AR-Based, Gesture Guided Human Robot Interaction

Akshay Naik
askhayn3@illinois.edu

Jay Mahajan
jaym2@illinois.edu

Samuel Folorunsho
sof3@illinois.edu

***Abstract*—This paper presents an integrated framework for Human-Robot Interaction (HRI) that leverages Virtual Reality (VR) to intuitively control a robotic manipulator. We address the challenge of remote teleoperation by providing a human operator with an immersive, bird's-eye view of the workspace, enabling them to guide a UR3 robot using natural hand gestures. The system architecture bridges a Unity-based VR interface with a ROS-based robot controller via a custom HTTP middleware, ensuring robust communication and real-time feedback. Perception is handled through overhead camera streaming and homography-based calibration, while motion planning utilizes an occupancy grid-based A* algorithm to ensure collision-free trajectories. We validate our approach through a series of pick-and-place experiments, achieving a 100% completion rate in low-to-medium clutter environments and an 80% success rate in highly cluttered scenarios. The system maintained a consistent communication latency of approximately 2.28 s and ensured a minimum safety clearance of at least 8.00 px from obstacles across all trials. We have included the project Github rAkshayepo[1] and a video of an example run.[2]**

***Index Terms*—Human-Robot Interaction, Virtual Reality, Teleoperation, Motion Planning, Gesture-Guided-Control.**

I. Introduction

Robotic manipulators are increasingly deployed in environments where direct human access is unsafe, impractical, or inefficient (e.g., hazardous handling, confined industrial workcells, and remote servicing). In such settings, *teleoperation* remains a practical approach because it retains human judgment for unstructured tasks, yet reduces operator exposure. However, conventional teleoperation interfaces frequently impose nontrivial cognitive load due to limited depth cues, indirect viewpoint control, and complex device-specific input mappings.

Extended Reality (XR), including Augmented Reality (AR) and Mixed Reality (MR), offers a pathway to improve teleoperation usability by aligning perception and action within an immersive 3D interface. Prior surveys show sustained interest in XR-enhanced telerobotics as a means to improve situation awareness, reduce mental workload, and provide more intuitive robot programming and control abstractions [1], [2]. In parallel, commodity head-mounted displays (HMDs) now provide robust inside-out tracking and markerless hand tracking, enabling controller-free interactions suitable for rapid prototyping and deployment on low-cost hardware platforms [3], [4].

[1]https://github.com/akshayn3/UR3_VR_teleop

[2]https://drive.google.com/file/d/1JQicGon3gMT7v2XJytUEQzuHEKDI03KX/view?usp=share_link

This project investigates an **AR-based, gesture-guided human–robot interaction** workflow for commanding a robot arm using only natural hand motion and simple hand gestures. Our prototype is built on the Meta Quest 3 HMD and a UR3-class manipulator. The operator views the robot workcell through a live camera stream rendered in the headset, and interacts with virtual overlays to specify task goals and motion intent. To ensure safe and reliable execution in cluttered workspaces, the system incorporates local motion reasoning (e.g., replanning around obstacles and enforcing workspace constraints) rather than relying solely on direct end-effector "dragging".

A. Contributions

This paper presents an end-to-end AR-based, gesture-guided human–robot interaction system and makes the following contributions:

- **Controller-free gesture interface:** A lightweight command set using hand tracking and gestures to support common teleoperation intents (target selection, confirmation, pause/stop, and speed modulation) on a consumer HMD [3], [4].
- **AR-guided motion specification:** An interaction flow that couples user-specified goals with robot motion generation, enabling path guidance and constraint-aware execution comparable in spirit to prior AR programming and teleoperation systems [5]–[7].
- **Evaluation-ready instrumentation:** Logging hooks and evaluation scaffolding (e.g., task completion time, ordering accuracy, and subjective workload via NASA-TLX) to support quantitative comparison against alternative interfaces [8].

The remainder of this paper describes the problem formulation and constraints, system architecture and implementation details, and an empirical evaluation plan and results.

II. Related Work

This section situates our AR-based, gesture-guided teleoperation project within prior work on (i) XR interfaces for robot teleoperation and programming, (ii) gesture-based path specification and intent communication, and (iii) evaluation methods for operator workload and usability.

A. XR Interfaces for Teleoperation and Robot Programming

XR has been widely explored as an interface layer for improving robot teleoperation and programming by increasing

spatial awareness and reducing the indirection inherent to 2D displays. Recent reviews summarize XR-enhanced telerobotic systems and highlight recurring design themes such as viewpoint management, virtual fixtures, digital twins, and mixed autonomy [1]. In the broader AR+robotics landscape, Suzuki *et al.* provide a taxonomy spanning AR-enhanced HRI and robotic interfaces, including common visualization and interaction modalities relevant to teleoperation [2].

A representative class of systems uses AR to define or preview robot motion trajectories. Quintero *et al.* demonstrate AR-based trajectory specification and visualization for robot programming, emphasizing preview and interactive refinement [5]. Other systems focus on direct teleoperation through AR overlays and natural user inputs. For example, *Hands-Free* enables end-effector control using vision-based hand tracking and AR cues [6]. In high-consequence environments, Regal *et al.* present AR remote operation of dual-arm manipulators using a HoloLens-based interface with environmental perception aids [7]. More recently, Wang *et al.* report an AR-enhanced teleoperation approach that emphasizes natural interaction metaphors and alignment between virtual and physical frames [9].

Our system aligns with these efforts in using AR overlays to support intuitive control, but emphasizes *controller-free* interaction on a commodity headset and integrates local constraint handling for safer execution in cluttered workcells.

B. Gesture-Guided Intent and Path Specification

Gesture-driven interfaces have been explored both for specifying robot motion and for conveying operator intent. Vysocký *et al.* implement and compare hand-gesture interfaces for collaborative robot path definition, illustrating how gesture vocabularies can support efficient waypointing and trajectory authoring [10]. In teleoperation assistance, Krishnan *et al.* study which AR visual cues users prefer under different autonomy levels, showing that cue selection and presentation materially affect operator comprehension and performance [11]. These findings motivate careful design of AR feedback (e.g., target hints, obstacle indicators, autonomy state) alongside the gesture vocabulary.

In the context of modern VR/MR headsets, OpenTeach demonstrates that low-cost consumer HMDs can support high-rate manipulation teleoperation with natural hand motion, providing evidence that commodity hand tracking can be viable for practical data collection and control [4]. Our project builds on the same general premise but targets a simplified, gesture-guided interaction flow geared toward robust execution with local motion constraints.

C. Workload and Usability Evaluation

Evaluating teleoperation interfaces typically requires both objective task metrics (completion time, error rate, path efficiency) and subjective user experience measures. NASA-TLX remains a widely used instrument for subjective workload assessment and is frequently adopted in HRI and teleoperation studies due to its multi-dimensional structure [8]. We adopt NASA-TLX to quantify perceived workload and complement it with task-level performance metrics, enabling comparison against alternative interaction schemes in future iterations.

III. Preliminaries

A. Problem Formulation

We consider the problem of teleoperating a robotic manipulator shown in Fig. 1a in a remote environment Fig. 1b using a Virtual Reality (VR) interface Fig. 1c to perform pick-and-place tasks. The system consists of two primary agents: a human operator equipped with a VR headset (MetaQuest 3) and a robotic agent (UR3 arm) located in a physical workspace. The workspace contains a set of objects to be manipulated and target zones where these objects must be placed. The operator's objective is to guide the robot to move objects from their initial configurations to goal configurations while avoiding collisions with static obstacles in the environment.

The quality of the teleoperation is evaluated based on: 1) task completion rate, 2) total time to completion, 3) communication latency, and 4) safety (minimum distance to obstacles). The system operates under a "human-in-the-loop" paradigm where the human provides high-level guidance (start and goal selection via gestures), and the robot executes low-level motion planning and control. Our objective is to design an integrated VR-robot framework that maximizes intuitiveness and task efficiency while ensuring safe robot operation.

B. Modeling

Robotic Agent: We model the UR3 robotic manipulator as a kinematic chain with $n = 6$ degrees of freedom. Let $q \in \mathbb{R}^n$ denote the joint configuration of the robot. The end-effector position $x_{ee} \in \mathbb{R}^3$ is related to the joint configuration by the forward kinematics map $x_{ee} = FK(q)$. The robot is controlled via a motion planner that generates a trajectory $\tau : [0, T] \to \mathbb{R}^n$ connecting a start configuration q_{start} to a goal configuration q_{goal}, subject to joint limits and collision constraints.

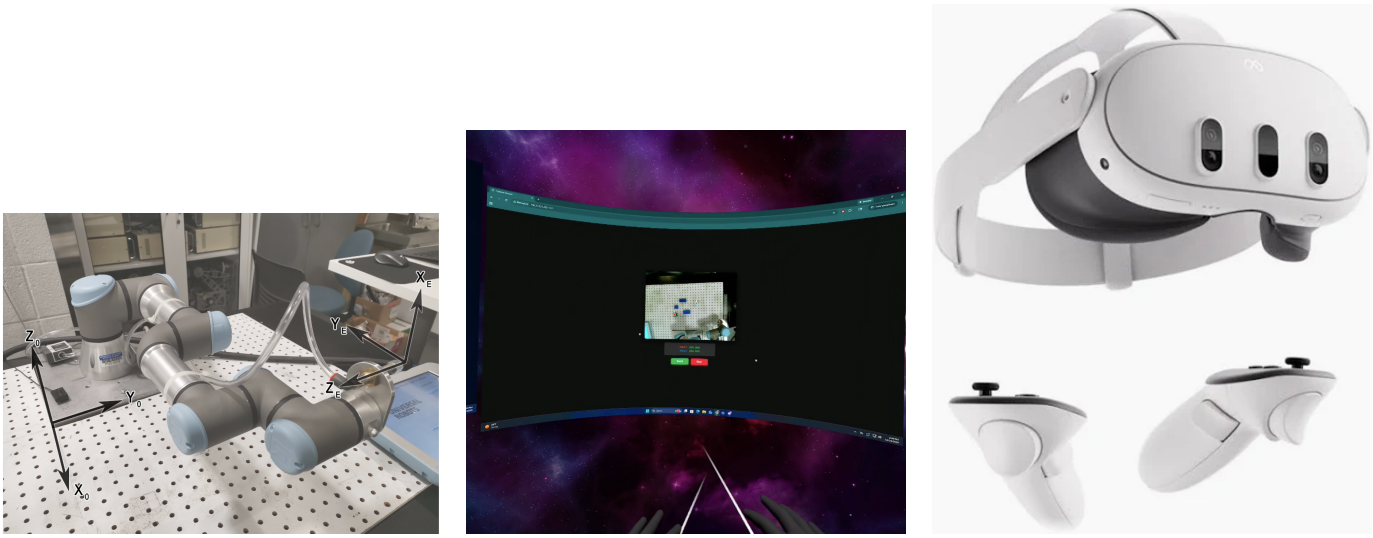
Virtual Environment & Interface: The VR environment $\mathcal{V}$ serves as a digital twin of the physical workspace $\mathcal{W}$. Let ${}^{V}T_{W} \in SE(3)$ denote the transformation from the physical workspace frame to the virtual environment frame. This calibration allows for the bidirectional mapping of coordinates:

$$p_{virtual} = {}^{V}T_{W} \cdot p_{physical} \tag{1}$$

$$p_{physical} = ({}^{V}T_{W})^{-1} \cdot p_{virtual} \tag{2}$$

where $p_{physical} \in \mathcal{W}$ and $p_{virtual} \in \mathcal{V}$. The operator interacts with $\mathcal{V}$ using hand gestures, specifying a goal position $g_{virtual} \in \mathcal{V}$, which is transformed to $g_{physical} \in \mathcal{W}$ for the robot planner.

Communication Constraints: The human operator and the robot are physically separated and communicate via a network bridge (HTTP API). We assume a discrete communication model where information (camera frames I_t, goal commands g_t, and status updates) is exchanged at discrete time steps t_k. The system must be robust to variable network latency,



(a) Robotic manipulator (b) Remote environment (c) VR interface (Meta Quest)

Fig. 1: Teleoperation setup for pick-and-place: a robotic manipulator (left) operates in a remote environment (middle) and is controlled through a VR interface (right).

ensuring that the visual feedback in VR remains consistent with the physical state of the robot.

IV. AR-Based Gesture Guided HRI

We present an integrated teleoperation framework that enables a human operator to control a robotic manipulator using intuitive hand gestures within a Virtual Reality (VR) environment. The system architecture, illustrated in Fig. 2, consists of three primary components: the VR interface (Unity), the robotic control system (ROS), and a middleware communication layer (HTTP API).

A. System Overview

The human operator views the remote workspace through a VR headset (MetaQuest 3), which renders a bird's-eye view of the environment. Hand gestures are captured by the headset's tracking system and used to send commands. These commands are transmitted via a custom HTTP middleware to the robot controller. The robot (UR3) executes the commands by planning collision-free trajectories and providing visual feedback to the operator through a live camera stream overlaid in the VR scene. See Fig. 2

B. Virtual Reality Interface

The VR environment is built in Unity and serves as the primary interface for human-robot interaction. It provides the operator with a digital twin of the physical workspace. To ensure accurate spatial reasoning, the interface overlays a live video feed from the robot's overhead camera onto the virtual ground plane. The operator interacts with this environment using ray-cast pointers attached to their virtual hands, allowing them to select pick-and-place locations directly on the video feed.

C. Perception and Workspace Calibration

To map the virtual interactions to physical robot motions, we employ a homography-based calibration. We define a set of four correspondence points $P_{world} \in \mathbb{R}^2$ (physical workspace corners) and their corresponding pixel coordinates $P_{pixel} \in \mathbb{R}^2$ in the camera frame. A homography matrix $H \in \mathbb{R}^{3\times3}$ is computed to map any pixel (u, v) to a world coordinate (x, y):

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \propto H \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{3}$$

For obstacle detection, the system processes the camera feed using HSV color thresholding to identify non-traversable regions. This binary obstacle map is discretized into a 20×6 occupancy grid, which serves as the cost map for motion planning.

D. Motion Planning and Control

The robot's motion is governed by a hybrid planner that combines high-level discrete search with low-level continuous control. Given a start and goal position from the VR interface, the system first updates the occupancy grid based on the latest perception data. An A* search algorithm then computes a collision-free path consisting of a sequence of waypoints $\{w_1, w_2, \ldots, w_n\}$ on the grid.

These Cartesian waypoints are converted into joint space trajectories using an analytic inverse kinematics solver ($q = \text{IK}(x, y, z, \psi)$). A finite state machine (FSM) orchestrates the execution, transitioning between 'Move', 'Descend', 'Grasp', 'Ascend', and 'Release' states. The controller operates at 20 Hz, continuously monitoring the robot's joint states to ensure precise trajectory tracking.

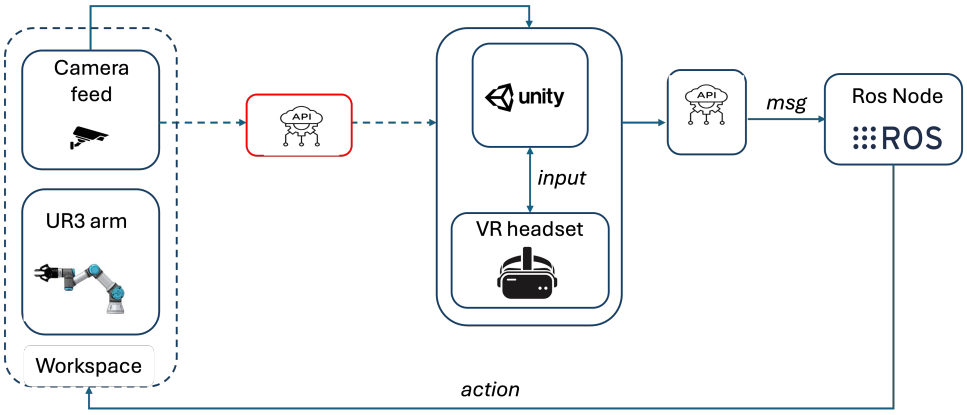


Fig. 2: General overview of the system

E. Communication Protocol

Data exchange between the disparate VR (C#/Unity) and Robot (Python/ROS) systems is managed via a RESTful HTTP API. The robot controller polls the middleware server at 1 Hz (or higher) to retrieve pending commands. The communication payload follows a JSON structure:

```
{ "action": [int], "x1": [int], "y1":
  [int], "x2": [int], "y2": [int] }
```

where `action` distinguishes between task execution commands (`START=1`) and safety overrides (`STOP=0`). This decoupling ensures that the safety-critical robot control loop remains strictly isolated from potential network latencies in the VR rendering pipeline.

V. Results

To validate our framework, we conducted a series of pick-and-place experiments using the UR3 robotic arm controlled via the VR interface. We established three difficulty levels: Easy, Medium, and Hard, varying the number and arrangement of obstacles in the workspace. For each difficulty, we performed multiple trials to evaluate the system's robustness, scalability, and safety. The experimental setup utilized a top-down camera feed segmented into a 20×6 grid for path planning, with obstacle detection performed in real-time using HSV color thresholding. All experiments were run on a standard desktop computer (Intel Xeon-E31245 CPU, 4 cores, 8 GB RAM) running Ubuntu 20.04, communicating with the VR headset over a custom HTTP API.

A. Quantitative Metrics

We evaluated the system performance using four key metrics:

1) **Completion Rate**: The percentage of trials where the robot successfully reached the goal without collision.
2) **Time to Complete**: The total duration from the start of the motion to the robot reaching the target.
3) **Latency**: The time delay between data acquisition and the start of the robot's motion.
4) **Minimum Distance to Obstacle**: The smallest clearance maintained from obstacles during the trajectory, ensuring safety.

Table I summarizes the results for all difficulty levels across 5 runs each. We observed a **100% completion rate** for both Easy and Medium scenarios, validating the robustness of our grid-based planner. In the Hard scenario, the success rate remained high at 80%, with failures primarily due to occlusion in the densest obstacle configurations. The latency was consistent across all trials ($\approx$ 2.28 s), demonstrating that our communication pipeline scales well regardless of scene complexity. Crucially, the minimum distance to obstacles remained safe ($\geq$ 8.00 px) even in the Hard scenario, confirming the planner's ability to generate collision-free paths. We further analyze the system's performance in Fig. 3. The results demonstrate: (a) consistent task completion times across difficulty levels, with slightly higher variance in hard scenarios; (b) a strong linear correlation between path length (number of waypoints) and execution time; and (c) a highly stable communication latency centered around 2.28s, validating the robustness of our HTTP middleware.

B. Trajectory Analysis

Fig. 4 illustrates representative trajectories generated by our path planner across the three difficulty levels. In all cases, the planner successfully identified the feasible workspace (green box) and obstacles (blue boxes), generating a smooth path (yellow line) from the start (green dot) to the goal (red dot). The planner effectively navigated tight corridors in the Hard scenario, further validating the geometric accuracy of our camera-to-world calibration.

C. Qualitative User Analysis

To assess the usability of the VR interface, we asked 10 students from UIUC to control the robot using our system. After a brief training session, participants performed a series of pick-and-place tasks. The feedback was overwhelmingly positive; users reported that the VR interface was "intuitive" and "natural" compared to traditional keyboard-and-mouse

TABLE I: Experimental Results Summary (Mean ± Standard Deviation)

Difficulty	Completion Rate	Time to Complete (s)	Latency (s)	Min Distance (px)
Easy	100.0%	24.18 ± 0.74	2.29 ± 0.01	10.00 ± 0.00
Medium	100.0%	20.49 ± 2.52	2.28 ± 0.01	10.00 ± 0.00
Hard	80.0%	19.86 ± 3.59	2.28 ± 0.01	8.00 ± 4.47

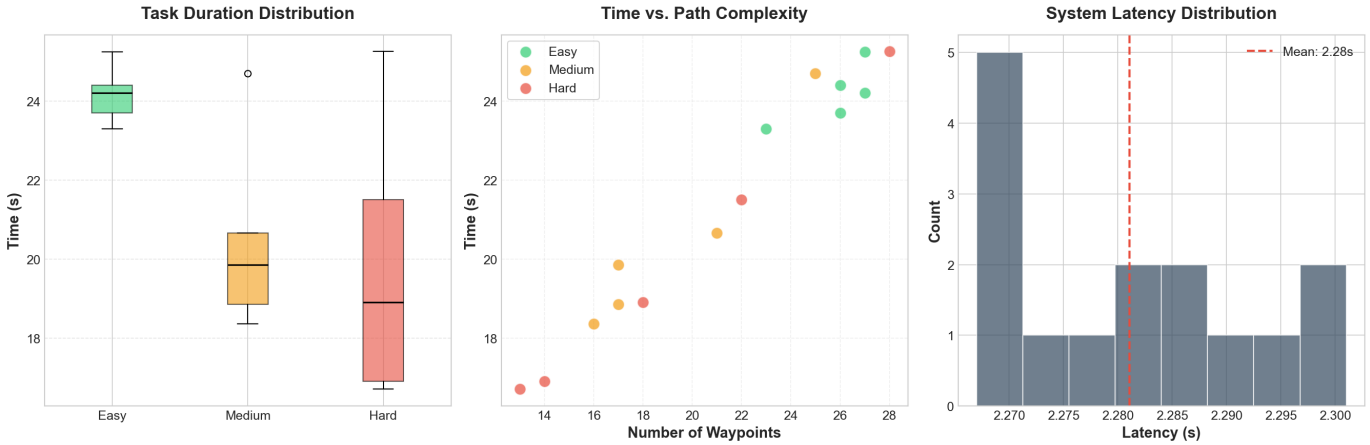


Fig. 3: Additional performance metrics: (left) Task Duration Distribution showing consistency across difficulties. (middle) Time vs. Path Complexity scatter plot confirming linear scaling with goal distance and not difficulty (right) System Latency Histogram demonstrating communication stability.

teleoperation. Specifically, the immersive 3D view and gesture-based controls allowed them to perceive depth more accurately, making it easier to select target locations without trial-and-error.

D. Distance to Collision Analysis

Finally, we analyzed the safety margin along the robot's path. Fig. 5 plots the distance to the nearest obstacle as a function of path progress. As expected, the distance decreases in the middle of the trajectory as the robot navigates through the obstacle field but recovers as it reaches the goal. In all recorded successful trials, the planner maintained a safety margin above the collision zone (10 px), ensuring safe operation even in cluttered environments.

VI. Conclusion and Future Works

We present a teleoperation framework using consumer-grade VR hardware for intuitive robotic manipulator control. An immersive bird's-eye view and gesture-based controls simplify remote pick-and-place tasks. Experiments show robust performance: 100% task completion in simple scenarios and safe, consistent obstacle avoidance in cluttered environments.

Future work will improve system interactivity and adaptability by refining Unity-VR integration for more intuitive gestures and adding haptic feedback for collision and grasp alerts. We will also extend the motion planner for real-time dynamic obstacle handling and conduct a larger user study to quantitatively assess cognitive load and learning curves versus traditional interfaces.

Acknowledgment

We thank Andre Schreiber, the course teaching assistant, for being consistently available to answer our questions.

References

[1] Y.-P. Su, X.-Q. Chen, C. Zhou, L. H. Pearson, C. G. Pretty, and J. G. Chase, "Integrating virtual, mixed, and augmented reality into remote robotic applications: A brief review of extended reality-enhanced robotic systems for intuitive telemanipulation and telemanufacturing tasks in hazardous conditions," *Applied Sciences*, vol. 13, no. 22, p. 12129, 2023. [Online]. Available: https://www.mdpi.com/2076-3417/13/22/12129

[2] R. Suzuki, A. Karim, T. Xia, H. Hedayati, and N. Marquardt, "Augmented reality and robotics: A survey and taxonomy for ar-enhanced human-robot interaction and robotic interfaces," in *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*, 2022, also available as arXiv:2203.03254. [Online]. Available: https://arxiv.org/abs/2203.03254

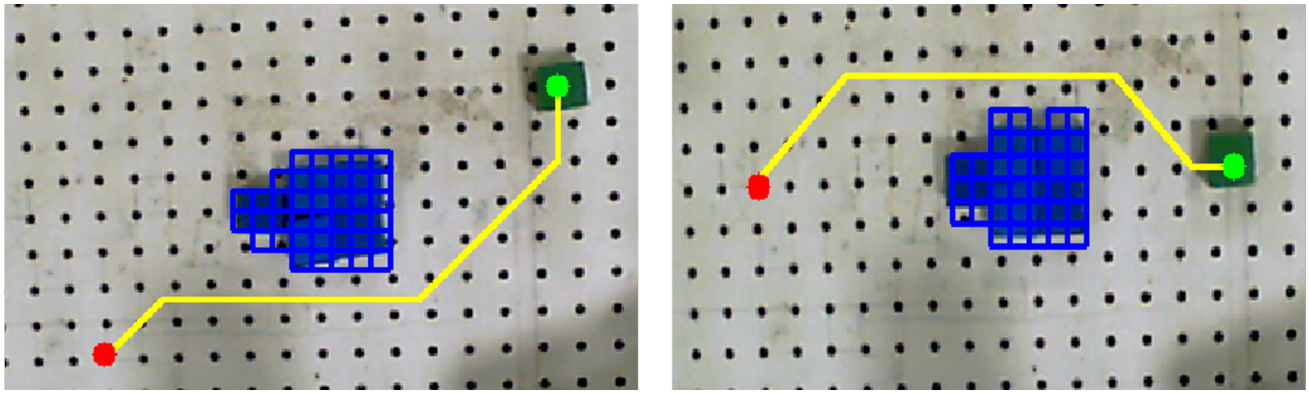
[3] Meta Platforms, Inc., "Meet meta quest 3, our mixed reality headset starting at $499.99," Meta Newsroom, 2023, accessed: 2025-12-19. [Online]. Available: https://about.fb.com/news/2023/09/meet-meta-quest-3-mixed-reality-headset/

[4] A. Iyer, Z. Peng, Y. Dai, I. Guzey, S. Haldar, S. Chintala, and L. Pinto, "OPEN TEACH: A versatile teleoperation system for robotic manipulation," arXiv:2403.07870, 2024. [Online]. Available: https://arxiv.org/abs/2403.07870

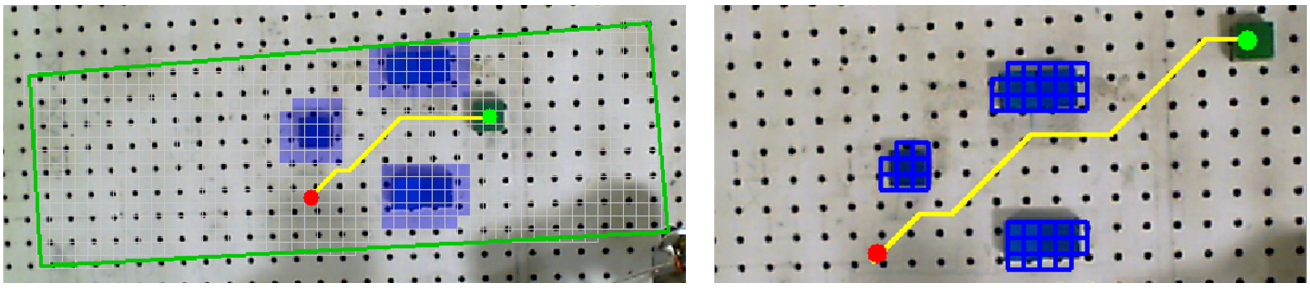
[5] C. P. Quintero, S. Li, M. K. Pan, W. P. Chan, H. F. M. Van Der Loos, and E. Croft, "Robot programming through augmented trajectories in augmented reality," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1838–1844.

[6] C. Nuzzi, S. Ghidini, R. Pagani, S. Pasinetti, G. Ragni, and G. Sansoni, "Hands-free: a robot augmented reality teleoperation system," in *2020 17th International Conference on Ubiquitous Robots (UR)*, 2020, pp. 617–624.

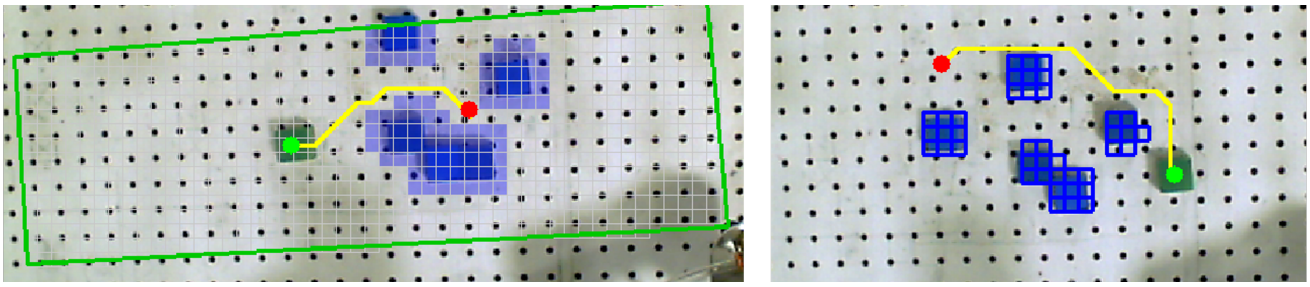
[7] F. Regal, Y. S. Park, J. Nolan, and M. Pryor, "Augmented reality remote operation of dual arm manipulators in hot boxes," arXiv:2303.16055, 2023. [Online]. Available: https://arxiv.org/abs/2303.16055



(a) Easy difficulty



(b) Medium difficulty



(c) Hard difficulty

Fig. 4: Robot path planning across difficulty levels. The red dot is the goal, the green dot is the start point, the yellow line is the path. The green box is the feasible workspace, the blue boxes are identified obstacles.

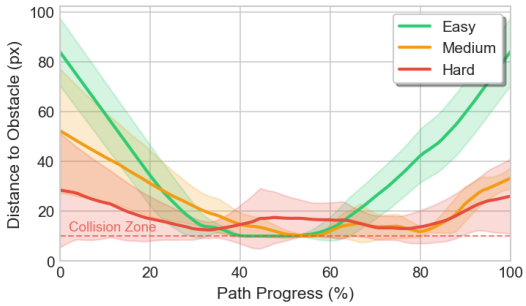


Fig. 5: Distance to Collision across path progress. The safety margin dips in the middle of the trajectory but consistently stays above the collision threshold (dashed red line) for successful runs.

[8] S. G. Hart and L. E. Staveland, "Development of NASA-TLX (task load index): Results of empirical and theoretical research," in *Human Mental Workload*, ser. Advances in Psychology, P. A. Hancock and N. Meshkati, Eds. North-Holland, 1988, vol. 52, pp. 139–183.

[9] X. Wang, S. Guo, Z. Xu, Z. Zhang, Z. Sun, and Y. Xu, "A robotic teleoperation system enhanced by augmented reality for natural human–robot interaction," *Cyborg and Bionic Systems*, vol. 5, p. 0098, 2024.

[10] A. Vysocký, T. Poštulka, J. Chlebek, T. Kot, J. Maslowski, and S. Grushko, "Hand gesture interface for robot path definition in collaborative applications: Implementation and comparative study," *Sensors*, vol. 23, no. 9, p. 4219, 2023. [Online]. Available: https://www.mdpi.com/1424-8220/23/9/4219

[11] A. U. Krishnan, T.-C. Lin, and Z. Li, "Human preferred augmented reality visual cues for remote robot manipulation assistance: From direct to supervisory control," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 7034–7039.