Co-Design of Communication, Computing and Control for Real-Time Interactions in Industrial Cyber-Physical Systems

Kan Chen

Submitted in fulfilment of the requirements for the
Degree of Doctor of Philosophy

School of Computing Science
College of Science and Engineering
University of Glasgow

September 2025

Abstract

As intelligent systems evolve toward tighter integration of communication, computation, and control, Cyber-Physical Systems (CPS) and the Industrial Metaverse have emerged as the next frontier for real-time human–machine collaboration. These systems promise immersive, task-aware, and adaptive interaction between humans, robots, and digital twins. Yet, existing frameworks remain fragmented: communication networks, control algorithms, and human feedback mechanisms are often optimized in isolation, making it difficult to guarantee task-level performance and human trust under dynamic conditions. This thesis develops a unified, task-oriented, and human-centered co-design framework that bridges these domains to enable real-time, adaptive, and trustworthy CPS operation.

The research begins by establishing a task-oriented cross-system design framework that integrates communication scheduling, computation offloading, and control policy optimization into a single learning-based architecture. A Human-in-the-Loop Reinforcement Learning (HITL-RL) mechanism is developed to enable adaptive policy refinement through interactive human feedback. Experimental validation on a teleoperation platform demonstrates that the proposed HITL-RL achieves an average position RMSE of 0.023 m, outperforming the data-based RL baseline by 47.3%, and significantly enhancing control smoothness and trajectory stability under stochastic delay conditions. These results confirm that incorporating human corrective inputs effectively improves policy robustness and task execution accuracy in dynamic environments.

Building on this foundation, a Human-in-the-Loop Meta-Learning (HITL-MAML) framework is introduced to enhance adaptability across operators and task contexts. By leveraging Model-Agnostic Meta-Learning, the system learns a transferable initialization that can be efficiently adapted with limited human feedback. To ensure that the framework is generalizable and robust, we propose the HITL-MAML algorithm, dynamically adjusting prediction horizons. To verify the proposed framework and algorithm, we build a prototype including a real-world robotic arm and its digital model in the CPS. The results demonstrate that our approach reduces the weighted sum of the Root Mean Squared Error (RMSE) from 0.0712 m to 0.0101 m, significantly outperforming various baseline methods. This substantial improvement enhances both the responsiveness and reliability of real-time CPS interactions.

To further align system behavior with human experience, a preference-driven reinforcement learning approach is developed, incorporating Reinforcement Learning from Human Feedback

(RLHF) into the co-design loop. By modeling implicit human responses as latent reward signals, the framework learns control strategies that optimize both task efficiency and perceived comfort. We validate our framework using a UR3e robotic arm for reactor tile inspection in a nuclear decommissioning scenario. Compared to baseline methods, our approach enhances scene representation while optimizing trajectory efficiency. The RLHF-based policy consistently outperforms baseline selection, prioritizing task-critical details. By unifying explicit 3D scene representations with implicit human-in-the-loop optimization, this work establishes a foundation for adaptive, safety-critical robotic perception systems, paving the way for enhanced automation for remote maintenance and other high-risk environments.

Comprehensive simulations and hardware-in-the-loop experiments confirm that the proposed frameworks achieve consistent improvements in latency, reliability, adaptability, and human satisfaction. Collectively, these contributions establish a theoretical and experimental foundation for scalable, task-oriented, and human-adaptive CPS. Looking ahead, this work envisions the evolution of co-design methodologies toward multi-human collaboration—where multiple operators interact with distributed agents through shared intent inference—and the incorporation of foundation model-driven optimization, leveraging large pre-trained models for semantic reasoning and zero-shot adaptation. Together, these directions point toward the realization of human-centric CPS and Industrial Metaverse ecosystems.

To my supervisors, colleagues, friends, and family—for their guidance, inspiration, and love.

Declaration

I confirm that this thesis is my own original work and has not been submitted for any other degree or professional qualification. Except where clearly indicated, all work presented is the result of my own research, conducted in accordance with the University of Glasgow's Code of Good Practice in Research. All necessary permissions for the inclusion of copyrighted material have been obtained. The thesis has not been edited by a third party beyond what is permitted by the University's PGR Code of Practice. The copyright of this thesis rests with the author, and no quotation from it is permitted without full acknowledgement.

List of Publications

[1] **K. Chen**, Z. Meng, X. Xu, C. She and P. G. Zhao, "Real-Time Interactions Between Human Controllers and Remote Devices in Metaverse," in *Proceedings of IEEE Conference on Metrology for eXtended Reality, Artificial Intelligence and Neural Engineering*, 2024, pp. 353–358, doi: 10.1109/MetroXRAINE62247.2024.10795969.

[2] **K. Chen**, Z. Meng, X. Xu, J. Yang, E. L. Li and P. G. Zhao, "Task-Oriented Edge-Assisted Cross-System Design for Real-Time Human-Robot Interaction in Industrial Metaverse," (2025), submitted to *IEEE Transactions on Mobile Computing*, Under Review (Major Revision).

[3] Z. Meng, **K. Chen**, Y. Diao, C. She, P. G. Zhao, M. A. Imran and B. Vucetic, "Task-Oriented Cross-System Design for Timely and Accurate Modeling in the Metaverse," (2024), in *IEEE Journal on Selected Areas in Communications*, vol. 42, no. 3, pp. 752-766, March 2024, doi: 10.1109/JSAC.2023.3345398.

[4] Z. Meng, **K. Chen**, X. Xu, J. Yang, E. L. Li and P. G. Zhao, "Preference-Driven Active 3D Scene Representation for Robotic Inspection in Nuclear Decommissioning," (2025), submitted to *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, Under Review.

Contents

List of Figures

List of Tables

List of Algorithms

List of Acronyms

3D	Three-Dimensional.
AI	Artificial Intelligence.
AoI	Age of Information.
AoL	Age of Loop.
API	Application Programming Interface.
AR	Augmented Reality.
ARMA	Auto-Regressive Moving Average.
CPS	Cyber-Physical System.
CSV	Comma Separated Values.
DoF	Degrees of Freedom.
DRL	Deep Reinforcement Learning.
DT	Digital twin.
E2E	End-to-End.
EDF	Earliest Deadline First.
FM	Foundation Model.
FTP	File Transfer Protocol.
HCI	Human-Computer Interaction.
HITL	Human-In-The-Loop.
HITL-MAML	Human-In-The-Loop Model-Agnostic Meta-Learning.
HRI	Human-Robot Interaction.
IIoT	Industrial Internet of Things.

KPI	Key Performance Indicator.
LPIPS	Learned Perceptual Image Patch Similarity.
MAML	Model-Agnostic Meta-Learning.
MEC	Mobile Edge Computing.
MLP	Multi-Layer Perception.
MPC	Model-Predictive Control.
MTP	Motion-To-Photon.
NBV	Next Best View.
NCS	Networked Control Systems.
NeRF	Neural Radiance Fields.
PID	Proportional Integral Derivative.
PPO	Proximal Policy Optimization.
PSNR	Peak Signal-to-Noise Ratio.
QoE	quality of experience.
QoS	Quality of Service.
RLHF	Reinforcement Learning from Human Feedback.
RMP	Riemannian Motion Policy.
RMS	Rate-Monotonic Scheduling.
RMSE	Root Mean Squared Error.
ROS	Robot Operating System.
RPC	Remote Procedure Call.
RTE	Real-Time Ethernet.
SGD	Stochastic Gradient Descent.
SSIM	Structural Similarity Index.
TC	Traffic Control.
TCP	Transmission Control Protocol.
UAV	Unmanned Aerial Vehicle.
UDP	User Datagram Protocol.
UGV	Unmanned Ground Vehicle.

URLLC	Ultra-Reliable Low-Latency Communication.
VR	Virtual Reality.
WNCS	Wireless Networked Control System.
XR	Extended Reality.

Chapter 1

Introduction

The evolution of Cyber-Physical System (CPS), empowered by the convergence of communication, computing, and control technologies, is transforming the paradigm of real-time human–machine interaction. With the emergence of Digital twins (DTs) and the Industrial Metaverse, physical assets and virtual representations are becoming tightly integrated, enabling continuous synchronization, predictive analytics, and immersive teleoperation [1]. However, as these systems operate in highly dynamic and heterogeneous environments, their performance is no longer dictated by isolated subsystem characteristics but by the cross-domain coupling among communication networks, computational resources, and control loops [2, 3]. In such tightly interconnected architectures, latency, reliability, and stability emerge as joint properties influenced by End-to-End (E2E) interactions rather than individual components. Consequently, traditional subsystem-level optimization becomes inadequate for ensuring real-time responsiveness and Human-In-The-Loop (HITL) adaptability [4]. To address these limitations, a co-design framework that jointly considers communication, computing, and control is imperative to achieve resilient, efficient, and adaptive performance in CPS.

CPS serve as the foundational framework for linking physical processes with digital intelligence. On this basis, the co-design of communication, computing, and control constitutes the integrative methodology through which these systems can operate efficiently across layers. The Real-time interaction represents the operational outcome of this integration, characterizing how the co-designed system functions under stringent timing constraints and HITL requirements. This conceptual linkage not only underpins the structure of this thesis but also guides the transition from system-level design principles to task-oriented, real-time operations in industrial environments.

To lay the foundation for this thesis, three interrelated aspects are introduced. First, the concepts of CPS, DT, and the Metaverse are presented to clarify the system-level landscape and highlight their potential in bridging physical and virtual spaces. Second, the notion of real-time interactions is examined, emphasizing the stringent requirements for seamless communication, haptic feedback, and control across distributed systems. Third, the principle of co-design of

communication, computing, and control is discussed as a unifying methodology to jointly optimize cross-system performance and ensure responsiveness in human-robot collaboration.

By integrating these perspectives, this thesis aims to establish a coherent framework for addressing the unique challenges of real-time interactions in CPS and the Metaverse. The subsequent sections of this chapter elaborate on each of these pillars in detail, before motivating the research questions, outlining key challenges, and summarizing the main contributions.

1.1 Cyber-Physical Systems, Digital Twins and Industrial Metaverse

The development of intelligent interconnected systems can be viewed as a continuum encompassing CPS, DT, and the Industrial Metaverse. Together, these paradigms progressively integrate physical processes, computational intelligence, and human interaction into a unified cyber–physical framework.

1) Cyber-Physical Systems: CPS represent the deep integration of computation, communication, and control with the physical world. Originating from the evolution of embedded and networked control systems, CPS provide the architectural foundation for real-time interaction between distributed sensing, computation, and actuation [5]. In industrial contexts, CPS enable large-scale coordination among heterogeneous devices—such as robots, sensors, edge servers, and controllers—operating under stringent timing and reliability constraints [6]. The complexity of these systems arises from tightly coupled feedback loops across communication networks, computational resources, and control mechanisms, where performance metrics such as latency, stability, and reliability emerge as interdependent system-level properties rather than isolated subsystem parameters [7]. These characteristics underscore the need for cross-domain optimization to ensure predictable and stable operation in dynamic industrial environments [8].

2) Digital Twin: DT is a living digital representation of a specific physical asset, process, or system that is continuously synchronized with its physical counterpart throughout its lifecycle via measured data, models, and contextual knowledge [9]. Unlike static digital models or one-way "digital shadows," a DT maintains a bidirectional link that supports online monitoring and diagnosis, state estimation and parameter identification, what-if simulation and prediction, as well as control and optimization. In CPS, DTs are designed with task-appropriate fidelity (rather than maximal fidelity) to meet timing and reliability constraints, thereby enabling proactive, adaptive decision-making and turning CPS from reactive loops into data- and model-driven closed-loop ecosystems [10].

3) Industrial Metaverse: The term Metaverse, first coined by Neal Stephenson in his 1992 novel Snow Crash, has evolved from a fictional concept of a shared virtual space into a technological paradigm encompassing interconnected, immersive, and persistent digital environments [11]. In recent years, this concept has been extended to industrial domains, giving rise

to the notion of the Industrial Metaverse. Building upon CPS and DT, the Industrial Metaverse envisions a cyber–physical environment where humans, machines, and digital agents interact seamlessly within immersive and synchronized virtual spaces. By fusing real-time control from CPS with the analytical and representational power of DTs, it enables teleoperation, collaborative design, and remote supervision in safety-critical domains [12]. Unlike consumer-oriented Metaverse platforms, the Industrial Metaverse prioritizes task-oriented performance metrics, emphasizing ultra-low latency, deterministic reliability, and HITL adaptability to guarantee both safety and efficiency during real-time operations [13].

The convergence of CPS, DT, and the Industrial Metaverse reveals a fundamental challenge: as physical and digital systems become increasingly integrated, their collective performance depends on end-to-end coordination across communication, computing, and control domains. Latency variations, computational delays, or control instabilities can no longer be analyzed in isolation, as they propagate and amplify across the entire system [14]. Moreover, the increasing involvement of humans in the control loop introduces additional complexity—real-time responsiveness must now account for perceptual thresholds, cognitive workload, and interactive feedback that directly affect task performance and safety [15]. To achieve resilient and human-centric real-time interactions, a co-design approach—jointly optimizing communication, computing, and control while aligning with human perceptual and operational constraints—is essential for next-generation industrial cyber–physical environments.

1.2 Real-Time Interactions

While CPS, DT, and the industrial Metaverse encompass a broad range of applications, not all of them are inherently mission-critical. In this thesis, we specifically focus on mission-critical scenarios, where stringent requirements on latency, reliability, and real-time responsiveness are essential for safe and effective operation, particularly in human-in-the-loop (HITL) and remote robotic systems.

Building upon the discussion of CPS, DT, and the industrial Metaverse, it becomes evident that real-time interactions are the cornerstone for enabling safe and efficient operation in such mission-critical environments. While CPS emphasize the integration of computation, communication, and control, it is the timeliness of these processes that ultimately determines whether the system can function reliably under stringent industrial constraints.

Historically, early feedback control systems in the 1970s–1980s operated under relatively loose timing requirements, where delays on the order of milliseconds to seconds were acceptable. With the rise of Networked Control Systems (NCS) in the 1990s, communication latency and jitter emerged as key factors limiting stability and performance [16]. More recently, the vision of the Tactile Internet and Ultra-Reliable Low-Latency Communication (URLLC) has shifted expectations to the sub-millisecond regime, highlighting the demand for near-instantaneous

feedback loops in human–machine interaction [17].

Real-time interactions are indispensable across a wide range of industrial applications. In manufacturing, collaborative robots working alongside humans require E2E latency below 10 ms to ensure safe physical interaction without unintended collisions [18]. In power systems, delays in adaptive load balancing can lead to cascading failures and large-scale blackouts. In healthcare, studies on telesurgery have shown that control delays exceeding 100 ms significantly degrade surgical precision and increase the risk of adverse outcomes [19]. Similarly, in hazardous domains such as nuclear decommissioning or offshore oil drilling, operators depend on real-time haptic and visual feedback to safely manipulate remote robots in unpredictable environments [20].

These cases illustrate that real-time interactions are not merely a performance enhancement but a fundamental requirement in mission-critical deployments of CPS and the Industrial Metaverse. They determine whether human operators can trust and effectively collaborate with remote systems. As such, this thesis places a central focus on understanding, modeling, and improving real-time interactions by jointly addressing the communication, computing, and control processes that underpin them.

1.3 Co-Design of Communication, Computing and Control

As highlighted in the previous section, achieving reliable real-time interactions in Industrial CPS cannot be easily guaranteed by optimizing communication, computing, or control subsystems in isolation. Traditionally, these components have been designed separately: communication protocols have targeted metrics such as throughput and bit error rate, computing platforms have focused on maximizing resource utilization, and control systems have aimed at stability and robustness [21]. While such separate design approaches are effective within each domain, they often lead to sub-optimal performance at the system level. For example, reducing packet error rates in communication does not necessarily improve end-to-end control accuracy if computational delays remain unaccounted for, and a highly stable controller cannot prevent instability if network-induced delays are not considereds [22].

To address these limitations, co-design has emerged as a unifying paradigm that jointly optimizes communication, computing, and control. The fundamental principle of co-design is to align subsystem objectives with task-oriented performance metrics rather than optimizing each subsystem independently. By coordinating resource allocation and algorithmic strategies across domains, co-design enables the system to meet stringent industrial requirements such as ultra-low latency, high reliability, and safe human-in-the-loop interactions [4]. For instance, in robotic teleoperation, co-design strategies can dynamically adjust prediction horizons and communication scheduling to minimize end-to-end tracking errors. In immersive industrial Metaverse applications, integrating rendering prediction with network scheduling reduces motion-to-photon

latency beyond what separate optimizations can achieve [23]. Similarly, in smart grids, joint optimization of communication scheduling and control strategies has been shown to reduce the Age of Information (AoI), improving stability and resilience [24].

Within the context of Industrial CPS, co-design has been widely recognized as an essential methodology for bridging the gap between theoretical performance metrics and practical task requirements. It shifts the focus from domain-specific optimization to holistic, cross-system design, where communication, computing, and control are considered as interdependent components of a unified system. This thesis builds upon this perspective, developing novel co-design frameworks that incorporate human preferences, predictive modeling, and reinforcement learning to ensure robust real-time interactions in complex industrial environments.

1.4 Motivation

1.4.1 Multi-Loop Latency and Uncertainty

Industrial human–machine systems rarely operate as a single, isolated feedback loop. Instead, operator actions simultaneously propagate through multiple concurrent and interacting pathways, as illustrated in Fig. 1.1. A typical system consists of at least two tightly coupled yet asynchronous feedback processes: a control loop, which governs motion commands and state feedback between the operator and the physical workspace and is well established in networked control systems [16], and a rendering loop, which provides visual and haptic feedback through the virtual environment or digital twin. Sensors capture the operator's motion, networked systems transmit information across physical and virtual domains, and both real and simulated environments respond in their own temporal scales. Because these loops differ in their sensing, computation, and communication delays, the information reaching the operator is often temporally misaligned, leading to perceptual and control inconsistencies.

Although the introduction of virtual domains may increase system complexity and introduce additional latency due to rendering and synchronization processes, they provide critical benefits that are not achievable through purely physical feedback loops. In particular, virtual domains enable predictive visualization, state estimation, and environment augmentation, which enhance the operator's situational awareness and decision-making capability. In human-in-the-loop scenarios, operators often rely on rendered visual and haptic feedback rather than direct physical observation, especially in remote or hazardous environments. Therefore, the rendering process is not merely an additional delay source but an essential component for delivering interpretable and actionable feedback.

This multi-loop architecture introduces inherent challenges in ensuring synchronized and stable feedback. In the control loop, delay and jitter directly affect actuation precision and system stability. In the rendering loop, latency in visual or haptic feedback alters the operator's

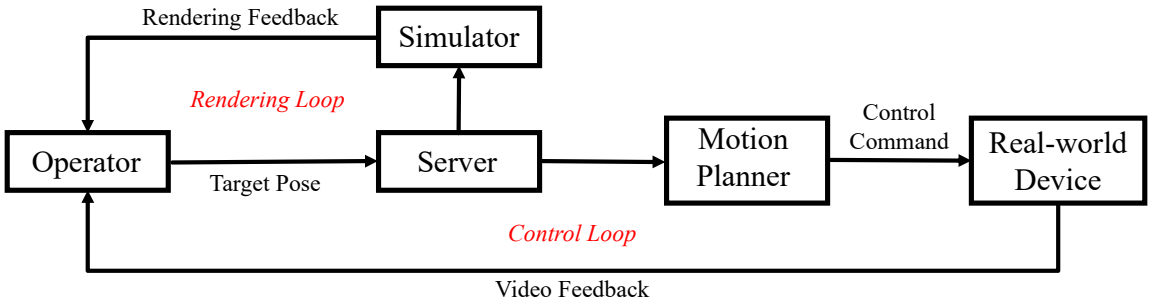

Figure 1.1: Structural diagram of multi-loop interaction in Industrial Human–Machine Systems. The figure illustrates how the operator, server, and real-world workspace are interconnected through the control and rendering loops. Information flows bidirectionally between the physical and virtual spaces, forming coupled feedback loops where asynchronous feedback and heterogeneous delay sources jointly influence system behavior.

perception of timing and spatial consistency. When these feedback channels operate at different temporal scales, the operator may experience perceptual dissonance—actions and responses appear inconsistent—causing overcompensation or hesitation during task execution. In industrial contexts, such as teleoperation, assembly, or remote inspection, this desynchronization can lead to degraded task accuracy, reduced operator confidence, or even safety risks [14].

The problem is further compounded by the stochastic and heterogeneous nature of latency. Network delay varies with congestion, computation delay depends on workload and algorithmic complexity, and the operator's reaction time introduces human variability into the loop. These uncertainties interact nonlinearly: even small timing mismatches between control and rendering feedback can amplify through the coupled loops, resulting in oscillatory behavior or unstable control [25]. Thus, overall performance is determined not only by the mean delay of each loop but also by their temporal alignment and cross-loop dynamics [26].

These observations highlight the central motivation of this research. Reliable and intuitive human–machine interaction requires not merely minimizing latency in each subsystem but understanding and managing the interplay between multiple asynchronous feedback loops. Effective system design must therefore account for heterogeneous feedback timing, multi-source uncertainty, and cross-domain synchronization to ensure coherent and stable perception–action coupling. Addressing these challenges forms the basis for the task-oriented modeling and co-design frameworks developed in the subsequent chapters of this thesis.

1.4.2 Complexity of Human-in-the-Loop Systems

The discussion of multi-loop latency and uncertainty has primarily focused on technological subsystems such as communication, computation, and control. However, in Industrial CPS and

the Industrial Metaverse an equally critical source of complexity arises from the human operator who remains in the feedback loop. HITL systems differ fundamentally from purely automated control in that the operator's perception, cognition, and action responses become integral components of the overall system dynamics. As a result, the performance, safety, and reliability of HITL systems depend not only on engineering design but also on the variability and adaptability of human behavior.

Historically, HITL concepts emerged in aviation and military applications, where pilots or operators interacted with complex systems under high-stakes conditions [17]. The central motivation was that humans provide situational awareness, adaptability, and decision-making that purely automated systems cannot match. With the rise of modern robotics, remote operation, and immersive interfaces, HITL has expanded to domains such as telesurgery, collaborative industrial robotics, and hazardous-environment teleoperation [18]. In these settings, human cognition and decision-making are indispensable, but they also introduce challenges absent in fully autonomous systems.

The complexity of HITL systems arises from multiple intertwined factors. First, human sensorimotor response times impose fundamental delays (typically on the order of hundreds of milliseconds), which interact with system-induced delays and can magnify control errors [27]. Second, human behavior is inherently non-stationary: operators learn and adapt during tasks, but they also exhibit fatigue, lapses of attention, or inconsistent strategies across sessions [28]. Third, significant variability exists across individuals, meaning that a system tuned to one operator may perform poorly for another [4]. Fourth, cognitive bandwidth is limited: when operators are overloaded with information (visual, haptic, auditory), their ability to respond accurately degrades, particularly under stressful or safety-critical conditions. These factors make modeling the "human component" of the loop considerably more difficult than modeling communication channels or control laws.

Concrete industrial examples illustrate these challenges. In telesurgery, surgeons rely on visual and haptic cues to guide sub-millimeter incisions, yet their performance can deteriorate rapidly under latency or when feedback is inconsistent with expectation, leading to increased error rates. In collaborative robotics, human workers sharing a workspace with cobots require the system to anticipate human trajectories and intentions in real time [29]; a failure to do so risks collisions or production interruptions. In hazardous remote operations such as nuclear decommissioning, human operators often face high cognitive loads while controlling robotic manipulators through indirect and delayed feedback, where mistakes can have catastrophic consequences. These cases highlight that human variability, perception–action delay, and bounded rationality are not peripheral issues but central determinants of HITL system performance [27].

From a research perspective, the presence of the human operator introduces both challenges and opportunities. On the one hand, HITL systems are difficult to analyze using classical control theory because the "plant" includes a human whose dynamics are stochastic, time-varying, and

partly unobservable [30]. On the other hand, humans provide adaptability and contextual understanding that automated systems still struggle to achieve. This duality motivates approaches that incorporate learning and adaptation directly into the control framework. In particular, reinforcement learning with human feedback, predictive modeling of human motion and intention, and task-oriented co-design that explicitly accounts for human variability are promising directions [31]. The central thesis of this work is that by treating the human not as a disturbance but as an integral and co-adaptive component of the loop, Industrial CPS and Industrial Metaverse systems can achieve safer, more robust, and more effective real-time interactions.

1.4.3 Human Preferences and Experience Modeling

Beyond the technical aspects of latency, uncertainty, and control stability, the effectiveness of HITL systems ultimately depends on how humans perceive and experience their interactions with the system. A control loop may be objectively stable and low in latency, yet if the feedback feels unnatural, the workload is excessive, or the sensory cues are misaligned with operator expectations, overall performance and safety can still degrade [4]. This recognition motivates explicit modeling of human preferences and experience as a central component of Industrial CPS and Industrial Metaverse design.

Research in Human-Computer Interaction (HCI), ergonomics, and quality of experience (QoE) consistently demonstrates that subjective human perception often diverges from objective system metrics. For instance, in video-streaming studies, user satisfaction correlates more strongly with playback interruptions than with average throughput [32]. Similarly, in haptic teleoperation, operators tend to prefer small, consistent delays over unstable or jittery feedback, even when average latency is equivalent [33]. These findings underline a key principle: human comfort, trust, and task efficiency cannot be inferred from engineering metrics alone. Instead, they require models that reflect how humans subjectively trade off delay, stability, precision, and cognitive effort.

Modeling human preferences, however, is inherently complex. Preferences are heterogeneous: different operators display varying tolerance to latency, noise, and workload, shaped by training, skill level, and even cultural or cognitive factors. They are also dynamic: tolerance evolves as users adapt, learn, or become fatigued. Moreover, perception is nonlinear: humans exhibit threshold effects—minor fluctuations below a perceptual boundary may go unnoticed, while exceeding it abruptly degrades perceived quality [34]. Finally, preferences are context-dependent: what an operator considers acceptable in a training environment may be intolerable in safety-critical or high-stress tasks such as tele-surgery [35]. These dimensions collectively make human preference modeling both scientifically challenging and practically essential.

Diverse industrial applications vividly illustrate these issues. In immersive industrial training using Virtual Reality (VR)/Augmented Reality (AR), high Motion-To-Photon (MTP) latency not only reduces task accuracy but can induce simulator sickness and lower long-term user engage-

ment [36]. In remote maintenance and inspection, operators frequently prioritize visual stability and situational awareness over communication speed, preferring slightly delayed but consistent feedback to avoid perceptual dissonance [33]. In collaborative robotics, predictability and transparency of robot motion are valued more than maximum efficiency, as they reduce cognitive workload and foster human trust [37]. In industrial control rooms or human–AI supervisory settings, operators often favor reliable, interpretable feedback and consistent alert timing over aggressive automation policies that might undermine confidence [35]. Together, these examples demonstrate that aligning system design with human preferences directly enhances safety, usability, and mission effectiveness.

Integrating human preferences into Industrial CPS design offers several tangible benefits. First, it enables task-oriented co-design that optimizes for human-perceived quality rather than purely technical performance indicators. Second, it facilitates personalization and adaptability: by capturing individual operator profiles, systems can dynamically tune prediction horizons, control gains, or communication scheduling to align with human comfort and cognitive limits. Third, it strengthens trust and acceptance—a crucial prerequisite for reliable HITL operation—by ensuring that system responses feel natural, transparent, and supportive of the operator's intent.

This thesis adopts a human-centered perspective: Industrial CPS and Industrial Metaverse systems must not only meet engineering requirements but also resonate with human expectations and perceptual realities. Accordingly, the subsequent chapters develop preference-aware and experience-driven models that integrate seamlessly into the broader task-oriented co-design framework, ensuring that system intelligence complements, rather than conflicts with, human experience.

1.5 Key Challenges

The preceding discussions highlight that enabling real-time, human-centered interaction in CPS and the Industrial Metaverse requires the integration of communication, computing, control, and human factors into a unified adaptive framework. Despite considerable progress in networked control, edge intelligence, and human–computer interaction, several key challenges remain open. These challenges form the technical foundation and motivation for the remainder of this thesis.

- **Multi-Loop Latency and Uncertainty.**

  Real-world CPS involve multiple concurrent feedback loops—such as control, rendering, and prediction—that operate under stochastic latency and interdependent timing constraints. The overall end-to-end delay is not simply additive but a nonlinear function of network conditions, computation load, and human response dynamics. Conventional control or communication design approaches cannot effectively capture these coupled effects,

leading to performance degradation and instability. Addressing this challenge requires task-oriented predictive modeling and adaptive delay-compensation strategies capable of mitigating heterogeneous delay sources in real time.

- **Cross-System Coupling and Co-Design.**

  Traditional separate optimization of communication, computing, and control subsystems leads to sub-optimal system-level performance under real-time constraints. Communication scheduling influences control stability; computation latency affects feedback timeliness; and all interact to determine the responsiveness of human–robot collaboration. The absence of a unified, task-oriented optimization objective makes it difficult to coordinate these domains coherently. Therefore, a co-design paradigm that jointly allocates resources and adapts policies according to task-level performance metrics (e.g., motion-to-photon latency, RMSE, Age-of-Loop) is necessary.

- **Human Variability, Adaptation, and Preference Modeling.**

  Even with optimized system parameters, overall performance ultimately depends on how human operators perceive and interact with the system. Humans introduce variability through differing reaction times, learning behaviors, fatigue, and cognitive limitations. Preferences and comfort thresholds are highly individual and context-dependent, making fixed control or feedback strategies inadequate. Modeling human intent, comfort, and trust in a computationally tractable yet personalized manner remains a major open problem. Adaptive and preference-aware mechanisms are required to enable continuous co-adaptation between humans and machines.

- **Toward Task-Oriented, Human-Centered CPS.**

  The challenges above are inherently interconnected: multi-loop delays propagate through coupled subsystems; co-design seeks to address these interdependencies; and human variability defines the ultimate limit of usability and safety. The overarching challenge is to design CPS that are not only technically optimized but also task-oriented and human-centered—systems capable of adapting dynamically to environmental conditions, resource constraints, and human feedback in real time. This holistic perspective underpins the integrated methodologies and experimental frameworks developed throughout this work.

1.6 Contributions

To address the aforementioned challenges, this thesis proposes a series of frameworks and methodologies that integrate predictive modeling, cross-system co-design, and human-in-the-loop learning for real-time, human-centered CPS and Industrial Metaverse applications. The main contributions of this thesis are summarized as follows:

- **Predictive Modeling and Real-Time Delay Compensation.**

  A task-oriented modeling framework is developed to analyze and mitigate the effects of multi-loop latency and uncertainty in real-time human–machine interactions. The proposed approach decomposes total system delay into communication, computation, prediction, and execution components, and introduces adaptive prediction mechanisms to compensate for E2E latency. By integrating Auto-Regressive Moving Average (ARMA)-based prediction and reinforcement learning, the framework dynamically adjusts prediction horizons according to observed network and operator states. Experimental validation demonstrates that this approach significantly reduces motion errors and enhances stability under variable delay conditions.

- **Task-Oriented Co-Design of Communication, Computing, and Control.**

  This thesis introduces a unified co-design framework that jointly optimizes communication scheduling, computation allocation, and control strategies with respect to task-level performance metrics. Instead of minimizing delay or maximizing throughput in isolation, the proposed framework formulates a Deep Reinforcement Learning (DRL)-based optimization that directly minimizes task-specific cost functions such as MTP, Age of Loop (AoL), and control error. The co-design framework incorporates uncertainty modeling and adaptive resource scheduling to achieve balanced performance between latency, reliability, and task accuracy. Both simulation and prototype experiments show substantial gains in responsiveness and convergence efficiency compared with separate design baselines.

- **Human-in-the-Loop Reinforcement Learning and Preference-Aware Adaptation.**

  To address the variability and complexity of human operators, this thesis develops learning-based methods that explicitly incorporate human feedback and preferences into the control and optimization process. A HITL DRL framework is proposed, combining Proximal Policy Optimization (PPO) with real-time human corrective inputs to enable co-adaptive behavior between the operator and the system. Additionally, a preference-aware adaptation mechanism models individual operator comfort and trust thresholds, allowing the system to personalize prediction and control parameters dynamically. Empirical results indicate that incorporating human feedback not only accelerates policy convergence but also improves perceived control stability and operator trust.

- **Integrated Experimental Platform and Validation.**

  A real-time experimental testbed is implemented to validate the proposed methods under realistic industrial scenarios involving remote robotic control and immersive feedback. The platform integrates sensing, communication, and control loops, enabling quantitative evaluation of MTP latency, control Root Mean Squared Error (RMSE), and subjective

experience measures. The system-level experiments confirm that the proposed predictive, co-design, and HITL methods can jointly deliver stable, low-latency, and human-centered operation even under stochastic delay and uncertain human behaviors.

- **Comprehensive Framework for Task-Oriented, Human-Centered CPS.**

  By combining predictive compensation, cross-layer co-design, and human preference modeling, this thesis establishes a unified framework for designing next-generation CPS and Industrial Metaverse systems. The framework bridges the gap between traditional engineering optimization and human-centered performance, demonstrating that task-oriented, adaptive co-design can achieve both technical efficiency and user trust in mission-critical industrial environments.

In conclusion, these contributions establish a comprehensive and systematic foundation for building next-generation CPS and Industrial Metaverse. The work advances beyond conventional boundaries between communication, computing, and control by introducing a task-oriented co-design paradigm that explicitly incorporates human perception, adaptation, and feedback into the system optimization process. Through predictive modeling, adaptive co-design, and human-in-the-loop learning, the thesis demonstrates how technical efficiency and human-centered performance can be jointly achieved in real-world, delay-prone industrial environments. Beyond specific algorithms or implementations, the overarching contribution lies in articulating a unified perspective—one that views real-time human–robot collaboration as an integrated HITL CPS. This perspective provides a conceptual and methodological foundation for future research on safe, adaptive, and trust-aware intelligent systems operating at the intersection of humans, robots, and the industrial metaverse.

1.7 Thesis Organization

The remainder of this thesis is organized as follows. Each chapter builds upon the previous one to address the key challenges identified earlier, progressing from system-level analysis to cross-layer optimization and finally to human-centered adaptation.

- **Chapter 2 – Literature Review.**

  This chapter reviews the fundamental concepts underlying CPS, Digital Twins, and the Industrial Metaverse. It surveys prior research on real-time control, co-design methodologies, and human-in-the-loop systems, identifying open problems and gaps that motivate the subsequent chapters.

- **Chapter 3 – User-Centric Reinforcement Learning in Co-Design CPS Framework.**

  This chapter investigates the problem of multi-loop latency and uncertainty in real-time human–machine interactions. It presents a task-oriented modeling framework and an adaptive

prediction mechanism to compensate for heterogeneous delays across communication, computation, and control loops. Experimental validation demonstrates the framework's ability to reduce motion error and improve real-time stability.

- **Chapter 4 – Meta-Learning in Co-Design CPS Framework for Multi-Task Scenarios.**

  Building on the previous analysis, this chapter develops a unified co-design framework that jointly optimizes communication scheduling, computation allocation, and control strategies under industrial constraints. A reinforcement-learning-based optimization is employed to achieve end-to-end task performance improvements while maintaining robustness against uncertainty.

- **Chapter 5 – Preference-Driven Reinforcement Learning in Co-Design CPS via Human Feedback.**

  This chapter addresses the variability of human operators by introducing learning-based methods that explicitly incorporate human feedback and preferences into the control loop. It presents a human-in-the-loop reinforcement learning (HITL-RL) approach for adaptive policy refinement and a preference-aware adaptation mechanism that personalizes interaction based on operator behavior and subjective comfort.

- **Chapter 6 – Conclusions and Future Directions.**

  The final chapter summarizes the main findings and contributions of the thesis, discusses their broader implications for Industrial CPS and the Industrial Metaverse, and outlines potential directions for future research in task-oriented and human-centered system design.

Chapter 2

Literature Review

2.1 Co-Design of Communication, Computing, and Control

The co-design of communication, computing, and control has become a cornerstone methodology for achieving real-time performance, reliability, and adaptability in CPS. In traditional architectures, these subsystems were designed and optimized independently—communication focused on throughput and reliability, computing on resource efficiency, and control on stability and robustness. However, as modern industrial systems evolve into tightly coupled, distributed, and human-in-the-loop environments, the boundaries between these domains blur. The end-to-end performance of an CPS—measured by latency, stability, and responsiveness—depends on the dynamic interaction among communication, computation, and control processes rather than the performance of any individual component.

This realization has motivated a paradigm shift from separate subsystem optimization to cross-system co-design, where the interdependencies among communication, computation, and control components are explicitly modeled and jointly optimized to achieve overall system objectives. Building on this foundation, researchers have proposed cross-system optimization frameworks that integrate resource allocation, feedback control, and computational scheduling to ensure real-time and deterministic system performance. At a higher level of abstraction, the task-oriented communication paradigm further redefines design objectives from maximizing conventional transmission metrics to ensuring mission success, emphasizing the semantic and control relevance of transmitted information rather than raw data throughput. Moreover, prediction and scheduling mechanisms have been incorporated into co-design architectures to enable proactive and adaptive decision-making, ensuring system stability and performance robustness under dynamic and uncertain operating conditions.

In conclusion, these research directions outline a coherent evolution—from structural integration to adaptive intelligence—within the co-design paradigm. By jointly orchestrating communication, computing, and control under unified objectives, CPS can transcend traditional trade-offs between latency, reliability, and scalability, achieving efficient, resilient, and human-

centered operation. The following subsections review these developments in detail, tracing the theoretical foundations, representative methodologies, and emerging challenges across modular, cross-layer, task-oriented, and predictive co-design approaches.

2.1.1 Separate Design for Communication, Control and Computing Module

Table 2.1: Key KPIs for CPS-Related Service Categories in 3GPP TS 22.261 (Release 20)

Category	**KPI**	**Target Value**	**User Experience Objective**
XR (AR/VR/MR)	Throughput (DL/UL)	DL $\geq$ 1 Gbps, UL $\geq$ 200 Mbps	Support for 8K/16K video streaming and real-time rendering
	Latency	$\leq$ 5 ms (ideal 1 ms)	Low motion sickness, real-time interaction
	Jitter	$\leq$ 1 ms	Smooth audio/video experience
	Multi-user synchronization	$\leq$ 10 ms	Consistency in collaborative XR environments
Holographic Communication	Throughput	DL $\geq$ 5–10 Gbps	Support for high-fidelity 3D holographic video
	Latency	$\leq$ 1 ms	Real-time holographic interaction
	Multi-modal synchronization	$\leq$ 20 ms	Alignment of audio and visual streams
DT	Throughput	$\geq$ 100 Mbps (interactive)	Support for industrial/city-scale twin systems
	Latency	$\leq$ 1–10 ms	High fidelity between digital and physical entities
	Positioning accuracy	Horizontal $\leq$ 0.2–0.5 m, Vertical $\leq$ 1 m	Precise spatial modeling for digital twin
	Time synchronization	Nanosecond (ns) level	CPS, energy grid, and financial applications
Multimodal Immersive Collaboration	Throughput	$\geq$ 1 Gbps	Simultaneous transmission of audio, video, and haptic data
	Cross-modal synchronization	$\leq$ 20–30 ms	Alignment of audio/visual/haptic perception
	Interaction loop latency	$\leq$ 1 ms	Tactile Internet, immediate response
	Reliability	$\geq$ 99.999%	Ensuring safe and reliable remote collaboration

Traditional industrial systems were often designed through separate design, where communication, computing, and control subsystems were developed independently according to their respective objectives. Communication research primarily focused on enhancing throughput, spectral efficiency, and link reliability [22,23,38]; control research targeted system stability and robustness under network-induced delay and noise [39,40]; and computing research emphasized resource efficiency and task scheduling across devices and cloud infrastructure [10,41]. While each module achieved remarkable progress in its own domain, such isolated designs often led to performance mismatches in integrated CPS. For example, unpredictable communication delay variations can destabilize control loops, while computation offloading decisions that disregard

network congestion can degrade responsiveness [7, 21, 42]. As industrial systems evolve toward distributed, networked, and human-in-the-loop architectures, the interdependencies among these domains become increasingly critical, revealing the inherent limitations of separate design. This motivates the development of integrated co-design paradigms that jointly account for the coupled dynamics of communication, computing, and control to ensure end-to-end performance.

Early studies in each subsystem nonetheless established solid theoretical and technical foundations for later co-design research.

Communication optimization: Significant efforts have been devoted to improving Quality of Service (QoS) through channel modeling, adaptive modulation, and scheduling algorithms. In this context, the performance metrics for CPS communication systems are often aligned with standardized Key Performance Indicators (KPIs) defined by 3GPP, as summarized in Table 2.1 [43]. These KPIs specify the quantitative targets for representative CPS-related applications, including Extended Reality (XR), holographic communication, digital twins, and multimodal immersive collaboration. Building upon these standardized requirements, numerous studies have sought to optimize the communication module to meet such stringent performance demands in practical CPS scenarios. In [24] and [44], the authors discussed the enhancements required for traditional Ethernet networks to achieve real-time performance and reviewed several Real-Time Ethernet (RTE) technologies. Similarly, [45] analyzed the challenges of introducing wireless communication into industrial environments, focusing on wireless sensor networks for process automation and remote actuation. However, as highlighted in [46], discrepancies between theoretical analysis and practical test results often emerge, with measured performance typically falling below expected values due to interference, congestion, and hardware limitations.

Control optimization: In networked control systems, the primary research focus has been maintaining closed-loop stability under stochastic delay and packet loss. For instance, [47] proposed a distributed autonomous control framework integrating multiple control schemes for fault analysis and decision-making, improving system reliability. To achieve higher flexibility and integration in production systems, [13] introduced a self-adaptive collaborative control model for intelligent logistics, enhancing coordination and resilience in dynamic manufacturing environments.

Computing optimization: With the advent of edge and cloud computing, numerous studies explored task scheduling, computation offloading, and model compression to improve latency and energy efficiency. For example, [48] introduced lightweight neural network architectures with tunable parameters to balance latency and accuracy, enabling deployment on embedded or mobile devices. Surveys such as [49] and [50] reviewed model compression techniques including pruning and quantization, which accelerate on-device inference while reducing memory footprint and power consumption.

Overall, these modular advances significantly improved performance within individual do-

mains but failed to account for the cross-domain dependencies that dominate modern CPS. The lack of coordinated design among communication, control, and computing modules often results in suboptimal E2E performance under dynamic and interconnected operating conditions.

2.1.2 From Separate Design to Cross-System Co-Design

Recognizing the limitations of separate design, researchers have shifted toward cross-system co-design, which treats communication, computing, and control as interdependent subsystems within a unified feedback loop. This paradigm aims to jointly optimize network scheduling, computation offloading, and control actions under a shared task objective, enabling end-to-end performance optimization instead of isolated efficiency gains. By explicitly modeling the dynamic coupling among communication, computing, and control processes, co-design frameworks seek to minimize latency, improve reliability, and enhance system-level stability under constrained resources. Significant contributions have been made to reduce the latency by the strategy of cross-System co-design [12, 14, 25, 26, 29, 51, 52]. The authors in [51] introduced an integrated scheduling method for sensing, communication, and control in Unmanned Aerial Vehicle (UAV) networks using mmWave/THz communications, enhancing backhaul data transmission. They analyzed the interaction between sensing and motion control, introducing a “state-to-noise-ratio" concept linking control patterns to data rates. The authors in [26] optimized the data transmission in Industrial Internet of Things (IIoT) systems, which derived the average AoI expression under packet loss and finite retransmissions, linking it to control performance and communication energy consumption. The authors in [14] considered URLLC scenarios, where mobile devices predict future states and send them to a data center. They optimized resources to improve delay-reliability tradeoffs, accounting for prediction errors and packet losses. The authors in [12] introduced a proactive tile-based video streaming method for wireless VR to reduce motion-to-photon latency by predicting and delivering tiles before playback. However. most of these efforts are not task-oriented, resulting in a gap between traditional metrics for communications and the KPIs. In addition, these efforts do not directly consider the interaction between the human and the environment in the CPS.

Although these studies demonstrate the potential of cross-system co-design to enhance system efficiency and responsiveness, most of them remain communication- or control-centric, rather than fully task-oriented. As a result, there remains a mismatch between traditional communication metrics and system-level KPIs, particularly when considering the human–environment interaction inherent in CPS. This observation motivates the next stage of research evolution—from co-design at the system level to cross-system optimization for real-time performance, where communication, computing, and control are not only jointly modeled but also dynamically coordinated to meet real-time, human-centered objectives.

2.1.3 Cross-System Optimization for Real-Time Performance

Real-time performance has become a defining requirement for modern CPS, where sensing, computation, communication, and control processes must operate within strict timing constraints [53, 54]. In applications such as industrial automation, autonomous systems, and the Industrial Metaverse, even millisecond-level latency can significantly degrade stability, safety, or user experience [55]. However, ensuring real-time operation is fundamentally challenging because delays and uncertainties arise from multiple domains—communication congestion, computational load, control-loop dynamics, and even human interaction. Therefore, recent research has shifted from domain-specific improvements to cross-system optimization, which jointly manages resources and decision processes across communication, computing, and control layers to maintain deterministic timing and stable performance under dynamic conditions [28].

Early studies on real-time optimization primarily focused on minimizing latency in Wireless Networked Control Systems (WNCSs) and edge-assisted industrial networks, where the coupling between data transmission, computation delay, and control sampling interval was explicitly modeled. In [28], the authors proposed a cross-domain optimization framework that simultaneously adjusts communication scheduling and control gains to ensure stability under time-varying network delay. The authors in [56] introduced a dynamic task offloading and feedback control strategy for edge computing systems, reducing computation and communication delays while preserving control accuracy. In addition, the authors in [57] investigated how packet loss, jitter, and delay variation influence control stability, proposing adaptive transmission policies that balance reliability and responsiveness. These studies collectively demonstrate that real-time guarantees cannot be achieved by optimizing a single subsystem, but rather through coordinated optimization across the entire cyber–physical stack.

Beyond delay minimization, recent works have emphasized deterministic and adaptive real-time performance, where system behavior remains predictable even under stochastic network and computational conditions. In our previous work [30], we explored cross-layer optimization strategies that combine physical-layer scheduling, network routing, and control adaptation to achieve URLLC performance. Similarly, [58] proposed a hierarchical optimization framework that integrates computation offloading with feedback scheduling, enabling time-sensitive applications to meet latency and stability constraints simultaneously. Emerging methods also leverage reinforcement learning and Model-Predictive Control (MPC) to perform real-time resource allocation, enabling the system to adaptively adjust to workload fluctuations and unpredictable network dynamics [59]. Such learning-driven cross-system optimization has shown strong potential for sustaining sub-millisecond delay and maintaining closed-loop stability in distributed and human-in-the-loop environments.

Overall, cross-system optimization for real-time performance represents a critical step in the evolution from conceptual co-design to operational implementation. By integrating communication, computing, and control decisions under a unified objective function, these approaches

enable proactive, coordinated resource management to ensure deterministic timing and high reliability. Yet, most existing optimization frameworks still treat performance in terms of system metrics such as delay or throughput, rather than explicit task success or semantic relevance. The next subsection therefore introduces the notion of task-oriented communication paradigms, which redefine real-time optimization goals around mission success and task-level performance in complex CPS environments.

2.1.4 Task-Oriented Communication Paradigms

Traditional communication paradigms often measure success based on bit error rates (BER) or throughput, emphasizing the reliability and efficiency of data transmission. However, such metrics are not always aligned with the ultimate goals of CPS, where the task at hand is the primary performance measure. Task-oriented communication shifts the focus from low-level transmission metrics to task completion success, optimizing communication strategies based on the specific needs of the task rather than raw data fidelity [60]. This paradigm has found increasing application in a wide range of CPS, including IIoT [61], Unmanned Ground Vehicle (UGV) and UAV communication systems [62], autonomous driving [63], and the Industrial Metaverse [30].

For example, in [61], the authors explore a WNCS scenario, proposing a goal-oriented scheduling and control co-design policy. This policy prioritizes the transmission of goal-relevant information, minimizing violation probability under resource constraints. Similarly, in UAV systems, the authors of [64] introduce a task-oriented communication framework that utilizes DRL to optimize control and communication data selection and repetition, outperforming traditional Proportional Integral Derivative (PID) methods. These approaches underscore the potential of task-oriented communication to dynamically adjust transmission strategies based on real-time application requirements.

In the Industrial Metaverse, task-oriented paradigms take on additional complexity. For instance, in [30], the authors propose a task-oriented cross-system design framework aimed at minimizing packet rates for accurate robotic arm modeling. This framework is tailored to the Industrial Metaverse, where virtual robotic arms serve as "Digital Shadows", mimicking the physical world without the ability to process environmental and user inputs to make autonomous decisions [65]. Furthermore, the study's design was based on offline data training and testing, leaving the critical challenge of addressing real-time latency in systems with human-in-the-loop unaddressed.

Our work in Chapter 3 [66] explored a similar framework for the Industrial Metaverse but with a limited focus on a single task, which restricted its generalizability and robustness across multiple tasks. As the complexity of task-oriented communication increases, particularly in dynamic and real-time environments like the Industrial Metaverse, future systems must consider how to scale and adapt across diverse tasks and contexts.

2.1.5 Prediction and Scheduling in Co-Design Frameworks

In the context of co-design frameworks for CPS, scheduling and prediction have emerged as fundamental research areas that directly influence system performance. Effective scheduling strategies ensure that communication, computation, and control tasks are allocated with minimal delays and maximum resource utilization. Similarly, prediction techniques are employed to forecast future system states, enabling proactive adjustments to prevent performance degradation. Together, these two elements are crucial for achieving real-time performance in complex, dynamic environments where uncertainties such as communication delays, computational overloads, and control disruptions are prevalent.

Over the years, scheduling has evolved from basic task allocation strategies in static systems to more adaptive and dynamic approaches that consider time-varying workloads, resource availability, and task priorities. Early works in scheduling mainly focused on cyclic scheduling and fixed priority schemes, where tasks were allocated at predefined intervals based on static priorities or deadlines [67]. However, as systems became more complex, researchers began integrating dynamic scheduling approaches that adapt to changing system conditions, such as network congestion or computational load [68]. This led to the development of real-time scheduling algorithms, such as Earliest Deadline First (EDF) and Rate-Monotonic Scheduling (RMS), which dynamically adjust task execution based on priority or deadline requirements [69]. Alongside scheduling, prediction has seen significant advancement as a tool for enhancing system efficiency. Early prediction methods were largely focused on predicting simple metrics, such as packet loss or communication delay, in isolated subsystems [70]. Over time, prediction techniques have become more sophisticated, with researchers developing models to forecast the state of the entire system, including control states, network conditions, and computational loads [71]. Modern prediction methods use machine learning and statistical modeling to make more accurate forecasts based on historical data, allowing systems to anticipate and mitigate delays before they occur [72].

DRL has played a pivotal role in transforming both scheduling and prediction within co-design frameworks [12,27,30,66,73]. In scheduling, DRL allows systems to dynamically adjust task priorities based on real-time feedback from the environment, improving both responsiveness and resource utilization [73]. By continuously learning from past decisions and outcomes, DRL-based scheduling approaches can adapt to system fluctuations, such as changes in network traffic or computational demands, ensuring that critical tasks are executed within strict time constraints. Similarly, in prediction, DRL has been applied to forecast future system states—such as network congestion, computation loads, or control actions—by learning from the system's behavior over time [66]. This ability to anticipate future conditions enables proactive resource management, ensuring that potential delays or system overloads are mitigated before they impact performance [12, 27]. The integration of DRL with scheduling and prediction not only enhances real-time performance but also enables systems to continuously adapt to uncertainties,

making them more resilient and efficient in dynamic environments.

In summary, reinforcement learning has revolutionized the way prediction and scheduling are approached in co-design frameworks. By continuously learning from system feedback and adapting to dynamic conditions, DRL enhances the system's ability to maintain real-time performance across multiple subsystems. As CPS become more complex and interconnected, the integration of DRL with co-design principles will play an increasingly critical role in ensuring real-time performance, adaptability, and resilience.

2.2 Cyber-Physical Systems, Digital Twins, and Industrial Metaverse

Table 2.2: Conceptual Relationship among Cyber-Physical System, Digital Twin, and Industrial Metaverse

Concept	**Definition**	**Function**	**Applications**	**Relation with Others**
CPS	Integration of physical and digital components via real-time data exchange.	Monitoring, control, optimization.	Smart manufacturing, IoT, automation.	Forms the foundation enabling DT and industrial Metaverse through real-time interaction.
DT	Digital model of a physical entity updated by real-time data.	Simulation, prediction, optimization.	Industry, logistics, smart cities.	Core element of CPS; bridges CPS and industrial Metaverse by mirroring physical systems.
Industrial Metaverse	Immersive virtual space combining VR/AR and digital twins.	Interaction, visualization, collaboration.	Education, industry, entertainment.	Extends CPS and DT into virtual environments for real-time human–machine interaction.

CPS integrate physical processes with computational elements, forming a feedback loop between the physical and digital worlds through real-time data collection and action execution via sensors and actuators. A key component of CPS is DT, which provide digital replicas of physical systems, continuously updated with real-time data for efficient monitoring, predictive analysis, and optimization. This integration of CPS and digital twins extends into the Industrial Metaverse, where immersive virtual environments, using real-time interaction between digital and physical replicas, enable new opportunities in areas like remote collaboration [74], training [75], and entertainment [76]. To clarify their interconnections and complementary roles, the key char-

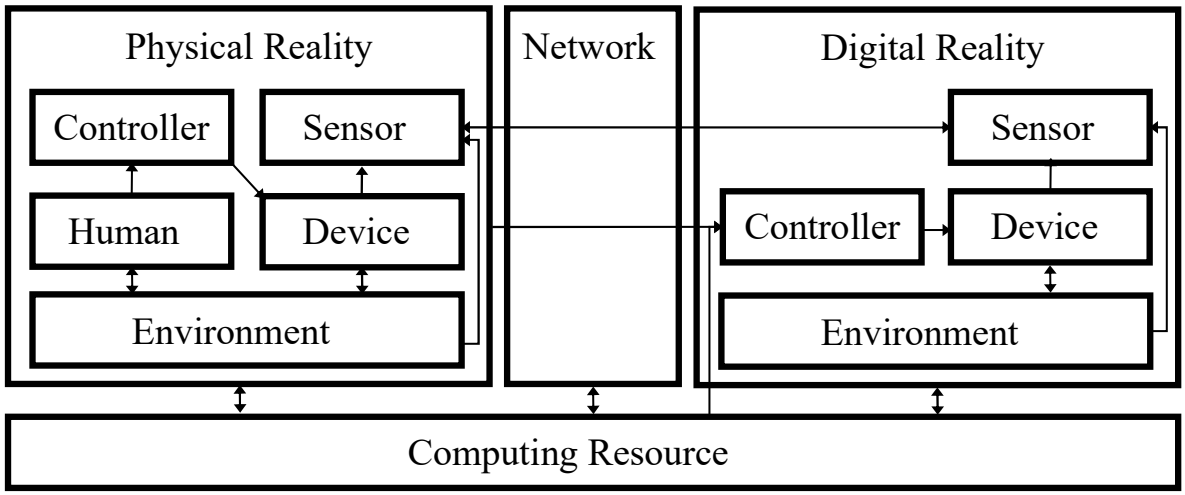

Figure 2.1: Illustration of a typical CPS framework. The system integrates the physical and cyber spaces through sensing, communication, computing, and control loops.

acteristics and relationships among CPS, DT, and the Industrial Metaverse are summarized in Table 2.2 [1, 16, 52, 74–77]. Together, these technologies enable enhanced decision-making, operational efficiency, and new modes of interaction in both physical and virtual realms.

Edge computing plays a critical role in CPS by offloading computational tasks from the cloud to edge devices. This reduces latency and ensures real-time processing, which is vital for time-sensitive applications. By distributing resources more efficiently, edge computing helps improve the responsiveness and scalability of CPS [27]. However, challenges related to latency and reliability persist in CPS, especially for safety-critical systems where delays or failures can lead to catastrophic consequences [14]. Ensuring real-time performance and high reliability requires advanced scheduling, prediction models, and fault tolerance strategies.

The following sections will explore each of these components in more detail, examining their roles, challenges, and emerging technologies.

2.2.1 Fundamentals of Cyber-Physical System and Digital Twin

A CPS establishes a tight integration between the physical world and the digital world, enabling real-time perception, computation, communication, and control. As shown in Fig. 2.1, the CPS framework consists of multiple interrelated layers spanning both physical and digital domains. In the physical reality, sensors continuously collect environmental and operational data—such as position, temperature, or vibration—from devices and production environments. These data are transmitted to controllers and actuators, which execute control commands to influence the physical system in real time. Humans also participate in this loop as operators or supervisors, providing high-level decision-making and ensuring system safety [15]. In the digital reality,

the collected data are transmitted through a communication network to computing resources distributed across edge and cloud infrastructures. Within this space, data are stored, processed, and analyzed to generate system insights. Algorithms running on these platforms perform tasks such as fault detection, predictive control, and optimization, thereby transforming raw sensor data into actionable intelligence [78].

Bridging the two realities, communication networks serve as the information backbone of CPS, ensuring reliable and low-latency transmission required for real-time feedback and coordinated control [47]. The tight feedback loop between sensing, computation, and actuation allows CPS to achieve high autonomy, adaptability, and robustness in dynamic environments. DT extend CPS by creating high-fidelity virtual representations of physical assets, processes, or systems. These digital replicas maintain continuous synchronization with their physical counterparts through real-time data exchange. DTs enable simulation, prediction, and optimization within the digital space, offering a virtual environment for testing and decision-making without interfering with the physical system [13].

A typical CPS operates as a closed-loop workflow that continuously integrates sensing, computation, communication, and control to maintain real-time interaction between the physical and digital domains. This process can be conceptually divided into six interdependent stages:

1. **Sensing:** Physical processes are continuously monitored through distributed sensors, capturing heterogeneous and time-varying environmental data.

2. **Semantic Abstraction:** The raw sensory data are preprocessed and abstracted into meaningful digital representations, often selecting or compressing information according to task relevance.

3. **Communication:** Processed data are transmitted across wired or wireless networks under latency and reliability constraints, ensuring that time-critical information is delivered promptly.

4. **Control and Decision:** Based on received data, computational modules perform state estimation, prediction, or decision-making to generate appropriate control actions.

5. **Rendering and Actuation:** The generated control commands are executed through actuators or visual interfaces, applying adjustments to the physical environment or providing real-time feedback to human operators.

6. **Feedback and Adaptation:** The results of these actions are observed and fed back into the loop, updating system models, refining control policies, or reprioritizing communication tasks to improve long-term performance.

This cyclical workflow emphasizes continuous perception, computation, and actuation, forming the foundation of modern task-oriented and adaptive CPS architectures.

Building upon this foundational architecture, numerous researchers have proposed optimizations and extensions tailored to specific application demands or system-level constraints. Some studies focus on enhancing communication reliability and latency, developing adaptive protocols and edge-assisted data transmission strategies to ensure stable feedback in dynamic industrial environments [79]. Others concentrate on computing and control integration, optimizing scheduling, task offloading, and resource allocation to improve real-time performance and energy efficiency [80]. In addition, task-oriented frameworks have been introduced to align system objectives directly with application goals, emphasizing end-to-end performance rather than isolated subsystem metrics [81]. These advancements collectively enrich the traditional CPS–DT architecture, evolving it from a static data–control loop into an intelligent, adaptive, and task-aware ecosystem that supports complex applications such as autonomous production, human–robot collaboration, and immersive industrial Metaverse systems.

2.2.2 Human–Digital–Physical Integration in the Industrial Metaverse

The Metaverse represents the next evolutionary stage of cyber–physical convergence, enabling seamless integration among humans, digital agents, and physical entities. Unlike conventional CPS, which primarily emphasize real-time control and monitoring, the Metaverse focuses on immersive interaction, collaborative operation, and bidirectional synchronization between the physical and digital domains [16, 17]. Through technologies such as XR, digital twins, haptic interfaces, and high-fidelity simulation, users are able to perceive, manipulate, and interact with virtual replicas of physical systems in real time [18, 60]. This human–digital–physical loop transforms traditional one-way monitoring architectures into shared, interactive ecosystems that emphasize perception, cognition, and decision-making.

In the industrial domain, this paradigm has evolved into the Industrial Metaverse, where digital twins, IIoT, and intelligent automation converge to support next-generation smart manufacturing [18]. By integrating immersive visualization and predictive analytics, the Industrial Metaverse enhances situational awareness, human–robot collaboration, and system-level adaptability [82]. Recent implementations have demonstrated its capability in remote collaboration, operator training, process optimization, and agile fault diagnosis [19]. These developments highlight the Industrial Metaverse as a core enabler of Industry 5.0, emphasizing human-centric intelligence and cyber–physical co-evolution.

However, achieving real-time human–digital–physical integration introduces stringent requirements on latency, reliability, and computational efficiency. The continuous synchronization between physical assets and their digital counterparts demands massive sensing data throughput, high computing density, and deterministic communication performance [83]. Traditional cloud-based infrastructures are often insufficient to support such requirements, particularly under dynamic workloads and heterogeneous device coordination [84]. Therefore, advanced edge computing architectures, distributed resource offloading, and task-oriented co-design frameworks

have been proposed to balance computation, communication, and control demands across multiple layers [60]. These challenges and emerging solutions are discussed in detail in the following sections on Edge Computing and Resource Offloading and Latency and Reliability Challenges in CPS.

2.2.3 Edge Computing and Resource Offloading in CPS

As CPS become increasingly distributed and data-intensive, traditional cloud-centric architectures often struggle to meet real-time and reliability requirements. To address this, edge computing has emerged as a key enabler that brings computation and intelligence closer to data sources, bridging the gap between end devices and centralized cloud servers. By distributing computation, communication, and control workloads across hierarchical layers — cloud, edge, and device — CPS can achieve faster response times, improved scalability, and enhanced robustness.

By balancing the heterogeneous communication, computation, and control capabilities of different devices, and offloading some or all tasks to the edge or cloud, collaborative computing and multidimensional resource sharing (including communication, caching, and processing) can be efficiently achieved in industrial CPS [85–91]. Recent research has further explored how intelligent offloading mechanisms can dynamically optimize this balance under varying workloads and network conditions.

For instance, to handle the complexity and diversity of tasks in CPS, the authors in [85] proposed a task offloading scheme for Mobile Edge Computing (MEC) systems that minimizes deadline violations by using task migration, merging, and deep learning to optimize execution order and reliability. To improve fault management and system agility, [86] introduced a low-code intelligent fault diagnosis platform that integrates cloud-based infrastructure with edge-adaptive systems, demonstrating enhanced responsiveness in wind turbine and reduction gear systems. Furthermore, leveraging accurate digital representations of heterogeneous device and network parameters in dynamic IIoT scenarios, edge-based DT models have been constructed to capture time-varying demands with low latency [87]. In [88], the authors proposed a cloud–edge–end collaborative computation offloading scheme driven by a DT-based Edge Coalition Formation methodology, improving the overall efficiency and utility of edge servers.

These developments collectively demonstrate that integrating edge computing and resource offloading into the CPS–DT architecture enables real-time intelligence, scalable deployment, and resilient control, laying the groundwork for future industrial applications that require both autonomy and human-in-the-loop interaction.

2.2.4 Latency and Reliability Challenges in CPS

Building on the advances in edge computing and collaborative offloading, ensuring real-time responsiveness and ultra-high reliability remains one of the most critical challenges in modern CPS. While edge computing alleviates network congestion and reduces end-to-end latency, the performance of large-scale distributed systems is still constrained by heterogeneous communication links, resource imbalance, and dynamic workloads. These challenges are further amplified in emerging immersive applications such as the CPS, which demand strict coordination between physical devices, digital models, and human interactions.

Despite rapid advances in 5G and ongoing developments toward 6G, a substantial performance gap persists between existing network capabilities and the stringent requirements of real-time CPS and the industrial Metaverse services. Conventional KPIs such as latency, reliability, and throughput do not directly correspond to task-level metrics like motion-to-photon delay, semantic perception accuracy, or control stability [16]. Furthermore, communication, computation, and control subsystems are often developed independently [92], leading to inefficient resource utilization and increased coordination overhead. Future 6G systems must therefore support massive heterogeneous devices and dynamic workloads through hierarchical cloud–edge architectures to ensure end-to-end performance [93].

Another fundamental challenge lies in timely and accurate synchronization between physical and digital spaces. Digital Twins, defined as multiphysical and probabilistic digital counterparts of real systems [3], are central to maintaining this synchronization. However, current infrastructures remain insufficient for the large-scale, real-time data exchange required by complex CPS and the industrial Metaverse environments. Recent studies have explored edge-assisted and learning-based optimization for digital twin placement, migration, and model synchronization [76,84,94], which have shown promising improvements in latency and fidelity. Nonetheless, coordinating heterogeneous computing, communication, and storage resources at scale remains an open research problem.

To bridge the mismatch between network-level and task-level performance, recent works have adopted task-oriented cross-system co-design approaches that jointly optimize communication, computation, and control processes. Such frameworks directly target task success—e.g., perception accuracy or interaction stability—rather than conventional communication metrics [83, 95]. By integrating edge intelligence, predictive modeling, and dynamic resource management, these methods significantly enhance real-time performance and user experience in large-scale, immersive CPS. However, challenges persist in developing unified semantic representations [96] and scalable optimization algorithms to solve the inherently non-convex problems of cross-system coordination.

2.3 Human Interaction in CPS

Within the CPS framework, human interaction serves as a critical component that enables adaptive control, situational awareness, and shared decision-making among humans, machines, and networked environments. However, the inherent complexity and dynamic coupling among communication, computation, and control modules make traditional model-based design insufficient for achieving optimal performance in real-world applications. To address these challenges, DRL-based methods have emerged as powerful tools for co-design optimization, offering the ability to learn adaptive policies that jointly enhance communication efficiency, control stability, and computational allocation [27, 29].

In this context, DRL provides a unified framework for optimizing system-wide objectives through continuous interaction with the environment. By observing feedback from both human operators and physical dynamics, the DRL agent can adjust communication strategies, control parameters, and task scheduling to maintain E2E performance under uncertainty [4, 27]. Meanwhile, meta-learning accelerates adaptation to new users or tasks by learning generalized priors that can be rapidly fine-tuned with minimal data, thereby supporting personalized and resilient human–machine collaboration [97]. When combined with HITL based DRL and Reinforcement Learning from Human Feedback (RLHF), these frameworks further align system behavior with human intent and safety constraints [4, 98].

2.3.1 Learning-Based Co-Design Optimization in CPS

DRL provides a unified framework for decision-making and optimization in dynamic and uncertain environments, making it particularly suitable for CPS co-design problems [27]. Unlike supervised learning, which requires labeled datasets, DRL enables an intelligent agent to learn optimal policies through continuous interaction with the environment. At each time step, the agent observes the system state, executes an action, and receives a scalar reward that reflects the task objective. By maximizing the long-term expected reward, the agent learns to balance competing objectives such as latency, reliability, energy efficiency, and control accuracy [29]. The core of DRL lies in policy optimization, where a parameterized policy $\pi_\theta(a|s)$ maps observed states s to actions a, and the parameters θ are iteratively updated using reward feedback and value estimation.

This learning mechanism aligns naturally with the closed-loop structure of CPS. In a typical CPS, sensing, communication, computation, and control form a continuous feedback loop that governs system performance. DRL agents can be embedded into this loop to optimize cross-domain decisions, such as selecting which sensory data to transmit, determining when to offload computational tasks, or adjusting control gains under uncertain network conditions [30]. By interacting with both digital models and real-world feedback, the DRL-based co-design framework learns policies that jointly optimize the communication–computation–control chain, improving

end-to-end stability and responsiveness [61].

Moreover, DRL provides a flexible mechanism for multi-objective optimization, where the system reward function can integrate heterogeneous metrics across layers—such as throughput, delay, control error, or user experience—into a unified performance criterion. This enables CPS to self-adapt to changing task requirements and operating conditions without explicit analytical modeling, overcoming the limitations of traditional modular optimization. Consequently, learning-based co-design represents a key step toward realizing intelligent, autonomous, and human-aware CPS capable of optimizing global performance through experience-driven adaptation.

2.3.2 Human-in-the-Loop Control and Tele-operation

HITL control represents a critical paradigm in CPS, where human operators, intelligent agents, and physical systems jointly close the perception–decision–action loop [4]. Unlike conventional automated control systems, HITL frameworks incorporate human perception, cognition, and decision-making capabilities, allowing flexible adaptation to uncertain, dynamic, or safety-critical environments [20]. Teleoperation extends this paradigm by enabling remote interaction through real-time sensing, communication, and actuation, allowing human operators to manipulate or supervise physical assets via digital interfaces or haptic feedback systems [37].

The inclusion of humans in the control loop introduces strong bidirectional coupling between human perception and system dynamics. Humans not only influence the system through control inputs, but also continuously adjust their responses based on system feedback. This mutual dependence makes *real-time performance* a critical requirement: any delay, jitter, or inconsistency in communication and rendering can degrade both system stability and human perception of transparency [31]. For instance, excessive motion-to-photon latency in teleoperation or virtual reality systems can cause motion sickness or operational errors, while control delay may destabilize feedback loops [82]. Hence, ensuring low-latency and high-reliability communication becomes essential not only for system performance but also for maintaining natural and safe human experience.

To mitigate these challenges, advanced HITL architectures employ predictive control, delay compensation, and shared autonomy to dynamically balance control authority between human operators and autonomous agents [33]. Transparency—the human's ability to perceive, understand, and anticipate system behavior—plays a pivotal role in preserving trust and operational efficiency, especially under constrained bandwidth or delayed feedback conditions. Recent research integrates learning-based techniques, where reinforcement learning and adaptive modeling are used to predict human intent and compensate for network-induced latency [20, 66]. These developments enable smoother coordination, improved responsiveness, and reduced cognitive load during complex teleoperation tasks.

In summary, HITL control transforms CPS from purely automated systems into human inter-

action ecosystems, where human perception and system intelligence must be co-optimized. The dual requirement of real-time responsiveness and transparency makes it essential to jointly design communication, computation, and control mechanisms that respect both physical dynamics and human cognitive constraints. This motivates the integration of learning-based adaptation, as discussed in the next section on meta-learning and preference-based human adaptation.

2.3.3 Meta-Learning for Human Adaptation

Meta-Learning provides a promising mechanism for rapid adaptation in CPS, where human behaviors, task dynamics, and environmental conditions vary significantly over time [97]. Unlike conventional learning methods that train models for specific tasks, meta-learning aims to acquire transferable priors that enable the system to quickly adapt to new situations with minimal retraining. This capability is particularly critical in HITL scenarios, where human motion patterns, control preferences, or cognitive responses are highly individual and non-stationary [4].

In the context of CPS and teleoperation, meta-learning facilitates personalized and adaptive control by continuously updating control policies according to human feedback. Model-Agnostic Meta-Learning (MAML) and its variants, such as Reptile and Meta-SGD, have been employed to initialize control policies that can be fine-tuned to specific operators using only a few interaction samples [97]. This enables the system to accommodate variations in human reaction speed, force application, and perception latency while maintaining overall control stability and transparency. For example, in shared control and telemanipulation, meta-learned policies allow the controller to adapt to new users or tasks in real time, reducing calibration time and improving task accuracy [99].

Beyond single-task adaptation, meta-learning also plays a key role in multi-task and sequential-task settings, which are common in industrial and collaborative CPS environments. In these systems, tasks often evolve over time—from perception and manipulation to decision-making and planning—each requiring different control strategies. Multi-task meta-learning frameworks enable the system to transfer knowledge across related tasks, improving generalization and reducing the data required for retraining [35]. Similarly, hierarchical or phase-based meta-learning approaches allow adaptation across different stages of complex workflows, such as assembly, inspection, or maintenance, ensuring consistency and robustness across changing objectives [32]. These capabilities make meta-learning particularly suited for dynamic, human-centered industrial systems, where task structures and operator interactions continuously evolve.

Furthermore, meta-learning serves as a bridge between DRL and human adaptation in co-design frameworks. Traditional RL approaches often require extensive interactions to converge, which is impractical in HITL systems due to user fatigue and safety constraints. Meta-RL methods, such as context-based policy adaptation or gradient-based meta-policy optimization, enable efficient policy transfer across tasks and users [32]. By embedding prior knowledge about human intent and system dynamics, these methods accelerate convergence and reduce exploration

overhead, making them well-suited for online adaptation in complex human–cyber–physical environments.

2.3.4 Preference Learning in Human–Robot Interaction

RLHF have emerged as key techniques for aligning machine behavior with human intent, comfort, and satisfaction in CPS and Human-Robot Interaction (HRI) systems [98]. Unlike traditional reinforcement learning, which relies solely on explicit reward functions, RLHF leverages human evaluations or preferences to guide policy learning, thereby bridging the gap between algorithmic optimization and subjective human experience. This alignment is essential in shared autonomy, teleoperation, and collaborative robotics, where task success is not determined purely by system efficiency but also by perceived safety, transparency, and trust [33].

In preference learning frameworks, human feedback is typically collected in the form of pairwise comparisons, rankings, or scalar ratings, which are used to infer an implicit reward model representing the human's latent objective [36]. By integrating these models with reinforcement learning algorithms, robots and control systems can iteratively refine their policies to better reflect human judgments, even in scenarios where explicit task rewards are difficult to define. Recent studies have demonstrated that preference-based RL can be employed to optimize complex behaviors in high-dimensional control tasks such as dexterous manipulation, telepresence control, or social interaction [34].

RLHF extends this concept by embedding human evaluative signals—such as approval, correction, or physiological responses—into the training loop [100]. In HRI, RLHF allows systems to learn nuanced human preferences for motion smoothness, timing, and response under real-time constraints. For example, an assistive robot can adapt its grasping speed or trajectory smoothness based on continuous human feedback, achieving not only functional accuracy but also user comfort and trust. These adaptive mechanisms are crucial in dynamic and uncertain environments where predefined cost functions fail to capture human-centered goals.

Moreover, combining RLHF with meta-learning enhances adaptability across users and contexts. Meta-preference learning frameworks can rapidly infer new users' preferences from limited interactions and generalize across different tasks or behavioral domains [33]. This integration enables the design of personalized co-adaptive systems capable of evolving their behavior to reflect individual needs, skill levels, and interaction histories.

In summary, preference learning and RLHF establish a foundation for truly human-aligned CPS, where system intelligence not only pursues objective performance but also accounts for subjective human experience. By embedding human feedback into the co-design loop, these approaches move beyond traditional automation toward collaborative intelligence—where humans and machines jointly optimize shared goals in real time.

2.4 Summary and Research Gaps

The reviewed literature has established the foundation for integrated CPS and Industrial Metaverse systems that combine communication, computation, and control through cross-system co-design. However, existing studies remain fragmented, often neglecting human adaptation and preference alignment as integral components of system intelligence. To address these limitations, this thesis develops a unified, task-oriented, and human-adaptive co-design framework that bridges learning-based optimization and human feedback integration. The following gaps summarize the key technical deficiencies that motivate this work:

- **Human-in-the-Loop Task-Oriented Co-Design.**

  Current co-design frameworks optimize subsystems independently and seldom consider dynamic human feedback during task execution. There is a lack of a real-time, task-oriented paradigm that unifies communication, computation, and control through HITL adaptation. This thesis introduces a DRL-based co-design model that jointly optimizes system resources and control actions according to task-level KPIs such as MTP and AoL, enabling adaptive coordination across heterogeneous domains.

- **Meta-Learning for Multi-Task and Human Adaptation.**

  Most existing DRL frameworks struggle to generalize across different tasks, users, or environments. A meta-learning-based methodology, such as MAML, is required to accelerate policy transfer and ensure personalized control in multi-task industrial scenarios. This thesis formulates a HITL meta-learning framework that continuously adapts to new operators and evolving task conditions, thereby enhancing generalization and reducing retraining overhead.

- **Preference-Aware Reinforcement Learning and Human Feedback Alignment.**

  Traditional DRL relies on explicit rewards, which fail to capture human intent and subjective comfort. Integrating preference learning and RLHF into the co-design loop remains an open challenge. This thesis bridges this gap by modeling implicit human feedback as latent reward signals, allowing the system to learn policies that balance task efficiency, transparency, and user satisfaction in a unified optimization process.

- **Integrated Evaluation and Human-Centered Validation.**

  Despite the conceptual advances in human-centered CPS, there remains a lack of experimental validation that links task-level performance metrics to human experience indicators such as perceived trust, transparency, and cognitive load. This thesis implements a prototype platform integrating predictive control, preference-aware learning, and real-time human feedback, providing a comprehensive evaluation framework for assessing the effectiveness of human–machine co-design.

In summary, these research gaps highlight the need for a unified co-design paradigm that synergizes human-in-the-loop learning, meta-adaptive generalization, and preference-aware optimization. Addressing them is crucial for advancing intelligent, trustworthy, and task-oriented CPS and Industrial Metaverse systems capable of real-time collaboration between humans and machines.

Chapter 3

User-Centric Reinforcement Learning in Co-design CPS Framework

3.1 Introduction

The Metaverse is envisioned as the next generation of the internet, which aims to revolutionize the connections among humans, real-world devices, and virtual objects [16]. The virtual objects in the Metaverse can be considered as the digital mirror of the real world which is called “digital twins" [3]. Digital twins can help to simulate real situations and their outcomes, ultimately allowing them to make better decisions. For example, remote orthopedic surgery enables patients in underserved areas to access specialized surgical expertise without the need for extensive travel. It can combine advanced technologies to replicate the surgeon’s skills, enhance visualization, and ensure real-time communication, ultimately improving patient outcomes and expanding the reach of specialized medical care [54].

Although the Metaverse realizes more advanced interaction, there are still many challenging problems remaining to achieve the full vision of the Metaverse. First, the increasingly ubiquitous application of AI and growing sophistication of rendering techniques in the Metaverse imposes a substantial burden on computational resources. At the same time, various heterogeneous data from different sensors need to be transmitted, thus challenging the data rate, latency, and reliability of the communication system to achieve seamless interconnectivity and immersive user experience [6]. The fine-grained KPIs in VR video streaming have been released in [43], where the typical KPIs include 1 Gbps (smooth play) or 2.35 Gbps (interactive) data rate [8] to support 8K resolution, 120 FPS, and 360 degree visual field VR video streaming. Besides, the seamless transmission of haptic sensor feedback is also crucial for achieving an immersive experience in the Metaverse. Latency in data transmission is particularly critical for haptic feedback since humans are more sensitive to touch delays than audio or visual ones [101]. In light of this, unlike traditional communication systems designed for high data rates in audio and visual media, the Tactile Internet (TI) prioritizes replicating the sense of touch and kinesthetics [5]. To cre-

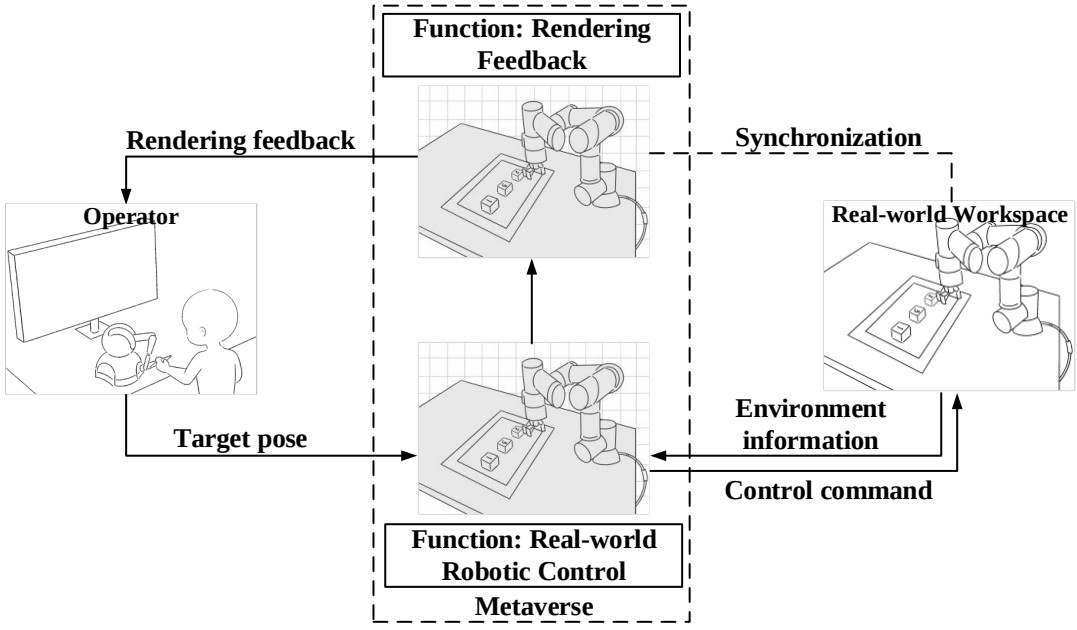

Figure 3.1: Proposed real-time interactions framework for humans, a real robotic arm, and its coupled virtual robotic arm in the Metaverse, where sensing, communication, predication, control, and rendering are considered.

ate a seamless experience in teleoperation, the Tactile Internet necessitates ultra-low round-trip latency, ideally below 1 ms [101]. This ensures near-instantaneous responses to user actions, mimicking human reaction times and providing realistic sensing feedback in the virtual world. However, existing 5G network standards struggle to meet the stringent requirements of tactile internet [5].

Furthermore, the human-in-the-loop control structure in the Metaverse makes eliminating the effects of time delay even more complicated. Delayed control commands and feedback in real-time interactions can significantly degrade the task performance in the Metaverse [102]. Although the effect of latency on human behavior has been initially investigated in teleoperation [9], the complex interactions and feedback in the Metaverse, as well as more precise quantification, to predict human behavior remains difficult. This further makes it challenging to deploy AI for training and testing. Therefore, how to design a system to improve real-time interaction in the Metaverse is still an open issue.

Significant contributions have been made to reduce the latency in communication systems. Prediction plays an important role in reducing the user-experienced delay in different VR systems [103] [12] [29] [27]. In [12], to reduce the latency, the author collaboratively optimized the duration of the observation window for predicting tiles and the duration for computing and transmitting these predicted tiles, aiming to achieve a balance in performance across prediction, communication, and computing tasks to maximize QoE given any arbitrary predictor and con-

figured resources. In [27], the authors considered the limitations of the computing capacity of end-user devices, where an MEC-enabled wireless VR network is proposed to predict the field of view (FoV) of each VR user by using the Recurrent Neural Network (RNN). However, these efforts only considered the real-time experience of the user on the rendering side and did not take into account the completion of the task in the case of real-time interaction and control. A recent metric named AoI aims to quantify the freshness of information and is gradually used to reduce the latency in real-time control scenarios [104]. In [105], the authors considered the computation of AoI in the case of real-time control with simultaneous sensing, controlling, and actuating. However, these analyses were model-based and lack considerations regarding real-time human interaction systems. It is not feasible for complex systems involving prediction, computation, rendering, and control, especially in the case of human-in-the-loop. Our work [30] predicted the behavior of a operator during real-time control and to reduce the control error of a robotic arm from a task-oriented perspective thereby eliminating the effect of latency. However, the analysis was based on offline data training and testing and did not directly address the latency problem of real-time systems with a human in the loop.

3.2 Conributions

In this chapter, we aim to solve the following problems: 1) How to eliminate the effects of time delay among different subsystem components to achieve real-time interactions? 2) How to do the cross-system design, integrating the domain knowledge of the user behavior to solve the human-in-the-loop real-time interaction issue. 3) How to design the prototype to verify the effectiveness of the proposed DRL algorithm. The main contributions of this paper are summarized as follows:

- We established a novel framework for real-time interactions through the virtual robotic arm in the Metaverse with its coupled real-world remote device, where sensing, communication, prediction, control, and rendering are jointly considered. Specifically, we jointly predict the motion of the human controller for 1) proactive rendering in the Metaverse and 2) generating control commands to the real-world robotic arm in advance. The virtual model is decoupled into two functions for real-world robotic control and rendering feedback, respectively.

- We developed a two-step human-in-the-loop continuous DRL approach based on the advanced PPO algorithm to adjust the prediction horizons for rendering and control dynamically. We take the expert knowledge, e.g., AoL as the state for improving the training efficiency.

- We built a prototype including input device, a real-world robotic arm and its digital model in the Metaverse. Extensive experiments are carried out in the prototype and our proposed

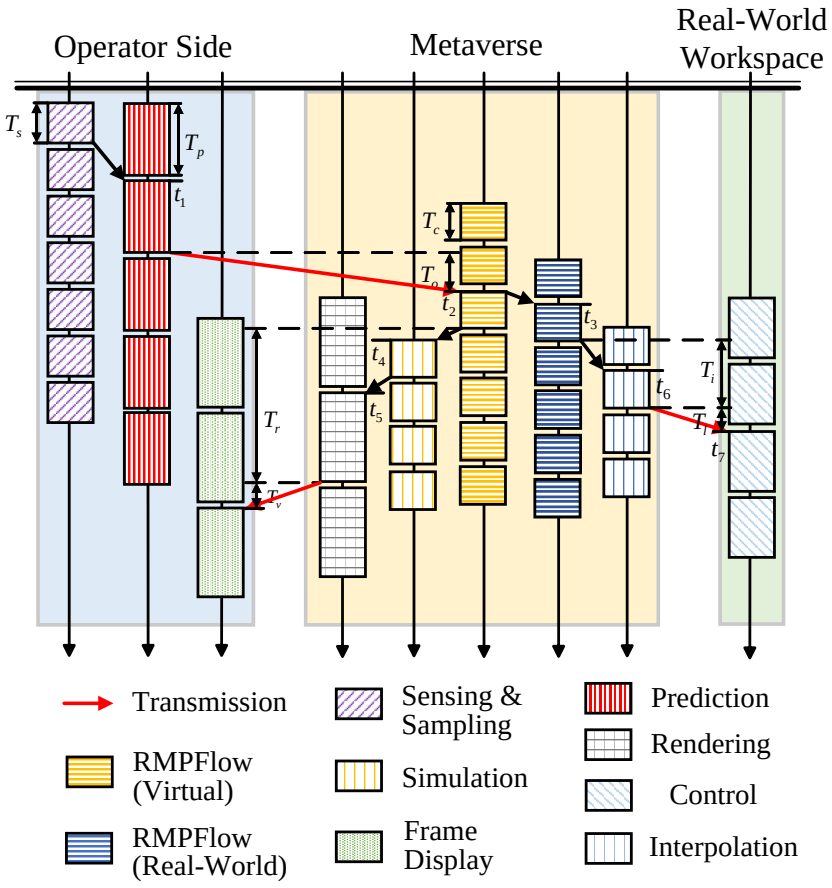

Figure 3.2: The workflow of the proposed framework, where the modeling accuracy and the latency need to be satisfied.

method can reduce 1) the MTP latency between human motion and rendering feedback and 2) the RMSE between human motion and real-world remote device significantly.

3.3 System Model

As shown in Fig 3.1, the framework consists of three parts, an operator operating the input device, the Metaverse with a virtual robotic arm, and a real-world workspace with a real robotic arm. The operator and the two workspaces are at a distance, communicating through a wireless channel. Due to communication delay between the operator and the two workspaces, and the delays caused by prediction, computation, and rendering processes, discrepancies exist in the perceived trajectory of the virtual robotic arm, the real-world remote robotic arm, and the operator's control commands. When the operator operates the controller, the commands are sampled and predicted to 1) compensate for the MTP latency between human motion and rendering feedback and 2) reduce the RMSE between human motion and the real-world remote device.

3.3.1 Information Flow

As shown in Fig 3.2, the proposed framework comprises two information flows served for two functions: real-world robotic control and the rendering feedback. The total time duration of real-world robotic control, denoted as T_{cc}, encompasses all delays from sensing the movement of the operator through the data transmission, Metaverse data processing, and real-world robotic control. It indicates the time duration from the initiation of a control command by the operator to its execution by the robotics. During the process, the operator operates the input device and generates a series of poses $\{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_i\}$ which are sampled by the built-in sensor. This introduces the sensing delay T_s. At the same time, we maintain a queue with a fixed length to store these samples. Next, an agent decides the length of the prediction horizon based on the freshest samples $\mathscr{P}(t)$ in the queue, followed by a predictor generates the predicted pose for the rendering $\hat{\mathbf{p}}_r(t)$ and control $\hat{\mathbf{p}}_c(t)$. Limited by the decision frequency, this process causes a queue delay T_q and a prediction delay T_p. Then, $\hat{\mathbf{p}}_r(t)$ and $\hat{\mathbf{p}}_c(t)$ are transmitted to the Metaverse via networks and the queue is emptied. This process introduces the communication delay T_o. After that, the predicted poses $\hat{\mathbf{p}}_r(t)$ and $\hat{\mathbf{p}}_c(t)$ will be processed in the Metaverse taking into account the simulation or feedback of the environment to generate a joint acceleration control command for the robot and then be interpolated for trajectory smoothing. This will incur a computational delay T_c and interpolation delay T_i. Finally, the command is transmitted to the real-world robotic arm via a communication channel for actuation, incurring a communication delay T_l. Therefore, the total time duration of the function of real-world robotic control T_{cc} is expressed as

$$T_{cc} = T_p + T_o + T_c + T_i + T_l. \tag{3.1}$$

On the other hand, the total time duration of the rendering feedback, denoted by T_{cr}, encompasses all delays from the sensing of movement of the operator, data transmission, Metaverse data processing, rendering, and feedback display. It indicates the duration from the initiation of a movement by the operator to his perception, which also refers to the MTP [106], which is measured as the time difference between the operator input event and the corresponding rendered visual feedback, including sensing, communication, computation, and rendering delays. Specifically, time delays before the step of the Metaverse process are identical to those of real-world robotic control. In addition to the delay of generating control commands for the robotic arm T_c, the virtual environment is also simulated and rendered in the Metaverse, which incurred a rendering delay T_r. Next, the rendered image is streamed back to the user side, which incurs a communication delay T_v. Therefore, the total time duration of the function of rendering feedback T_{cr} is expressed as,

$$T_{cr} = T_p + T_o + T_c + T_r + T_v. \tag{3.2}$$

In the following subsection, we discuss these sub-components in detail.

3.3.2 Operator Side

On the operator side, we utilize the 3D Touch haptic device [107] as the input device. In the t-th time slot, when the operator operates the handle of the 3D Touch, the sampled pose $\mathbf{p}(t)$ is denoted by

$$\mathbf{p}(t) = [l_{\mathrm{x}}(t), l_{\mathrm{y}}(t), l_{\mathrm{z}}(t), q_{\mathrm{x}}(t), q_{\mathrm{y}}(t), q_{\mathrm{z}}(t), q_{\mathrm{w}}(t)], \tag{3.3}$$

where $[l_{\mathrm{x}}(t), l_{\mathrm{y}}(t), l_{\mathrm{z}}(t)]$ is the position of the handle in the Cartesian coordinate system and $[q_{\mathrm{x}}(t), q_{\mathrm{y}}(t), q_{\mathrm{z}}(t), q_{\mathrm{w}}(t)]$ represents the orientation vector in quaternion [108]. To compensate for the time delay discussed in Section 3.3.1, we propose to use ARMA as the predictor for its low computational complexity, robustness, and no need for pre-training [109]. It is worth noting that our designed framework can be easily replaceable with other prediction algorithms. Specifically, in time slot t_1, given historical poses $\mathscr{P}(t_1) = [\mathbf{p}(t_1 - W_p), \mathbf{p}(t_1 - W_p + 1), ..., \mathbf{p}(t_1)]$, ARMA predicts poses for real-time robotic control $\hat{\mathbf{p}}_c(t_1 + H_c(t_1))$ and rendering feedback $\hat{\mathbf{p}}_r(t_1 + H_r(t_1))$ respectively. This is expressed by,

$$\begin{aligned}[\hat{\mathbf{p}}_c(t_1 + H_c(t_1)), &\hat{\mathbf{p}}_r(t_1 + H_r(t_1))] = \\ &\mathscr{F}_p(\mathscr{P}(t_1), \theta_p, H_c(t_1), H_r(t_1)),\end{aligned} \tag{3.4}$$

where θ_p denotes the parameters of the prediction. $H_c(t_1)$ and $H_r(t_1)$ represent the lengths of prediction horizons for both functions. These lengths are dynamically determined by an agent following the policy π_θ trained by the DRL algorithm. This is expressed by,

$$[H_c(t_1), H_r(t_1)] = \pi_\theta([\mathbf{p}(t_1), \Delta_L(t_1)]), \tag{3.5}$$

where $\Delta_L(t_1)$ is the AoL, which is defined as

$$\Delta_L(t) = t - U(\mathbf{p}'(t)), \tag{3.6}$$

where $\mathbf{p}'(t)$ is the last sampled pose that controls the robotic arm and $U(\cdot)$ is the generation time.

3.3.3 The Metaverse

We establish the Metaverse based on the Nvidia Omniverse Isaac Sim [110]. We decouple the two functions; i.e., real-time robotic control and rendering feedback in the Metaverse. After receiving the predicted poses $[\hat{\mathbf{p}}_r(t_1), \hat{\mathbf{p}}_c(t_1)]$ from the operator side, in time slot t_2, the control command of virtual robotic arm for two functions are computed separately, which is expressed

by,

$$\widetilde{\mathscr{T}_c}(t_3) = [\widetilde{\tau}_1(t_3), ..., \widetilde{\tau}_I(t_3)] = \mathscr{F}c(\hat{\mathbf{p}}_c(t_1), \theta c), \tag{3.7}$$

$$\widetilde{\mathscr{T}_r}(t_2) = [\widetilde{\tau}_1(t_2), ..., \widetilde{\tau}_I(t_2)] = \mathscr{F}r(\hat{\mathbf{p}}_r(t_1), \theta r), \tag{3.8}$$

where the parameters of both controlling functions are denoted by θ_r and θ_c, respectively. The target joint angle of the i-th robotic arm joint is denoted by $\widetilde{\tau}_i(t)$. Here, we propose to use the RMPFlow control algorithm to compute the joint acceleration for the virtual robotic arm in the Metaverse, which is a highly effective robotic control algorithm that combines local strategies to generate a cohesive global strategy based on the Riemannian Motion Policies (Riemannian Motion Policy (RMP)) [111,112]. Specifically, for the i-th joint of a robot, the target acceleration $\ddot{\mathbf{q}}_i(t)$ in the t-th time slot for virtual robotic arm actuation is calculated by,

$$\ddot{\mathbf{q}}_i(t) = \pi_r(\mathbf{q}(\mathbf{t}), \dot{\mathbf{q}}(\mathbf{t})) = k_p\mathscr{R}(\mathbf{q}'_i(t) - \mathbf{q}_i(t)) - k_d\dot{\mathbf{q}}_i(t), \tag{3.9}$$

where $\mathbf{q}'_i(t)$ is the target position vector, k_p represents the position gain determining the strength of configuration pull towards the target, and k_d represents the damping gain determining the amount of resistance. The robust capping function $\mathscr{R}(\mathbf{u}(t))$ is defined as,

$$\mathscr{R}(\mathbf{u}(t)) = \begin{cases} \mathbf{u}(t), & ||\mathbf{u}(t)|| < \theta \\ \theta\mathbf{u}(t)/||\mathbf{u}(t)||, & \text{otherwise} \end{cases} \tag{3.10}$$

where the position vector is defined as $\mathbf{u}(t)$, θ represents the distance at which the position correction vector is capped, and $||\cdot||$ is the Euclidean norm of a position vector.

Then, the computing results of RMPFlow for real-world robotic control are further interpolated by a trajectory smoothing algorithm [112] for narrowing the gap between sim and real, which is expressed by

$$\hat{\mathbf{q}}_i(t) = \begin{cases} \alpha \cdot \hat{\mathbf{q}}_i(t') + (1-\alpha) \cdot \mathbf{q}_i(t), t_e < t_d \\ \hat{\mathbf{q}}_i(t'), \quad \text{otherwise} \end{cases} \tag{3.11}$$

where $\mathbf{q}_i(t')$ is the last command and α represents the smooth parameter, which goes linearly from zero to one with quadratic increase. The initialization of α is $\alpha = t_e/t_d$ and the update of α is denoted by $\alpha = \alpha' \cdot \alpha'$.

For the function of rendering feedback, firstly, the virtual environment in the metaverse is simulated discretely based on the minimum simulation step T_{si}. In the j-th simulation step, the Metaverse simulates the changes in the virtual environment based on the physical law over a certain time, which is denoted by the simulation step length Δ_t. During the simulation process, the virtual robotic arm in the Metaverse can interact with the virtual objects. The simulation

process in time slot t_4 can be represented as

$$\widetilde{\mathscr{T}_s}(t_4) = \mathscr{F}_u(\widetilde{\mathscr{T}_r}(t_2), \theta_u, \theta_e), \tag{3.12}$$

where $\mathscr{F}_u(\cdot, \theta_u, \theta_e)$ represents the rendering process θ_u represents the settings and parameters for physical updates and θ_e represents the interaction with the environment.

Then, we render the image on the Metaverse side and send these frames back to the user side. The rendering process is expressed by

$$\mathbf{I}_r(t_5) = \mathscr{F}_r(\widetilde{\mathscr{T}_r}(t_4), \theta_r), \tag{3.13}$$

where $\mathscr{F}_r(\cdot, \theta_r)$ represents the rendering process and θ_r represents the settings and parameters for rendering. The rendered framed is denoted by $\mathbf{I}_r(t)$. We assume that the processing time for each image is represented by T_{im} time slots, thus the generated video refresh rate is $f_r = 1/(T_s \cdot T_{im})$. These rendering frames are then transmitted back to the user side through the communication system.

The metaverse is also responsible for interpolating the control command $\widetilde{\mathscr{T}_c}(t_3)$ generated by the RMPFlow controller before sending it to the real-world workspace. This process will smooth the trajectory, which is expressed by

$$\widetilde{\mathscr{T}_{ci}}(t_6) = [\widetilde{\tau}_1(t_6), ..., \widetilde{\tau}_I(t_6)] = \mathscr{F}ci(\widetilde{\mathscr{T}_c}(t_3), \theta ci), \tag{3.14}$$

where $\mathscr{F}_{ci}(\cdot, \theta ci)$ denotes the interpolation function, whose parameters are denoted by θ_{ci}.

3.3.4 Real-world workspace

In the real-world workspace, we employ the UR3e robotic arm [113] as the physical actuator for executing control commands generated in the Metaverse. Upon receiving the command trajectory $\widetilde{\mathscr{T}_{ci}}(t_6)$ transmitted from the Metaverse, the motion of the UR3e is planned and executed through the Robot Operating System (ROS)-based MoveIt motion planning framework [114]. Within this framework, a joint position controller is adopted to achieve precise pose-level control, ensuring that the robot's actual joint configuration follows the target trajectory computed in the Metaverse with minimal deviation.

The mapping between the Metaverse-generated trajectory and the executed motion in the real workspace can be formulated as:

$$\tilde{\mathscr{T}}rw(t_7) = \mathscr{F}rw(\widetilde{\mathscr{T}}ci(t_6), \theta rw), \tag{3.15}$$

where $\mathscr{F}rw(\cdot, \theta rw)$ denotes the motion planning and execution process in the MoveIt framework, and θ_{rw} represents the control parameters and system configurations of the real robotic

arm.

The MoveIt framework performs inverse kinematics and trajectory planning in real time, transforming the received end-effector commands into feasible joint-space trajectories while ensuring smooth and collision-free motion. By maintaining the same control logic and parameterization between the virtual and real robotic arms, the proposed framework achieves synchronized motion across both environments. This design ensures that the operator perceives consistent haptic and visual feedback, thereby preserving immersion and controllability in the Metaverse interaction loop.

To emulate practical communication scenarios, a delay d_{vr} is considered between the virtual command generation and real-world execution, corresponding to the transmission latency introduced by the network channel described in Section 3.3.5. This delay is experimentally adjusted to analyze its influence on control performance and user experience in real–virtual coordination tasks.

3.3.5 Networks

To ensure reliable communication between the operator, the Metaverse, and the physical robotic workspace, we adopt the publish–subscribe communication mechanism provided by the ROS framework [115]. This mechanism enables flexible, scalable, and modular data exchange, making it well suited for complex Metaverse applications that require real-time synchronization across heterogeneous devices and software systems. Through topic-based asynchronous communication, various modules (e.g., haptic control, visualization, and robot control) can operate independently while maintaining seamless interaction within the system.

To establish a baseline for network performance, we connect the control haptic device, the Metaverse simulation server, and the robot-control computer using a wired local area network. All components communicate via the User Datagram Protocol (UDP) protocol [116], which offers low-latency and connectionless data transmission, thereby minimizing the impact of communication delays. Under this setup, the end-to-end latency becomes negligible, allowing us to accurately evaluate the system's intrinsic performance.

To further analyze the influence of network latency on user experience and control stability, we artificially introduce communication delays using the Linux Traffic Control (TC) tool [117]. This allows precise simulation of various delay conditions that mimic long-distance or unstable network environments. By systematically adjusting the delay parameters, we can assess the robustness and adaptability of the proposed framework under different network conditions, which is essential for real-world Metaverse–robot interaction scenarios.

In addition, the ROS communication architecture is compatible with a wide range of robotic middleware and simulation platforms, such as NVIDIA Isaac Sim, facilitating seamless integration between real and virtual systems [110]. This flexibility ensures that the proposed network setup can be easily extended to multi-robot or distributed Metaverse environments in future re-

search.

3.4 Problem Formulation

3.4.1 Deep Reinforcement Learning

To achieve synchronization between the operator, the Metaverse, and the real-world robotic system, we formulate an optimization problem that minimizes the root mean square error (RMSE) between the poses of the input device and both robotic arms by adaptively optimizing the prediction horizons $H_c(t)$ and $H_r(t)$. We employ the PPO algorithm as the baseline due to its high sampling efficiency and stable performance in continuous control tasks [118]. To address the challenge of human-in-the-loop data collection, a two-stage DRL training scheme is adopted: the agent is first trained using recorded pose sequences and subsequently fine-tuned through human-in-the-loop interaction.

State: In each time slot t, the system state $\mathbf{s}_t$ is represented as a concatenation of the latest input pose $\mathbf{p}(t)$ and the AoL $\Delta_L(t)$:

$$\mathbf{s}_t = \left[\mathbf{s}_t^{[1]}, \mathbf{s}_t^{[2]}\right] = \left[\mathbf{p}(t), \Delta_L(t)\right], \tag{3.16}$$

where $\mathbf{s}_t^{[1]} \in [-1,1]^{1\times 7}$ and $\mathbf{s}_t^{[2]} \in \mathbb{N}$.

Action: The action at time t determines the prediction horizon for both the real-world robotic control and rendering feedback processes:

$$\mathbf{a}_t = \left[a_t^{[1]}, a_t^{[2]}\right] = \left[H_r(t), H_c(t)\right], \tag{3.17}$$

where $\mathbf{a}_t \in \mathbb{N}^{1\times 2}$.

Reward: Given $(\mathbf{s}_t, \mathbf{a}_t)$, the instantaneous reward is defined as a weighted sum of the RMSEs in position and orientation among the input device, the virtual model, and the real-world robot:

$$\begin{aligned} r(\mathbf{s}_t, \mathbf{a}_t) = \; & w_1 \times (\mathbf{RMSE_p}(\mathbf{o}(t), \mathbf{m}(t))) \\ & + w_2 \times (\mathbf{RMSE_p}(\mathbf{m}(t), \mathbf{r}(t))) \\ & + w_3 \times (\mathbf{RMSE_o}(\mathbf{o}(t), \mathbf{m}(t))) \\ & + w_4 \times (\mathbf{RMSE_o}(\mathbf{m}(t), \mathbf{r}(t))), \end{aligned} \tag{3.18}$$

where $\mathrm{RMSE}_p(\cdot,\cdot)$ and $\mathrm{RMSE}_o(\cdot,\cdot)$ denote the RMSEs of position and orientation, respectively. Here, $\mathbf{o}(t)$, $\mathbf{m}(t)$, and $\mathbf{r}(t)$ correspond to the poses of the input device, the virtual model, and the real-world robotic arm.

3.4.2 Problem Formulation

The control policy $\pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)$ defines a probabilistic mapping from the current system state $\mathbf{s}_t$ to the action $\mathbf{a}_t$, where θ represents the trainable parameters of the policy network. The policy aims to dynamically adjust the prediction horizons of the control and rendering loops, thereby minimizing discrepancies among the user's operation, the virtual model, and the real robotic arm under time-varying network conditions.

Following the policy π_θ, the expected cumulative reward over time is defined as

$$R^{\pi_\theta} = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t)\right], \tag{3.19}$$

where $\gamma \in [0,1)$ is the discount factor that balances the influence of immediate and future rewards. This reward formulation encourages the agent to not only minimize the instantaneous RMSE between trajectories but also maintain long-term stability and synchronization across the Metaverse–robot interaction loop.

The optimization objective is to find an optimal policy π_θ^* that maximizes the expected long-term reward, i.e.,

$$\pi_\theta^* = \arg\max_\theta Q^{\pi_\theta}(\mathbf{s}_t, \mathbf{a}_t), \tag{3.20}$$

$$Q^{\pi_\theta}(\mathbf{s}_t, \mathbf{a}_t) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t) \,\middle|\, \mathbf{s}_0 = \mathbf{s},\, \mathbf{a}_0 = \mathbf{a},\, \pi_\theta\right], \tag{3.20a}$$

where $Q^{\pi_\theta}(\mathbf{s}_t, \mathbf{a}_t)$ denotes the expected return starting from $(\mathbf{s}_t, \mathbf{a}_t)$ and following policy π_θ thereafter.

To ensure stable prediction and control within feasible physical and communication limits, the prediction horizons are constrained by

$$0 < P_r(t) < H_r, \tag{3.20b}$$

$$0 < P_c(t) < H_c, \tag{3.20c}$$

where H_r and H_c denote the upper bounds of the prediction horizons for the rendering and control loops, respectively. These bounds prevent overestimation of the prediction range, which could otherwise lead to control instability or misalignment between virtual and real trajectories.

In summary, the formulated problem seeks an optimal adaptive policy π_θ^* that minimizes trajectory errors across multiple domains by dynamically adjusting $H_c(t)$ and $H_r(t)$ according to the system's perceived latency and state, ensuring synchronized behavior across the operator, the Metaverse, and the physical workspace.

3.5 Prototype Design and Results

To validate the proposed co-design framework, a real–virtual hybrid prototype system is established, as shown in Fig. 3.3. The system integrates the operator side, the Metaverse, and the real-world workspace into a closed interaction loop, enabling synchronized control, prediction, and feedback for real-time human–robot collaboration.

On the operator side, a 3D Touch haptic device [107] is employed for motion input and feedback. The device captures the operator's 6-DoF hand pose, including both position and orientation, at a high sampling rate. To mitigate perception and transmission delays, an ARMA-based predictor is implemented to estimate future poses for both robotic control and visual rendering tasks. The prediction horizon is dynamically adjusted by a deep reinforcement learning agent following a policy π_θ, allowing the system to adapt to variable latency conditions.

The Metaverse environment is constructed on the NVIDIA Omniverse Isaac Sim platform [110], which supports both robotic control simulation and high-fidelity rendering. Two functional modules are decoupled in the virtual domain: real-time robotic control and rendering feedback. The virtual robotic arm is actuated using the RMPFlow algorithm [111, 112], which generates smooth and dynamically consistent joint accelerations. To bridge the simulation-to-reality gap, a trajectory interpolation mechanism is applied for smoothing the computed trajectories before dispatching them to the real-world controller. The Metaverse also simulates physics-based interactions and renders visual frames, which are streamed back to the operator to ensure immersive and coherent feedback.

In the real-world workspace, a UR3e robotic arm [113] is utilized as the physical actuator to execute the control trajectories generated in the Metaverse. The system is developed on the ROS and MoveIt motion planning frameworks [114], enabling real-time inverse kinematics, trajectory optimization, and collision-free control. The same control parameters and logic are maintained between the virtual and physical robotic arms, ensuring motion consistency across domains. The robot operates under a joint position controller for precise tracking of target poses, achieving minimal deviation between commanded and executed motion.

All modules communicate through a ROS-based publish–subscribe architecture [115] using UDP transport [116], ensuring low-latency, modular, and scalable data exchange. The system is deployed over a wired LAN to minimize baseline delay, while additional latency is artificially introduced via the Linux TC tool [117] to evaluate the framework's performance under variable network conditions.

This prototype provides an integrated experimental platform for evaluating both the predictive accuracy of the motion model and the adaptive control performance of the proposed reinforcement learning framework in realistic human–robot interaction scenarios. Based on the prototype, we first evaluate the performance of prediction model and then evaluate the effectiveness of the proposed framework.

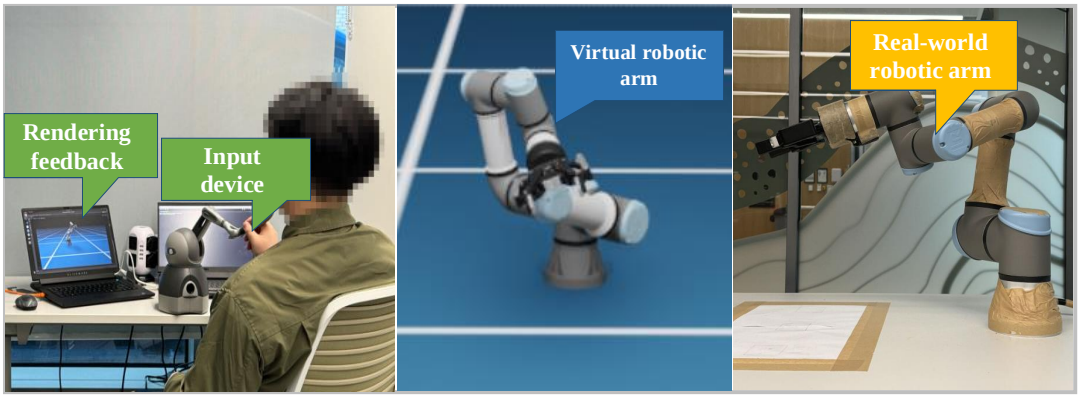

Figure 3.3: Illustration of our prototype system. 1)Left photo: Operator controls the robotic arm with the rendered feedback in a monitor, 2) Central photo: Virtual model of the robotic arm in the Metaverse, 3) Right photo: A real-world robotic arm is controlled by the Metaverse.

3.5.1 Precision of prediction model

As shown in Fig. 3.4, the prediction performance of all models declines as the interpolation interval increases. This trend is reflected by the rising RMSE between the predicted trajectory and the baseline, indicating that a longer prediction horizon introduces higher uncertainty and error accumulation. Specifically, our proposed method demonstrates strong prediction accuracy within short-term horizons (less than 180 ms), maintaining an RMSE below 0.005 m. Beyond this range, the predictive accuracy gradually deteriorates, consistent with the expectation that longer-term forecasts are inherently more challenging due to dynamic variations and compounding estimation errors.

Table 3.1 quantitatively compares the performance of different prediction models, including the proposed approach, the ARMA model, the Informer-based deep learning model, and a traditional autoregressive (AR) model. The results show that our proposed method consistently achieves the lowest mean RMSE and variance across both position and orientation predictions. For position estimation, the proposed model achieves an average RMSE of 0.00100 m with a small variance of 5.264×10^{-5}, outperforming both Informer (0.00842 m) and AR (0.00174 m). Similarly, in orientation prediction, our approach attains an average RMSE of 0.00101 with variance 5.650×10^{-5}, again surpassing the alternatives by a significant margin.

These results confirm that the proposed prediction model effectively captures both spatial and temporal dependencies, enabling accurate short-term motion estimation. In contrast, the Informer model exhibits larger errors and higher variance, suggesting limited generalization under the same experimental setup. Overall, the findings demonstrate the robustness and reliability of our method for real-time predictive modeling, particularly in short-horizon scenarios critical for responsive human-robot interaction systems.

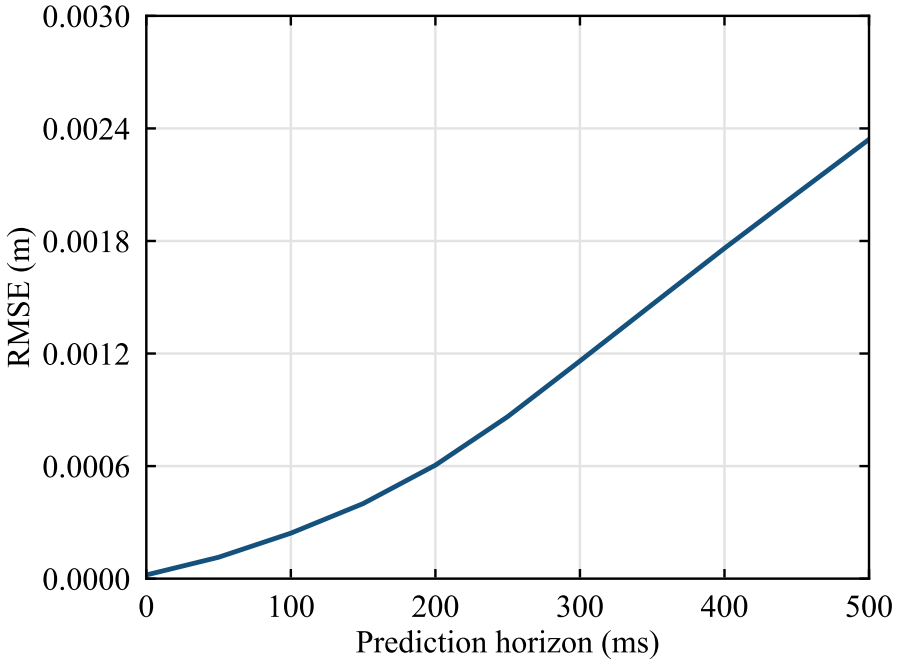

Figure 3.4: Average RMSE of different prediction horizons for ARMA model

Table 3.1: Evaluation of the Prediction Model

Average RMSE		ARMA	Informer	AR
Position (m)	Mean	0.00100	0.00842	0.00174
	Variance	5.264×10^{-5}	0.00343	1.108×10^{-4}
Orientation	Mean	0.00101	0.01031	0.00201
	Variance	5.650×10^{-5}	0.00638	0.912×10^{-5}

3.5.2 Evaluation of the proposed framework

To assess the decision-making and control performance, we further evaluate the proposed PPO-based reinforcement learning (RL) algorithm. The training process is divided into two stages: a data-driven baseline training phase and a human-in-the-loop fine-tuning phase.

In the first stage, the PPO algorithm is trained for 150 episodes, repeated over 10 independent runs to ensure statistical consistency. The communication latency is modeled as a stochastic variable with a mean of 75 ms and a standard deviation of 12.5 ms, representing realistic network conditions. The reward function weights are configured uniformly as $w_1 = w_2 = w_3 = w_4 = -1$ to balance accuracy and stability objectives.

As shown in Fig. 3.5, the training reward exhibits an initial fluctuation during the first 30 episodes, which gradually stabilizes as the policy converges. After approximately 60 episodes, the model achieves convergence with an average RMSE of 0.0443 m, indicating reliable baseline performance.

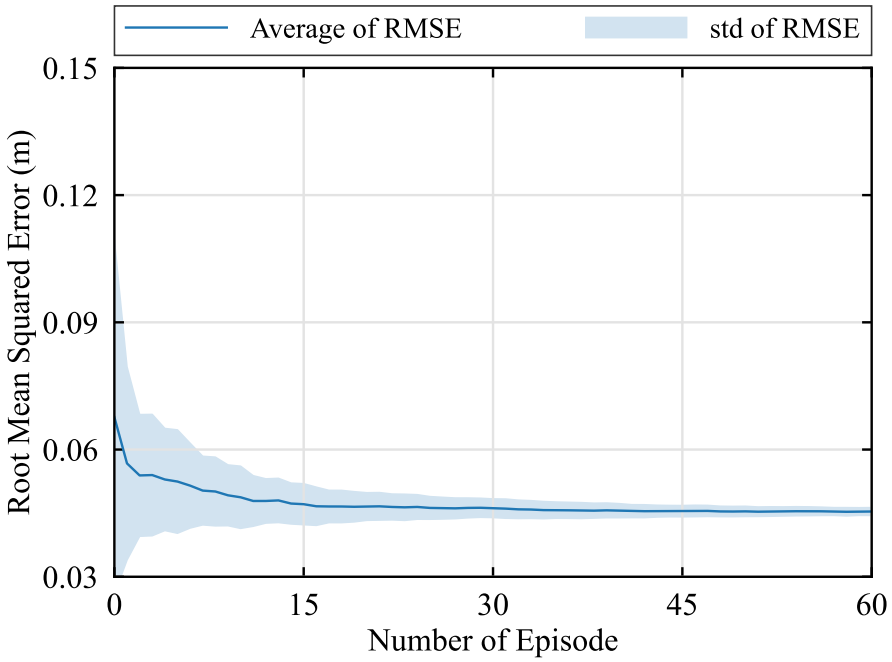

Figure 3.5: Average RMSE in each training episode for Data-based RL Training Progress.

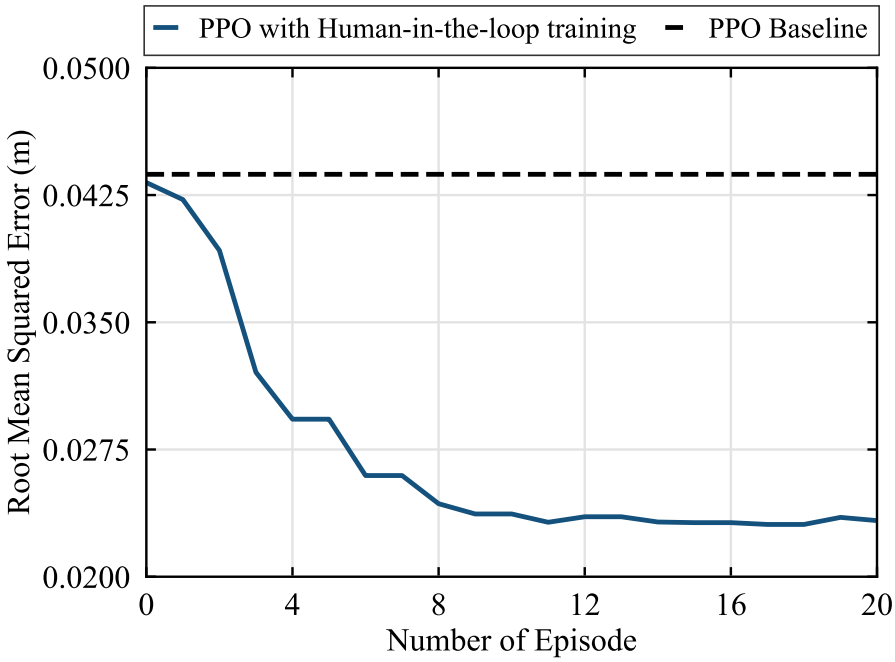

Figure 3.6: Average RMSE in each training episode for Human-in-the-Loop RL Training Progress.

To further enhance adaptability and robustness, a human-in-the-loop training phase is conducted using the pre-trained PPO baseline model as initialization. During this phase, human feedback is incorporated across an additional 20 episodes, allowing the agent to refine its policy based on interactive corrections. As illustrated in Fig. 3.6, the human-in-the-loop phase leads to a notable improvement, reducing the average RMSE to 0.0241 m. This corresponds to an 83.82% performance gain relative to the baseline model, validating the effectiveness of integrating human guidance into the RL training process.

Overall, these results highlight the capability of the proposed framework to achieve both accurate motion prediction and adaptive control optimization. The combination of the data-driven ARMA model and human-guided RL training ensures robustness against uncertainty and enhances real-time responsiveness in dynamic human-robot interaction environments.

3.6 Conclusion

In this work, we established a human-in-the-loop cross-system design framework to minimize the motion-to-photon latency on the modeling error of a real-world robotic arm and its matched virtual model in the Metaverse. We built a framework structure with separated display/control functions and human-in-the-loop DRL training process to improve the performance of the framework. The results our proposed method can significantly reduced 1) the MTP latency between human motion and rendering feedback and 2) the RMSE between human motion and real-world remote devices.

Chapter 4

Meta-learning in Co-design CPS Framework in Multi-task Situations

4.1 Introduction

CPS represents a transformative integration of computational elements, networked communication, and physical processes. These systems are designed to interact seamlessly with the physical world, enabling real-time monitoring, control, and automation in various domains, including manufacturing, healthcare, transportation, and energy systems [1]. A key component of CPS is the concept of “DTs", which serves as precise virtual replicas of physical entities [3]. DTs enable the simulation, analysis, and optimization of physical processes, leading to improved decision-making and operational efficiency. For instance, DTs and robotic operators for nuclear power plants can reduce costs and simplify the planning and implementation of decommissioning [20]. By merging DT with healthcare, a new, efficient approach for providing accurate and swift services, especially for elderly care is emerging [2].

Despite the advanced capabilities of CPS, numerous challenges remain in achieving their full potential. One of the primary issues is the computational demand associated with processing vast amounts of data from diverse sensors and devices in real time. Additionally, the integration of Artificial Intelligence (AI) within CPS for tasks such as anomaly detection, predictive maintenance, and autonomous decision-making further amplifies the demand for computational power [6]. Furthermore, the communication infrastructure is crucial for ensuring the reliable and timely exchange of data between the physical and computational elements. The communication requirements in CPS are particularly stringent for applications involving real-time control and feedback loops, where delays can have serious consequences [6]. In this context, the performance metrics established for CPS communication systems often include stringent KPIs such as ultra-low latency, high reliability, and sufficient bandwidth to support high-resolution data streams from multiple sensors simultaneously [43]. For instance, fine-grained KPIs for video streaming in CPS have been defined in [43], suggesting that data rates of 1 Gbps (for smooth

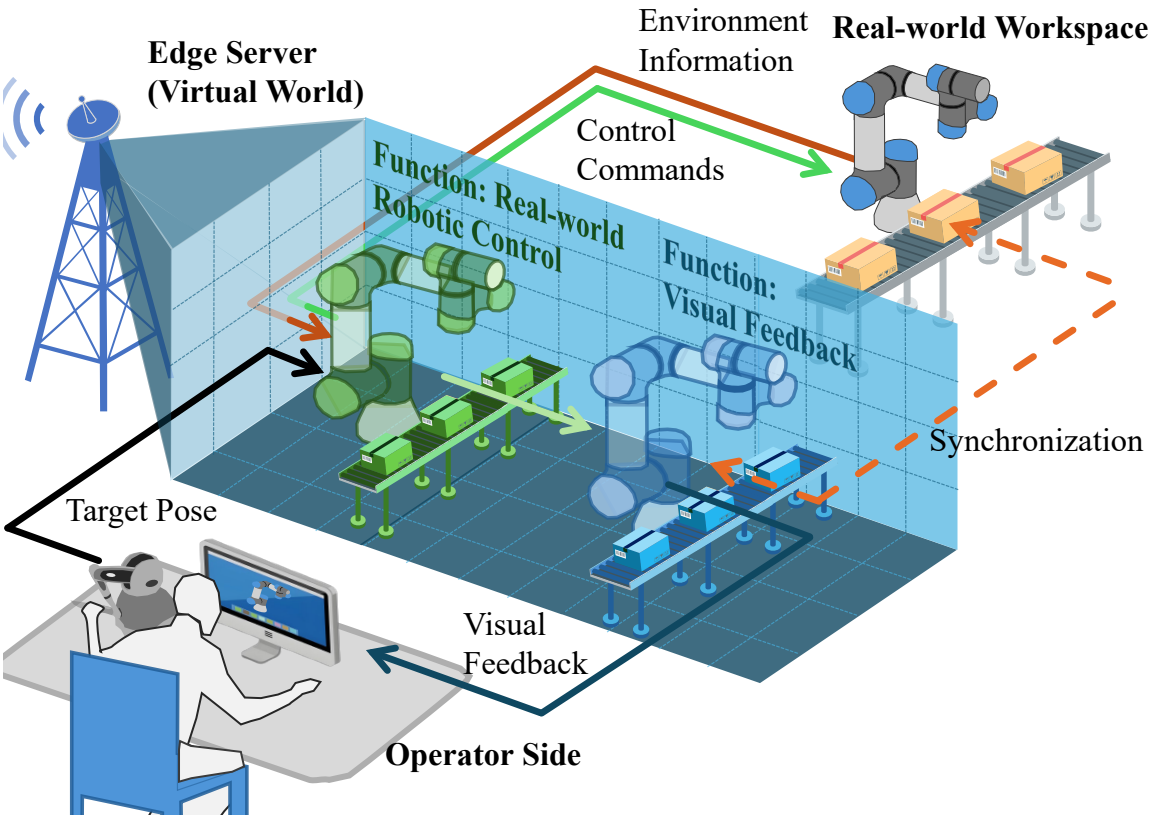

Figure 4.1: Proposed task-oriented cross-system design framework.

playback) or 2.35 Gbps (for interactive experiences) are required to support 8K resolution, 120 FPS, and 360-degree visual field video streaming [8]. The Tactile Internet prioritizes ultra-low latency to ensure that tactile sensations and control commands are transmitted almost instantaneously, with latency requirements often below 1 ms [101], as humans are more sensitive to delays in tactile sensations than to audio or visual ones. This is essential for applications where even slight delays in feedback can lead to errors or degraded performance, particularly in scenarios involving human-machine interaction or fine motor control [119]. However, existing communication infrastructures are often insufficient to meet these stringent requirements, necessitating further advancements in network technologies and protocols [5].

Furthermore, another significant challenge in CPS is the intricate complexity of HITL systems, where seamless interaction between human operators and automated systems is critical yet profoundly challenging. In these scenarios, delays in control commands and feedback loops can significantly affect system performance and safety [102]. For instance, in smart grids, where human operators manage and control critical infrastructure, latency in decision-making can lead to suboptimal responses to dynamic conditions such as load changes or fault detection [4]. While research on the effects of latency in teleoperation provides a foundational understanding [9], the intricacies of human behavior and interaction with complex CPS environments require more sophisticated models and simulations to predict and mitigate these effects. This presents a sig-

nificant challenge for deploying AI-driven control systems that can adapt to human behavior and ensure seamless, real-time operation.

Recently, emerging task-oriented communication is well-suited for handling the rich data sources and diverse complex tasks found in CPS, making it a promising candidate for future communication paradigms [60]. From a machine's viewpoint, achieving perfectly accurate reconstruction is considered sufficient but not essential for completing tasks, as it often includes redundant information. The primary goal of task-oriented communications is to enable the successful completion of subsequent tasks. This approach is particularly beneficial in CPS, such as industrial factories typically demanding URLLC environments, since it can relax stringent reliability and latency requirements to reduce communication overhead [120]. However, several challenges still need to be addressed: 1) Defining Task Success: Success criteria can vary widely across different CPS applications, making it difficult to establish a universal standard for task completion. 2) Verification of Robustness: Ensuring that task-oriented communication systems are reliable under diverse conditions and failures while still achieving task goals requires thorough testing and validation. 3) Generalisability: Developing task-oriented communication strategies that are both effective and adaptable across various CPS applications and environments remains an ongoing challenge.

4.2 Contributions

In this chapter, we aim to solve the following problems: 1) How to eliminate the effects of time delay among different subsystem components to achieve real-time interactions? 2) How to design strategy leverage user behavior to address real-time HITL interactions, while ensuring the algorithm's generalizability and robustness across various robotic arm operating behaviors? 3) How to build the prototype to verify the effectiveness of the proposed algorithm to ensure a reliable and adaptive CPS operation?

The main contributions of this paper are summarized as follows:

- We established a task-oriented cross-system design framework for real-time interactions between human controllers and remote devices in CPS, where sensing, communication, prediction, control, and rendering are jointly considered. Specifically, we decouple the function of the virtual world for two: 1) proactive rendering form the visual feedback and 2) generating control commands to the real-world robotic arm.

- We developed a Human-In-The-Loop Model-Agnostic Meta-Learning (HITL-MAML) algorithm that dynamically changes the prediction horizons for 1) visual feedback and 2) real-world robotic control. we proposed the two-stage training process, where the MAML model is first trained on the collected dataset, followed by the HITL online learning.

- We built a complete prototype of a CPS, which consists of the input and visual display devices on the operator side, the virtual world deployed on the edge server, and a robotic arm in the real-world workspace. We validated the algorithm on the prototype, varying the communication delay for multiple experiments, and the results proved the effectiveness of the algorithm and outperformed multiple baselines.

4.3 Task-Oriented Cross-System Design

In this section, we propose a task-oriented cross-system design framework. We introduce two functions of the framework, different sub-modules, the time flow chart, and the time delay composition.

4.3.1 System Overview

The framework, as illustrated in Fig. 4.1, is composed of three primary components: 1) an input device for issuing control commands and a visual feedback system located at the operator's side, 2) a virtual world containing a DT of a robotic arm deployed on an edge server, and 3) a remote physical workspace deployed with a real-world robotic arm. These components are spatially separated and communicate via the wireless network. First, the operator views the visual display of the virtual environment including the DT of the robotic arm. Based on the visual display, the operator manipulates the input device and sends the control commands to the DT of the robotic arm. Then, the DT of the robotic arm syntheses the information: 1) the control commands from the operator 2) the simulation information of the virtual world, and 3) the feedback from the real-world world to generate control instructions for the real-robotic arm. Next, these commands are then transmitted to the real robot arm in the remote workspace, enabling it to perform the required actions and interact with the real world. The results of these interactions are further feed back to the virtual world via wireless transmission.

However, due to time delays introduced by the subsystems including sampling, computation, control, communication and rendering, discrepancies arise between the perceived trajectories of the DT of the robotic arm, the real-world robotic arm, and the operator's input commands. Even worse, in this human-in-the-loop cyber-physical system, time delays in visual feedback can lead to delays in the operator's control, and delays in control can further affect the generation of visual feedback, which finally significantly affects the robotic arm's performance of the task. To tackle this problem, the system predicts the operator's commands and dynamically changes the prediction horizons in real time. This prediction serves two main purposes: (1) to compensate for the delay between the operator's motion and the visual feedback rendering of the DT, and (2) to minimize the latency between the operator's control inputs and the actions of the real-world robotic arm in the real-world environment.

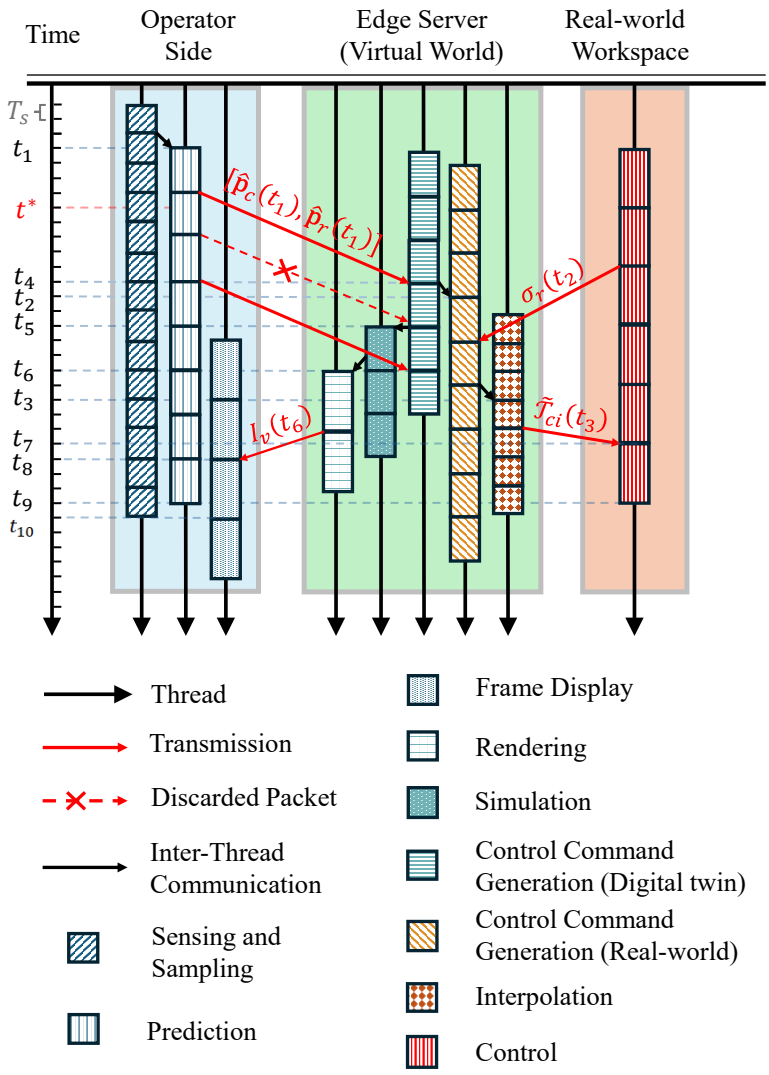

Figure 4.2: Illustration of the information flow.

4.3.2 Information Flow

As shown in Fig. 4.2, time is divided into time slots. We modified the entire working process queue as $M/M/1/2*$ [121]. In this case, only one packet is kept in the buffer waiting for processing, and packets waiting for processing are replaced upon arrival of a more up-to-date packet. No packets are blocked from entering the queue. If a new packet arrives while the system is full, the packet waiting is discarded. The proposed framework comprises two information flows served for two functions: 1) real-world robotic control and 2) visual feedback. The function of real-world robotic control, encompasses all processes from sensing the movement of the operator through the data transmission, virtual world data processing, and real-world robotic control. During the process, the operator operates the input device and generates a series of poses $\{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_i\}$ which are sampled by the built-in sensor. Next, In the t_1-th time slot, an agent decides the length of the prediction horizon. followed by a predictor generates the pre-

dicted pose for the visual feedback $\hat{\mathbf{p}}_v(t_1)$ and real-world robotic control $\hat{\mathbf{p}}_r(t_1)$. Then, $\hat{\mathbf{p}}_v(t_1)$ and $\hat{\mathbf{p}}_r(t_1)$ are transmitted to the CPSs via networks. After that, in the t_2-th time slot, with the feedback of the environment, the received predicted poses $\hat{\mathbf{p}}_r(t_2)$ is processed to generate a control command in the CPSs, and the virtual world updates based on the corresponding command and the feedback of the environment. In the t_3-th time slot, to maintain smoothing motion trajectories in real-world workspace, the command is further interpolated to generate a joint position control command and transmitted via a communication channel for actuation. Finally, in the t_8-th time slot, the command is received by the real-world robotic arm and further transformed into low-level commands that can be directly executed. Considering the completed real-world robotic control workflow, we define the E2E delay for the real-world robotic control by:

$$T_r = t_9 - t_1. \tag{4.1}$$

On the other hand, the function of the visual feedback, encompasses all processes from the sensing of movement of the operator, data transmission, CPS data processing, rendering, and feedback display. Specifically, the processes before the step of the CPS process are identical to those of real-world robotic control. In addition, in the t_4-th time slot, the received predicted poses $\hat{\mathbf{p}}_v(t_4)$ is processed to generate a control command, After that, in the t_5-th time slot, the update in the virtual world is simulated based on the corresponding command and the feedback of the environment. Next, in the t_6-th time slot, the virtual environment is rendered. Finally, in the t_7-th time slot, the rendered image $\mathbf{I}_v(t)$ is streamed back to the user side. Similarly, we define the E2E delay for the visual feedback by:

$$T_v = t_{10} - t_1. \tag{4.2}$$

4.3.3 System Components

Operator Side

On the operator side, based on the position of the robot arm in the visual display, the operator controls the input devices to issue control commands

$$\mathbf{p}(t) = [l_{\mathrm{x}}(t), l_{\mathrm{y}}(t), l_{\mathrm{z}}(t), q_{\mathrm{x}}(t), q_{\mathrm{y}}(t), q_{\mathrm{z}}(t), q_{\mathrm{w}}(t)], \tag{4.3}$$

where $[l_{\mathrm{x}}(t), l_{\mathrm{y}}(t), l_{\mathrm{z}}(t)]$ is the desired position in the Cartesian coordinate system, and $[q_{\mathrm{x}}(t), q_{\mathrm{y}}(t), q_{\mathrm{z}}(t), q_{\mathrm{w}}(t)]$ is the desired orientation in quaternion [108]. To compensate the time delay latent in all the subsystems, we introduce the predictor to predict the operator's pose which is used to 1) proactive rendering of virtual environments and 2) preemptive generation of control commands for real robotic arm. Specifically, in the t_1-th time slot, given historical poses $\mathscr{P}(t_1) = [\mathbf{p}(t_1 - W_p), \mathbf{p}(t_1 - W_p + 1), ..., \mathbf{p}(t_1)]$, the predictor predicts the pose for the real-world

robotic arm control $\hat{\mathbf{p}}_r(t_1)$, and the proactive rendering $\hat{\mathbf{p}}_v(t_1)$, respectively. This is expressed by

$$[\hat{\mathbf{p}}_r(t_1),\hat{\mathbf{p}}_v(t_1)] = \mathscr{F}_p(\mathscr{P}(t_1),\theta_p,H_r(t_1),H_v(t_1)), \tag{4.4}$$

where θ_p denotes the parameters of the prediction. $H_c(t_1)$ and $H_r(t_1)$ represent the lengths of prediction horizons. To illustrate, we propose to use ARMA for its low computational complexity, robustness, and no need for pre-training [109], but the framework works equally well with other predictors. Specifically, in the t_1-th time slot, The one-step predicted value $\hat{\mathbf{p}}(t_1)$ is expressed as a combination of autoregressive (AR) and moving average (MA) components, formulated as follows:

$$\hat{\mathbf{p}}(t_1) = c + \sum_{a=1}^{p} \phi_a \mathbf{p}(t_1 - a) + \sum_{b=1}^{q} \theta_b \varepsilon(t_1 - b), \tag{4.5}$$

where $\hat{\mathbf{p}}(t_1)$ is the predicted value at time t_1, c is the constant term that allows the model to fit time series data with a non-zero mean, ϕ_a and θ_b are the coefficients of the AR and MA components, respectively, and $\varepsilon(t_1 - b)$ denotes the prediction errors. The AR term $\sum_{a=1}^{p} \phi_a \mathbf{p}(t_1 - a)$ captures the influence of the p-lagged past values of $\mathbf{p}(t_1)$, while the MA term $\sum_{b=1}^{q} \theta_b \varepsilon(t_1 - b)$ models the effects of past forecast errors up to a lag of q. By recursively applying the ARMA model over H steps, we create a sequence of future predictions based on the current and previous observations. In addition, to dynamically capturing the time delay characters in difference subsystem, we introduce a DRL agent adaptive decides the prediction length $H_r(t_1)$ and $H_v(t_1)$, which is expressed by

$$[H_r(t_1),H_v(t_1)] = \pi_\theta([\mathbf{p}(t_1),T_r(t_1),T_v(t_1)]), \tag{4.6}$$

where $T_r(t_1),T_v(t_1)$ are the measured E2E delay defined by (4.1) and (4.2).

Virtual World

We deploy the virtual world on an edge server to generate two key outputs: (1) control commands for a real-world robotic arm and (2) visual feedback for the visual display. Specifically, in the t_2-th time slot, the DT of the robotic arm in the virtual world generates the control commands by synthesizing the information: 1) the received predicted control commands $[\hat{\mathbf{p}}_r(t_1),\hat{\mathbf{p}}_v(t_1)]$ from the operator generated in the t_1-th time slot 2) the last update of the simulation information $\sigma_v(t_2)$ of the virtual world, and 3) the feedback $\sigma_r(t_2)$ from the real-world world to generate control instructions for the real-robotic arm. This is expressed by,

$$\widetilde{\mathscr{T}_r}(t_2) = [\widetilde{\tau}_1(t_2),...,\widetilde{\tau}_I(t_2)] = \mathscr{F}_r(\hat{\mathbf{p}}_c(t_2),\sigma_v(t_2),\sigma_r(t_2),\theta_c), \tag{4.7}$$

where θ_c represents the parameters for the respective control functions. The angle value of the i-th robotic arm joint, $i = 1, 2, ..., I$, in the t_2-th time slot is denoted by $\widetilde{\tau}_i(t_2)$.

To illustrate, we propose to use the RMPFlow control algorithm, which is an effective method that merges local control strategies to form a cohesive global policy based on Riemannian Motion Policies (RMPs) [111]. Specifically, for the i-th joint of the robotic arm, the desired joint acceleration $\ddot{\mathbf{q}}_i(t_2)$ in the t_2 th time slot is calculated by

$$\ddot{\mathbf{q}}_i(t_2) = k_p \cdot \mathscr{R}(\mathbf{q}'_i(t_2) - \mathbf{q}_i(t_2)) - k_d \cdot \dot{\mathbf{q}}_i(t_2), \tag{4.8}$$

where $\mathbf{q}_i(t_2)$ is the current angle value, $\mathbf{q}'_i(t_2)$ represents the target position vector, k_p is the position gain that determines the strength of the pull towards the target configuration, and k_d is the damping gain, which provides resistance to movement. The robust capping function $\mathscr{R}(\mathbf{u}(t_2))$ is defined as

$$\mathscr{R}(\mathbf{u}(t_2)) = \begin{cases} \mathbf{u}(t_2), & \text{if } ||\mathbf{u}(t_2)|| < \theta_{th} \\ \theta_{th} \cdot \frac{\mathbf{u}(t_2)}{||\mathbf{u}(t_2)||}, & \text{otherwise,} \end{cases} \tag{4.9}$$

where θ_{th} is a threshold that caps the correction vector, and $||\cdot||$ represents the Euclidean norm. Further refinement of the control commands to adapt the sampling rate of the real-world robotic arm is also necessary,

$$\widetilde{\mathscr{T}_{ri}}(t_3) = [\widetilde{\tau}_1(t_3), ..., \widetilde{\tau}_I(t_3)] = \mathscr{F}_{ri}(\widetilde{\mathscr{T}_r}(t_3), \theta_{ri}), \tag{4.10}$$

where $\mathscr{F}_{ri}(\cdot, \theta_{ri})$ is the smoothing function, and θ_{ri} denotes the parameters for smoothing the control command. To illustrate, we apply the linear interpolation based trajectory smoothing algorithm [112],

$$\hat{\mathbf{q}}_i(t_3) = \begin{cases} \alpha \cdot \hat{\mathbf{q}}_i(t_3 - T_{co}) + (1 - \alpha_s) \cdot \mathbf{q}_i(t_3), & t_e < t_d \\ \hat{\mathbf{q}}_i(t_3 - \mathrm{T}_{co}), & \text{otherwise,} \end{cases} \tag{4.11}$$

where T_{co} is the time interval between the last generated interpolated command $\hat{\mathbf{q}}_i(t_3 - T_{co})$ and $\mathbf{q}_i(t_3)$, and α_s is a smoothing parameter that transitions from 0 to 1 with a quadratic growth rate. The initial value of α_s is given by $\alpha_s = t_e/t_d$, and it is updated as $\alpha'_s = \alpha_s^2$.

In addition to generating the control commands for the DT of the robotic arm, vitual world generates the rendering frame for the visual feedback. Similar to (4.7), in the t_4-th time slot, the process to generate control instructions for the DT is denoted by

$$\widetilde{\mathscr{T}_v}(t_4) = [\widetilde{\tau}_1(t_4), ..., \widetilde{\tau}_I(t_4)] = \mathscr{F}_v(\hat{\mathbf{p}}_r(t_4), \sigma_v(t_4), \sigma_r(t_4), \theta_v), \tag{4.12}$$

where θ_v represents the parameters for the respective visual feedback functions. The virtual

world also updates itself on the edge server T_{si} time steps by applying physical laws. In each simulation step, the virtual robotic arm interacts with virtual objects, and physical process simulations including collision detection are also updated here. We denote the simulation time length in each time step as Δ_t. Thus, at time slot t_5, the simulation function $\mathscr{F}_u$ can be expressed as

$$\widetilde{\mathscr{T}_s}(t_5) = \mathscr{F}_u(\widetilde{\mathscr{T}_v}(t_5), \theta_u, \theta_e), \tag{4.13}$$

where θ_u refers to parameters associated with the physical update process and θ_e is the parameter of the interaction between the robotic arm and the environment. Next, in the t_6-time slot, the edge server renders the simulation result at the edge server. The rendering function $\mathscr{F}_r$ is expressed by

$$\mathbf{I}_v(t_6) = \mathscr{F}_{re}(\widetilde{\mathscr{T}_s}(t_6), \theta_{re}), \tag{4.14}$$

where θ_{re} is the parameter of rendering. The resulting rendered frame is denoted as $\mathbf{I}_v(t_6)$. We denote the rendering time for each frame by T_{im}, thus, the frame rate for the generated video is calculated by $f_r = \frac{1}{T_{im}}$. Then, $\mathbf{I}_v(t_6)$ is transmitted back to the operator's side via a communications network, received in the t_10-th time slot and displayed in the t_10-th time slot.

Real-World Workspace

In real-world workspace, the received motion control commands are further transformed into low-level commands that can be directly executed by the hardware by Application Programming Interfaces (APIs) [113]. In the t_7-th time slot, the joint trajectory of the real-world robotic arm to be executed is expressed by

$$\widetilde{\mathscr{T}_{rw}}(t_7) = \mathscr{F}_{rw}(\widetilde{\mathscr{T}_{ri}}(t_7), \theta_{rw}), \tag{4.15}$$

where $\mathscr{F}_{rw}(\cdot, \theta_{rw})$ represents the API, and θ_{rw} denotes the corresponding parameters.

To illustrate, we propose to use the PID control algorithm for its common ways to implement the control API [114]. Specifically, for the i-th joint of a robot, the joint position command in the t_7-th time slot is calculated by

$$u_i(t_7) = K_p \tau_i(t_7) + K_i \int_0^{t_7} \tau_i(\dot{t}) \mathbf{d}\dot{t} + K_d \frac{\mathbf{d}}{\mathbf{d}t} \tau_i(t_7), \tag{4.16}$$

where K_p, K_i and K_d, all non-negative, denote the coefficients for the proportional, integral, and derivative terms, respectively, $\dot{t}$ represents the differentiation of t. The control is finished in the t_9-th time slot. Various sensors are also deployed in the real workspace to monitor the completion of the real-world robotic arm's tasks and feed the results back to the virtual world on the edge server.

4.3.4 Task-Oriented KPIs

The success of the real-time interaction task depends on the synchronization between the robot arm trajectory and the user-generated control commands. Out-of-sync can lead to 1) operator dizziness and nausea leading to deformation of the interaction and thus task failure 2) significant lag leading to repetitive behaviors and misbehavior. To achieve the task, we propose to use the KPIs:

- 1) MTP latency, defined as the time delay between the operators' physical motion (e.g., moving the input device) and the corresponding visual update displayed on the screen.
- 2) RMSE between operator motions and real-world robotic arm.

Due to inconsistencies between the workspace of the robotic arm and the input devices, it is necessary to map the input values to the robotic arm's coordinate system. We represent the end effector's pose using Cartesian coordinates and quaternions, which can be reconstructed in a 4×4 homogeneous transformation matrix. Thus, in the t-the time slot, the transformed pose $\tilde{\mathbf{p}}_i(t)$ is expressed as $[l_{\mathrm{x,i}}(t), l_{\mathrm{y,i}}(t), l_{\mathrm{z,i}}(t), q_{\mathrm{x,i}}(t), q_{\mathrm{y,i}}(t), q_{\mathrm{z,i}}(t), q_{\mathrm{w,i}}(t)]$. Specifically, the position $[l_{\mathrm{x,i}}(t), l_{\mathrm{y,i}}(t), l_{\mathrm{z,i}}(t)]$ is calculated by

$$[l_{\mathrm{x,i}}(t), l_{\mathrm{y,i}}(t), l_{\mathrm{z,i}}(t)] = \mathbf{s} \cdot \mathbf{R}_p \cdot [l_{\mathrm{x}}(t), l_{\mathrm{y}}(t), l_{\mathrm{z}}(t)] + \mathbf{d}, \tag{4.17}$$

where $\mathbf{s}$ is the scaling matrix, $\mathbf{R}_p$ is the rotation matrix, and $\mathbf{d}$ is the translation matrix. For orientation, the quaternions are first converted into a rotation matrix $\mathbf{R}_i(t)$ as (4.18).

$$\mathbf{R}_i(t) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1-2(q_{\mathrm{y,i}}^2(t)+q_{\mathrm{z,i}}^2(t)) & 2(q_{\mathrm{x,i}}(t)q_{\mathrm{y,i}}(t)-q_{\mathrm{w,i}}(t)q_{\mathrm{z,i}}(t)) & 2(q_{\mathrm{x,i}}(t)q_{\mathrm{z,i}}(t)+q_{\mathrm{w,i}}(t)q_{\mathrm{z,i}}(t)) \\ 0 & 2(q_{\mathrm{x,i}}(t)q_{\mathrm{y,i}}(t)+q_{\mathrm{w,i}}(t)q_{\mathrm{z,i}}(t)) & 1-2(q_{\mathrm{x,i}}^2(t)+q_{\mathrm{z,i}}^2(t)) & 2(q_{\mathrm{y,i}}(t)q_{\mathrm{z,i}}(t)-q_{\mathrm{w,i}}(t)q_{\mathrm{x,i}}(t)) \\ 0 & 2(q_{\mathrm{x,i}}(t)q_{\mathrm{z,i}}(t)-q_{\mathrm{w,i}}(t)q_{\mathrm{y,i}}(t)) & 2(q_{\mathrm{y,i}}(t)q_{\mathrm{z,i}}(t)+q_{\mathrm{w,i}}(t)q_{\mathrm{x,i}}(t)) & 1-2(q_{\mathrm{x,i}}^2(t)+q_{\mathrm{y,i}}^2(t)) \end{bmatrix} \tag{4.18}$$

The transformed rotation matrix is then computed by multiplying $\mathbf{R}_i(t)$ with the orientation rotation matrix $\mathbf{R}_o$,

$$\tilde{\mathbf{R}}_i(t) = \mathbf{R}_i(t)\mathbf{R}_o. \tag{4.19}$$

Finally, $\tilde{\mathbf{R}}_i(t)$ is transformed back to the presentations of quaternions $[q_{\mathrm{x,i}}(t), q_{\mathrm{y,i}}(t), q_{\mathrm{z,i}}(t), q_{\mathrm{w,i}}(t)]$, representing the orientation in the robotic arm's coordinate system.

For the visual feedback, the pose of the end-effector of DT of the robotic arm is denoted by $\tilde{\mathbf{p}}_v(t) = [l_{\mathrm{x,v}}(t), l_{\mathrm{y,v}}(t), l_{\mathrm{z,v}}(t), q_{\mathrm{x,v}}(t), q_{\mathrm{y,v}}(t), q_{\mathrm{z,v}}(t), q_{\mathrm{w,v}}(t)]$. We quantify the error caused by MTP by the weighted sum of the RMSE between motions of operators $\tilde{\mathbf{p}}_i(t)$ and received

rendered virtual robotic arm $\tilde{\mathbf{p}}_v(t)$, which is denoted by

$$\begin{aligned}\mathbf{e}_v(t) = \omega_1 \cdot \mathbf{RMSE_p}([l_{\mathrm{x,i}}(t), l_{\mathrm{y,i}}(t), l_{\mathrm{z,i}}(t)], \\ [l_{\mathrm{x,v}}(t), l_{\mathrm{y,v}}(t), l_{\mathrm{z,v}}(t)]) \\ +\omega_2 \cdot \mathbf{RMSE_o}([q_{\mathrm{x,i}}(t), q_{\mathrm{y,i}}(t), q_{\mathrm{z,i}}(t), q_{\mathrm{w,i}}(t)], \\ [q_{\mathrm{x,v}}(t), q_{\mathrm{y,v}}(t), q_{\mathrm{z,v}}(t), q_{\mathrm{w,v}}(t)]),\end{aligned} \tag{4.20}$$

where $\mathbf{RMSE_p}(\cdot,\cdot)$ is the RMSE of position, $\mathbf{RMSE_o}(\cdot,\cdot)$ is the RMSE of orientation, and ω_1 and ω_2 are the weighting coefficients. The first term is the position error and the second is the orientation error. Depending on the accuracy requirements of different tasks in the cyber-physical systems, ω_1 and ω_2 can be set to different values. It is worth noting that the end effector of the robotic arm could be a gripper, a drill bit, or a sensor, depending on the specific task.

For the real-world robotic control, we denote and the pose of the real-world robotic arm by $\tilde{\mathbf{p}}_r(t) = [l_{\mathrm{x,r}}(t), l_{\mathrm{y,r}}(t), l_{\mathrm{z,r}}(t), q_{\mathrm{x,r}}(t), q_{\mathrm{y,r}}(t), q_{\mathrm{z,r}}(t), q_{\mathrm{w,r}}(t)]$ and evaluate the success of the task by the RMSE between motions of operators $\tilde{\mathbf{p}}_i(t)$ and real-world robotic arm $\tilde{\mathbf{p}}_r(t)$, which is expressed by

$$\begin{aligned}\mathbf{e}_r(t) = \omega_3 \cdot \mathbf{RMSE_p}([l_{\mathrm{x,i}}(t), l_{\mathrm{y,i}}(t), l_{\mathrm{z,i}}(t)], \\ [l_{\mathrm{x,r}}(t), l_{\mathrm{y,r}}(t), l_{\mathrm{z,r}}(t)]) \\ +\omega_4 \cdot \mathbf{RMSE_o}([q_{\mathrm{x,i}}(t), q_{\mathrm{y,i}}(t), q_{\mathrm{z,i}}(t), q_{\mathrm{w,i}}(t)], \\ [q_{\mathrm{x,r}}(t), q_{\mathrm{y,r}}(t), q_{\mathrm{z,r}}(t), q_{\mathrm{w,r}}(t)]),\end{aligned} \tag{4.21}$$

where ω_3 and ω_4 are the corresponding weighting coefficients.

4.4 Human-in-the-loop Model-Agnostic Meta-Learning

In this section, we introduce the Human-in-the-loop Model-Agnostic Meta-Learning algorithm, where a two-stage training process is included. In the first stage, we train the MAML based on the collected input data from the operators, functioning as a preheater that converges on multiple pre-defined tasks [97]. In the second stage, we use a human-in-the-loop online training strategy, where the operator interacts with the cyber-physical space in real-time, controlling the input device to generate control commands, and receive visual feedback to complete pre-defined tasks. In this stage, the model converges further based on the user's personalised characters and behaviours.

State

The state in the t-th time slot is a combined vector of the latest min-max normalized input pose $\mathbf{p}(t)$ and the E2E latencies for real-world robotic control and the visual feedback, denoted by $\mathbf{s}_t = [\mathbf{s}_t^{[1]}, \mathbf{s}_t^{[2]}] = [\mathbf{p}(t), T_r(t), T_v(t)], \mathbf{s}_t^{[1]} = \mathbf{p}(t) \in [-1,1]^{1\times 7}, \mathbf{s}_t^{[2]} = [T_r(t), T_v(t)] \in \mathbb{N}^{1\times 2}$.

Action

In the t-th time slot, the action to be taken are the lengths of prediction horizons for real-world robotic arm control $H_r(t)$ and for visual feedback $H_v(t)$, denoted by $\mathbf{a}_t = [\mathbf{a}_t^{[1]}, \mathbf{a}_t^{[2]}] = [H_r(t), H_v(t)]$.

Instantaneous Reward

In the t-th time slot, given state $\mathbf{s}_t$ and action $\mathbf{a}_t$, the instantaneous reward is a weighted sum of RMSEs, denoted by

$$r(\mathbf{s}_t, \mathbf{a}_t) = \mathbf{e}_v(t) + \mathbf{e}_r(t), \tag{4.22}$$

where $\mathbf{e}_r(t)$ is the RMSE for the real-world robotic control and $\mathbf{e}_v(t)$ is the RMSE for the visual feedback.

Policy

The policy is a mapping from the state to the probabilities of taking different actions, denoted by $\pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)$, where θ are the training parameters of the policy network. The network consists of multiple fully connected layers. Specifically, the first two layers are designed for feature extraction, whose the input, denoted by $[\mathbf{p}(t), T_r(t), T_v(t)]$, is transformed into a more compact and informative representation that captures the underlying patterns. After feature extraction, the network continues with fully connected layers that jointly optimize the prediction horizon for both real-world control and visual feedback. These layers are followed by a final linear layer. After applying the Softmax activation function, the distributions for the two actions, i.e., $\rho_t^{[1]}$ and $\rho_t^{[2]}$, are generated. Finally, the actions $\mathbf{a}_t^{[1]}$ and $\mathbf{a}_t^{[2]}$ are sampled from $\rho_t^{[1]}$ and $\rho_t^{[2]}$, respectively. Specifically, $\pi_\theta^{[1]}$ maps the state $\mathbf{s}_t$ to the distribution of $\mathbf{a}_t^{[1]}$, denoted by $\rho_t^{[1]} \in \mathbb{R}^{H_r}$ and $\pi_\theta^{[2]}$ maps the state $\mathbf{s}_t$ to the distribution of $\mathbf{a}_t^{[2]}$, denoted by $\rho_t^{[2]} \in \mathbb{R}^{H_v}$. The transmit distribution $\rho_t^{[i]}$ is defined as follows,

$$\rho_t^{[i]} \triangleq \begin{pmatrix} \Pr\{a_t^{[i]} = H\} \\ \cdots \\ \Pr\{a_t^{[i]} = 1\} \\ \Pr\{a_t^{[i]} = 0\} \end{pmatrix}, i = \{1,2\}, \tag{4.23}$$

where H represents the maximum prediction length. Similarly, $\pi_\theta^{[2]}$ maps the state $\mathbf{s}_t$ to the distribution of $\mathbf{a}_t^{[2]}$, denoted by $\rho_t^{[2]} \in \mathbb{R}^H$ with the process of (4.23). Once the distributions are obtained, $\mathbf{a}_t^{[1]}$ and $\mathbf{a}_t^{[2]}$ can be sampled from (4.23), respectively. Here, the probability of each action being sampled is based on the weight located in the corresponding elements [118].

4.4.1 Preliminary of MAML

MAML is a widely used algorithm in meta-learning, where the goal is to quickly adapt a model to new tasks with limited data [97]. The idea is that the convergence of the model on multiple tasks can be transformed into an aggregation that converges on each task. In our algorithm design, we use an extended version of its reinforcement learning to make it generalisable across multiple predefined real-time interactive tasks, as well as allowing for rapid adapting to the second stage online learning.

During meta-learning, the model is trained to be able to adapt to a large or infinite number of tasks. Here, we denote that there are $N \in \mathbb{N}^+$ tasks, $\mathbf{T} = T_1, T_2, \ldots, T_N$, and each task has finite length L. For example, The n-th task T_n can be defined as

$$T_n = \{\mathscr{L}_{T_n}(\pi_\theta), \{(\mathbf{a}_{t_1}, \mathbf{s}_{t_1}), ..., (\mathbf{a}_{t_L}, \mathbf{s}_{t_L})\}, \rho_t, L\}, \tag{4.24}$$

where $\{(\mathbf{a}_{t_1}, \mathbf{s}_{t_1}), ..., (\mathbf{a}_{t_L}, \mathbf{s}_{t_L})\}$ represents a sequence of L state-action pairs generated by following policy π_θ, ρ_t is the transition distribution of actions defined by (4.23).

Here, we adapt the loss function in MAML by incorporating the PPO loss function [118],

$$\begin{aligned} \mathscr{L}_{T_n}(\pi_\theta) = \min \Bigg(& \frac{\pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)}{\pi_{\theta^k}(\mathbf{a}_t \mid \mathbf{s}_t)} A^{\pi_\theta}(\mathbf{s}_t, \mathbf{a}_t), \\ & \text{clip}\left(\frac{\pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t)}{\pi_{\theta^k}(\mathbf{a}_t \mid \mathbf{s}_t)}, 1-\varepsilon, 1+\varepsilon\right) A^{\pi_{\theta_t}}(\mathbf{s}_t, \mathbf{a}_t) \Bigg), \end{aligned} \tag{4.25}$$

where θ^k represents the parameters of policy network in previous k steps, $A(\mathbf{s}_t, \mathbf{a}_t)$ is the advantage function defined as the difference between the state-action value function, $Q^{\pi_\theta}(\mathbf{s}_t, \mathbf{a}_t)$, and the state value function, $V^{\pi_\theta}(\mathbf{s}_t)$,

$$A^{\pi_\theta}(\mathbf{s}_t, \mathbf{a}_t) = Q^{\pi_\theta}(\mathbf{s}_t, \mathbf{a}_t) - V^{\pi_\theta}(\mathbf{s}_t), \tag{4.26}$$

which estimates the advantage of taking action $\mathbf{a}_t$ in state $\mathbf{s}_t$, over other possible actions in the same state [118].

Based on (4.25), we compute the adapted parameters with one step gradient descent by each time sampling K trajectories under policy π_{θ_n} for task T_n,

$$\theta_n = \theta - \alpha \cdot \nabla_\theta \mathscr{L}_{T_n}(\pi_\theta), \tag{4.27}$$

where α represents the step length and ∇_θ is the gradient with respect to the parameter θ. After performing (4.27) for each task $\mathbf{T} = T_1, T_2, \ldots, T_N$, we then implement the meta-optimization via Stochastic Gradient Descent (SGD), where the θ are further updated as follows,

$$\theta' \leftarrow \theta - \beta \cdot \nabla_\theta \sum_{n=1}^{N} \mathscr{L}_{T_n}(\pi_{\theta_n}), \tag{4.28}$$

where β represents the meta-step length.

4.4.2 Problem Formulation

We optimize the two prediction horizons for functions 1) real-world robotic arm control, and 2) visual feedback, to minimize the weighted sum of 1) the RMSE between the operator's input pose and the end-effector of the robotic arm $e_r(t)$, and 2) the RMSE between the operator's input pose and the pose of the DT of the robotic arm displayed in the screen $e_v(t)$ in $N \in \mathbb{N}^+$ tasks, $\mathbf{T} = T_1, T_2, \ldots, T_N$. For this purpose, we formulate it as a meta-reinforcement learning problem to find the optimal policy π^*_θ,

$$\underset{\theta, H_v(t), H_r(t)}{\arg\min} \; \mathbb{E}_{T_n \sim \mathbf{T}}[\mathscr{L}_{T_n}(\theta - \alpha \cdot \nabla_\theta \mathscr{L}_{T_n}(\theta))] \tag{4.29}$$

s.t.

$$\mathscr{L}_{T_n}(\theta) = \mathbb{E}_{\pi_\theta, T_n}[\sum_{t=0}^{L-1} \gamma^t r_n(\mathbf{a}_{t_1}, \mathbf{s}_{t_1})], \forall T_n \sim \mathbf{T} \tag{4.29a}$$

$$0 < H_v(t) < H_v \tag{4.29b}$$

$$0 < H_r(t) < H_r \tag{4.29c}$$

where γ represents the discount factor, H_v denotes the maximum prediction horizon for the rendering loop and H_r is the maximum prediction horizon for the control loop.

4.4.3 Two-Stage Training Process

Our proposed two-stage training algorithm is shown in Algorithm 1. In the first stage, we train the MAML based on the collected data which is detailed in Section 4.5.2. The goal of this stage is worked as a pre-heater, which enables the agent to quickly acquire a policy for the next stage of training, i.e., real-time interaction using only a small amount of experience.

The second stage, HITL online training is shown in Fig. 4.3. The operator interacts with a reinforcement learning environment - manipulating an input device in real time based on visual feedback to control a real-world robotic arm to perform a predetermined action. In this process, states and actions are generated online and reward is computed in real time. As the interaction continues, the model is trained until it converges.

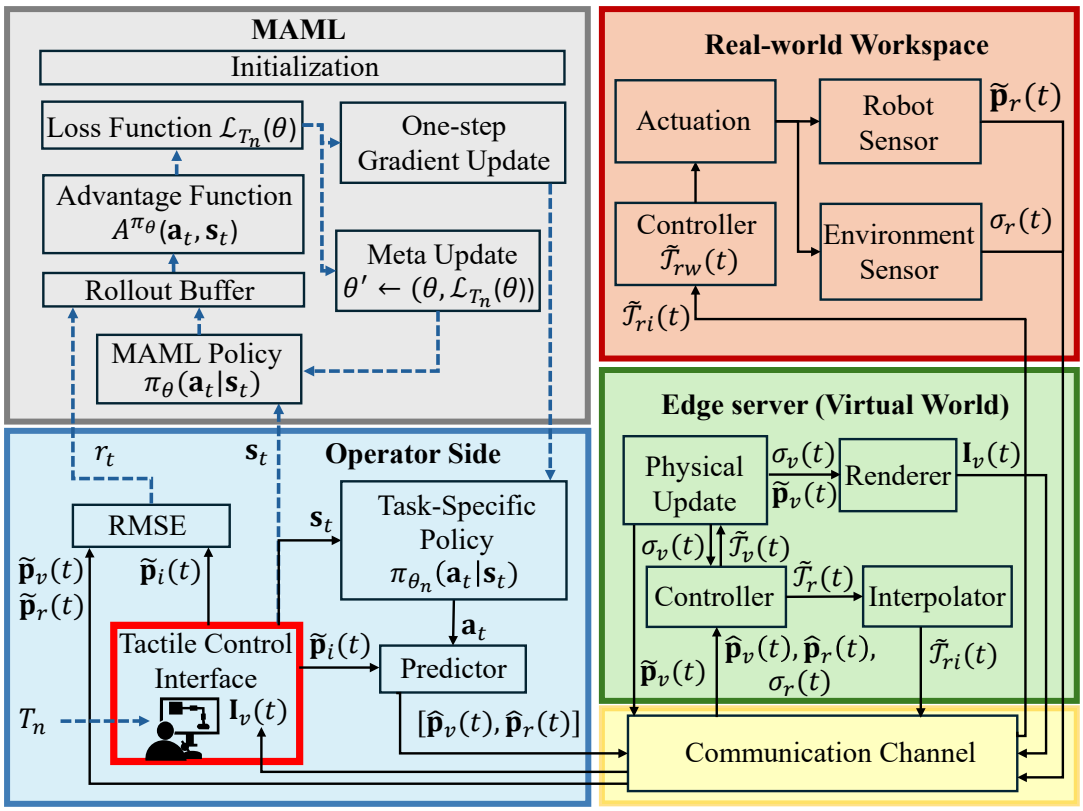

Figure 4.3: Illustration of proposed HITL online training architecture.

4.5 Prototype Design

4.5.1 System Setup

Input Device

As shown in Fig. 4.4, the Touch Haptic Device is employed as the primary input mechanism, enabling the operator to seamlessly interact with the system [107]. This device provides six Degrees of Freedom (DoF) for position and orientation sensing, alongside three degrees of force feedback. These capabilities offer an intuitive and immersive control interface, allowing for a tactile interaction experience. With a positional accuracy of up to 0.055 mm and a force feedback resolution as fine as 0.03 N, the device ensures precise input tracking, making it well-suited for tasks that require meticulous control. This high level of accuracy facilitates the fine manipulation of virtual objects, contributing to a smooth and responsive interaction between the operator and the virtual world. In the context of our framework, the Touch Haptic Device enhances the operator's ability to perform high-precision tasks with minimal error. Its integration into the system is crucial for scenarios demanding delicate and accurate manipulation, thus improving the overall functionality and efficiency of the framework. The experiments are conducted with a single operator to ensure controlled evaluation, while extending to multiple operators is left for future work.

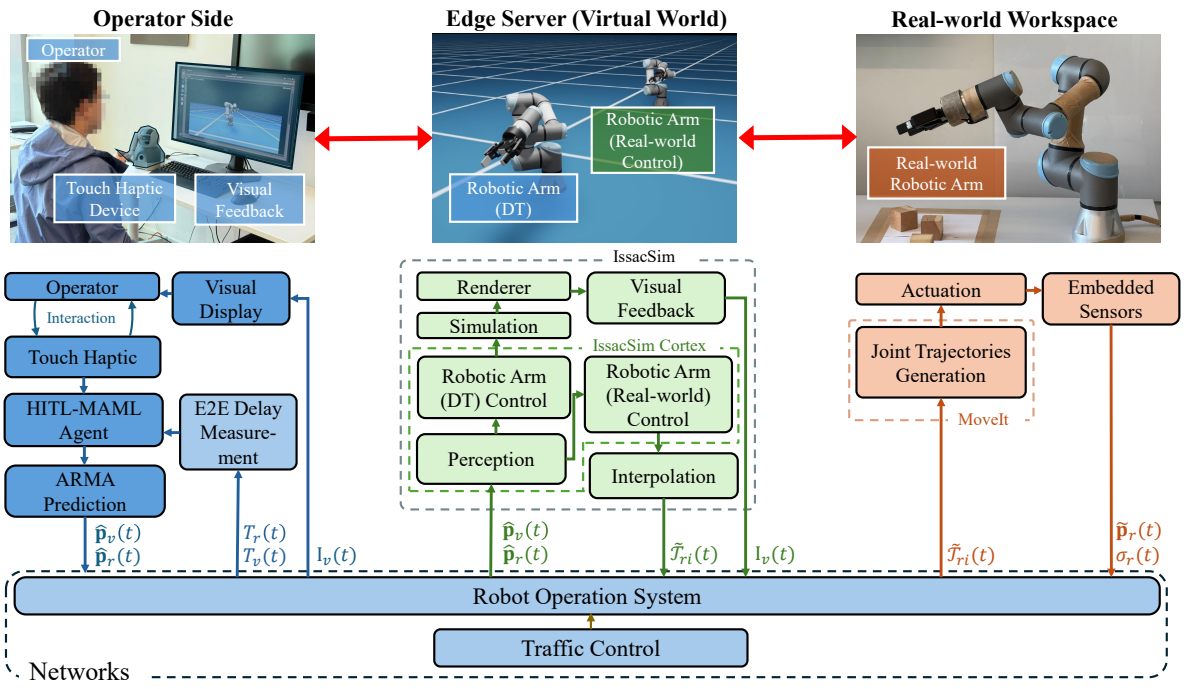

Figure 4.4: Illustration of our prototype system. 1) Left: A Touch Haptic Device and the screen for visual display, 2) Central: A DT of the robotic arm and the virtual environment deployed on the edge server, 3) Right: robotic arm in the real-world workspace.

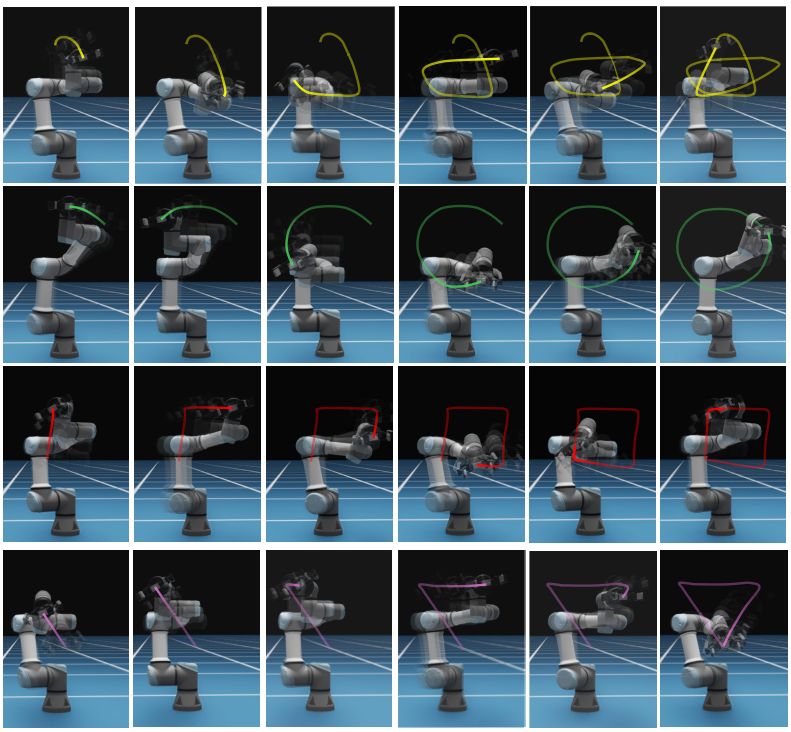

Figure 4.5: Illustration of four real-time interaction tasks, i.e., pentagram, circle, square, and triangle.

Algorithm 1 HITL-MAML

Input: initial all the parameters of neural network including randomly initial θ_0, initial state $\mathbf{s}_0$, step length α, meta-step length β, Episode number for two stages M_1, M_2

Stage 1:

while not done **do**
  Sample batch of tasks $T_n \in \mathbf{T}$
  for all T_n **do**
    Sample trajectories $\mathscr{D} = \{(\mathbf{a}_{t_1}, \mathbf{s}_{t_1}), ..., (\mathbf{a}_{t_L}, \mathbf{s}_{t_L})\}$ using π_θ in T_n
    Evaluate $\nabla_\theta \mathscr{L}_{T_n}(\pi_\theta)$ using $\mathscr{D}$ and $\mathscr{L}_{T_n}(\pi_\theta)$ in (4.25)
    Compute adapted parameters with gradient descent: $\theta_n = \theta - \alpha \cdot \nabla_\theta \mathscr{L}_{T_n}(\pi_\theta)$
    Sample trajectories $\mathscr{D}'_n = \{(\mathbf{a}_{t_1}, \mathbf{s}_{t_1}), ..., (\mathbf{a}_{t_L}, \mathbf{s}_{t_L})\}$ using $\pi_{\theta+1}$ in T_n
  end for
  Update $\theta' \leftarrow \theta - \beta \cdot \sum_{n=1}^{N} \mathscr{L}_{T_n}(\pi_{\theta_n})$ using each $\mathscr{D}'_n$ and $\mathscr{L}_{T_n}(\pi_{\theta_n})$
end while

Stage 2:

while not done **do**
  Operators interact with the DRL environment
  Generate trajectories $\mathscr{D} = \{(\mathbf{a}_{t_1}, \mathbf{s}_{t_1}), ..., (\mathbf{a}_{t_L}, \mathbf{s}_{t_L})\}$ based on π_θ and store in T'_n
  Repeat step 1-10 based on new tasks T'_n
end while

Output: Optimal policy π^*_θ with parameters θ^*

Visual Feedback

We utilize a Dell G3223Q 4K monitor displaying the virtual work environment hosted on the edge server [122]. This monitor provides high-resolution visual feedback of the DT of the robotic arm and the virtual world, with a refresh rate of 165 Hz, ensuring smooth and uninterrupted viewing. This allows the operator to assess the extent of the task based on visual feedback and make timely control via the touch device. Thus, the combination of control and visual feedback enhances the overall interaction, ensuring optimal performance in the proposed cyber-physical system.

Virtual world

The virtual world is built on the Nvidia Omniverse platform, utilizing Isaac Sim to simulate and control robotic systems [110], which is running on a high-performance edge server equipped with 32 GB RAM, an Nvidia RTX 3070 GPU, and an Intel Core i7-11700 16-core CPU. We employ Isaac Sim to create a highly accurate DT of the real-world robotic arm, modeling its dimensions and functionalities in detail. This virtual representation allows us to synchronize the actions in the simulated environment with the real-world robotic arm. In addition to the physical simulation, our framework integrates Isaac Cortex, which enhances task awareness and allows for more advanced decision-making capabilities in robotic systems [123]. Instead of merely simulating physical interactions, our framework uses this decision-making layer to optimize

how robots approach and complete tasks, leading to more efficient operations. We also decouple the physics updates from visual rendering in Isaac Sim, allowing us to maintain high-frequency physical updates while ensuring smooth visual feedback on the operator's monitor. This setup ensures that the operator receives continuous real-time feedback while controlling the virtual environment, ensuring accurate task execution. The overall framework, with its integration of Isaac Sim and Isaac Cortex, creates a tightly coupled system between the operator side, the virtual world, and the real-world robotic arm, enabling precise and efficient task management.

Real-World Robotic Arm

In our framework, we employ the Universal Robots UR3e industrial robotic arm as the core component for executing physical tasks [113]. The UR3e offers six DoF and achieves a repeatability of ±0.03 mm, making it well-suited for tasks that require high precision. With a payload capacity of 3 kg and a reach of 500 mm, the UR3e is optimized for small-scale, detailed operations while maintaining flexibility in diverse working environments. In the real-world workspace, the UR3e interacts with objects that are digitally mirrored in the virtual world, ensuring seamless integration between the cyber and physical domains. After receiving the control command from the edge server, the UR3e implements the control commands by utilizing the MoveIt Kinematics Plugin [114], which enables smooth and accurate joint motion execution. In addition, during the entire working process, the built-in sensors of the robotic arm monitor the joint angles, angular velocities, and applied forces to maintain consistent and precise control. This sensing system is critical for maintaining the accuracy required in our task-oriented framework, ensuring that every command is executed with the necessary precision and, accurate task-oriented KPIs calculations.

Networks

Our framework implements a publish-and-subscribe mechanism within the ROS environment to facilitate communication between the operator, the edge server, and the real-world workspace [115]. This mechanism allows for flexible and scalable data exchange, enabling the modular development of complex cyber-physical applications. Communication between processes relies on two protocols: Remote Procedure Call (RPC) and Transmission Control Protocol (TCP) [124]. When transmitting data, the local process first involves a publisher (talker) registering with the ROS Master on a designated port, including the topic name in its registration. A subscriber (listener) then registers with ROS Master and requests information based on its subscription. The ROS Master, upon finding a matching topic publisher, provides the listener with the talker's RPC address. The listener sends an RPC request to the talker, initiating the connection by specifying the topic name and message type. Once the talker confirms this request, it provides its TCP address, and the listener establishes a network connection. The talker can then begin publishing data to the listener over TCP. To simulate and control communication delays within this

Table 4.1: System Parameters for Performance Evaluation

Prototype Setup		**Learning Setup**	
Parameters	**Values**	**Parameters**	**Values**
Input device sampling frequency	120 Hz	Step length of one-step gradient update α	1×10^{-3}
Input length of prediction function W_p	4000 ms	Meta-step length β	1×10^{-5}
Prediction horizon of prediction function H	1000 ms	Batch size	256
Simulation frequency $\frac{T_{si}}{1}$	240 Hz	Discount factor γ	0.99
Rendering step length T_{im}	16 ms	General advantage estimation parameter λ	0.99
Rendering resolution	1920px $\times$ 1080px	Clip range	0.2
Real-world robotic arm sampling frequency	1000 Hz	Weighting coefficient $\omega_1, \omega_2, \omega_3, \omega_4$	-1

network, we used the network cables to connect the three via Ethernet during training, making the communication delay of the benchmarks nearly zero. Then, we utilize the TC method in Linux [117]. This utility allows us to manipulate network delay by introducing artificial delays, enabling controlled experiments on how varying levels of network delay affect the system's performance. By systematically varying communication delays with TC, we can observe their impact on the control and response of the robotic system. These controlled network conditions allow us to generate reproducible results for training and testing.

4.5.2 Data collection

We designed four real-time interaction tasks, where the operator utilizes the touch haptic device to control the real-world robotic arm in real time. The tasks involve completing four different trajectories—circle, triangle, pentagram, and square—within a specified workspace, guided by real-time visual feedback. The selected trajectories simulate common industrial robotic arm operations, such as cutting, mixing, and welding [37]. We further clarify the real-time control here, We further clarify the real-time control here: during the operation, the operator modifies in real-time the representation (size, shape, speed of movement, etc.) of the trajectories based on 3D visual feedback from a digital twin and virtual environment. These adjustments ensure that the final movement trajectory aligns with the task's requirements. The target pose of the touch haptic and the pose of the end-effector of the robotic arm were recorded in real-time and saved in Comma Separated Values (CSV). In the first stage, the operator's four maneuvers were captured for 150 repetitions of each maneuver, taking about an hour. In the second stage, the operator repeats these four maneuvers while training online until the model is converged.

4.6 Results

4.6.1 Training Settings

Algorithm training was conducted on a server with an Intel Core i9-13900F CPU, 64 GB RAM, and an Nvidia RTX 4090 GPU. The default parameters of the prototype and learning algorithm

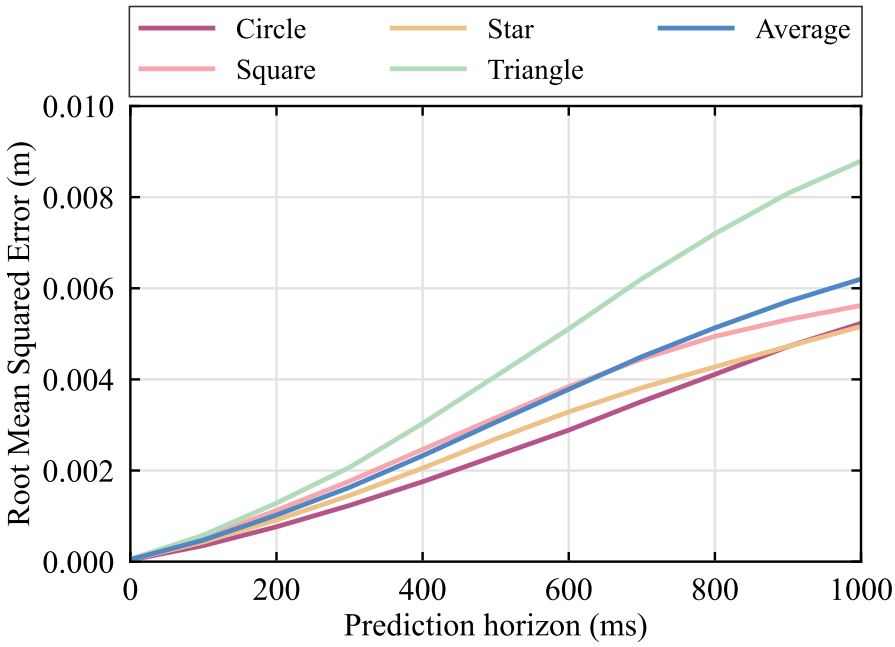

Figure 4.6: Evaluation of prediction model.

are listed in Table 4.1, unless otherwise specified. The parameters of HITL-MAML are updated following the procedure outlined in Algorithm 1. In the first stage, we use the collected motion data mentioned in Section 4.5.2. The framework sequentially executes each task over 4×10^4 ms. In the second stage, the operator interacts with the DRL environment. The training time of each task is set to 2×10^4 ms. To simulate network delays T_d, a normal distribution with a mean of $\mu_c = 50$ ms and a standard deviation of $\sigma_c = 10$ ms is set between the operator side and the edge server.

4.6.2 Performance Evaluation

The Prediction Model

As shown in Fig. 4.6, the performance of the predictor ARMA is evaluated by the RMSE with different prediction horizons. The results show that the predictor achieves similar performance in four predefined trajectories, with less than 0.01m RMSE when the prediction horizon is less than 1000 ms. It is intuitively that as the prediction horizon increases, the prediction error also rises. Thus, there exists an optimal point where there is neither an increase in error due to prediction by compensating for too much delay, nor an insufficient prediction where there is still an error. This lays the foundation of success when dynamically changing the prediction horizons when operators interact with the DRL environment.

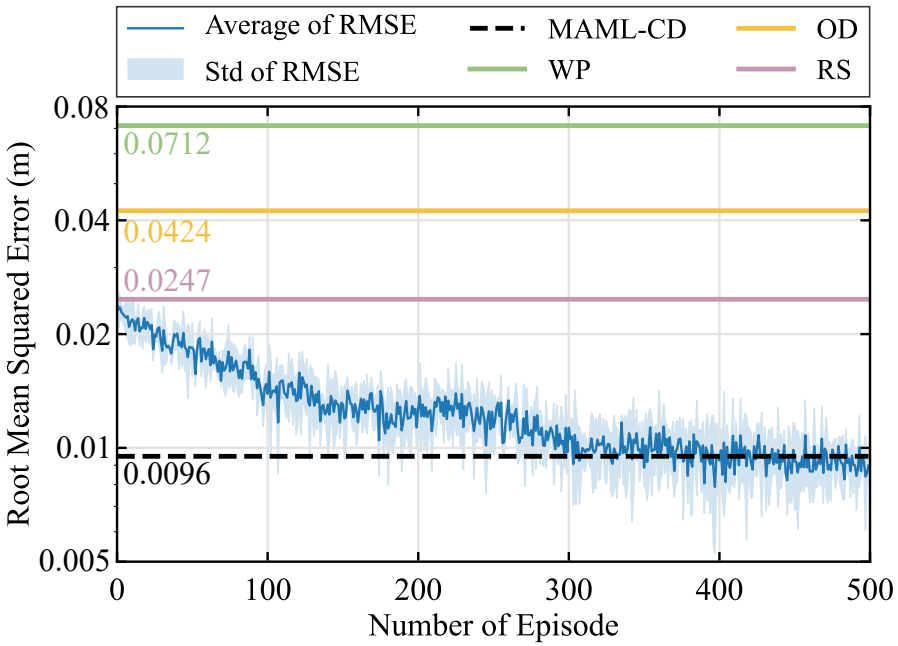

Figure 4.7: Evaluation of MAML-CD compared to three baselines.

Performance on the collected dataset

As shown in Fig. 4.7, we train the MAML algorithm over 500 episodes for three times to evaluate the proposed MAML training process on the collected dataset. To better illustrate our algorithm in the figure, we denote the MAML training process on the collected dataset as "MAML-CD". The RMSE gradually decreases during the first 300 episodes and reaches convergence with RMSE of 0.0096 m after 350 episodes of training. “Std" represents the standard deviation. To verify the effectiveness of the proposed algorithm, We compare the performance of the trained MAML with three baselines,

- WP: Without any delay compensation strategy, the pose of the input device is directly transmitted.
- RS: We predict the pose for the two functions, where the prediction horizons $H_r(t)$ and $H_v(t)$, are randomly sampled within the maximum value of H.
- OD: The prediction horizons $H_r(t)$ and $H_v(t)$, are dynamically changed depends on the measured E2E delays, $T_r(t)$ and $T_v(t)$. For example, if E2E delay of real-world robotic control and visual feedback is 127ms and 133ms, respectively, we set $H_r(t) = 127$ and $H_v(t) = 133$.

The results show a significant superiority over the three baselines. The RMSE decrease by 249.39%, 132.79%, and 61.13% compared to ’WP’, ’OD’, and ’RS’, respectively.

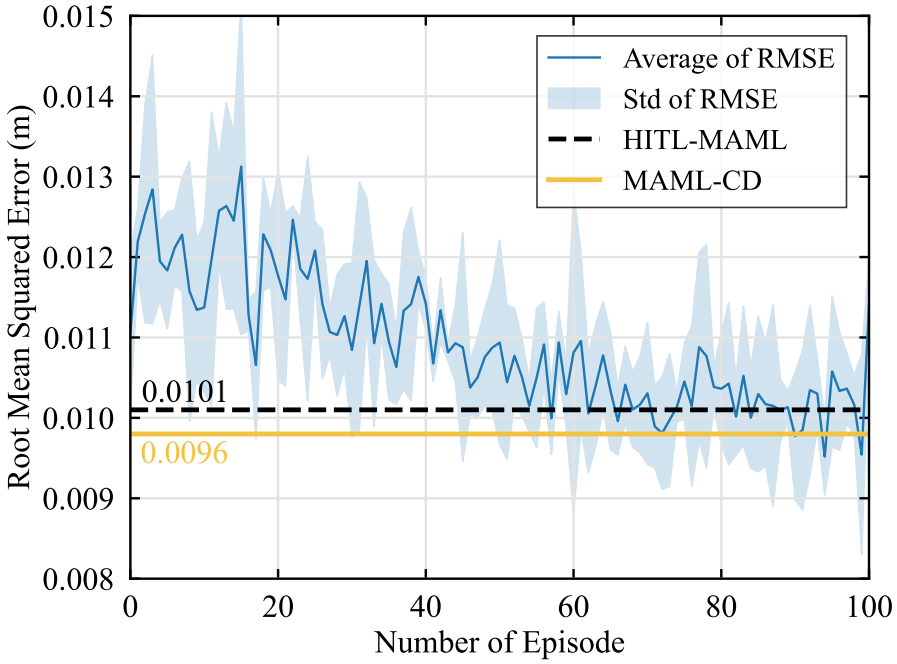

Figure 4.8: Evaluation of Proposed HITL-MAML.

Performance of HITL Online Training

As shown in Fig. 4.8, the performance of the HITL-MAML training process is evaluated by incorporating human operators controlling the input device during training. we compare the results with MAML-CD and the other three baselines. The results show that glshitl-MAML-CD rapidly converges after training for 80 episodes with an average RMSE of 0.0101 m. which is close to the RMSE of MAML-CD. This indicates that our proposed algorithm can quickly adapt to the Human-in-the-loop mode after a short period of online training, which verified the effectiveness and efficiency of our algorithm.

Evaluation of Performance for Different Tasks

As shown in Tab. 4.3, we further examined the RMSE of the proposed algorithm in each task, and compared it to the three baselines mentioned above. The results show that the average RMSE of our proposed algorithm is less than that of the three baselines in each task, further illustrating the effectiveness of our algorithm. We get the lowest RMSE in task 'square', where the average RMSE of position and orientation are 0.0068 m and 0.0089, respectively.

Evaluation in Different Communication Conditions

As shown in Fig. 4.9, We evaluate the performance of the proposed algorithm under different communication delay conditions. Specifically, In addition to the 0 ms communication delay as a benchmark, we add different normally distributed delays, with mean $\mu_c = 50, 100$ ms and

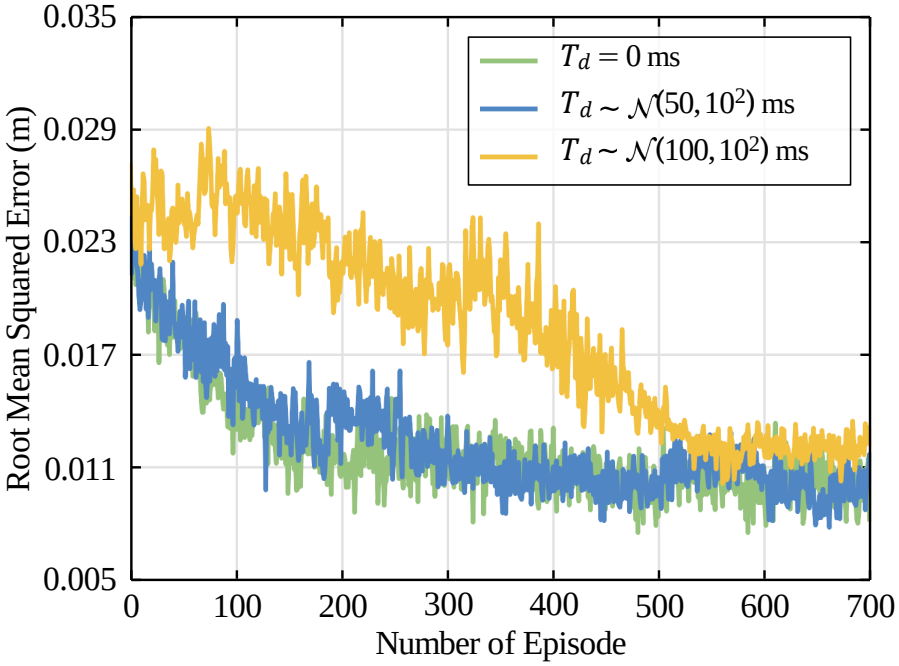

Figure 4.9: Evaluation of performance for different communications delays.

Table 4.2: Measured E2E delays and the Number of Convergence Episode

Communication delay (ms)	$T_d = 0$	$T_d \sim \mathcal{N}(50, 10^2)$	$T_d \sim \mathcal{N}(100, 10^2)$
Average RMSE (m)	0.0098	0.0094	0.01435
Convergence episode number	220	350	540
Average E2E delay (virtual world) (ms)	29.1	127.4	236.2
Average E2E delay (real-world workspace) (ms)	35.5	143.8	251.7

Table 4.3: Evaluation of Performance for Different Tasks

RMSE	Pose	HITL-MAML	"WP"	"OD"	"RS"
Circle	Position-vitual (m)	0.0091	0.0694	0.0398	0.0195
	Orientation-vitual	0.0086	0.0475	0.0445	0.0223
	Position-real (m)	0.0098	0.0812	0.0413	0.0264
	Orientation-real	0.0104	0.0546	0.0476	0.0211
Square	Position-vitual (m)	0.0067	0.0612	0.0365	0.0265
	Orientation-vitual	0.0068	0.0461	0.0278	0.0189
	Position-real (m)	0.0089	0.0689	0.0374	0.0210
	Orientation-real	0.0089	0.0691	0.0377	0.0227
Star	Position-vitual (m)	0.0099	0.0649	0.0514	0.0342
	Orientation-vitual	0.0123	0.0832	0.0446	0.0258
	Position-real (m)	0.0137	0.0792	0.0481	0.0298
	Orientation-real	0.0121	0.0711	0.0475	0.0243
Triangle	Position-vitual (m)	0.0106	0.0715	0.0427	0.0279
	Orientation-vitual	0.0114	0.0692	0.0392	0.0332
	Position-real (m)	0.0125	0.0811	0.0479	0.0295
	Orientation-real	0.0118	0.0732	0.0401	0.0224

a standard deviation of $\sigma_c = 10$ ms is set between the operator side and the edge server. The results show that our proposed algorithm can successfully converge and reach an RMSE of approximately 0.011 m. We also measured the E2E as shown in Tab. 4.2. The results also indicate that a higher E2E delay needs to make the algorithm take more time to converge.

Demonstration of RMSE

As shown in Fig. 4.10, we demonstrate the RMSE by showing different trajectories for two seconds. The input device's trajectories are transformed into the robotic arm's coordinate system by (4.17)(4.18)(4.19). Due to the fact that the input device is the one that issued the command, it is logical that it is ahead of the other trajectories at the start point. It can be visualized that our proposed algorithm achieves the lowest RMSE at the endpoint, with the point that is closest to that of the input device. This is a more graphic proof of the effectiveness of our algorithm.

4.7 Conclusions

In conclusion, we presented a novel framework that leveraged DTs to enhance the real-time interaction between human controllers and remote devices in CPS. By decoupling the DT into rendering and control components, we optimized two prediction horizons to minimize the weighted sum of 1) the RMSE between the operator's input pose and the end-effector of the robotic arm, and 2) the RMSE between the operator's input pose and the pose of the DT of the robotic arm

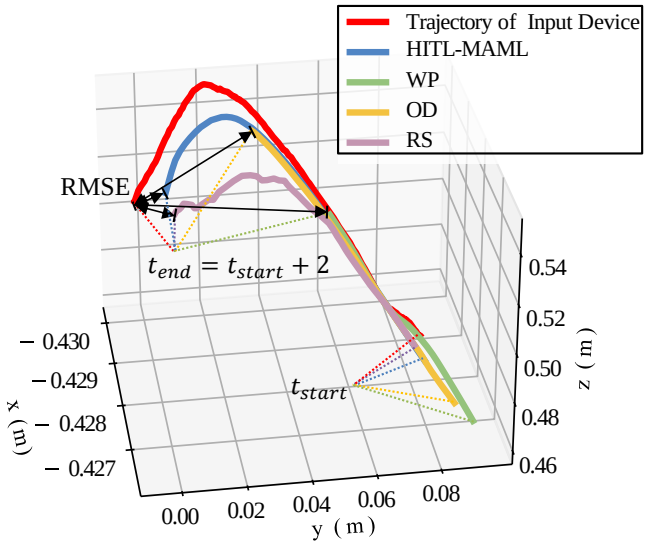

Figure 4.10: Demonstration of the trajectories of the end effector.

displayed in the screen for four tasks. By prediction of operator motions using HITL-MAML learning process, our system enabled proactive rendering and preemptive control generation. The average RMSE of our framework reached 0.0101 m and robust performance across various task executions. These findings highlight the potential of our framework to address key CPS challenges and pave the way for future applications in robotics and beyond.

Chapter 5

Preference-Driven Reinforcement Learning in Co-Design CPS via Human Feedback

5.1 Introduction

Active Three-Dimensional (3D) scene representation is a critical component of robotic vision, enabling a wide range of applications such as teleoperation [125], navigation [126], and manipulation [127]. In these scenarios, a robotic platform typically acquires visual data from multiple viewpoints and synthesizes a detailed representation of the scene. Most existing robotic viewpoint-planning methods rely on explicit geometric modelling or internal algorithmic parameters to guide decision-making [128]. Although such approaches leverage metrics like geometric uncertainty or model residuals to inform the selection of Next Best Views (NBVs), they face several critical limitations.

- Metric–User Mismatch: Quality measures based on novel-view rendering (e.g., Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), Learned Perceptual Image Patch Similarity (LPIPS)) demand additional viewpoints for evaluation and often lack clear selection criteria [129]. Moreover, these metrics do not always align with real-world user requirements for 3D visualization, particularly when only specific regions or perspectives matter.

- Geometry–Visualization Gap: Optimizing NBVs based on geometric accuracy or mesh completeness can theoretically enhance the fidelity of scene representations [130]. However, prioritizing highly accurate meshes does not always align with user expectations, particularly when the goal is to emphasize critical areas or distinctive structures rather than achieve full-scene completeness. Furthermore, geometric accuracy and visual rendering quality do not always correlate—an aesthetically pleasing rendering may not necessarily

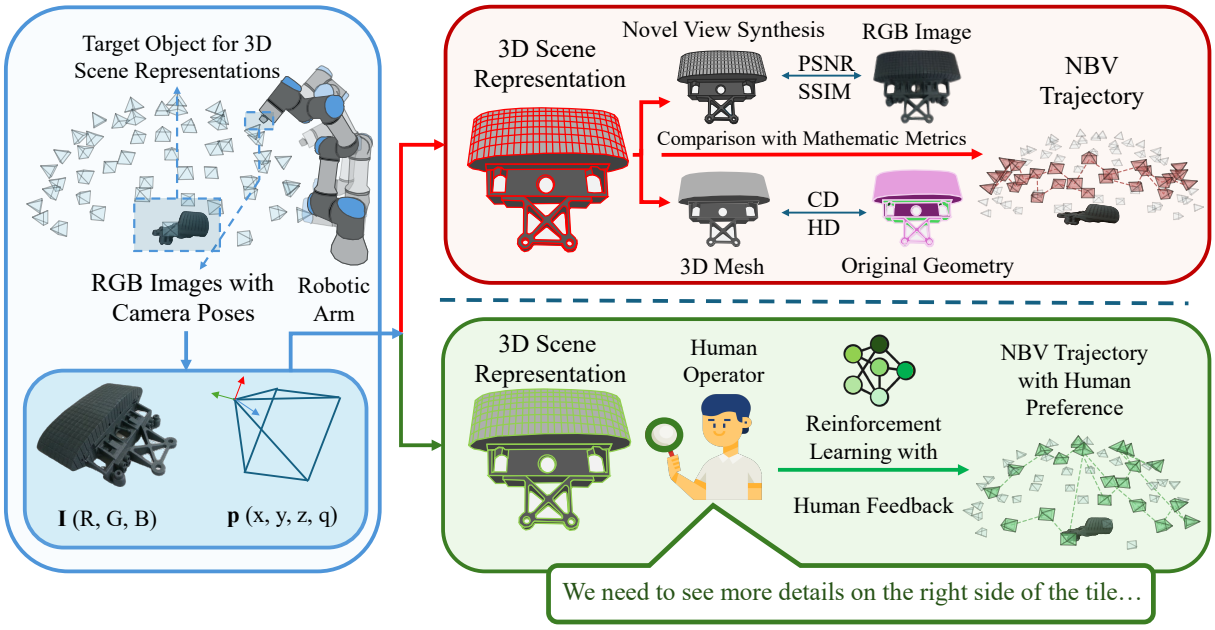

Figure 5.1: Motivation: traditional methods rely on static metrics without considering task- and preference-based 3D representations.

produce a clean or accurate mesh. [131].

In many industrial tasks involved with 3D scene representations, users require prioritized visualizations of specific regions, emphasizing user preferences over broad scene representation quality. Some task-specific objectives often prioritize user-centric visualization of critical regions rather than global scene representation accuracy [132]. For example, in teleoperated nuclear decommissioning, operators may need high-fidelity visualizations of a potentially hazardous material before interacting with it, while less critical background details can be simplified [133]. Similarly, in remote maintenance of nuclear facilities, ensuring clear visibility of occluded or safety-critical components, such as valve connections or radiation hotspots, takes precedence over globally consistent scene representation [134]. Thus, existing metrics often do not match human perception of visual quality, highlighting a need for methods directly optimizing user experience. Even worse, in constrained environments such as the nuclear decommissioning process, where robot mobility and the data acquisition range are severely limited, full geometric information acquisition is not practical [135].

Consequently, incorporating expert preferences and human-centric demands into active 3D scene representations remains an important yet under-explored direction. While human-in-the-loop approaches have been explored in nuclear Robotic and Autonomous Systems (RAS) and Robotics and Artificial Intelligence (RAI) [136], their integration into active 3D scene representation for optimizing viewpoint selection remains underdeveloped. Existing methods primarily

focus on predefined heuristics or geometric accuracy, lacking adaptability to operator-driven priorities in safety-critical environments. On the one hand, users often prioritize specific regions or viewpoints of interest; on the other, standard metrics are unable to capture subjective requirements. While RLHF has shown promise in natural language processing and 2D computer vision, its potential in robotics and 3D scene representations has not been fully realized [100].

Active 3D scene representations and NBV planning have evolved considerably over the past decade. Early approaches focused on hand-crafted heuristics, employing volumetric coverage or geometric uncertainty metrics to guide viewpoint selection [137], [138]. While these methods are effective in controlled or static environments, their reliance on predefined action spaces limits adaptability in complex, unstructured scenarios such as cluttered industrial settings or nuclear decommissioning facilities.

Recent advances leverage neural implicit representations, exemplified by Neural Radiance Fields (NeRF), to achieve more expressive and continuous scene modelling [139], [140]. Although these representations facilitate high-fidelity scene representations and flexible novel view synthesis, they commonly assume dense multi-view inputs and high computational resources, challenges that constrain real-time or resource-limited robotic applications. To address these issues, methods like NeU-NBV integrate uncertainty estimation into neural rendering, enabling NBV planning that selectively focuses on less certain regions while reducing data requirements [141]. Similarly, GenNBV employs DRL with multi-source feature embeddings to enhance generalization in large-scale and diverse environments [142].

A key trend is the explicit incorporation of uncertainty modelling to guide active exploration. Techniques such as FisherRF use information-theoretic measures to select viewpoints without relying on ground-truth data [143]. These strategies adaptively allocate sensing resources to informative directions, improving resilience against sparse observations and dynamic conditions. Such uncertainty-driven frameworks have proven beneficial for tasks like NeRF-based localization, where identifying high-value viewpoints can substantially reduce ambiguity in robot pose estimation [144]. In sparse-view scenarios, balancing data efficiency and scene representation quality is critical. Approaches like PVP-Recon iteratively select viewpoints based on warping consistency scores, maintaining scene representations fidelity under limited data budgets [130]. Further, methods like SparseNeuS and ReVoRF leverage geometric or semantic priors to mitigate overfitting, thereby ensuring stable performance with minimal input views [145], [146]. However, these approaches are often benchmarked by conventional metrics (e.g.,PSNR, SSIM), which do not fully capture user preferences or task-specific requirements.

A promising research direction involves integrating human feedback into the NBV planning loop while leveraging recent advances in embodied intelligence and large-scale language models. For example, the author in [36] argues that humans can leverage contextual understanding to supplement missing data during the scene representations process. Therefore, a human-robot interaction mechanism is introduced, enabling remote users to participate in and guide the robot

in completing the scene representations task. Specifically, Reinforcement Learning from Human Feedback (RLHF) allows robots to prioritize subjective user-defined objectives—such as region-specific coverage or safety-critical areas—without relying solely on predefined heuristics [100]. Meanwhile, embodiment enables active physical interaction (e.g., moving or manipulating objects) to reduce occlusions and refine uncertain areas [147]. Large-scale pretrained language models further enrich these pipelines by providing high-level reasoning and dynamic task planning in complex scenes [148]. For example, the author in [149] introduces an AIR-Embodied model that integrates embodied AI with multi-modal large language models to enable high-level reasoning, interactive object manipulations, and closed-loop verification, improving scene representations efficiency and generalization across diverse environments. Such approaches illustrate a shift toward user-centric, task-adaptive strategies, where traditional objective metrics are augmented by subjective preferences and operational constraints.

Despite these advancements, critical challenges persist—namely, resource limitations, environmental complexity, and the inherent difficulty of encoding subjective human goals directly into robotic policies. These gaps highlight the need for closer integration of uncertainty modelling, neural radiance representations, and RLHF paradigms, ultimately fostering NBV planning frameworks that are simultaneously robust, efficient, and aligned with the nuanced requirements of real-world robotic applications.

5.2 Contributions

In this chapter, we propose an RLHF-based framework for active 3D scene representations that incorporates human preferences into viewpoint planning. As shown in Fig. 5.1, instead of relying on model-specific internal variables, our method draws on user evaluations of rendered 3D models to inform the policy-learning process. We demonstrate the effectiveness of this user-centric priority in a nuclear decommissioning scenario. The primary contributions of this work can be summarized as follows:

- 1) Preference-driven NBV: We introduce a DRL strategy that explicitly accounts for the 3D visualization preference of the expert operator, thus eliminating reliance on fixed geometry- or rendering-based scores.
- 2) Scalable Policy Learning: Our approach integrates seamlessly with the robotic platforms and diverse 3D scene representations pipelines by avoiding dependence on algorithm-specific internal cues.
- 3) Empirical Validation: We evaluate our method on a real-world robotic testbed, showing that operator-guided policies significantly improve localized scene representations quality.

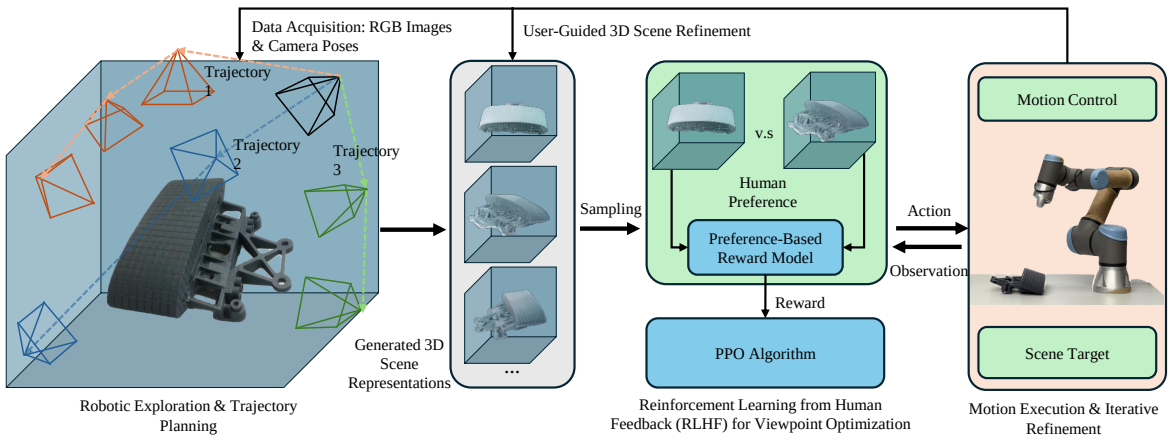

Figure 5.2: Overview of the proposed framework. The RLHF pipeline consists of five key stages: (1) Robotic exploration and trajectory planning, where the robotic system collects observations; (2) Expert operator preference evaluation, where operators select preferred scene representations; (3) Learning a reward model based on collected human feedback; (4) Policy optimization using PPO algorithms; and (5) Online training, where new data continuously refines the learned policy for improved viewpoint selection.

5.3 Method

In this section, we introduce our algorithm that actively selects the next-best view for a robotic arm equipped with a camera to improve the quality of 3D scene representations in a limited data setting. By leveraging a calibrated hand-eye system, we determine the camera pose from the robotic arm's end-effector pose. To enhance the performance of the 3D scene representations, our approach employs RLHF to optimize the policy, allowing humans to compare scene representations outcomes and provide feedback on the most effective next views.

5.3.1 Preliminaries

NeRF represents continuous radiance fields of static scenes through a Multi-Layer Perception (MLP) network. For a given 3D point $[x, y, z]$ in the scene space and ray direction $[\theta, \phi]$ from the camera's center of projection $\mathbf{o} = [x_o, y_o, z_o]$, NeRF learns an implicit function that maps these inputs to volume density σ and color $\mathbf{c} = [r, g, b]$ via an MLP, denoted by

$$\sigma, \mathbf{c} = F_\Theta((x, y, z, \theta, \phi), \alpha_\Theta) \tag{5.1}$$

where the parameters of the MLP are denoted by α_Θ.

To validate and visualize the 3D scene representations, a 2D image $\hat{\mathbf{I}}$ can be obtained by volume rendering [150], which is expressed by

$$\hat{\mathbf{I}} = F_r\left(\mathbf{c}, \sigma, (x, y, z, \theta, \phi), \alpha_r\right), \tag{5.2}$$

where α_r is the parameters of the volume rendering function $F_r(\cdot)$. For $F_r(\cdot)$, given a ray $\mathbf{r}(q) = \mathbf{o} + q\mathbf{d}, \mathbf{d} \in \mathbb{R}^3, ||\mathbf{d}|| = 1$ emanating from the camera position $\mathbf{o}$ and direction $\mathbf{d}$ defined by the specified camera pose and intrinsic parameters. The rendered color $C(\mathbf{r})$ of each pixel corresponding to this ray $\mathbf{r}$ can be calculated by integrating the cumulative opacity-weighted radiance, which is expressed by

$$C(r) = \int_{q_n}^{q_f} T(q)\sigma(\mathbf{r}(q))\mathbf{c}(\mathbf{r}(q))dq, \tag{5.3}$$

where $[q_n, q_f]$ defines the integration bounds and $T(t)$ represents the accumulated transmittance, defined by

$$T(q) = \exp(-\int_{q_n}^{q} \sigma(r(s))ds), \tag{5.4}$$

where s is the distance that the ray has traveled from its starting point q_n to the current point q along the ray's path.

To accelerate the training and rendering process of the 3D scene representations, we adopt the Instant-NGP [151] in the proposed framework. The Instant-NGP introduces a multi-resolution hash encoding that efficiently maps input coordinates to feature vectors through a parameterized hash table H, this encoding transforms the input coordinates before they are processed by the MLP. This encoding scheme significantly reduces computation time while maintaining high scene representations quality.

This encoding transforms the input coordinates before they are processed by the MLP, expressed as

$$E(\mathbf{p}) = \sum_{l=1}^{L} w_l(\mathbf{p})\mathscr{H}l(\lfloor 2^l\mathbf{p} \rfloor), \tag{5.5}$$

where $\mathbf{p} = [x, y, z]$ represents the spatial coordinates, L is the number of resolution levels, and $w_l(\mathbf{p})$ are the interpolation weights. The hash tables $\mathscr{H}ll = 1^L$ are trained alongside the MLP parameters, with each level corresponding to a different spatial resolution 2^l. The modified MLP then takes this encoded input: $\sigma, \mathbf{c} = F\Theta(E(\mathbf{p}), [\theta, \phi], \alpha_\Theta)$. This encoding scheme significantly reduces computation time while maintaining high scene representations quality, enabling real-time rendering of novel views through efficient feature lookup and interpolation.

5.3.2 Optimizing Active 3D Scene Representations via RLHF

Active 3D scene representations seek to optimize the actions of a robotic agent to generate high-quality 3D representations of objects or environments. This involves addressing two key challenges:

- Exploration vs. Refinement: Balancing the discovery of new perspectives with revisiting regions to improve 3D scene representations accuracy.
- Quantifying Scene Representations Quality: Defining a reward function that captures 3D scene representations quality from a robotic perspective is inherently complex and subjective.

To address these challenges, as shown in Fig. 5.2, we propose leveraging RLHF to integrate the preference of the expert operator into the optimization process, enabling the agent to learn a reward function aligned with subjective quality assessments.

Step 1: Robotic Exploration and Trajectory Planning

At each time step t, the robot agent performs the following steps:

- **Observation:** The agent captures an RGB image using its camera and extracts relevant features through the YOLO v11 model [152]. We selected YOLO v11 due to its real-time inference capability and strong feature extraction performance. We used a pre-trained YOLO v11 model without fine-tuning, as its generalization ability to common object features was sufficient for our scene representation task. The processed observation is represented as $o_t \in [-1,1]^{d_1 \times d_2 \times d_3}$, where d_1, d_2, and d_3 denote the dimensions of the extracted feature map.
- **Action:** The agent selects an action a_t from the discrete action space $\mathbb{A} = \{1,2,3,\ldots,\mathscr{A}\}$, where $\mathscr{A}$ represents the total number of possible camera poses and corresponding image acquisitions.
- **Update:** The robotic arm moves within the workspace according to the selected action. Upon reaching the new pose, a fresh observation o_t is captured and stored in the dataset.

The interaction process generates a trajectory $\sigma = (o_t, a_t)_{t=1}^{k}$, which consists of a sequence of k observation-action pairs. These trajectories serve as the foundation for subsequent reward evaluation and policy learning.

Step 2:Evaluating the Preferences of Expert Operators

To ground the quality of scene representations in expert operator preference:

- Model Comparison: As shown in Fig. 5.3, the expert operator compares pairs of 3D scene representations M_1 and M_2, derived from trajectory segments σ_1 and σ_2, respectively. Since the focus is on task-relevant scenes or objects, we employed the Segment Anything Model (SAM) to segment the target regions of interest [153]. To ensure accurate segmentation aligned with the expert operator's comparison process, we fine-tuned SAM using

images from the first-step observation, enabling more precise scene representation tailored to the task.

- Preference Indication: Evaluators provide a binary preference $\mu \in \{1,2\}$, signifying which scene representations better captures the object or environment.

Step 3: Reward Function Estimation

A reward function $\hat{r}(o_t, a_t)$ is trained to predict the expert operator preferences [100]:

- Preference Modelling: Define the likelihood of the expert operator preference for trajectory σ_1 over σ_2 as:

$$P[\sigma_1 \succ \sigma_2] = \frac{\exp\left(\sum_{t=1}^{k} \hat{r}(o_t^1, a_t^1)\right)}{\exp\left(\sum_{t=1}^{k} \hat{r}(o_t^1, a_t^1)\right) + \exp\left(\sum_{t=1}^{k} \hat{r}(o_t^2, a_t^2)\right)}.$$

- Loss Function: Train $\hat{r}$ by minimizing the cross-entropy loss:

$$\begin{aligned} \text{Loss}(\hat{r}) = -\sum_{(\sigma_1,\sigma_2,\mu)\in D} \big[&\mu(1)\log P[\sigma_1 \succ \sigma_2] \\ &+ \mu(2)\log P[\sigma_2 \succ \sigma_1]\big], \end{aligned} \tag{5.6}$$

where D is the dataset of trajectory pairs and preferences.

Step 4: Policy Optimization

Using the learned reward function $\hat{r}$, we optimize a policy π to maximize cumulative rewards:

$$R = \sum_{t=1}^{T} \hat{r}(o_t, a_t).$$

In this study, we employ PPO as the primary DRL algorithm [118]. However, our approach is not restricted to PPO and can be generalized to a wide range of DRL algorithms.

Step 5: Online Training for Fine-Tuning

After executing the actions dictated by the optimized policy, the robotic arm follows the planned trajectory to generate a 3D scene representation model. The newly generated representations can then be incorporated into the user preference selection process, enabling continuous refinement through online training. As the robotic arm moves, it captures images of regions with a higher probability of user interest, along with corresponding camera poses. These newly acquired images and poses can be fed back into the first step to generate additional 3D scene representations,

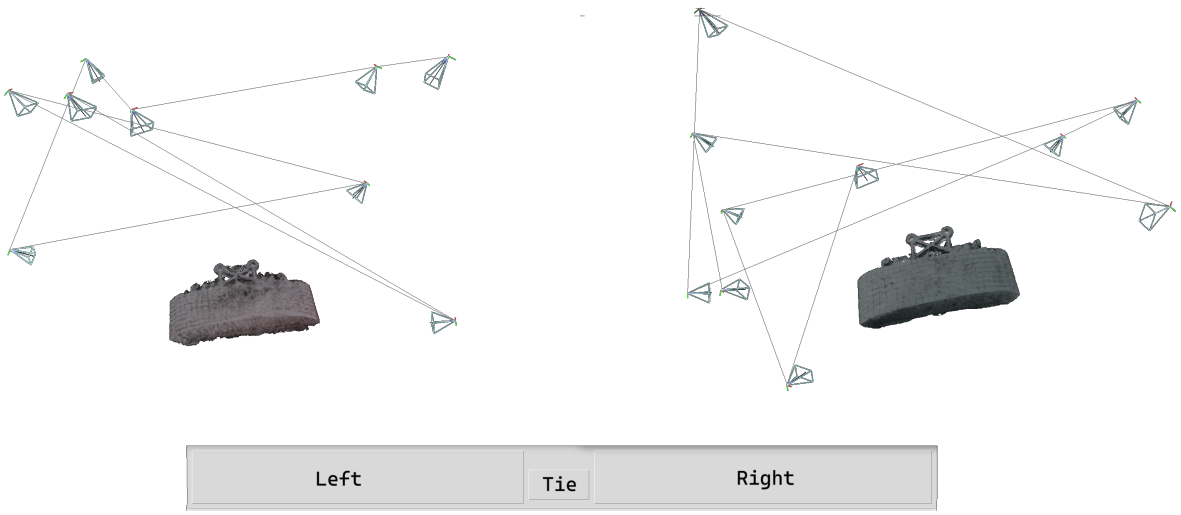

Figure 5.3: Illustrations of user interface for preference-based 3D scene selection and sequence representation. The interface allows users to compare and select preferred 3D scene representations, which are used to train a reward predictor for viewpoint optimization. Users can zoom, rotate, and inspect models for detailed evaluation. Notably, the illustrated line indicates the viewpoint selection order, not actual robotic motion. The real UR3e motion is planned using the Isaac Sim motion planner for smooth and optimized execution.

further enriching the dataset. This process can be iteratively repeated, progressively optimizing the robotic motion trajectory and improving both viewpoint efficiency and scene representation quality over time. The complete workflow of our proposed algorithm is demonstrated in ALG. 2.

5.4 Experiment Setup

5.4.1 Robotic System Configuration

To validate our RLHF-based active 3D scene representations framework, as shown in Fig. 5.4, we employed a Universal Robots UR3e robotic manipulator [113], with an Intel RealSense d435i camera mounted on its end-effector [154]. Although this camera is capable of capturing depth data, only the RGB output is utilized, aligning with our three 3D scene representations baseline method, aligned with the baseline methods in Section 5.5. The UR3e offers six DoF and achieves a repeatability of $\pm 0.03\,\text{mm}$, ensuring precise and consistent positioning during viewpoint adjustments.

To accommodate real-time robot control and the computationally intensive 3D scene representations processes, our testbed relies on two high-performance servers:

- **Robotic Arm Control Server:** Executes real-time motion trajectories and processes control signals. It leverages rmpflow motion generation tools in the *NVIDIA Isaac Sim* [110]

Algorithm 2 RLHF-based Active Robotic 3D Scene Representations

Input: Initialized policy π_θ, reward predictor $\hat{r}_\phi$, dataset $D = \emptyset$

Phase 1: Train reward predictor

for $t = 1, \ldots, T_r$ **do**
  Collect trajectories σ_1, σ_2 with π_θ
  Expert provides preference $\mu \in \{1, 2\}$
  Update $D \leftarrow D \cup \{(\sigma_1, \sigma_2, \mu)\}$
  Train $\hat{r}$ with loss $\text{Loss}(\hat{r})$ (Eq. 5.6)
end for

Phase 2: Policy optimization

for $t = 1, \ldots, T_p$ **do**
  Collect trajectories σ with π_θ
  Compute rewards $r_t = \hat{r}_\phi(o_t, a_t)$
  Update π_θ using PPO surrogate objective
end for

Phase 3: Online refinement

for $t = 1, \ldots, T_f$ **do**
  Collect new trajectories with π_θ
  Expert provides additional preferences
  Update $\hat{r}_\phi$ and fine-tune π_θ
end for

Output: Optimized policy π_θ^*

for trajectory optimization, and use *MoveIt* Kinematics Plugin for smooth and accurate joint motion execution [114]. The hardware setup includes an *Intel Core i7-11700 CPU, 32 GB RAM, and an NVIDIA RTX 3070 GPU*, enabling low-latency motion control and on-the-fly image acquisition.

- **DRL and 3D Scene Representations Server:** Hosts the RLHF training and the scene representations algorithm. An *Intel Core i9-13900 CPU, 64 GB RAM, and an NVIDIA RTX 4090 GPU* enables efficient policy optimization and real-time rendering.

A Robot Operating System (ROS)-based framework coordinates communication among the robotic arm, RGB cameras, and control modules. The control server transmits motion commands and receives sensor feedback via ROS topics, ensuring synchronized execution. The DRL and Scene Representations Server receives images captured by the camera and the end-effector's pose from the control server via the File Transfer Protocol (FTP). It then sends the target pose of the end-effector back to the control server. The FTP facilitates large data transfers between servers, particularly for scene representation and DRL updates. The real-world robotic workspace integrates a robotic arm, RGB cameras, and an operational tile-based environment, ensuring robust testing conditions.

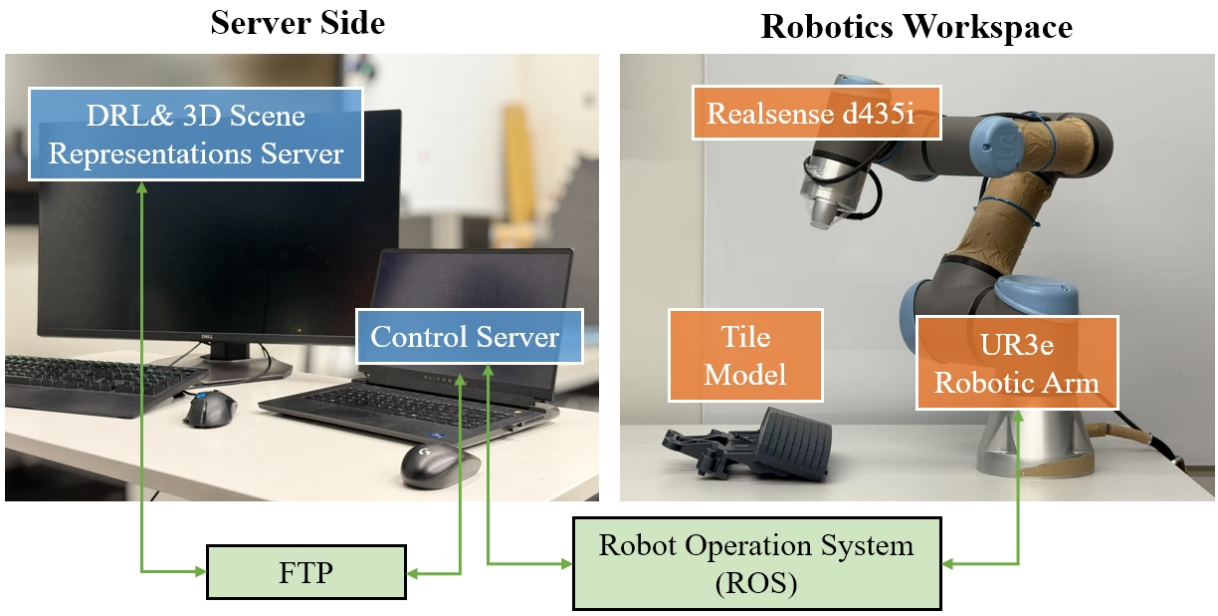

Figure 5.4: Experimental setup of the RLHF-based 3D scene representation system. The setup consists of a UR3e robotic arm with an Intel RealSense D435i camera, controlled via a ROS-based framework. A control server handles motion execution, while a DRL server optimizes viewpoint selection based on human feedback. An FTP ensures efficient data transfer, enabling real-time policy refinement for 3D scene representation.

5.4.2 Experiment Implementations

We focus on a critical task in the automation process of nuclear decommissioning: utilizing a robotic arm for high-precision 3D visualization and inspection of reactor tiles [155]. These tiles, typically composed of beryllium-coated Inconel and arranged in a slightly tilted manner to regulate plasma circulation within the reactor core, play a crucial role in maintaining structural integrity. Damage to these tiles can lead to plasma leakage and further degradation of internal reactor components, making accurate inspection essential for safe decommissioning operations [156].

To facilitate this process, we employ high-precision, full-scale 3D-printed replicas of reactor tiles as test samples, ensuring a realistic and controlled evaluation environment. The designed pipeline systematically acquires RGB images and corresponding poses from multiple viewpoints, integrating the preference of the expert operator to refine the RLHF-based policy for enhanced inspection accuracy. The details are as follows:

1. **Scene Initialization and Calibration:**

   A structured environment is established within the UR3e workspace to simulate realistic 3D scene representation scenarios. Before data collection, hand-eye calibration is per-

formed to accurately determine the spatial relationship between the camera and the robot's base frame [157]. This process includes calibrating the camera's intrinsic parameters to ensure precise image measurements, while also estimating the camera's pose relative to the robotic arm. By obtaining an accurate transformation between the camera and the robot base, the system can achieve high-precision 3D scene representations. In addition, this calibration ensures that the robotic arm can move to the required poses with precision, enabling reliable perception and interaction within the workspace.

2. **Preference of Expert Operator:**

   Five experienced personnel collaborating with Remote Applications in Challenging Environments, UK Atomic Energy Authority (UKAEA) [158], who possess in-depth knowledge of nuclear decommissioning environments and 3D scene representations quality requirements, are selected as the expert operators. The expert is familiar with the objectives of scene representation, including clarity, accuracy, and task-specific relevance. Although the experiment involves only a single subject, its reliability is enhanced through multiple repetitions and various controlled experiments. Furthermore, an in-depth discussion on the differences in personalized requirements among multiple users is beyond the scope of this study. Then, multiple 3D scene representations, generated from different viewpoint trajectories, are presented to the expert operators for comparative evaluation. Users interact with the models through zooming and rotation to conduct a detailed assessment. Preferences are selected based on task-relevant criteria such as focus area, visual clarity, feature coverage, and geometric fidelity. These pairwise comparisons are aggregated to train a reward model that encapsulates human-centric quality judgments. Every 10 captures, a new 3D scene representations is generated, repeating for 400 instances under varying viewpoints and conditions. These representations are then paired into 200 comparison groups for the expert operator's evaluation.

3. **Policy Optimization:**

   Using PPO, the DRL agent updates its viewpoint-selection policy to maximize the reward function derived from the preference of the expert operator data. The training parameters for DRL can be found in Table I. In addition, the agent autonomously refines its viewpoint selection policy through an iterative learning process. Initially, it explores the environment using random viewpoint sampling to collect a diverse set of observations. As learning progresses, the policy is progressively optimized based on the expert operator preference, guiding the selection toward viewpoints that align with user preferences. By leveraging RLHF, the system adapts to prioritize views that enhance critical scene representation quality while maintaining efficiency in data collection and robotic motion.

Table 5.1: RL Parameters for Performance Evaluation

Parameters	**Values**
learning rate	$1e-5$
batch size	32
n-step return	256
discount factor	0.99
value function coefficient	0.99
clip range	0.2

5.5 Results

5.5.1 Baseline 3D Scene Representation Methods

We evaluate three 3D scene representation approaches to establish broader performance baselines,

- **Instant-NGP [159]:** A *NeRF-based* method that uses multi-resolution hash encoding for rapid and high-fidelity scene representations. We adopt it as the main reference point for implicitly modelling radiance fields directly from RGB images.
- **3DGS [160]:** An advanced volumetric approach designed to capture complex scene geometry. It leverages a grid-based structure and tailored optimization strategies to handle varying scene complexities.
- **PGSR [161]:** A geometry-optimized scene representations pipeline that emphasizes precise surface modelling. By focusing on explicit geometric cues, PGSR aims to achieve accurate mesh representations, particularly in scenarios demanding fine-grained detail.

All three methods are integrated with identical data-capture protocols, enabling a consistent evaluation of their strengths, weaknesses, and overall scene representation quality when operating on RGB images as they encompass implicit, volumetric, and explicit modelling paradigms. By covering these distinct methodologies, our selection provides a diverse and representative evaluation framework, demonstrating that our RLHF-based approach can effectively leverage existing well-established benchmarks for broad applicability.

5.5.2 Evaluation Metrics

Evaluation of our proposed framework combines both objective quality assessments and subjective alignment with the preference of the expert operator:

- **Scene Representations Fidelity:**
  - *PSNR:* Quantifies the alignment between rendered novel views and reference images.
  - *SSIM:* Measures perceptual consistency in local intensity and texture patterns.
  - *LPIPS:* Uses deep network embeddings to gauge perceptual similarity between synthetic and real images.
- **Trajectory Efficiency:**
  - *Robot path length:* The distance travelled by the end-effector, reflecting mechanical and time efficiency in the exploration.
- **Human Preference Alignment:**
  - *Subjective Visual Assessment:* Evaluators compare 3D scene renderings from the optimized policy and a random strategy, assessing clarity, completeness, and detail.
  - *Comparison with Random Strategy:* Evaluators reviewed paired renderings and indicated which provided better scene understanding.

By combining these objective and human-informed metrics, we demonstrate that our RLHF-based method not only achieves high-fidelity 3D scene representations from RGB images alone, but also aligns closely with human judgments of scene quality.

5.5.3 Convergence of Proposed Algorithm with Different 3D Representation Methods

Our proposed RLHF framework demonstrates strong convergence behavior when trained on three different 3D representation methods: Instant-NGP, 3DGS, and PGSR. As shown in Fig. 5.5, our proposed framework achieves stable policy updates efficiently across all three methods, indicating its robustness and adaptability to different scene representations. The convergence results validate the effectiveness of incorporating human preference feedback, which accelerates learning and improves overall scene representation quality. To ensure statistical reliability, each algorithm was repeated five times, and the standard deviation (std) was computed to verify the robustness of the proposed approach. This demonstrates that our proposed framework can effectively guides the robotic system to capture viewpoints that maximize visual clarity and align with user-centric representation.

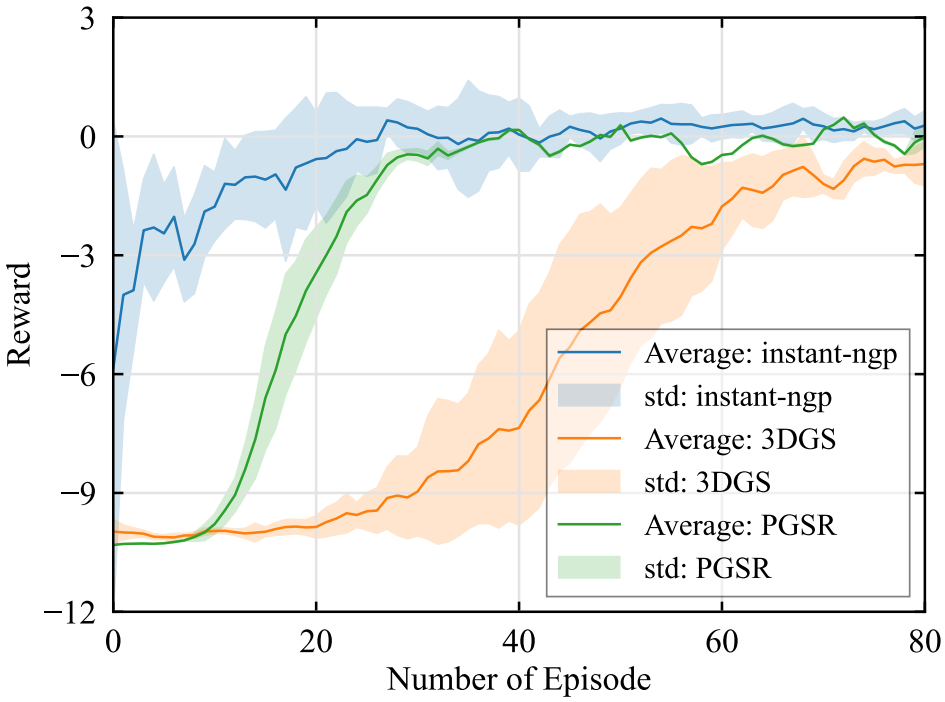

Figure 5.5: Convergence performance of our proposed framework across different 3D scene representation methods.

Table 5.2: Comparative Analysis of Different Viewpoint Numbers with Task Times and Representation Performance

Number of Photos	5	8	10	13	15	20
Average Task Time (s)	26.34	34.72	38.91	50.18	57.44	76.21
Average PSNR	20.19	19.64	21.83	20.67	22.79	22.74
Average SSIM	0.8266	0.7699	0.8760	0.8763	0.8808	0.8904
Average LPIPS	0.1813	0.2238	0.1412	0.1291	0.1215	0.1274

5.5.4 Effectiveness Verification of Proposed Framework with Different Viewpoint Numbers

To quantify the effectiveness of RLHF, we evaluate multiple metrics, including PSNR, SSIM, and LPIPS, across different representation methods. As shown in Tab. 5.2, increasing the number of captured photos generally improves representation quality but also leads to a significant growth in task execution time. Notably, when twenty photos are used, the representation reaches an SSIM of 0.8904 with a PSNR of 22.74, but the average task time rises to 76.21 seconds, which is almost double that of ten photos. In contrast, using ten photos yields a PSNR of 21.83, an SSIM of 0.8760, and an LPIPS of 0.1412, while keeping the task time at 38.91 seconds. Therefore, we select ten photos in the following experiments as it offers the best trade-off between representation accuracy and task efficiency.

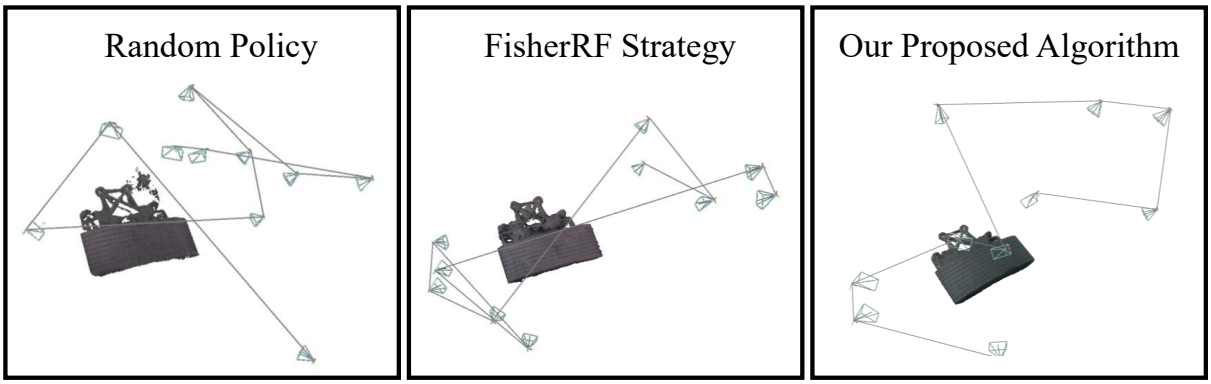

Figure 5.6: Comparative visualization of trajectory efficiency.

Table 5.3: Comparative Analysis of Representation Performance with Different Expert Operators

Average PSNR/Operator	1	2	3	4	5
Ground Truth - 1	22.23	20.89	22.00	21.33	21.33
Ground Truth - 2	21.85	25.28	19.94	20.37	21.52
Ground Truth - 3	21.48	21.38	22.18	20.37	22.34
Ground Truth - 4	22.51	23.16	21.50	23.72	21.85
Ground Truth - 5	21.38	22.68	22.19	21.74	23.35

5.5.5 Evaluations on Different Expert Operators

As shown in Tab. 5.3, to further investigate the impact of operator preferences on representation quality, we collected data from five expert operators. Each operator first evaluated the candidate models and selected according to their own preferences, and the collected data were further utilized to train the Proposed algorithm. Specifically, each operator was asked to choose inspection viewpoints that best reflect their individual preference, and these operator-specific viewpoints were then used to construct five sets of ground truths. Following the proposed framework, each operator completed the experiment with a representative 3D scene representation model (here, instant-ngp is used as a case study), ensuring that the operator-specific data and preferences were consistently integrated into the training and evaluation process. For evaluation, the representation performance was tested against all ground truths in terms of PSNR. In other words, each represented model was rendered from the four preference-specific viewpoints of every operator, and the results were compared against the corresponding ground truths. The results show that when the evaluation is conducted under a fixed ground truth, the model trained with the corresponding operator's preference consistently achieves the highest or near-highest PSNR. This finding indicates that operator-specific preferences are well captured during training, and models adapted to such preferences demonstrate superior performance when evaluated under matching conditions.

5.5.6 Comparison of 3D Scene Representations Quality and Trajectory Efficiency on Different Baselines

As shown in Tab. 5.4, we present a comparative analysis of 3D scene representation quality and trajectory efficiency across different baselines and 3D representation methods. The evaluation metrics are computed as averages between the training ground truth images and the rendered novel views sampled at identical camera poses. To emphasize practical significance, the novel views are selected near regions of interest identified by the expert operator, ensuring that the evaluation highlights areas most critical for operational inspection. The results show that our proposed algorithm achieves consistently shorter path lengths compared with both FisherRF and the random policy, demonstrating improved efficiency in robotic movement planning. In terms of representation quality, the proposed method achieves competitive PSNR values, reaching 22.33 with instant-ngp and 23.54 with 3DGS, which are comparable to or higher than those obtained by FisherRF. Moreover, the perceptual metrics indicate clear advantages: the proposed algorithm maintains low LPIPS values (0.109 with instant-ngp and 0.0426 with 3DGS), reflecting closer alignment with human visual perception, while also sustaining high SSIM scores.

5.5.7 Comparative Visualization of Trajectory Efficiency and Local Fidelity

As illustrated in Fig. 5.6, the trajectory visualization further highlights the smoother and more structured execution of the proposed method compared with other baselines. Collectively, these results confirm that our algorithm not only enhances representation fidelity but also improves trajectory efficiency, leading to reduced operational cost and computational overhead. To further validate RLHF's effectiveness, we check the local rendering quality between our approach FisherRF and random viewpoint selection strategies. Fig. 5.7 highlights instances where our proposed algorithm viewpoint selection achieves sharpest details in occluded or complex regions, whereas random selection often fails to capture fine-grained details. These results confirm that our proposed algorithm not only improves overall scene representation but also enhances local rendering accuracy, making it particularly suitable for applications requiring precise representation of critical areas in nuclear decommissioning and remote inspection tasks.

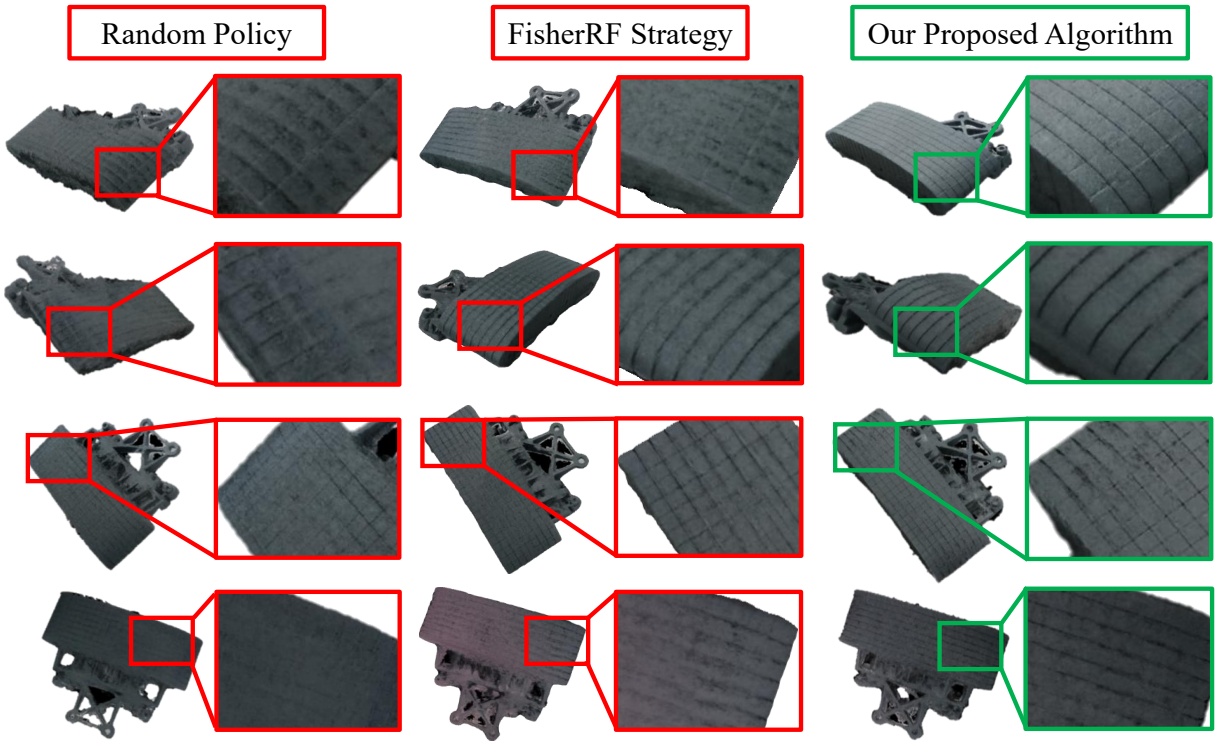

Figure 5.7: Comparative visualization of local fidelity

Table 5.4: Comparative Analysis of Proposed algorithm Performance Across Different 3D Representation Methods

	Random Policy			**FisherRF Strategy**			**Proposed Algorithm**		
Approaches	Instant-ngp	3DGS	PGSR	Instant-ngp	3DGS	PGSR	Instant-ngp	3DGS	PGSR
Average Path Length (m)	3.28			2.61			2.34		
Average PSNR	16.27	18.32	N/A	23.10	23.16	N/A	22.33	23.54	N/A
Average SSIM	0.812	0.8341	N/A	0.9007	0.97	N/A	0.885	0.879	N/A
Average LPIPS	0.234	0.201	N/A	0.1107	0.039	N/A	0.109	0.0426	N/A

5.6 Conclusions

This chapter presented an RLHF-based framework for active 3D scene representation that aligns robotic viewpoint planning with human preferences instead of relying only on task-agnostic geometric or coverage criteria. By placing human feedback inside the viewpoint selection loop, the method closes the gap between classical active vision and the user-specific, safety-aware constraints common in industrial scenarios, allowing the robot to prioritize viewpoints that better support inspection, monitoring, or manipulation even when they are not globally best for coverage.

The core idea is to cast viewpoint planning as preference-informed reinforcement learning: the system proposes alternative viewpoints or rendering sequences, collects human feedback, learns a latent reward capturing subjective factors such as perceived safety, visibility of task-relevant regions, and cognitive ease, and then trains a DRL policy against this learned reward.

Experiments in a nuclear decommissioning setting showed improvements in local scene fidelity on critical surfaces, more efficient viewpoint usage, and better alignment with operator objectives, indicating that preference-aware optimization can act as a practical interface between autonomous planning and human decision-making in safety-critical environments.

Chapter 6

Conclusions and Future Directions

This chapter summarizes the key findings, methodological contributions, and technical insights developed throughout this thesis. Building upon the motivation and challenges introduced in Chapter 1, the proposed frameworks collectively address the critical problem of achieving real-time, task-oriented, and human-adaptive collaboration in CPS and Industrial Metaverse systems. The chapter is organized as follows. Section 6.1 presents the main conclusions and a discussion of the three core research themes investigated in this thesis. Section 6.2 outlines potential future research directions that extend the current work toward scalable, trustworthy, and human-centered cyber–physical intelligence.

6.1 Conclusions and Discussion

This thesis proposed an integrated, human-centered, and task-oriented co-design paradigm that bridges communication, computation, control, and human adaptation in next-generation CPS. Through theoretical modeling, algorithmic development, and prototype validation, the work contributes to a unified understanding of how real-time performance and human experience can be jointly optimized. The major technical contributions are summarized across three main components as follows.

1. Task-Oriented Cross-System Co-Design Framework

Chapter 3 first develops a task-oriented cross-system co-design framework that abandons isolated optimization of communication, computing, and control, and instead jointly decides across these subsystems according to end-to-end task objectives such as MTP latency, control stability, trajectory accuracy, and operator-perceived responsiveness. By coupling DRL-based resource scheduling with adaptive control, the framework can dynamically allocate bandwidth, computing, and rendering/update rates to the current task context while adjusting control parameters to compensate for multi-loop latency and uncertainty, keeping distributed visual rendering and

physical actuation synchronized. Experiments show that, compared with modular baselines, this task-driven co-design significantly reduces MTP latency, improves control stability, and increases task completion accuracy.

2. Human-in-the-Loop Meta-Learning for Adaptive Control

To cope with non-stationarity caused by changing operators, tasks, and environments, Chapter 4 proposes a human-in-the-loop meta-learning control framework built on MAML, which learns an "easy-to-adapt" initialization so that, at deployment, only a small amount of human interaction data and a few updates are needed to specialize to the current human–task pair. Offline meta-training over a distribution of tasks and operators produces parameters that can be quickly personalized, and the online phase uses the operator's demonstrations, corrections, or performance traces for lightweight fine-tuning. In this way, the controller reuses the cross-domain knowledge learned in the co-design stage while achieving personalization without sacrificing real-time performance. Experiments confirm noticeable reductions in tracking error and stable operation even under fluctuating network delay.

3. Preference-Driven Reinforcement Learning in Co-Design CPS

Chapter 5 further places the human at the center of optimization by introducing preference-based feedback and RLHF, enabling the system to learn not only task-oriented rewards but also subjective criteria such as comfort, transparency, and trust. The system generates alternative trajectories or viewpoint sequences, the operator provides pairwise or selective feedback, and a latent reward model is trained from these sparse preferences; a DRL agent then optimizes its policy against this learned reward. The resulting policies explicitly trade off efficiency and human experience, yielding behaviors that operators actually prefer to work with, and laying the groundwork for future multi-user, multi-preference human-centered autonomy.

The research conducted in this thesis was motivated by four key challenges identified in Chapter 1, namely multi-loop latency and uncertainty, cross-system coupling, human adaptability, and preference alignment. By examining the technical developments in Chapters 3–5, each of these challenges can be mapped to a concrete solution pathway, as summarized below.

- **Multi-Loop Latency and Uncertainty.**

  Chapter 3 analyzed that real-time human–machine interaction in CPS/Industrial Metaverse scenarios is constrained by heterogeneous and time-varying delays across communication, computation, rendering, and control loops. To handle this, Chapter 3 proposed a task-oriented modeling and predictive compensation framework, combined with DRL-based adaptive scheduling, so that delay on one loop can be offset by proactive decisions on another loop. Experiments in Chapter 3 showed that this approach reduces motion er-

ror and improves real-time stability under stochastic latency, thus directly addressing the first challenge.

- **Cross-System Coupling and Co-Design.**

  Optimizing communication, computing, and control in isolation cannot guarantee task-level performance because their effects are tightly coupled. Building on the task-oriented view of Chapter 3, Chapter 4 formulated a unified co-design/co-optimization scheme that puts network scheduling, computation offloading/allocation, and control policy adjustment into one task-level objective, enabling explicit trade-offs among latency, reliability, and control accuracy across heterogeneous domains. In this way, the second challenge—lack of a unified cross-system design principle—is resolved through the integrated framework presented in Chapter 4.

- **Human Variability, Adaptation, and Preference Modeling.** As highlighted in Chapter 1, human operators, task demands, and networked environments do not remain stationary, so a fixed control or decision policy cannot guarantee consistent performance. Chapter 4 therefore introduced a HITL-MAML paradigm to learn a generalizable initialization that can be rapidly fine-tuned with only a small amount of operator-specific interaction data. In this way, the co-designed CPS can re-personalize itself when the current operator changes, when the task switches, or when the delay profile drifts, turning the third challenge—coping with human variability and task-dependent adaptation—into a practical online adaptation process.

- **Toward Task-Oriented, Human-Centered CPS.** Chapter 1 also noted that optimizing only for task/physics-based metrics may diverge from what human operators actually prefer in long-term operation. To close this gap, Chapter 5 incorporated RLHF into the co-design CPS pipeline: the system generates alternative behaviors or scene/rendering choices, collects human feedback, learns a latent reward that encodes the operator's subjective criteria, and then uses DRL to optimize policies with respect to this learned reward. This mechanism aligns algorithmic optimization with human perception, completing the progression from task-oriented co-design to truly human-centered CPS and thereby addressing the fourth challenge raised in Chapter 1.

Collectively, the solutions developed in Chapters 3– 5 map one-to-one to the four challenges formulated in Chapter 1, turning them from high-level problem statements into an experimentally validated, task-oriented, and human-centered co-design paradigm for CPS and Industrial Metaverse systems.

6.2 Future Directions

6.2.1 Overall Significance and Limitations

The research presented in this thesis provides a comprehensive exploration of task-oriented and human-centered co-design for next-generation CPS and Industrial Metaverse systems. By integrating communication, computation, control, and human adaptation within a unified optimization framework, the proposed methodologies advance the understanding of how real-time system performance can be jointly optimized with human experience and trust. This unified perspective bridges traditionally isolated research areas—such as networked control systems, edge intelligence, and human–computer interaction—into a coherent foundation for cross-system collaboration and co-adaptive autonomy.

From a theoretical standpoint, the thesis establishes a task-driven modeling and optimization paradigm that links network-level KPIs with human-perceptual metrics such as motion-to-photon latency, control transparency, and operator comfort. The proposed frameworks demonstrate that the integration of meta-learning, human-in-the-loop reinforcement learning, and preference alignment can yield tangible improvements in responsiveness, stability, and human satisfaction across diverse application scenarios. These results contribute to a deeper understanding of how intelligent CPS can evolve from purely reactive systems to human-symbiotic agents capable of continuous learning and adaptation.

Nevertheless, several limitations must be acknowledged. First, while the developed frameworks have been validated through simulations and small-scale experiments, large-scale industrial deployment under realistic network and human variability conditions remains an open challenge. Second, the meta-learning and RLHF mechanisms rely on structured and well-calibrated feedback; in real-world environments, human feedback can be noisy, delayed, or inconsistent, which may affect convergence stability. Third, although this work emphasizes the integration of human factors into optimization, the current formulations still rely on simplified models of human cognition and preference, which may not fully capture the diversity and contextual nuances of real operators. Finally, the computational complexity of multi-agent co-design and online learning presents scalability bottlenecks that must be addressed for real-time implementation in resource-constrained environments.

In summary, while this thesis advances the theoretical and practical understanding of task-oriented and human-adaptive co-design, it also highlights the need for scalable learning architectures, more expressive human modeling, and robust cross-system synchronization mechanisms. These reflections motivate the future research directions discussed in the following subsections.

6.2.2 Multi-Human Interaction and Collaboration

As intelligent CPS continue to evolve, future research must extend beyond single human–machine loops toward scalable multi-human–multi-machine collaboration. In real-world industrial and immersive environments, multiple human operators frequently share control and perception responsibilities with a network of intelligent agents, robots, or digital twins. Such scenarios introduce new challenges in coordination, feedback fusion, and collective adaptation that go beyond the single-operator settings considered in this thesis.

To address these challenges, several promising research directions can be identified:

- **Distributed Intent Inference and Shared Situational Awareness.**

  As the number of human participants grows, coordination mechanisms must infer group intent and shared goals in real time. Traditional hierarchical control architectures are insufficient for modeling dynamic human teaming, where multiple operators may issue concurrent commands or influence collective objectives. Future co-design frameworks should incorporate distributed intent inference, conflict resolution, and cooperative decision-making to maintain coherent task execution [162].

- **Trust Calibration and Dynamic Role Allocation.**

  Multi-human–machine systems require adaptive strategies to manage varying levels of trust, confidence, and workload among operators. Dynamically calibrating trust and reallocating control authority based on individual reliability and task context can ensure stable performance and balanced autonomy across the team.

- **Team-Level Preference Learning.**

  Extending the proposed HITL meta-learning and preference-driven DRL frameworks to multi-human environments opens opportunities for learning collective behaviors [163]. Future work should focus on aggregating multiple, possibly conflicting, human feedback signals into unified team-level reward models. Such mechanisms can enable systems to adapt to both individual and collective objectives while maintaining fairness, responsiveness, and stability.

- **Scalability and Communication Efficiency.** With increasing numbers of human and machine participants, scalability becomes critical [164]. Future research should explore distributed learning architectures and communication-efficient policy sharing to handle high-dimensional feedback, minimize network congestion, and coordinate multi-agent decisions under real-time constraints.

In summary, extending task-oriented co-design toward multi-human collaboration will enable the next generation of human-symbiotic CPS that can adapt not only to individual operators

but also to collaborative teams. This evolution will bridge the gap between personalized intelligence and collective intelligence within the Industrial Metaverse.

6.2.3 Foundation Model-Driven Co-Design and Interaction Optimization

In recent years, the emergence of Foundation Models (FMs) has radically expanded the scope of learning and reasoning capabilities in artificial intelligence. Studies show that FMs can unify multimodal inputs (vision, language, sensor data), enable in-context adaptation, and generalize across tasks and environments beyond those seen during training. In the context of CPS and Industrial Metaverse platforms, these capabilities open new possibilities for interaction optimization, semantic communication, and real-time co-design.

Several future research directions are particularly relevant:

- **Semantic Co-Design and Representation Learning.**

  FM can be leveraged to learn unified representations of communication, computation, control, and human-in-the-loop feedback [165]. Instead of treating network latency, control error, and human preference as isolated metrics, a FM-based system could encode them into a common semantic latent space, enabling more holistic optimization and cross-domain adaptation. Recent surveys emphasise the potential of FMs for decision-making across multimodal inputs.

- **Zero-Shot and Few-Shot Policy Adaptation.**

  A significant limitation of current RL and co-design frameworks is the need for extensive task- or operator-specific data. Foundation models offer an avenue for zero-shot or few-shot adaptation—where the system can generalize to new tasks, new human operators, or new system configurations with minimal additional training [166]. For example, work in robotics shows FMs controlling physical systems with less data than traditional DNN-based approaches.

- **Interactive Human–Machine Communication via FMs.**

  FMs excel at natural-language and multimodal interaction, which can transform how humans and machines collaborate. In CPS contexts, this could mean systems that understand human intent expressed via speech or gesture, translate it into control or scheduling actions, and adapt their behavior accordingly. This pathway aligns perfectly with the HITL and preference-aware research theme. The challenge lies in integrating FMs inference with real-time control loops that demand low latency and high reliability.

- **Scalable Real-Time Co-Design with FMs Guided Simulation and Policy Generation.**

  To support large-scale, heterogeneous systems (many devices, many human operators, many tasks), simulation and policy generation can be enhanced by FMs. For instance, FMs

can generate novel task scenarios, propose candidate policies, or simulate human–machine interaction pathways [166]. This accelerates development and adaptation of co-design strategies in changing environments. Surveys in autonomous driving and industrial systems highlight this trend.

Embedding FM techniques within the co-design of communication, computing, control and human interaction offers a promising frontier. Achieving this will require addressing key challenges such as inference latency, alignment of FMs outputs with control-theoretic constraints, data efficiency, and trust/safety in human-machine feedback loops. If successful, this evolution could significantly enhance the adaptability, generalisation, and semantic coherence of future human-centric CPS and Industrial Metaverse systems.

Reference

[1] N. Jazdi, "Cyber physical systems in the context of industry 4.0," in *2014 IEEE Int. Conf. Autom., Qual. Test., Robot.*, 2014, pp. 1–4.

[2] Y. Liu, L. Zhang, Y. Yang, L. Zhou, L. Ren, F. Wang, R. Liu, Z. Pang, and M. J. Deen, "A novel cloud-based framework for the elderly healthcare services using digital twin," *IEEE Access*, vol. 7, pp. 49 088–49 101, 2019.

[3] E. Glaessgen and D. Stargel, "The digital twin paradigm for future NASA and U.S. air force vehicles," in *Proc. 53rd AIAA/ASME/ASCE/AHS/ASC Struct. Struct. Dyn.*, 2012, pp. 1–14.

[4] T. Samad, *Human-in-the-Loop Control and Cyber–Physical–Human Systems: Applications and Categorization*. John Wiley & Sons, Ltd, 2023, ch. 1, pp. 1–23. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119857433.ch1

[5] Z. Hou, C. She, Y. Li, D. Niyato, M. Dohler, and B. Vucetic, "Intelligent communications for tactile internet in 6g: Requirements, technologies, and challenges," *IEEE Commun. Mag.*, vol. 59, no. 12, pp. 82–88, 2021.

[6] Z. Meng, C. She, G. Zhao, M. A. Imran, M. Dohler, Y. Li, and B. Vucetic, "Task-oriented metaverse design in the 6g era," *IEEE Wireless Commun.*, pp. 1–7, 2024.

[7] A. Islam, A. Debnath, M. Ghose, and S. Chakraborty, "A survey on task offloading in multi-access edge computing," *J. Syst. Arch.*, vol. 118, p. 102225, 2021.

[8] S. Mangiante, G. Klas, A. Navon, Z. GuanHua, J. Ran, and M. D. Silva, "VR is on the edge: How to deliver 360 videos in mobile networks," in *Proc. Workshop Virtual Reality Augmented Reality Netw.*, 2017, pp. 30–35.

[9] S. Fani, S. Ciotti, M. G. Catalano, G. Grioli, A. Tognetti, G. Valenza, and etc., "Simplifying telerobotics: Wearability and teleimpedance improves human-robot interactions in teleoperation," *IEEE Robot. Automat. Mag.*, vol. 25, no. 1, pp. 77–88, 2018.

[10] K. Cao, S. Hu, Y. Shi, A. W. Colombo, S. Karnouskos, and X. Li, "A survey on edge and edge-cloud computing assisted cyber-physical systems," *IEEE Trans. Ind. Inform.*, vol. 17, no. 11, pp. 7806–7819, 2021.

[11] N. Stephenson, *Snow Crash: A Novel*. Random House Worlds, 2003. [Online]. Available: https://books.google.co.uk/books?id=RMd3GpIFxcUC

[12] X. Wei, C. Yang, and S. Han, "Prediction, communication, and computing duration optimization for vr video streaming," *IEEE Trans. Commun.*, vol. 69, no. 3, pp. 1947–1959, 2021.

[13] Z. Guo, Y. Zhang, X. Zhao, and X. Song, "Cps-based self-adaptive collaborative control for smart production-logistics systems," *IEEE Trans. Cybern.*, vol. 51, no. 1, pp. 188–198, 2021.

[14] Z. Hou, C. She, Y. Li, L. Zhuo, and B. Vucetic, "Prediction and communication co-design for ultra-reliable and low-latency communications," *IEEE Trans. Wireless Commun.*, vol. 19, no. 2, pp. 1196–1209, 2020.

[15] K. Gross, K. Baclawski, E. S. Chan, D. Gawlick, A. Ghoneimy, and Z. H. Liu, "A supervisory control loop with prognostics for human-in-the-loop decision support and control applications," in *2017 IEEE Conf. Cognitive Comput. Aspects Situation Manag. (CogSIMA)*. IEEE, 2017, pp. 1–7.

[16] L.-H. Lee, T. Braud, P. Zhou, L. Wang, D. Xu, Z. Lin, A. Kumar, C. Bermejo, and P. Hui, "All one needs to know about metaverse: A complete survey on technological singularity, virtual ecosystem, and research agenda," *arXiv preprint arXiv:2110.05352*, 2021.

[17] G. R. E. Said, "Metaverse-based learning opportunities and challenges: A phenomenological metaverse human–computer interaction study," *Electron.*, vol. 12, no. 6, p. 1379, 2023.

[18] J. Xie, Y. Liu, X. Wang, S. Fang, and S. Liu, "A new xr-based human-robot collaboration assembly system based on industrial metaverse," *J. Manufacturing Syst.*, vol. 74, pp. 949–964, 2024.

[19] H. Yang, X.-Y. Liu, S. Zhong, and A. Walid, "Deep reinforcement learning for automated stock trading: An ensemble strategy," in *Proc. First ACM Int. Conf. AI Finance*, 2020, pp. 1–8.

[20] D. Mitchell, P. D. E. Baniqued, A. Zahid, A. West, B. Nouri Rahmat Abadi, and et al, "Lessons learned: Symbiotic autonomous robot ecosystem for nuclear environments," *IET Cyber-Syst. Robot.*, vol. 5, no. 4, p. e12103, 2023. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/csy2.12103

[21] G. Zhao, M. A. Imran, Z. Pang, Z. Chen, and L. Li, "Toward real-time control in future wireless networks: Communication-control co-design," *IEEE Commun. Mag.*, vol. 57, no. 2, pp. 138–144, 2018.

[22] G. Aceto, V. Persico, and A. Pescapé, "A survey on information and communication technologies for industry 4.0: State-of-the-art, taxonomies, perspectives, and challenges," *IEEE Commun. Surv. Tut.*, vol. 21, no. 4, pp. 3467–3501, 2019.

[23] M. A. Khalighi and M. Uysal, "Survey on free space optical communication: A communication theory perspective," *IEEE Commun. Surv. Tut.*, vol. 16, no. 4, pp. 2231–2258, 2014.

[24] J.-D. Decotignie, "Ethernet-based real-time and industrial communications," *Proc. IEEE*, vol. 93, no. 6, pp. 1102–1117, 2005.

[25] G. Zhao, M. A. Imran, Z. Pang, Z. Chen, and L. Li, "Toward real-time control in future wireless networks: Communication-control co-design," *IEEE Commun. Mag.*, vol. 57, no. 2, pp. 138–144, 2019.

[26] X. Wang, C. Chen, J. He, S. Zhu, and X. Guan, "Aoi-aware control and communication co-design for industrial iot systems," *IEEE Internet Things J.*, vol. 8, no. 10, pp. 8464–8473, 2021.

[27] X. Liu and Y. Deng, "Learning-based prediction, rendering and association optimization for mec-enabled wireless virtual reality (vr) networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 10, pp. 6356–6370, 2021.

[28] W.-j. Kim, K. Ji, and A. Ambike, "Real-time operating environmentfor networked control systems," *IEEE transactions on automation science and engineering*, vol. 3, no. 3, pp. 287–296, 2006.

[29] S. Li, C. She, Y. Li, and B. Vucetic, "Constrained deep reinforcement learning for low-latency wireless vr video streaming," in *Proc. 2021 IEEE Global Commun. Conf. (GLOBECOM)*, 2021, pp. 01–06.

[30] Z. Meng, K. Chen, Y. Diao, C. She, G. Zhao, M. A. Imran, and B. Vucetic, "Task-oriented cross-system design for timely and accurate modeling in the metaverse," *IEEE J. Sel. Areas Commun.*, vol. 42, no. 3, pp. 752–766, 2024.

[31] N. Mitsunaga, C. Smith, T. Kanda, H. Ishiguro, and N. Hagita, "Adapting robot behavior for human–robot interaction," *IEEE Trans. Robotics*, vol. 24, no. 4, pp. 911–916, 2008.

[32] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-learning in neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 5149–5169, 2021.

[33] S. Poudel, A. Serwadda, and V. V. Phoha, "On humanoid robots imitating human touch gestures on the smart phone," in *2015 IEEE 7th Int. Conf. Biometrics Theory, Appl. Syst. (BTAS)*. IEEE, 2015, pp. 1–7.

[34] C. Liebers, N. Pfützenreuter, M. Prochazka, P. Megarajan, E. Furuno, J. Löber, T. C. Stratmann, J. Auda, D. Degraen, U. Gruenefeld *et al.*, "Look over here! comparing interaction methods for user-assisted remote scene reconstruction," in *Proc. Extended Abstr. CHI Conf. Human Factors Comput. Syst.*, 2024, pp. 1–8.

[35] T. Yu, C. Finn, A. Xie, S. Dasari, T. Zhang, P. Abbeel, and S. Levine, "One-shot imitation from observing humans via domain-adaptive meta-learning," *arXiv preprint arXiv:1802.01557*, 2018.

[36] C. Liebers, N. Pfützenreuter, M. Prochazka, P. Megarajan, E. Furuno, J. Löber, T. C. Stratmann, J. Auda, D. Degraen, U. Gruenefeld, and S. Schneegass, "Look over here! comparing interaction methods for user-assisted remote scene reconstruction," in *Proc. Extended Abstr. CHI Conf. Human Factors Comput. Syst.*, ser. CHI EA '24. New York, NY, USA: Association for Computing Machinery, 2024. [Online]. Available: https://doi.org/10.1145/3613905.3650982

[37] H. A. F. Almurib, H. F. Al-Qrimli, and N. Kumar, "A review of application industrial robotic design," in *Proc. 2011 9th Int. Conf. ICT Knowledge Eng.*, 2012, pp. 105–112.

[38] A. Liu, Z. Huang, M. Li, Y. Wan, W. Li, T. X. Han, C. Liu, R. Du, D. K. P. Tan, J. Lu *et al.*, "A survey on fundamental limits of integrated sensing and communication," *IEEE Commun. Surv. Tut.*, vol. 24, no. 2, pp. 994–1034, 2022.

[39] Z.-S. Hou and Z. Wang, "From model-based control to data-driven control: Survey, classification and perspective," *Inform. Sci.*, vol. 235, pp. 3–35, 2013.

[40] X.-M. Zhang, Q.-L. Han, X. Ge, D. Ding, L. Ding, D. Yue, and C. Peng, "Networked control systems: A survey of trends and techniques," *IEEE/CAA J. Automatica Sinica*, vol. 7, no. 1, pp. 1–17, 2019.

[41] M. K. Habib, C. Chimsom *et al.*, "Cps: Role, characteristics, architectures and future potentials," *Procedia Computer Science*, vol. 200, pp. 1347–1358, 2022.

[42] Z. Zhao, J. Perazzone, G. Verma, and S. Segarra, "Congestion-aware distributed task offloading in wireless multi-hop networks using graph neural networks," in *ICASSP*

2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2024, pp. 8951–8955.

[43] "Study on scenarios and requirements for next generation access technologies," document 3GPP, TSG RAN TR38.913 R14, Jun. 2017.

[44] M. Felser, "Real-time ethernet-industry prospective," *Proceedings of the IEEE*, vol. 93, no. 6, pp. 1118–1129, 2005.

[45] P. Zand, S. Chatterjea, K. Das, and P. Havinga, "Wireless industrial monitoring and control networks: The journey so far and the road ahead," *Journal of sensor and actuator networks*, vol. 1, no. 2, pp. 123–152, 2012.

[46] L. Seno, F. Tramarin, and S. Vitturi, "Performance of industrial communication systems: Real application contexts," *IEEE Ind. Electron. Mag.*, vol. 6, no. 2, pp. 27–37, 2012.

[47] J. Park, S. Lee, T. Yoon, and J. M. Kim, "An autonomic control system for high-reliable cps," *Cluster Computing*, vol. 18, no. 2, pp. 587–598, 2015.

[48] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[49] H. Cheng, M. Zhang, and J. Q. Shi, "A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2024.

[50] T. Liang, J. Glossner, L. Wang, S. Shi, and X. Zhang, "Pruning and quantization for deep neural network acceleration: A survey," *Neurocomputing*, vol. 461, pp. 370–403, 2021.

[51] B. Chang, W. Tang, X. Yan, X. Tong, and Z. Chen, "Integrated scheduling of sensing, communication, and control for mmwave/thz communications in cellular connected uav networks," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 7, pp. 2103–2113, 2022.

[52] Z. Meng, C. She, G. Zhao, and D. De Martini, "Sampling, communication, and prediction co-design for synchronizing the real-world device and digital model in metaverse," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 1, pp. 288–300, 2023.

[53] B. Akesson, M. Nasri, G. Nelissen, S. Altmeyer, and R. I. Davis, "A comprehensive survey of industry practice in real-time systems," *Real-Time Syst.*, vol. 58, no. 3, pp. 358–398, 2022.

[54] H. Laaki, Y. Miche, and K. Tammi, "Prototyping a digital twin for real time remote control over mobile networks: Application of remote surgery," *IEEE Access*, vol. 7, pp. 20 325–20 336, 2019.

[55] D. Calvaresi, M. Marinoni, A. Sturm, M. Schumacher, and G. Buttazzo, "The challenge of real-time multi-agent systems for enabling iot and cps," in *Proc. Int. Conf. Web Intell.*, 2017, pp. 356–364.

[56] B. Shen and et al., "Dynamic task offloading and online scheduling for edge computing systems in multi-access iot," *Future Gener. Comput. Syst.*, vol. 50, pp. 45–58, 2025.

[57] K. Ravindran, A. Adiththan, and S. Mukhopadhyay, "Stability of networked control systems with time-varying transmission period," in *Proc. 2014 Int. Conf. Cyber-Physical Syst. (ICCPS)*, 2014, pp. 25–34.

[58] L. Ren, X. Ning, and J. Li, "Hierarchical reinforcement-learning for real-time scheduling of agile satellites," *IEEE Access*, vol. 8, pp. 220 523–220 532, 2020.

[59] X. Zhao, Y. Sun, Y. Li, N. Jia, and J. Xu, "Applications of machine learning in real-time control systems: a review," *Meas. Sci. Technol.*, 2024.

[60] A. Mostaani, T. X. Vu, S. K. Sharma, V.-D. Nguyen, Q. Liao, and S. Chatzinotas, "Task-oriented communication design in cyber-physical systems: A survey on theory and applications," *IEEE Access*, vol. 10, pp. 133 842–133 868, 2022.

[61] J. Cao, E. Kurniawan, A. Boonkajay, and S. Sun, "Goal-oriented scheduling and control co-design in wireless networked control systems," in *Proc. 2023 IEEE Global Commun. Conf. (GLOBECOM)*, 2023, pp. 1090–1095.

[62] N. Pappas and M. Kountouris, "Goal-oriented communication for real-time tracking in autonomous systems," in *2021 IEEE Int. Conf. Auton. Syst. (ICAS)*, 2021, pp. 1–5.

[63] J. Shao, T. Li, and J. Zhang, "Task-oriented communication for vehicle-to-infrastructure cooperative perception," 2024. [Online]. Available: https://arxiv.org/abs/2407.20748

[64] W. Wu, Y. Wu, Y. Yang, and Y. Deng, "Goal-oriented UAV communication design and optimization for target tracking: A machine learning approach," *IEEE Commun. Lett.*, pp. 1–1, 2024.

[65] S. M. Sepasgozar, "Differentiating digital twin from digital shadow: Elucidating a paradigm shift to expedite a smart, sustainable built environment," *Buildings*, vol. 11, no. 4, p. 151, 2021.

[66] K. Chen, Z. Meng, X. Xu, C. She, and P. G. Zhao, "Real-time interactions between human controllers and remote devices in metaverse," 2024. [Online]. Available: https://arxiv.org/abs/2407.16591

[67] H. Chetto, M. Silly, and T. Bouchentouf, "Dynamic scheduling of real-time tasks under precedence constraints," *Real-Time Syst.*, vol. 2, no. 3, pp. 181–194, 1990.

[68] X. Jin, H. H. Liu, R. Gandhi, S. Kandula, R. Mahajan, M. Zhang, J. Rexford, and R. Wattenhofer, "Dynamic scheduling of network updates," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 4, pp. 539–550, 2014.

[69] J. A. Stankovic, M. Spuri, K. Ramamritham, and G. Buttazzo, *Deadline scheduling for real-time systems: EDF and related algorithms*. Springer Science & Business Media, 1998, vol. 460.

[70] A. Raake, "Short-and long-term packet loss behavior: towards speech quality prediction for arbitrary loss distributions," *IEEE Trans. Audio, Speech, and Lang. Process.*, vol. 14, no. 6, pp. 1957–1968, 2006.

[71] N. Ahmad, Y. Ghadi, M. Adnan, and M. Ali, "Load forecasting techniques for power system: Research challenges and survey," *IEEE Access*, vol. 10, pp. 71 054–71 090, 2022.

[72] A. Aljohani, "Predictive analytics and machine learning for real-time supply chain risk mitigation and agility," *Sustainability*, vol. 15, no. 20, p. 15088, 2023.

[73] Y. Liu, P. Zeng, J. Cui, and C. Xia, "Co-design of control, computation, and network scheduling based on reinforcement learning," *IEEE Internet Things J.*, vol. 11, no. 3, pp. 5249–5258, 2023.

[74] D. Elias, D. Ziegenbein, P. Mundhenk, A. Hamann, and A. Rowe, "The cyber-physical metaverse - where digital twins and humans come together," in *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2023, pp. 1–2.

[75] A. Elhagry and A. El Saddik, "Text-to-metaverse: Towards a digital twin-enabled multimodal conditional generative metaverse," in *2023 IEEE International Conference on Metaverse Computing, Networking and Applications (MetaCom)*, 2023, pp. 666–669.

[76] Y. Han, D. Niyato, C. Leung, D. I. Kim, K. Zhu, S. Feng, X. Shen, and C. Miao, "A dynamic hierarchical framework for iot-assisted digital twin synchronization in the metaverse," *IEEE Internet Things J.*, vol. 10, no. 1, pp. 268–284, 2023.

[77] A. Humayed, J. Lin, F. Li, and B. Luo, "Cyber-physical systems security—a survey," *IEEE Internet Things J.*, vol. 4, no. 6, pp. 1802–1831, 2017.

[78] M. Andronie, G. Lăzăroiu, M. Iatagan, I. Hurloiu, R. Ștefănescu, A. Dijmărescu, and I. Dijmărescu, "Big data management algorithms, deep learning-based object detection technologies, and geospatial simulation and sensor fusion tools in the internet of robotic things," *ISPRS Int. J. Geo-Information*, vol. 12, no. 2, p. 35, 2023.

[79] T. Qiu, J. Chi, X. Zhou, Z. Ning, M. Atiquzzaman, and D. O. Wu, "Edge computing in industrial internet of things: Architecture, advances and challenges," *IEEE Commun. Surv. Tut.*, vol. 22, no. 4, pp. 2462–2488, 2020.

[80] J. Yang, A. A. Shah, and D. Pezaros, "A survey of energy optimization approaches for computational task offloading and resource allocation in mec networks," *Electron.*, vol. 12, no. 17, p. 3548, 2023.

[81] S. Xie, H. He, S. Song, J. Zhang, and K. B. Letaief, "Toward real-time edge ai: Model-agnostic task-oriented communication with visual feature alignment," *arXiv preprint arXiv:2412.00862*, 2024.

[82] C. Moro, G. Nejat, and A. Mihailidis, "Learning and personalizing socially assistive robot behaviors to aid with activities of daily living," *ACM Trans. Human-Robot Interact. (THRI)*, vol. 7, no. 2, pp. 1–25, 2018.

[83] P. M. d. Santana, N. Marchenko, B. Soret, and P. Popovski, "Goal-oriented wireless communication for a remotely controlled autonomous guided vehicle," *IEEE Wireless Commun. Lett.*, 2023.

[84] Y. Lu, S. Maharjan, and Y. Zhang, "Adaptive edge association for wireless digital twin networks in 6G," *IEEE Internet Things J.*, vol. 8, no. 22, pp. 16 219–16 230, 2021.

[85] S. Liu, Y. Yu, X. Lian, Y. Feng, C. She, P. L. Yeoh, L. Guo, B. Vucetic, and Y. Li, "Dependent task scheduling and offloading for minimizing deadline violation ratio in mobile edge computing networks," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 2, pp. 538–554, 2023.

[86] L. Fang, J. Wan, H. Cai, S. Wang, Z. Pang, M. Safran, and S. A. AlQahtani, "A scalable cloud–edge collaborative approach for intelligent low-code fault diagnosis: Successful applications of agile migration deployment in heterogeneous fault diagnosis scenarios," *IEEE Ind. Electron. Mag.*, pp. 2–15, 2024.

[87] J. Jin, K. Yu, J. Kua, N. Zhang, Z. Pang, and Q.-L. Han, "Cloud-fog automation: Vision, enabling technologies, and future research directions," *IEEE Trans. Ind. Informat.*, vol. 20, no. 2, pp. 1039–1054, 2024.

[88] X. Li, B. Chen, J. Fan, J. Kang, J. Ye, X. Wang, and D. Niyato, "Cloud-edge-end collaborative intelligent service computation offloading: A digital twin driven edge coalition approach for industrial IoT," *IEEE Trans. Netw. Serv. Manag.*, pp. 1–1, 2024.

[89] K. Wang, J. Jin, Y. Yang, T. Zhang, A. Nallanathan, C. Tellambura, and B. Jabbari, "Task offloading with multi-tier computing resources in next generation wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 2, pp. 306–319, 2023.

[90] M. Xu, D. Niyato, J. Kang, Z. Xiong, S. Mao, Z. Han, D. I. Kim, and K. B. Letaief, "When large language model agents meet 6G networks: Perception, grounding, and alignment," *IEEE Wireless Commun.*, pp. 1–9, 2024.

[91] R. Zhang, H. Du, Y. Liu, D. Niyato, J. Kang, Z. Xiong, A. Jamalipour, and D. I. Kim, "Generative AI agents with large language model for satellite networks via a mixture of experts transmission," *IEEE J. Sel. Areas Commun.*, pp. 1–1, 2024.

[92] L. U. Khan, M. Guizani, D. Niyato, A. Al-Fuqaha, and M. Debbah, "Metaverse for wireless systems: Architecture, advances, standardization, and open challenges," 2023. [Online]. Available: https://arxiv.org/abs/2301.11441

[93] K. Wang, J. Jin, Y. Yang, T. Zhang, A. Nallanathan, C. Tellambura, and B. Jabbari, "Task offloading with multi-tier computing resources in next generation wireless networks," 2022. [Online]. Available: https://arxiv.org/abs/2205.13866

[94] O. Hashash, C. Chaccour, and W. Saad, "Edge continual learning for dynamic digital twins over wireless networks," 2022.

[95] J. Shao, Y. Mao, and J. Zhang, "Learning task-oriented communication for edge inference: An information bottleneck approach," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 1, pp. 197–211, 2021.

[96] S. K. Jagatheesaperumal, Z. Yang, Q. Yang, C. Huang, W. Xu, M. Shikh-Bahaei, and Z. Zhang, "Semantic-aware digital twin for metaverse: A comprehensive review," 2023. [Online]. Available: https://arxiv.org/abs/2305.18304

[97] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. Int. Conf. Mach. Learn.* PMLR, 2017, pp. 1126–1135.

[98] T. Kaufmann, P. Weng, V. Bengs, and E. Hüllermeier, "A survey of reinforcement learning from human feedback," 2024. [Online]. Available: https://arxiv.org/abs/2312.14925

[99] K. Chen, Z. Meng, X. Xu, J. Yang, E. Li, and P. G. Zhao, "Task-oriented edge-assisted cross-system design for real-time human-robot interaction in industrial metaverse," *arXiv preprint arXiv:2508.20664*, 2025.

[100] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," *Advances Neural Inf. Process. Syst.*, vol. 30, 2017.

[101] M. Maier, M. Chowdhury, B. P. Rimal, and D. P. Van, "The tactile internet: vision, recent progress, and open challenges," *IEEE Commun. Mag.*, vol. 54, no. 5, pp. 138–145, 2016.

[102] J. Schulz, C. Dubslaff, P. Seeling, S.-C. Li, S. Speidel, and F. H. P. Fitzek, "Negative latency in the tactile internet as enabler for global metaverse immersion," *IEEE Netw.*, pp. 1–1, 2024.

[103] X. Hou, J. Zhang, M. Budagavi, and S. Dey, "Head and body motion prediction to enable mobile vr experiences with low latency," in *Proc. 2019 IEEE Global Commun. Conf. (GLOBECOM)*, 2019, pp. 1–7.

[104] A. Kosta, N. Pappas, and V. Angelakis, "Age of information: A new concept, metric, and tool," *Foundations and Trends® in Networking*, vol. 12, no. 3, pp. 162–259, 2017. [Online]. Available: http://dx.doi.org/10.1561/1300000060

[105] B. Chang, L. Li, G. Zhao, Z. Meng, M. A. Imran, and Z. Chen, "Age of information for actuation update in real-time wireless control systems," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, 2020, pp. 26–30.

[106] J. Zhao, R. S. Allison, M. Vinnikov, and S. Jennings, "Estimating the motion-to-photon latency in head mounted displays," in *Proc. 2017 IEEE Virtual Reality (VR)*. IEEE, 2017, pp. 313–314.

[107] "Touch Haptic Device," https://www.3dsystems.com/haptics-devices/touch, (accessed Oct. 2023).

[108] J. B. Kuipers, *Quaternions and rotation sequences: a primer with applications to orbits, aerospace, and virtual reality*. Princeton Univ. Press, 1999.

[109] B. Choi, *ARMA model identification*. Springer Science & Business Media, 2012.

[110] "What is Isaac Sim?" https://docs.omniverse.nvidia.com/isaac-sim/latest/index.html, accessed Feb. 2024.

[111] N. D. Ratliff, J. Issac, D. Kappler, S. Birchfield, and D. Fox, "Riemannian motion policies," *arXiv preprint arXiv:1801.02854*, 2018.

[112] "Motion Generation," https://docs.omniverse.nvidia.com/isaacsim/latest/concepts/motion_generation/index.html, (accessed Feb. 2024).

[113] "UR3e: Ultra-Lightweight, Compact Cobot," https://www.universal-robots.com/products/ur3-robot/, accessed Feb. 2024.

[114] "JointPositionController," http://wiki.ros.org/robot_mechanism_controllers/JointPositionController, (accessed Aug. 2024).

[115] "ROS/Introduction," https://wiki.ros.org/ROS/Introduction, (accessed Aug. 2018).

[116] "ROS/UDPROS," https://wiki.ros.org/ROS/UDPROS, (accessed Aug. 2018).

[117] " Linux Advanced Routing & Traffic Control," https://lartc.org/howto/, (accessed Jun. 2012).

[118] "Proximal Policy Optimization," https://spinningup.openai.com/en/latest/algorithms/ppo.html, (accessed Jan. 2020).

[119] N. Promwongsa, A. Ebrahimzadeh, D. Naboulsi, S. Kianpisheh, F. Belqasmi, R. Glitho, N. Crespi, and O. Alfandi, "A comprehensive survey of the tactile internet: State-of-the-art and research directions," *IEEE Commun. Surv. Tut.*, vol. 23, no. 1, pp. 472–523, 2021.

[120] D. Gündüz, Z. Qin, I. E. Aguerri, H. S. Dhillon, Z. Yang, A. Yener, K. K. Wong, and C.-B. Chae, "Beyond transmitting bits: Context, semantics, and task-oriented communications," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 1, pp. 5–41, 2022.

[121] M. Costa, M. Codreanu, and A. Ephremides, "On the age of information in status update systems with packet management," *IEEE Trans. Inf. Theory*, vol. 62, no. 4, pp. 1897–1910, 2016.

[122] "Dell 32 Inch 4K UHD Gaming Monitor," https://www.3dsystems.com/haptics-devices/touch, (accessed Sep. 2024).

[123] "Isaac Cortex Overview," https://docs.omniverse.nvidia.com/isaacsim/latest/cortex_tutorials/tutorial_cortex_1_overview.html, (accessed Jul. 2024).

[124] "ROS/Patterns/Communication," https://wiki.ros.org/ROS/Patterns/Communication, (accessed Aug. 2018).

[125] V. Patil and M. Hutter, "Radiance fields for robotic teleoperation," in *Proc. 2024 IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*. IEEE, 2024, pp. 13 861–13 868.

[126] T. Chen, O. Shorinwa, J. Bruno, A. Swann, J. Yu, W. Zeng, K. Nagami, P. Dames, and M. Schwager, "Splat-nav: Safe real-time robot navigation in gaussian splatting maps," 2025. [Online]. Available: https://arxiv.org/abs/2403.02751

[127] G. Lu, S. Zhang, Z. Wang, C. Liu, J. Lu, and Y. Tang, "Manigaussian: Dynamic gaussian splatting for multi-task robotic manipulation," in *Proc. European Conf. Comput. Vision (ECCV)*. Springer, 2024, pp. 349–366.

[128] S. Zhu, G. Wang, X. Kong, D. Kong, and H. Wang, "3D gaussian splatting in robotics: A survey," 2024. [Online]. Available: https://arxiv.org/abs/2410.12262

[129] M. Strong, B. Lei, A. Swann, W. Jiang, K. Daniilidis, and M. Kennedy III, "Next best sense: Guiding vision and touch with fisherrf for 3D gaussian splatting," 2024. [Online]. Available: https://arxiv.org/abs/2410.04680

[130] S. Ye, Y. He, M. Lin, J. Sheng, R. Fan, Y. Han, Y. Hu, R. Yi, Y.-H. Wen, Y.-J. Liu *et al.*, "Pvp-recon: Progressive view planning via warping consistency for sparse-view surface reconstruction," *ACM Trans. Graph.*, vol. 43, no. 6, pp. 1–13, 2024.

[131] S. Lee, L. Chen, J. Wang, A. Liniger, S. Kumar, and F. Yu, "Uncertainty guided policy for active robotic 3D reconstruction using neural radiance fields," *IEEE Robot. Automat. Lett.*, vol. 7, no. 4, pp. 12 070–12 077, 2022.

[132] D. Mitchell, P. D. Emor Baniqued, A. Zahid, A. West, B. Nouri Rahmat Abadi, B. Lennox, B. Liu, B. Kizilkaya, D. Flynn, D. J. Francis *et al.*, "Lessons learned: Symbiotic autonomous robot ecosystem for nuclear environments," *IET Cyber-Syst. Robot.*, vol. 5, no. 4, p. e12103, 2023.

[133] H. Stedman, B. B. Kocer, N. van Zalk, M. Kovac, and V. M. Pawar, "Evaluating immersive teleoperation interfaces: Coordinating robot radiation monitoring tasks in nuclear facilities," in *Proc. 2023 IEEE Int. Conf. Robotics. Automat. (ICRA)*, 2023, pp. 11 972–11 978.

[134] S. Pacheco-Gutierrez, H. Niu, I. Caliskanelli, and R. Skilton, "A multiple level-of-detail 3D data transmission approach for low-latency remote visualisation in teleoperation tasks," *Robot.*, vol. 10, no. 3, p. 89, 2021.

[135] E. J. Lopez Pulgarin, O. Tokatli, G. Burroughes, and G. Herrmann, "Assessing telemanipulation systems using task performance for glovebox operations," *Frontiers Robot. AI*, vol. 9, p. 932538, 2022.

[136] P. D. E. Baniqued, P. Bremner, M. Sandison, S. Harper, S. Agrawal, J. Bolarinwa, J. Blanche, Z. Jiang, T. Johnson, D. Mitchell *et al.*, "Multimodal immersive digital twin platform for cyber–physical robot fleets in nuclear environments," *J Field Robot.*, vol. 41, no. 5, pp. 1521–1540, 2024.

[137] W. R. Scott, G. Roth, and J.-F. Rivest, "View planning for automated three-dimensional object reconstruction and inspection," vol. 35, no. 1, p. 64–96, Mar. 2003. [Online]. Available: https://doi.org/10.1145/641865.641868

[138] S. Chen, Y. Li, and N. M. Kwok, "Active vision in robotic systems: A survey of recent developments," *Int. J. Robot. Res.*, vol. 30, no. 11, pp. 1343–1377, 2011.

[139] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Commun. ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[140] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "Deepsdf: Learning continuous signed distance functions for shape representation," in *Proc. IEEE/CVF Conf. Comput. Vision Pattern Recognit. (CVPR)*, 2019, pp. 165–174.

[141] L. Jin, X. Chen, J. Rückin, and M. Popović, "Neu-nbv: Next best view planning using uncertainty estimation in image-based neural rendering," in *Proc. 2023 IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*. IEEE, 2023, pp. 11 305–11 312.

[142] X. Chen, Q. Li, T. Wang, T. Xue, and J. Pang, "Gennbv: Generalizable next-best-view policy for active 3D reconstruction," in *Proc. IEEE/CVF Conf. Comput. Vision Pattern Recognit. (CVPR)*, June 2024, pp. 16 436–16 445.

[143] W. Jiang, B. Lei, and K. Daniilidis, "Fisherrf: Active view selection and mapping with radiance fields using fisher information," in *Proc. European Conf. Comput. Vision (ECCV)*. Springer, 2025, pp. 422–440.

[144] L. Chen, W. Chen, R. Wang, and M. Pollefeys, "Leveraging neural radiance fields for uncertainty-aware visual localization," in *Proc. 2024 IEEE Int. Conf. Robot. Automat. (ICRA)*. IEEE, 2024, pp. 6298–6305.

[145] X. Long, C. Lin, P. Wang, T. Komura, and W. Wang, "Sparseneus: Fast generalizable neural surface reconstruction from sparse views," in *Proc. European Conf. Comput. Vision (ECCV)*. Springer, 2022, pp. 210–227.

[146] Y. Xu, B. Liu, H. Tang, B. Deng, and S. He, "Learning with unreliability: Fast few-shot voxel radiance fields with relative geometric consistency," in *Proc. IEEE/CVF Conf. Comput. Vision Pattern Recognit. (CVPR)*, 2024, pp. 20 342–20 351.

[147] H. Jiang, B. Huang, R. Wu, Z. Li, S. Garg, H. Nayyeri, S. Wang, and Y. Li, "Roboexp: Action-conditioned scene graph via interactive exploration for robotic manipulation," 2024. [Online]. Available: https://arxiv.org/abs/2402.15487

[148] R. Firoozi, J. Tucker, S. Tian, A. Majumdar, J. Sun, W. Liu, Y. Zhu, S. Song, A. Kapoor, K. Hausman *et al.*, "Foundation models in robotics: Applications, challenges, and the future," *Int. J. Robot. Res.*, p. 02783649241281508, 2023.

[149] Z. Qi, S. Yuan, F. Liu, H. Cao, T. Deng, J. Yang, and L. Xie, "Air-embodied: An efficient active 3DGS-based interaction and reconstruction framework with embodied large language model," *arXiv preprint arXiv:2409.16019*, 2024.

[150] J. T. Kajiya and B. P. Von Herzen, "Ray tracing volume densities," in *Proc. ACM SIGGRAPH Comput. Graph. (SIGGRAPH)*, vol. 18, no. 3, 1984, pp. 165–174.

[151] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Trans. Graph. (TOG)*, vol. 41, no. 4, pp. 1–15, 2022.

[152] R. Khanam and M. Hussain, "Yolov11: An overview of the key architectural enhancements," *arXiv preprint arXiv:2410.17725*, 2024.

[153] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo *et al.*, "Segment anything," in *Proc. the IEEE/CVF Int. Conf. Comput. Vision (ICCV)*, 2023, pp. 4015–4026.

[154] "RealSense Depth and Tracking Cameras 2025," https://www.intelrealsense.com/depth-camera-d435i/, accessed Jan. 2025.

[155] I. Chapman and A. Morris, "Ukaea capabilities to address the challenges on the path to delivering fusion power," *Philos. Trans. Roy. Soc. A*, vol. 377, no. 2141, p. 20170436, 2019.

[156] T. Austin-Morgan, "Re-tiling a fusion reactor," https://www.eurekamagazine.co.uk/content/technology/re-tiling-a-fusion-reactor/, accessed Feb. 2024.

[157] "moveit calibration: Hand-eye calibration tools for robot arms," https://github.com/moveit/moveit_calibration/, accessed Feb. 2024.

[158] UKAEA, "Remote applications in challenging environments," https://race.ukaea.uk/, accessed Feb. 2024.

[159] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Trans. Graph.*, vol. 41, no. 4, pp. 102:1–102:15, Jul. 2022. [Online]. Available: https://doi.org/10.1145/3528223.3530127

[160] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3D gaussian splatting for real-time radiance field rendering." *ACM Trans. Graph.*, vol. 42, no. 4, pp. 139–1, 2023.

[161] D. Chen, H. Li, W. Ye, Y. Wang, W. Xie, S. Zhai, N. Wang, H. Liu, H. Bao, and G. Zhang, "Pgsr: Planar-based gaussian splatting for efficient and high-fidelity surface reconstruction," *IEEE Tran. Visualization Comput. Graph.*, p. 1–12, 2024. [Online]. Available: http://dx.doi.org/10.1109/TVCG.2024.3494046

[162] A. Dahiya, A. M. Aroyo, K. Dautenhahn, and S. L. Smith, "A survey of multi-agent human-robot interaction systems," 2022. [Online]. Available: https://arxiv.org/abs/2212.05286

[163] M. Zhao, R. Simmons, and H. Admoni, "The role of adaptation in collective human–ai teaming," *Topics in cognitive science*, vol. 17, no. 2, pp. 291–323, 2025.

[164] W. Duan, C. Flathmann, N. McNeese, M. J. Scalia, R. Zhang, J. Gorman, G. Freeman, S. Zhou, A. I. Hauptman, and X. Yin, "Trusting autonomous teammates in human-ai teams-a literature review," in *Proc. 2025 CHI Conf. Human Factors Comput. Syst.*, 2025, pp. 1–23.

[165] K. Kawaharazuka, T. Matsushima, A. Gambardella, J. Guo, C. Paxton, and A. Zeng, "Real-world robot applications of foundation models: A review," *Adv. Robot.*, vol. 38, no. 18, pp. 1232–1254, 2024.

[166] R. Firoozi, J. Tucker, S. Tian, A. Majumdar, J. Sun, W. Liu, Y. Zhu, S. Song, A. Kapoor, K. Hausman *et al.*, "Foundation models in robotics: Applications, challenges, and the future," *The Int. J. Robot. Res.*, vol. 44, no. 5, pp. 701–739, 2025.