information to infer when the discrete state will change and where the operator will likely move afterward. This is the reason why HOMCs, which receive (as input) sequences of past discrete states within a given time horizon, are used in this work in place of standard Markov chains. The length of these sequences – referred to as *order* of the HOMC – is fixed, and its value must be chosen based on the specific task. As the HOMC naturally outputs a tree of possible human motions with associated probabilities, scenario-based MPC (henceforth, S-MPC) is adopted in this paper. S-MPC is one of the various stochastic MPC algorithms proposed in the literature; for an overview, the reader is referred to the works of Mesbah (2016) and Mesbah, Kolmanovsky, and Di Cairano (2019). In S-MPC (Bernardini & Bemporad, 2009, 2011; Di Cairano, Bernardini, Bemporad, & Kolmanovsky, 2013), sometimes referred to as *branch MPC*, the MPC algorithm considers a finite number of realizations of parametric uncertainties and/or additive disturbances to generate a scenario tree, and then imposes constraint satisfaction for all scenarios. As S-MPC typically relies on multi-stage stochastic programming, it is also often referred to as *multi-stage MPC* (Krishnamoorthy, Suwartadi, Foss, Skogestad, & Jäschke, 2018; Lucia, Finkler, & Engell, 2013; Lucia, Subramanian, Limon, & Engell, 2020). S-MPC is not the only stochastic MPC algorithm. For example, a related approach – mainly known as well as *scenario-based MPC* but sometimes referred to as *randomized MPC* (Calafiore & Fagiano, 2013; de Groot, Brito, Ferranti, Gavrila, & Alonso-Mora, 2021; Pedersen & Petersen, 2018; Schildbach, Fagiano, Frei, & Morari, 2014; Shang & You, 2019) – transforms an underlying chance-constrained stochastic optimization problem into a deterministic problem with a large number of deterministic constraints, obtained for multiple realizations of the disturbances, sampled according to their probability density functions. Other well-known approaches are *tube-based stochastic MPC* (Cannon, Kouvaritakis, Raković, & Cheng, 2010) and *polynomial chaos-based stochastic MPC* (Kim & Braatz, 2013). In this work an S-MPC (more precisely, S-NMPC) strategy is proposed following the ideas of Bernardini and Bemporad (2009, 2011), building the scenario tree in order to predict a factor (the future human pose) that influences the formulation of cost function and constraints in the FHOCP. In the standard formulation of Bernardini and Bemporad (2009, 2011), instead, the scenario tree was built to describe the prediction of disturbances or uncertainties that affect the system dynamics.

S-MPC was recently applied in the field of mobile robotics, to an autonomous quadruped robot in the presence of an uncontrolled quadruped robot (with experimental results) in Chen, Rosolia, Ubellacker, Csomay-Shanklin, and Ames (2022) and to autonomous driving (in simulation) in Chen et al. (2022) and Wang, Schuurmans, and Patrinos (2023). To the best of the authors' knowledge, S-MPC was never applied to safe pHRI for robot manipulators, and this is the second contribution of this paper.

The third and last contribution is the implementation of the proposed method on a Kinova Gen3 manipulator, and the experimental validation of the proposed approach with a human participant.

Paper structure

The remainder of the paper is organized as follows. Section 2 describes the modeling of the human operator for motion planning purposes, including the proposed prediction strategy via HOMCs. Section 3 focuses on robot modeling, with specific attention to the constraints to be imposed, including those that guarantee human safety via SSM. Section 4 describes the proposed S-NMPC strategy, and Section 5 describes the related theoretical results. Sections 6 and 7 describe the

considered case study, in which a Kinova Gen3 robot carries out a pick-and-place task while sharing its workspace with a human operator, and the related experimental results. Conclusions are finally drawn in Section 8.

Notation

A sequence of integer numbers from n_1 to n_2 included is referred to as $\mathbb{N}_{[n_1,n_2]}$; for example, $\mathbb{N}_{[3,6]} = \{3,4,5,6\}$. $\|\boldsymbol{g}\|$ indicates the Euclidean norm of vector $\boldsymbol{g} \in \mathbb{R}^n$. For the same vector, $\|\boldsymbol{g}\|^2_{\boldsymbol{M}} \triangleq \boldsymbol{g}'\boldsymbol{M}\boldsymbol{g}$, where $\boldsymbol{M} \in \mathbb{R}^{n\times n}$, and $'$ is the transposition operator. When writing $\boldsymbol{M} \succeq 0$ and $\boldsymbol{M} \succ 0$, it is meant that matrix $\boldsymbol{M}$ is either positive semi-definite or positive definite, respectively. Given $\boldsymbol{M} \succeq 0$, $\lambda_{\min}(\boldsymbol{M})$ and $\lambda_{\max}(\boldsymbol{M})$ are its minimum and maximum eigenvalues, respectively. In order to concisely express predictions of variables at the discrete time index $k+\tau$ given the information at time k or $k+1$, the following notation is used: $k_\tau \triangleq k+\tau \mid k$ and $k^+_\tau \triangleq k+\tau \mid k+1$.

2. Human modeling and prediction

2.1. Human space occupancy and Markov chain states

In order to represent its space occupancy, the portion of the human body involved in the considered task is included in the union of n_h spheres, each named $S_{h,j}$, $j \in \mathbb{N}_h \triangleq \mathbb{N}_{[1,n_h]}$, with the radius of each sphere equal to $R_{h,j}$. In the $O-xyz$ fixed reference frame centered at the robot base, the time-varying position of the center of each sphere (located inside the human body) is indicated as $\boldsymbol{p}_{h,j} \in \mathbb{R}^3$. As the radius of each sphere is fixed, the human pose can be expressed by the coordinates of the relevant sphere centers, which are concatenated in a single vector $\mathcal{H} \in \mathbb{R}^{3n_h}$, defined as

$$\mathcal{H} \triangleq \begin{bmatrix} \boldsymbol{p}'_{h,1} & \boldsymbol{p}'_{h,2} & \cdots & \boldsymbol{p}'_{h,n_h} \end{bmatrix}'. \tag{1}$$

The discrete states of the considered Markov chain are defined within a finite set, as $\xi \in \Xi \triangleq \mathbb{N}_{[1,n_\xi]}$, with $n_\xi \in \mathbb{N}_{>0}$. When detecting a given operator's pose $\mathcal{H}$, this must be associated with a discrete state $\xi = f(\mathcal{H})$ through a suitably-defined function $f : \mathbb{R}^{3n_h} \to \Xi$. On the other hand, in order to determine the "typical" human pose $\mathcal{H}$ associated with a given discrete state ξ, one could obtain the typical posture of the operator for each activity, thus obtaining a function defining $\mathcal{H} = g(\xi)$, with $g : \Xi \to \mathbb{R}^{3n_h}$. A specific example of how to determine these functions will be provided for the considered case study.

2.2. HOMC for human prediction

In this work, HOMCs are used to predict ξ, using a discrete-time framework where subsequent time instants k, $k+1$, etc., are generated using a fixed sampling interval T_s. Differently from standard Markov chains, in which only the current state is used to generate a prediction, HOMCs use a window of n_m values of ξ, i.e., the current value and n_m-1 past values. The probability of the next discrete state being equal to a fixed $\bar{\xi}_1 \in \Xi$ can be expressed as

$$\mathcal{P}\left(\xi(k+1) = \bar{\xi}_1 \mid \xi(k-n_m+1) = \bar{\xi}_{-n_m+1}, \dots, \xi(k-1) = \bar{\xi}_{-1}, \xi(k) = \bar{\xi}_0\right) \tag{2}$$

where, for $i = \mathbb{N}_{[0,n_m-1]}$, each $\bar{\xi}_{-i} \in \Xi$ represents current or past values of ξ. Defining a HOMC requires in general $n_\xi^{n_m+1}(n_\xi-1)$ parameters, which results in exponential complexity with respect to n_m, referred to as *order* of the HOMC (Raftery, 1985).

To reduce the number of parameters needed to describe the HOMC, *mixture transition distribution* (MTD) models (Berchtold & Raftery, 2002) are employed in this paper. In MTD models, the probability distribution in (2) is approximated by a convex combination of multiple-steps transition probabilities, as

$$\sum_{i=0}^{n_m-1} \lambda_i \mathcal{P}\left(\xi(k+1) = \bar{\xi}_1 \mid \xi(k-i) = \bar{\xi}_{-i}\right), \tag{3}$$

where λ_i are the $n_\xi^2(n_m+1)$ MTD model parameters, whose number increases only linearly – rather than exponentially – with n_m. A new notation $\mathbf{X}(k) \in \mathbb{N}^{n_\xi}$ is introduced for the Markov state at time k: $\mathbf{X}(k)$ is a column vector of zeros, except for the element corresponding to $\xi(k)$, which is equal to 1. Using this notation, following the approach of Raftery (1985), in this work the prediction of the probability distribution at time $k+1$ is calculated as

$$\hat{\mathbf{X}}(k+1) = \sum_{i=0}^{n_m} \lambda_i Q_i \mathbf{X}(k-i), \tag{4}$$

where $Q_i \in \mathbb{R}^{n_\xi \times n_\xi}$ denote the transition probability matrices, estimated from data together with the parameters λ_i. The most common solution to estimate these parameters, also used in this paper, is described in the work of Lèbre and Bourguignon (2008), and consists of approximating the maximum-likelihood estimate of the MTD model via expectation–maximization (EM). To use the EM algorithm, Lèbre and Bourguignon (2008) fitted the MTD model into the general framework of hidden-variable models to compute a maximum-likelihood estimate from incomplete data. Maximizing the log-likelihood of the incomplete data is equivalent to maximizing the log-likelihood of the complete model conditional on the observed sequence and the current parameter. The latter is computed in the expectation step of the EM algorithm, and maximized using the Lagrange method in the maximization step.

2.3. Generation of the scenario tree

The estimates of future discrete states defined in (4) are used to construct a scenario tree, which provides a probabilistic description of the future human motion, and is used to plan the robot motion via S-NMPC. A *scenario tree* is simply a connected acyclic graph of all possible state transitions with associated probabilities. The root node, which represents $\xi(k)$, has n_ξ children that correspond to each of the possible n_ξ states of the Markov chain at time $k+1$. The tree is expanded by assigning n_ξ children to each child node, up to a fixed horizon N_s in the future, i.e., up to time $k+N_s$. To each node after the root node is assigned the probability of the corresponding state occurring according to the HOMC prediction (4), determined based on the observed and/or already predicted discrete states, in a window of n_m values. The probability of any given branch of the tree (corresponding to a specific sequence of discrete states) can be calculated as the product of the probabilities of each node in that branch.

Directly using the scenario tree in the S-NMPC algorithm (which would be the typical approach in multi-stage MPC) would lead to considerable computational burden, especially relevant for fast systems such as robotic manipulators. Therefore, after the tree is constructed, for a fixed $n_s \in \mathbb{N}_{>0}$ only the n_s most likely sequences of human motion, referred to in the remainder of the paper as *scenarios*, are determined, with associated probability values $P^{(\sigma)}$, $\sigma \in \mathbb{N}_s \triangleq \mathbb{N}_{[1,n_s]}$.

2.4. Human motion prediction for S-NMPC

After sorting the n_s highest branch probabilities $P^{(\sigma)}$ in descending order, the σ^{th} sequence of predicted human poses can be obtained applying function $\mathcal{H} = g(\xi)$, already introduced in Section 2.1. The sequence is indicated as follows (please refer to the *Notation* in Section 1 for the meaning of k_τ):

$$\boldsymbol{\mathcal{H}}^{(\sigma)}_{N_s}(k) \triangleq \begin{bmatrix} \mathcal{H}(k)' & \mathcal{H}^{(\sigma)}(k_1)' & \dots & \mathcal{H}^{(\sigma)}(k_{N_s})' \end{bmatrix}' \in \mathbb{R}^{3N_s n_s}. \tag{5}$$

While the prediction of the human pose from time k_1 onward is obtained from the discrete states of the Markov chain using function $g(\xi)$, the current value $\mathcal{H}(k)$ simply corresponds to the measured human pose. The S-NMPC strategy considered in the following might need a longer sequence of values than $\boldsymbol{\mathcal{H}}^{(\sigma)}_{N_s}(k)$; this is the case if the prediction horizon of S-NMPC, namely N, is such that $N > N_s$. This longer sequence, referred to as $\boldsymbol{\mathcal{H}}^{(\sigma)}(k) \in \mathbb{R}^{3Nn_s}$, can be constructed by appending the last pose $\mathcal{H}^{(\sigma)}(k_{N_s})$ until the end of the S-NMPC prediction

horizon, thus implicitly assuming that the human will maintain the last predicted pose. Finally, the overall information for all n_s scenarios is referred to as

$$\boldsymbol{\mathcal{H}}(k) \triangleq \left\{\boldsymbol{\mathcal{H}}^{(1)}(k), \dots, \boldsymbol{\mathcal{H}}^{(n_s)}(k)\right\}. \tag{6}$$

3. Robot modeling

3.1. Robot space occupancy and kinematics

The space occupied by the manipulator in the above-mentioned $O{-}xyz$ reference frame is over-approximated as the union of n_r spheres. Their centers $\boldsymbol{p}_{r,i} \in \mathbb{R}^3$, $i \in \mathbb{N}_r \triangleq \mathbb{N}_{[1,n_r]}$ are placed at pre-determined locations along the robot links, and thus their coordinates in the $O{-}xyz$ frame depend on the robot configuration. The corresponding spheres, with radii equal to $R_{r,i}$, $i \in \mathbb{N}_r \triangleq \mathbb{N}_{[1,n_r]}$, are referred to as $\mathcal{S}_{r,i}$. For simplicity, it is assumed that $\boldsymbol{p}_{r,n_r}$ is chosen to correspond to the robot end effector location in the $O-xyz$ frame. Using unions of spheres to represent the space occupancy of both human and robot is convenient to easily obtain their distance (which only depends on sphere radii and centers, and not on the orientation of human limbs or robot links), and to directly use it as a closed-form expression in the FHOCP.

The robot configuration is given by the vector of joint variables $\boldsymbol{\theta} \in \mathbb{R}^{n_\theta}$. The position of each sphere center $\boldsymbol{p}_{r,i}$, $i \in \mathbb{N}_r$, is determined via forward kinematics through a set of suitable functions, here simply referred to as $\boldsymbol{\Phi}_{\text{fwd,i}}(\cdot) : \mathbb{R}^{n_\theta} \to \mathbb{R}^3$, as $\boldsymbol{p}_{r,i} = \boldsymbol{\Phi}_{\text{fwd,i}}(\boldsymbol{\theta})$. The differential kinematics function, which is indicated in this paper simply as $\boldsymbol{\Phi}_{\text{diff,i}}(\cdot) : \mathbb{R}^{n_\theta} \to \mathbb{R}^3$ is used to determine the velocity $\boldsymbol{v}_{r,i}$ of $\boldsymbol{p}_{r,i}$ as $\boldsymbol{v}_{r,i} = \boldsymbol{\Phi}_{\text{diff,i}}(\boldsymbol{\theta}, \boldsymbol{\omega})$, in which $\boldsymbol{\omega} \triangleq \dot{\boldsymbol{\theta}} \in \mathbb{R}^{n_\theta}$ is the vector of joint speeds.

3.2. Prediction of the robot motion

In the considered approach, reference joint velocities are provided to the manipulator inner control loops. When predicting the robot motion, it is assumed that these references are perfectly tracked. This choice was dictated by two reasons. Firstly, as a nonlinear program is solved by S-NMPC at each sampling time based on the robot dynamics, this simple model allows a much faster computation of the control law as compared to a full dynamic model with joint torques as inputs. Secondly, the actual joint torques cannot be directly imposed in most commercial robots, which instead rely on velocity references and inner control loops to track them. Thus, the proposed framework solves a real-time motion planning problem without direct focus on the low-level control loops.

As the S-NMPC law will be introduced in discrete time, the perfect tracking of the reference speeds, coinciding with the actual speeds $\boldsymbol{\omega}$, is defined as

$$\boldsymbol{\theta}(k+1) = \boldsymbol{\theta}(k) + T_s \boldsymbol{\omega}(k), \tag{7}$$

in which $k \in \mathbb{N}$ is the discrete-time index already used in Section 2. Eq. (7) can be interpreted as the exact discretization of continuous-time integrators. The S-NMPC controller will explore n_s different realizations, within a prediction horizon of N time steps, of the discrete-time control sequence

$$\bar{\boldsymbol{\omega}}^{(\sigma)}(k) \triangleq \left\{\boldsymbol{\omega}^{(\sigma)}(k_0), \boldsymbol{\omega}^{(\sigma)}(k_1), \dots, \boldsymbol{\omega}^{(\sigma)}(k_{N-1})\right\}, \tag{8}$$

where $\sigma \in \mathbb{N}_s$, and $\boldsymbol{\omega}^{(\sigma)}(k_\tau)$ is the prediction of $\boldsymbol{\omega}$ at time $k+\tau$, $\tau \in \mathbb{N}_{[0,N-1]}$, associated with the σ^{th} scenario of human motion determined via HOMCs. A related variable is the sequence of states obtained via (7) using $\bar{\boldsymbol{\omega}}^{(\sigma)}(k)$ and initial state $\boldsymbol{\theta}^{(\sigma)}(k_0)$ equal to the measured state $\boldsymbol{\theta}(k)$, namely

$$\bar{\boldsymbol{\theta}}^{(\sigma)}(k) \triangleq \left\{\boldsymbol{\theta}^{(\sigma)}(k_0), \boldsymbol{\theta}^{(\sigma)}(k_1), \dots, \boldsymbol{\theta}^{(\sigma)}(k_N)\right\}. \tag{9}$$

For compactness of notation, all n_s input or state sequences are written as single variables, and precisely

$$\bar{\boldsymbol{\omega}}(k) \triangleq \left\{\bar{\boldsymbol{\omega}}^{(1)}(k), \dots, \bar{\boldsymbol{\omega}}^{(n_s)}(k)\right\},$$

$$\bar{\boldsymbol{\theta}}(k) \triangleq \left\{\bar{\boldsymbol{\theta}}^{(1)}(k), \dots, \bar{\boldsymbol{\theta}}^{(n_s)}(k)\right\}.$$

3.3. Motion constraints

Certain constraints to be imposed on the robot motion do not depend on the human operator. The first of these constraints consists of bounds on joint speeds; these constraints are referred to as $\boldsymbol{\omega} \in \Omega$, where $\Omega \subseteq \mathbb{R}^{n_\theta}$ is a compact set (typically, a hyper-rectangle) that includes the origin in its interior (otherwise, some joints would be able to move only in one direction).

A second set of constraints concerns the limits on joint angles, which can also be imposed to avoid self-collisions. These constraints are usually imposed on each joint angle separately so the set $\Theta \subseteq \mathbb{R}^{n_\theta}$ defined to impose $\boldsymbol{\theta} \in \Theta$ is typically a box.

The third considered type of constraints is the avoidance of fixed obstacles, which can be imposed using different methods. In general, it is imposed that each point $\boldsymbol{p}_{r,i}$, $i \in \mathbb{N}_r$, belongs to a set $\mathcal{P}_{r,i}$ defined such that the corresponding sphere $\mathcal{S}_{r,i}$ does not intersects with any obstacles. In short,

$$\boldsymbol{p}_{r,i} \in \mathcal{P}_{r,i}, \ i \in \mathbb{N}_r. \tag{10}$$

In the considered case study, it will be imposed that the robot remains above a flat horizontal surface. Thus, condition (10) is obtained requiring that the Euclidean distance between each sphere center $\boldsymbol{p}_{r,i}$, $i \in \mathbb{N}_r$, and the hyperplane that describes the obstacle surface is larger than $R_{r,i}$. As the robot tasks considered in this work always consist of point-to-point motions, it is assumed that the initial configuration and all configurations that are used as goals – the current goal configuration is referred to as $\boldsymbol{\theta}_g$ – are included in Θ and defined such that the corresponding sphere centers satisfy (10).

The last type of constraints are safety constraints aimed at guaranteeing that, under the assumption that no points of the human operator will ever exceed a given speed v_h (as defined in the ISO/TS 15066 standard), an impact between the operator and a moving part of the robot will never occur. In other words, the robot will be able to stop the links involved in a collision before the collision happens, under the conservative assumption that the human operator can always move its closest point to the robot towards the robot itself with speed v_h. This idea was already explained by Oleinikov et al. (2021), in turn based on an adaptation of Eq. (14) in the paper by Marvel (2013), which corresponds to the ISO/TS 15066 SSM criterion. A summary of the main concepts is here reported for the reader's convenience.

First of all, let $d_{ij} \triangleq \|\boldsymbol{p}_{r,i} - \boldsymbol{p}_{h,j}\|$, $i,j \in \mathbb{N}_r \times \mathbb{N}_h$, from which one can define the distance between the ith robot sphere and the closest human sphere as $d_{ih} \triangleq \min_{j \in \mathbb{N}_h} \left\{d_{ij} - (R_{r,i} + R_{h,j})\right\}$. Let the speed of each robot sphere center be $v_i \triangleq \|\boldsymbol{v}_i\|$, $i \in \mathbb{N}_r$.

The SSM criterion can be expressed as follows:

$$\bar{v}_h \left(T_{dr} + \frac{v_i}{\bar{a}_r}\right) + v_i T_{dr} + \frac{v_i^2}{2\bar{a}_r} + \epsilon_s \le d_{ih}, \ i \in \mathbb{N}_r \tag{11}$$

where $\bar{a}_r$ is the maximum robot deceleration, T_{dr} is the time of detection of the human pose and related reaction time, and ϵ_s is the precision of the motion capture system that detects the human position. For each robot sphere center, the inequality in (11) can be read as follows: the maximum distance that the human can cover while the robot detects it, reacts to it, and comes to a stop, plus the distance that the robot would cover in the same time interval, plus the human measurement precision, must not exceed the current human–robot distance.

Condition (11) is difficult to implement in a numerical optimization problem, as v_i and d_{ih} are Euclidean norms, and as such they contain square roots in their expressions. This typically causes problems for the numerical solver; in order to avoid these issues, a condition slightly more conservative than (11) is instead imposed, which only contains v_i^2 and d_{ij}^2, so as to avoid square roots. The condition is the following:

$$v_i^2 \le \alpha^2 \left(d_{ij}^2 - \left(R_{r,i} + R_{h,j} + \bar{d}\right)^2\right), \ i,j \in \mathbb{N}_r \times \mathbb{N}_h \tag{12}$$

in which α and $\bar{d}$ are positive real tuning parameters, set such that (11) is satisfied for all values of interest of d_{ih}. These parameters are

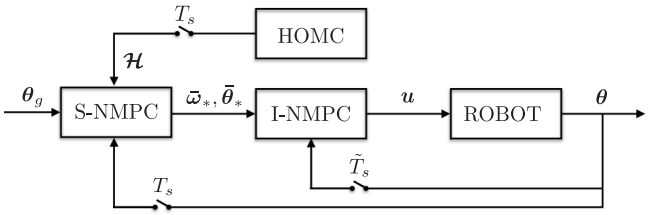


Fig. 1. Schematic of the overall robot motion planning and control system.

tuned via trial and error, aiming at obtaining a boundary that is the least conservative possible with respect to that defined by (11). For a graphical representation of these curves, which are not a contribution of this work, the interested reader is referred to Fig. 3 in the paper by Oleinikov et al. (2021).

Assumption 1. The human motion always happens so that there exists a fixed threshold $\delta_h \in \mathbb{N}_{>0}$ such that $\boldsymbol{\theta} = \boldsymbol{\theta}_g$ implies $d_{ij}^2 \geq (R_{r,i} + R_{h,j} + \bar{d})^2 + \delta_h$ for $i, j \in \mathbb{N}_r \times \mathbb{N}_h$. □

Remark 1. Assumption 1 requires that, when the robot is at its goal configuration with zero speed, the human is always maintaining a distance that guarantees the satisfaction of the SSM constraints. This condition is needed because the SSM constraints account for the detection and reaction time in addition to the robot speed also when the latter is equal to zero. As a consequence, even when the robot has zero speed, if the human moves too close to it the SSM constraints might be violated.

4. S-NMPC-based robot motion planning

4.1. Overall control scheme

The closed-loop control scheme proposed in this work is made of a cascade of different controllers, explained in the following and depicted in Fig. 1. The highest level consists of the proposed S-NMPC, which acquires the current robot state $\boldsymbol{\theta}(k)$ and the predictions of the human motion $\boldsymbol{\mathcal{H}}(k)$ from the HOMC, and generates the control variable $\boldsymbol{\omega}(k)$ by solving a FHOCP. Ideally, one should use a sampling interval T_s that is (i) small enough to capture the robot dynamics and to generate an acceptably low value of T_{dr} in (11), and (ii) large enough to allow reaching the goal configuration $\boldsymbol{\theta}_g$ for a computationally reasonable value of N. However, to satisfy condition (i), one would need a very large N, which, with the current state of the art in optimization algorithms and computational capabilities, would lead to extremely large solution times. Therefore, T_s was chosen as the smallest possible interval that still satisfies condition (ii), which is typically in the order of fractions of a second. The optimal input and state sequences $\bar{\boldsymbol{\omega}}_*$ and $\bar{\boldsymbol{\theta}}_*$ are then passed as a reference to another NMPC controller (referred to in the following simply as inner-loop NMPC, or I-NMPC), analogous to that explained by Oleinikov et al. (2021), which acts with a smaller sampling time $\tilde{T}_s$ in the order of tens of milliseconds. I-NMPC solves a tracking problem using the same kinematic model (7) of S-NMPC, and re-determines the robot motion imposing the same constraints. Also, it assumes that the currently measured human pose (with its own sampling interval) remains constant throughout the prediction horizon. As already mentioned above, an important aim of I-NMPC is to reduce the value of T_{dr}, which would be too high with the sampling interval of S-NMPC. The joint speed values, namely $\boldsymbol{u}$, determined by I-NMPC are passed to the embedded robot controller that acts with a sampling time ranging from hundreds of microseconds to a few milliseconds. This embedded control loop is typically already implemented and impossible to modify in commercial robots. It enforces limits on joint torques and accelerations, to protect the manipulator against reaching the joint angle limits and against torque overload.

To design and analyze the S-NMPC algorithm it will be assumed that I-NMPC and the embedded robot controllers manage to achieve perfect tracking of the reference joint speeds $\boldsymbol{\omega}(k)$ coming from S-NMPC. The actual performance of the closed-loop system will be then verified experimentally.

4.2. S-NMPC: cost function

The cost function to be minimized by S-NMPC contains four types of terms. The first (which can be referred to as "state term") is given by

$$e_\theta^{(\sigma)}(k_\tau) \triangleq \|\boldsymbol{\theta}^{(\sigma)}(k_\tau) - \boldsymbol{\theta}_g\|_{\boldsymbol{Q}}^2 \tag{13}$$

where $\boldsymbol{Q} = \boldsymbol{Q}' \in \mathbb{R}^{n_\theta \times n_\theta}$, with $\boldsymbol{Q} \succ 0$, which penalizes the error on robot pose. The second ("control term") is

$$e_\omega^{(\sigma)}(k_\tau) \triangleq \|\boldsymbol{\omega}^{(\sigma)}(k_\tau)\|_{\boldsymbol{R}}^2 \tag{14}$$

where $\boldsymbol{R} = \boldsymbol{R}' \in \mathbb{R}^{n_\theta \times n_\theta}$, with $\boldsymbol{R} \succ 0$, which penalizes unnecessarily large joint speeds. The third (which can be called "repulsive term" for reasons that will be explained later in this paragraph), influenced by the predicted human motion, is defined as

$$e_\varphi^{(\sigma)}(k_\tau) \triangleq \gamma \cdot \left(r^{(\sigma)}(k_\tau)\right)^2, \tag{15}$$

with

$$r^{(\sigma)}(k_\tau) \triangleq \begin{cases} \exp\left(-\beta \dfrac{\left(\delta_h^{(\sigma)}(k_\tau)\right)^2}{\|\boldsymbol{\theta}^{(\sigma)}(k_\tau) - \boldsymbol{\theta}_g\|^2}\right), & \text{if } \boldsymbol{\theta}^{(\sigma)}(k_\tau) \neq \boldsymbol{\theta}_g, \\ 0, & \text{otherwise}, \end{cases} \tag{16}$$

where $\gamma, \beta \in \mathbb{R}_{>0}$ are design parameters, whereas $\delta_h^{(\sigma)}(k_\tau)$ is the distance, for the σ^{th} scenario, between the end-effector location and the location of a specific human sphere center calculated based on $\mathcal{H}^{(\sigma)}(k_\tau)$. As these are all defined inside the human body (see Section 2), then there exists $\delta_{\min} \in \mathbb{R}_{>0}$ such that

$$\delta_h^{(\sigma)}(k_\tau) \geq \delta_{\min}, \ \sigma \in \mathbb{N}_s, \ \tau \in \mathbb{N}_{[0,N-1]}. \tag{17}$$

Notice that in practice $\boldsymbol{\theta}^{(\sigma)}(k_\tau)$, depending on actual measurements, is never exactly equal to $\boldsymbol{\theta}_g$, so the value of $r^{(\sigma)}(k_\tau)$ can be directly defined as the first option in (16), without checking the condition on $\boldsymbol{\theta}^{(\sigma)}(k_\tau)$. However, the definition in (16) is cleaner from the mathematical standpoint, as, to be rigorous, the exponential function in (16) is not defined for $\boldsymbol{\theta}^{(\sigma)}(k_\tau) = \boldsymbol{\theta}_g$. The term $e_\varphi^{(\sigma)}(k_\tau)$ decreases as the end effector moves away from the human; it has the role of "pushing" the end effector away from the human (similarly to a repulsive artificial potential field, see, e.g., Khatib (1986)) in order to avoid motions that would result in the robot moving too close to the operator and, due to the presence of the safety constraints, often come to a stop. Note that the value of $\delta_{h,i}^{(h)}$ is used squared to avoid calculating square roots when solving the optimization problem, for the same reason already explained for (12).

The fourth term ("penalty term") in the cost function is given by

$$e_\varepsilon^{(\sigma)}(k_\tau) \triangleq \rho \sum_{i=1}^{n_r} \sum_{j=1}^{n_h} \left(\varepsilon_{ij}^{(\sigma)}(k_\tau)\right)^2 \tag{18}$$

where $\rho \in \mathbb{R}$, $\rho \gg 1$, is a scalar weight (tuning parameter), whereas $\varepsilon_{ij}^{(\sigma)}(k_\tau) \in \mathbb{R}$ is a slack variable related to soft constraints. Its definition will be provided in the next subsection.

Overall, the cost function for the σ^{th} scenario is

$$\boldsymbol{J}^{(\sigma)}\left(\bar{\boldsymbol{\omega}}^{(\sigma)}(k), \bar{\boldsymbol{\theta}}^{(\sigma)}(k), \boldsymbol{\mathcal{H}}^{(\sigma)}(k)\right) \triangleq \sum_{\tau=0}^{N-1} \ell^{(\sigma)}(k_\tau), \tag{19}$$

with

$$\ell^{(\sigma)}(k_\tau) \triangleq e_\theta^{(\sigma)}(k_\tau) + e_\omega^{(\sigma)}(k_\tau) + e_\varphi^{(\sigma)}(k_\tau) + e_\varepsilon^{(\sigma)}(k_\tau). \tag{20}$$

As S-NMPC will make use of the n_s most likely realizations $\mathcal{H}^{(\sigma)}(k)$, the overall cost function is constructed to account for all of these scenarios, weighting them based on the corresponding probabilities $P^{(\sigma)}$, or more precisely on their normalized values $P_n^{(\sigma)} \triangleq P^{(\sigma)} / \sum_{\sigma=1}^{n_s} P^{(\sigma)}$. In conclusion, the S-NMPC cost function is defined as

$$J\left(\bar{\boldsymbol{\omega}}(k), \bar{\boldsymbol{\theta}}(k), \mathcal{H}(k)\right) \triangleq \sum_{\sigma=1}^{n_s} P_n^{(\sigma)} J^{(\sigma)}\left(\bar{\boldsymbol{\omega}}^{(\sigma)}(k), \bar{\boldsymbol{\theta}}^{(\sigma)}(k), \mathcal{H}^{(\sigma)}(k)\right). \tag{21}$$

4.3. S-NMPC: constraints

The first constraint considered in the FHOCP is the system dynamics (7), which determines how the predicted state values for each scenario σ are obtained along the prediction horizon based on the predicted inputs:

$$\boldsymbol{\theta}^{(\sigma)}(k_{\tau+1}) = \boldsymbol{\theta}^{(\sigma)}(k_\tau) + T_s \boldsymbol{\omega}^{(\sigma)}(k_\tau), \tag{22a}$$

for $\tau \in \mathbb{N}_{[0,N-1]}$ and $\sigma \in \mathbb{N}_s$. The second constraint imposes that all predictions of the robot motion start at the measured robot configuration $\boldsymbol{\theta}(k)$:

$$\boldsymbol{\theta}^{(\sigma)}(k_0) = \boldsymbol{\theta}(k), \ \sigma \in \mathbb{N}_s. \tag{22b}$$

Similarly, the third condition requires that all predictions of the robot motion end, after N time instants, at the goal configuration $\boldsymbol{\theta}_g$:

$$\boldsymbol{\theta}^{(\sigma)}(k_N) = \boldsymbol{\theta}_g, \ \sigma \in \mathbb{N}_s. \tag{22c}$$

As several system trajectories are planned by S-NMPC, but only one control move can be applied to the system, a fourth (causality) constraint is imposed, requiring that the first $n_q \in \mathbb{N}_{[1,N]}$ moves of all n_s scenarios be the same:

$$\boldsymbol{\omega}^{(1)}(k_\tau) = \boldsymbol{\omega}^{(2)}(k_\tau) = \cdots = \boldsymbol{\omega}^{(n_s)}(k_\tau), \ \tau \in \mathbb{N}_{[0,n_q-1]}. \tag{22d}$$

Setting $n_q = 1$ will reduce the conservativity of S-NMPC as compared to higher values of n_q, which can anyway be used for example to reduce the complexity of the resulting optimization problem, or due to the need to provide a single speed profile for more than one sampling interval to I-NMPC (see Section 4.5). The fifth and sixth constraints impose the joint speed and angle constraints described in Section 3.3:

$$\boldsymbol{\omega}^{(\sigma)}(k_\tau) \in \Omega, \ \tau \in \mathbb{N}_{[0,N-1]}, \ \ \sigma \in \mathbb{N}_s, \tag{22e}$$

$$\boldsymbol{\theta}^{(\sigma)}(k_\tau) \in \Theta, \ \tau \in \mathbb{N}_{[0,N-1]}, \ \ \sigma \in \mathbb{N}_s. \tag{22f}$$

The seventh constraint concerns the avoidance of fixed obstacles, also explained in Section 3.3. By imposing condition (10) for all time instants and all scenarios, one obtains

$$\boldsymbol{p}_{r,i}^{(\sigma)}(k_\tau) \in \mathcal{P}_{r,i}, \ i \in \mathbb{N}_r, \ \tau \in \mathbb{N}_{[0,N-1]}, \ \ \sigma \in \mathbb{N}_s. \tag{22g}$$

The eighth constraint imposes safety based on (12) for all scenarios and all time instants:

$$\left(v_i^{(\sigma)}(k_\tau)\right)^2 \leq \alpha^2 \left[\left(d_{ij}^{(\sigma)}(k_\tau)\right)^2 - \left(R_{r,i} + R_{h,j} + \bar{d}\right)^2\right] + \varepsilon_{ij}^{(\sigma)}(k_\tau), \tag{22h}$$

for $i, j \in \mathbb{N}_r \times \mathbb{N}_h$, $\tau \in \mathbb{N}_{[0,N-1]}$ and $\sigma \in \mathbb{N}_s$, where $\varepsilon_{ij}^{(\sigma)}(k_\tau)$ are the slack variables introduced in (18). Soft constraints are used here to avoid situations in which S-NMPC cannot plan feasible robot motions due to unforeseen changes in the human pose. This might seem to pose a threat to human safety, however condition (12) is verified at each sampling instant of the I-NMPC loop, and the robot can be stopped before a possible collision happens, as guaranteed by the SSM principle. A ninth constraint has to be imposed to ensure that the slack variables are non-negative, i.e.,

$$\varepsilon_{ij}^{(\sigma)}(k_\tau) \geq 0, \tag{22i}$$

for all $i, j \in \mathbb{N}_r \times \mathbb{N}_h$, $\tau \in \mathbb{N}_{[0,N-1]}$, and $\sigma \in \mathbb{N}_s$.

4.4. S-NMPC: FHOCP

With reference to the input and state sequences defined in (8) and (9), the goal of S-NMPC is to find their optimal realizations, namely $\bar{\boldsymbol{\omega}}_*^{(\sigma)}(k)$ and $\bar{\boldsymbol{\theta}}_*^{(\sigma)}(k)$, for all considered values of σ. All optimal realizations can be aggregated into single variables, namely $\bar{\boldsymbol{\omega}}_*(k)$ and $\bar{\boldsymbol{\theta}}_*(k)$. The optimal sequences are found by solving the following FHOCP:

$$\left(\bar{\boldsymbol{\omega}}_*(k), \bar{\boldsymbol{\theta}}_*(k)\right) = \arg \min_{\bar{\boldsymbol{\omega}}(k), \bar{\boldsymbol{\theta}}(k)} J\left(\bar{\boldsymbol{\omega}}(k), \bar{\boldsymbol{\theta}}(k), \mathcal{H}(k)\right), \tag{23}$$
$$\text{subj. to (22a)–(22i).}$$

After all optimal control sequences and corresponding state evolutions are obtained, only the first control move $\boldsymbol{\omega}_*^{(\sigma)}(k_0)$ (which is the same for all $\sigma \in \mathbb{N}_s$, according to condition (22d)) is passed to the inner control loop consisting of I-NMPC planner, embedded root controller and manipulator (see Fig. 1) as $\boldsymbol{\omega}(k)$. Another FHOCP in form (23) is then solved again at time $k+1$ based on newly available information on robot pose and human motion predictions.

4.5. Overview of I-NMPC

It is important to highlight that, when analyzing the theoretical results on S-NMPC stability, the tracking operated by I-NMPC and robot controllers is assumed to be perfect, and thus $\boldsymbol{\omega}(k)$ is considered as input of the controlled system. In practice, the angular speed signal from S-NMPC is resampled with a sampling interval $\tilde{T}_s$ (sub-multiple of T_s), and renamed as $\boldsymbol{u}_d(k_\tau)$. Given the last measured state (robot angular positions) considered by S-NMPC – here simply referred to as $\boldsymbol{\zeta}_0$ – a reference signal is generated as

$$\boldsymbol{\zeta}_d(k_{\tau+1}) = \boldsymbol{\zeta}_d(k_\tau) + \tilde{T}_s \boldsymbol{u}_d(k_\tau), \tag{24}$$

with $\boldsymbol{\zeta}_d(k_0) = \boldsymbol{\zeta}_0$. The value of $\boldsymbol{\zeta}_d$ is reinitialized every time a new S-NMPC solution becomes available. Naming $\boldsymbol{\zeta}$ the state of the system for the I-NMPC planner (acquired from the robot sensors with sampling time $\tilde{T}_s$), a cost function is minimized that – after changing the names of input and state to $\boldsymbol{u}$ and $\boldsymbol{\zeta}$ – is identical to (21), apart from the following differences. First of all, the human pose used in the prediction remains constant at the last value $\mathcal{H}$ measured when running S-NMPC. This implies that only one scenario is used, i.e., $n_s = 1$ for I-NMPC. Second, the "state term" (13) becomes equal to

$$\|\boldsymbol{\zeta}(k_\tau) - \boldsymbol{\zeta}_d(k_\tau)\|_{\boldsymbol{Q}}^2 \tag{25}$$

to track the ideal robot motion planned by S-NMPC.

The solved FHOCP, again after changing the sampling interval and the variable names, has the same constraints as problem (23), again with some notable differences. First, the terminal constraint (22c) is not imposed, as I-NMPC does not solve a regulation problem. Second, (22d) is also not imposed: it would be redundant, as only one scenario is considered. Third, (22h) is formulated as hard constraint, to enforce SSM in practice. This also leads to the removal of (22i), as no slack variables are present. The prediction horizon of I-NMPC, namely $\tilde{N}$, can in principle be different from N, and in any case will correspond to a much shorter time interval than that of S-NMPC.

Given the computational delays of S-NMPC and the need for I-NMPC to use a non-negligible prediction horizon, in practice one should use more than one control move from S-NMPC to define $\boldsymbol{u}_d(k_\tau)$. Therefore, it is advisable to set values of n_q in S-NMPC greater than one, such as $n_q = 2$ or $n_q = 3$. Excessively high values of n_q would force the same robot motion for all scenarios for a considerable portion of the S-NMPC prediction horizon, thus increasing conservativity.

5. Theoretical results

Implementing a computationally intensive control method such as S-NMPC on a fast system such as a robot manipulator is extremely challenging, and thus some simplifying assumptions had to be introduced, in particular approximating the system dynamics as simple integrators (Section 3.2). The effectiveness of the proposed approach is first shown by proving theoretical properties under ideal assumptions (e.g., no uncertainties in the system dynamics), and then (in the next two sections) by showing the practical results obtained in the shown case study.

Theorem 1 (*Recursive Feasibility*). *Consider the closed-loop system given by (7), with $\boldsymbol{\omega}(k)$ given by the solution of (23), and assume that the FHOCP (23) is feasible at time k. Then, the FHOCP is still feasible at time $k+1$.*

Proof. See Appendix A. ■

To prove a general result on closed-loop stability, two assumptions are introduced. The first is on the characteristics of the predicted robot motion.

Assumption 2. Given the FHOCP solution at time k, there exists a scenario $\hat{\sigma} \in \mathbb{N}_s$ that satisfies the following conditions:

$$\sum_{\tau=n_q+1}^{N-1} \|\boldsymbol{\theta}_*^{(\hat{\sigma})}(k_\tau)\|_{\boldsymbol{\theta}_g}^2 \leq \frac{\lambda_{\max}(\boldsymbol{Q})}{\lambda_{\min}(\boldsymbol{Q})} \sum_{\tau=n_q+1}^{N-1} \sum_{\substack{\sigma=1 \\ \sigma\neq\hat{\sigma}}}^{n_s} P_{nn}^{(\sigma)} \|\boldsymbol{\theta}_*^{(\sigma)}(k_\tau)\|_{\boldsymbol{\theta}_g}^2, \tag{26}$$

$$\sum_{\tau=n_q}^{N-1} \|\boldsymbol{\omega}_*^{(\hat{\sigma})}(k_\tau)\|_{\boldsymbol{\theta}_g}^2 \leq \frac{\lambda_{\max}(\boldsymbol{R})}{\lambda_{\min}(\boldsymbol{R})} \sum_{\tau=n_q}^{N-1} \sum_{\substack{\sigma=1 \\ \sigma\neq\hat{\sigma}}}^{n_s} P_{nn}^{(\sigma)} \|\boldsymbol{\omega}_*^{(\sigma)}(k_\tau)\|_{\boldsymbol{\theta}_g}^2, \tag{27}$$

where $P_{nn}^{(\sigma)} \triangleq P_n^{(\sigma)} / \sum_{\substack{j=1 \\ j\neq\hat{\sigma}}}^{n_s} P_n^{(j)}$. □

The second assumption accounts for the fact that the convergence of the robot configuration to its goal is strongly influenced by the human motion. Given an arbitrary scenario value $\sigma \in \mathbb{N}_s$, define $\boldsymbol{\mathcal{H}}^{(\sigma)}(k_\tau)$ as the predicted human pose extracted from $\boldsymbol{\mathcal{H}}(k)$ relative to time $k+\tau$, and $\boldsymbol{\mathcal{H}}^{(\sigma)}(k_\tau^+)$ as the predicted human pose extracted from $\boldsymbol{\mathcal{H}}(k+1)$ relative to time $k+\tau$. Given a specific scenario $\hat{\sigma} \in \mathbb{N}_s$, define $\tilde{\delta}_{h,\hat{\sigma}}^{(\sigma)}\left(k_\tau^+\right)$ as the value of $\delta_h^{(\sigma)}$ (introduced after Eq. (16)) associated with time $k+\tau$, calculated using the human pose $\boldsymbol{\mathcal{H}}^{(\sigma)}(k_\tau^+)$ (as defined above) and the robot state $\boldsymbol{\theta}_*^{(\hat{\sigma})}(k_\tau)$. Also, define $\tilde{e}_{\varepsilon,\hat{\sigma}}^{(\sigma)}(k_\tau^+)$ as the value of $e_\varepsilon^{(\sigma)}$ (introduced in (18)) associated with time $k+\tau$, calculated using the human pose $\boldsymbol{\mathcal{H}}^{(\sigma)}(k_\tau^+)$ (as defined above) and robot position and velocity equal to $\boldsymbol{\theta}_*^{(\hat{\sigma})}(k_\tau)$ and $\boldsymbol{\omega}_*^{(\hat{\sigma})}(k_\tau)$, respectively.

Assumption 3. Given the FHOCP solution at time k and the human motion predictions $\boldsymbol{\mathcal{H}}(k)$ and $\boldsymbol{\mathcal{H}}(k+1)$, there exists $\hat{\sigma} \in \mathbb{N}_s$ such that

$$\sum_{\tau=1}^{N-1} \min_{\sigma\in\mathbb{N}_s} \tilde{\delta}_{h,\hat{\sigma}}^{(\sigma)}\left(k_\tau^+\right) \geq \sum_{\tau=1}^{N-1} \max_{\sigma\in\mathbb{N}_s} \delta_{h*}^{(\sigma)}(k_\tau), \tag{28}$$

$$\sum_{\tau=1}^{N-1} \max_{\sigma\in\mathbb{N}_s} \tilde{e}_{\varepsilon,\hat{\sigma}}^{(\sigma)}(k_\tau^+) \leq \sum_{\tau=1}^{N-1} \min_{\sigma\in\mathbb{N}_s} \tilde{e}_{\varepsilon*}^{(\sigma)}(k_\tau), \tag{29}$$

where $\delta_{h*}^{(\sigma)}(k_\tau)$ and $\tilde{e}_{\varepsilon*}^{(\sigma)}(k_\tau)$ are the values of $\delta_h^{(\sigma)}(k_\tau)$ and $\tilde{e}_\varepsilon^{(\sigma)}(k_\tau)$, respectively, associated with the optimal FHOCP solution at time k. □

Theorem 2 (*Stability*). *Consider the closed-loop system given by (7), with $\boldsymbol{\omega}(k)$ given by the solution of (23). Assume that a feasible solution of the FHOCP (23) exists at time k, and that Assumptions 1–3 hold, with Assumptions 2 and 3 being satisfied by the same scenario $\hat{\sigma}$. Then, the goal configuration $\boldsymbol{\theta}_g$ is a uniformly asymptotically stable equilibrium point for the closed-loop system.*

Proof. See Appendix B. ■

Remark 2. Assumptions 2 and 3 represent sufficient conditions that provide some insight into what determines the robot pose to converge to its goal. Assumption 2 requires that the planned robot position and velocity associated with one specific scenario $\hat{\sigma}$ correspond to cost terms that are in a way lower than those associated with alternative scenarios, so that using only this scenario to determine the robot motion at the next time instant will prevent the corresponding cost terms from increasing. Assumption 3 instead has to rely on the human motion predicted at time $k+1$ as well. Its first part requires that the predicted distances between a specific point on the human and the robot end effector identified by δ_h increase from k to $k+1$: this is intuitively explained by the fact that, if the human moves away from the robot, the corresponding cost term decreases. Its second part, instead, requires a decrease of the slack variables which is difficult to be explained intuitively, as it depends on the whole predicted human and robot motions. A simple case in which this part of the assumption is satisfied is when the SSM constraints hold with zero values of the slack variables at time $k+1$.

Remark 3. As detailed in the proof in Appendix B, closed-loop stability, as implied by the sufficient conditions of Assumptions 2 and 3, relies on the constant decrease of the optimal S-NMPC cost function. In practical applications, this might not hold at the time instants in which the re-planned robot motion, due to a change in the human motion prediction, is associated with a larger value of the cost function. As a matter of fact, it is impossible to guarantee that the robot reaches the goal point in general, as the human operator might in theory even move to prevent it on purpose. When describing the experimental results, it will be shown that, in practice, the robot always reached the goal configuration.

6. Case study

In the case study analyzed in this paper, a *Kinova Gen3 collaborative robot* was employed, together with an OptiTrack optical motion capture system, consisting of 12 OptiTrack PrimeX13 cameras placed on a cubic cage with side of 3 m. The human motion was tracked based on the position of infrared markers attached to a special suit. The markers position was processed by the Motive software on a Windows PC (Intel Core i9-7900X CPU with 16 GB RAM). Data from this PC were continuously broadcast to a Linux PC (also Intel Core i9-7900X CPU with 16 GB RAM), interfaced with the robot, and on which the motion planning algorithms were implemented. Two tables, with surfaces at the same level, were placed inside the cube: the manipulator was located on the first table, while the human subject sat at the second table, facing the manipulator.

The task to be executed by the human operator, for which the reader can refer to Fig. 2, consisted of first picking up a 3D-printed plastic screw from a container (Box #1) placed close by. Three different types of screws were present in the container, identified by three different colors: gray, white, and olive. Each color corresponded to another box on the table (Box #2, #3 and #4), in which the screw had to be inserted. After inserting a screw, the operator would pick up another, and continue repeating the task until the end of the given time interval, equal to ten minutes. While the human performed this task, the manipulator picked up 3D-printed cubes on the same table, and placed them at given different locations. As the motions of human and robot shared the same workspace, safety was ensured via SSM for all the employed algorithms. In the following, the parameters of the HOMC and S-NMPC implementations are described. The described experimental procedure was approved by the Nazarbayev University Institutional Research Ethics Committee (NU-IREC), and participant's informed consent was obtained in writing.

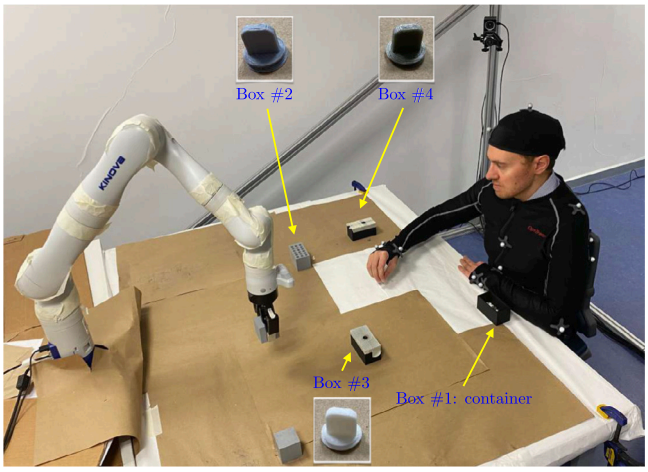
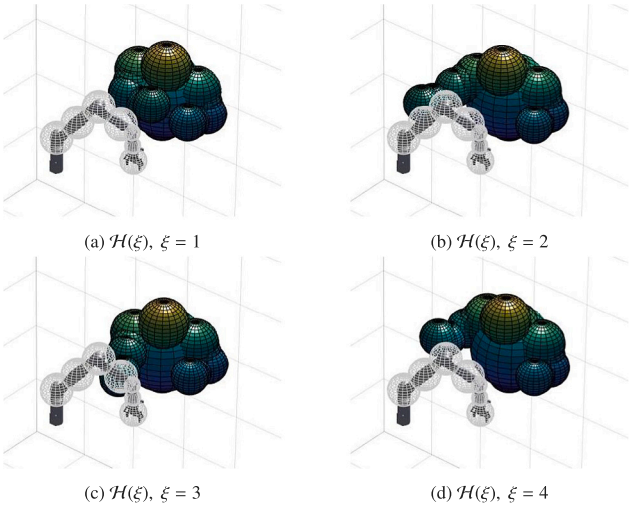


Fig. 2. In the shared workspace, it is possible to see the robot while it moves a cube between given locations. It is also possible to see four boxes where the operator picks up and places the screws. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) $\mathcal{H}(\xi)$, $\xi = 1$ (b) $\mathcal{H}(\xi)$, $\xi = 2$

(c) $\mathcal{H}(\xi)$, $\xi = 3$ (d) $\mathcal{H}(\xi)$, $\xi = 4$

Fig. 3. Space occupancy representation of the robot, and of the human for the four "typical" configurations $\mathcal{H}(\xi)$ associated with the four discrete states $\xi \in \mathbb{N}_{[1,4]}$ of the HOMC in the shown case study. Notice how each of these poses corresponds to the human operator reaching the box of the same number in Fig. 2.

6.1. Human modeling and prediction

Given the locations of human and robot, the latter could never reach the operator's legs, and therefore these were not accounted for when describing human space occupancy. The operator's upper body was instead covered by $n_h = 14$ spheres $\mathcal{S}_{h,j}$. Their radii $R_{h,j}$ took values, in meters, equal, from $j = 1$ to $j = 14$, to 0.19, 0.33, 0.18, 0.18, 0.15, 0.15, 0.15, 0.15, 0.13, 0.13, 0.14, 0.14, 0.19, and 0.17. As, given the task description, the operator was expected to stop mainly at four locations, it was decided to choose $n_\xi = 4$ discrete states for the HOMC. To obtain functions $f(\cdot)$ and $g(\cdot)$, data were acquired from a human participant while executing the above-described task without moving robot, for a total duration of 10 min. The same human participant will later execute the task with the moving robot. As n_ξ was known, the K-means clustering algorithm was used to group the measured human poses into four clusters, based on the subset of $\mathcal{H}$ that identified the Cartesian position of the participant's right hand in the $O - xyz$ frame. This was done by first acquiring the samples of the right human hand position, and then excluding those samples with a Cartesian velocity higher than the average value, to focus on stationary poses. Once the centroids of the clusters were determined and each data point was labeled as belonging to a given cluster, the centroids could be used to define $\xi = f(\mathcal{H})$ by simply determining the discrete state $\xi \in \mathbb{N}_{[1,4]}$ corresponding to the centroid that was closest to the measured human hand position. For a given $\bar{\xi}$ (four in total), the "typical" human pose $\mathcal{H} = g(\bar{\xi})$ was instead found as follows: first, the element-wise mean value of $\mathcal{H}$ was calculated for the data cluster corresponding to $\bar{\xi}$; then, $g(\bar{\xi})$ was determined as the actual value of $\mathcal{H}$ present in the data cluster, which was the closest to the element-wise mean value in terms of mean square error. The resulting four human poses are shown in Fig. 3, where it is also possible to see the spheres used to over-approximate the volumes occupied by human and robot.

These functions were used to build a HOMC with $T_s = 0.5$ s (equal to the S-NMPC sampling interval), which iteratively built the tree of probabilistic transitions between the four activities, based on past measured values on a window of $n_m = 6$ time instants, and predicting the possible human motions for $N_s = 6$ instants ahead. For a given human participant, the values of all parameters λ_i and Q_i of the MTD model were obtained – based on the above-mentioned 10 min of recorded data executing the described task without moving robot – using the approach of Lèbre and Bourguignon (2008) mentioned in Section 2.2. An available version of this algorithm could be directly used, as it is implemented in the *mtdlearn* Python library.

When running the HOMC in real time, the n_s most likely scenarios – with n_s ranging from 1 to 5 – were extracted, thus obtaining sequences of predicted human poses $\boldsymbol{\mathcal{H}}(k)$ for $N = 10$ time instants, covering the whole S-NMPC prediction horizon. For this purpose, one should notice that, at each depth of the tree, the probability of a branch can only decrease or remain unchanged due to the fact that the probability assigned to each node is less than or equal to one by definition. Therefore, at each step of the expansion, the node associated with the branch with the highest probability is expanded, and a heap data structure is used to store unexpanded nodes. The algorithm expands the tree iteratively until N_s steps in the future are reached, and the node at the top of the heap corresponds to the branch with the highest probability. The same calculations are repeated to determine the n_s most likely scenarios. This procedure leads to considerable time saving as compared to expanding the whole tree, whose number of nodes increases exponentially with N_s, as $2n_\xi^{N_s} - 1$. Algorithm 1 describes the scenario tree generation more in detail. An example of partially calculated scenario tree for the simpler case $n_\xi = n_m = N_s = 3$ can be found in Fig. 4.

6.2. Robot modeling

The joint angles of the robot, which constitute the state $\boldsymbol{\theta}$, were directly available for measurement via absolute rotary encoders (see Kinova Inc. (2022)). The space occupied by the robot was covered by $n_r = 7$ spheres $\mathcal{S}_{r,i}$, centered at the joints and end-effector positions, with radii $R_{r,i}$, from $i = 1$ to $i = 7$, equal, in meters, to 0.12, 0.12, 0.12, 0.12, 0.06, 0.06, and 0.10. The functions defining forward and differential kinematics were constructed from the Denavit-Hartenberg parameters (see, e.g., Siciliano et al. (2010)) provided by the robot manufacturer in the user guide (Kinova Inc., 2022). Different values of $\boldsymbol{\theta}_g$ were defined in sequence, with the manipulator always moving – using the same motion planning algorithm – from one goal configuration to the next, as follows: (i) the robot would be driven, with downward orientation, about 15 cm above a cube to be picked; (ii) $\boldsymbol{\theta}_g$ would be changed so that the robot end effector would move to actually pick up the cube; (iii) the robot would come back to the previous location, now holding the cube; (iv) the robot would move about 15 cm on top of the location where the cube had to be placed, located about 90 cm far from the picking location; (v) the robot would move close to the table and release the cube; (vi) the robot would be moved back to the previous value of

Algorithm 1: Construction of the scenario tree

Input: root node *root*, MTD model *mtd*, N_s, n_s
Output: array of most likely leaf nodes *sequences*
initialize *heap* as max heap with the *root* node as the only element;
set $depth = 0$;
set $prob = 1.0$;
set *sequences* as an empty array ;
set *current* = *root* ;
while $sequences.length < n_s$ **do**
  if $depth = N_s$ **then** // the horizon is reached
    append *current* to the *sequences* ; // add this node to the results
  else // expand the next node
    for $child \in mtd.predict(current)$ **do**
      push *child* to *heap* ;
      add *child* as descendent of *current* ;
    end
  end
  // pick a node with the largest probability
  set *current* = pop top from *heap* ;
end
return *sequences* ;

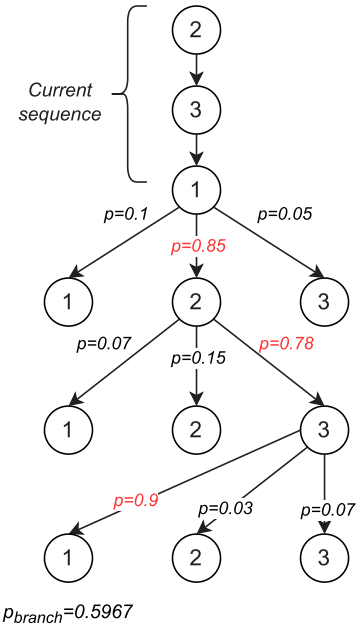


Fig. 4. An example of partially calculated scenario tree for $n_\xi = n_m = N_s = 3$. The first three nodes marked "current sequence" are the observed states that are used for prediction. Each node has the same number of children, equal to n_ξ. Edges show the probability of the next state given the previously-observed or predicted sequence of n_m states. An example of a calculated probability of a branch is shown in red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

θ_g. Then, the whole sequence will be repeated for picking and placing another cube.

As for the bounds on joint variables introduced in Section 3.3, Ω was defined limiting each component of $\boldsymbol{\omega}$ in the $[-0.8, 0.8]$ rad/s interval. Set Θ was instead defined by imposing, respectively (in rad), -2π, -2.25, -2π, -2.58, -2π, -2.10 and -2π as lower bounds, and 2π, 2.25, 2π, 2.58, 2π, 2.10 and 2π as upper bounds on each component of $\boldsymbol{\theta}$. The constraints in (10) were constructed to avoid collisions with the table, as

$$\begin{bmatrix} 0 & 0 & 1 \end{bmatrix} p_{r,i} \geq R_{r,i},$$

as the table surface had the same height as the robot base, in which the origin of the $O - xyz$ frame was located. As for safety constraints, the maximum human speed was set as $v_h = 1.6$ m/s, the maximum robot deceleration as $a_r = 5$ m/s^2, the precision of the motion capture system as $\epsilon = 0.02$ m and the detection-reaction time of the robot as $T_{dr} = 100$ ms. The latter value was determined based on the fact that the I-NMPC sampling interval was equal to 50 ms, and it is reasonable to choose T_{dr} equal to twice this value. The parameters for the calculation of the robot speed bounds in (12) based on the above-mentioned SSM parameters were set as $\alpha = 0.85$ and $\bar{d} = 0.21$.

6.3. S-NMPC definition and implementation

The cost function of the S-NMPC law was defined by the following parameters:

$$Q = \mathrm{diag}\left(\begin{bmatrix} 20 & 20 & 15 & 15 & 10 & 10 & 10 \end{bmatrix}\right),$$

$$R = \mathrm{diag}\left(\begin{bmatrix} 1.4 & 1.4 & 1.4 & 1.4 & 1.4 & 1.4 & 1.4 \end{bmatrix}\right),$$

$\gamma = 2200$ $\beta = 6.0$, $\rho = 10^5$ and $n_q = 2$, with the values of $\delta_h^{(\sigma)}$ defined with respect to the location of the human right hand.

The overall motion planning scheme of Fig. 1 was defined using the above-mentioned I-NMPC control law with sampling interval $\tilde{T}_s = 50$ ms and prediction horizon $\tilde{N} = 10$, corresponding to 500 ms, which tracked the reference variables generated by S-NMPC using the same parameters for cost function and constraints. The inner control loops of the Kinova Gen3 robot would then generate the needed joint torques.

The described human-prediction and NMPC routines were run via Robot Operating System (ROS). The NMPC controllers, obtained via the acados (Verschueren et al., 2021) C-language code generation tool, were implemented as separate C++ nodes in ROS. The S-NMPC parameters were defined using the acados MATLAB interface, and the nonlinear program into which the FHOCP was cast was solved via sequential quadratic programming, using the interior-point HPIPM solver (Frison & Diehl, 2020) with full condensing option to solve each quadratic program. All calculations relative to the human motion prediction via HOMC were implemented in Python in a single ROS node. Every 50 ms, the joint velocity commands generated by the S-NMPC/I-NMPC cascade were sent from the computer to the inner control loop of the Kinova Gen3 robot via ROS-Kortex driver. The joint positions and velocities of the robot were instead read via the ROS-Kortex interface, also every 50 ms.

7. Experimental results

7.1. Performance with a human subject

The proposed S-NMPC scheme was implemented for real-time interaction with human participants in cases $n_s = 1$ (i.e., only accounting for the most likely human prediction) and $n_s = 2$, referred to in the following as S-NMPC-1 and S-NMPC-2, respectively. As will be described in Section 7.2, higher value of n_s did not provide an acceptable performance in practice, mainly due to the increase of computational complexity. To compare them with other approaches, two additional motion planning routines were implemented. In the first one, the S-NMPC block in Fig. 1 was substituted by an NMPC planner with the same structure and the same parameters (in terms of cost function and constraints) as S-NMPC-1, but with the important difference that no HOMC was used to predict the time evolution of the human pose, which was instead set as constant and equal to the currently measured value. Assuming a constant human pose is probably the most reasonable solution if one does not possess prediction tools such as the

proposed HOMC-based method; indeed, solutions such as extrapolating the current speed of the human sphere centers would not work for prediction horizons corresponding to several seconds. This approach assuming constant human pose, referred to in the following as NP-NMPC, was first proposed by Oleinikov et al. (2021). The second approach is a continuous SSM (CSSM) planner (Marvel, 2013; Marvel & Norcross, 2017), in which both S-NMPC and I-NMPC are removed. In CSSM the robot follows the path determined by the NP-NMPC planner in the absence of the human (which approximately coincides with the S-NMPC-1 and S-NMPC-2 paths in the same condition), and continuously modulates the robot speed to satisfy condition (11). This method has the advantage of using the non-conservative SSM condition (11), but the disadvantage of not being able to change its path when the human moves close to the robot. CSSM can be considered as an advanced version of the fixed-path algorithms for safe pHRI used in industrial practice.

Although all four planning methods could run in real time in the presence of the operator, it would have been unfair to test each method on a different sequence of human motions. As a consequence, several human motions of the duration of ten minutes were recorded for the described task, and later provided to all of the four methods. The HOMC-based human prediction routine, when present, was also run in real time. For each of these sequences, the performance of the robot was measured with different metrics explained in the following.

The first metric is the average value of the sums of the FHOCP stage costs at each sampling interval, based on the measured robot and human poses, for a whole sequence of motions as described in Section 6.2. This metric, referred to as J_{avg}, expresses how well the robot optimizes the given cost function on the actual motion. CSSM did not optimize the same cost function in real time, and thus could not be compared with the other algorithms via this metric.

The average time T_{avg} that the robot took to move from a cube location on one side of the human to a cube location on the other side was used as second metric. This can correspond either to a pick-and-place operation (from when a cube is grasped until when it is released), or to the motion to reach the next cube after having released the previous one. During this time interval the robot would reach three different values of θ_g in sequence (above the current cube location, above the next cube location, and the next cube location itself). As $T_{\text{avg}} \simeq 12$ s for the NMPC controllers (as can be seen in the data presented in the following of this section), a prediction horizon corresponding to 5 s for reaching each value of θ_g turned out to be a reasonable choice; consider also that, in case the human operator obstructed the robot path, the NMPC planner could always decide to violate the soft SSM constraints (which would be then imposed by I-NMPC as hard constraints to guarantee safety) in order to achieve a feasible solution. A third and closely related metric is the robot productivity for the considered method, defined by Marvel (2013) as $P_{\text{avg}} \triangleq T'_{\text{avg}}/T_{\text{avg}}$, where T'_{avg} is defined as for T_{avg}, but without human presence. As slightly different values of T'_{avg} were obtained for each planning method, P_{avg} was calculated based on that specific value to be totally fair, and to exactly account for the loss of productivity with respect to using the same algorithm without human.

Experiments were executed in which all four algorithms were run for the same 12 motions of a single human participant. The results are reported in Table 1, from which one can see that S-NMPC-1 and S-NMPC-2 improved the average J_{avg} compared to NP-NMPC of about 3.9% and 15.0%, respectively. This shows that the use of two scenarios considerably improves the actual minimization of the cost function compared to using only one scenario, or no predictions at all. As for T_{avg} and P_{avg}, it is immediately clear that all NMPC-based planners outperform CSSM – for instance, a 26.4% improvement is observed for S-NMPC-2 – thanks to their ability to re-plan the robot path as well as its speed. In terms of T_{avg}, with respect to NP-NMPC, using S-NMPC-2 led to an improvement of about 4.3%, while using S-NMPC-1 caused a slight worsening of 0.3%. In terms of P_{avg} instead, compared to NP-NMPC, S-NMPC-2 improved the performance only of 0.59% (due to the fact that S-NMPC-2 was already faster to complete the task without human compared to NP-NMPC), whereas S-NMPC-1 worsened the performance of 0.6%. Overall, it is apparent that using one scenario in this case study does not give any advantage in terms of T_{avg} and P_{avg} and a relatively small advantage in terms of J_{avg}; using S-NMPC-2 improves these metrics, but the improvement is much lower than that observed for J_{avg}, which is indeed tightly related to the cost function actually minimized by the planners.

It might be interesting to analyze the robot behavior when directly using I-NMPC for regulation purposes, substituting the term in (25) with $\|\boldsymbol{\zeta}(k_\tau) - \boldsymbol{\theta}_g\|^2_{\boldsymbol{Q}}$. No stabilizing terminal constraint was inserted, as its satisfaction would not have been possible with the given prediction horizon of only 500 ms. Even though such an approach did not provide any guarantees of closed-loop stability, the manipulator succeeded in completing a number of pick-and-place tasks. The robot followed a path close to that observed with CSSM, with frequent stops. Due to the absence of a reference to track included in the state term in (25), the robot would often approach the goal point with relatively high speed, thus oscillating around it with decreasing amplitude before eventually converging. This behavior was still observed after tuning the I-NMPC weights so as to minimize the observed value of T_{avg}. As a result, an average time T'_{avg} without human of 11.52 s was obtained, which is considerably higher than the corresponding value for CSSM, equal to 8.79 s. In the presence of the human, average values of $T_{avg} = 18.22$ s and $P_{avg} = 63.86\%$ were obtained. Notice that this value of P_{avg}, though higher than the corresponding CSSM value of 55.14%, accounts for performance reduction with respect to an already much higher value of T'_{avg}. These results show that using I-NMPC alone is not preferable to using CSSM, and that a long-term planner is necessary to properly adapt the robot motion to that of the human.

In order to give an intuitive representation of the properties of closed-loop stability and satisfaction of the SSM constraints, Fig. 5 shows the time evolution of the four motion planning algorithms for the same initial condition and with a sequence of 100 s of human motion. As can be seen in the figure, the distance to the goal – in this case represented only for the robot end effector and thus equal to $\|\boldsymbol{p}_{r,7} - \boldsymbol{\Phi}_{\text{fwd},7}(\boldsymbol{\theta}_g)\|$ – always converges to zero, before a new goal configuration is set. The speed, here also represented in Cartesian coordinates for the end effector, i.e., equal to v_7, is shown to be always upper bounded by the corresponding SSM limit from Eq. (12).

To illustrate the prediction of the robot motion for one time instant k in the experiments, Fig. 6 shows, for NP-NMPC, S-NMPC-1 and S-NMPC-2, the human pose prediction $\mathcal{H}(k_\tau)$ for $\tau = 0, 2, 4, 6, 8, 10$ (represented as in Fig. 3) and the corresponding robot pose described by $\boldsymbol{\theta}_*(k_\tau)$. In the case of S-NMPC-2, two different scenarios $\sigma = 1$ and $\sigma = 2$ are generated with the corresponding values of $\mathcal{H}^{(1)}(k_\tau)$ and $\mathcal{H}^{(2)}(k_\tau)$, which in turn, through the FHOCP solution, generate $\boldsymbol{\theta}^{(1)}_*(k_\tau)$ and $\boldsymbol{\theta}^{(2)}_*(k_\tau)$.

A video is provided as supplementary material with this paper to provide an overview of the behavior of the four implemented algorithms on the Kinova Gen3 robot. A screenshot of the video is provided in Fig. 7. To always show an actual operator interacting with the robot in the video, it was impossible to replicate exactly the same human motion for all four algorithms. Therefore, the video – whose data were not used to generate the results shown in the tables – only provides a general idea of the behavior of each algorithm, and its content should not be intended as a proof of the superiority of S-NMPC-2 over the other algorithms. The human operator of the video is the same person shown in Figs. 2 and 7, and is one of the authors of this paper. On the contrary, the comparison between different algorithms reported in Table 1 was made on the same prerecorded human data (with a human operator who is not an author of the paper) and is therefore fair.

Table 1
Comparison of the different algorithms via three different metrics. The values of T_{avg} and T'_{avg} are expressed in seconds, while the values of J_{avg} and P_{avg} are dimensionless.

		Motion												Average
		1	2	3	4	5	6	7	8	9	10	11	12	
CSSM	J_{avg}	–	–	–	–	–	–	–	–	–	–	–	–	–
$T'_{avg} = 8.79$	T_{avg}	18.03	15.36	13.54	17.23	17.34	15.35	14.76	18.70	17.33	15.41	13.21	17.19	16.12
	P_{avg}	48.73%	57.18%	64.87%	50.98%	50.68%	57.27%	59.55%	47.01%	50.72%	57.04%	66.52%	51.13%	55.14%
NP-NMPC	J_{avg}	2238.03	2081.21	1607.57	1717.43	1855.56	2315.04	1851.88	1824.67	2170.35	2160.86	1965.77	2143.24	1994.30
$T'_{avg} = 8.68$	T_{avg}	12.47	13.13	11.66	12.30	12.49	12.62	12.04	12.30	12.63	12.39	12.26	12.40	12.39
	P_{avg}	69.58%	66.06%	74.39%	70.54%	69.48%	68.76%	72.10%	70.57%	68.74%	70.07%	70.79%	70.00%	70.09%
S-NMPC-1	J_{avg}	1939.46	2141.78	1651.23	1787.38	1871.85	2138.20	1873.24	1868.26	1935.28	1944.26	1881.93	1957.65	1915.88
$T'_{avg} = 8.66$	T_{avg}	12.29	13.33	12.06	12.55	12.30	12.75	12.04	12.27	12.64	12.32	12.26	12.38	12.43
	P_{avg}	70.47%	64.94%	71.80%	69.00%	70.41%	67.93%	71.94%	70.58%	68.54%	70.30%	70.65%	69.98%	69.71%
S-NMPC-2	J_{avg}	1493.83	1517.41	1391.50	1674.12	1927.57	1932.25	1998.13	1670.72	1496.41	1945.01	1718.23	1576.85	1695.17
$T'_{avg} = 8.35$	T_{avg}	11.78	12.12	10.78	11.52	12.50	11.81	12.12	11.76	11.84	12.50	12.14	11.46	11.86
	P_{avg}	70.86%	68.88%	77.45%	72.47%	66.80%	70.72%	68.90%	71.00%	70.51%	66.78%	68.76%	72.89%	70.50%

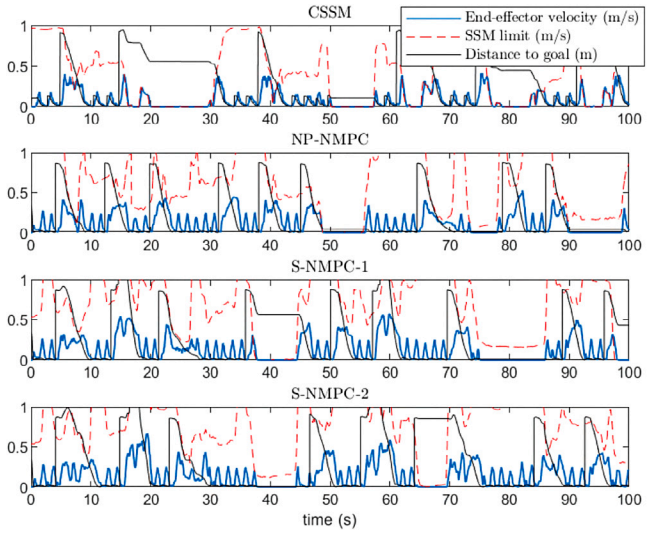


Fig. 5. Time evolution from the experiments of end-effector Cartesian velocity, corresponding SSM limit and distance of the end effector from its goal location in the presence of the human operator.

Table 2
Average time for calculation of human position prediction (ms).

		N_s									
		1	2	3	4	5	6	7	8	9	10
n_s	1	1.10	1.71	2.19	2.72	3.22	3.85	4.43	6.14	6.88	7.98
	2	1.19	2.32	3.56	4.64	5.65	7.30	8.95	10.07	13.77	17.15
	3	1.40	2.54	4.21	5.93	7.57	10.21	12.31	13.89	18.10	22.40
	4	1.39	3.00	4.73	6.80	9.52	12.03	15.22	17.21	22.25	25.56

7.2. Computational complexity

In order to get a better insight into the computation time of the different components of the developed algorithms, in this section the time intervals needed (using the hardware described in Section 6) to calculate the human pose prediction (via Algorithm 1) and the S-NMPC solution are reported. Using S-NMPC-2, a sequence of robot configurations was generated based on one of the human motions described above, collecting a total of 2317 samples. Based on these configurations $\theta(k)$ and the related human poses $\mathcal{H}(k)$ at the same time instants, human pose predictions and S-NMPC solutions were calculated offline for a range of parameter values. Offline calculations allowed us to test the devised methods on the same dataset (to allow a fair comparison), and to avoid problems related to real-time execution (for example, if the S-NMPC controller would require too much computation time).

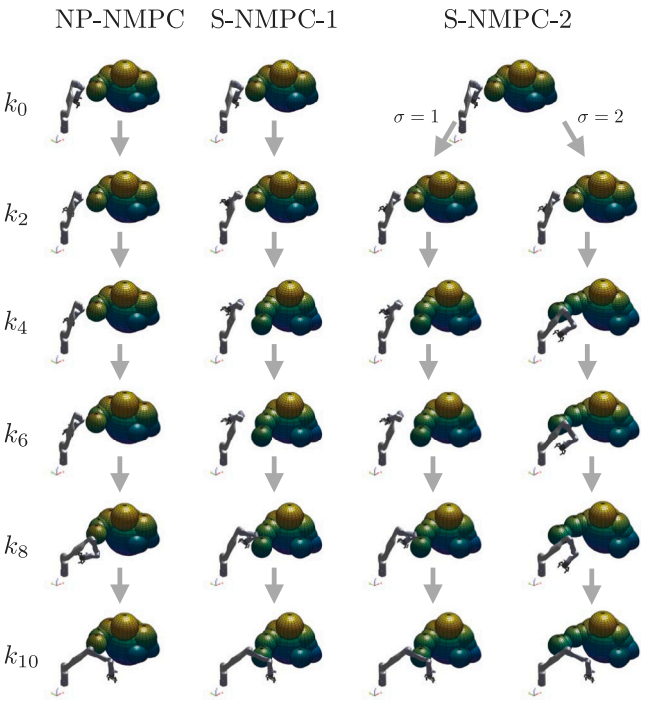


Fig. 6. Prediction of human motion and corresponding optimal robot motion (possibly for multiple scenarios) for the three implemented NMPC motion planners. These predictions correspond to the FHOCP solution for the same time instant k during the experiments.

Fig. 7. Screenshot of the video provided as supplementary material, which shows the robot behavior for the four algorithms (CSSM, NP-NMPC, S-NMPC-1, S-NMPC-2) executed in the experiments.

Table 3
Average S-NMPC calculation time (ms).

		N									
		1	2	3	4	5	6	7	8	9	10
NMPC	NP-NMPC	1.02	2.15	3.62	5.25	7.70	10.62	13.95	18.04	22.23	28.07
	S-NMPC-1	1.03	2.13	3.65	5.31	7.80	10.51	13.76	17.85	21.68	27.92
	S-NMPC-2	2.60	6.03	11.05	18.73	32.15	52.67	80.36	107.66	149.53	178.11
	S-NMPC-3	4.29	10.89	23.35	47.31	91.44	144.56	207.43	286.19	378.27	497.89
	S-NMPC-4	6.06	17.12	44.76	97.01	174.60	297.14	419.18	597.42	850.39	1171.98

The prediction of the human motion was obtained for a number of scenarios $n_s \in \mathbb{N}_{[1,4]}$ and for a time horizon $N_s \in \mathbb{N}_{[1,10]}$. The corresponding results in terms of average computation time are reported in Table 2. The computation time even for the most complex case is equal to 25.56 ms, which is relatively low if compared with the sampling interval $T_s = 500$ ms. The fact that the prediction of the human motion after 3 s (i.e., for $n_s > 6$) did not provide any reliable information was used to decide to avoid using higher values of N_s, which would have increased the computation time without providing any advantages.

The S-NMPC solution was calculated for the case without human prediction (NP-NMPC) and for a number of scenarios up to four (S-NMPC-4), using a prediction horizon $N \in \mathbb{N}_{[1,10]}$. The resulting average computation times are shown in Table 3. It is apparent that the computation time increases with N and even more notably with the number of scenarios. It is important to highlight that the cases NP-NMPC and S-NMPC-1 have very similar computation time for the same value of N, as a single evolution of the robot motion is calculated in both cases. Calculating the NMPC law requires on average more time than predicting the human motion via HOMC. A high value of N was necessary for S-NMPC to plan a reasonable robot motion that could reach the goal point (sometimes violating the soft SSM constraints, which were in any case enforced by I-NMPC), and this is why $N = 10$ was chosen. Also, the S-NMPC variants with three and four scenarios, due to their computation time that often exceeded the sampling interval, could not be used in real time without causing a considerable loss of performance and the frequent inability of the robot to reach the goal configuration in practice; this is the reason why, in Section 7.1, S-NMPC algorithms more complex than S-NMPC-2 were not considered.

8. Conclusions and outlook

This paper has proposed an S-NMPC method for pHRI, based on HOMC for human motion prediction. The closed-loop stability of the system was guaranteed under suitable assumptions, and the effectiveness of the approach was tested experimentally with actual human participants. As a result, the implemented S-NMPC scheme with two scenarios outperformed all other implemented schemes, i.e., S-NMPC with one scenario, NMPC without human prediction, and continuous SSM. This shows the potential of stochastic NMPC planners in pHRI.

The limitations of the proposed approach are mainly related to the computational burden of the S-NMPC calculations, which made it impossible to use more than two scenarios in real-time implementations. Although computational complexity is an inherent limitation of NMPC, its improvement will be considered as future work, together with implementations with multiple human participants in more realistic manufacturing case studies. Another topic to be investigated is the guarantee of stability based on assumptions on the human motion. Indeed, at the moment no assumptions are made on how $\mathcal{H}$ can vary from time k to $k+1$; specific human prediction methodologies could be used to provide probabilistic information on this variation, which could be exploited by a specifically designed NMPC strategy. Furthermore, considering set terminal constraints rather than point terminal constraints, or no terminal constraints at all with a sufficiently long prediction horizon, might be helpful to further improve the performance of the designed motion planning schemes.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by Nazarbayev University under Collaborative Research Project no. 091019CRP2118.

Appendix A. Proof of Theorem 1

After determining the optimal set of control sequences $\bar{\boldsymbol{\omega}}_*(k)$, one can define sub-optimal sequences of inputs $\tilde{\bar{\boldsymbol{\omega}}}(k+1)$ and corresponding states $\tilde{\bar{\boldsymbol{\theta}}}(k+1)$ for the next time instant, whose elements are defined, for all $\sigma \in \mathbb{N}_s$, as

$$\tilde{\boldsymbol{\omega}}^{(\sigma)}(k_\tau^+) \triangleq \begin{cases} \boldsymbol{\omega}_*^{(\hat{\sigma})}(k_\tau), & \tau \in \mathbb{N}_{[1,N-1]}, \\ 0 & \tau = N. \end{cases} \tag{A.1}$$

and

$$\tilde{\boldsymbol{\theta}}^{(\sigma)}(k_\tau) \triangleq \begin{cases} \boldsymbol{\theta}_*^{(\hat{\sigma})}(k_\tau), & \tau \in \mathbb{N}_{[1,N]}, \\ \boldsymbol{\theta}_g, & \tau = N+1, \end{cases} \tag{A.2}$$

where $\hat{\sigma} \in \mathbb{N}_s$ is fixed, e.g., $\hat{\sigma} = 1$. This set of sequences of control inputs, together with the corresponding sequences of states, leads to satisfying constraints (22a)–(22g), as one can immediately verify: indeed, these constraints do not depend on the human motion associated with a specific scenario, which is why one can choose any $\hat{\sigma} \in \mathbb{N}_s$. On the other hand, constraints (22h)–(22i) can always be satisfied. The existence of the feasible solution associated with (A.1), (A.2) implies the existence of an optimal solution at $k+1$.

Appendix B. Proof of Theorem 2

The optimal control move $\boldsymbol{\omega}(k)$ obtained from (23) for a given measured robot configuration $\boldsymbol{\theta}(k)$ also depends on the whole set $\mathcal{H}(k)$ of human motion predictions. In general, the applied control input can be written as $\boldsymbol{\omega}(\boldsymbol{\theta}(k), \mathcal{H}(k))$, or, using a more general notation for time-varying systems, as $\boldsymbol{\omega}(\boldsymbol{\theta}, k)$. The closed-loop dynamics (7) will be referred to as

$$\boldsymbol{\theta}^+ = f(\boldsymbol{\theta}, k) \triangleq \boldsymbol{\theta} + T_s \boldsymbol{\omega}(\boldsymbol{\theta}, k), \tag{B.1}$$

with $\boldsymbol{\omega}(\boldsymbol{\theta}, k)$ again obtained as solution of (23). The set of values of $\boldsymbol{\theta}$ for which a solution of the FHOCP (23) with prediction horizon N exists is denoted as Θ^N_{MPC}; this set is not time-varying, as all the imposed hard constraints are time-invariant, and thus a change in $\mathcal{H}(k)$ does not affect feasibility.

From Lyapunov theory (see, e.g., Theorem 2.32 in the book by Rawlings, Mayne, and Diehl (2017)) it is known that $\boldsymbol{\theta} = \boldsymbol{\theta}_g$ is a uniformly asymptotically stable equilibrium of system (B.1) with domain of attraction Θ^N_{MPC} if there exists a function (Lyapunov function) $V(\boldsymbol{\theta}, k) : \mathbb{R}^{n_\theta} \times \mathbb{N}_{\geq 0} \to \mathbb{R}_{\geq 0}$ defined such that, if $\boldsymbol{\theta} \in \Theta^N_{\text{MPC}}$,

$$\alpha_1(\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g}) \leq V(\boldsymbol{\theta}, k) \leq \alpha_2(\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g}), \tag{B.2}$$

$$V(f(\boldsymbol{\theta},k),k+1)-V(\boldsymbol{\theta},k)\leq-\alpha_3(\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g}), \tag{B.3}$$

where $\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g}\triangleq\|\boldsymbol{\theta}-\boldsymbol{\theta}_g\|$, whereas $\alpha_i(\cdot):\mathbb{R}_{\geq0}\to\mathbb{R}_{\geq0}$, $i=1,2,3$, are $\mathcal{K}_\infty$ functions (i.e., for $i=1,2,3$, $\alpha_i(0)=0$, $\alpha_i(\cdot)$ is strictly increasing and unbounded above). As Lyapunov function candidate, consider the optimal cost

$$V(\boldsymbol{\theta},k)\triangleq\min_{\bar{\boldsymbol{\omega}},\bar{\boldsymbol{\theta}}}J\left(\bar{\boldsymbol{\omega}}(k),\bar{\boldsymbol{\theta}}(k),\boldsymbol{\mathcal{H}}(k)\right),\ \text{subj. to (22a)–(22i).}$$

Determining $\alpha_1(\cdot)$

From (13), (19), (21), and recalling that $\sum_{\sigma=1}^{n_s}P_n^{(\sigma)}=1$ by construction, it is possible to obtain

$$\begin{aligned}&\sum_{\sigma=1}^{n_s}P_n^{(\sigma)}J^{(\sigma)}\left(\bar{\boldsymbol{\omega}}_*^{(\sigma)}(k),\bar{\boldsymbol{\theta}}_*^{(\sigma)}(k),\boldsymbol{\mathcal{H}}^{(\sigma)}(k)\right)\\&\geq\sum_{\sigma=1}^{n_s}P_n^{(\sigma)}\sum_{\tau=0}^{N-1}\|\boldsymbol{\theta}_*^{(\sigma)}(k_\tau)-\boldsymbol{\theta}_g\|_{\boldsymbol{Q}}^2\geq\sum_{\sigma=1}^{n_s}P_n^{(\sigma)}\|\boldsymbol{\theta}_*^{(\sigma)}(k|k)-\boldsymbol{\theta}_g\|_{\boldsymbol{Q}}^2\\&=\sum_{\sigma=1}^{n_s}P_n^{(\sigma)}\|\boldsymbol{\theta}(k)-\boldsymbol{\theta}_g\|_{\boldsymbol{Q}}^2\geq\lambda_{\min}(\boldsymbol{Q})\|\boldsymbol{\theta}(k)\|_{\boldsymbol{\theta}_g}^2\triangleq\alpha_1(\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g}).\end{aligned}$$

Determining $\alpha_2(\cdot)$

The basic idea is to find a subset of a neighborhood of $\boldsymbol{\theta}_g$ in which all hard state constraints are satisfied. In this subset, an explicit expression for the control action should be defined such that $\boldsymbol{\theta}=\boldsymbol{\theta}_g$ at the next time instant, i.e., $\boldsymbol{\omega}=\boldsymbol{\omega}_g\triangleq\frac{1}{T_s}(\boldsymbol{\theta}_g-\boldsymbol{\theta})$. The above-mentioned subset should be defined such that $\bar{\boldsymbol{\omega}}=\boldsymbol{\omega}_g$ can be applied satisfying both the input constraints and the SSM constraints (in the last case, with zero value of the slack variables). This will allow us to define an upper bounding function in this subset, which will be then extended to the whole set Θ_{MPC}^N.

Set Ω introduced in Section 3.3, which defines the input constraint (22e), is by definition a compact set that includes the origin in its interior. It is thus possible to determine a value $d_\omega\in\mathbb{R}_{>0}$, such that

$$\mathcal{B}(\boldsymbol{\omega},\mathbf{0},d_\omega)\triangleq\left\{\boldsymbol{\omega}\in\mathbb{R}^{n_\theta}\ :\ \|\boldsymbol{\omega}\|<d_\omega\right\}\subset\Omega, \tag{B.4}$$

where in general $\mathcal{B}(\boldsymbol{v},\bar{\boldsymbol{v}},\rho)$ is the open ball of radius ρ in the Euclidean space defined by the input vector $\boldsymbol{v}$, and centered at $\boldsymbol{v}=\bar{\boldsymbol{v}}$. Moving to the two hard state constraints, one can notice that both (22f) and (22g) can be expressed as depending on $\boldsymbol{\theta}$, since $\boldsymbol{p}_{r,i}=\boldsymbol{\Phi}_{\text{fwd,i}}(\boldsymbol{\theta})$. Thus, one can define $\tilde{\Theta}\subset\mathbb{R}^{n_\theta}$ as the set of values of $\boldsymbol{\theta}$ that satisfy both conditions (22f) and (22g). From the considerations in the paragraph after Eq. (10), it immediately follows that $\boldsymbol{\theta}_g\in\tilde{\Theta}$. Moreover, given Assumption 1, there exists $\tilde{d}_\theta\in\mathbb{R}_{>0}$ such that, if $\boldsymbol{\theta}\in\mathcal{B}(\boldsymbol{\theta},\boldsymbol{\theta}_g,\tilde{d}_\theta)$, then applying $\boldsymbol{\omega}_g$ leads to satisfying constraint (22h) with $\varepsilon_{ij}=0$. Another set is introduced in the robot state space, i.e., $\mathcal{B}(\boldsymbol{\theta},\boldsymbol{\theta}_g,d_\theta)$, where $d_\theta=\min(\tilde{d}_\theta,d_\omega T_s)$. From the definitions of the previously introduced sets it follows that, if one chooses $\boldsymbol{\theta}\in\mathcal{B}(\boldsymbol{\theta},\boldsymbol{\theta}_g,d_\theta)\cap\tilde{\Theta}$, then applying $\boldsymbol{\omega}_g$ satisfies both $\boldsymbol{\omega}_g\in\mathcal{B}(\boldsymbol{\omega},\mathbf{0},d_\omega)$ and constraint (22h) with $\varepsilon_{ij}=0$. The stage cost associated with state $\boldsymbol{\theta}\in\mathcal{B}(\boldsymbol{\theta},\boldsymbol{\theta}_g,d_\theta)\cap\tilde{\Theta}$, input $\boldsymbol{\omega}_g$ and to a generic human pose $\mathcal{H}$ can be written (with a slight abuse of notation and not considering any specific scenario σ), based on (13), (14) and (15) as

$$\begin{aligned}\ell(\boldsymbol{\theta},\boldsymbol{\omega}_g,\mathcal{H})&=\|\boldsymbol{\theta}-\boldsymbol{\theta}_g\|_{\boldsymbol{Q}}^2+\left\|\frac{1}{T_s}(\boldsymbol{\theta}_g-\boldsymbol{\theta})\right\|_{\boldsymbol{R}}^2+\gamma e^{-2\beta\frac{\delta_R^2}{\|\boldsymbol{\theta}-\boldsymbol{\theta}_g\|^2}}\\&\leq\left(\lambda_{\max}(\boldsymbol{Q})+\frac{\lambda_{\max}(\boldsymbol{R})}{T_s}\right)\|\boldsymbol{\theta}-\boldsymbol{\theta}_g\|^2+\gamma e^{-2\beta\frac{\delta_{\min}^2}{\|\boldsymbol{\theta}-\boldsymbol{\theta}_g\|^2}}\\&\triangleq\tilde{\alpha}_2(\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g}),\end{aligned} \tag{B.5}$$

with $\delta_{\min}$ as defined in (17). Notice that the portion of stage cost that accounts for the violation of the SSM constraints is not present, as applying $\boldsymbol{\omega}_g$ satisfies these constraints by construction. In (B.5), the case $\boldsymbol{\theta}\neq\boldsymbol{\theta}_g$ in (16) was taken into account: if $\boldsymbol{\theta}=\boldsymbol{\theta}_g$, then, again according to (16), the term multiplying γ is equal to zero, and the upper bound expressed by $\tilde{\alpha}_2(\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g})$ (which is a $\mathcal{K}_\infty$ function) is still valid. Assuming to solve the FHOCP (23) with $N=1$ from $\boldsymbol{\theta}\in\mathcal{B}(\boldsymbol{\theta},\boldsymbol{\theta}_g,d_\theta)\cap\tilde{\Theta}$ with generic human pose $\mathcal{H}(k)$, the only feasible solution, and thus coinciding with the optimal one, would be that corresponding to $\boldsymbol{\omega}_*^{(\sigma)}(k|k)=\boldsymbol{\omega}_g$ for all n_s scenarios. This leads to $V(\boldsymbol{\theta},k)\leq\tilde{\alpha}_2(\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g})$ for $N=1$. However, with the same initial condition $\boldsymbol{\theta}\in\mathcal{B}(\boldsymbol{\theta},\boldsymbol{\theta}_g,d_\theta)\cap\tilde{\Theta}$ and human pose $\mathcal{H}(k)$, but with $N>1$, the optimal value of the cost function cannot be worse than for $N=1$, and therefore $V(\boldsymbol{\theta},k)\leq\tilde{\alpha}_2(\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g})$ holds for any positive value of N. To determine an upper bound valid also for $\boldsymbol{\theta}\in\Theta_{\text{MPC}}^N\setminus\mathcal{B}(\boldsymbol{\theta},\boldsymbol{\theta}_g,d_\theta)$, one can proceed by defining

$$\begin{aligned}\hat{\alpha}_2(\nu)\triangleq\max_{\bar{\boldsymbol{\omega}},\bar{\boldsymbol{\theta}}}\ &J\left(\bar{\boldsymbol{\omega}}(k),\bar{\boldsymbol{\theta}}(k),\boldsymbol{\mathcal{H}}(k)\right)\\&\text{subj. to }\boldsymbol{\theta}\in\bar{\mathcal{B}}(\boldsymbol{\theta},\boldsymbol{\theta}_g,\nu)\\&\text{and to (22a)–(22b)}\end{aligned} \tag{B.6}$$

with $\bar{\mathcal{B}}(\boldsymbol{\theta},\boldsymbol{\theta}_g,\nu)\triangleq\left\{\boldsymbol{\theta}\in\mathbb{R}^{n_\theta}\ :\ \|\boldsymbol{\theta}-\boldsymbol{\theta}_g\|\leq\nu\right\}$ defined as a closed ball with a generic radius $\nu\in\mathbb{R}_{>0}$. Since both $f(\cdot,\cdot)$ and $J(\cdot,\cdot,\cdot)$ are continuous in their arguments and Ω is a compact set, similarly to Proposition 5.7(ii) in the book of Grüne and Pannek (2017) one can conclude that $\hat{\alpha}_2(\nu)$ is continuous and monotonically non-increasing (with, in general, $\hat{\alpha}_2(0)\neq0$), and easily claim that $V(\boldsymbol{\theta},k)\leq\hat{\alpha}_2(\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g})$. Again following the arguments in Proposition 5.7(ii) by Grüne and Pannek (2017), one can show that a suitable $\mathcal{K}_\infty$ upper-bounding function, as needed in (B.2), can be defined by combining $\tilde{\alpha}_2(\cdot)$ and $\hat{\alpha}_2(\cdot)$, as

$$\alpha_2(\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g})\triangleq\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g}\left(\tilde{\alpha}_2(\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g})+\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g}\hat{\alpha}_2(d_\theta)/d_\theta\right) \tag{B.7}$$

if $\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g}\in[0,d_\theta)$, and as

$$\alpha_2(\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g})\triangleq\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g}\left(\tilde{\alpha}_2(d_\theta)+\hat{\alpha}_2(\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g})\right) \tag{B.8}$$

if $\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g}\geq d_\theta$.

Determining $\alpha_3(\cdot)$

The optimal solution of (23) at time k associated with $\boldsymbol{\mathcal{H}}(k)$ provides sequences of control inputs for all scenarios $\sigma\in\mathbb{N}_s$ as $\boldsymbol{\omega}_*^{(\sigma)}(k_\tau)$, $\tau\in\mathbb{N}_{[0,N-1]}$ and corresponding robot configurations $\boldsymbol{\theta}_*^{(\sigma)}(k_\tau)$, $\tau\in\mathbb{N}_{[0,N]}$, with all $\boldsymbol{\theta}_*^{(\sigma)}(k_N)=\boldsymbol{\theta}_g$. These sequences, together with $\boldsymbol{\mathcal{H}}(k)$, determine the value of $V(\boldsymbol{\theta},k)$. At time $k+1$, one can employ the set of suboptimal sequences defined by (A.1) and (A.2), for a fixed $\hat{\sigma}\in\mathbb{N}_s$. These suboptimal sequences, together with $\boldsymbol{\mathcal{H}}(k+1)$, will be associated with a cost value $\tilde{V}(f(\boldsymbol{\theta},k),k+1)$. As the latter constitutes an upper bound for the optimal cost $V(f(\boldsymbol{\theta},k),k+1)$, uniform asymptotic stability will be proven if one can determine $\alpha_3(\cdot)$ such that

$$\Delta V\triangleq\tilde{V}(f(\boldsymbol{\theta},k),k+1)-V(\boldsymbol{\theta},k)\leq-\alpha_3(\|\boldsymbol{\theta}\|_{\boldsymbol{\theta}_g}). \tag{B.9}$$

$V(\boldsymbol{\theta},k)$ is composed of the sum of the optimal stage costs in (20), namely

$$\ell_*^{(\sigma)}(k_\tau)\triangleq e_{\theta*}^{(\sigma)}(k_\tau)+e_{\omega*}^{(\sigma)}(k_\tau)+e_{\varphi*}^{(\sigma)}(k_\tau)+e_{\varepsilon*}^{(\sigma)}(k_\tau). \tag{B.10}$$

Analogous terms for $\tilde{V}(f(\boldsymbol{\theta},k),k+1)$ are defined as

$$\tilde{\ell}^{(\sigma)}(k_\tau^+)\triangleq\tilde{e}_\theta^{(\sigma)}(k_\tau^+)+\tilde{e}_\omega^{(\sigma)}(k_\tau^+)+\tilde{e}_\varphi^{(\sigma)}(k_\tau^+)+\tilde{e}_\varepsilon^{(\sigma)}(k_\tau^+). \tag{B.11}$$

One can rewrite ΔV defined in (B.9) as

$$\Delta V=\sum_{\sigma=1}^{n_s}\sum_{\tau=1}^{N}\tilde{P}_n^{(\sigma)}\tilde{\ell}^{(\sigma)}(k_\tau^+)-\sum_{\sigma=1}^{n_s}\sum_{\tau=0}^{N-1}P_n^{(\sigma)}\ell_*^{(\sigma)}(k_\tau), \tag{B.12}$$

in which $P_n^{(\sigma)}$ and $\tilde{P}_n^{(\sigma)}$ are the normalized probabilities associated with a generic scenario $\sigma\in\mathbb{N}_s$ in $\boldsymbol{\mathcal{H}}(k)$ and $\boldsymbol{\mathcal{H}}(k+1)$, respectively. In the following, ΔV is split into several terms as $\Delta V=\sum_{\tau=0}^{N}\Delta V_\tau$, with each ΔV_τ including the sum of all stage costs in (B.12) associated with

time $k+\tau \mid k$. A sufficient condition implying the existence of $\alpha_3(\cdot)$ is $\Delta V_0 \leq -\alpha_3(\|\theta\|_{\theta_g})$, and $\Delta V_\tau \leq 0$ for $\tau \in \mathbb{N}_{1,N}$. In the remainder of the proof, it will be shown that Assumptions 2 and 3 imply these inequalities.

The first term to be analyzed is

$$\Delta V_N = \sum_{\sigma=1}^{n_s} \tilde{P}_n^{(\sigma)} \tilde{\ell}^{(\sigma)}(k_N^+). \tag{B.13}$$

For all scenarios, one has that $\tilde{\theta}^{(\sigma)}(k_N^+) = \theta_*^{(\hat{\sigma})}(k_N) = \theta_g$ (see (22c) and (A.1)) and $\tilde{\omega}^{(\sigma)}(k_N^+) = 0$ (see (A.2)). Therefore, the terms in (B.11) are as follows: $\tilde{e}_\theta^{(\sigma)}(k_N^+) = 0$ (see (13)); $\tilde{e}_\omega^{(\sigma)}(k_\tau^+) = 0$ (see (14)); $\tilde{e}_\varphi^{(\sigma)}(k_\tau^+) = 0$, again as $\tilde{\theta}^{(\sigma)}(k_N^+) = \theta_*^{(\hat{\sigma})}(k_N) = \theta_g$ (see (15) and (16)); finally, $\tilde{e}_\varepsilon^{(\sigma)}(k_\tau^+) = 0$ (see (18)), as Assumption 1 guarantees the satisfaction of the SSM constraints with zero speed. This immediately proves that condition $\Delta V_N \leq 0$ is satisfied.

The next considered term is the sum of all

$$\Delta V_\tau = \sum_{\sigma=1}^{n_s} \tilde{P}_n^{(\sigma)} \tilde{\ell}^{(\sigma)}(k_\tau^+) - \sum_{\sigma=1}^{n_s} P_n^{(\sigma)} \ell_*^{(\sigma)}(k_\tau), \quad \tau \in \mathbb{N}_{[1,N-1]}, \tag{B.14}$$

which is further split for convenience as $\Delta V_\tau = \Delta V_{\tau,\theta} + \Delta V_{\tau,\omega} + \Delta V_{\tau,\varphi} + \Delta V_{\tau,\varepsilon}$, with each sub-term containing the portions of stage costs related to the same subscript θ, ω, φ or ε. The first sum of sub-terms is

$$\sum_{\tau=1}^{N-1} \Delta V_{\tau,\theta} \triangleq \sum_{\tau=1}^{N-1} \sum_{\sigma=1}^{n_s} \tilde{P}_n^{(\sigma)} \tilde{e}_\theta^{(\sigma)}(k_\tau^+) - \sum_{\tau=1}^{N-1} \sum_{\sigma=1}^{n_s} P_n^{(\sigma)} e_{\theta*}^{(\sigma)}(k_\tau). \tag{B.15}$$

As $\tilde{\theta}^{(\sigma)}(k_\tau^+) = \theta_*^{(\hat{\sigma})}(k_\tau)$ by construction and $\sum_{\sigma=1}^{n_s} \tilde{P}_n^{(\sigma)} = 1$, the first inner sum on the right-hand side of (B.15) equals $e_{\theta*}^{(\hat{\sigma})}(k_\tau)$. Thus, $\sum_{\tau=1}^{N-1} \Delta V_{\tau,\theta} \leq 0$ can be obtained by imposing $\sum_{\tau=1}^{N-1} e_{\theta*}^{(\hat{\sigma})}(k_\tau) \leq \sum_{\tau=1}^{N-1} \sum_{\sigma=1}^{n_s} P_n^{(\sigma)} e_{\theta*}^{(\sigma)}(k_\tau)$ or, equivalently,

$$\sum_{\tau=1}^{N-1} \left(1 - P_n^{(\hat{\sigma})}\right) e_{\theta*}^{(\hat{\sigma})}(k_\tau) \leq \sum_{\tau=1}^{N-1} \sum_{\substack{\sigma=1 \\ \sigma \neq \hat{\sigma}}}^{n_s} P_n^{(\sigma)} e_{\theta*}^{(\sigma)}(k_\tau). \tag{B.16}$$

As for $\tau \in \mathbb{N}_{[1,n_q]}$ all state evolutions coincide, and thus $e_{\theta*}^{(\hat{\sigma})}(k_\tau) = e_{\theta*}^{(\sigma)}(k_\tau)$ for all $\sigma \in \mathbb{N}_s$, the sums in both sides of (B.16) only have to be imposed for $\tau \in \mathbb{N}_{n_q+1,N-1}$, as all other terms cancel each other. By also recalling the formulation of $P_{nn}^{(\sigma)}$ introduced in Assumption 2, a condition equivalent to (B.16) is

$$\left(1 - P_n^{(\hat{\sigma})}\right) \sum_{\tau=n_q+1}^{N-1} e_{\theta*}^{(\hat{\sigma})}(k_\tau) \leq \sum_{\substack{\sigma=1 \\ \sigma \neq \hat{\sigma}}}^{n_s} P_n^{(\sigma)} \cdot \sum_{\tau=n_q+1}^{N-1} \sum_{\substack{\sigma=1 \\ \sigma \neq \hat{\sigma}}}^{n_s} P_{nn}^{(\sigma)} e_{\theta*}^{(\sigma)}(k_\tau). \tag{B.17}$$

Since $\left(1 - P_n^{(\hat{\sigma})}\right) = \sum_{\sigma=1}^{n_s} P_n^{(\sigma)}$ by definition, (B.17) is equivalent to $\sum_{\tau=n_q+1}^{N-1} e_{\theta*}^{(\hat{\sigma})}(k_\tau) \leq \sum_{\tau=n_q+1}^{N-1} \sum_{\substack{\sigma=1 \\ \sigma \neq \hat{\sigma}}}^{n_s} P_{nn}^{(\sigma)} e_{\theta*}^{(\sigma)}(k_\tau)$. As in general $\lambda_{\min}(\boldsymbol{Q}) \left\|\theta_*^{(\hat{\sigma})}(k_\tau)\right\|_{\theta_g}^2 \leq e_{\theta*}^{(\sigma)}(k_\tau) \leq \lambda_{\max}(\boldsymbol{Q}) \left\|\theta_*^{(\hat{\sigma})}(k_\tau)\right\|_{\theta_g}^2$ regardless of the value of τ, it is now possible to see that (26) in Assumption 2 implies $\sum_{\tau=1}^{N-1} \Delta V_{\tau,\theta} \leq 0$. The second sum of sub-terms is

$$\sum_{\tau=1}^{N-1} \Delta V_{\tau,\omega} \triangleq \sum_{\tau=1}^{N-1} \sum_{\sigma=1}^{n_s} \tilde{P}_n^{(\sigma)} \tilde{e}_\omega^{(\sigma)}(k_\tau^+) - \sum_{\tau=1}^{N-1} \sum_{\sigma=1}^{n_s} P_n^{(\sigma)} e_{\omega*}^{(\sigma)}(k_\tau). \tag{B.18}$$

By following the same rationale used to analyze $\Delta V_{\tau,\theta}$, one can show that $\Delta V_{\tau,\omega} = 0$ for $\tau \in \mathbb{N}_{[1,n_q-1]}$, in which the control input is the same for all scenarios, and therefore (27) in Assumption 2 is needed to impose $\sum_{\tau=1}^{N-1} \Delta V_{\tau,\omega} \leq 0$. The third sum of sub-terms is

$$\sum_{\tau=1}^{N-1} \Delta V_{\tau,\varphi} \triangleq \sum_{\tau=1}^{N-1} \sum_{\sigma=1}^{n_s} \tilde{P}_n^{(\sigma)} \tilde{e}_{\varphi,\hat{\sigma}}^{(\sigma)}(k_\tau^+) - \sum_{\tau=1}^{N-1} \sum_{\sigma=1}^{n_s} P_n^{(\sigma)} e_{\varphi*}^{(\sigma)}(k_\tau), \tag{B.19}$$

where $\tilde{e}_{\varphi,\hat{\sigma}}^{(\sigma)}(k_\tau^+)$ is the value of $\tilde{e}_\varphi^{(\sigma)}(k_\tau^+)$ in which the state value $\tilde{\theta}^{(\sigma)}(k_\tau^+) = \theta_*^{(\hat{\sigma})}(k_\tau)$ corresponds to the suboptimal sequence (a single sequence corresponding to scenario $\hat{\sigma}$), whereas the human motion predicted at time $k+1$ comprises of n_s different scenarios σ. This is why the first inner sum in the right-hand side of (B.19) cannot be represented as a single term only related to scenario $\hat{\sigma}$, as was done for the previous two sums of sub-terms. Regardless of the values of the probabilities $P_n^{(\sigma)}$ and $\tilde{P}_n^{(\sigma)}$, recalling that the values of $e_{\varphi*}^{(\sigma)}(k_\tau)$ and $\delta_{h*}^{(\sigma)}(k_\tau)$ (and similarly $\tilde{e}_{\varphi,\hat{\sigma}}^{(\sigma)}(k_\tau^+)$ and $\tilde{\delta}_{h,\hat{\sigma}}^{(\sigma)}$) are both non-negative and inversely proportional to one another, it is possible to see that (28) in Assumption 3 implies $\sum_{\tau=1}^{N-1} \Delta V_{\tau,\varphi} \leq 0$. The fourth sum of sub-terms is

$$\sum_{\tau=1}^{N-1} \Delta V_{\tau,\varepsilon} \triangleq \sum_{\tau=1}^{N-1} \sum_{\sigma=1}^{n_s} \tilde{P}_n^{(\sigma)} \tilde{e}_{\varepsilon,\hat{\sigma}}^{(\sigma)}(k_\tau^+) - \sum_{\tau=1}^{N-1} \sum_{\sigma=1}^{n_s} P_n^{(\sigma)} e_{\varepsilon*}^{(\sigma)}(k_\tau). \tag{B.20}$$

Similarly to the third sum of sub-terms, one can see that (29) in Assumption 3 implies $\sum_{\tau=1}^{N-1} \Delta V_{\tau,\varepsilon} \leq 0$.

One can conclude by analyzing ΔV_0. Its expression can be inferred from (B.12) as

$$\Delta V_0 = -\sum_{\sigma=1}^{n_s} P_n^{(\sigma)} \ell_*^{(\sigma)}(k_0) \leq -e_{\theta*}^{(\tilde{\sigma})}(k_0), \tag{B.21}$$

where $\tilde{\sigma}$ is an arbitrary scenario value. The inequality in (B.21) is obtained by recalling that $\sum_{\sigma=1}^{n_s} P_n^{(\sigma)} = 1$ and that the initial robot state is the same for all scenarios. More in detail, as this state is equal to $\theta(k)$, then $\Delta V_0 \leq -\lambda_{\min}(\boldsymbol{Q})(\|\theta(k)\|_{\theta_g}) \triangleq -\alpha_3(\|\theta\|_{\theta_g})$, which concludes the proof.

Appendix C. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.conengprac.2023.105769.

References

Ajoudani, A., Zanchettin, A. M., Ivaldi, S., Albu-Schäffer, A., Kosuge, K., & Khatib, O. (2018). Progress and prospects of the human–robot collaboration. *Autonomous Robots*, *42*(5), 957–975.

Berchtold, A., & Raftery, A. (2002). The mixture transition distribution model for high-order Markov chains and non-Gaussian time series. *Statistical Science*, *17*(3), 328–356.

Bernardini, D., & Bemporad, A. (2009). Scenario-based model predictive control of stochastic constrained linear systems. In *Proc. IEEE conference on decision and control* (pp. 6333–6338).

Bernardini, D., & Bemporad, A. (2011). Stabilizing model predictive control of stochastic constrained linear systems. *IEEE Transactions on Automatic Control*, *57*(6), 1468–1480.

Calafiore, G. C., & Fagiano, L. (2013). Robust model predictive control via scenario optimization. *IEEE Transactions on Automatic Control*, *58*(1), 219–224.

Cannon, M., Kouvaritakis, B., Raković, S. V., & Cheng, Q. (2010). Stochastic tubes in model predictive control with probabilistic constraints. *IEEE Transactions on Automatic Control*, *56*(1), 194–200.

Chemweno, P., Pintelon, L., & Decre, W. (2020). Orienting safety assurance with outcomes of hazard analysis and risk assessment: A review of the ISO 15066 standard for collaborative robot systems. *Safety Science*, *129*, Article 104832.

Chen, Y., Rosolia, U., Ubellacker, W., Csomay-Shanklin, N., & Ames, A. D. (2022). Interactive multi-modal motion planning with branch model predictive control. *IEEE Robotics and Automation Letters*, *7*(2), 5365–5372.

de Groot, O., Brito, B., Ferranti, L., Gavrila, D., & Alonso-Mora, J. (2021). Scenario-based trajectory optimization in uncertain dynamic environments. *IEEE Robotics and Automation Letters*, *6*(3), 5389–5396.

Di Cairano, S., Bernardini, D., Bemporad, A., & Kolmanovsky, I. V. (2013). Stochastic MPC with learning for driver-predictive vehicle control and its application to HEV energy management. *IEEE Transactions on Control Systems Technology*, *22*(3), 1018–1031.

Ding, H., Reißig, G., Wijaya, K., Bortot, D., Bengler, K., & Stursberg, O. (2011). Human arm motion modeling and long-term prediction for safe and efficient human-robot-interaction. In *Proc. IEEE international conference on robotics and automation* (pp. 5875–5880).

Eckhoff, M., Kirschner, R. J., Kern, E., Abdolshah, S., & Haddadin, S. (2022). An MPC framework for planning safe & trustworthy robot motions. In *Proc. IEEE international conference on robotics and automation* (pp. 4737–4742).

Elfring, J., Van De Molengraft, R., & Steinbuch, M. (2014). Learning intentions for improved human motion prediction. *Robotics and Autonomous Systems*, *62*(4), 591–602.

Flowers, J., & Wiens, G. (2023). A spatio-temporal prediction and planning framework for proactive human-robot collaboration. *Journal of Manufacturing Science and Engineering*, *145*(12), Article 121011.

Frison, G., & Diehl, M. (2020). HPIPM: a high-performance quadratic programming framework for model predictive control. *IFAC-PapersOnLine*, *53*(2), 6563–6569.
Grüne, L., & Pannek, J. (2017). *Nonlinear model predictive control*. Springer.
Haddadin, S., & Croft, E. (2016). Physical human–robot interaction. In *Springer handbook of robotics* (pp. 1835–1874). Springer.
Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research*, *5*(1), 90–98.
Kim, K.-K. K., & Braatz, R. D. (2013). Generalised polynomial chaos expansion approaches to approximate stochastic model predictive control. *International Journal of Control*, *86*(8), 1324–1337.
Kim, E., Kirschner, R., Yamada, Y., & Okamoto, S. (2020). Estimating probability of human hand intrusion for speed and separation monitoring using interference theory. *Robotics and Computer-Integrated Manufacturing*, *61*, Article 101819.
Kinova Inc. (2022). User guide – KINOVA Gen3 ultra lightweight robot. URL https://www.kinovarobotics.com/uploads/User-Guide-Gen3-R07.pdf.
Krishnamoorthy, D., Suwartadi, E., Foss, B., Skogestad, S., & Jäschke, J. (2018). Improving scenario decomposition for multistage MPC using a sensitivity-based path-following algorithm. *IEEE Control Systems Letters*, *2*(4), 581–586.
Lèbre, S., & Bourguignon, P.-Y. (2008). An EM algorithm for estimation in the mixture transition distribution model. *Journal of Statistical Computation and Simulation*, *78*(8), 713–729.
Li, S., Zheng, P., Liu, S., Wang, Z., Wang, X. V., Zheng, L., et al. (2023). Proactive human–robot collaboration: Mutual-cognitive, predictable, and self-organising perspectives. *Robotics and Computer-Integrated Manufacturing*, *81*, Article 102510.
Liu, H., & Wang, L. (2017). Human motion prediction for human-robot collaboration. *Journal of Manufacturing Systems*, *44*, 287–294.
Lucia, S., Finkler, T., & Engell, S. (2013). Multi-stage nonlinear model predictive control applied to a semi-batch polymerization reactor under uncertainty. *Journal of Process Control*, *23*(9), 1306–1319.
Lucia, S., Subramanian, S., Limon, D., & Engell, S. (2020). Stability properties of multi-stage nonlinear model predictive control. *Systems & Control Letters*, *143*, Article 104743.
Luo, R., & Berenson, D. (2015). A framework for unsupervised online human reaching motion recognition and early prediction. In *Proc. IEEE/RSJ international conference on intelligent robots and systems* (pp. 2426–2433).
Marvel, J. A. (2013). Performance metrics of speed and separation monitoring in shared workspaces. *IEEE Transactions on Automation Science and Engineering*, *10*(2), 405–414.
Marvel, J. A., & Norcross, R. (2017). Implementing speed and separation monitoring in collaborative robot workcells. *Robotics and Computer-Integrated Manufacturing*, *44*, 144–155.
Mesbah, A. (2016). Stochastic model predictive control: An overview and perspectives for future research. *IEEE Control Systems Magazine*, *36*(6), 30–44.
Mesbah, A., Kolmanovsky, I. V., & Di Cairano, S. (2019). Stochastic model predictive control. In *Handbook of model predictive control* (pp. 75–97). Springer.
Nurbayeva, A., Shintemirov, A., & Rubagotti, M. (2023). Deep imitation learning of nonlinear model predictive control laws for safe physical human-robot interaction. *IEEE Transactions on Industrial Informatics*, *19*(7), 8384–8395.
Oleinikov, A., Kusdavletov, S., Shintemirov, A., & Rubagotti, M. (2021). Safety-aware nonlinear model predictive control for physical human-robot interaction. *IEEE Robotics and Automation Letters*, *6*(3), 5665–5672.
Pedersen, T. H., & Petersen, S. (2018). Investigating the performance of scenario-based model predictive control of space heating in residential buildings. *Journal of Building Performance Simulation*, *11*(4), 485–498.
Raftery, A. E. (1985). A model for high-order Markov chains. *Journal of the Royal Statistical Society. Series B. Statistical Methodology*, *47*(3), 528–539.
Ravichandar, H. C., & Dani, A. P. (2017). Human intention inference using expectation-maximization algorithm with online model learning. *IEEE Transactions on Automation Science and Engineering*, *14*(2), 855–868.
Rawlings, J. B., Mayne, D. Q., & Diehl, M. (2017). *Model predictive control: theory, computation, and design* (2nd ed.). Nob-Hill Publishing.
Rohrmuller, F., Althoff, M., Wollherr, D., & Buss, M. (2008). Probabilistic mapping of dynamic obstacles using Markov chains for replanning in dynamic environments. In *Proc. IEEE/RSJ international conference on intelligent robots and systems* (pp. 2504–2510).
Rosenstrauch, M. J., Pannen, T. J., & Krüger, J. (2018). Human robot collaboration-using kinect v2 for ISO/TS 15066 speed and separation monitoring. *Procedia CIRP*, *76*, 183–186.
Schildbach, G., Fagiano, L., Frei, C., & Morari, M. (2014). The scenario approach for stochastic model predictive control with bounds on closed-loop constraint violations. *Automatica*, *50*(12), 3009–3018.
Shang, C., & You, F. (2019). A data-driven robust optimization approach to scenario-based stochastic model predictive control. *Journal of Process Control*, *75*, 24–39.
Siciliano, B., et al. (2010). *Robotics: modeling, planning and control*. Springer.
Sung, C., Feldman, D., & Rus, D. (2012). Trajectory clustering for motion prediction. In *Proc. IEEE/RSJ international conference on intelligent robots and systems* (pp. 1547–1552).
Tonola, C., Faroni, M., Pedrocchi, N., & Beschi, M. (2021). Anytime informed path re-planning and optimization for human-robot collaboration. In *Proc. IEEE international conference on robot & human interactive communication* (pp. 997–1002).
Tusseyeva, I., Oleinikov, A., Sandygulova, A., & Rubagotti, M. (2022). Perceived safety in human–cobot interaction for fixed-path and real-time motion planning algorithms. *Scientific Reports*, *12*, 20438.
Vasquez, D., Fraichard, T., & Laugier, C. (2009). Incremental learning of statistical motion patterns with growing hidden Markov models. *IEEE Transactions on Intelligent Transportation Systems*, *10*(3), 403–416.
Verschueren, R., Frison, G., Kouzoupis, D., Frey, J., Duijkeren, N. v., Zanelli, A., et al. (2021). acados–a modular open-source framework for fast embedded optimal control. *Mathematical Programming Computation*, 1–37.
Wang, R., Schuurmans, M., & Patrinos, P. (2023). Interaction-aware model predictive control for autonomous driving. In *Proc. European control conference* (pp. 1–6).
Zanchettin, A. M., Casalino, A., Piroddi, L., & Rocco, P. (2019). Prediction of human activity patterns for human–robot collaborative assembly tasks. *IEEE Transactions on Industrial Informatics*, *15*(7), 3934–3942.
Zheng, P., Wieber, P.-B., & Aycard, O. (2020). Online optimal motion generation with guaranteed safety in shared workspace. In *Proc. IEEE international conference on robotics and automation* (pp. 9210–9215).