machines

MDPI

Article

An NMPC-ECBF Framework for Dynamic Motion Planning and Execution in Vision-Based Human–Robot Collaboration

Dianhao Zhang [1,2,*], Mien Van [2], Pantelis Sopasakis [2] and Seán McLoone [2]

[1] School of Electrical Engineering and Automation, Nantong University, Nantong 226019, China
[2] The Centre for Intelligent Autonomous Manufacturing Systems (i-AMS), School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Belfast BT9 5AG, UK; m.van@qub.ac.uk (M.V.); p.sopasakis@qub.ac.uk (P.S.); s.mcloone@qub.ac.uk (S.M.)
* Correspondence: dzhang07@qub.ac.uk

Abstract

To enable safe and effective human–robot collaboration (HRC) in smart manufacturing, it is critical to seamlessly integrate sensing, cognition, and prediction into the robot controller for real-time awareness, response, and communication inside a heterogeneous environment (robots, humans, and equipment). The proposed approach takes advantage of the prediction capabilities of nonlinear model predictive control (NMPC) to execute safe path planning based on feedback from a vision system. To satisfy the requirements of real-time path planning, an embedded solver based on a penalty method is applied. However, due to tight sampling times, NMPC solutions are approximate; therefore, the safety of the system cannot be guaranteed. To address this, we formulate a novel safety-critical paradigm that uses an exponential control barrier function (ECBF) as a safety filter. Several common human–robot assembly subtasks have been integrated into a real-life HRC assembly task to validate the performance of the proposed controller and to investigate whether integrating human pose prediction can help with safe and efficient collaboration. The robot uses OptiTrack cameras for perception and dynamically generates collision-free trajectories to the predicted target interactive position. Results for a number of different configurations confirm the efficiency of the proposed motion planning and execution framework, with a 23.2% reduction in execution time achieved for the HRC task compared to an implementation without human motion prediction.

Keywords: human-centric manufacturing; embedded path planning; safe human–robot collaboration; safety-critical control

check for updates

Academic Editors: Wenlong Li and Wei Xu

Received: 28 May 2025
Revised: 4 July 2025
Accepted: 11 July 2025
Published: 1 August 2025

Citation: Zhang, D.; Van, M.; Sopasakis, P.; McLoone, S. An NMPC-ECBF Framework for Dynamic Motion Planning and Execution in Vision-Based Human–Robot Collaboration. *Machines* **2025**, *13*, 672. https://doi.org/10.3390/machines13080672

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

With advancements in motion prediction and embedded path planning technologies, accurate and online human–robot collaboration (HRC) has been rapidly developed in smart manufacturing [1,2]. The learning space for executing HRC tasks with flexible joint manipulators is vast, characterized by nonlinear, time-varying, and highly complex dynamics. Ensuring safe and smooth HRC remains an open challenge, particularly for autonomous systems operating in spaces shared with humans, such as intra-logistics and service robotics [3,4]. To address these challenges, robots must be equipped with the ability to comprehend the ongoing task, precisely detect human positions, and anticipate future actions based on human demonstrations [5].

Vision-based human action recognition and motion prediction are crucial problems in an HRC system [6,7]. Action recognition aims to classify the categories of a human's current

dynamics. To activate a robot at the proper time, the estimated classification category can be used as a prior condition for motion prediction. Motion prediction is concerned with forecasting future body movements and poses based on observations of past movements and can be used to improve the collaboration efficiency in HRC [8]. Estimates of the future evolution of body poses can be used to define both the target position for the robot end-effector and the dynamic obstacles (body parts) for collision avoidance. In our work, an OptiTrack camera is used to achieve high-performance optical tracking combined with wearable devices to obtain a streaming skeletal representation of the human body. These data are then used as input to the task recognition and motion prediction modules.

Safety regulation ISO 10218 [9] and technical specification ISO 15066 [10] must be adhered to if a robot is deployed in a working environment shared with a human. These define cooperative robot operation as either contact-type (e.g., power and force limiting (PFL), hand guiding (HG)) or non-contact-type (e.g., safety-rated monitored stop (SRMS), speed and separation monitoring (SSM)) interactions. In this work, we consider the design of a control framework for SSM-based non-contact HRC, which guarantees safety by remaining a safe distance from humans during interactions.

A model predictive control (MPC)-based control system is employed to ensure compliance with safety standards while processing real-time feedback from a vision system. Given the dynamic and stochastic nature of HRC environments, which cannot be fully predicted a priori, MPC has gained increasing popularity in motion planning due to its ability to handle diverse kinematic and dynamic constraints [11–13]. The nonlinear model predictive control (NMPC) law is derived by solving an online nonlinear optimal control problem, incorporating a kinematic model. Previous work, such as [14], demonstrates the application of NMPC for task-constrained motion planning. For safe physical human–robot interaction (pHRI), ref. [15] further develops an online NMPC approach using a kinematic model with vision-based feedback.

NMPC can guarantee asymptotic stability and collision avoidance in most cases [16]. However, infeasible solutions might be obtained in some cases, especially within the stringent sampling time requirements encountered in HRC (typically a few milliseconds). This presents challenges for the real-time computation of NMPC, as there is the potential for collisions if an inaccurate solution is computed within the available time [17]. To overcome this limitation of NMPC, it is necessary to calculate the minimum distance between the end-effector and a human's interactive body component more accurately and to add a safety filter to constrain the robot's behavior within a safe area. To solve this problem, we use the Gilbert–Johnson–Keerth (GJK) algorithm [18] to solve the collision detection problem and add a control barrier function (CBF)-based constraint into the NMPC problem to guarantee the safety of the operator. In [19], a CBF-based approach is presented to constrain a redundant manipulator within a safe working area when interacting with a human operator. High-order CBF (HOCBF) and exponential CBF (ECBF) extensions are proposed in [20,21] to solve higher-order relative degree problems. In this paper, we combine a vision system with NMPC-ECBF to solve the safe pHRI problem online.

In summary, the existing research gaps are as follows. First, there is no established approach for designing a safe controller that uses the information from human motion prediction and action recognition. Second, collaboration under a tight sampling regime limits the accuracy of the NMPC solver, leading to errors that may present a risk to the human operators in the shared workspace.

With estimates of the future target and obstacle position in the working environment, we propose a real-time motion planning algorithm for the physical human-robot interaction (PHRI) task based on NMPC-ECBF. In this work, to test the performance of the proposed controller, we set up several HRC scenarios with a seven-degrees-of-freedom (DOFs)

manipulator (a Baxter robot) whereby motion capture, action recognition, motion prediction, and trajectory planning modules are integrated into a single system. The screwdriver usage scenario in the manufacturing assembly task includes subtasks such as picking up, moving forward, picking up a screw, operating, and moving backward. Each subtask is divided into a sequence of subtasks, with the robot end-effector required to reach a continuously updated predicted interactive position without collision when a subtask is triggered. However, as human motion and pose predictions are subject to approximation errors, the risk of collision still exists in HRC. Therefore, the output configuration of a manipulator is followed by a safety-critical controller implemented using an ECBF-based approach.

There have been several works [22] that consider integrating motion prediction into HRC tasks, but no corresponding safe controllers have been proposed to guarantee the safety of human operators. In this work, we focus on the NMPC-ECBF system to enable safe and real-time human–robot collaboration (HRC). The controller integrates a safety filter to enforce ISO safety regulations, with experimental validation demonstrating its performance in dynamic environments. This paper proposes a novel NMPC-ECBF framework for ensuring safety in real-time human-robot collaboration (HRC) scenarios. The developed controller integrates a safety-critical filter to guarantee strict compliance with ISO safety standards. Experimental results demonstrate the system's effectiveness in maintaining robust performance within dynamic operational environments. Furthermore, the implemented vision system provides crucial human action recognition and trajectory prediction capabilities, thereby validating the practical feasibility of incorporating motion prediction techniques in HRC applications. The main contributions of the paper are as follows.

(1) We integrate a safety filter based on ECBF into the predictive control system to constrain the movement of the end-effector, which guarantees human safety during collaboration. Additionally, a human motion prediction module is adopted to reduce the idle time for agents in HRC compared to the turn-taking collaboration.

(2) Since safe HRC requires an accurate estimate of the minimum distance between agents to ensure safe distance, a point-cloud-based perception system is proposed using a skeleton-aware mesh recovery system to capture human boundary information.

(3) We develop several real-life human–robot collaboration manufacturing scenarios to test the safety and efficiency of the proposed human motion prediction enhanced HRC control methodology.

This paper is organized as follows: Section 2 describes the overall HRC system. Related work is introduced in Section 3. The formulations and derivations of the target problem are presented in Section 4. Section 5 discusses the methodology for solving the problem, consisting of formulations for the system and obstacle detection, and introduces the NMPC algorithm and the solver used in the optimization problem. The controller design is introduced in Section 6. Then, Section 7 describes the experimental setup and presents the results of the simulations conducted to evaluate the performance of the proposed NMPC-ECBF-based control solution for human–robot collaboration. Finally, Section 8 provides the conclusions.

Notation: The following notation is used throughout the paper. Let $\mathbb{R}$, $\mathbb{R}_+$, $\mathbb{R}^n$, $\mathbb{R}^{m\times n}$ and $\mathbb{N}$ denote the sets of real numbers, the sets of positive real numbers, the set of real vectors of length n, the set of m by n real matrices, and non-negative integers, respectively. Let x, $\boldsymbol{x}$, and $\boldsymbol{X}$ denote scalar, vector and matrix quantities, respectively, with $\mathcal{X}$ a set, and X a constant. For any non-negative integers $k_1 < k_2$, the finite set $\{k_1, \ldots, k_2\}$ is denoted by $\mathbb{N}_{[k_1,k_2]}$. For $\boldsymbol{x} \in \mathbb{R}^n$, $[\boldsymbol{x}]_+ = \max\{0, \boldsymbol{x}\}$ is defined element wise. Given a set C, $InfC$ gives the maximum lower bound of the set C, while $supC$ gives the least upper bound of the set C.

2. Overall System Design

The overall design of the collision-free HRC system is shown in Figure 1. The proposed HRC system comprises motion capture, action recognition, motion prediction, decision-making, path planning, and robot controller modules. The motion capture module relies on OptiTrack cameras to generate high-quality 3D streaming skeleton pose estimates at a sampling rate of 20 frames per second. Using recorded spatial-temporal data for a human performing task-related motions/activities as training data, a motion prediction module is pretrained to predict the future one-second evolution of human poses (20 frames) based on the motion observed over the previous one-second interval. The decision-making module is implemented using NMPC-ECBF-based motion planning and robot execution mechanisms. In our work, a robot starts from a random position in the working environment and continuously searches for a collision-free path to the estimated interaction coordinates based on the received one-second ahead pose estimation feedback provided by the predictor module.

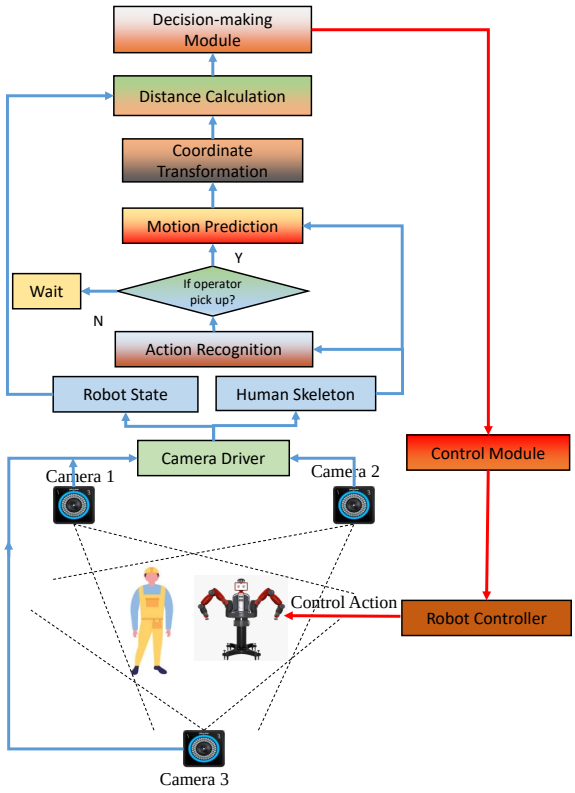


Figure 1. Proposed human–robot collaboration system design.

3. Related Works

3.1. Methods to Solve Trajectory Optimization in MPC

When the system is linear, the constraints are affine and the cost functions are quadratic, the associated MPC optimization problem is a quadratic program. Several convex optimization algorithms have been proposed [23,24] that are fast, robust, and possess global convergence guarantees. In contrast to the first-order approaches mentioned above, Newton-type methods are able to deal with general constraints and have faster rates of convergence [25]. This leads to a considerable reduction in computation time and makes them better suited to handling real-time nonlinear problems.

The proximal averaged Newton-type method for optimal control (PANOC) is applied, considering obstacle avoidance [26]. Compared with SQP and IP, PANOC does not require the solution of linear systems or quadratic programming problems at every iteration, and it involves only very simple operations such as vector additions and scalar and inner

products. Additionally, PANOC is able to converge globally to a point that satisfies the first-order optimal conditions of the problem from any initial guess. As an extension, refs. [27,28] combines the proximal averaged Newton-type method for optimal control (PANOC) with the penalty and augmented Lagrangian methods to compute approximate stationary points of non-convex problems.

Although MPC is widely used in practice, it is known that suboptimality and infeasibility caused by the numerical optimization method can compromise the stability and positive invariance (satisfaction of constraints) properties of the MPC-controlled system. There have been a few works, such as [29], that attempt to remedy this issue, but these are limited to linear dynamical systems, convex formulations, and specific numerical optimization methods. To the best of the authors' knowledge, no results exist on the problem of NMPC stability under inexact numerical optimization. In this work, we focus on the issue of collision avoidance and demonstrate that the combination of nonlinear MPC, PANOC, and a CBF-based framework leads to a significant improvement in safety in terms of obstacle avoidance.

3.2. Safety-Critical Manipulator Control

A system is commonly defined as safe when its state remains within a chosen set, known as the safety set. This forms the basis of controlled invariance [30], that is, finding a control strategy that ensures the system always remains in the safety set. In [30], a safety filter is used in the control of a nonlinear system to restrict the desired inputs in a way that ensures the safety of the system when necessary. The method proposed in [31] formulates robust control Lyapunov and barrier functions to provide guarantees of stability and safety in the presence of model uncertainty. A control barrier function (CBF) is applied in our work to solve the collision avoidance problem. Applications of the CBF approach to the control of a redundant manipulator can be found in [32,33], which reduces the model dependence and conservativeness. As an extension, Ding et al. [34] present a purely kinematic implementation of a velocity-based CBF to match the manipulator's configuration space with the obstacles' space, thereby improving safety when using the CBF-based QP motion control.

4. Problem Formulation

Formally, the skeleton-format input to the action recognition and motion prediction modules is represented by 3D joint positions of m joints for a sequence of frames recorded over time, which is the same format as in [35]. The action pose at time (frame) t is $\boldsymbol{p}(t) \in \mathbb{R}^{3m}$. The observed time length is $t \in [1, \tau_o]$ and the unobserved time length is $t \in [\tau_o + 1, \tau_o + \tau_p]$ for prediction.

For a single human action sequence belonging to a set of N_a possible actions, $\mathcal{A} = \{1, \ldots, N_a\}$, we have the observed and future motion matrices, $\boldsymbol{P}_{\text{prev}}$ and $\boldsymbol{P}_{\text{fut}}$, where $\boldsymbol{P}_{\text{prev}} = [\boldsymbol{p}(1), \cdots, \boldsymbol{p}(\tau_o)] \in \mathbb{R}^{\tau_o \times 3m}$ and $\boldsymbol{P}_{\text{fut}} = [\hat{\boldsymbol{p}}(\tau_o + 1), \cdots, \hat{\boldsymbol{p}}(\tau_o + \tau_p)] \in \mathbb{R}^{\tau_p \times 3m}$ and the action classification label encoded as a one-hot vector $\boldsymbol{y} \in \{0, 1\}^{N_a}$. The predicted human pose at time t is denoted as $\hat{\boldsymbol{p}}(t) \in \mathbb{R}^{3m}$, which can be divided into the right hand's position $\hat{\boldsymbol{p}}_{rh}(t) \in \mathbb{R}^3$ as the target position for the robot and the remaining body joints' positions $\hat{\boldsymbol{p}}_o(t) \in \mathbb{R}^{3(m-1)}$ to identify the positions of obstacles.

Human action recognition and motion prediction are achieved using the two-stage deep learning computer vision approach introduced in [35], as depicted in Figure 2. The 3D-skeleton data stream provided by the OptiTrack motion capture system is first processed by a bidirectional LSTM and a GCN architecture incorporating the attention mechanism to extract spatial-temporal features of the human motion dynamics. The generated feature map, denoted as $\boldsymbol{\Psi}$, is then fed simultaneously to two modules, one that performs prediction

and one that performs recognition. The motion prediction task is achieved using an LSTM-based decoder, while the action recognition task is performed using a CNN followed by conditional random fields (CRF). Initially, the models are trained offline using collected HRC task samples. Following training, streaming data are fed to the model to generate the predicted future pose sequence $\hat{\boldsymbol{P}}_{\text{fut}}$ and predicted action category $\hat{y}$. Denoting the feature extraction, motion prediction and action recognition functions within the Deep Neural Network model as f_{ext}, f_{pred} and f_{recg}, respectively, and $\boldsymbol{\theta}_{\text{ext}}$, $\boldsymbol{\theta}_{\text{pred}}$ and $\boldsymbol{\theta}_{\text{recg}}$ as their trainable parameters, the extracted feature map $\boldsymbol{\Psi}$, the estimated future motion $\hat{\boldsymbol{P}}_{\text{fut}}$ and one-hot vector class category $\hat{\boldsymbol{y}}$ can be expressed as

$$\boldsymbol{\Psi} = f_{\text{ext}}(\boldsymbol{P}_{\text{prev}}; \boldsymbol{\theta}_{\text{ext}}), \tag{1a}$$

$$\hat{\boldsymbol{P}}_{\text{fut}} = f_{\text{pred}}(\boldsymbol{\Psi}; \boldsymbol{\theta}_{\text{pred}}), \tag{1b}$$

$$\hat{\boldsymbol{y}} = f_{\text{recg}}(\boldsymbol{\Psi}; \boldsymbol{\theta}_{\text{recg}}). \tag{1c}$$

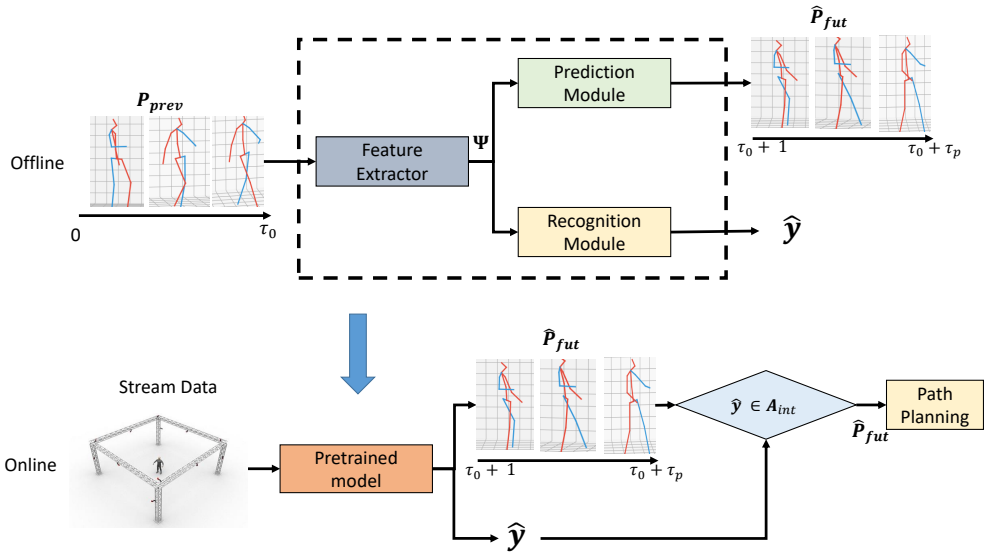


Figure 2. Work flow from non-local graph convolution (NGC) based vision system to path planning. The outputs from the recognition and prediction modules are the estimated human action category and the future pose sequence, respectively.

If the predicted action is one that requires robot interaction, the predicted final position of the right hand (denoted $\hat{\boldsymbol{p}}_{\text{rh}}(\tau_p)$) is set as the target interactive position for the robot end-effector and the predicted final position of the other joints are set as the center of capsule-format obstacles.

The motion planning and execution control problem can then be stated as follows: Given the predicted future human motion sequence $\hat{\boldsymbol{P}}_{\text{fut}}$ and the initial position of the robot arm, and assuming that:

- the dynamics of the manipulator are known;
- the error in the prediction of the future human position is bounded;
- the maximum velocity of the human hand is slower than that of the manipulator;

then the objective is to determine a torque input $\boldsymbol{\tau}_{\text{act}}$ to the robot, which enables it to track the predicted position of the human's right hand $\hat{\boldsymbol{p}}_{\text{rh}}(t)$ to the target position and pose, while at all times avoiding collision with the human's body.

Denoting the set of interactive human actions requiring robot interaction as $\mathcal{A}_{\text{int}}$, where $\mathcal{A}_{\text{int}} \subset \mathcal{A}$, the motion prediction and path planning tasks are triggered if the predicted HRC task $\hat{y} \in \mathcal{A}_{\text{int}}$, as shown in Figure 1.

5. NMPC Motion Planning Scheme

5.1. System Description

To model the n-DOF robotic manipulator (chosen as the right arm of our Baxter robot) with joint positions $\boldsymbol{q} \in \mathbb{R}^n$, we consider an input-affine nonlinear system of the form

$$\dot{\boldsymbol{\chi}} = \mathfrak{f}(\boldsymbol{\chi}) + \mathfrak{g}(\boldsymbol{\chi})\boldsymbol{\tau}, \tag{2}$$

where $\boldsymbol{\chi} = [\boldsymbol{q}, \dot{\boldsymbol{q}}] \in \mathbb{R}^{2n}$ is the the state of the system and $\boldsymbol{\tau} \in \mathbb{R}^n$ is the control input. Functions $\mathfrak{f} : \mathbb{R}^{2n} \to \mathbb{R}^{2n}$ and $\mathfrak{g} : \mathbb{R}^{2n} \to \mathbb{R}^{2n\times n}$ are continuously differentiable in $\boldsymbol{\chi}$.

Consider an n-degrees-of-freedom manipulator with equations of motion defined as

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}, \tag{3}$$

$$\boldsymbol{x} = \boldsymbol{f}_{\text{fwd}}(\boldsymbol{q}), \tag{4}$$

where $\boldsymbol{M}(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$ is a symmetric positive definite matrix, $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^{n\times n}$ is a matrix of Coriolis and centrifugal forces terms, $\boldsymbol{g}(\boldsymbol{q}) \in \mathbb{R}^n$ is a vector of gravitational forces, $\boldsymbol{q} \in \mathbb{R}^7$ are the joint positions, and $\boldsymbol{f}_{\text{fwd}} : \mathbb{R}^7 \to \mathbb{R}^3$ is the kinematic transformation from the joint angle positions $\boldsymbol{q}$ to the Cartesian position of the end-effector $\boldsymbol{x} \in \mathbb{R}^3$. We consider the joint torque $\boldsymbol{\tau} \in \mathbb{R}^7$ as the system input and $\boldsymbol{x}$ as the system output.

Defining $\boldsymbol{J} = \boldsymbol{J}(\boldsymbol{q}) = \frac{\partial f_{fwd}}{\partial \boldsymbol{q}}$ as the Jacobian matrix, it follows that

$$\dot{\boldsymbol{x}} = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}, \tag{5a}$$

$$\ddot{\boldsymbol{x}} = \boldsymbol{J}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}(\boldsymbol{q})\dot{\boldsymbol{q}}. \tag{5b}$$

By substituting (5a) into (3), the Cartesian robotic system dynamics can be expressed as

$$\boldsymbol{M}_x(\boldsymbol{q})\ddot{\boldsymbol{x}} + \boldsymbol{C}_x(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{x}} + \boldsymbol{g}_x(\boldsymbol{q}) = \boldsymbol{f}_h, \tag{6}$$

where $\boldsymbol{M}_x(\boldsymbol{q}) = \boldsymbol{J}^{\dagger\intercal}\boldsymbol{M}(\boldsymbol{q})\boldsymbol{J}^{\dagger}$, $\boldsymbol{C}_x(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{J}^{\dagger\intercal}(\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) - \boldsymbol{M}(\boldsymbol{q})\boldsymbol{J}^{\dagger}\dot{\boldsymbol{J}})\boldsymbol{J}^{\dagger}$, $\boldsymbol{g}_x(\boldsymbol{q}) = \boldsymbol{J}^{\dagger\intercal}\boldsymbol{g}(\boldsymbol{q})$ and $\boldsymbol{f}_h = \boldsymbol{J}^{\dagger}\boldsymbol{\tau}$ is the vector of generated control forces. $\boldsymbol{J}^{\dagger} = \boldsymbol{J}^{\dagger}(\boldsymbol{q})$ denotes the pseudo-inverse of $\boldsymbol{J}(\boldsymbol{q})$, that is:

$$\boldsymbol{J}^{\dagger}(\boldsymbol{q}) = (\boldsymbol{J}^{\intercal}(\boldsymbol{q})\boldsymbol{J}(\boldsymbol{q}))^{-1}\boldsymbol{J}^{\intercal}(\boldsymbol{q}). \tag{7}$$

For notational simplicity, $\boldsymbol{M}_x$, $\boldsymbol{C}_x$, $\boldsymbol{g}_x$ will be used to denote $\boldsymbol{M}_x(\boldsymbol{q})$, $\boldsymbol{C}_x(\boldsymbol{q}, \dot{\boldsymbol{q}})$, $\boldsymbol{g}_x(\boldsymbol{q})$, respectively. The system (2) is converted from joint space to task space, which can be expressed as

$$\begin{bmatrix} \dot{\boldsymbol{x}} \\ \ddot{\boldsymbol{x}} \end{bmatrix} = \underbrace{\begin{bmatrix} \dot{\boldsymbol{x}} \\ -\boldsymbol{M}_x^{-1}(\boldsymbol{C}_x\dot{\boldsymbol{x}} + \boldsymbol{g}_x) \end{bmatrix}}_{f(\boldsymbol{x},\dot{\boldsymbol{x}})} + \underbrace{\begin{bmatrix} 0 \\ -\boldsymbol{M}_x^{-1} \end{bmatrix}}_{g(\boldsymbol{x})} \boldsymbol{f}_h. \tag{8}$$

Thus, the robotic system (8) is equivalent to the nonlinear affine system (2) with respect to the input $\boldsymbol{f}_h$. Therefore, the ECBF can be applied to the robotic system (8) to deal with the time-varying output constraints.

5.2. Minimum Distance Calculation

The minimum distance is the main input for most collision avoidance, HRI, robot decision-making, and robot navigation methods [36]. Given the joint coordinates provided by the skeletal tracking algorithm and the point clouds on the boundary of the human

body, bounding capsules (cylinders with semi-spherical ends) can be computed for the human body compo as a simplified representation of the human in the distance calculation. We generate minimum bounding capsules along the bones in the skeleton, as shown in Figure 3. This is repeated for the predicted skeleton in each frame over the prediction horizon. For the 32-joint skeleton model representation, $\hat{\boldsymbol{p}}_{\text{rh}}$ is the position of the center of the right hand.

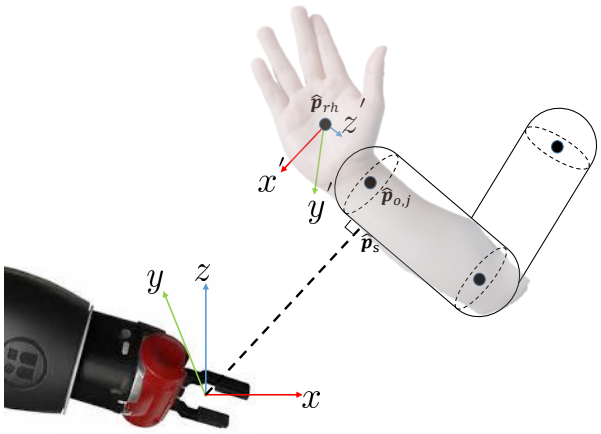


Figure 3. Calculation of the distance, position, and orientation of the capsules for the representation of humans.

We chose the capsules model to represent robot links since it considers both the skeleton and the boundary of the human, which models the non-geometric shape of the human body [37]. The 32-joint skeleton model, as used in the Human 3.6 M dataset [38], maps to a total of 15 capsules for the human body. The approach to measure the distance between capsules is depicted in Figure 3. The distance calculation is implemented using the GJK (Gilbert–Johnson–Keerthi) algorithm [18].

The algorithm returns $\lambda(t)$, the minimum distance between the boundary of the capsules representing the human and the robot in the HRC task at time t.

5.3. Nonlinear Model Predictive Control for Collision Avoidance

The robot's objective is for its end-effector to follow the estimated position of the human's right hand while avoiding collisions. To that end, we propose an NMPC formulation as a high-level control system that commands angular velocity references to a low-level control system (see Section 6.2). The overall controller architecture is shown in Figure 4.

The input of the high-level control system is the joint velocity commands $\boldsymbol{u}(t)$. At every sampling time instant, we need to solve a finite-horizon optimal control problem with prediction horizon $T_p > 0$, stage cost function ℓ, and terminal cost function ℓ_f, where ℓ and ℓ_f are given by

$$\ell(\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}}) = e(\hat{\boldsymbol{p}}_{\text{rh}}, \hat{\boldsymbol{x}})^{\intercal} \boldsymbol{R}_p e(\hat{\boldsymbol{p}}_{\text{rh}}, \hat{\boldsymbol{x}}) + \hat{\boldsymbol{u}}^{\intercal} \boldsymbol{R}_v \hat{\boldsymbol{u}}, \tag{9}$$

and

$$\ell_f(\hat{\boldsymbol{x}}) = \boldsymbol{e}(\hat{\boldsymbol{p}}_{\text{rh}}(T_p), \hat{\boldsymbol{x}}(T_p))^{\intercal} \boldsymbol{Q}_p \boldsymbol{e}(\hat{\boldsymbol{p}}_{\text{rh}}(T_p), \hat{\boldsymbol{x}}(T_p)) + \hat{\boldsymbol{u}}^{\intercal}(T_p) \boldsymbol{Q}_v \hat{\boldsymbol{u}}(T_p), \tag{10}$$

respectively. Here, $\boldsymbol{e}(\hat{\boldsymbol{p}}_{rh}(T_p), \boldsymbol{x}(t))$ is an error metric between the given Cartesian goal $\hat{\boldsymbol{p}}_{rh}(T_p)$ and the current Cartesian state of the robot $\boldsymbol{x}(t)$. The reference position $\hat{\boldsymbol{p}}_{\text{rh}}(T_p)$ is obtained by the motion predictor. $\boldsymbol{Q}_p$ and $\boldsymbol{Q}_v$ are constant weight matrices for terms in the

terminal cost function, while $\boldsymbol{R}_p$ and $\boldsymbol{R}_v$ are the weight matrices for terms in the stage cost function. Following [39], the error function $e(\boldsymbol{x}_d, \hat{\boldsymbol{x}}(t))$ is defined as

$$\boldsymbol{e}(\hat{\boldsymbol{p}}_{\mathrm{rh}}(t), \boldsymbol{x}(t)) = \begin{bmatrix} -\boldsymbol{I}_3 & 0 \\ 0 & \boldsymbol{\Pi}_d \end{bmatrix} \hat{\boldsymbol{x}}(t) + \begin{bmatrix} \hat{\boldsymbol{p}}_{\mathrm{rh}}(t) \\ 0 \end{bmatrix}, \tag{11}$$

where matrix $\boldsymbol{\Pi}_d$ is built from the desired quaternion, which remains constant during the optimization.

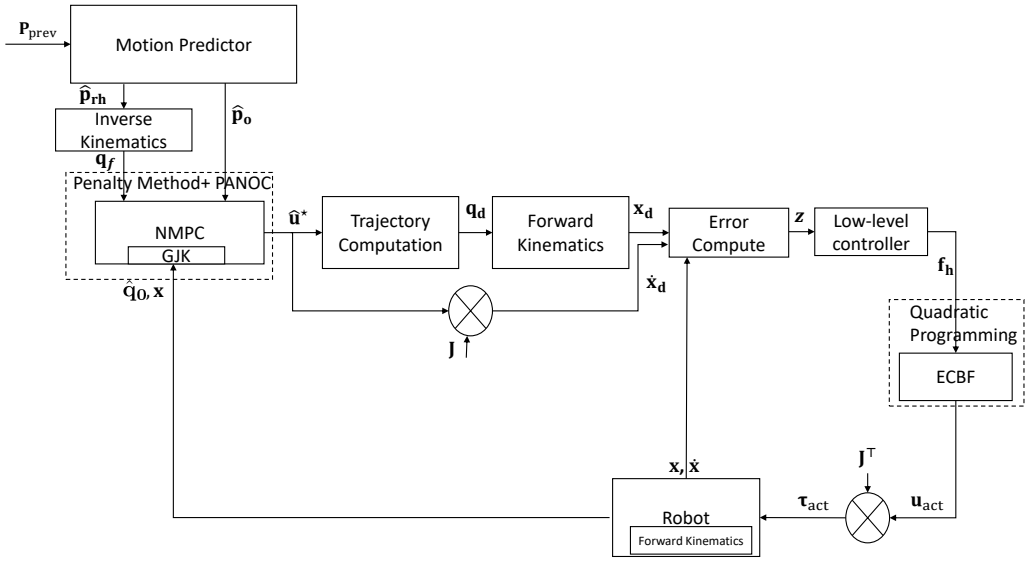


Figure 4. NMPC-based motion planning system.

The initial and final configurations of the robot $\boldsymbol{q}_0 = [q_{01}, \ldots, q_{07}]$ and $\boldsymbol{q}_f = [q_{f1}, \ldots, q_{f7}]$ are obtained using the inverse kinematics function $\boldsymbol{f}_{\mathrm{inv}} : \mathbb{R}^6 \to \mathbb{R}^7$, which converts the Cartesian end effector poses (position and orientation) to the corresponding joint positions. Using the GJK algorithm, the minimum distance calculation at time t is performed based on the Cartesian coordinates of the robot, $\boldsymbol{x}(t)$, and the predicted human pose, $\hat{\boldsymbol{p}}_o(t)$. The distance between the end-effector and the predicted operator's position $\lambda(\hat{\boldsymbol{x}}(t); \hat{\boldsymbol{p}}_o(t))$ must be greater than the minimum safe distance d_{safe}. The value of d_{safe} is chosen to be greater than the maximum prediction error bound to guarantee the safety of humans. The constraints are described as

$$\dot{\hat{\boldsymbol{q}}}(t) = \hat{\boldsymbol{u}}(t), t \in [0, T_p], \tag{12a}$$
$$\hat{\boldsymbol{q}}(0) = \boldsymbol{q}_0, \tag{12b}$$
$$\hat{\boldsymbol{x}}(t) = \boldsymbol{f}_{\mathrm{fwd}}(\hat{\boldsymbol{q}}(t)), t \in [0, T_p], \tag{12c}$$
$$\lambda(\hat{\boldsymbol{x}}(t); \hat{\boldsymbol{p}}_o(t)) \geq d_{\mathrm{safe}}, t \in [0, T_p], \tag{12d}$$

where $\boldsymbol{q} = (q_1, \ldots, q_7)$ denotes the state of the robotic arm. The control actions are constrained in the set $\mathcal{U}$. Then, the optimal solution $\hat{\boldsymbol{u}}^\star$ is obtained by minimizing the optimal control cost function, that is,

$$\underset{\hat{\boldsymbol{u}}:[0,T_p]\to\mathcal{U}}{\text{minimize}}\, \ell_f(\hat{\boldsymbol{x}}(T_p)) + \int_0^{T_p} \ell(\hat{\boldsymbol{x}}(t), \hat{\boldsymbol{u}}(t))\mathrm{d}t, \tag{13}$$

subject to the constraints (12).

In this work, we solve the NMPC problem using a numerical optimization method. To that end, we convert the above problem to a discrete-time optimal control problem using an explicit discretization and integration method. Assuming that $T_p = N_h T_s$, the NMPC problem in Equation (13) is reformulated to $\mathbb{P}_d$ and depends on the current state $\boldsymbol{q}_0$, the set-point $\boldsymbol{q}_f$ for the joint positions, input $\boldsymbol{u}$ and the estimated obstacle distance $\hat{\lambda}$ computed by the GJK algorithm as follows:

$$\mathbb{P}_d(\hat{\boldsymbol{q}}_0, \hat{\boldsymbol{p}}_o, \boldsymbol{q}_f) : \underset{\hat{\boldsymbol{u}}_0,\ldots,\hat{\boldsymbol{u}}_{N_h-1}\in\mathcal{U}}{\text{minimize}} \ \ell_{d,f}(\hat{\boldsymbol{x}}_{N_h}) \sum_{k=0}^{N_h-1} \ell_d(\hat{\boldsymbol{x}}_k, \hat{\boldsymbol{u}}_k), \tag{14a}$$

subject to the constraints

$$\hat{\boldsymbol{q}}_{k+1} = \hat{\boldsymbol{q}}_k + T_s\hat{\boldsymbol{u}}_k, k \in \mathbb{N}_{[0,N_h-1]}, \tag{14b}$$

$$\hat{\boldsymbol{q}}_0 = \boldsymbol{q}_0, \tag{14c}$$

$$\hat{\boldsymbol{q}}_{N_h} = \boldsymbol{q}_f, \tag{14d}$$

$$\hat{\boldsymbol{x}}_k = \boldsymbol{f}_{\text{fwd}}(\hat{\boldsymbol{q}}_k), k \in \mathbb{N}_{[0,N_h-1]}, \tag{14e}$$

$$\lambda(\hat{\boldsymbol{x}}_k; \hat{\boldsymbol{p}}_{o,k}) \geq d_{\text{safe}}, k \in \mathbb{N}_{[0,N_h-1]}. \tag{14f}$$

By solving $\mathbb{P}_d$ at every discrete time instant, we obtain the solution $\hat{\boldsymbol{u}}_k^\star$, $k \in \mathbb{N}_{[0,N_h-1]}$, only the first value of which, $\hat{\boldsymbol{u}}_0^\star$, is applied to the system. $\mathbb{P}_d$ is solved again at the subsequent time instant in a receding horizon fashion.

5.4. Numerical Solution

To solve $\mathbb{P}_d$ numerically, we employ OpEn [27], a code generation software package that generates optimizers in the Rust programming language for deployment on embedded devices. OpEn combines the PANOC numerical algorithm with the penalty method to solve the above problem. However, OpEn requires that we cast $\mathbb{P}_d$ in the following general form:

$$\underset{\boldsymbol{u}\in\tilde{\mathcal{U}}}{\text{minimize}} \ F_0(\boldsymbol{u}), \tag{15a}$$

$$\text{subject to: } \boldsymbol{F}_i(\boldsymbol{u}) = \boldsymbol{0}, i \in \mathbb{N}_{[1,n_c]}, \tag{15b}$$

where $\boldsymbol{u} \in \mathbb{R}^n$ is the decision variable, $\tilde{\mathcal{U}} \subseteq \mathbb{R}^n$ is a nonempty closed set on which it is easy to compute projections, $F_0 : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable cost function—not necessarily convex—with the Lipschitz gradient, and $\boldsymbol{F}_i : \mathbb{R}^n \to \mathbb{R}^{m_i}$ are functions such that $\|\boldsymbol{F}_i\|^2$ is continuously differentiable with the Lipschitz gradient.

To cast problem $\mathbb{P}_d$ in this form, we choose the decision variable to be $\boldsymbol{u} = (\hat{\boldsymbol{u}}_0, \hat{\boldsymbol{u}}_1, \ldots, \hat{\boldsymbol{u}}_{N-1})$ and define the sequence of functions

$$\boldsymbol{\Phi}_{k+1}(\boldsymbol{u}, \boldsymbol{x}) = \begin{bmatrix} \boldsymbol{q}_0 + T_s \sum_{t=0}^{k} \boldsymbol{u}_k \\ \boldsymbol{f}_{\text{fwd}}\left(\boldsymbol{q}_0 + T_s \sum_{t=0}^{k} \boldsymbol{u}_k\right), \end{bmatrix} \tag{16}$$

with $\boldsymbol{\Phi}_0(\boldsymbol{u}, \boldsymbol{x}) = [\boldsymbol{q}^\intercal \ \ \boldsymbol{x}^\intercal]^\intercal$. Note that $\boldsymbol{\Phi}_{k+1}(\boldsymbol{u}, \boldsymbol{x})$ contains the location of the robot in both joint space and task space at time $k+1$ as a function of the initial state and the sequence of control actions $\boldsymbol{u}$. This allows us to eliminate the sequence of states and define the cost function,

$$F_0(\boldsymbol{u}, \boldsymbol{x}) = \ell_{d,f}(\boldsymbol{\Phi}_{N_h}(\boldsymbol{u}, \boldsymbol{x})) + \sum_{k=0}^{N_h-1} \ell_d(\boldsymbol{\Phi}_k(\boldsymbol{u}, \boldsymbol{x}), \boldsymbol{u}_k). \tag{17}$$

In order to impose the terminal condition of Equation (14d), we define

$$\boldsymbol{F}_1(\boldsymbol{u}) = \hat{\boldsymbol{q}}_{N_h} - \boldsymbol{q}_f, \tag{18}$$

where $\hat{\boldsymbol{q}}_{N_h} = \hat{\boldsymbol{q}}_{N_h}(\boldsymbol{u}, \boldsymbol{x})$. To impose the obstacle avoidance condition of Equation (14f), we define

$$\boldsymbol{F}_{k+1}(\boldsymbol{u}) = [d_{\text{safe}} - \lambda(\boldsymbol{\Phi}_k(\boldsymbol{u}, \boldsymbol{x}); \hat{\boldsymbol{p}}_{o,k})]_+, \tag{19}$$

for $k \in \mathbb{N}_{[0,N_h-1]}$. Finally, we choose $\tilde{\mathcal{U}} = \mathcal{U}^{N_h}$ for the input constraints.

OpEn solves a series of optimization problems of the form

$$\mathbb{P}_{\text{in}} : \underset{\boldsymbol{u}\in U}{\text{minimize}}\, F_0(\boldsymbol{u}) + c \underbrace{\sum_{i=1}^{n_c} \|\boldsymbol{F}_i(\boldsymbol{u})\|^2}_{\text{infeasibility}}, \tag{20}$$

for some $c > 0$ termed the *inner* problems. Essentially, the hard constraints of the original problem are converted into soft constraints. Starting with an initial value of c, a sequence of inner problems is solved numerically using the PANOC method while driving c to infinity and warm-starting each next inner problem with the solution of the previous one [27]. The algorithm terminates once the infeasibility, as defined in Equation (20), drops below a given tolerance, $\epsilon_{\text{infeas}} > 0$.

6. Controller Design

6.1. Trajectory Computation

Given the estimated angular velocity command $\hat{\boldsymbol{u}}^\star$, the desired joint state $\boldsymbol{q}_d$ is updated as

$$\boldsymbol{q}_{d,k+1} = \boldsymbol{q}_{d,k} + T_s \hat{\boldsymbol{u}}^\star. \tag{21}$$

Then, the generated desired joint position is used to calculate the desired Cartesian position $\boldsymbol{x}_d$ and velocity $\dot{\boldsymbol{x}}_d$.

6.2. Low-Level Controller

To achieve asymptotic stability, the control force $\boldsymbol{f_h}$ input to the safety filter is defined as

$$\boldsymbol{f_h} = \boldsymbol{C_x}(\dot{\boldsymbol{x}}_d - \Lambda\tilde{\boldsymbol{x}}) + \boldsymbol{g_x} + \boldsymbol{M_x}(\ddot{\boldsymbol{x}}_d - \Lambda\dot{\tilde{\boldsymbol{x}}}) - k_z \text{sign}(\boldsymbol{z}), \tag{22}$$

where Λ and k_z are positive constants. A composite state error is denoted as $\boldsymbol{z} = \dot{\tilde{\boldsymbol{x}}} + \Lambda\tilde{\boldsymbol{x}}$, where $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{x}_d$ and $\dot{\tilde{\boldsymbol{x}}} = \dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}_d$ are the error of the Cartesian position and velocity, respectively. $\text{sign}(\boldsymbol{z})$ is a sign function depends on the state error. Considering the chattering problem caused by the sign function, (22) is rewritten as

$$\boldsymbol{f_h} = \boldsymbol{C_x}(\dot{\boldsymbol{x}}_d - \Lambda\tilde{\boldsymbol{x}}) + \boldsymbol{g_x} + \boldsymbol{M_x}(\ddot{\boldsymbol{x}}_d - \Lambda\dot{\tilde{\boldsymbol{x}}}) - k_z \frac{\boldsymbol{z}}{\|\boldsymbol{z}\| + c_1}, \tag{23}$$

where c_1 is a small positive number.

6.3. Exponential Control Barrier Function as a Safety Filter

In this section, an ECBF-based safety filter is introduced to prevent collisions with the human operator by constraining the trajectory generated by NMPC to be in a safe zone. ECBFs are formally defined as follows:

Definition 1 (ECBF [40]). *Consider the system (8) and a set* $\mathcal{C} = \{\boldsymbol{x} \in \mathbb{R}^3 | h(t, \boldsymbol{x}) \geq 0, \forall t > 0\}$ *defined by a continuously differentiable function* $h(t, \boldsymbol{x}) : \mathbb{R}^3 \to \mathbb{R}$ *with relative degree* $\kappa \geq 2$. *Then,* $h(t, \boldsymbol{x})$ *is an exponential control barrier function (ECBF) if there is a* $\boldsymbol{k}_b \in \mathbb{R}^{\kappa-1}$, *such that*

$$\inf_{\boldsymbol{u} \in \mathcal{U}} [L_f^{\kappa} h(t, \boldsymbol{x}) + L_g L_f^{\kappa-1} h(t, \boldsymbol{x}) \boldsymbol{u} + \boldsymbol{k}_b \boldsymbol{\zeta}_{\boldsymbol{b}}(\boldsymbol{x}) \geq 0, \tag{24}$$

where $\boldsymbol{\zeta_b}$ *can be written as*

$$\boldsymbol{\zeta}_b(\boldsymbol{x}) := \begin{bmatrix} h(\boldsymbol{x}) \\ \dot{h}(\boldsymbol{x}) \\ \vdots \\ h^{(\kappa-1)}(\boldsymbol{x}) \end{bmatrix} = \begin{bmatrix} h(\boldsymbol{x}) \\ L_f h(\boldsymbol{x}) \\ \vdots \\ L_f^{(\kappa-1)} h(\boldsymbol{x}) \end{bmatrix}. \tag{25}$$

The method for selecting $\boldsymbol{k}_b$ *is detailed in [40]. The Lie derivative of* $h(\boldsymbol{x})$ *along* $f(\boldsymbol{x})$ *is defined as*

$$L_f h(t, \boldsymbol{x}) = \frac{\partial h(t, \boldsymbol{x})}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}, \dot{\boldsymbol{x}}), \tag{26}$$

where $\boldsymbol{x}$ *is the state of the system. The Lie derivative of* $h(\boldsymbol{x})$ *along* $\boldsymbol{g}(\boldsymbol{x})$ *is defined as* $L_g h(t, \boldsymbol{x})$.

Denote $\boldsymbol{p}_{oi}(t) \in \mathbb{R}^3$ as the support point (i.e., a point on the boundary of the convex set that is closest to the robot) on the convex set representing the area of the human obstacle for the i^{th} robot joint computed by GJK at time t. To ensure the robot does not collide with the moving human, the Cartesian position of the robot joints $\boldsymbol{x}_i \in \mathbb{R}^3$, $i \in \mathbb{N}_{[1,7]}$ are under the following constraints

$$\|\boldsymbol{x}_i(t) - \hat{\boldsymbol{p}}_s(t)\|_2 > d_{\text{safe}}, \text{ for } i \in \mathbb{N}_{[1,7]}, \ t \geq 0, \tag{27}$$

where $\hat{\boldsymbol{p}}_s$ is a support point on the boundary of the capsule with the minimum distance to the i^{th} robot joint, as shown in Figure 3. We define a candidate ECBF $h_i(t, \boldsymbol{x}_i)$ as

$$h_i(t, \boldsymbol{x}_i) = \|\boldsymbol{x}_i(t) - \hat{\boldsymbol{p}}_s(t)\|_2^2 - d_{\text{safe}}^2. \tag{28}$$

To modify the applied forces, $\boldsymbol{f_h}$, to guarantee safety, a quadratic programming (QP)-based safety filter is employed. Considering the position constraints with relative degree 2, the optimization problem can be formulated as minimizing the normed difference between $\boldsymbol{f_h}$ and the actual input $\boldsymbol{u}_{\text{act}}$, in order to maintain the robot within the safety region. In other words, $\boldsymbol{u}_{\text{act}}(\boldsymbol{x}, t)$ is the control input that minimizes the following QP:

$$\underset{\boldsymbol{u}_{\text{act}} \in \mathcal{U}}{\text{minimize}} \ \|\boldsymbol{f_h} - \boldsymbol{u}_{\text{act}}\|^2, \tag{29a}$$

subject to

$$\begin{aligned} &L_f^2 h_i(t, \boldsymbol{x}) + L_g L_f h_i(t, \boldsymbol{x}) \boldsymbol{u}_{\text{act}} + k_1 h_i(t, \boldsymbol{x}) \\ &\quad + k_2 L_f h_i(t, \boldsymbol{x}) \geq 0, i \in \mathbb{N}_{[1,7]}, \end{aligned} \tag{29b}$$

where $h_i(t, \boldsymbol{x})$ is as defined in (28). The robot joint torques are then obtained as $\boldsymbol{\tau}_{\text{act}} = \boldsymbol{J}^{\intercal}(\boldsymbol{q}) \boldsymbol{u}_{\text{act}}$. Further details on the implementation of ECBFs are given in [40].

6.4. System Stability Analysis

Proving the global stability of our ECBF-NMPC dynamic motion planning framework is challenging as it incorporates approximate numerical methods, a neural network-based motion predictor, the GJK algorithm, and a safety filter. To guarantee local asymptotic stability, a control Lyapunov constraint is incorporated into the ECBF formulation and activated when the robot is close to the destination.

Property 1 ([41]). *The inertia matrix $\boldsymbol{M}(\boldsymbol{q})$ and $\boldsymbol{M}_{\boldsymbol{x}}$ are symmetric positive definite.*

Property 2 ([41]). *The matrix $2\boldsymbol{C}_x(\boldsymbol{q}, \dot{\boldsymbol{q}}) - \dot{\boldsymbol{M}}_x(\boldsymbol{q})$ is skew-symmetric.*

Definition 2 ($\mathcal{K}_\infty$-class function [42]). *A continuous function $\beta : \mathbb{R}_+ \to \mathbb{R}_+$ is a $\mathcal{K}_\infty$-class function (denoted as $\beta \in \mathcal{K}_\infty$), if*

- $\beta(0) = 0$
- *β is increasing*
- $\lim_{a\to\infty} \beta(a) = \infty$.

Definition 3 (Control Lyapunov Functions [43]). *Given the system (8) and the composite error $\boldsymbol{z}(\boldsymbol{x}, t) = \dot{\tilde{\boldsymbol{x}}} + \Lambda\tilde{\boldsymbol{x}}$, with $\Lambda > 0$, consider a continuously differentiable function $V : \mathbb{R}^3 \to \mathbb{R}_+$ on set $\mathcal{Z} = \{\boldsymbol{z}(\boldsymbol{x}, t) \in \mathbb{R}^3 : \boldsymbol{x} \in \mathcal{C}, \forall t > 0\}$, where $\mathcal{C}$ is the safe set as defined for the ECBF (Definition 1). Then, V is a control Lyapunov function (CLF) for the system in (8) if there exist β_1, β_2, $\beta_3 \in \mathcal{K}_\infty$ such that for all $\boldsymbol{z} \in \mathcal{Z}$*

$$\beta_1(\|\boldsymbol{z}\|) \leq V(\boldsymbol{z}) \leq \beta_2(\|\boldsymbol{z}\|), \tag{30}$$

$$\inf_{\boldsymbol{u}\in\mathcal{U}} \dot{V}(\boldsymbol{z}, \boldsymbol{u}) \leq -\beta_3(\|\boldsymbol{z}\|). \tag{31}$$

The existence of a CLF implies the existence of a control law that renders the closed-loop system locally asymptotically stable [44]. The target when employing a CLF constraint is to make the composite error $\boldsymbol{z}(\boldsymbol{x}, \dot{\boldsymbol{x}})$, which models the generalized Cartesian position and velocity errors with respect to the desired references, locally asymptotically converge to 0. Then, the problem (29) needs to satisfy both the ECBF and CLF constraints, which are written as

$$\underset{\boldsymbol{u}_{\text{act}}\in\mathcal{U}}{\text{minimize}} \|\boldsymbol{f}_{\boldsymbol{h}} - \boldsymbol{u}_{\text{act}}\|^2, \tag{32a}$$

subject to

$$L_f^2 h_i(t, \boldsymbol{x}_i) + L_g L_f h_i(t, \boldsymbol{x}_i)\boldsymbol{u}_{\text{act},i} + k_1 h_i(t, \boldsymbol{x}_i) + k_2 L_f h_i(t, \boldsymbol{x}_i) \geq 0, i \in {}_{[1,7]}, \tag{32b}$$

$$\boldsymbol{z}_i^\top \left(\boldsymbol{u}_{\text{act},i} + k_z \frac{\boldsymbol{z}_i}{\|\boldsymbol{z}_i\| + c_1} \right) - \boldsymbol{z}_i^\top \boldsymbol{K}_D \boldsymbol{z}_i \geq 0, i \in {}_{[1,7]}, \tag{32c}$$

where $\boldsymbol{K}_D$ is a symmetric positive definite matrix.

The candidate Lyapunov function V is defined as

$$V(\boldsymbol{z}_i) = \tfrac{1}{2}\boldsymbol{z}_i^\top \boldsymbol{M}_{\boldsymbol{x}} \boldsymbol{z}_i. \tag{33}$$

Since $\boldsymbol{M}_{\boldsymbol{x}}(\boldsymbol{q})$ is positive definite for all $\boldsymbol{q}$, its minimum eigenvalue is positive. Assuming that $\inf_{\boldsymbol{q}} \lambda_{\min}(\boldsymbol{M}_{\boldsymbol{x}}(\boldsymbol{q})) > 0$ and $\sup_{\boldsymbol{q}} \lambda_{\max}(\boldsymbol{M}_{\boldsymbol{x}}(\boldsymbol{q})) < \infty$, the inequality in (30) is satisfied with $\beta_1(\|z\|) = \inf_{\boldsymbol{q}} \lambda_{\min}(\boldsymbol{M}_{\boldsymbol{x}}(\boldsymbol{q}))\|z\|^2$ and $\beta_2(\|z\|) = \sup_{\boldsymbol{q}} \lambda_{\max}(\boldsymbol{M}_{\boldsymbol{x}}(\boldsymbol{q}))\|z\|^2$.

Differentiating (33) with respect to time and using Property 1 to combine terms gives

$$\dot{V} = \tfrac{1}{2} z_i^\intercal \dot{\boldsymbol{M}}_x z_i + z_i^\intercal \boldsymbol{M}_x \dot{z}_i. \tag{34}$$

Noting that $\dot{z}_i = \ddot{x}_i - \ddot{x}_{d,i} + \Lambda(\dot{x}_i - \dot{x}_{d,i})$, and substituting for $\dot{\boldsymbol{M}}_x \ddot{x}_i$ using (6) with $\boldsymbol{f_h}$ replaced by the modified control force $\boldsymbol{f_h} - \boldsymbol{u}_{\text{act},i}$, $\boldsymbol{M}_x \dot{z}_i$ can be written as

$$\begin{aligned} \boldsymbol{M}_x \dot{z}_i = \boldsymbol{f_h} - \boldsymbol{u}_{\text{act},i} - \boldsymbol{C}_x (z_i + \dot{x}_{d,i} \\ - \Lambda(x_i - x_{d,i})) - \boldsymbol{g}_x - \boldsymbol{M}_x (\ddot{x}_{d,i} - \Lambda \dot{\tilde{x}}_i). \end{aligned} \tag{35}$$

Substituting (35) into (34) yields

$$\begin{aligned} \dot{V} &= \tfrac{1}{2} z_i^\intercal \dot{\boldsymbol{M}}_x z_i + z_i^\intercal [\boldsymbol{f_h} - \boldsymbol{u}_{\text{act},i} - \boldsymbol{C}_x (z_i + \dot{x}_{d,i} \\ &\quad - \Lambda(x_i - x_{d,i})) - \boldsymbol{g}_x - \boldsymbol{M}_x (\ddot{x}_{d,i} - \Lambda \dot{\tilde{x}}_i)] \\ &= \tfrac{1}{2} z_i^\intercal (\dot{\boldsymbol{M}}_x - 2\boldsymbol{C}_x) z_i + z_i^\intercal [\boldsymbol{f_h} - \boldsymbol{u}_{\text{act},i} \\ &\quad - \boldsymbol{C}_x (\dot{x}_{d,i} - \Lambda \tilde{x}_i) - \boldsymbol{g}_x] - z_i^\intercal \boldsymbol{M}_x (\ddot{x}_{d,i} - \Lambda \dot{\tilde{x}}_i). \end{aligned} \tag{36}$$

By Property 2, $z_i^\intercal (\dot{\boldsymbol{M}}_x - 2\boldsymbol{C}_x) z_i = 0$, hence the derivative of V can be expressed as

$$\dot{V} = z_i^\intercal [(\boldsymbol{f_h} - \boldsymbol{u}_{\text{act},i}) - \boldsymbol{C}_x (\dot{x}_{d,i} - \Lambda \tilde{x}_i) - \boldsymbol{g}_x] - z_i^\intercal \boldsymbol{M}_x (\ddot{x}_{d,i} - \Lambda \dot{\tilde{x}}_i). \tag{37}$$

Substituting (23) into (37) yields

$$\dot{V} = -z_i^\intercal \boldsymbol{u}_{\text{act},i} - \frac{k_z z_i^\intercal z_i}{\|z_i\| + c_1}. \tag{38}$$

According to Definition 2 and (32c), the stability of the low-level controller is guaranteed when $\dot{V} \leq -z_i^\intercal \boldsymbol{K}_D z_i$ assuming problem (32) is feasible.

7. Experimental Setup

The collaboration scenario shown in Figure 5 is a typical screwdriver usage scenario where the robot holds the screw at an initial position and delivers it to the human operator before the operator picks up the screwdriver. The operator's responsibility is to use the screwdriver and take the screw from the robot end-effector, which stops at a safe interactive distance d_{safe}. Active collision avoidance is activated when the human body appears in the robot's original trajectory. For seamless human–robot collaboration, the robot starts to calculate a new trajectory based on the predicted future pose sequence for the operator (1-sec horizon, N_h = 20). The overall procedure for using the screwdriver is illustrated in Algorithm 1. After the initialization of the robot and the OptiTrack camera system, the software enters the main execution loop. Here, the action being undertaken by the operator is predicted at each sample instant. If an action requiring interaction is predicted, the software calls Algorithm 2 to compute the required robot joint torques in accordance with the proposed ECBF-NMPC dynamic motion planning framework.



Figure 5. Human motion capture in the screwdriver usage task.

Algorithm 1: Vision-based screw-driver usage task execution

Input: f_{recg} (pretrained task classifier).
Data: OptiTrack data stream
Initialize the robot to its original position.
Initialize the Optitrack camera system and $\boldsymbol{P}_{\text{prev}}$.
for $t = 1, \ldots$ **do**
  Read $\boldsymbol{P}(t)$ from the OptiTrack camera system.
  $\boldsymbol{P}_{\text{prev}}(t) = [\boldsymbol{P}(t - \tau_o + 1), \ldots, \boldsymbol{P}(t)]$.
  $\hat{y} = f_{\text{recg}}(\boldsymbol{P}_{\text{prev}}(t))$.
  if $\hat{y}(t) \in \mathcal{A}_{\text{int}}$ **then**
    Get the current state of the robot $\boldsymbol{x}$.
    $\hat{\boldsymbol{\tau}}_{\text{act}}(t) = Algorithm2(\boldsymbol{P}_{\text{prev}}(t), \boldsymbol{x})$.
    Apply $\hat{\boldsymbol{\tau}}_{\text{act}}(t)$ to the robot.
  else
    Stop the robot.

Algorithm 2: Motion-prediction-based path planning

Input: f_{pred} (pretrained motion predictor), $\boldsymbol{P}_{\text{prev}}$ (previous one second motion sequence), $\boldsymbol{x}$ (current state of the robot).
Output: input to robot $\boldsymbol{\tau}_{\text{act}}$
Define the controller.
$[\hat{\boldsymbol{p}}_{\text{rh}}, \hat{\boldsymbol{p}}_{\text{o}}] = \underline{\text{f}}_{\text{pred}}(\boldsymbol{P}_{\text{prev}})$
Build the capsule-format representation of the human body and compute the minimum distance $\lambda(\boldsymbol{x}, \hat{\boldsymbol{p}}_{\boldsymbol{o}}))$.
Solve the defined problem in OpEn and compute the optimal input to the high-level controller $\hat{\boldsymbol{u}}^{\star}$.
Compute the desired robot state $\boldsymbol{q}_d$ and $\dot{\boldsymbol{q}}_d$.
Design the control input to the safety filter $\boldsymbol{f}_{\boldsymbol{h}}(t)$.
Solve the ECBF quadratic program to determine $\boldsymbol{u}_{act}(t)$.
Compute the input to the robot: $\boldsymbol{\tau}_{\text{act}}(t) = \boldsymbol{J}^{\intercal}(\boldsymbol{q}(t))\boldsymbol{u}_{\text{act}}(t)$.

7.1. Motion Capture

In this work, we generate high-quality skeleton data using an OptiTrack camera system, which includes four volunteers. The volunteers enact the required behavior in a working environment with three cameras, with the volunteers' behavior tracked from different perspectives, as shown in Figure 5. Volunteers' movements captured in task coordinates are transformed into corresponding trajectories within the CoppeliaSim environment, enabling accurate replication of human motion patterns by the simulated agent. The generated

human skeleton model has 32 joints, with joint data recorded at a sampling interval of 50 ms (i.e., 20 frames per second).

The capture space evaluated was based on typical use and had an extent of 2.5 m × 3.5 m × 2 m = 17.5 m^3. This space can be covered by three cameras. The number of cameras required will increase for larger spaces (or volumes). We have opted for a marker-based approach in this work, regardless of the high cost of system setup and calibration, to ensure high-quality input into the perception system.

7.2. Vision System

For the vision module, we use a non-local graph convolution (NGC)-based method developed in [35] for action recognition and motion prediction. This implements a spatial-temporal attention mechanism to extract nonlocal spatial-temporal features, enabling more accurate motion prediction over longer horizons than alternative approaches. As this model, referred to as the NGC-model, performs well for the Human 3.6 M, CMU Mocap, and NTU RGB-D datasets (see [35]), we repurpose it to predict the human motion associated with the screwdriver task and classify it into one of six subtasks that constitute the overall activity, namely, 'pick up', 'move forward', 'take the screw', 'operate screwdriver', 'move backward,' and 'put down.' This is achieved by retraining the network with a dataset collected for the screwdriver task, consisting of skeletal data and associated subtask labels for three repetitions of the task. In this work, 4372 frames including six different actions are used for training, while 1287 frames are used for testing. Each frame includes 3D joint positions with 32 joints, as well as ground-truth action labels for each frame.

Performance of the Vision Module

As shown by the confusion matrix in Figure 6, the model achieves high prediction accuracy for each subtask (99.0%, 100%, 97.9%, 99.0%, 98.3%, and 99.5%) and an overall accuracy of 98.9%. For motion prediction, the mean per joint position error (MPJPE) is used to quantify the mean deviation between the predicted and actual locations of the human body joints in each frame. As shown in Table 1, the NGC-model achieves the lowest error compared to selected baseline models from the literature (LSTM-3LR [45], Res-sup [46], Traj-GCN [47] and SPGSN [48]). Table 1 compares the MPJPE of our model and the baselines for all actions under the same conditions, with the prediction horizon of 1000ms, showing that our model obtains the best performance for all the actions.

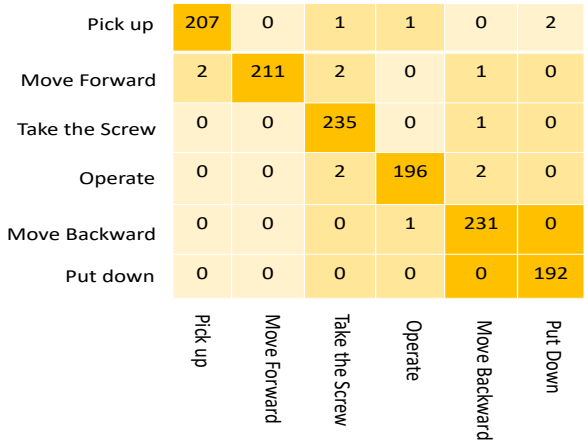


Figure 6. Confusion matrix describing the accuracy of action recognition in the screwdriver usage task.

Table 1. Comparison of MPJPE between models for 1000 ms prediction.

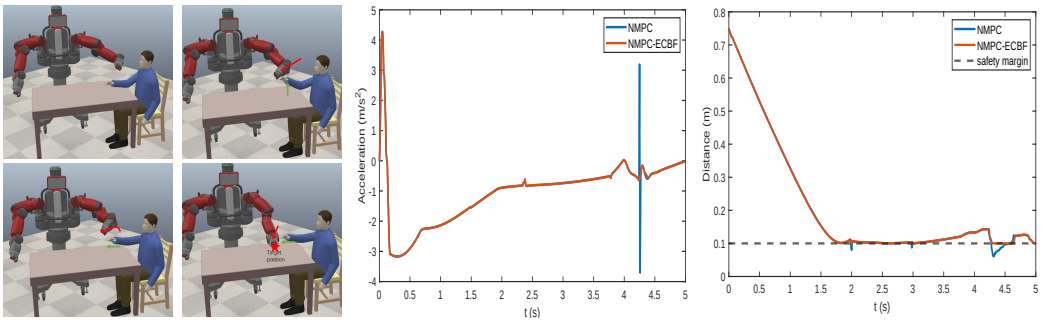
	Pick Up	Move Forward	Take the Screw	Operate	Move Backward	Put Down
LSTM-3LR [45]	2.41	2.26	2.73	2.52	2.68	2.88
Res-sup [46]	2.18	2.14	2.42	2.38	2.57	2.71
Traj-GCN [47]	1.95	2.01	2.29	2.21	2.46	2.61
SPGSN [48]	1.33	1.96	2.01	2.09	2.41	2.37
Ours	**1.29**	**1.92**	**1.98**	**2.07**	**2.39**	**2.35**

This shows that the proposed NGC model can accurately and effectively identify the intended human body movement in real-time. Using this capability, the HRC system can understand the intention of the human being when the human body begins an action, making it feasible for it to provide assistance to the human to complete the task efficiently.

7.3. Control System Performance Test

To evaluate NMPC-ECBF, we implement it for a 7-DOF Baxter robot simulated in CoppeliaSim using a standard NMPC controller as a baseline comparison [15]. The system model employed in the MPC controller is discretized using the Euler method. The weight matrices in ℓ and ℓ_f are selected as $\boldsymbol{Q}_v = \text{diag}(1,1,1,1,1,1,1)$, $\boldsymbol{Q}_p = \text{diag}(5,5,5,1,1,0)$, $\boldsymbol{R}_v = \text{diag}(0.1,0.1,0.1,0.1,0.1,0.1,0.1)$ and $\boldsymbol{R}_p = \text{diag}(3,3,3,0,0,0)$. Solution tolerances $\epsilon_a = 10^{-2}$ and $\epsilon_b = 10^{-4}$ are set for the fixed-point residual and the infeasibility, respectively. For the safety filter, safe set $\mathcal{U}$ for input $\boldsymbol{u}_{\text{act}}$ is $(-40\,\text{N}, 40\,\text{N})$. The parameters of the low-level controller are selected as $k_z = 5$ and $c_1 = 0.01$. Noting the maximum joint prediction errors in Figure 6, we set the safe distance (d_{safe}) as 10 cm. The ECBF coefficients $\boldsymbol{k}_b$ are chosen as $[7,7]$, while $\boldsymbol{K}_D = \text{diag}(5,5,5)$; these values are determined within the range to satisfy Equation (32b).

In the CoppeliaSim simulation, the movement of the human is modeled using the recorded trajectories of the skeleton joints. The Baxter robot starts from different initial positions and is required to reach the estimated position without collision with the simulated human, as shown in Figure 7a.

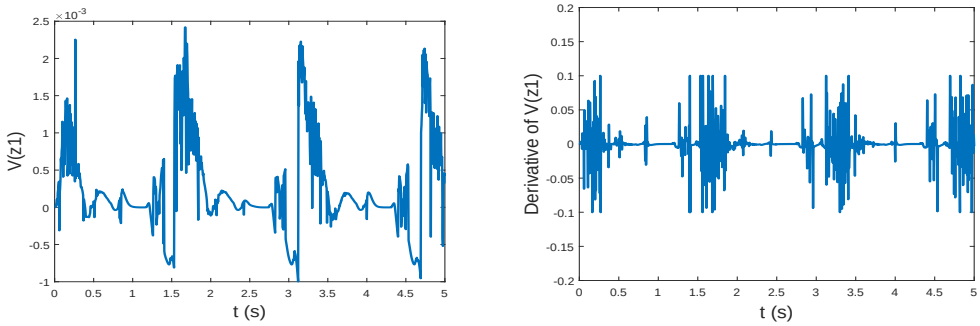


(**a**) (**b**) (**c**)

Figure 7. (**a**) HRC scenario when the initial position is close to the operator. (**b**) The acceleration of the end-effector; (**c**) The minimum distance to the obstacles among all the robot links.

Under the same conditions, the acceleration and the minimum distance between the end-effector and human obstacle using different algorithms in scenario 1 are plotted in Figure 7b and 7c, respectively. The acceleration plot shows that NMPC-ECBF yields fewer abrupt changes in acceleration than the baseline NMPC path planner. The minimum distance plot highlights that the trajectory generated by NMPC violates the safe area at several points during the robot's motion and that the inclusion of the ECBF safety filter has the desired effect of maintaining the motion within the safe area.

To prove the stability of the controller, Figure 8a,b illustrates the state of $V(z_1)$ and $\dot{V}(z_1)$, respectively. It shows the maximum value of $V(z_1)$ is less than 2.5×10^{-3}, which is close to 0. At the same time, the derivative of the Lyapunov function is less than 0.1, which proves the marginal stability of the system.



(**a**) (**b**)

Figure 8. (**a**) The plot of $V(z1)$; (**b**) The plot of $\dot{V}(z1)$.

To further assess the robustness and reliability of the controller, we perform 100 dynamic path-planning HRC experiments, with each one corresponding to a randomly generated robot start position and one of the three sets of human screwdriver usage task movement trajectories. For each experiment, we recorded the maximum acceleration of the end-effector and the minimum distance between the end-effector and the target safety margin during the task execution. Table 2 reports the mean and most extreme values observed over the 100 experiments. The results show that NMCP-ECBF is able to strictly keep the robot motion within the safe area for all 100 simulations, with no violations of the safety margin ($d = 0$). In contrast, the NMPC path planner frequently breaches the safety margin, with an average violation of 1.62 cm and a maximum violation of 3.26 cm. Furthermore, NMPC-ECBF outperforms NMPC in terms of the average and maximum joint accelerations, achieving a reduction in these values of 48.0% and 78.2%, respectively. It is obvious that the proposed controller improves the safety and smoothness in the HRC task.

Table 2. Controller performance for 100 HRC task instances.

	Max Acc (m/s^2)	**Min d (cm)**	**Avg Max Acc (m/s^2)**	**Avg Min d (cm)**
NMPC	12.41	−3.26	2.73	−1.62
NMPC-ECBF	2.70	0	1.42	0

7.4. Idle Time for Agents in the HRC Scenario

Figure 9 compares the human idle time (H-IDL) and robot idle time (R-IDL) in the screwdriver usage HRC task (i.e., Scenario 1) to highlight the benefit of integrating motion prediction into HRC. The robot is triggered when the human's action is detected to be 'pick up' and is stopped when the 'take the screw' action is detected. In the proposed screw-driver usage task, the original length of this HRC task is 14 s, which is the time used in human-human collaboration. Red blocks indicate when the operator is in an idle state, waiting for the screw to arrive. White blocks are the times the robot is idle for computation and data loading. Green blocks are the times when both agents are executing the commanded task. When employing motion-prediction-based control, the R-IDL consists of the time for motion prediction and path planning. The robot's execution time accounts for the majority of the total time when employing the motion predictor and embedded optimization engine. In the HRC task implemented without motion prediction, the robot

has to start path planning after the human operator moves their right hand to the target interactive position.

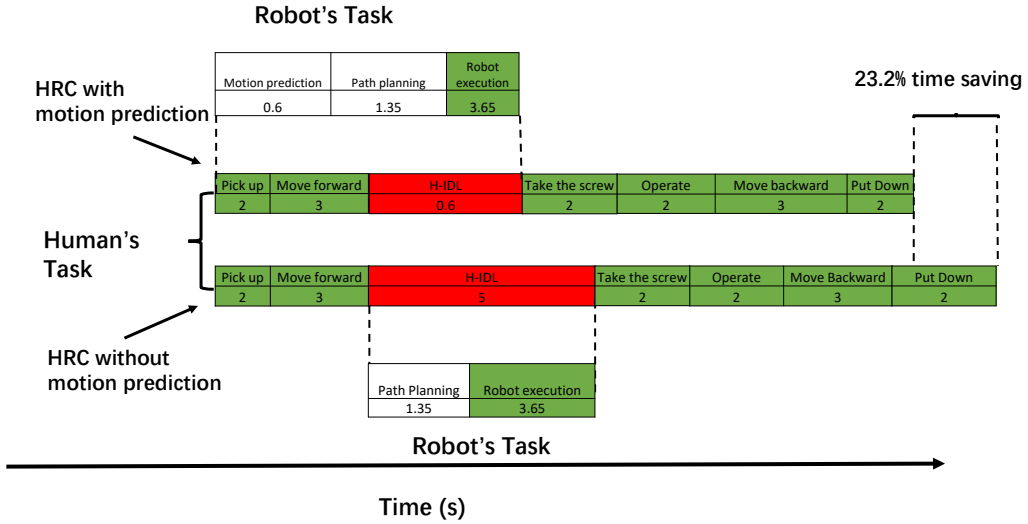


Figure 9. Task assignment chart for the baseline approach of strict turn-taking, where each action is immediately followed by the next action of the other teammate.

After repeating the screwdriver usage task 100 times, Table 3 compares the average human idle time (H-IDL) and robot idle time (R-IDL) for the human–robot collaboration task without motion prediction. In this work, the HRC without motion prediction can be treated as a strict turn-taking task, where each agent strictly takes action after the partner finishes. The H-IDL rate is reduced by 4.1% and the R-IDL rate by 13.4% in the HRC task using the motion-prediction-based approach. Without motion prediction, the H-IDL rate and R-IDL rate are 26.3% and 7.1%, respectively. Therefore, using motion prediction reduces the idle time for both the human and the robot in the HRC task. As a result, the total execution time is reduced from 19.0 to 14.6 s, a saving of 23.2%.

Table 3. Comparison of the idle time between the tasks with or without motion prediction.

	With Motion Prediction (s)	Without Motion Prediction (s)
H-IDL	0.6	5
R-IDL	1.95	1.35
Total Time	14.6	19.0

8. Conclusions

The proposed NMPC-ECBF control framework allows a robot to safely replan its motion in the presence of a human, delivering smooth acceleration and adaptivity in the presence of uncertainty. The results demonstrate the necessity of adding a safety filter to the NMPC controller in the trajectory planning problem. This represents a promising approach for manufacturing applications due to its characteristic of strictly constraining the robot's motion. In addition, by employing human motion prediction within the framework, tasks can be executed more efficiently with a lower idle rate achievable for both agents in an HRC task than with traditional turn-taking task execution.

Future work will be devoted to testing the proposed algorithm when there is uncertainty regarding the system dynamics (e.g., caused by joint friction). The possibility of promoting the proposed model for scenarios involving multiple robots and coworkers will also be investigated. In addition, we will explore the applicability of our approach to scenarios involving both physical and non-physical HRC interactions, with the robot

switching between different control modes based on the prevailing circumstances, as determined by a higher-level supervisory system. Furthermore, we will integrate multimodal learning (e.g., radar sensors, tactile sensors, and auditory sensors) into the HRC system for more robust and safer collaboration.

Author Contributions: Methodology, D.Z.; Supervision, M.V., P.S. and S.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Centre for Intelligent Autonomous Manufacturing Systems (i-AMS), School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Belfast BT9 5AG, UK.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Nicora, M.L.; Ambrosetti, R.; Wiens, G.J.; Fassi, I. Human—Robot Collaboration in Smart Manufacturing: Robot Reactive Behavior Intelligence. *J. Manuf. Sci. Eng.* **2021**, *143*, 031009. [CrossRef]
2. Kumar, S.; Savur, C.; Sahin, F. Survey of Human—Robot Collaboration in Industrial Settings: Awareness, Intelligence, and Compliance. *IEEE Trans. Syst. Man Cybern. Syst.* **2021**, *51*, 280–297. [CrossRef]
3. Cao, R.; Cheng, L.; Li, H. Passive Model-Predictive Impedance Control for Safe Physical Human–Robot Interaction. *IEEE Trans. Cogn. Dev. Syst.* **2024**, *16*, 426–435. [CrossRef]
4. Han, L.; Zhao, L.; Huang, Y.; Xu, W. Variable admittance control for safe physical human–robot interaction considering intuitive human intention. *Mechatronics* **2024**, *97*, 12. [CrossRef]
5. Wang, W.; Li, R.; Chen, Y.; Diekel, Z.M.; Jia, Y. Facilitating human–robot collaborative tasks by teaching-learning-collaboration from human demonstrations. *IEEE Trans. Autom. Sci. Eng.* **2018**, *16*, 640–653. [CrossRef]
6. Terreran, M.; Barcellona, L.; Ghidoni, S. A general skeleton-based action and gesture recognition framework for human–robot collaboration. *Robot. Auton. Syst.* **2023**, *170*, 104523. [CrossRef]
7. Qi, W.; Ovur, S.E.; Li, Z.; Marzullo, A.; Song, R. Multi-sensor Guided Hand Gestures Recognition for Teleoperated Robot using Recurrent Neural Network. *IEEE Robot. Autom. Lett.* **2021**, *6*, 6039–6045. [CrossRef]
8. Quintas, J.; Martins, G.S.; Santos, L.; Menezes, P.; Dias, J. Toward a Context-Aware Human–Robot Interaction Framework Based on Cognitive Development. *IEEE Trans. Syst. Man Cybern. Syst.* **2019**, *49*, 227–237. [CrossRef]
9. Huck, T.P.; Münch, N.; Hornung, L.; Ledermann, C.; Wurll, C. Risk assessment tools for industrial human-robot collaboration: Novel approaches and practical needs. *Saf. Sci.* **2021**, *141*, 105288. [CrossRef]
10. Chemweno, P.; Pintelon, L.; Decre, W. Orienting safety assurance with outcomes of hazard analysis and risk assessment: A review of the ISO 15066 standard for collaborative robot systems. *Saf. Sci.* **2020**, *129*, 104832. [CrossRef]
11. Ide, S.; Takubo, T.; Ohara, K.; Mae, Y.; Arai, T. Real-time trajectory planning for mobile manipulator using model predictive control with constraints. In Proceedings of the 2011 8th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), Incheon, Republic of Korea, 23–26 November 2011; pp. 244–249.
12. Salaj, M.; Gulan, M. Pendubot control scheme based on nonlinear MPC and MHE exploiting parallelization. In Proceedings of the 2015 IEEE 19th International Conference on Intelligent Engineering Systems (INES), Bratislava, Slovakia, 3–5 September 2015; pp. 353–358.
13. Su, H.; Schmirander, Y.; Li, Z.; Zhou, X.; Ferrigno, G.; Momi, E.D. Bilateral Teleoperation Control of a Redundant Manipulator with an RCM Kinematic Constraint. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 4477–4482.
14. Li, W.; Xiong, R. Dynamical obstacle avoidance of task-constrained mobile manipulation using model predictive control. *IEEE Access* **2019**, *7*, 88301–88311. [CrossRef]
15. Oleinikov, A.; Kusdavletov, S.; Shintemirov, A.; Rubagotti, M. Safety-Aware Nonlinear Model Predictive Control for Physical Human-Robot Interaction. *IEEE Robot. Autom. Lett.* **2021**, *6*, 5665–5672. [CrossRef]
16. Rawlings, J.; Mayne, D. *Model Predictive Control: Theory, Computation, and Design*; Nob Hill Publishing: Madison, WI, USA, 2022; Volume 2.
17. Wolf, I.J.; Marquardt, W. Fast NMPC schemes for regulatory and economic NMPC–A review. *J. Process Control* **2016**, *44*, 162–183. [CrossRef]
18. Secil, S.; Ozkan, M. Minimum distance calculation using skeletal tracking for safe human-robot interaction. *Robot. Comput. Integr. Manuf.* **2022**, *73*, 102253. [CrossRef]

19. Wang, H.; Peng, J.; Zhang, F.; Zhang, H.; Wang, Y. High-order control barrier functions-based impedance control of a robotic manipulator with time-varying output constraints. *ISA Trans.* **2022**, *129*, 361–369. [CrossRef]
20. Son, T.D.; Nguyen, Q. Safety-critical control for non-affine nonlinear systems with application on autonomous vehicle. In Proceedings of the 2019 IEEE 58th Conference on Decision and Control (CDC), Nice, France, 11–13 December 2019; pp. 7623–7628.
21. Wang, C.; Meng, Y.; Li, Y.; Smith, S.L.; Liu, J. Learning control barrier functions with high relative degree for safety-critical control. In Proceedings of the 2021 European Control Conference (ECC), Delft, The Netherlands, 29 June–2 July 2021; pp. 1459–1464.
22. Liu, H.; Wang, L. Human motion prediction for human-robot collaboration. *J. Manuf. Syst.* **2017**, *44*, 287–294. [CrossRef]
23. Lindqvist, B.; Mansouri, S.S.; Agha-mohammadi, A.A.; Nikolakopoulos, G. Nonlinear MPC for collision avoidance and control of UAVs with dynamic obstacles. *IEEE Robot. Autom. Lett.* **2020**, *5*, 6001–6008. [CrossRef]
24. Lindqvist, B.; Mansouri, S.S.; Kanellakis, C.; Nikolakopoulos, G. Collision Free Path Planning based on Local 2D Point-Clouds for MAV Navigation. In Proceedings of the 2020 28th Mediterranean Conference on Control and Automation (MED), Saint-Raphaël, France, 16–19 June 2020; pp. 538–543.
25. Jerez, J.L.; Goulart, P.J.; Richter, S.; Constantinides, G.A.; Kerrigan, E.C.; Morari, M. Embedded online optimization for model predictive control at megahertz rates. *IEEE Trans. Autom. Control* **2014**, *59*, 3238–3251. [CrossRef]
26. Sathya, A.; Sopasakis, P.; Van Parys, R.; Themelis, A.; Pipeleers, G.; Patrinos, P. Embedded nonlinear model predictive control for obstacle avoidance using PANOC. In Proceedings of the 2018 European control conference (ECC), Limassol, Cyprus, 12–15 June 2018; pp. 1523–1528.
27. Sopasakis, P.; Fresk, E.; Patrinos, P. OpEn: Code generation for embedded nonconvex optimization. *IFAC-PapersOnLine* **2020**, *53*, 6548–6554. [CrossRef]
28. Trimble, S.; Naeem, W.; McLoone, S.; Sopasakis, P. Context-aware robotic arm using fast embedded model predictive control. In Proceedings of the 2020 31st Irish Signals and Systems Conference (ISSC), Letterkenny, Ireland, 11–12 June 2020; pp. 1–6.
29. Rubagotti, M.; Patrinos, P.; Bemporad, A. Stabilizing Linear Model Predictive Control Under Inexact Numerical Optimization. *IEEE Trans. Autom. Control* **2014**, *59*, 1660–1666. [CrossRef]
30. Gurriet, T.; Mote, M.; Singletary, A.; Nilsson, P.; Feron, E.; Ames, A.D. A scalable safety critical control framework for nonlinear systems. *IEEE Access* **2020**, *8*, 187249–187275. [CrossRef]
31. Nguyen, Q.; Sreenath, K. Robust safety-critical control for dynamic robotics. *IEEE Trans. Autom. Control* **2022**, *67*, 1073–1088. [CrossRef]
32. Singletary, A.; Kolathaya, S.; Ames, A.D. Safety-critical kinematic control of robotic systems. *IEEE Control Syst. Lett.* **2021**, *6*, 139–144. [CrossRef]
33. Landi, C.T.; Ferraguti, F.; Costi, S.; Bonfè, M.; Secchi, C. Safety barrier functions for human-robot interaction with industrial manipulators. In Proceedings of the 2019 18th European Control Conference (ECC), Naples, Italy, 25–28 June 2019; pp. 2565–2570.
34. Ding, X.; Wang, H.; Ren, Y.; Zheng, Y.; Chen, C.; He, J. Online Control Barrier Function Construction for Safety-Critical Motion Control of Manipulators. *IEEE Trans. Syst. Man Cybern. Syst.* **2024**, *54*, 4761–4771. [CrossRef]
35. Zhang, D.; Vien, N.A.; Van, M.; McLoone, S. Non-local Graph Convolutional Network for joint Activity Recognition and Motion Prediction. In Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 27 September–1 October 2021; pp. 2970–2977.
36. Safeea, M.; Mendes, N.; Neto, P. Minimum distance calculation for safe human robot interaction. *Procedia Manuf.* **2017**, *11*, 99–106. [CrossRef]
37. Lin, H.C.; Liu, C.; Fan, Y.; Tomizuka, M. Real-time collision avoidance algorithm on industrial manipulators. In Proceedings of the 2017 IEEE Conference on Control Technology and Applications (CCTA), Maui, HI, USA, 27–30 August 2017; pp. 1294–1299.
38. Ionescu, C.; Papava, D.; Olaru, V.; Sminchisescu, C. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *36*, 1325–1339. [CrossRef] [PubMed]
39. Hu, S.; Babaians, E.; Karimi, M.; Steinbach, E. NMPC-MP: Real-time Nonlinear Model Predictive Control for Safe Motion Planning in Manipulator Teleoperation. In Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 27 September–1 October 2021; pp. 8309–8316.
40. Nguyen, Q.; Sreenath, K. Exponential control barrier functions for enforcing high relative-degree safety-critical constraints. In Proceedings of the 2016 American Control Conference (ACC), Boston, MA, USA, 6–8 July 2016; pp. 322–328.
41. He, W.; Ge, S.S.; Li, Y.; Chew, E.; Ng, Y.S. Neural network control of a rehabilitation robot by state and output feedback. *J. Intell. Robot. Syst.* **2015**, *80*, 15–31. [CrossRef]
42. Grandia, R.; Taylor, A.J.; Singletary, A.; Hutter, M.; Ames, A.D. Nonlinear model predictive control of robotic systems with control lyapunov functions. *arXiv* **2020**, arXiv:2006.01229. [CrossRef]
43. Minniti, M.V.; Grandia, R.; Farshidian, F.; Hutter, M. Adaptive CLF-MPC with application to quadrupedal robots. *IEEE Robot. Autom. Lett.* **2021**, *7*, 565–572. [CrossRef]
44. Sontag, E.D. A Lyapunov-Like Characterization of Asymptotic Controllability. *SIAM J. Control Optim.* **1983**, *21*, 462–471. [CrossRef]

45. Gui, L.Y.; Zhang, K.; Wang, Y.X.; Liang, X.; Moura, J.M.; Veloso, M. Teaching robots to predict human motion. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 562–567.
46. Martinez, J.; Black, M.J.; Romero, J. On human motion prediction using recurrent neural networks. In Proceedings of the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2891–2900.
47. Mao, W.; Liu, M.; Salzmann, M.; Li, H. Learning trajectory dependencies for human motion prediction. In Proceedings of the Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 9489–9497.
48. Li, M.; Chen, S.; Zhang, Z.; Xie, L.; Tian, Q.; Zhang, Y. Skeleton-Parted Graph Scattering Networks for 3D Human Motion Prediction. In Proceedings of the Computer Vision—ECCV: 17th European Conference, Tel Aviv, Israel, 23–27 October 2022; Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T., Eds.; Springer: Cham, Switzerland, 2022; pp. 18–36.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.