Safety-Aware Nonlinear Model Predictive Control for Physical Human-Robot Interaction

Artemiy Oleinikov, Sanzhar Kusdavletov, Almas Shintemirov, and Matteo Rubagotti

***Abstract*—This letter proposes a nonlinear model predictive control (NMPC) approach for real-time planning of point-to-point motions of serial robot manipulators that share their workspace with a human. The NMPC law solves a nonlinear program online, based on a kinematic model, and guarantees safety by constraining the robot speed within the time-varying bounds determined by the speed-and-separation-monitoring (SSM) principle. Closed-loop stability is proven in detail, and the performance (in terms of productivity) of the proposed method is tested against standard SSM schemes via experiments on a Kinova Gen3 robot.**

***Index Terms*—Optimization and optimal control, human-aware motion planning, physical human-robot interaction, nonlinear model predictive control, speed and separation monitoring.**

I. Introduction

PHYSICAL human-robot interaction (pHRI) has recently driven a large amount of research [1]. Safety (i.e., "ensuring that only mild contusions may occur in worst-case scenarios" [2]) is at the basis of the first standard for industrial applications of collaborative robots, namely ISO/TS 15066 ("Robots and Robotic Devices - Industrial Safety Requirements: Collaborative Industrial Robots") [3]. In this framework, safety can be achieved by using lightweight torque-controlled robots, and by adapting the robot speed to the presence of the human. For example, when using *speed and separation monitoring* (SSM), the robot reduces its speed according to the distance with the human [4]–[6]. To further enhance productivity, a current research line focuses on improvements of the standard SSM framework: see, e.g., [7] (which considers the direction of the robot motion, in addition to the separation distance, to determine the robot speed), and the references in it. Alternatively, other researchers are focusing on real-time motion planning methods for robot manipulators in the presence of humans (see, e.g., [1] and the references therein).

Thanks to faster microprocessors, improved optimization solvers, and ad-hoc toolboxes such as ACADO [8], model predictive control (MPC, see, e.g., [9]) has become a viable solution for the control of manipulators [10]–[15]. Regarding MPC used for motion planning in the presence of humans, the approaches proposed in [16], [17] were validated in simulation, while experimental results were obtained in [18], [19], focusing, respectively, on an MPC approach for human avoidance based on mixed-integer programming and on a nonlinear MPC (NMPC) scheme to coordinate human and robot motions during pick-and-place operations.

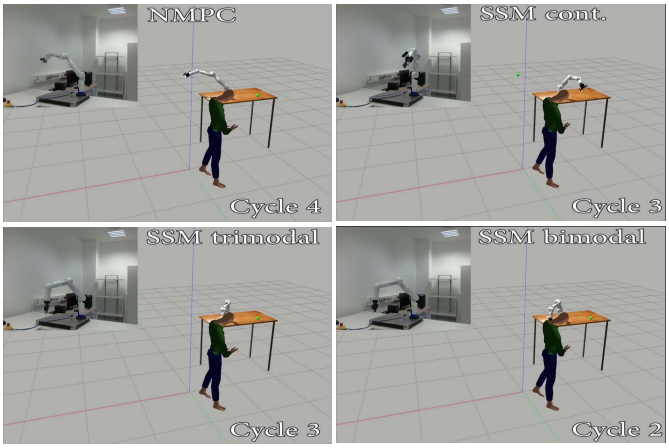

Fig. 1. Screenshot of the video provided as supplementary material, in which one can see the real robot (with its motion reproduced in the simulation) and the simulated human for our case study.

In this paper, we propose a real-time motion planning algorithm for pHRI based on NMPC. A serial manipulator has to independently complete a sequence of point-to-point motions, while a human carries out different tasks near the robot, with no a-priori assumptions on his/her motion. The proposed algorithm keeps replanning the robot motion based on the current human pose (measured via motion capture). It modulates the robot speed depending on the relative distance with the human, by satisfying the SSM specifications defined in [4]. The following two properties are guaranteed under suitable assumptions detailed in the remainder of the paper. First, the stability of the closed-loop system with respect to the current reference configuration is guaranteed. Second, thanks to the SSM-based speed modulation, the robot will stop before a contact occurs with the human.

This type of approach to safe NMPC based on SSM specifications, including theoretical proofs, constitutes a novelty, and is the first contribution of this paper. An additional contribution is in the experimental implementation of the proposed NMPC method, and of different SSM algorithms described in [4], on a Kinova Gen3 manipulator (see Fig. 1): the comparison shows that our NMPC algorithm for motion planning provides

Manuscript received: December 17th, 2020; Revised: April 2nd, 2021; Accepted: May 14th, 2021. This paper was recommended for publication by Editor Lucia Pallottino upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by Nazarbayev University collaborative research project no. 091019CRP2118 "Stochastic and Learning-Based Predictive Control Methods for Physical Human-Robot interaction".

The authors are with the Dept. of Robotics and Mechatronics, School of Engineering and Digital Sciences, Nazarbayev University, 010000 Nur-Sultan, Kazakhstan. Corresponding author: Matteo Rubagotti, email: matteo.rubagotti@nu.edu.kz.

Digital Object Identifier (DOI): see top of this page.

higher productivity than the SSM schemes in [4], at the same time ensuring safety. In this work, by *safety* we mean that the SSM velocity bounds are enforced. One should remember, though, that in specific industrial applications an additional risk assessment procedure has to be carried out to account for any possible hazards. The human motion is simulated by replicating the recorded motions in industry-like activities described in [20].

Notation: The positive semi-definiteness and positive definiteness of matrix $\boldsymbol{M}$, are denoted as $\boldsymbol{M} \succeq 0$ and $\boldsymbol{M} \succ 0$, respectively. Given $\boldsymbol{g} \in \mathbb{R}^n$, $\|\boldsymbol{g}\|$ indicates its Euclidean norm. Given $\boldsymbol{g} \in \mathbb{R}^n$ and $\boldsymbol{M} \in \mathbb{R}^{n \times n}$, $\|\boldsymbol{g}\|_{\boldsymbol{M}}^2 \triangleq \boldsymbol{g}'\boldsymbol{M}\boldsymbol{g}$, where $'$ is the transposition operator. A sequence of integers from ν_1 to ν_2 included is indicated as $\mathbb{N}_{[\nu_1,\nu_2]}$, e.g., $\mathbb{N}_{[1,4]} = \{1,2,3,4\}$.

II. Kinematics, dynamics, and constraints

The end effector pose of the considered manipulator is referred to as $\boldsymbol{y} = \begin{bmatrix}\boldsymbol{p}' & \boldsymbol{\phi}'\end{bmatrix}' \in \mathbb{R}^6$, in which $\boldsymbol{p} = (x_p, y_p, z_p) \in \mathbb{R}^3$ is the end-effector position in the fixed reference frame $O{-}xyz$ centered at the robot base, while $\boldsymbol{\phi} = (X, Y, Z) \in \mathbb{R}^3$ is the end-effector orientation in Euler angles. We consider the problem of moving the end effector from an initial pose $\boldsymbol{y}_0$ to a final pose $\boldsymbol{y}_f$. The preliminary building blocks of the proposed control scheme are analyzed in the following.

1) Robot and human space occupancy: To avoid collisions with the human, a number n_r of *test points* are defined on the robot frame: their positions $\boldsymbol{p}_{r,i} \in \mathbb{R}^3$, $i \in \mathbb{N}_r \triangleq \mathbb{N}_{[1,n_r]}$ in the $O - xyz$ reference frame changes as the manipulator modifies its configuration. The mentioned test points are the centers of spheres $\mathcal{S}_{r,i}$, $i \in \mathbb{N}_r$, with radii respectively equal to $R_{r,i}$. It is assumed that the union of all spheres $\mathcal{S}_{r,i}$, $i \in \mathbb{N}_r$, includes the robot frame. On the other hand, a number n_h of spheres $\mathcal{S}_{h,j}$, $j \in \mathbb{N}_h \triangleq \mathbb{N}_{[1,n_h]}$, with radii respectively equal to $R_{h,j}$, are defined such that their union covers the volume occupied by the human, plus a tolerance that prevents the robot from getting too close, especially to sensitive areas such as the face. The positions of their centers $\boldsymbol{p}_{h,j} \in \mathbb{R}^3$, $j \in \mathbb{N}_h$, in the $O - xyz$ reference frame also evolves as the human moves. The distance from the i-th sphere of the robot to the human (including the above-mentioned tolerances and based on the described union-of-spheres approximation) is defined as

$$d_{ih} = \min_{j \in \mathbb{N}_h} \left\{ d_{ij} - (R_{r,i} + R_{h,j}) \right\}, \tag{1}$$

where $d_{ij} \triangleq \|\boldsymbol{p}_{r,i} - \boldsymbol{p}_{h,j}\|$, with a negative d_{ih} indicating collision. The human-robot distance using the same approximation is thus defined as

$$d_{rh} = \min_{i \in \mathbb{N}_r} d_{ih}. \tag{2}$$

2) Robot kinematics and dynamics: The proposed control scheme acts in the manipulator joint space. The initial and final configurations of the robot end-effector are translated into corresponding joint positions $\boldsymbol{\theta} \in \mathbb{R}^{n_\theta}$ as $\boldsymbol{\theta}_0 = k_{\text{inv}}(\boldsymbol{y}_0)$, $\boldsymbol{\theta}_f = k_{\text{inv}}(\boldsymbol{y}_f)$, where $k_{\text{inv}}(\cdot) : \mathbb{R}^6 \to \mathbb{R}^{n_\theta}$ represents the robot inverse kinematics function. In general, $k_{\text{inv}}(\cdot)$ is not uniquely defined, however in this work we will assume that the initial and final values $\boldsymbol{\theta}_0$ and $\boldsymbol{\theta}_f$ are constant and assigned offline, during task definition. On the other hand, to determine the positions of the test points during task execution, one has to use forward kinematics: in particular, we have $\boldsymbol{p}_i = k_{\text{fwd,i}}(\boldsymbol{\theta})$, where $i \in \mathbb{N}_r$, while $k_{\text{fwd,i}}(\cdot) : \mathbb{R}^{n_\theta} \to \mathbb{R}^3$ is the forward kinematics function for the position of the i-th test point. Differential kinematics is needed to obtain the Cartesian velocity vectors of the test points during task execution, as $\boldsymbol{v}_i = k_{\text{diff,i}}(\boldsymbol{\theta}, \boldsymbol{\omega})$, where $\boldsymbol{\omega} \in \mathbb{R}^{n_\theta}$ is the vector of joint speeds, and $k_{\text{diff,i}}(\cdot) : \mathbb{R}^{n_\theta} \to \mathbb{R}^3$ is the differential kinematics function for the the i-th test point.

The robot system dynamics is defined in the joint space, as

$$\dot{\boldsymbol{\theta}} = \boldsymbol{\omega} \tag{3}$$

where the joint speeds in $\boldsymbol{\omega}$ are assumed here as control variables (i.e., a purely kinematic model is used). The equation is to be intended component-wise, i.e., $\dot{\boldsymbol{\theta}}_i = \boldsymbol{\omega}_i$, for $i \in \mathbb{N}_{[1,n_\theta]}$. The choice of equation (3) has two reasons: first, joint torques are not accessible in most commercial robots, but one can impose reference speeds for each joint which will be tracked by inner control loops; second, the use of such a simple model will allow to reduce the computational burden of the NMPC controller compared to a full nonlinear dynamic model.

To make notation compact, the information on robot test point positions and velocities, and radii of the corresponding spheres, is referred to as $\mathcal{R}$, while the overall information on position of test points on the human, and radii of the corresponding spheres, is referred to as $\mathcal{H}$.

3) Limits on joint angles and speeds: When executing the robot motion, the link speeds must remain within certain bounds, to avoid damaging the manipulator. Also, link positions are typically limited to avoid self-collisions, according to the robot specifications. The general set of constraints on joint positions can be written as $\boldsymbol{\theta} \in \Theta$ (implicitly assuming that $\boldsymbol{\theta}_0 \in \Theta$ and $\boldsymbol{\theta}_f \in \Theta$), while the overall set of constraints on joint speeds is $\boldsymbol{\omega} \in \Omega$, where $\Theta, \Omega \subseteq \mathbb{R}^{n_\theta}$ are closed sets.

4) Avoidance of fixed obstacles: Fixed obstacles (e.g., a wall or a table) can be present in the robot workspace. To avoid them, given the volume occupied by them (denoted as $\mathcal{O}$), we impose that, at all time instants,

$$\mathcal{I}(\mathcal{R}) \triangleq \left\{ \bigcup_{i \in \mathbb{N}_r} \mathcal{S}_{r,i} \right\} \cap \mathcal{O} = \emptyset. \tag{4}$$

If the obstacle is a half-space (e.g., a wall or a ceiling), then (4) can be imposed just requesting that each $p_{r,i}$, $i \in \mathbb{N}_r$, has a distance from the obstacle surface greater or equal than $R_{r,i}$. In case of more complex shapes, one can define a number of spheres that cover the whole obstacle volume, and impose that the distance between each $p_{r,i}$, $i \in \mathbb{N}_r$, and each sphere on the obstacle is greater or equal than $R_{r,i}$.

III. Robot speed modulation

This section provides an overview of the SSM algorithms described in [4], together with the last type of constraint that will be enforced in the NMPC problem, in addition to those already mentioned in Section II.

Assumption 1: The speed of each test point $\boldsymbol{p}_{h,j} \in \mathbb{R}^3$, $j \in \mathbb{N}_h$, on the human never exceeds $\bar{v}_H$, i.e., the maximum human speed according to safety standards. □

The SSM algorithms here considered are based on the adaptation of [4, Eq. 14] to our framework, which, in turn, implements a specific version of the ISO/TS 15066 SSM criterion detailed in [4, Eq. 2]). This condition ensures that, under the conservative assumption that robot and human are moving towards each other (with the human moving at his/her maximum speed), the robot can stop before a collision happens. Precisely, each $v_i \triangleq \|\boldsymbol{v}_i\|$, $i \in \mathbb{N}_r$, has to satisfy the following condition point-wise in time:

$$\bar{d}_{ih} + \bar{d}_{ir} \leq d_{ih} + \varepsilon_s, \tag{5}$$

where d_{ih} is defined in (1), while $\bar{d}_{ih}$ and $\bar{d}_{ir}$ are, respectively, the maximum distances that human and $\mathcal{S}_{r,i}$ can cover before one can achieve $v_i = 0$. Also, ε_s is the maximum error with which the positions of the test points of the human are detected by the employed motion capture system. We define $\bar{d}_{ih} \triangleq \bar{v}_h \left(T_{dr} + \frac{v_i}{\bar{a}_r}\right)$, where $T_{dr} \triangleq T_s + T_r$ is the time interval necessary for detection of the human position (assumed equal to the sampling interval T_s) and consequent reaction (depending on hardware and used algorithm, and referred to as T_r), while $\frac{v_i}{\bar{a}_r}$ is the time needed to bring $\mathcal{S}_{r,i}$ to a controlled stop (with $\bar{a}_r$ being the maximum allowable robot deceleration, equal for all spheres and assumed known). Also, we define $\bar{d}_{ir} \triangleq v_i T_{dr} + \frac{v_i^2}{2\bar{a}_r}$, with $v_i T_{dr}$ being the distance travelled by $\mathcal{S}_{r,i}$ while the human position is detected, and $\frac{v_i^2}{2\bar{a}_r}$ representing the distance needed to stop $\mathcal{S}_{r,i}$.

One can solve (5), which is quadratic in v_i, as an equation and obtain the maximum allowable speed of $\mathcal{S}_{r,i}$, namely $\bar{v}_i$, given the current value of d_{ih}, for all $i \in \mathbb{N}_r$. Given a nominal robot trajectory, defined in the absence of the human, the corresponding vector of joint speeds $\boldsymbol{\omega}$ is scaled multiplying it by $c \triangleq \min_{i \in \mathbb{N}_r} \bar{v}_i / v_i$ (v_i being here the robot test point speed for the nominal trajectory), whenever $c < 1$. This way, the robot will still follow the path given by the nominal trajectory in the joint space, only with reduced speed. This algorithm has been implemented in our case study, and is referred to in the remainder of the paper as *continuous SSM* (CSSM).

In industrial practice, the boundary $\bar{v}_i$ is not determined for each sphere, but as a single value $\bar{v}_r$ for the whole robot, obtained from (5) in which d_{rh}, defined in (2), is substituted to d_{ih}: this leads to a slightly more conservative scaling of the robot speed. In addition, the robot speed is not continuously modulated, but is changed based on one or more threshold values of d_{rh}: this is clearly explained in the *bimodal SSM* (BSSM) and *trimodal SSM* (TSSM) schemes described in [4], also implemented in our case study.

In our NMPC scheme, we aim at implementing a criterion similar to CSSM, but while being able to change the robot trajectory. The following set of constraints is imposed:

$$v_i^2 \leq \alpha^2 \left(d_{ij}^2 - \left(R_{r,i} + R_{h,j} + \bar{d}\right)^2\right), \ \forall (i,j) \in \mathbb{N}_r \times \mathbb{N}_h \tag{6}$$

where $\alpha, \bar{d} \in \mathbb{R}_{\geq 0}$ are tuning parameters. Condition (6) aims at substituting (5), as it would not be possible to directly implement (5) in our NMPC problem. The main reason is that v_i always has to appear squared, as the square root function (necessary to directly obtain v_i) is not differentiable at the origin, and thus would cause problems to the optimization solver.

Remark 1: Tuning α and $\bar{d}$ so that (6) implies (5) for all d_{ih} (which can be verified graphically, as will be done for our case study) ensures that the NMPC algorithm implicitly guarantees the satisfaction of the SSM limits. □

By defining, for notation convenience, $f_{ij}(\mathcal{R}, \mathcal{H}) \triangleq v_i^2 - \alpha^2 \left(d_{ij}^2 - \left(R_{r,i} + R_{h,j} + \bar{d}\right)^2\right)$, we can rewrite equation (6) as $f_{ij}(\mathcal{R}, \mathcal{H}) \leq 0$, for all $(i,j) \in \mathbb{N}_r \times \mathbb{N}_h$, or, more compactly, as $f(\mathcal{R}, \mathcal{H}) \leq 0$, where $f(\mathcal{R}, \mathcal{H}) \in \mathbb{R}^{\mathbb{N}_r \mathbb{N}_h}$ is a vector containing all the scalar functions $f_{ij}(\mathcal{R}, \mathcal{H})$.

IV. The NMPC Motion Planning Scheme

The proposed control scheme is made of two cascaded NMPC controllers. In Section IV-A, we will consider only the highest-level controller, namely the *long-term NMPC* (L-NMPC) controller, which has the task of planning the robot motion. Then, Section IV-B will introduce the presence of the *short-term NMPC* (S-NMPC) controller, mainly aimed at a fast adaptation of the robot speed depending on the human motion. As apparent from Section III, a high value of T_{dr} would make the robot velocity modulation profile more conservative, thus hindering robot productivity. In order to keep T_{dr} low, one needs to decrease the sampling interval (which coincides with the discretization interval of the NMPC controller) as much as possible; however, the discretization interval cannot be decreased too much, or else one would not be able to plan the robot trajectory until the goal configuration without exceeding the available computation time. This is the reason why in this work we propose the above-described scheme with two different NMPC controllers.

A. Long-term NMPC controller

The robot state $\boldsymbol{\theta}_c \triangleq \boldsymbol{\theta}(t_c)$ is acquired at the current time instant t_c together with the positions of the test points $\boldsymbol{p}_{h,j}$, $j \in \mathbb{N}_h$, on the human. The so-called *prediction horizon* $N_\ell \in \mathbb{N}_{>0}$ (the subscript ℓ stands for "long-term") defines how many integer multiples of the sampling interval $T_{s,\ell}$ are considered for predicting and optimizing the system dynamics for L-NMPC. The exact discretization of dynamics (3), with the given sampling interval for a sample-and-hold realization of the input signal, is given by

$$\boldsymbol{\theta}(\kappa + 1) = \boldsymbol{\theta}(\kappa) + T_{s,\ell} \boldsymbol{\omega}(\kappa), \tag{7}$$

with $\kappa \in \mathbb{N}$ being the discrete-time index, and in which $\boldsymbol{\omega}(\kappa)$ and $\boldsymbol{\theta}(\kappa)$ represent the vectors of joint speeds $\boldsymbol{\omega}$ and joint angles $\boldsymbol{\theta}$ at time $t = \kappa T_{s,\ell}$. The NMPC controller will explore different realizations of the discrete-time control sequence

$$\bar{\boldsymbol{\omega}}_\ell \triangleq \{\boldsymbol{\omega}_\ell(0), \boldsymbol{\omega}_\ell(1), \ldots, \boldsymbol{\omega}_\ell(N_\ell - 1)\}, \tag{8}$$

where $\boldsymbol{\omega}_\ell(k)$ and $\boldsymbol{\theta}_\ell(k)$ (with $k \in \mathbb{N}_{\geq 0}$ being the discrete-time index in the prediction) represent the predictions of $\boldsymbol{\omega}(k)$ and $\boldsymbol{\theta}(k)$, respectively, at time $t_c + k T_{s,\ell}$ (i.e., $k = 0$ corresponds to time instant t_c for the predicted sequences). The corresponding sequence of states, obtained applying dynamics (7) to the predicted state values, is referred to as

$$\bar{\boldsymbol{\theta}}_\ell \triangleq \{\boldsymbol{\theta}_\ell(0), \boldsymbol{\theta}_\ell(1), \ldots, \boldsymbol{\theta}_\ell(N_\ell)\}. \tag{9}$$

For each time instant of the predicted input/state sequences, we introduce the scalar variable $\varphi_\ell(k)$, defined as

$$\varphi_\ell(k) \triangleq e^{-\beta \frac{\delta_h\left(\mathcal{R}_\ell(k),\mathcal{H}_0\right)^2}{\delta_g\left(\mathcal{R}_\ell(k)\right)^2}}, \tag{10}$$

where $\beta \in \mathbb{R}_{>0}$ is a design parameter, $\delta_h\left(\mathcal{R}_\ell(k),\mathcal{H}_0\right)$ is the distance of the robot end effector from the currently measured location of a single sphere $\mathcal{S}_{h,j}$ on the human, while $\delta_g\left(\mathcal{R}_\ell(k)\right)$ is the distance of the robot end effector from the end-effector position corresponding to the goal configuration $\boldsymbol{y}_f$. In (10), $\mathcal{H}_0 \triangleq \mathcal{H}_\ell(0)$ represents the information on the human at time t_c. Instead, $\mathcal{R}_\ell(k)$ represents the expressions of the predicted overall information $\mathcal{R}$ on the robot at time k, which can be calculated given $\boldsymbol{\theta}_\ell(k)$ and $\boldsymbol{\omega}_\ell(k)$.

The controller will determine the optimal sequences $\bar{\boldsymbol{\omega}}_\ell^*$ and $\bar{\boldsymbol{\theta}}_\ell^*$ (containing the corresponding elements $\boldsymbol{\omega}_\ell^*(k)$ and $\boldsymbol{\theta}_\ell^*(k)$) by minimizing the following quadratic cost function:

$$J_\ell\left(\bar{\boldsymbol{\omega}}_\ell, \bar{\boldsymbol{\theta}}_\ell, \mathcal{H}_0\right) \triangleq \sum_{k=0}^{N_\ell-1} \|\boldsymbol{\theta}_\ell(k) - \boldsymbol{\theta}_f\|_{\boldsymbol{Q}}^2 + \|\boldsymbol{\omega}_\ell(k)\|_{\boldsymbol{R}}^2 + \gamma\varphi_\ell^2(k) \tag{11}$$

where $\boldsymbol{Q} = \boldsymbol{Q}' \in \mathbb{R}^{n_\theta \times n_\theta}$ and $\boldsymbol{R} = \boldsymbol{R}' \in \mathbb{R}^{n_\theta \times n_\theta}$ satisfy $\boldsymbol{Q} \succeq 0$, and $\boldsymbol{R} \succ 0$, while $\gamma \in \mathbb{R}_{>0}$. The cost function aims at steering the regulation error $\boldsymbol{\theta}_\ell - \boldsymbol{\theta}_f$ to zero, while avoiding using high joint speeds $\boldsymbol{\omega}_\ell$ when not necessary, in order to obtain a smooth robot motion. Also, in order to maintain $\varphi_\ell^2(k)$ low, the robot end effector should stay far from the human, and close to the goal point: this term is introduced to provide further incentive for the robot to move away from the human and find alternative trajectories, rather than simply slow down and stop as in a standard SSM scheme.

The following finite-horizon optimal control problem (FHOCP) determines $\bar{\boldsymbol{\omega}}_\ell^*$ and $\bar{\boldsymbol{\theta}}_\ell^*$ for L-NMPC:

$$\left(\bar{\boldsymbol{\omega}}_\ell^*, \bar{\boldsymbol{\theta}}_\ell^*\right) = \arg\min_{\bar{\boldsymbol{\omega}}_\ell, \bar{\boldsymbol{\theta}}_\ell} \; J_\ell(\bar{\boldsymbol{\omega}}_\ell, \bar{\boldsymbol{\theta}}_\ell, \mathcal{H}_0) \tag{12a}$$

$$\text{subj. to } \boldsymbol{\theta}_\ell(0) = \boldsymbol{\theta}_c, \;\; \boldsymbol{\theta}_\ell(N_\ell) = \boldsymbol{\theta}_f \tag{12b}$$

$$\boldsymbol{\theta}_\ell(k+1) = \boldsymbol{\theta}_\ell(k) + T_{s,\ell}\boldsymbol{\omega}_\ell(k), \;\; k \in \mathbb{N}_{[0,N_\ell-1]} \tag{12c}$$

$$\boldsymbol{\omega}_\ell(k) \in \Omega, \;\; k \in \mathbb{N}_{[0,N_\ell-1]} \tag{12d}$$

$$\boldsymbol{\theta}_\ell(k) \in \Theta, \;\; k \in \mathbb{N}_{[0,N_\ell]} \tag{12e}$$

$$\mathcal{I}(\mathcal{R}_\ell(k)) = \emptyset, \;\; k \in \mathbb{N}_{[0,N_\ell]}, \tag{12f}$$

$$f(\mathcal{R}_\ell(k), \mathcal{H}_0) \leq 0, \;\; k \in \mathbb{N}_{[0,N_\ell]}. \tag{12g}$$

Condition (12b) defines the system state at the beginning of the prediction horizon (corresponding with the currently measured joint positions), and imposes that the robot configuration at the end of the prediction horizon is equal to the goal configuration $\boldsymbol{\theta}_f$. Equation (12c) requires that the predicted system evolution happens according to the discrete-time model (7). The inequality constraints (12d)-(12g) impose, respectively, joint speed constraints, joint position constraints, avoidance of fixed obstacles, and velocity modulation constraints. The use of $\mathcal{H}(0)$ in the latter implies that the predictions made by the NMPC controller are based on the simplifying assumption that the human maintains his/her current pose. However, at the next sampling interval, a new realization of $\mathcal{H}_0$ will be obtained from sensors, to account for the change of posture of the human within the sampling interval.

After the solution of (12) is calculated at time $\kappa T_{s,\ell}$, only the first control move $\boldsymbol{\omega}_\ell^*(0)$ is applied to the discrete-time system (7), as $\boldsymbol{\omega}(\kappa) = \boldsymbol{\omega}_\ell^*(0)$; after acquiring the new state measurement and the new information on the human at time $(\kappa+1)T_{s,\ell}$, the FHOCP is solved again, according to the so-called *receding-horizon* principle [21].

Remark 2: When implementing the control law, condition $f(\mathcal{R}_\ell(0), \mathcal{H}_0) \leq 0$ is checked before solving the FHOCP (12) at each sampling instant: if true, the FHOCP is solved; if false, then $\boldsymbol{\omega}(\kappa) = \boldsymbol{0}$ is set without solving the FHOCP, and the FHOCP is solved again as soon as (12g) becomes true. The imposition of $\boldsymbol{\omega}(\kappa) = \boldsymbol{0}$ is also carried out in case no feasible solution of the FHOCP is found. □

Theorem 1: Assume that the control law for the considered human-robot system is defined as in Remark 2, and applied directly to the robot dynamics described in (7). Also, assume that a solution of (12) is available at time $\kappa T_{s,\ell}$ (with any $\kappa \in \mathbb{N}_{\geq 0}$), and that the human does not change his/her position for $t \geq \kappa T_{s,\ell}$. Then, (i) the FHOCP (12) remains feasible for all sampling instants $\kappa T_{s,\ell}$, $\kappa \in \mathbb{N}_{\geq 0}$, and (ii) the goal configuration $\boldsymbol{\theta}_f$ is an asymptotically stable equilibrium point for the closed-loop discrete-time system given by (7) with $\boldsymbol{\omega}(\kappa) = \boldsymbol{\omega}_\ell^*(0)$ defined as in Remark 2.

Proof: Given the optimal control sequence $\bar{\boldsymbol{\omega}}_\ell^*$ defined at time $\kappa T_{s,\ell}$, a feasible (and, in general, sub-optimal) control sequence at time $(\kappa+1)T_{s,\ell}$ is given by

$$\tilde{\boldsymbol{\omega}}_\ell(k) \triangleq \begin{cases} \boldsymbol{\omega}_\ell^*(k+1), & k \in \mathbb{N}_{[0,N_\ell-2]} \\ 0 & k = N_\ell - 1. \end{cases} \tag{13}$$

Indeed, one can easily verify that, applying zero joint speeds at $k = N_\ell - 1$, all the constraints in the FHOCP (12), shifting time forward of one discrete time instant, are satisfied. As a feasible solution exists, a new optimal input sequence will exist at time $\kappa + 1$, which proves (i). In order to prove (ii), we first refer to the optimal cost function (i.e., the solution of the FHOCP (12)) as $J_\ell(\boldsymbol{\omega}_\ell^*, \boldsymbol{\theta}_\ell^*, \mathcal{H}_0)$. As the latter is uniquely determined as a function of the robot state and $\mathcal{H}_0$ is assumed to remain known and constant, we will refer to it in short as $V(\boldsymbol{\theta}) \triangleq J_\ell(\boldsymbol{\omega}_\ell^*, \boldsymbol{\theta}_\ell^*, \mathcal{H}_0)$, and consider it as a Lyapunov function candidate for the considered closed-loop system. By construction, $V(\boldsymbol{\theta}) > 0$ for all $\boldsymbol{\theta} \neq \boldsymbol{\theta}_f$, and $V(\boldsymbol{\theta}_f) = 0$. Also, a possible value of the cost function associated to the control sequence in (13) at the discrete time instant $\kappa + 1$, given the optimal sequence at time κ, can be determined by construction as $\tilde{V}\left((\boldsymbol{\theta}(\kappa+1)|\kappa) = V(\boldsymbol{\theta}(\kappa)) - \|\boldsymbol{\theta}(\kappa) - \boldsymbol{\theta}_f\|_{\boldsymbol{Q}}^2 - \|\boldsymbol{\omega}_\ell(\kappa)\|_{\boldsymbol{R}}^2 - \gamma\varphi_\ell^2(\kappa) < V(\boldsymbol{\theta}(\kappa))\right)$ for $\boldsymbol{\theta}(\kappa) \neq \boldsymbol{\theta}_f$. As the optimal value of the cost function at the discrete-time instant $\kappa + 1$ will surely satisfy $V(\boldsymbol{\theta}(\kappa+1)) \leq \tilde{V}\left((\boldsymbol{\theta}(\kappa+1)|\kappa\right)$, then we conclude that $V(\boldsymbol{\theta}(\kappa+1)) < V(\boldsymbol{\theta}(\kappa))$ for $\boldsymbol{\theta}(\kappa) \neq \boldsymbol{\theta}_f$. Thus, $V(\boldsymbol{\theta})$ is a Lyapunov function, which proves that $\boldsymbol{\theta}_f$ is an asymptotically stable equilibrium point for the closed-loop system. This result follows from the general approach known as *zero terminal constraint* [21], which is applicable thanks to the imposition of constraint $\boldsymbol{\theta}_\ell(N_\ell) = \boldsymbol{\theta}_f$. ■

Remark 3: The FHOCP (12) guarantees that, once a feasible motion of the system has been determined, it is possible

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2021.3083581, IEEE Robotics and Automation Letters

to guarantee that the robot configuration will converge to its goal, *assuming that the human will not move*. If the human moves without assumptions on his/her future trajectory, it is not possible to guarantee convergence: imagine the case in which the operator keeps moving to purposely block the robot motion. Thus, our aim is not to avoid the human and reach the goal point in any situation, but to guarantee that the goal point is reached, at least when the human stops at a configuration for which a feasible robot trajectory exists. □

Remark 4: Safety is ensured by the velocity modulation constraints imposed in the FHOCP (12) (see Remark 1), which keep being enforced thanks to the recursive feasibility property proven in Theorem 1. □

B. Short-term NMPC controller

S-NMPC aims at tracking the trajectory already predicted by L-NMPC, but over a much shorter prediction time interval, and with a sampling interval $T_{s,\sigma}$ (where the subscript σ stands for "short-term") shorter than $T_{s,\ell}$ so as to obtain $T_{dr} = T_{s,\sigma} + T_r$, which leads to the imposition of less restrictive speed modulation constraints as compared to using L-NMPC alone. S-NMPC generates a new optimal sequence of link velocities and corresponding sequence of link positions, namely $\bar{\boldsymbol{\omega}}^*_\sigma$ and $\bar{\boldsymbol{\theta}}^*_\sigma$, with the mentioned sampling interval $T_{s,\sigma}$: these are the reference speeds that will be sent to the inner velocity control loop of each link.

$T_{s,\sigma}$ is defined to be a submultiple of $T_{s,\ell}$, according to $T_{s,\ell} = N_m T_{s,\sigma}$, with $N_m \in \mathbb{N}_{>0}$. The state prediction $\bar{\boldsymbol{\theta}}^*_\ell$ is used as a reference for S-NMPC, after having been resampled via linear interpolation. As the system dynamics is given by one integrator for each joint angle, the resampled time evolution of the robot joint angles exactly corresponds to the application of constant joint speeds during each sampling interval $T_{s,\ell}$.

The S-NMPC solution is obtained by solving a FHOCP with the same structure as (12), with the difference that it solves a tracking problem (tracking the time-varying resampled evolution of $\bar{\boldsymbol{\theta}}^*_\ell$), with a discretization interval of $T_{s,\sigma}$ coinciding with the actual sampling interval, a prediction horizon of N_σ discretization intervals, and no terminal constraint. The thread that executes S-NMPC runs in parallel with the one executing L-NMPC, and the reference for S-NMPC is updated as soon as a new L-NMPC solution is available.

Remark 5: As both L-NMPC and S-NMPC are based on kinematic models, we assume that the joint speed reference generated by S-NMPC is passed to a low-level embedded robot controller that, similarly to [22], guarantees limits on joint torques, accelerations, speeds and displacements. This is done in order to safeguard the robot against effects such as torque overload and hitting of mechanical joint limits. The overall NMPC control scheme, including the robot controller, is shown in Fig. 2. □

Remark 6: In terms of stability of the overall system, Theorem 2 would remain valid in case both S-NMPC and the low-level control loops perfectly track their references. This is never exactly true; however, it is reasonable to expect that a solution of the FHOCP (12) will be found if after $T_{s,\ell}$ (i) the joint positions do not differ greatly from their prediction made by L-NMPC, and (ii) the human does not completely change his/her position. An alternative to our approach would have been the use of robust NMPC to account for imperfect tracking, which would pose several additional challenges and is out of the scope of this paper. □

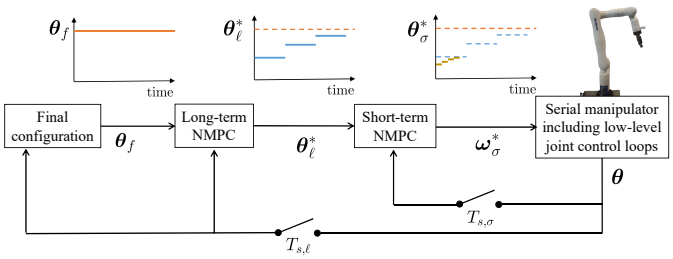

Fig. 2. Block diagram of the cascaded NMPC scheme. Both L-NMPC and S-NMPC also receive human motion capture information, which is not represented here for the sake of simplicity.

V. Case Study

Our case study applies the described NMPC scheme to a *Kinova Gen3 robot*, a torque-controlled collaborative manipulator with 7 degrees of freedom (i.e., $n_\theta = 7$). The robot motion simulates a sequence of pick-and-place movements, as the reference for the robot configuration changes from $\boldsymbol{\theta}_0 = [0.37 \;\; -0.84 \;\; 0.31 \;\; -0.58 \;\; -0.26 \;\; -0.56 \;\; 0.82]'$ to $\boldsymbol{\theta}_f = [-2.55 \;\; -0.94 \;\; 0.31 \;\; -0.88 \;\; -0.26 \;\; -1.36 \;\; 0.82]'$ and back, as soon as the robot reaches its currently set goal point. The forward, inverse and differential kinematics, needed to convert variables from the $O-xyz$ reference frame into the joint space and vice-versa, are formulated via homogeneous transformation matrices using the Denavit-Hartenberg parameters provided by the manufacturer. As the robot executes its task while being placed on a table with height of 1.2 m from the ground, the human is simulated in the same workspace by replicating the motion of one of the subjects in [20][1]. The human motion in [20] was recorded using both inertial (Xsens MVN Link system) and optical (Qualisys system with 12 Oqus cameras) motion capture systems, while the human carried out industry-like activities, such as executing a screwing task at three different heights, untying a knot, carrying loads of 5 kg and 10 kg and place them on shelves.

We start defining the parameters described in Section II. To account for the robot space occupancy, a total of $n_r = 7$ spheres $\mathcal{S}_{r,i}$ are located on the robot kinematic chain. The spheres have radii $R_{r,i} = 0.12$ m for $i = 1, 2, 3, 4$, $R_{r,i} = 0.06$ m for $i = 5, 6$, and $R_{r,7} = 0.1$ m. Regarding avoidance of fixed obstacles, condition (4) is defined by imposing that all spheres $\mathcal{S}_{r,i}$ are above the height of the table: in practice, this is achieved imposing that the center $\boldsymbol{p}_{r,i}$ of each sphere is above the table level of a quantity equal to $R_{r,i}$. This constraint is imposed because, assuming that the human's legs will never be lifted above the level of the table, it makes it possible not to directly impose the avoidance of the human's legs, so as to reduce the number of constraints for the NMPC controllers.

[1]https://zenodo.org/record/3254403#.X8HcD2gzZPY

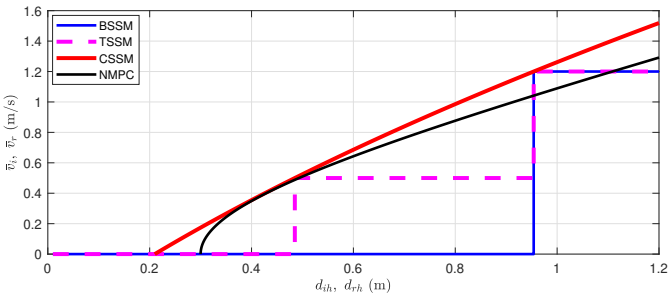

Fig. 3. Velocity modulation profiles for the four controllers considered in the case study.

As a consequence, $n_h = 14$ spheres $\mathcal{S}_{h,j}$ are located on the human body, excluding legs. The positions of their centers are obtainable for all human poses from the motion capture data described in [20], and the spheres have radii $R_{h,j} = 0.25$ m for $j = 1, 13$, $R_{h,j} = 0.3$ m for $j = 2, 14$, $R_{h,j} = 0.2$ m for $j = 3, 4, 5, 6, 7, 8$, $R_{h,j} = 0.15$ m for $j = 9, 10$, and $R_{h,j} = 0.18$ m for $j = 11, 12$. The chosen values of n_r and n_h, and the consequent values of sphere centers and radii, were determined experimentally in order to maintain the computational complexity of the FHOCPs within reasonable bounds, at the same time avoiding an excessively conservative coverage of human and robot frames. In terms of limits on joint angles and speeds, it is imposed that $|\theta_2| \leq 2.2497$ rad, $|\theta_4| \leq 2.5795$ rad, and $|\theta_6| \leq 2.0996$ rad to avoid self-collisions, while $|\omega_i| \leq 1.2$ rad/s for all $i \in n_r$.

We now move to the terms defined in Section III. The value of $\bar{v}_H = 2$ m/s is set as the maximum human velocity according to the ISO 13855 standard [4, Sec. V], while the sensor position error for the motion capture system is estimated as $\varepsilon_s = 10^{-3}$ m. Also, based on the robot characteristics, we set $\bar{a}_r = 5$ m/s^2. Given an S-NMPC sampling time $T_{s,\sigma} = 50$ ms, and setting $T_r = T_{s,\sigma}$ to account for the worst-case reaction scenario, we obtain $T_{dr} = 0.1$ s. To ensure that (6) implies (5) for all d_{ih}, we set $\alpha = 0.89$ and $\bar{d} = 0.3$ m, which ensures that the NMPC velocity modulation profile never exceeds the corresponding CSSM curve (Fig. 3).

We are now ready to describe the implementation details of the NMPC controllers (Section IV). L-NMPC is defined with $T_{s,\ell} = 500$ ms and a prediction horizon $N_\ell = 10$, leading to a total prediction time interval of 5 s. The 5 s time interval makes it possible, in our case study, to plan the robot motion until $\boldsymbol{\theta}_f$ is reached, so as to satisfy constraint $\boldsymbol{\theta}_\ell(N_\ell) = \boldsymbol{\theta}_f$ in (12b). On the other hand, $T_{s,\ell} = 500$ ms is chosen to allow for the computation of the FHOCP solution within the sampling interval. The value of β in (10) is set to $\beta = 3$, while the FHOCP cost function in (11) is defined by $\boldsymbol{Q} = \text{diag}\{20, 20, 15, 15, 10, 10, 10\}$ and $\gamma = 500$, while $\boldsymbol{R}$ is an identity matrix. These weights are determined by trial-and-error tuning. In particular, the values of β and γ influence how far the robot will tend to pass from the human: if too close, the robot might often have to stop; if too far, the time to reach $\boldsymbol{\theta}_f$ tends to increase. The FHOCP constraints in (12) are defined using the above-described obstacle and speed modulation parameters. Regarding S-NMPC, we set $T_{s,\sigma} = 50$ ms (already mentioned above) and a prediction horizon $N_\sigma = 10$, leading to a total prediction time interval of 500 ms. In contrast to L-NMPC, there is no need to reach any goal configuration by the end of the prediction horizon, so $T_{s,\sigma} = 50$ ms is set to be able to solve the corresponding FHOCP within the sampling interval, while the prediction horizon is determined by trial and error tuning. The inequality constraints imposed in the FHOCP for S-NMPC are the same as in L-NMPC. Both FHOCP are cast into nonlinear programs, after discretizing the system dynamics with a time interval equal to the sampling time, via a multiple-shooting approach using the ACADO Toolkit [8]. The solutions are based on sequential quadratic programming (SQP), with a number of SQP steps equal to 10 and 20, respectively, for L-NMPC and S-NMPC. A Gauss-Newton approximation of the Hessian of the Lagrangian is employed in the SQP steps, while qpOASES [23] is used to solve, in condensed form, each quadratic program.

To compare the proposed approach with state-of-the-art methods, the three SSM schemes described in Section III are also implemented. The first scheme (BSSM) simply allows the robot speed v_r to take any values as long as $d_{rh} \geq \hat{d}_{rh}$, where $\hat{d}_{rh} = 0.954$ is a constant threshold, and stops the robot whenever $d_{rh} < \hat{d}_{rh}$ (see Fig. 3). The value of $\hat{d}_{rh}$ is determined as the distance corresponding, on the CSSM curve, to the speed $\hat{v}_r = 1.2$ m/s. In turn, the latter is an upper bound to the values of v_i, $i \in \mathbb{N}_r$, obtained when following the nominal trajectory without human presence: this ensures that the value of $\hat{v}_r$ is never reached by any robot test points, thus allowing the robot to follow the nominal trajectory with full speed. The second scheme (TSSM) still imposes the same conditions on robot speed as BSSM, but, additionally, imposes a maximum value of v_r equal to 0.5 m/s whenever $d_{rh} \in [0.5, \hat{d}_{rh}]$ (also shown in Fig. 3). Both of these methods were implemented in [4] for a different case study. Finally, the third scheme (CSSM) operates as described in Section III, and its velocity modulation curve constitutes an upper bound for all the other curves. All three SSM schemes require a nominal trajectory, which, in order to provide a fair comparison with NMPC, is defined as the robot trajectory with the proposed NMPC scheme, without human. When the human approaches the robot, the same path in the joint space, corresponding to the nominal trajectory, is followed with reduced speed.

All controllers are implemented on a desktop computer with i9-7900X CPU and 16 GB RAM, running Robot Operating System (ROS). The NMPC controllers, obtained via C-language code generation via ACADO, are implemented as separate C++ nodes in ROS. Every 50 ms, joint velocity commands are sent from the computer to the internal motion control system of the Kinova Gen3 robot via ROS-Kortex driver[2]. The current joint positions and velocities of the robot are read via the ROS-Kortex interface, also every 50 ms.

VI. Experimental Results and Discussion

The four considered methods are applied to the real Kinova Gen3 robot avoiding the above-mentioned simulated human.

[2]https://github.com/Kinovarobotics/ros_kortex

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LRA.2021.3083581, IEEE Robotics and Automation Letters

Safety is guaranteed by the imposed velocity modulation constraints for all methods, and these constraints are always satisfied in the experiments. Therefore, what differentiates the four methods is robot productivity, defined in [4] as

$$P_R = \frac{\hat{\tau}}{\tau_R} \tag{14}$$

where $\hat{\tau}$ is the ideal task execution time (i.e., the time that the robot would employ to move from $\boldsymbol{\theta}_0$ to $\boldsymbol{\theta}_f$ and back when no human is present), and τ_R is the average execution time (i.e., the average time that the robot employs for the whole cycle consisting of moving from $\boldsymbol{\theta}_0$ to $\boldsymbol{\theta}_f$ and back) when the human is moving in the robot workspace. To evaluate the productivity of the four methods over a time interval long enough to present a significant number of situations, the overall human motion, which has a duration of 85 s, is replicated multiple times with various time delays at the beginning of the motion itself, so as to obtain many different combinations of human and robot motion. The resulting total duration of the experiments for each method is equal to 28 min and 20 s.

TABLE I
COMPARISON OF THE VALUES OF $\hat{\tau}$, τ_R, AND P_R FOR THE EXPERIMENTAL RESULTS.

	No pauses			4 sec. pauses		
	$\hat{\tau}$ (s)	τ_R (s)	$P_R(\%)$	$\hat{\tau}$ (s)	τ_R (s)	$P_R(\%)$
NMPC	9.53	14.57	65.41	9.53	16.40	58.78%
CSSM	9.76	16.17	60.36	9.76	19.11	51.07%
TSSM	9.77	16.59	58.89	9.77	20.61	47.40%
BSSM	9.77	18.24	53.56	9.77	22.17	44.07%
	8 sec. pauses			12 sec. pauses		
	$\hat{\tau}$ (s)	τ_R (s)	$P_R(\%)$	$\hat{\tau}$ (s)	τ_R (s)	$P_R(\%)$
NMPC	9.53	18.60	51.24	9.53	17.96	53.06%
CSSM	9.76	22.66	43.07	9.76	25.82	37.80%
TSSM	9.77	24.55	39.80	9.77	27.81	35.13%
BSSM	9.77	26.17	37.33	9.77	30.16	32.39%

The result can be seen in the upper-left part of Table I. The value of $\hat{\tau}$ is very similar for all four methods, and slightly lower for NMPC: this is due to the fact that NMPC directly determines joint speeds, while the other methods track the time evolution of joint positions determined by NMPC for the case without human. To compensate for this slight advantage of NMPC, the value of P_R for each method is calculated dividing by the corresponding value of $\hat{\tau}$. As expected, the productivity increases from BSSM up to NMPC, with the latter having a productivity increment relative to CSSM of 8.37%. Still, one could argue that a motion (such as the one in the data set) in which the human never stops would not represent the ideal case for showing the full potential of NMPC. The least favourable case for SSM is when the operator maintains a position that obstructs the robot motion for some time: in such a case, NMPC can replan the robot trajectory, while the SSM schemes just have to wait. In order to show this concept, pauses of 4 s, 8 s, and 12 s, respectively, have been added whenever the operator executes one of the tasks listed in Section V, thus obtaining a total duration of the motion sequence of 33 min 20 s, 38 min 40 s, and 44 min, respectively. The results, analogous to those of the case with no pauses in the human motion, are also shown in Table I: even if the overall productivity typically decreases as the human occupies the robot workspace for longer, the advantage of NMPC compared to the SSM schemes keeps increasing, with a relative advantage on CSSM of 15.10%, 18.97%, and 40.37%, respectively, for the 4 s, 8 s, and 12 s cases.

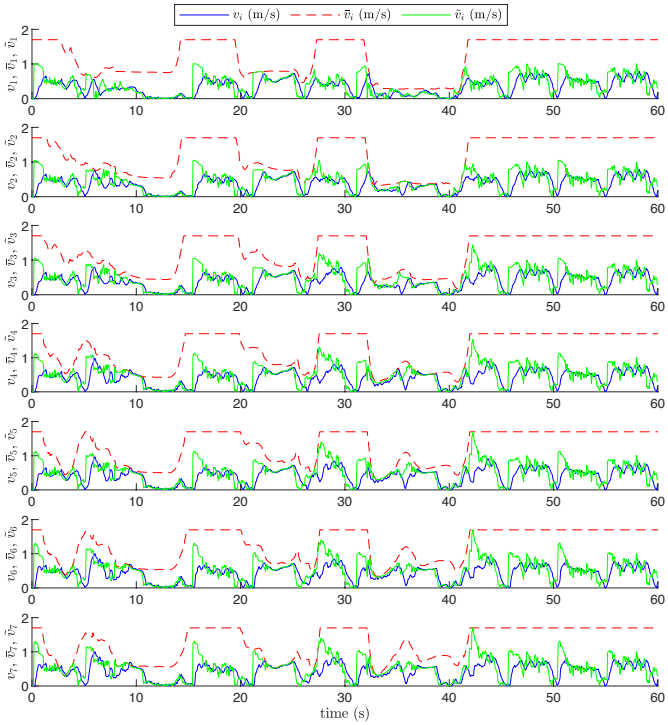

Fig. 4. Linear velocity of the robot under two-level MPC controller for the 7 test points on the robot.

The simple integrator model (3) that is also implemented (after time discretization) in NMPC implicitly assumes that reference speed and actual speed of the joints coincide. This might not be accurate, especially when the reference speeds undergo a sudden change. However, as the actual speed of each test point (simply indicated in Fig. 4 as v_i) is in practice always lower than its ideal value from S-NMPC (indicated as $\tilde{v}_i$ in Fig. 4), the fact that all $\tilde{v}_i$ never exceed the boundary given by the speed modulation constraint implies that also the values of v_i are within safety bounds. To show this concept, the time evolution of v_i and $\tilde{v}_i$, together with the upper bound $\bar{v}_i$, are shown in Fig. 4 for all seven test points, for one minute of experimental results.

For the same time interval, the tracking of the reference robot position is shown in Fig. 5 in terms of end effector pose (position $p = (x_p, y_p, z_p)$ and orientation $\phi = (X, Y, Z)$). The robot reaches the prescribed goal configurations at every cycle. However, it is difficult to relate these graphs with the corresponding human motion. For this reason, a video is provided as supplementary material: in this video (see Fig. 1), we show a 3D simulation in Gazebo of the Kinova Gen3 manipulator (which exactly replicates the motion measured from the experimental setup) and of the human for a subset of the cases summarized in Table I, and for all four methods.

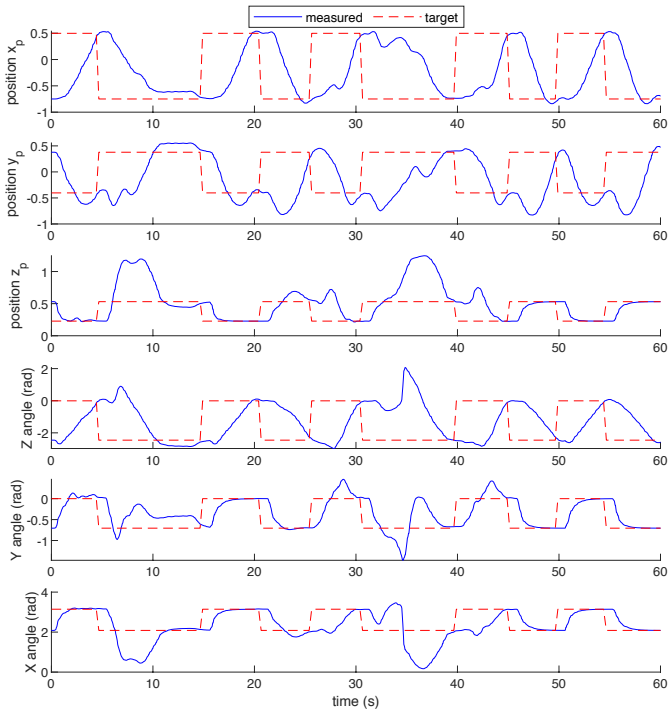

Fig. 5. Cartesian coordinates of end-effector position for robot controlled by two-level MPC.

Among other factors, the performance of the described control laws depends on T_{dr}, which in turn depends on the sampling time. In the described results, the SSM schemes were implemented with the same sampling time of S-NMPC, i.e., 50 ms, so as to provide a fair comparison. However, as the low-level robot control loops run at 40 Hz, one can further lower the sampling time of the SSM schemes to 25 ms, thus using a less conservative velocity modulation profile. As a result, CSSM would slightly outperform the previously-described NMPC scheme in the case without pauses ($P_R = 67.62\%$), but not in the cases with pauses (P_R equal to 55.77%, 47.13% and 40.80%, respectively, for the cases with 4, 8, and 12 s pauses).

VII. Conclusions and Outlook

The proposed NMPC law allows a robot to safely replan its motion in the presence of a human: both theoretical stability properties and the practical assessment of its productivity compared to SSM schemes have shown its potential for industrial applications. Future work will be devoted to testing the proposed algorithm with actual human participants, in order to account for the effect of unexpected human reactions to certain motions of the robot induced by our controllers. Also, the possible use of robust NMPC to account for imperfect tracking of the predicted L-NMPC trajectory will be investigated.

References

[1] A. Ajoudani, A. M. Zanchettin, S. Ivaldi, A. Albu-Schäffer, K. Kosuge, and O. Khatib, "Progress and prospects of the human–robot collaboration," *Auton Robot*, vol. 42, no. 5, pp. 957–975, 2018.

[2] S. Haddadin and E. Croft, "Physical human–robot interaction," in *Springer Handbook of Robotics*. Springer, 2016, pp. 1835–1874.

[3] *ISO-TS 15066: Robots and robotic devices – Collaborative robots*, Int. Organization for Standardization, Geneva, Switzerland, 2016.

[4] J. A. Marvel, "Performance metrics of speed and separation monitoring in shared workspaces," *IEEE T Autom Sci Eng*, vol. 10, no. 2, pp. 405–414, 2013.

[5] J. A. Marvel and R. Norcross, "Implementing speed and separation monitoring in collaborative robot workcells," *Robot Com-Int Manuf*, vol. 44, pp. 144–155, 2017.

[6] V. Villani, F. Pini, F. Leali, and C. Secchi, "Survey on human–robot collaboration in industrial settings: Safety, intuitive interfaces and applications," *Mechatronics*, vol. 55, pp. 248–266, 2018.

[7] C. Byner, B. Matthias, and H. Ding, "Dynamic speed and separation monitoring for collaborative robot applications–concepts and performance," *Robot Com-Int Manuf*, vol. 58, pp. 239–252, 2019.

[8] B. Houska, H. J. Ferreau, and M. Diehl, "ACADO toolkit - an open-source framework for automatic control and dynamic optimization," *Optim Contr Appl Met*, vol. 32, no. 3, pp. 298–312, 2011.

[9] L. Grüne and J. Pannek, *Nonlinear model predictive control*. Springer, 2017.

[10] T. Faulwasser, T. Weber, P. Zometa, and R. Findeisen, "Implementation of nonlinear model predictive path-following control for an industrial robot," *IEEE T Contr Syst T*, vol. 25, no. 4, pp. 1505–1511, 2017.

[11] M. M. G. Ardakani, B. Olofsson, A. Robertsson, and R. Johansson, "Model predictive control for real-time point-to-point trajectory generation," *IEEE T Autom Sci Eng*, vol. 16, no. 2, pp. 972–983, 2018.

[12] M. Rubagotti, T. Taunyazov, B. Omarali, and A. Shintemirov, "Semi-autonomous robot teleoperation with obstacle avoidance via model predictive control," *IEEE Robot Autom Let*, vol. 4, no. 3, pp. 2746–2753, 2019.

[13] F. R. Hogan and A. Rodriguez, "Reactive planar non-prehensile manipulation with hybrid model predictive control," *Int J Robot Res*, vol. 39, no. 7, pp. 755–773, 2020.

[14] L. Dai, Y. Yu, D.-H. Zhai, T. Huang, and Y. Xia, "Robust model predictive tracking control for robot manipulators with disturbances," *IEEE T Ind Electron*, vol. 68, no. 5, pp. 4288–4297, 2021.

[15] M. Faroni, M. Beschi, C. G. L. Bianco, and A. Visioli, "Predictive joint trajectory scaling for manipulators with kinodynamic constraints," *Control Eng Pract*, vol. 95, p. 104264, 2020.

[16] A. Casalino, P. Rocco, and M. Prandini, "Hybrid control of manipulators in human-robot coexistence scenarios," in *American Contr Conf*, 2018, pp. 1172–1177.

[17] M. Faroni, M. Beschi, and N. Pedrocchi, "An MPC framework for online motion planning in human-robot collaborative tasks," in *IEEE Int Conf Emerging Tech and Factory Autom*, 2019, pp. 1555–1558.

[18] H. Ding, K. Wijaya, G. Reißig, and O. Stursberg, "Optimizing motion of robotic manipulators in interaction with human operators," in *Int Conf Intell Robot Appl*, 2011, pp. 520–531.

[19] B. Sadrfaridpour and Y. Wang, "Collaborative assembly in hybrid manufacturing cells: An integrated framework for human-robot interaction," *IEEE T Autom Sci Eng*, vol. 15, no. 3, pp. 1178–1192, 2017.

[20] P. Maurice, A. Malaisé, C. Amiot, N. Paris, G.-J. Richard, O. Rochel, and S. Ivaldi, "Human movement and ergonomics: An industry-oriented dataset for collaborative robotics," *Int J Robot Res*, vol. 38, no. 14, pp. 1529–1537, 2019.

[21] D. Q. Mayne and H. Michalska, "Receding horizon control of nonlinear systems," *IEEE T Autom Cont*, vol. 35, no. 7, pp. 814–824, 1990.

[22] A. Del Prete, "Joint position and velocity bounds in discrete-time acceleration/torque control of robot manipulators," *IEEE Robot Autom Lett*, vol. 3, no. 1, pp. 281–288, 2017.

[23] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpOASES: A parametric active-set algorithm for quadratic programming," *Math Progr Comp*, vol. 6, no. 4, pp. 327–363, 2014.