Robotics and Autonomous Systems 197 (2026) 105291

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

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

Uncertainty Aware-Predictive Control Barrier Functions: Safer human–robot interaction through probabilistic motion forecasting☆

Lorenzo Busellato [a,1], Federico Cunico [a,*,1], Diego Dall'Alba [a], Marco Emporio [a], Andrea Giachetti [a], Riccardo Muradore [a], Marco Cristani [a,b]

[a] University of Verona, Department of Engineering for Innovation Medicine, Strada Le Grazie, 15, Verona, 37134, Veneto, Italy
[b] Reykjavik University, Department of Computer Science, Menntavegur 1, Reykjavik, 102, Iceland

ARTICLE INFO

Keywords:
Control Barrier Functions
Human–robot cooperation
Hand trajectory forecasting
Motion planning
Collision avoidance

ABSTRACT

To enable flexible, high-throughput automation in settings where people and robots share workspaces, collaborative robotic cells must reconcile stringent safety guarantees with the need for responsive and effective behavior. A dynamic obstacle is the stochastic, task-dependent variability of human motion: when robots fall back on purely reactive or worst-case envelopes, they brake unnecessarily, stall task progress, and tamper with the fluidity that true Human–Robot Interaction (HRI) demands. In recent years, learning-based human-motion prediction has rapidly advanced, although most approaches produce worst-case scenario forecasts that often do not treat prediction uncertainty in a well-structured way, resulting in over-conservative planning algorithms, limiting their flexibility. This paper introduces Uncertainty-Aware Predictive Control Barrier Functions (UA-PCBFs), a unified framework that fuses probabilistic human hand motion forecasting with the formal safety guarantees of Control Barrier Functions (CBFs). In contrast to CBFs and other variants, our framework allows for a dynamic adjustment of the safety margin thanks to the human motion uncertainty estimation provided by the deep-learning forecasting module. Thanks to the awareness of prediction uncertainty, UA-PCBFs empower collaborative robots with a deeper understanding of future human states, facilitating more fluid and intelligent interactions through informed motion planning. Our key contribution is the first integration of epistemic prediction uncertainty directly into predictive CBFs, dynamically adjusting safety margins based on forecast confidence without assumptions about uncertainty evolution. We validate UA-PCBFs through comprehensive real-world experiments with an increasing level of realism, including automated setups (to perform exactly repeatable motions) with a robotic hand and direct human–robot interactions (to validate promptness, usability, and human confidence). Relative to state-of-the-art HRI architectures, UA-PCBFs show better performance in task-critical metrics, significantly reducing the number of violations of the robot's safe space during interaction with respect to the state-of-the-art. Data and code will be released upon acceptance.

1. Introduction

Human–Robot Interaction (HRI) must adhere to stringent safety constraints to ensure reliable and efficient interaction between humans and robots. With the transition towards Industry 5.0 and the integration of collaborative robots (cobots) into human workspaces, there is the potential to increase productivity, improve worker ergonomics, and enable more flexible automation [1]. However, achieving seamless and safe interaction, fundamental for the Industry 5.0 core task of Human–Robot Collaboration, remains a significant challenge due to the unpredictability of human movement. Unlike traditional robotic systems operating in structured environments with predefined tasks, mixed human–robot systems must dynamically adapt to human behavior while ensuring safety and efficiency [2].

One of the key issues in HRI is the inherent variability and uncertainty in human motion. Human operators exhibit natural, non-deterministic behavior, which makes it difficult for robots to predict their movements accurately and therefore plan/re-plan their trajectory

☆ This article is part of a Special issue entitled: 'Planning & Learning' published in Robotics and Autonomous Systems.

* Corresponding author.
E-mail addresses: lorenzo.busellato@univr.it (L. Busellato), federico.cunico@univr.it (F. Cunico), diego.dallalba@univr.it (D. Dall'Alba), marco.emporio@univr.it (M. Emporio), andrea.giachetti@univr.it (A. Giachetti), riccardo.muradore@univr.it (R. Muradore), marco.cristani@univr.it (M. Cristani).

[1] The authors contributed equally.

https://doi.org/10.1016/j.robot.2025.105291

Available online 11 December 2025

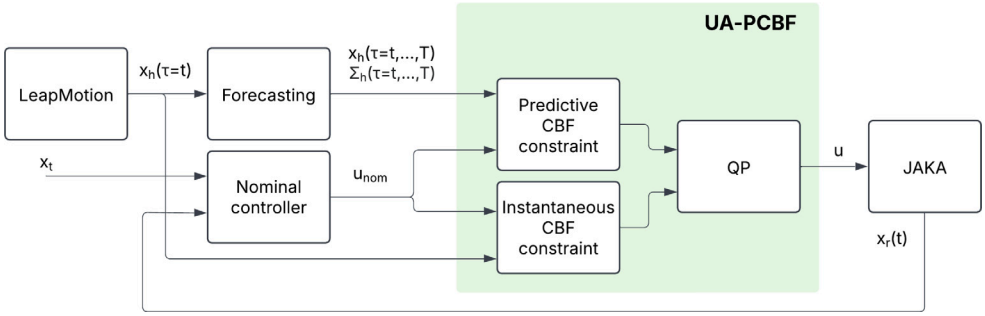

Fig. 1. Block diagram of the proposed framework. x_t and $x_r(t)$ denote the target and current Cartesian position of the end effector respectively, $x_h(\tau = t)$ denotes the current Cartesian position of the hand, $x_h(\tau = t, \dots, T)$ and $\Sigma_h(\tau = t, \dots, T)$ denote the predicted sequences of cartesian positions and covariance matrices respectively. u_{nom} and u denote, respectively, the nominal control and the computed probabilistically safe control.

accordingly [3–5]. This unpredictability can lead to inefficiencies, interruptions, or even safety hazards if the robot reacts too conservatively or aggressively. To manage this unpredictability, traditional control approaches typically employ predefined safety zones or reactive obstacle avoidance strategies, which limit adaptability and often result in reduced operational efficiency [6].

Control Barrier Functions (CBFs) have been widely explored as a means to enforce safety constraints while maintaining real-time adaptability in robotic systems. CBFs are particularly well-suited for HRI due to their ability to enforce safety constraints in a minimally invasive and mathematically rigorous manner. Unlike traditional motion planning techniques, which often separate planning and control or require replanning when unexpected human motion is detected, CBFs operate directly at the control level, enabling real-time safety enforcement and offering provable guarantees on constraint satisfaction without sacrificing responsiveness or adaptability [7–9].

However, conventional CBFs primarily rely on instantaneous observations and are often designed for rigid-body obstacle avoidance. This leads to a purely *reactive* control system, which is poorly suited for highly dynamic environments. Additionally, conventional CBF approaches fail to account for noise in human pose estimation, further complicating the task of achieving smooth and proactive interaction. Existing approaches extend CBFs to account for these shortcomings, such as Predictive Control Barrier Functions (PCBFs) [10], which incorporate predictions on the evolution of the system into the CBF definition, and Stochastic Control Barrier Functions (SCBFs), which introduce stochasticity directly into the system's definition. However, neither approach fully captures the uncertainty inherent in human motion predictions: PCBFs assume deterministic forecasts, whereas SCBFs require full probabilistic modeling of system dynamics, an impractical requirement when dealing with human behavior.

To address these limitations, we propose a novel framework called Uncertainty Aware-Predictive Control Barrier Functions (UA-PCBFs). This approach extends traditional CBF formulations by incorporating the human hand movement uncertainty, thanks to a deep learning forecasting model capable of providing an uncertainty measure for the future hand position. Thanks to that, our framework enables robots to proactively adjust their trajectories based on estimated future human behavior, rather than relying solely on reactive responses, accounting also for the uncertainty of the estimation. This predictive capability enhances both safety and efficiency, allowing for more intuitive and cooperative interactions between humans and robots. The forecasting approach is an autoregressive Deep Learning (DL) model based on LSTM [11]. The non-linear nature of this model is the ability to predict the hand movement, capturing both short and long-term temporal dependencies as well as complex kinematic patterns, while at the same time can achieve an execution speed fast enough to match the robot control's requirements. Furthermore, since it is based solely on computer vision estimated 3D hand pose estimation, it guarantees fast applicability in various contexts.

To the best of our knowledge, this work is the first to integrate epistemic prediction uncertainty directly into predictive CBFs — embedding it in the safe-set definition and enabling dynamic adjustment of the safe-space barrier based on the current uncertainty level — without making additional assumptions about how that uncertainty will evolve.

We validate our framework by performing real-world experiments on a 6-DoF robot in a manufacturing facility, showing how our system can handle complex scenarios. Besides validation tests with a human operator, to guarantee reproducibility, we employ a mockup hand operated by another robot to perform the tests. Results show that our method breaks down the state-of-the-art CBFs by an order of magnitude, showing a significant increase in the safety of interactions. Fig. 1 shows a complete overview of our proposed framework.

In summary, our contributions are the following:

- We propose UA-PCBF, the first framework to integrate epistemic prediction uncertainty directly into predictive CBFs by embedding it in both the safe-set definition and QP constraints, enabling dynamic safety margin adjustment without assumptions about future uncertainty evolution.
- Our UA-PCBF framework is implemented on a full 6-DoF robotic manipulator, demonstrating its feasibility and effectiveness in complex, realistic collaborative settings beyond simplified or low-dimensional scenarios;
- We incorporate the uncertainty of human future movement to plan the robot movement, using a novel forecasting approach, providing proactive safety in HRI scenarios;
- Our method shows superior performance in terms of the number and gravity of robot's safe-space violations compared to state-of-the-art approaches, thereby improving both safety and efficiency in shared workspaces;
- Our setup requires no markers or wearable devices, relying solely on vision-based 3D hand pose estimation. This makes the system practical and easily extensible to real-world industrial or collaborative applications.

The rest of the paper is structured as follows: Section 2 briefly recalls the literature on CBFs and forecasting in HRI scenarios, motivating our work. Section 3 presents our framework, from the 3D hand trajectory forecasting to the UA-PCBF. Section 4 describes our experimental setups, and Section 5 shows the results and discusses them. Finally, Section 7 draws some conclusions of our work and presents some final considerations.

2. Related works

In this section we review the state of the art that motivates the introduction of our approach. In Section 2.1 we survey Control Barrier Functions, ranging from their classic reactive formulation and exploring

their predictive, stochastic and uncertainty-aware extensions, highlighting both strengths and shortcomings in dynamic, safety-critical settings. Then, in Section 2.3, we examine human motion forecasting methods, motivating the choice of hand trajectory models as opposed to full-body pose predictors, discussing their real-time applicability and treatment of uncertainty. We conclude the review by identifying the unaddressed needs for a unified framework that fuses probabilistic hand-motion prediction with formally guaranteed safety, paving the way for our UA-PCBF formulation.

2.1. Control Barrier Functions

Collision avoidance systems, or more generally avoidance frameworks, have been extensively studied in the literature. A classical example is represented by Velocity Obstacles [12], which require a geometric representation of obstacles [13]. Although Velocity Obstacles have been widely used, they often lack formal guarantees of safety and may become overly conservative or poorly scalable in complex dynamic settings. In contrast, CBFs have emerged as a superior alternative because they allow the specification of safe-sets and provide mathematical guarantees of forward invariance [14].

However, traditional CBFs are inherently reactive, guaranteeing safety only in the current state without explicitly considering the future evolution of the system [15]. This myopic behavior can lead to situations where large or rapid control actions are required, potentially resulting in unsafe interactions with humans or necessitating overly conservative safe sets that degrade task performance [16].

To address these limitations, several extensions to the standard CBF framework have been proposed. High Order CBFs [17] and Exponential CBFs [18] introduce implicit look-ahead capabilities through the design of class-$\mathcal{K}$ functions. While these methods offer improved performance [16–18], they do not explicitly model the system's future behavior and rely heavily on the careful tuning of class-$\mathcal{K}$ functions, which can be challenging in practice.

A more direct approach to incorporating future system behavior is through Model Predictive Control (MPC), which has been combined with CBFs [19]. For instance, the Backup-CBF framework introduces backup control strategies to ensure safety when nominal plans fail [20]. However, this approach can be overly conservative and requires the explicit design of backup controllers and sets. Similarly, in the work of Davoodi et al. [21], another integration of MPC and CBFs is presented to ensure safe motion during human–robot interaction, where movements are modeled using Probabilistic Movement Primitives (ProMPs). Predictive CBFs (PCBFs) [22], on the other hand, offer a less conservative alternative by encoding the safety of a nominal trajectory over a receding horizon directly into a CBF structure. This allows the controller to proactively adjust the trajectory before it becomes unsafe [23], without requiring predefined backup strategies.

While MPC-based methods provide a structured way to reason about future safety, they often involve solving complex optimization problems whose computational cost scales up with the length of the prediction [23]. To mitigate this, alternative model-based approaches have been proposed. For instance, Future-Focused CBFs [24] aim to minimize unnecessary interventions by evaluating safety along a predicted trajectory, though they have primarily been applied in autonomous driving contexts. Environment-aware CBFs (ECBFs) [25] and observer-based methods [26] enhance robustness to state estimation errors, particularly in dynamic environments with moving obstacles. However, these methods typically assume knowledge of the environment or the statistical properties of measurement errors, which limits their applicability in HRI scenarios.

Gaussian CBFs [27] learn a candidate barrier function by online training a Gaussian Process (GP) on noisy safety evaluations, tightening the barrier with a confidence-bound term proportional to the posterior variance of the GP. Uncertainty-Separated CBFs [28] build upon this by fitting two independent GPs, one for residual dynamics and one for the barrier, mixing their posteriors into the Quadratic Program (QP) constraints to guarantee robustness under state-dependent perturbations. However, both approaches enforce safety only instantaneously, without looking ahead over a prediction horizon, making them reactive and often overly conservative in highly dynamic environments. Furthermore, they cannot dynamically adjust the safety margin based on prediction confidence, since disturbances are treated as unknown physical functions, rather than epistemic uncertainty.

In contrast to prior work, our approach does not assume any specific structure or predictability of the input disturbances. This feature is essential for integrating user motion forecasting models into the method, where it is not possible to accurately model the disturbance characteristics. Building on the PCBF framework introduced in [10, 23], we propose an extension that incorporates prediction uncertainty estimation into the safety margin itself. To the best of authors' knowledge, this is the first application of predictive CBFs that accounts for prediction uncertainty in an epistemic fashion, without requiring assumptions about its future evolution, by integrating it into the definition of the safe set. This enables proactive safety enforcement in uncertain environments, making the approach particularly suitable for HRI applications.

2.2. Ensuring safe interaction

While CBFs provide formal safety guarantees through forward invariance, alternative frameworks have long addressed safe human–robot interaction. Among the most established are Artificial Potential Fields (APFs) [29] and Time-to-Collision (TTC)–based methods, which offer intuitive, model-free collision avoidance. APFs create artificial potential landscapes combining attractive and repulsive forces to drive motion. They are simple, reactive, and efficient but prone to local minima and oscillations. Recent studies have shown that many APFs can be reformulated as Control Barrier Functions, thus inheriting formal safety properties [30]. TTC-based methods instead assess safety temporally, estimating the time before potential impact under constant-velocity assumptions. When the predicted TTC falls below a threshold, avoidance maneuvers are triggered [31]. Although effective for fast reactive control, TTC lacks predictive accuracy for nonlinear human motion and formal safety guarantees.

2.3. Forecasting in HRI

The majority of forecasting approaches in HRI focus on full-body pose prediction, where future human movements are estimated using whole-body skeletal representations [3,32,33]. While this provides an overview of human motion dynamics, it often lacks the granularity required for fine-grained interactions, particularly those involving the hands.

Full-pose forecasting typically leverages standard 3D human pose estimation methods, which are well-established for capturing body motion but are often coarse and fail to include precise hand dynamics. Many pose forecasting approaches rely on datasets that do not explicitly capture hand kinematics, limiting their usefulness in scenarios requiring dexterous manipulation or hand–object interaction [34,35]. Since the hands are the primary means of physical interaction in HRI tasks, ignoring them restricts the ability to model fine-grained human intent and limits the effectiveness of predictive models in collaborative tasks such as handovers, object manipulation, and shared tool use.

To incorporate hands in motion forecasting, two primary approaches can be considered: predicting the full skeletal hand along with the body pose [36] or forecasting the 3D hand trajectory, which represents the spatial movement of the hand in an interaction volume. The first one often requires the full person to be visible by the system and can be computationally intensive. The latter is widely adopted, particularly in Virtual Reality (VR) [37] and head-mounted device setups [38,39]. The latter are the most appropriate for real-time performance, and optimal

hand observation point of view. However, head-mounted cameras suffer from self-occlusions, narrow fields of view, and the burden of wearable hardware. By contrast, our system uses a desktop-mounted Leap Motion sensor—an infrared stereoscopic tracker that delivers high-frequency, low-latency 3D hand poses without any head-worn equipment. This setup offers more consistent spatial coverage, reduces line-of-sight failures, and eliminates the user discomfort and calibration overhead inherent to egovision solutions.

Many trajectory forecasting approaches achieve high accuracy at the expense of heavy computational requirements, making them unsuitable for real-time applications [40–42]. To bridge this gap, we develop a lightweight and real-time forecasting model that can efficiently predict hand trajectory while ensuring fluid and responsive interaction.

3. Methodology

Our goal is to enforce a minimum separation between the robot and the operator in the context of interaction tasks. We use simple geometric models to ensure fast and reliable distance computations, making the approach practical for real-time safety. More complex skeletal models could also be integrated into the framework by appropriately defining the distance function.

Our safety objective is to maintain a minimum distance d_{min} between:

- a sphere of radius r_{hand} enclosing the operator's hand, and
- a cylinder of radius r_{cyl} and height h_{cyl} enclosing the robot's last link and end-effector

The instantaneous distance, which will be used in the definitions of the barrier functions, is as follows:

$$d(\tau, x) = \|\mathbf{o} - \mathbf{c}(x)\| - r_{cyl} \tag{1}$$

where $\mathbf{o}$ is the center of the sphere, $\mathbf{c}(x)$ is the point on the cylinder's axis that is closest to the sphere, obtained by projecting $\mathbf{o}$ onto the cylinder's axis, and x is the state of the system, defined as the actual pose of the robot's Tool Center Point (TCP).

We provide this safety guarantee in a predictive, uncertainty-aware way. To achieve this, we first introduce a deep learning-based human hand forecasting model in Section 3.1. We then use the model to infuse a novel CBF formulation with probabilistic and predictive capabilities in Section 3.2.

The full diagram block of our framework, with the role of each component, is shown in Fig. 1.

3.1. 3D hand trajectory forecasting

The task of 3D hand trajectory forecasting is to predict future hand positions from past observations; here, we use the palm center as a proxy, represented in global Cartesian coordinates, since downstream modules reconstruct the full hand volume starting from the palm. During HRI tasks, robots must react within milliseconds to human unpredictable motion. To meet these stringent latency requirements, we designed a compact, autoregressive LSTM-based neural model that runs in real-time (see Section 4.3) while striking an optimal trade-off between efficient forecasting and capturing the inherent unpredictability of human motion.

3.1.1. Problem formulation

At each discrete time step t, the palm position is denoted

$$\mathbf{p}_t = (x_t, y_t, z_t) \in \mathbb{R}^3, \tag{2}$$

where (x_t, y_t, z_t) are the global 3D coordinates of the palm center. Given a historical window of length T_{in},

$$\mathcal{P}_{\text{in}} = \{\mathbf{p}_{t-T_{\text{in}}+1}, \mathbf{p}_{t-T_{\text{in}}+2}, \cdots, \mathbf{p}_t\}, \tag{3}$$

the objective is to forecast the next T_{out} positions,

$$\mathcal{P}_{\text{out}} = \{\mathbf{p}_{t+1}, \mathbf{p}_{t+2}, \cdots, \mathbf{p}_{t+T_{\text{out}}}\}. \tag{4}$$

3.1.2. Model overview

We propose a lightweight encoder–decoder recurrent architecture $\mathcal{F}_\theta$ designed to predict future palm trajectories while modeling motion uncertainty. The network processes an observed sequence of 3D palm positions and predicts, for each future time step, a Gaussian distribution parameterized by a mean and a log-variance for each coordinate:

$$\mathcal{F}_\theta : \mathbb{R}^{B\times T_{\text{in}}\times 3} \longrightarrow \Big(\underbrace{\mu}_{\in \mathbb{R}^{B\times T_{\text{out}}\times 3}}, \underbrace{\log\sigma^2}_{\in \mathbb{R}^{B\times T_{\text{out}}\times 3}} \Big), \tag{5}$$

where B is the batch size, and T_{in}, T_{out} denote input and output sequence lengths, respectively. The encoder consists of a two-layer LSTM with hidden size 128, which encodes the observed motion history into a latent state. The decoder, also a two-layer LSTM with hidden size 128, is initialized with the encoder's final hidden and cell states and generates future positions autoregressively. At each decoding step, the output is passed through a linear layer of dimension $128 \to 6$, producing both the 3D mean and log-variance $(\mu_t, \log\sigma_t^2)$. The predicted mean μ_t is then fed back as input for the next time step, yielding an autoregressive probabilistic forecast of the future palm trajectory.

3.1.3. Architecture

The model comprises three modules: (i) an *encoder* that compresses the observed 3-D coordinates into a latent vector, (ii) an *autoregressive decoder* that unfolds this latent information step-by-step to produce a sequence of latent vectors o_k, and (iii) a *probabilistic head* that maps each o_k to a distribution $[\mu_k, \log\sigma_k^2]$, where μ_k is the predicted palm position and $\log\sigma_k^2$ quantifies its uncertainty (see Fig. 2).

The encoder is implemented as an LSTM block [11]

$$(h_{T_{\text{in}}}^{\text{enc}}, c_{T_{\text{in}}}^{\text{enc}}) = \text{LSTM}_{\text{enc}}(\mathcal{P}_{\text{in}}),$$

where LSTM_{enc} has input size 3, hidden size H, and L layers, so that $(h_t^{\text{enc}}, c_t^{\text{enc}}) \in \mathbb{R}^{L\times B\times H}$ for $t = 1, \ldots, T_{\text{in}}$.

To generate the future trajectory latent vector, we autoregressively generate one step at a time for $t = 1, \ldots, T_{\text{out}}$. Let the encoder provide the final hidden state and denote the last observed palm position as

$$(h_0^{\text{dec}}, c_0^{\text{dec}}) = (h_{T_{\text{in}}}^{\text{enc}}, c_{T_{\text{in}}}^{\text{enc}}), \qquad \mathbf{x}_0^{\text{dec}} = \mathbf{p}_t.$$

For each prediction step $k = 1, \ldots, T_{\text{out}}$ we iterate

$$\big(o_k, h_k^{\text{dec}}, c_k^{\text{dec}}\big) = \text{LSTM}_{\text{dec}}\big(\mathbf{x}_{k-1}^{\text{dec}}, (h_{k-1}^{\text{dec}}, c_{k-1}^{\text{dec}})\big), \tag{6}$$

$$[\mu_k, \log\sigma_k^2] = W_o\, o_k + b_o, \quad o_k \in \mathbb{R}^{B\times 1\times H}, \tag{7}$$

$$\mathbf{x}_k^{\text{dec}} = \begin{cases} \mathbf{p}_t & \text{if k=0 (\textit{i.e.}, the first iteration),} \\ \mu_{k-1} & \text{otherwise,} \end{cases} \tag{8}$$

with Eq. (7) being the probabilistic head based on a multi-perceptron layer with W_o weights and b_o bias.

Finally, stacking over $t = 1, \ldots, T_{\text{out}}$ yields the output $[\mu, \log\sigma^2]$, with $\mu \in \mathbb{R}^{B\times T_{\text{out}}\times 3}$ and $\log\sigma^2 \in \mathbb{R}^{B\times T_{\text{out}}\times 3}$.

3.1.4. Training objective

Denote the ground-truth future sequence by $\mathbf{p}^{\text{true}} \in \mathcal{P}_{\text{out}}$. We minimize a composite loss of Gaussian negative log-likelihood (NLL) and mean squared error (MSE). The NLL is defined as

$$\mathcal{L}_{\text{NLL}} = \frac{1}{B}\sum_{i=1}^{B}\sum_{t=1}^{T_{\text{out}}}\left[\frac{1}{2}\log\sigma_{i,t}^2 + \frac{(\mathbf{p}_{i,t}^{\text{true}} - \mu_{i,t})^2}{2\,\sigma_{i,t}^2}\right], \tag{9}$$

and the MSE is defined as

$$\mathcal{L}_{\text{MSE}} = \frac{1}{B\,T_{\text{out}}}\sum_{i=1}^{B}\sum_{t=1}^{T_{\text{out}}}\|\mu_{i,t} - \mathbf{p}_{i,t}^{\text{true}}\|_2^2. \tag{10}$$

The final loss is $\mathcal{L} = \rho\,\mathcal{L}_{\text{NLL}} + \omega\,\mathcal{L}_{\text{MSE}}$, with hyperparameters $\rho, \omega > 0$.

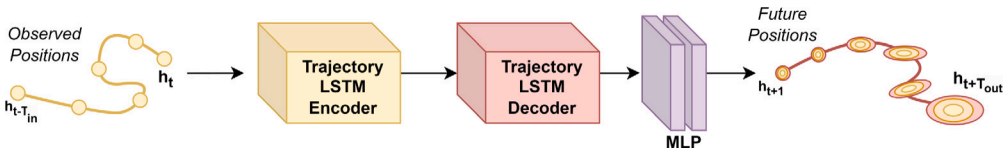

Fig. 2. The architecture of the hand forecasting neural network. The model is composed of an LSTM-based encoder for spatial–temporal features and an autoregressive LSTM-based decoder that produces the final latent space states for each future prediction. Finally, the final states are used to produce the mean and log-variances of the output sequence.

The Gaussian NLL term not only encourages accurate mean predictions but also explicitly penalizes mis-estimated variances, thereby guaranteeing non-degenerate uncertainty estimates. Concretely, the NLL contains a $\frac{1}{2}\log\sigma^2$ penalty which diverges if $\sigma^2 \to 0$, and a residual term $\frac{(\mathbf{p}^{\text{true}}-\mu)^2}{2\sigma^2}$ which blows up if the variance is underestimated when the mean prediction errors. This creates an inherent trade-off: the model may only predict very low variance when it is confident that its mean is highly accurate, else the loss becomes prohibitively large. As a result, the network is driven to learn meaningful, data-dependent uncertainty rather than collapsing to a trivial "mean with zero variance".

3.2. Control Barrier Functions

We consider the control-affine system

$$\dot{x} = f(x) + g(x)\,u \tag{11}$$

and define a safe set $\mathcal{S} = \{x : h(x) \leq 0\}$ via a continuously differentiable barrier function h. Given a nominal control signal u_{nom}, we compute the control signal $u^\star$ that best approximates it while ensuring that the state of the system never leaves the safe set, by solving the following QP

$$\begin{aligned} u^\star = \arg\min_{u\in\mathbb{R}^m} \quad & \frac{1}{2}\|u-u_{nom}\|^2 \\ s.t. \quad & L_g h\, u \leq -L_f h - \alpha(h) \end{aligned} \tag{12}$$

where $L_f h$ and $L_g h$ are the Lie derivatives of h along f and g, respectively, and $\alpha(h)$ is a class-$\mathcal{K}$ function. When $h(x)$ is well below zero, the constraint is inactive and $u^\star \approx u_{nom}$. As $h(x)$ approaches the boundary, the QP smoothly adapts $u^\star$ to maintain safety.

While effective, this approach is myopic since it enforces the safety constraints only at the current state, which can lead to over-conservatism or infeasibility in constrained scenarios.

3.3. Predictive Control Barrier Functions

To address the limitations of standard CBFs, Predictive CBFs (PCBFs) introduce a receding-horizon safety metric, evaluating safety along a predicted nominal trajectory over a finite time horizon $[t, t+T_{out}]$.

Given a nominal control policy $u_{nom}(t,x)$, the evolution of the system over a future horizon T_{out} is defined as

$$\frac{d}{d\tau}p(\tau,t,x) = f(p(\tau,t,x)) + g(p(\tau,t,x))u_{nom} \tag{13}$$

where $p(\tau,t,x)$ is a path function representing the state of the system at time τ assuming the system is driven by u_{nom}.

To assess whether the predicted evolution of the system is safe or not, the barrier function is evaluated over the entire trajectory in the $[t, t+T_{out}]$ time horizon, resulting in a Predictive Control Barrier Function

$$h_p(\tau,t,x) = h(\tau, p(\tau,t,x)) - m(R(\tau,t,x)-t) \tag{14}$$

where $h(\tau,p(\tau,t,x))$ is the function that measures safety along the path, $R(\tau,t,x)$ is the earliest time on the horizon where a safety violation might occur, and m is a class-$\mathcal{K}$ margin function. Formulated in this way, the predictive barrier function evaluates not only how close the robot is to the safety boundary in space, but also how soon that boundary will be reached following the predicted trajectory. This anticipatory behavior is encoded by the margin function, which penalizes imminent violations more strongly than distant ones, effectively pulling the base barrier h towards or away from zero.

The control signal that adjusts the evolution of the system is then obtained by solving the following constrained QP

$$\begin{aligned} u^\star_{pcbf} = \arg\min_{u\in\mathbb{R}^m} \quad & \frac{1}{2}\|u-u_{nom}\|^2 \qquad \forall\tau\in[t,t+T_{out}] \\ s.t. \quad & L_g h_p\, u \leq -L_f h_p - \alpha(h_p). \end{aligned} \tag{15}$$

PCBFs improve upon CBFs by adopting a proactive strategy that considers the predicted evolution of the system, enabling early corrective actions and reducing overly conservative behaviors. However, PCBFs assume that the predicted trajectory is deterministic and accurate, disregarding the uncertainty inherent in real-world forecasting models, especially in highly dynamic environments that are commonly encountered in HRI.

3.4. Uncertainty Aware-Predictive Control Barrier Functions

To enforce both reactive and predictive safety constraints under uncertainty, we now introduce our Uncertainty Aware-Predictive Control Barrier Functions (UA-PCBFs). In UA-PCBFs, the uncertainty is used to dynamically scale a predictive barrier function, relaxing and tightening safety requirements based on the confidence of the prediction. We then combine a reactive and a predictive constraint through the introduction of slack variables. Embedding these uncertainty-aware constraints into a single QP allows us to obtain a controller that smoothly balances responsiveness and safety compliance.

3.4.1. Treating forecast uncertainty

To introduce uncertainty into the barrier function formulation, we first recover the covariance matrix from the forecaster's log-var output at time τ

$$\sigma^2(\tau) = \exp\left(\frac{1}{2}\log\sigma^2(\tau)\right) \in \mathbb{R}^3 \implies \Sigma(\tau) = \text{diag}(\sigma^2(\tau)) \in \mathbb{R}^{3\times3}. \tag{16}$$

Since we are dealing with interaction tasks we focus on the direction of interaction, which is identified by the unit vector u that goes from the cylinder to the center of the hand, along the shortest path. By projecting the covariance matrix $\Sigma(\tau)$ along u, we get a scalar measure of dispersion that the covariance matrix induces on the interaction axis

$$\sigma_{proj}(\tau,x) = u^T\,\Sigma(\tau)u, \quad u = \frac{\mathbf{o}-\mathbf{c}(x)}{\|\mathbf{o}-\mathbf{c}(x)\|} \tag{17}$$

To handle the cases in which the forecaster overestimates the log-variance, we clamp the obtained projected variance to d_{min}, ensuring that uncertainty is never overestimated at the control phase

$$\overline{\sigma}(\tau,x) = \begin{cases} 0 & \tau = 0 \\ \gamma\sigma_{proj}(\tau,x) & \gamma\sigma_{proj}(\tau,x) < d_{min} \\ d_{min} & \text{otherwise,} \end{cases} \tag{18}$$

where γ is a parameter that allows for the tuning of the influence of the uncertainty on the controller. The uncertainty at time $\tau = 0$ is set to zero by definition, since the first point in the future trajectory will always be the current position of the operator's hand, and thus it is not affected by prediction accuracy.

3.4.2. Uncertainty-aware barrier function

We can now define the probabilistic barrier function h_{ua}

$$h_{ua}(\tau, x) = d_{min} + \overline{\sigma}(\tau, x) - d(\tau, x) \quad \tau \in [t, t + T_{out}] \tag{19}$$

The introduction of the uncertainty $\overline{\sigma}$ results in a dynamic modulation of the safety distance between robot and operator. When the forecasting is uncertain about its prediction, the barrier function adapts itself by ensuring a higher safety distance is maintained, and vice versa.

To ensure rapid reaction time, we mix an instantaneous barrier function, based on the current hand position, and a predictive barrier function. We achieve this by introducing slack variables δ_r and δ_p in the two constraints we impose on the QP

$$\begin{aligned} L_g h_{ua}(0, x)\, u &\leq -L_f h_{ua}(0, x) - \alpha(h_{ua}(0, x)) + \delta_r, \quad \delta_r \geq 0 \\ L_g h_{ua}(\tau, x)\, u &\leq -L_f h_{ua}(\tau, x) - \alpha(h_{ua}(\tau, x)) + \delta_p, \quad \delta_p \geq 0. \end{aligned} \tag{20}$$

The slack variables δ_r and δ_p affect the reactive (instantaneous) and predictive barrier functions, respectively. They are nonnegative quantities, introduced to ensure the feasibility of the problem by allowing minimal, well-controlled violations when either constraint becomes too restrictive.

The control input is then computed by solving the QP

$$\begin{aligned} u^{\star}_{ours} = \underset{u \in \mathbb{R}^m}{\arg\min} \quad & \frac{1}{2}\|u - u_{nom}\|^2 + \lambda_r \delta_r^2 + \lambda_p \delta_p^2 \\ s.t. \ \ & \text{Eq. (20)} \end{aligned} \tag{21}$$

where λ_r and λ_p are penalty parameters that discourage the violation, respectively of the reactive and predictive constraints, unless strictly necessary. As such, the slack variables in the QP act as decision variables, allowing the controller to dynamically relax the safety constraints based on current task conditions and predicted future risk. We set the penalty terms as

$$\lambda_r = 100, \quad \lambda_p = \lambda_r - \frac{\gamma \overline{\sigma}}{d_{min}}. \tag{22}$$

The penalty terms λ_r and λ_p were tuned empirically to ensure balance between strict safety enforcement and numerical stability of the QP. In practice, smaller values led to frequent constraint relaxation, thus increasing violations. Conversely, larger values resulted in infeasibility or abrupt control actions due to the increased strictness in enforcing the safety constraints.

The penalty for the reactive term, λ_r, is fixed to always ensure that the optimizer avoids relaxing the constraint related to the current position of the hand. The penalty for the predictive term λ_p is influenced instead by the uncertainty in the prediction itself. When the uncertainty is low $\lambda_p \approx \lambda_r$ so the optimizer is discouraged from violating the predictive constraint. In contrast, when the uncertainty is high $\lambda_p \approx \lambda_r - \gamma$ so the optimizer is allowed to violate the constraint in a controlled manner.

In summary, the uncertainty coming from the prediction model is integrated in two ways: directly into the barrier function, by dynamically modulating the safety margin, and indirectly inside the QP, allowing the controller itself to relax or tighten the predictive constraint when needed.

3.5. Robustness

The proposed UA-PCBF constraint inherits and extends robustness to disturbances from the base barrier function formulation.

Consider the addition of a bounded disturbance w on the system dynamics defined by Eq. (11), with $\|w\| \leq w_{max}$. The time derivative of the barrier function under this disturbance is:

$$\dot{h} = L_f\, h(x) + L_g\, h(x) u + \nabla h(x)^T w \tag{23}$$

The disturbance contribution $\nabla h(x)^T w$ can be upper-bounded for all admissible w as:

$$\nabla h(x)^T w \leq \max_{\|w\| \leq w_{max}} \nabla h(x)^T w = w_{max} \|\nabla h(x)\| \tag{24}$$

Substituting Eq. (24) into Eq. (23) yields the CBF condition:

$$L_f\, h(x) + L_g\, h(x) u + \alpha(h(x)) \leq -w_{max} \|\nabla h(x)\| \tag{25}$$

which guarantees the forward invariance of the safe set S under any disturbance bounded by w_{max}. This is achieved by strengthening the nominal barrier constraint proportionally to the worst case scenario of disturbance.

An analogous robustness notion can be introduced for the uncertainty in the predicted human motion. A stochastic safety requirement can then be defined as:

$$\Pr[d(\tau, x) \geq d_{min}] \geq 1 - \varepsilon \tag{26}$$

where ε is the admissible probability of a constraint violation. Assuming the random variable $d(\tau, x)$ can be approximated as Gaussian with mean $\mu(\tau, x)$ and variance $\sigma^2(\tau, x)$:

$$\Pr[d(\tau, x) \geq d_{min}] = 1 - \Phi\left(\frac{d_{min} - \mu(\tau, x)}{\sigma(\tau, x)}\right) \tag{27}$$

where Φ is the cumulative distribution function. To satisfy the stochastic requirement, it suffices to require:

$$\frac{d_{min} - \mu(\tau, x)}{\sigma(\tau, x)} \leq z_{1-\varepsilon} \implies \mu(\tau, x) \geq d_{min} + z_{1-\varepsilon} \sigma(\tau, x) \tag{28}$$

where $z_{1-\varepsilon}$ is the standard normal quantile.

This formulation corresponds to the uncertainty-dependent margin introduced in Eq. (19), where the quantile multiplier $z_{1-\varepsilon}$ is implemented through the parameter γ. Hence, the UA-PCBF framework naturally satisfies the chance-constrained robustness condition defined above.

4. Experimental setup

This section details the experimental protocols that we used to validate both the hand-forecasting model and the proposed UA-PCBF approach, in progressively more realistic settings. In Sections 4.1–4.2, we describe the data collection setup, which uses a Leap Motion v2 sensor to record 3D hand trajectories under different motion patterns. These recordings are structured into a dataset that is then used in training and testing the deep learning-based forecasting model. We present the model's architecture and the related evaluation metrics. Finally, in Section 4.4 we present the two real-world scenarios in which the proposed UA-PCBF framework was evaluated: a highly repeatable mockup experiment using a hand model actuated by a 6 DoF manipulator, and a human-in-the-loop handover task assessing performance in realistic HRI conditions. Fig. 4 shows the robotic cell where the real-world experiments are performed. In Fig. 4(a), the reference frames of each experiment unit: the robot's base and TCP, and the Leap Motion, can be seen. In Fig. 4(b), the bounding regions for both robot and operator are shown.

For each scenario, we define quantitative metrics spanning spatial accuracy, motion efficiency, and safety compliance. We conclude the section with implementation specifics on both the forecasting network and the UA-PCBF robot controller implementation.

4.1. Hand tracking data

To track the hand position, we rely on the Leap Motion v2 device. This device can track the hands' tracking data in real-time (up to 120 Hz). We leverage this tool to obtain the motion trajectory of the hand, both in real-world experiments and for data collection for the training data. Using the Leap Motion v2, we created a dataset consisting of several trajectories of human hand motion and orientation, for a total of 160k individual samples, collecting over 33k subsequences of 1 s each of hand positions at 30 Hz. Considering all the subsequences, we have a per-sequence average speed of 0.1664 ± 0.1210 m/s with a maximum observed speed of 7.4130 m/s, and a per-sequence average acceleration of 1.4448 ± 1.0008 m/s^2, with a maximum observed acceleration of 256.4722 m/s^2. The data will be released with the code upon acceptance for applicability.

4.2. Forecasting system

The forecasting model, detailed in Section 3.1, has been trained on the dataset collected using the Leap Motion v2 device (see Section 4.1). The next sections detail how the forecasting system is evaluated against different time horizons and which metrics are used to evaluate it.

To assess the forecasting performance, we trained the model on various time horizons, and we opted for 1000 ms for the real-world experiments, as is typical in human–robot interaction and collaboration tasks [3].

Trajectory forecasting is measured in terms of final displacement error (FDE) and average displacement error (ADE), standard metrics in trajectory forecasting [43]. These metrics quantify how well the predicted hand trajectory aligns with the ground truth over a given prediction horizon.

The ADE measures the average Euclidean distance between the predicted and ground truth trajectory points over all time steps

$$ADE = \frac{1}{T_{\text{out}}} \sum_{t=1}^{T_{\text{out}}} \|\mathbf{h}_t^{\text{pred}} - \mathbf{h}_t^{\text{true}}\|_2, \tag{29}$$

where T_{out} is the number of future time steps, and $\mathbf{h}_t^{\text{pred}}, \mathbf{h}_t^{\text{true}} \in \mathbb{R}^3$ are the predicted and ground truth hand positions at time t.

The FDE evaluates the positional error at the final predicted time step

$$FDE = \|\mathbf{h}_{T_{\text{out}}}^{\text{pred}} - \mathbf{h}_{T_{\text{out}}}^{\text{true}}\|_2. \tag{30}$$

The model can run at 30 Hz, processing the Leap Motion data with a sliding window of the last N tracking events and producing the subsequent sequence of M positions and rotations, as explained in Section 3.1

4.3. Implementation details

This section presents implementation details for the two main software components: the 3D hand forecasting and the robotic control.

4.3.1. 3D hand forecasting

The 3D hand forecasting model was trained using Python 3.11 and CUDA 12.1 on an Ubuntu 22.04 system equipped with an NVIDIA RTX 3090 GPU. Training was performed for a total of 200 epochs using the Adam optimizer with an initial learning rate of 1×10^{-3}. A cosine annealing learning rate scheduler was employed, progressively reducing the learning rate from 1×10^{-3} to 1×10^{-6} over the course of training. Mini-batches of size 128 were used throughout. All random seeds for PyTorch, NumPy, and Python were fixed to ensure reproducibility. The dataset was divided into 80%–20% data split over the collected data (see Section 4.1). The model was trained for 200 epochs with $T_{\text{in}} = 30$ observed frames predicting $T_{\text{out}} = 30$ future frames. Inference was conducted on an RTX 3060 GPU.

4.3.2. Robot and control

The experiments were carried out with a JAKA ZU 5 6-DoF robotic manipulator, controlled through a dedicated workstation. Fig. 4 shows the robot placed in its collaborative cell, together with the relevant reference frames.

To control the robot, a ROS2 Jazzy environment was set up on the workstation. Dedicated ROS2 nodes were then wrote to encapsulate JAKA's Python SDK v.2.2.2, which exposes API functions related to robot status, control and communication. The only exception were the functions related to kinematics (i.e. direct/inverse kinematics and Jacobian computation), of which the SDK's implementations were deemed either too slow, in the case of direct/inverse kinematics, or non-existent, in the case of Jacobian computation. For those functions, the Robotics Toolbox for Python [44] was used.

The robot was controlled with a task space proportional velocity controller, which generates the nominal control u_{nom} used in the various method formulations. For the barrier function methods, the QP was set up and solved using the CVXPY python package. The default OSQP solver was used, without altering the default parameters (max iterations $max\_iter = 10000$, absolute/relative accuracy $eps\_abs = eps\_rel = 1e^{-5}$)

Table 1
Summary of the parameters used during the runs of the mock hand experimental scenario.

Method	α (h)	m (τ)	γ	λ_p
CBF	125 h	–	–	–
PCBF	125 h	τ^2	–	–
UA-PCBF	125 h	τ^2	0	[a]
UA-PCBF	125 h	τ^2	[0,0.5,1,2.5,5]	λ_r
UA-PCBF	125 h	τ^2	[0,0.5,1,2.5,5]	[a]

[a] Computed with Eq. (22).

4.4. Experimental protocol

To evaluate the performance of the proposed framework, two experimental scenarios were designed. In the first one, a hand model was used to collect consistent, repeatable results in simplified conditions. In the second scenario, the system was tested by a human operator, showcasing its applicability in realistic conditions.

4.4.1. Hand mockup experiment

A wooden human hand model was rigidly mounted to the end-effector of a UR5e robotic manipulator to ensure highly repeatable and controlled hand trajectories across experimental trials. The setup is shown in Fig. 3. During each run, the hand executed a sequence of rapid upward motions, characterized by a peak velocity of 1.0 m/s and an acceleration of 3.5 m/s^2. These motion parameters were selected to emulate sudden, high-dynamic hand movements that are particularly challenging for real-time obstacle avoidance algorithms.

Simultaneously, a JAKA robotic arm (i.e., the cobot in our HRI cell) performed a linear sweeping motion above the hand, moving back and forth along a path orthogonal to the hand's trajectory. This setup was designed to simulate a dynamic, shared workspace scenario in which the robot must continuously adapt its motion in response to fast, human-like movements.

Ten experimental trials were conducted, each employing a different control strategy: four baseline methods (APF, TTC, CBF and PCBF), two ablated variants of the proposed method (with $\gamma = 0$ and $\lambda_p = \lambda_r$), and the proposed UA-PCBF approach. The control parameters used in each configuration are summarized in Table 1. For each trial, ten independent runs were recorded to compute mean and standard deviation of each metric.

4.4.2. Baseline methods

To provide a comprehensive evaluation of the performance of the proposed method, two additional, non-control barrier function based baselines were implemented: Artificial Potential Fields (APFs) and a Time-To-Collision safety filter.

Artificial potential fields. The APF baseline superimposes the attraction towards the goal, generated by the nominal controller, with a repulsive potential field centered on the operator's hand. The field produces a velocity correction, activated when the robot–hand distance falls between a safety threshold d_{min}:

$$v_{rep}(\tau, x) = \begin{cases} k_{rep} \left(\frac{1}{d(\tau,x)} - \frac{1}{d_{min}} \right) \frac{1}{d(\tau,x)^2} \mathbf{u} & d < d_{min} \\ 0 & \text{otherwise} \end{cases} \tag{31}$$

where $\mathbf{u}$ is the unit vector from the hand to the TCP and $k_{rep} = 0.1$ is a proportional gain on the repulsive component.

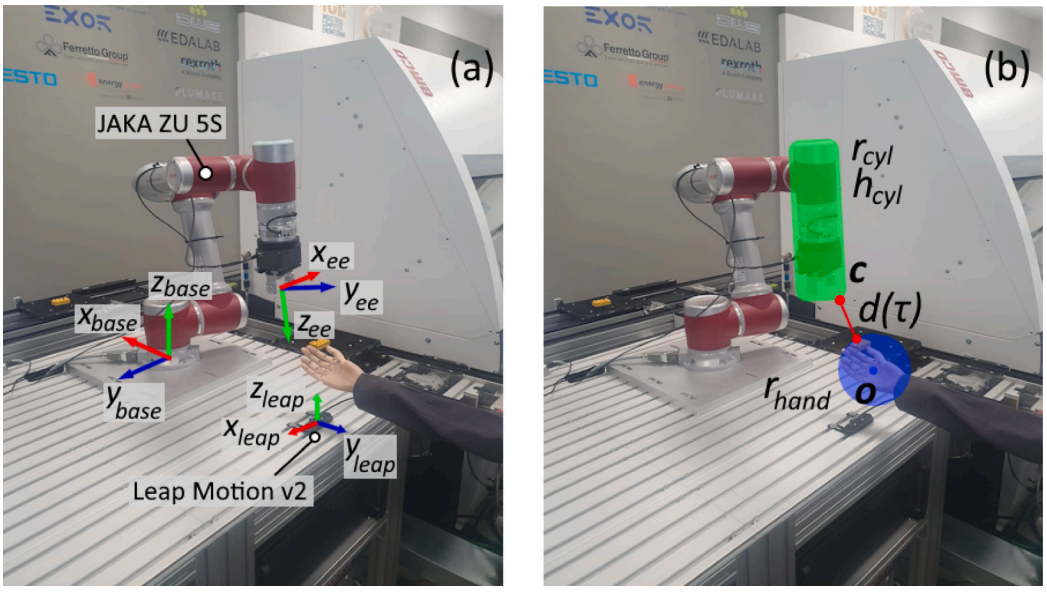

Fig. 3. The experimental setup for the hand mockup experiment. (a) shows the relevant frames of reference. (b) shows the bounding regions of both robot and hand model, with the corresponding distance $d(\tau)$.

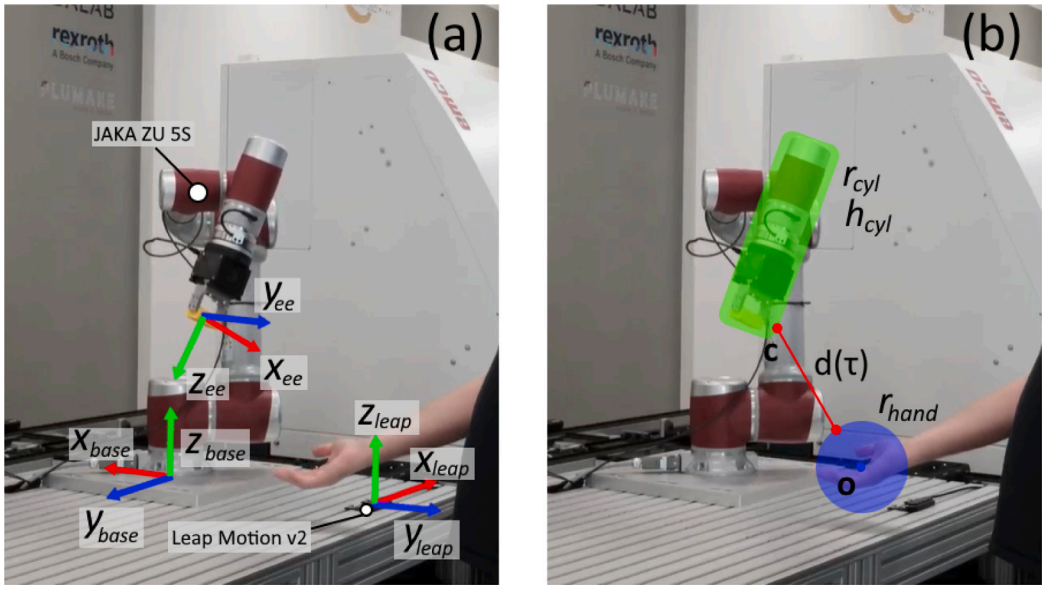

Fig. 4. The experimental setup for the human operator experiment. (a) shows the relevant frames of reference. (b) shows the bounding regions of both robot and operator hand, with the corresponding distance $d(\tau)$.

The corrective velocity is mapped to the joint space through the robot's Jacobian J, and then added to the nominal velocity command:

$$u = u_{nom} + J v_{rep} \tag{32}$$

Time-to-collision safety filter. The TTC filter scales the nominal velocity according to the predicted time to contact between the TCP and the operator's hand.

Let d and $\dot{d}$ be the relative distance and the approach rate between the hand and the TCP. The instantaneous TTC is then:

$$TTC = \begin{cases} \infty & \dot{d} \geq 0 \\ \frac{d}{-\dot{d}} & d < 0 \end{cases} \tag{33}$$

From the TTC, a continuous scaling factor is computed:

$$\beta = min\left(1, \frac{TTC - T_{min}}{T_{max} - T_{min}}\right) \tag{34}$$

The scaling factor is then applied to the nominal joint velocity:

$$u = \beta u_{nom} \tag{35}$$

4.4.3. Human operator experiment

To evaluate the applicability of the method in realistic HRI conditions, a handover task was designed. The robot carried out a pick-and-place operation, picking up an object from a bay and placing it on the operator's left hand. During the motion between the pick and place operations, the operator was asked to replicate the movement carried out by the UR5e manipulator in the hand mockup experiment. Fig. 4 shows the reference frames of the different systems involved in the experiment and the bounding regions, while in Fig. 5 we show the execution of one take of the experiments, with four frames of the task. We show that the right hand is tracked and forecasted, while our method uses the future prediction and its uncertainty to adjust the robot's path planning.

Fig. 5. Example of the framework in a real-world implementation. In this example, the right hand is the only one tracked with the forecasting method. Note that this is a testing choice, not a limitation of the framework. The operator stands in front of the robot, and the hand is tracked in 3D using a Leap Motion V2 device. The device streams the data in real-time to the neural network that predicts the future position of the hand with a 1000 ms time horizon. When the operator moves the robot control, seeing that the hand will collide within the future time horizon, adjusts the planning by keeping a safe distance from the operator's hand. Finally, the delivery is performed on the left hand.

Table 2
Comparison between different types of forecasting methods over different horizons in milliseconds (H). Metrics are presented as mean±standard deviation ($\mu \pm \sigma$) in meters (the lower the better). Leading zero omitted for formatting reasons. Our model outperforms in both ADE and FDE metrics with respect to all other approaches event with a very short horizon.

	Linear interpolation		Kalman filter		Particle filtering		OUR model	
	ADE ↓	FDE ↓	ADE ↓	FDE ↓	ADE ↓	FDE ↓	ADE ↓	FDE ↓
100	.004 ± .009	.007 ± .015	.038 ± .045	.044 ± .051	.007 ± .009	.009 ± .014	**.003 ± .003**	**.005 ± .006**
200	.009 ± .014	.018 ± .028	.046 ± .052	.062 ± .069	.011 ± .013	.020 ± .028	**.006 ± .008**	**.012 ± .016**
300	.016 ± .030	.036 ± .061	.058 ± .061	.084 ± .088	.018 ± .029	.037 ± .061	**.009 ± .012**	**.019 ± .027**
400	.025 ± .034	.056 ± .077	.068 ± .071	.107 ± .110	.026 ± .034	.057 ± .076	**.013 ± .016**	**.028 ± .038**
500	.033 ± .046	.076 ± .101	.078 ± .081	.127 ± .129	.035 ± .045	.077 ± .100	**.016 ± .021**	**.035 ± .048**
566	.037 ± .050	.085 ± .111	.082 ± .083	.134 ± .132	.039 ± .049	.086 ± .110	**.018 ± .024**	**.040 ± .054**
1000	.084 ± .101	.189 ± .218	.129 ± .121	.230 ± .212	.085 ± .101	.189 ± .218	**.034 ± .043**	**.073 ± .086**
1200	.109 ± .134	.238 ± .287	.153 ± .137	.270 ± .242	.110 ± .134	.239 ± .286	**.040 ± .048**	**.084 ± .093**

The registration between the reference frames of the different systems of Fig. 4a was conducted by precise measurement and placement of the Leap Motion with respect to the origin (base) of the robot. The forecasting method matches the Leap's reference system, so the actual registration is a simple rigid transformation of the device.

In this scenario, ten runs per method were collected, using the parameters that showed the best performance with respect to interaction metrics, which will be discussed in the following section.

4.4.4. Metrics

To evaluate the performance of the proposed framework in handling human motion, executing collaborative tasks, and ensuring safety, we consider six interaction-based performance metrics as proposed in [45, 46]. These metrics are grouped into three categories — spatial accuracy, dynamics, and safety compliance — to provide a comprehensive assessment of the human–robot interaction.

1. Spatial accuracy
   - Mean hand–TCP distance (meters): the average Euclidean distance between the palm of the human hand and the robot's Tool Center Point (TCP) over the course of the execution.
   - Total TCP path length (meters): the cumulative length of the trajectory traversed by the TCP during the task.
2. Dynamics
   - Average TCP velocity (meters per second): the mean of the TCP's instantaneous linear speeds during execution.
   - Total execution time (seconds): The overall duration required to complete the task from start to finish.
3. Safety compliance
   - Number of safety limit violations: the total number of instances in which the safety function h exceeds a predefined threshold, indicating a breach of the robot's safety boundary. In our evaluation, this threshold is set to 10 mm.
   - Mean violation magnitude (meters): how far the TCP breached the safety boundary, when a violation occurred, averaged over all breach events.

5. Experimental results

In this section, we provide our empirical findings. In Section 5.1, we compare the hand-trajectory predictor against baselines with respect to average/end-point errors, reporting quantitative forecasting performance. In Sections 5.2–5.3, we analyze the UA-PCBF controller's behavior in both mockup and human-in-the-loop trials, focusing on interaction accuracy, smoothness, and safety-compliance.

5.1. Forecasting results

Table 2 presents the performance of various standard methods to deal with the forecasting problem. All these approaches perform forecasting of T_{out} steps from the position at time t, given the past observation of T_{in} steps. We analyze three alternatives: a simple linear interpolation model, a particle filtering method, and a Kalman filter approach. All of them produce high errors, in particular regarding the reference time horizon of 1000 ms, for which the FDE value, *i.e.*, the final position, has more than 0.18 m of error on average. Instead, our neural approach is more robust and achieves remarkably lower ADE and FDE values, as reported in Table 2, where the metrics on different time horizons are presented. Notably, the 1000 ms reference horizon has an FDE of just 0.073 m on average, considerably lower than traditional approaches.

To assess model capability of estimating the uncertainty of the predicted trajectories, we examinate the "calibration", following the

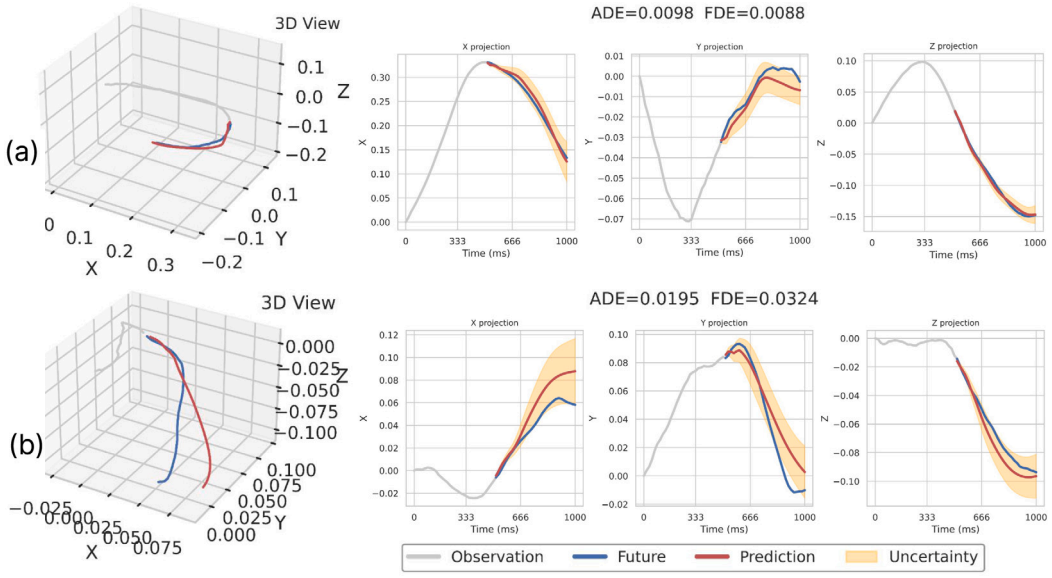

Fig. 6. Qualitative examples of the forecasting model on the data validation set, with uncertainty visualization. Figure (a) shows the model that is perfectly able to follow a curved motion, with low uncertainty in X and Z axes, and a slight uncertainty (less than 2 cm) in the Y axis. Figure (b) shows the model uncertainty power: despite producing a trajectory, the displacement at the final element is high with respect to the ground truth, reflected as well in the ADE and FDE metrics. Nonetheless, the model understands that there could be some error, and the uncertainty is increased significantly towards the end of the trajectory, particularly in the X and Z axes.

procedure described in [47], *i.e.*, we compute for each confidence level (0.90, 0.95, 0.99), the empirical coverage defined as the fraction of the true future points that lie within the corresponding predicted confidence intervals, and then compare these empirical frequencies to their nominal values.

Our results at the 1000 ms horizon are highly encouraging: we observe empirical coverages of 90.7%, 94.3%, and 97.6% for the 90%, 95%, and 99% intervals, respectively. The 90% and 95% intervals nearly coincide with their targets, demonstrating that the model's variance predictions are well tuned to the typical error distribution. Even the 99% interval, despite the greater challenge of capturing rare, large deviations, exhibits only modest under-coverage, indicating that extreme-tail uncertainty is still largely captured.

Furthermore, we show qualitative results of the uncertainty estimation in Fig. 6. In Fig. 6(a), an example of a trajectory with low uncertainty is shown. As can be seen, the model is perfectly able to reconstruct the circular motion with a very low ADE and FDE, and the uncertainty is overall small in value, with the only exception of the Y axis, for which, however, it is between 1–2 cm of uncertainty, therefore quite precise. In Fig. 6(b), we can see a prediction with high ADE and FDE. In this case, the final predicted trajectory is not so close to the real one, and indeed, even if the model produces a trajectory, the uncertainty is considerably higher, denoting the uncertainty awareness intrinsic in the model. A trajectory like the one presented in Fig. 6(b) is the perfect example of why we need a planning framework that takes into account the uncertainty of the hard-to-predict human movement.

5.2. Hand mockup experiment

We describe in this section the results of the hand mockup experiment, first considering the influence of the uncertainty weight parameter γ and subsequently presenting the comparison with the baseline methods and ablated variants of the proposed method.

Table 3 presents the performance of the proposed UA-PCBF method as the uncertainty weight parameter γ varies, considering the values $\gamma \in \{0, 0.5, 1, 2.5, 5\}$. This parameter modulates the influence of the predicted uncertainty in human motion on the robot's control decisions.

Taking into account spatial accuracy, the mean hand–TCP distance remains stable across all values of γ, with the best results (32 cm) consistently achieved for $\gamma \in \{0, 0.5, 1, 5\}$. This indicates that incorporating uncertainty does not compromise the robot's ability to maintain proximity to the human hand.

The total TCP path length and average velocity show a clear improvement as γ increases. The shortest trajectory (1.89 m) and lowest average velocity (0.31 m/s) are achieved at $\gamma = 5$, suggesting that the robot moves more efficiently when uncertainty is fully accounted for.

Similarly, overall task completion time decreases with increasing γ, reaching a minimum of 5.9 s at $\gamma = 5$, which is close to the ideal task duration of 5.3 s. This suggests that higher uncertainty awareness enables the robot to plan more confidently and avoid unnecessary slowdowns.

Considering the system's ability to perform the task with maximum safety, the number and magnitude of safety violations drop significantly as γ increases. At $\gamma = 5$, the robot incurs only 0.2 violations on average, with a minimal breach magnitude of 2 cm. This represents an order-of-magnitude improvement over lower γ values and highlights the effectiveness of uncertainty-aware control in maintaining safety.

In summary, increasing γ enhances both *efficiency* and *safety*, with $\gamma = 5$ yielding the best overall performance. This supports the hypothesis that leveraging uncertainty in human motion prediction leads to more intelligent and fluid robot behavior.

Table 4 compares the full UA-PCBF method (with $\gamma = 5$) against four baselines (APF, TTS, CBF and PCBF) and two ablated variants: one with $\gamma = 0$ (no uncertainty awareness in the barrier function) and one with $\lambda_p = \lambda_r$ (no uncertainty awareness in the QP optimization).

UA-PCBF and the $\lambda_p = \lambda_r$ variant achieve the best hand–TCP distance (31 cm), outperforming both PCBF, TTC (both 32 cm), APF (35 cm) and CBF (37 cm). This suggests that both prediction and uncertainty contribute to maintaining close coordination with the human.

UA-PCBF achieves the shortest trajectory (1.89 m) and the lowest velocity (0.04 m/s) close to the ideal (0.03 m/s), indicating efficient and natural motion. In contrast, PCBF and CBF result in longer paths and, in the case of CBF, a significantly higher velocity (0.23 m/s), likely due to the purely predictive nature of the method, in contrast with the more reactive CBF/UA-PCBF approaches. In terms of overall execution time, UA-PCBF and CBF both complete the task in 5.9 s, but UA-PCBF does so with far fewer safety violations, highlighting its superior balance between speed and safety.

Table 3
Average values of the metrics considered obtained by the UA-PCBF method as the γ parameter varies. Metrics are presented as mean $\pm$ standard deviation ($\mu \pm \sigma$), where values in bold represent the best results.

γ	Hand–TCP distance	Trajectory length	Trajectory velocity	Completion time	Violation count	Violation magnitude
0	**0.32 ± 0.01**	2.17 ± 0.16	0.33 ± 0.02	6.47 ± 0.01	8.4 ± 2.1	0.024 ± 0.005
0.5	**0.32 ± 0.01**	2.07 ± 0.12	0.32 ± 0.01	6.42 ± 0.02	5.2 ± 4.8	0.016 ± 0.1
1	**0.32 ± 0.01**	2.15 ± 0.07	0.33 ± 0.02	6.5 ± 0.02	7 ± 6.8	0.014 ± 0.009
2.5	0.33 ± 0.01	2.14 ± 0.26	0.36 ± 0.03	6 ± 1.01	2.2 ± 2.3	0.014 ± 0.014
5	**0.32 ± 0.01**	**1.89 ± 0.14**	**0.31 ± 0.03**	**5.9 ± 0.3**	**0.2 ± 0.4**	**0.002 ± 0.004**

Table 4
Mockup experiment: average values of the considered metrics obtained from the baseline (i.e. APF, TTC, CBF and PCBF methods) and UA-PCBF method, also considering the variants with $\gamma = 0$ and $\lambda_p = \lambda_r$. Metrics are presented as mean $\pm$ standard deviation ($\mu \pm \sigma$), where values in bold represent the best results.

Method	Hand–TCP distance	Trajectory length	Trajectory velocity	Completion time	Violation count	Violation magnitude
APF	0.35 ± 0.02	2.38 ± 0.24	0.12 ± 0.01	20.3 ± 0.23	91.75 ± 19.38	0.043 ± 0.003
TTC	0.32 ± 0.01	**1.6 ± 0.01**	0.07 ± 0.01	22.74 ± 0.09	144.5 ± 31.4	0.062 ± 0.011
CBF	0.37 ± 0.02	2.33 ± 0.73	0.23 ± 0.13	**16.14 ± 0.99**	36 ± 7.71	0.025 ± 0.003
PCBF	0.32 ± 0.01	2.12 ± 0.07	**0.04 ± 0.01**	49.5 ± 0.2	20.5 ± 5.54	0.023 ± 0.003
Ours ($\lambda_p = \lambda_r$)	**0.31 ± 0.01**	2.04 ± 0.07	**0.04 ± 0.01**	48.9 ± 0.1	7.0 ± 1.8	0.021 ± 0.006
Ours ($\gamma = 0$)	0.33 ± 0.01	2.09 ± 0.15	0.05 ± 0.02	49.2 ± 0.4	8.8 ± 2.5	0.020 ± 0.005
Ours	**0.31 ± 0.01**	1.89 ± 0.06	**0.04 ± 0.01**	48.11 ± 0.62	**3.8 ± 2.15**	**0.014 ± 0.006**

Table 5
Human operator: average values of the considered metrics obtained from the baseline (i.e. CBF and PCBF methods) and UA-PCBF method during interaction with human operator. Metrics are presented as mean $\pm$ standard deviation ($\mu \pm \sigma$), where values in bold represent the best results.

Method	Hand–TCP distance	Trajectory length	Trajectory velocity	Completion time	Violation count	Violation magnitude
CBF	0.39 ± 0.02	2.41 ± 0.11	0.16 ± 0.02	**24.38 ± 0.16**	35.25 ± 10.51	0.036 ± 0.006
PCBF	0.36 ± 0.04	2.22 ± 0.21	0.05 ± 0.01	44.05 ± 0.31	21.13 ± 4.52	0.025 ± 0.006
UAPCBF	**0.34 ± 0.03**	**2.05 ± 0.09**	**0.04 ± 0.01**	46.05 ± 0.59	**2.12 ± 2.15**	**0.008 ± 0.008**

Considering the comparison in terms of safety compliance, UA-PCBF achieves the lowest violation count (0.2) and smallest violation magnitude (0.14 cm), significantly outperforming PCBF (20.5 violations) and CBF (36 violations). The $\gamma = 0$ variant performs poorly in this regard, confirming the critical role of uncertainty modeling. Interestingly, the $\lambda_p = \lambda_r$ variant also performs well in safety, suggesting that even without prediction-based relaxation or tightening of optimization constraints, uncertainty-aware control can significantly enhance safety.

These results demonstrate that UA-PCBF enables robots to navigate the trade-off between responsiveness and safety more effectively than existing methods. By incorporating uncertainty in human motion prediction, robots can plan more confidently, avoid unnecessary braking, and maintain safe, efficient, and fluid interactions in shared workspaces.

5.3. Human operator experiment

Table 5 presents the values of the metrics obtained in the experiment with a human operator, also considering the two baseline methods CBF and PCBF as in the previous mockup experiments.

CBF achieves the lowest average distance (34 cm), followed by PCBF (36 cm) and UA-PCBF (39 cm). This suggests that CBF maintains closer proximity to the human hand, but this may come at the cost of safety. In fact, CBF's high velocity (0.16 m/s) reflects aggressive behavior, potentially unsafe considering the close proximity to humans. Instead UA-PCBF achieves the lowest average velocity (0.04 m/s), lower than those obtained from PCBF (0.05 m/s). While this might suggest an overly conservative behavior of UA-PCBFs, it is not reflected in a reduction in task execution efficiency.

In fact, UA-PCBF yields the shortest trajectory (2.05 m), slightly better than PCBF (2.22 m), and significantly shorter than CBF (2.41 m). This indicates more efficient motion planning under UA-PCBF, an aspect that is also confirmed by the task completion times. UA-PCBF completes the task in 5.34 s on average, faster than PCBF (5.65 s) and comparable to CBF (5.84 s), despite its more conservative velocity.

If we consider the performance in terms of safety compliance, UA-PCBF reduces safety violations to just 2.12 on average, compared to 21.13 for PCBF and 35.5 for CBF. Also, UA-PCBF also achieves the lowest breach depth (0.8 cm), outperforming PCBF (2.5 cm) and CBF (3.6 cm).

The results confirm that UA-PCBF maintains its advantages even in dynamic, human-in-the-loop scenarios. It balances efficiency and safety better than both CBF and PCBF, adapting to unpredictable human motion while minimizing risky behavior. The UA-PCBF method demonstrated slightly reduced performance in the human operator scenario compared to the hand mockup experiment, with a higher average violation count (2 vs. 0.2) and greater violation magnitude (1.2 cm vs. 0.2 cm), reflecting the increased unpredictability of real human motion. The degradation in performance compared to the mockup experiment is expected and underscores the importance of robust uncertainty modeling.

5.4. Feasibility analysis

The robustness of the optimization layer was evaluated through the slack variables δ_r and δ_p, which quantify the relaxation applied to the safety constraints when strict feasibility would otherwise be violated.

Table 6 presents the results of the analysis on the usage of slack variables for both mock hand and human operator experiments. Across all trials, the QP remained feasible, with no numerical failures recorded. Slack activations occurred in less than 5% of the control steps, and the mean active values remained below 10^{-3}, confirming that constraint relaxations were both rare and of negligible amplitude. The results

Table 6
Slack-variable statistics. The table reports slack variable usage as mean±standard deviation ($\mu \pm \sigma$), and maximum slack values for δ_r and δ_p, together with the percentage of timesteps where each slack was active.

Trial	δ_r	δ_r max	δ_r active [%]	δ_p	δ_p max	δ_p active [%]
Mock hand	$8.4 \pm 9.4 \cdot 10^{-4}$	$3.3 \cdot 10^{-3}$	4.65	$2.9 \pm 4.8 \cdot 10^{-4}$	$1.5 \cdot 10^{-3}$	2.88
Human operator	$9.7 \pm 9.1 \cdot 10^{-4}$	$2.8 \cdot 10^{-3}$	4.17	$4.2 \pm 5.1 \cdot 10^{-4}$	$1.5 \cdot 10^{-3}$	3.82

show a comparable behavior between the mockup and operator experiments, indicating consistent safety enforcement under both nominal and realistic conditions. The bounded and infrequent slack activity and the constant feasibility of the optimization problem demonstrate that the UA-PCBF framework maintained numerical stability and efficiently enforced safety constraints throughout all runs.

6. Limitations

Our study exhibits several limitations. First, the experimental scenarios remain highly simplified: initial validation relied on a static hand model executing repeatable trajectories, and even the human-in-the-loop handover task involved only a single operator following prescribed motions, rather than a broad set of unstructured collaborative activities. Second, no formal user testing or usability studies were conducted, so we lack data on how real workers perceive system responsiveness and cognitive load during prolonged interaction. Finally, our safety model employs geometric approximations of the bounding region for the hand and for the robot's end-effector and links.

7. Conclusions

This work introduced Uncertainty-Aware Predictive Control Barrier Functions (UA-PCBFs), a novel framework for enabling safe and efficient human–robot interaction in shared workspaces. We integrate probabilistic human motion prediction with the formal safety guarantees of Control Barrier Functions, allowing robots to reason not only about the most likely future human states but also about the uncertainty surrounding those predictions. This allows us to have a dynamic adjustment of the robot's planning and control, based on the uncertainty estimation of the human motion.

Through a series of controlled experiments involving a robotic manipulator interacting with a human hand model, we demonstrated that UA-PCBFs outperform both classical CBFs and prediction-augmented PCBFs across a range of metrics. Specifically, UA-PCBFs achieved significant reductions in safety violations — both in frequency and magnitude — while maintaining or improving task efficiency and spatial coordination. These results validate the core hypothesis that uncerta-
inty-aware planning leads to more fluid and intelligent robot behavior in collaborative settings.

Future works will extend the current framework to more complex tasks, such as those of Industry 5.0 and Human–Robot Collaboration, with an extensive user study to validate both usability and impact on people.

CRediT authorship contribution statement

Lorenzo Busellato: Writing – original draft, Software, Methodology, Formal analysis. **Federico Cunico:** Writing – original draft, Software, Methodology, Data curation. **Diego Dall'Alba:** Writing – review & editing, Visualization, Investigation. **Marco Emporio:** Validation, Resources. **Andrea Giachetti:** Supervision, Resources. **Riccardo Muradore:** Writing – review & editing, Supervision, Conceptualization. **Marco Cristani:** Writing – review & editing, Supervision, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Lorenzo Busellato reports financial support was provided by iNEST. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This study was carried out within the MICS (Made in Italy – Circular and Sustainable) Extended Partnership and received funding from Next-Generation EU (Italian PNRR – M4 C2, Invest 1.3 – D.D. 1551.11-10-2022, PE00000004). CUP MICS, Italy D43C22003120001 - Cascade funding project CollaborICE.

This manuscript reflects only the Authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them.

References

[1] N. Robinson, B. Tidd, D. Campbell, D. Kulić, P. Corke, Robotic vision for human-robot interaction and collaboration: A survey and systematic review, ACM Trans. Human-Robot Interact. 12 (1) (2023) 1–66.
[2] J. Arents, V. Abolins, J. Judvaitis, O. Vismanis, A. Oraby, K. Ozols, Human-robot collaboration trends and safety aspects: A systematic review, J. Sens. Actuator Netw. 10 (3) (2021) 48.
[3] A. Sampieri, G.M.D. di Melendugno, A. Avogaro, F. Cunico, F. Setti, G. Skenderi, M. Cristani, F. Galasso, Pose forecasting in industrial human-robot collaboration, in: European Conference on Computer Vision, Springer, 2022, pp. 51–69.
[4] N.F. Duarte, M. Raković, J. Tasevski, M.I. Coco, A. Billard, J. Santos-Victor, Action anticipation: Reading the intentions of humans and robots, IEEE Robot. Autom. Lett. 3 (4) (2018) 4132–4139.
[5] M. Li, S. Chen, Y. Zhao, Y. Zhang, Y. Wang, Q. Tian, Dynamic multiscale graph neural networks for 3D skeleton based human motion prediction, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2020, pp. 211–220.
[6] S.-Y. Lo, B. Fernandez, P. Stone, A.L. Thomaz, Towards safe motion planning in human workspaces: A robust multi-agent approach, in: 2021 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2021, pp. 7929–7935.
[7] S. He, J. Zeng, B. Zhang, K. Sreenath, Rule-based safety-critical control design using control barrier functions with application to autonomous lane change, in: 2021 American Control Conference, ACC, IEEE, 2021, pp. 178–185.
[8] A. Palleschi, M. Hamad, S. Abdolshah, M. Garabini, S. Haddadin, L. Pallottino, Fast and safe trajectory planning: Solving the cobot performance/safety trade-off in human-robot shared environments, IEEE Robot. Autom. Lett. 6 (3) (2021) 5445–5452.
[9] C. Byner, B. Matthias, H. Ding, Dynamic speed and separation monitoring for collaborative robot applications-concepts and performance, Robot. Comput.-Integr. Manuf. 58 (2019) 239–252.
[10] J. Breeden, D. Panagou, Predictive control barrier functions for online safety critical control, in: 2022 IEEE 61st Conference on Decision and Control, CDC, IEEE, 2022, pp. 924–931.
[11] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[12] F. Vesentini, R. Muradore, P. Fiorini, A survey on velocity obstacle paradigm, Robot. Auton. Syst. 174 (2024) 104645, http://dx.doi.org/10.1016/j.robot.2024.104645, URL https://www.sciencedirect.com/science/article/pii/S0921889024000289.
[13] F. Trotti, E. Ghignoni, R. Muradore, A modified recursive Newton-Euler algorithm embedding a collision avoidance module, in: 2021 European Control Conference, ECC, 2021, pp. 1150–1155, http://dx.doi.org/10.23919/ECC54610.2021.9655037.
[14] A.D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, P. Tabuada, Control barrier functions: Theory and applications, in: 2019 18th European Control Conference, ECC, IEEE, 2019, pp. 3420–3431.

[15] M.H. Cohen, C. Belta, Approximate optimal control for safety-critical systems with control barrier functions, in: 2020 59th IEEE Conference on Decision and Control, CDC, IEEE, 2020, pp. 2062–2067.
[16] J. Breeden, D. Panagou, Robust control barrier functions under high relative degree and input constraints for satellite trajectories, Automatica 155 (2023) 111109.
[17] W. Xiao, C. Belta, High-order control barrier functions, IEEE Trans. Autom. Control 67 (7) (2021) 3655–3662.
[18] Q. Nguyen, K. Sreenath, Exponential control barrier functions for enforcing high relative-degree safety-critical constraints, in: 2016 American Control Conference, ACC, IEEE, 2016, pp. 322–328.
[19] A.A. Do Nascimento, A. Papachristodoulou, K. Margellos, Probabilistically safe controllers based on control barrier functions and scenario model predictive control, in: 2024 IEEE 63rd Conference on Decision and Control, CDC, IEEE, 2024, pp. 1814–1819.
[20] Y. Chen, M. Jankovic, M. Santillo, A.D. Ames, Backup control barrier functions: Formulation and comparative study, in: 2021 60th IEEE Conference on Decision and Control, CDC, IEEE, 2021, pp. 6835–6841.
[21] M. Davoodi, J.M. Cloud, A. Iqbal, W.J. Beksi, N.R. Gans, Safe human-robot coetaneousness through model predictive control barrier functions and motion distributions, IFAC-PapersOnLine 54 (20) (2021) 271–277, http://dx.doi.org/10.1016/j.ifacol.2021.11.186, URL https://www.sciencedirect.com/science/article/pii/S2405896321022308. Modeling, Estimation and Control Conference MECC 2021.
[22] K.P. Wabersich, M.N. Zeilinger, Predictive control barrier functions: Enhanced safety mechanisms for learning-based control, IEEE Trans. Autom. Control 68 (5) (2022) 2638–2651.
[23] J. Breeden, D. Panagou, Predictive control barrier functions for online safety critical control, in: 2022 IEEE 61st Conference on Decision and Control, CDC, IEEE, 2022, pp. 924–931.
[24] M. Black, M. Jankovic, A. Sharma, D. Panagou, Future-focused control barrier functions for autonomous vehicle control, in: 2023 American Control Conference, ACC, IEEE, 2023, pp. 3324–3331.
[25] V. Hamdipoor, N. Meskin, C.G. Cassandras, Safe control synthesis using environmentally robust control barrier functions, Eur. J. Control 74 (2023) 100840.
[26] Y.S. Quan, J. Zhou, E. Frisk, C.C. Chung, Observer-based environment robust control barrier functions for safety-critical control with dynamic obstacles, 2024, arXiv preprint arXiv:2403.13288.
[27] M.A. Khan, T. Ibuki, A. Chatterjee, Gaussian control barrier functions: Non-parametric paradigm to safety, IEEE Access 10 (2022) 99823–99836, http://dx.doi.org/10.1109/ACCESS.2022.3206372.
[28] J. Li, Q. Liu, W. Jin, J. Qin, S. Hirche, Robust safe learning and control in an unknown environment: An uncertainty-separated control barrier function approach, IEEE Robot. Autom. Lett. 8 (10) (2023) 6539–6546, http://dx.doi.org/10.1109/LRA.2023.3309130.
[29] B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, Robotics: Modelling, Planning and Control, first ed., Springer Publishing Company, Incorporated, 2008, p. 546.
[30] A. Singletary, K. Klingebiel, J. Bourne, A. Browning, P. Tokumaru, A. Ames, Comparative analysis of control barrier functions and artificial potential fields for obstacle avoidance, in: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2021, pp. 8129–8136.
[31] D. Fridovich-Keil, A. Bajcsy, J.F. Fisac, S.L. Herbert, S. Wang, A.D. Dragan, C.J. Tomlin, Confidence-aware motion prediction for real-time collision avoidance1, Int. J. Robot. Res. 39 (2–3) (2020) 250–265.
[32] A. Avogaro, A. Toaiari, F. Cunico, X. Xu, H. Dafas, A. Vinciarelli, E. Li, M. Cristani, Exploring 3D human pose estimation and forecasting from the robot's perspective: The HARPER dataset, in: 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2024, pp. 5828–5835.
[33] W. Guo, Y. Du, X. Shen, V. Lepetit, X. Alameda-Pineda, F. Moreno-Noguer, Back to mlp: A simple baseline for human motion prediction, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2023, pp. 4809–4819.
[34] N. Mahmood, N. Ghorbani, N.F. Troje, G. Pons-Moll, M.J. Black, AMASS: Archive of motion capture as surface shapes, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 5442–5451.
[35] C. Ionescu, D. Papava, V. Olaru, C. Sminchisescu, Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments, IEEE Trans. Pattern Anal. Mach. Intell. 36 (7) (2013) 1325–1339.
[36] H. Yan, Q. Cui, J. Xie, S. Guo, Forecasting of 3d whole-body human poses with grasping objects, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 1726–1736.
[37] N.M. Gamage, D. Ishtaweera, M. Weigel, A. Withana, So predictable! continuous 3D hand trajectory prediction in virtual reality, in: The 34th Annual ACM Symposium on User Interface Software and Technology, UIST '21, Association for Computing Machinery, New York, NY, USA, 2021, pp. 332–343, http://dx.doi.org/10.1145/3472749.3474753.
[38] W. Bao, L. Chen, L. Zeng, Z. Li, Y. Xu, J. Yuan, Y. Kong, Uncertainty-aware state space transformer for egocentric 3D hand trajectory forecasting, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, ICCV, 2023, pp. 13702–13711.
[39] F. Cunico, F. Girella, A. Avogaro, M. Emporio, A. Giachetti, M. Cristani, Oo-dmvmt: A deep multi-view multi-task classification framework for real-time 3d hand gesture classification and segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 2745–2754.
[40] C. Yu, X. Ma, J. Ren, H. Zhao, S. Yi, Spatio-temporal graph transformer networks for pedestrian trajectory prediction, in: European Conference on Computer Vision, Springer, 2020, pp. 507–523.
[41] F. Giuliari, I. Hasan, M. Cristani, F. Galasso, Transformer networks for trajectory forecasting, in: 2020 25th International Conference on Pattern Recognition, ICPR, IEEE, 2021, pp. 10335–10342.
[42] C. Xu, W. Mao, W. Zhang, S. Chen, Remember intentions: Retrospective-memory-based trajectory prediction, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 6488–6497.
[43] A. Mohamed, K. Qian, M. Elhoseiny, C. Claudel, Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 14424–14432.
[44] P. Corke, J. Haviland, Not your grandmother's toolbox-the robotics toolbox reinvented for Python, in: 2021 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2021, pp. 11357–11363.
[45] T.B. Tuli, M. Manns, S. Zeller, Human motion quality and accuracy measuring method for human-robot physical interactions, Intell. Serv. Robot. 15 (4) (2022) 503–512.
[46] C. Jost, B. Le Pévédic, T. Belpaeme, C. Bethel, D. Chrysostomou, N. Crook, M. Grandgeorge, N. Mirnig, Human-Robot Interaction, Springer, 2020.
[47] B. Lakshminarayanan, A. Pritzel, C. Blundell, Simple and scalable predictive uncertainty estimation using deep ensembles, Adv. Neural Inf. Process. Syst. 30 (2017).