

AI-Informed Interactive Task Guidance in Augmented Reality

Viacheslav Tekae^a and Raffaele de Amicis^b

Oregon State University, Corvallis, U.S.A.

{tekaev, deamicis}@oregonstate.edu

Keywords: Augmented Reality, Artificial Intelligence, Task Guidance System, 3D Positioning, Oculus Quest Pro, Adaptive Feedback, Calibration and Synchronization.

Abstract: This paper presents a proof of concept for an augmented reality (AR) and artificial intelligence (AI)-powered task guidance system, demonstrated through the task of opening a door handle. The system integrates an AR frontend, deployed on an Oculus Quest Pro, with an AI backend that combines computer vision for real-time object detection and tracking, and natural language processing (NLP) for dynamic user interaction. Objects such as door handles are identified using YOLOv8-seg, and their 3D positions are calculated to align with the user's environment, ensuring accurate task guidance. The AI backend supports local and cloud processing, maintaining performance even without internet connectivity. The system provides adaptive feedback, adjusting guidance based on user actions, such as correcting improper rotation of a knob. Real-time communication between components is achieved via WebSocket, minimizing latency. Technical challenges like tracking accuracy, latency, and synchronization are addressed through calibration and stress testing under varying conditions. The study emphasizes the system's adaptability to complex scenarios, offering error-handling mechanisms and smooth interaction through AR overlays. This proof of concept highlights the potential of AR-AI integration for task guidance in diverse applications.

1 INTRODUCTION


Task guidance plays a critical role in a variety of fields, from manufacturing and maintenance to healthcare and education (Mendoza-Ramírez et al., 2023; Lapointe et al., 2020). Effective task guidance systems enable users to perform complex procedures by providing step-by-step instructions, ensuring accuracy, and efficiency, and reducing the likelihood of errors (Simões et al., 2019). Traditionally, such guidance has been delivered through manuals, videos, or expert system supervision (Osti et al., 2021; Tarallo et al., 2018). However, these methods can be limited in real-time adaptability and contextual understanding, especially in dynamic environments (Simões et al., 2021).


Recent augmented reality (AR) advancements have introduced new possibilities for task guidance by overlaying digital information directly onto the physical world (Morales Méndez and del Cerro Velázquez, 2024). AR enhances the user's perception of their environment by providing contextual, visual instructions that are interactive and intuitive. This immersive

approach has the potential to transform how users receive guidance, enabling them to complete tasks with greater spatial awareness and precision (Henderson and Feiner, 2011; Funk et al., 2015).

Incorporating artificial intelligence (AI) into AR systems further enhances these capabilities. AI algorithms can process and interpret the physical environment, recognize objects, detect user actions, and adapt the guidance provided in near real-time (Castelo et al., 2023). This combination of AR and AI creates an intelligent task guidance system that instructs users and responds dynamically to their needs and interactions (Stover and Bowman, 2024).

This paper presents a proof of concept for such an advanced AR system, demonstrating its application in a specific task: guiding a user to open the handle of a door. This scenario provides a controlled environment to showcase how AR and AI can work together to deliver effective real-time task guidance, illustrating the potential of these technologies to improve task performance across diverse use cases.

^a  <https://orcid.org/0009-0007-1692-5840>

^b  <https://orcid.org/0000-0002-6435-4364>

2 BACKGROUND

The concept of employing AR technologies to enhance real-world experiences by integrating virtual content has roots stretching back to the last century. These systems are designed to have an internal model of the physical environment, allowing them to overlay digital information seamlessly onto the user's field of view. By integrating real-time data with a contextual understanding of the surroundings, AR can augment what the user sees, making interactions with both the digital and physical worlds more immersive and intuitive. This makes AR a perfect visualization instrument for a task guidance system. (Van Krevelen, 2007)

Augmented reality has shown potential in reducing cognitive load and errors across various domains (Buchner et al., 2022; Puladi et al., 2022). Studies indicate that AR can decrease the extraneous cognitive load in learning environments (Thees et al., 2020; Herbert et al., 2022) and assembly tasks (Yang et al., 2019). In industrial settings, AR glasses have been found to lower cognitive load for assembly operators (Atici-Ulusu et al., 2021). AR assistance can improve performance by shortening task completion time and reducing mistakes in assembly processes (Yang et al., 2019). In circuit prototyping, AR visual instructions have demonstrated effectiveness in reducing errors and mental workload for novice users (Bellucci et al., 2018). Moreover, AR has proven useful in real-time fault detection and analysis in manufacturing (Becher et al., 2022; Fiorentino et al., 2014). The authors propose a method that integrates spatio-temporal analysis of time series data through a handheld touch device with augmented reality to implement visual analysis on the shop floor, enabling real-time responses to faults. The approach was designed and tested on an active production line. However, some research suggests that AR's impact on cognitive load may vary depending on task complexity and design (Buchner et al., 2022). While AR shows promise in reducing cognitive load, its effects on performance are not always significant (Moncur et al., 2023), highlighting the need for further research and optimized AR design in various applications.

The integration of machine learning into AR systems is enabling more adaptive, personalized, and immersive user experiences that can transform the way individuals interact with digital and physical environments. By enabling more accurate and real-time object recognition and environmental understanding, AI allows AR applications to overlay digital information more precisely onto the physical world. Additionally, AI provides adaptive and personalized user ex-

periences in AR through machine learning algorithms that learn from user interactions, improving usability and enabling natural interfaces like voice and gesture recognition. (Park et al., 2020) showed in their paper how to apply task guidance visual clues to real-world objects in a hand held mobile device with a camera based on a reconstructed 3D model of the target object using deep learning and RGB-D data from the integrated camera. The system detects and segments real-world objects using Mask-RCNN, allowing the extraction of corresponding 3D point cloud data. The virtual model is spatially matched with the real object using the 3D position and pose of the real object. Recent research explores the integration of AI with AR for task guidance in multiple direction. AI enhances AR systems by improving user activity recognition, and adaptive guidance (Ng et al., 2020; Truong-Allié et al., 2021). These AI-AR systems show promise in various applications, including maintenance, assembly, and manufacturing (Lapointe et al., 2020; Chandan K. Sahu and Rai, 2021). Studies demonstrate that AI-enhanced AR guidance can significantly improve task performance and learning outcomes (Westerfield et al., 2013). Researchers have developed frameworks and systems to automate workflow modeling, task monitoring, and guidance generation (Han et al., 2017; Konin et al., 2022). Visualization tools have been created to support the development and analysis of AI-AR assistants (Castelo et al., 2023). While challenges remain, the integration of AI with AR shows potential for revolutionizing task guidance across various industries by providing more efficient, adaptive, and context-aware support to users.

One of the key challenges in AR task guidance systems is ensuring that the system can quickly and accurately detect and understand real-world objects in a scene, enabling a seamless interaction between the physical and digital worlds. For example, object detection and instance segmentation algorithms such as YOLOv8, MaskFormer, and Mask R-CNN are commonly used to achieve real-time object detection and segmentation. Each of these methods has its own strengths and weaknesses depending on the use case. However, YOLOv8 is the fastest and most efficient of the three, making it the best option for applications that require real-time object detection with minimal latency (Jocher et al., 2023). Mask R-CNN is slower due to its two-step process, but it provides excellent accuracy (He et al., 2018). Another model, MaskFormer, is even more accurate, but because of its precision and large and complex internal structure, the inference speed is very low (Cheng et al., 2021).

Another important aspect in AR task guidance systems is the capability to interpret user's commands

and requests. Such language assistants are usually built using natural language processing (NLP) techniques. Large language models (LLMs) like LLaMA 3.1 (Dubey, 2024) and Mistral 7B (Jiang et al., 2023) represent the state-of-the-art of modern NLP, each offering distinct advantages depending on the application. LLaMA 3.1 excels in complex tasks requiring deeper contextual understanding. On the other hand, Mistral is faster due to its smaller size.

Previous research often relied on mobile phones to display AR content directly on the screen, which can be inconvenient for users, as it requires them to hold the device, leaving their hands unavailable for task execution. Alternatively, custom and expensive AR devices were used, limiting accessibility. In this paper, we focus on a mass-market AR head-mounted display, aiming to overcome its limitations and constraints to develop a proof of concept for a task guidance system.

3 SYSTEM ARCHITECTURE

This section outlines the architecture of the task guidance system, describing how the various components interact to deliver real-time guidance using augmented reality and artificial intelligence. The system is designed to be modular, consisting of an AR frontend and an AI backend, as depicted in Figure 1, which communicate over a network to ensure seamless guidance for users. Each component plays a distinct role, ensuring that the system remains responsive, scalable, and adaptable to different hardware setups.



Figure 1: Overall system architecture.

The system architecture is built around two primary components:

- **AI Backend:** It serves as the system’s processing hub, combining advanced algorithms for real-time object detection, tracking, and natural language interaction. It is responsible for handling computationally intensive tasks, thus offloading the workload from the AR frontend.
- **AR Frontend:** It is implemented for an Oculus Quest Pro HMD and acts as the user interface and controller for the system. It coordinates all task-related processes, including managing the user in-

terface, visualizing AR guidance overlays, and tracking user interactions through controllers or hand tracking. Specifically, the AR frontend:

- Monitors the user’s hand movements, controller positions, and headset orientation to ensure precise interaction with virtual objects.
- Provides real-time visual hints, such as arrows or highlights, guiding users through each step of the task.
- Receives data from the AI backend, including detected object positions, their class, and their probability.

These two components communicate over a wireless network, exchanging data using HTTP or WebSocket protocols. This setup ensures low-latency communication, enabling the AR system to remain responsive as users interact with the environment and complete tasks in real-time.

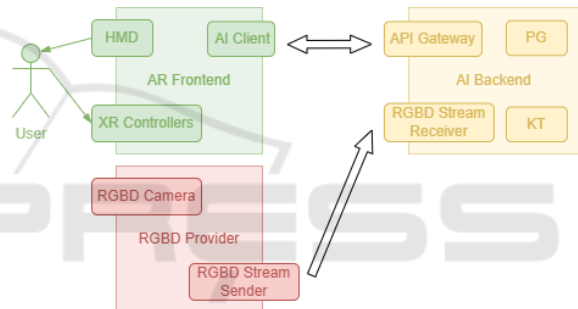


Figure 2: Alternative system architecture.

3.1 Alternative Architecture for Mobile Scenarios

In addition to the primary architecture, the system supports an alternative configuration, depicted in Figure 2, designed specifically for mobile and lightweight scenarios. While the primary architecture offers direct and efficient communication between the AR frontend (HMD) and the AI backend, it assumes a stationary setup, where the external RGBD camera and backend server are colocated, often in a lab or controlled environment. However, this design becomes impractical in scenarios that demand field operations, mobility, or rapid deployment.

The primary motivation for this alternative architecture arises from a critical limitation of the Meta Quest HMD: it does not provide access to raw video streams from its internal cameras. While the Quest is a mobile device capable of processing AR content, this limitation restricts its ability to use advanced computer vision techniques that require direct control

over the RGB and depth data. To overcome this constraint, we introduce an external RGBD camera that is carefully calibrated with the HMD's coordinate system, enabling precise object detection and alignment.

However, connecting an external camera directly to the Quest HMD is not supported. The next logical step - connecting the camera directly to the backend server - poses challenges when the backend is deployed on a stationary PC, as the system cannot be easily transported to environments where mobility is essential (e.g. remote sites, field operations, or outdoor maintenance tasks).

To solve this problem, we propose a decoupled mobile architecture that incorporates a portable intermediary processing device, such as Nvidia Jetson or Raspberry Pi, to bridge the gap between the external camera and the AI backend. The portable device:

1. Reads and processes the video stream from the lightweight external RGBD camera.
2. Transmits the video stream over the network to the AI backend for further analysis and processing.

This mobile configuration offers several key benefits. First, the user can move freely, as the camera and processing device are compact and lightweight. This makes the system ideal for use in environments where mobility is essential, such as field operations or remote maintenance tasks. Secondary, in this configuration, the camera captures RGB and depth data, which is transmitted to the AI backend over the network for processing. The use of ZeroMQ ensures minimal communication latency between the RGBD provider and the AI backend, enabling real-time object detection and segmentation.

The mobile setup is also compatible with HMDs that feature internal RGBD cameras, allowing the system to access the camera stream directly from the headset. However, using an external RGBD provider provides flexibility in scenarios where the headset's built-in sensors may not be sufficient.

A potential drawback of this decoupled architecture is the introduction of increased latency due to the added network communication step. While the primary architecture allows for more direct data flow, the mobile architecture must transmit data between multiple devices, slightly impacting system responsiveness. Nevertheless, the trade-off in latency is offset by the portability and flexibility gained, making the alternative configuration suitable for tasks that require mobility or quick deployment.

3.2 AR Frontend

The AR Frontend, displayed in Figure 3, mainly serves as a controller for the task guidance system

processing input data from different sources and advancing the internal state machine accordingly.

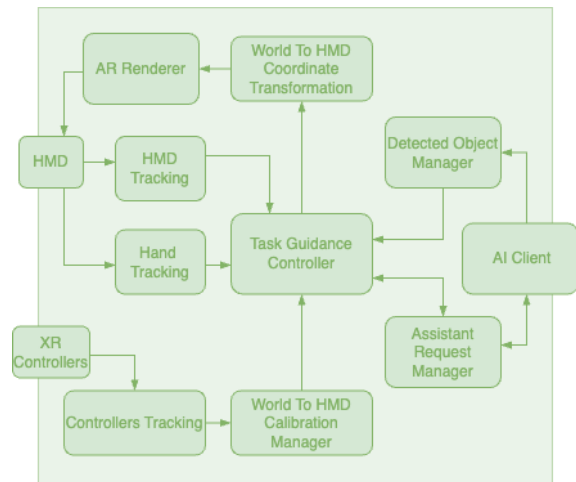


Figure 3: Implemented Architecture of the AR Frontend.

First, in the idle state it waits for the command from the user to start the guidance process. When the start command is received it gathers all tracked objects with corresponding classes, world poses, and estimated bounding boxes from the AI backend. Based on each object's position it chooses the closest pair of door-handle and starts guiding the user via AR elements indicating current recommended action. Examples of provided AR task guidance hints are demonstrated in Figure 4.

3.3 AI Backend

The AI backend serves as the computational engine of the task guidance system, managing essential processes such as object detection, tracking, and natural language processing. It operates with a modular design, enabling tasks to be executed locally on the hardware or, when needed, offloaded to cloud-based services. This flexibility ensures scalability and adaptability to various use cases and deployment environments.

The entry point to the AI backend is an API Gateway, which functions as a centralized interface for routing external requests to the appropriate internal components. This approach allows processing units to be modified or replaced at runtime without requiring reconfiguration of the user's device. For example, if computationally intensive tasks demand more resources, they can be offloaded to cloud services, while simpler requests are handled locally. The default operation is entirely local, ensuring that the system can function even without internet connectivity, making it suitable for scenarios where network access is limited



Figure 4: Screenshots from the AR headset demonstrating task guidance hints.

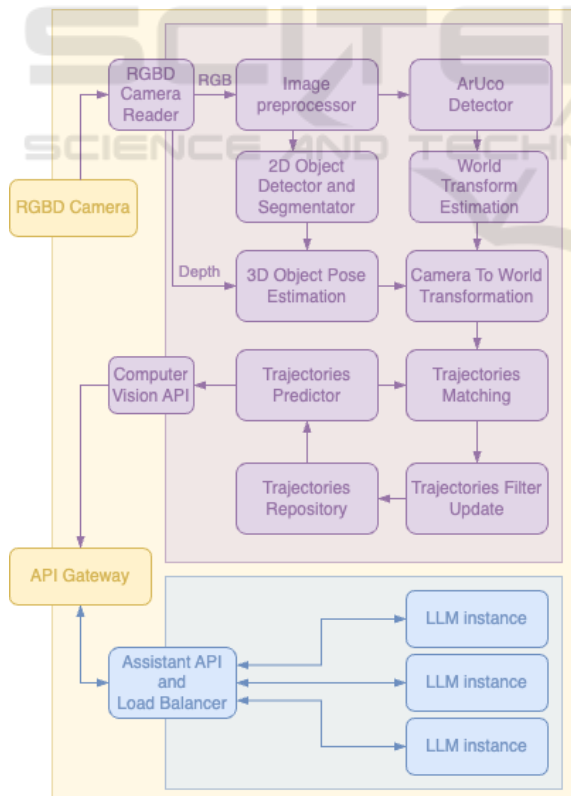


Figure 5: Implemented Architecture of the AI Backend.

or unreliable.

The AI backend is structured around two primary functional components. The first, known as the perceptual grounding component, focuses on understanding the user’s environment. It’s main subcomponent is Computer Vision subcomponent. It processes the video feed, identifying and tracking objects such as doors and handles. Additionally, it is responsible for calibrating the external camera to world coordinate system.

The second component of the AI backend is the knowledge transfer module, which acts as the intelligent assistant for the system. The major part of this component is Language Assistant subcomponent. This module leverages natural language processing algorithms to interpret user commands and convert them into actionable steps. When the user interacts with the system through voice commands the knowledge transfer component analyzes the input and identifies the intended action. For example, when the user says “Open the door”, the system understands the command and initiates the appropriate guidance sequence.

In summary, the AI backend is a sophisticated and modular component that ensures the system can accurately perceive its environment, process user commands, and provide adaptive task guidance. Its dual subcomponents — perceptual grounding for object

detection and tracking, and knowledge transfer for NLP-based interaction — work in tandem to deliver precise and responsive support to users. With the flexibility to operate locally or integrate cloud services, the AI backend ensures scalability and robustness, making it a versatile solution for a wide range of task guidance scenarios.

3.3.1 Computer Vision Subcomponent

The computer vision subcomponent is responsible for capturing, processing, and interpreting visual information from the physical environment to enable precise object detection and tracking. This component relies on an external camera equipped with both a depth sensor and a color sensor.

Before deploying the system, both the depth and color sensors undergo calibration to estimate their intrinsic and extrinsic parameters. Each sensor's intrinsic parameters, which describe the focal length, optical center, and lens distortion, are captured in two separate intrinsic matrices. Additionally, two arrays store the distortion coefficients for the sensors, accounting for lens-related aberrations. An extrinsic matrix is also computed, representing the transformation required to align objects detected in the color image with the corresponding depth image. For simplicity and reliability, the calibration parameters used in this system are derived directly from those provided by the camera manufacturer.

The object detection and tracking pipeline in the perceptual grounding component is multistaged, ensuring accurate interpretation of the scene. The first step involves 2D object detection on the color image, where the system runs the YOLOv8-seg model trained on custom manually labeled dataset of 3000 images to generate masks and bounding boxes for detected objects. This algorithm identifies and segments objects of interest, such as door handles, with high accuracy, allowing the system to focus on relevant elements within the scene.

Once 2D detection is completed, the system aligns the depth image with the color image using the previously calibrated intrinsic and extrinsic parameters. This alignment ensures that the depth information corresponds precisely with the visual data, providing a coherent spatial representation of the scene. After aligning the two images, the system estimates the median depth of each detected object by applying the segmentation mask to the depth image. This median depth value plays a critical role in the next stage of the pipeline, where the 3D position of the detected object is computed.

To calculate the 3D position, the system uses the intrinsic matrix of the color sensor to map the cen-

ter of the object's bounding box into camera coordinates. This transformation allows the backend to understand where the object is located relative to the camera, providing the foundation for accurate object tracking. In parallel with object detection, a calibration process operates continuously to maintain synchronization between the camera frame and the world frame. This synchronization is essential, as both the camera and the AR frontend rely on the same spatial references to ensure that virtual guidance aligns seamlessly with the physical environment. A detailed description of the calibration process is provided in Subsection 3.4.

The final step is the transformation of the detected objects' positions from camera coordinates into the world frame. This transformation uses the estimated transform matrix obtained during calibration, ensuring that all detected objects are referenced consistently within the shared spatial framework. To keep track of objects throughout the camera stream frames all new detections are matched with already tracked objects and their state is estimated using Unscented Kalman filter (Wan and Van Der Merwe, 2000) and constant velocity motion model. The tracked objects are then stored and made accessible to the AR Frontend via a WebSocket connection, allowing for real-time updates to be transmitted with minimal latency. To compensate for the processing time of the detection and tracking pipeline and possible latencies in image frame acquisition we extrapolate each detected object's state into the future using an estimated motion model state. This continuous flow of data ensures that the task guidance system remains responsive and adaptive, providing users with precise instructions that align accurately with their environment.

In summary, the perceptual grounding component plays a crucial role in the task guidance system by combining advanced computer vision techniques with precise calibration and spatial mapping. Through its multistage detection pipeline, the system captures, aligns, and interprets visual information in real time, enabling effective guidance for users. By ensuring synchronization between the camera frame and the world frame, the component provides a stable foundation for the AR frontend to deliver accurate and context-aware instructions throughout the task.

3.3.2 Language Assistant Subcomponent

The knowledge transfer component contains several open-source LLM instances behind a load balancer. We must balance the load because one LLM instance can process only one request at a time. The LLMs models are fine-tuned to instruct the users based on

their inputs. Primarily for this PoC we used Llama 3.1 70B quantized as Q5_K_M in GGML format and llama.cpp as a runtime. The frontend part is capable of translating user speech into text using Voice SDK for Unreal Engine by Meta. Then this text is augmented with auxiliary context about the current guidance step and a list of available functions (start guidance, stop guidance). Based on the context and user's requests the assistant is capable of providing users with the information about the guidance and enabling or disabling the guidance per user's request. The LLM's responses are preprocessed to remove internal tags that llm might use to control guidance and are played back to the user using text-to-speech feature of Unreal Engine.

3.4 Calibration

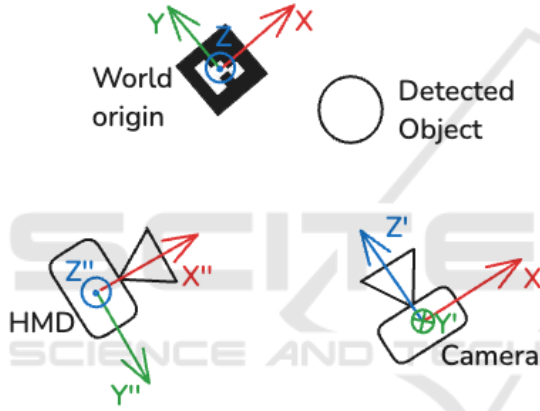


Figure 6: Coordinate systems top-down scheme.

A crucial moment in the integration of AR Frontend and AI Backend into one system is a coordinate calibration between them because both an HMD and the computer vision part work in their local coordinates. To address this inconsistency we propose to track objects in a stationary reference frame (we call it a world frame) relative to some starting point to eliminate synchronization issues caused by the relative movement of an HMD and a camera. Then all we need is to calibrate both an HMD and a camera to the stationary frame using a reference point known to both parts in their local coordinates and in the stationary frame. Our proposed solution is to use a stationary calibration board with an ArUco marker of size 0.16m*0.16m whose 6DOF pose can be estimated by a computer vision component using the ArUco detector (Garrido-Jurado et al., 2014) during the initialization stage. When the marker is detected on the color image the transform A' from the camera coordinate system to the world coordinate system is estimated using Infinitesimal Plane-Based pose estimation (Collins and Bartoli, 2014). The more trickier part is to estimate a transform from the world frame to the HMD frame since the raw RGB stream from the headset's internal camera is not available for reading nor any marker detector is provided by the manufacturer. To overcome this issue we decided to use an XR controller which position is tracked by the headset's software relative to the HMD coordinate system by fusing visual tracking and internal controller's IMU. This means that the position of the marker's corners in the HMD's coordinate system can be estimated via directly touching them by the controller as depicted on figure 7.

coordinate system to the world coordinate system is estimated using Infinitesimal Plane-Based pose estimation (Collins and Bartoli, 2014). The more trickier part is to estimate a transform from the world frame to the HMD frame since the raw RGB stream from the headset's internal camera is not available for reading nor any marker detector is provided by the manufacturer. To overcome this issue we decided to use an XR controller which position is tracked by the headset's software relative to the HMD coordinate system by fusing visual tracking and internal controller's IMU. This means that the position of the marker's corners in the HMD's coordinate system can be estimated via directly touching them by the controller as depicted on figure 7.

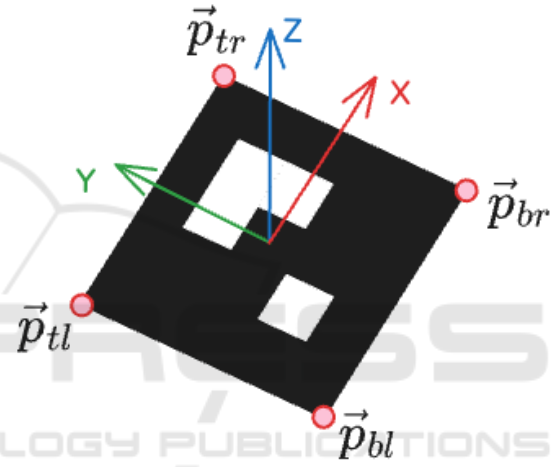


Figure 7: Probed points at each ArUco marker's corner.

In consequence, it is possible to estimate the transform from the world frame to the HMD frame using the following set of equations:

$$\vec{T} = -\frac{1}{4} \sum_{i=1}^4 \vec{p}_i \quad (1)$$

$$\vec{X} = \frac{\vec{p}_{br} - \vec{p}_{bl} + \vec{p}_{tr} - \vec{p}_{tl}}{\|\vec{p}_{br} - \vec{p}_{bl} + \vec{p}_{tr} - \vec{p}_{tl}\|} \quad (2)$$

$$\hat{Y} = \frac{\vec{p}_{tr} - \vec{p}_{br} + \vec{p}_{tl} - \vec{p}_{bl}}{\|\vec{p}_{tr} - \vec{p}_{br} + \vec{p}_{tl} - \vec{p}_{bl}\|} \quad (3)$$

$$\vec{Z} = \frac{\vec{X} \times \hat{Y}}{\|\vec{X} \times \hat{Y}\|}; \quad \vec{Y} = \vec{Z} \times \vec{X} \quad (4)$$

$$R = (\vec{X} \quad \vec{Y} \quad \vec{Z}) \quad (5)$$

$$A'' = \begin{pmatrix} R^T & \vec{T} \\ 0 & 1 \end{pmatrix} \quad (6)$$

Using the described approach we can organize all communication between the components using poses

in the world coordinate system. So that the detected object poses which are defined as 4x4 matrices are transformed from the camera frame to the world frame:

$$P = A'P' \quad (7)$$

where P is an object's pose in the world frame and P' is an object's pose in the camera frame. Then they transferred over the network to an AR headset and then in the AR Frontend transformed to HMD frame:

$$P'' = A''P \quad (8)$$

where P is an object's pose in the world frame and P'' is an object's pose in the HMD frame.

4 IMPLEMENTATION OF THE PROOF OF CONCEPT

The task guidance system developed in this study is demonstrated through the simple, yet illustrative task of opening a door handle. Although the action of opening a door may appear trivial, the diversity in handle types and door mechanisms across environments makes it an ideal scenario to showcase the flexibility and adaptability of the system. This section provides a detailed description of how the proof of concept (PoC) was implemented, highlighting the task flow, system interactions, and key technical considerations.

The selected task involves guiding the user through the steps required to operate different types of handles, such as static handles, knobs, or levers. This variety introduces different interaction patterns, requiring the system to detect, classify, and adapt to the type of handle present. Moreover, the task involves not only recognizing the appropriate handle but also determining additional factors such as the rotation direction for knobs and levers, ensuring the user receives correct and precise guidance.

4.1 Task Flow

The guidance sequence follows a well-defined task flow, designed to ensure that the user completes the door operation smoothly and efficiently. Upon initiating the guidance session, the system detects all visible doors and handles within the scene using the computer vision subcomponent in the AI backend. A spatial mapping process matches the detected handles with the corresponding doors to identify a suitable target for the user. Typically, the system selects the closest door-handle pair as the target to minimize user effort. If the handle type requires rota-

tional movement, the system estimates the rotation direction (clockwise or counterclockwise) based on the handle's orientation and provides corresponding guidance. The system combines informed guessing and real-time monitoring to adapt its guidance dynamically. At the start of the task, the user is guided to grasp the target handle. After the handle is grasped, the system instructs the user to turn or pull the handle in the required direction. The system makes an informed guess about whether the user should push or pull the door or twist the handle clockwise or counterclockwise. This guess serves as the starting point for the guidance process. Once the system provides an initial recommendation, it continuously monitors for movement in both the handle and the door. The system leverages the presence or absence of movement as a key indicator to determine whether the user is encountering difficulties. If no movement is detected after suggesting a push or pull action within a predefined duration, the system infers that the user may be struggling and dynamically adjusts its guidance to suggest the opposite action. For rotational actions, the system tracks the user's hand orientation and movement relative to the handle. If no rotational motion or progress is observed — such as when the handle remains stationary — the system identifies this as an incorrect turning attempt. It then adapts its guidance to recommend the opposite direction (e.g., counterclockwise instead of clockwise). This dynamic, adaptive feedback mechanism ensures that the system can provide corrective suggestions in real time, reducing the likelihood of errors and enabling successful task completion. While the approach relies on probabilistic reasoning rather than absolute certainty, it provides a practical solution to handle ambiguous scenarios in a responsive manner.

4.2 System Workflow and Technical Considerations

The task of opening a door can be represented as a state machine where each state corresponds to a specific step in the interaction, and user actions or system events trigger transitions between states. The state diagram is available in supplemental materials. This modular representation allows the task guidance system to handle complex interactions efficiently by adapting to unexpected situations. For example, the system may detect that the user has missed a step (e.g., attempting to turn the handle without a proper grip) and provide updated instructions to correct the course of action.

In addition to managing user interaction, the system must address technical challenges such as latency

and synchronization. Real-time communication between the AI backend and the AR frontend is critical to ensure that guidance remains responsive and aligned with the user’s movements. The WebSocket protocol, which supports low-latency data transmission, plays a vital role in maintaining this synchronization.

Another key consideration is the classification and tracking of different handle types. The computer vision component must correctly identify the handle type from the video feed and determine the appropriate interaction pattern. For static handles, the system focuses on grip detection and pull or push guidance. For knobs and levers, it analyzes the handle’s orientation and estimates the required rotation direction. This classification enables the system to provide task-specific guidance flows, ensuring the user receives precise instructions tailored to the situation.

The PoC also incorporates error handling mechanisms to account for potential misinterpretations or deviations during task execution. For example, if the system incorrectly estimates the rotation direction of a knob, it immediately adjusts the guidance and suggests the opposite direction. This adaptability ensures that the task remains on track, even if initial instructions are not perfectly followed.

4.3 Interaction with the AR Frontend

The AR frontend plays a critical role in delivering the task guidance experience by providing immersive and intuitive visual feedback to the user. Through the HMD, the user sees virtual overlays aligned with physical objects, helping them understand each step in the process. The AR frontend receives continuous updates from the AI backend, ensuring that guidance remains synchronized with the user’s actions.

The interaction between the user and the system is captured in real time using hand tracking. When the user’s hand touches the handle, the AR frontend triggers a visual confirmation, such as a highlighted grip or an arrow indicating the next action. As the user proceeds through the task, the AR frontend updates the visual hints dynamically, reflecting any changes in the task graph or corrections provided by the guidance system.

5 EVALUATION

To evaluate the proof of concept of the task guidance system we measured several important metrics in different working conditions and various configurations.

5.1 Direct Architecture

First, we tested the architecture described in Section 3. For this architecture the external camera RealSense D455 is connected directly to the AI Backend and the AR Frontend is connected to the AI Backend using 5G WiFi local network. The AI Backend is connected to the router using 1Gbit Ethernet cable. The hardware for the backend is an Alienware laptop with Intel i9 CPU, Nvidia RTX4090 GPU and 64 GB RAM. The AR frontend is a native android application compiled using Unreal Engine 5.4 that is run on Oculus Quest Pro.

To start with, we evaluated the performance of the AI Backend to make sure that it meets real-time performance requirements (>20 Frames Per Second (FPS)). Each measurement is aggregated over 1-second window. The results are in Table 1.

Table 1: AI Backend’s FPS in Direct Mode.

# Measure	Min	Max	Avg
1	15.55	41.61	37.75
2	22.37	36.30	36.35
3	29.96	39.81	36.86
4	27.58	42.72	38.09
5	28.53	42.16	38.63
6	24.44	36.62	34.14
7	15.38	42.97	36.76
8	26.38	35.70	32.24
9	13.83	45.74	41.25
10	17.27	44.25	39.76

One of the most important metrics for a task guidance system in AR setting is accuracy of the estimated 3D position of detected objects. To evaluate it we aligned one edge of the world frame ArUco marker with the door’s edge as shown on Figure 8 and benefiting from the fact that the door is a flat surface we manually measured the x, y and z distances from the marker to detected objects using ruler. For each camera placement we calculated root mean squared errors (RMSE) along each world axis in meters were using

$$\text{RMSE}(y, \hat{y}) = \sqrt{\frac{\sum_{i=0}^{N-1} (y_i - \hat{y}_i)^2}{N}} \quad (9)$$

where y_i is a ground truth value, \hat{y}_i - estimated value, N - number of measurements.

The results for different camera placements shown in Figure 9 relative to the world frame marker are gathered in table 2.

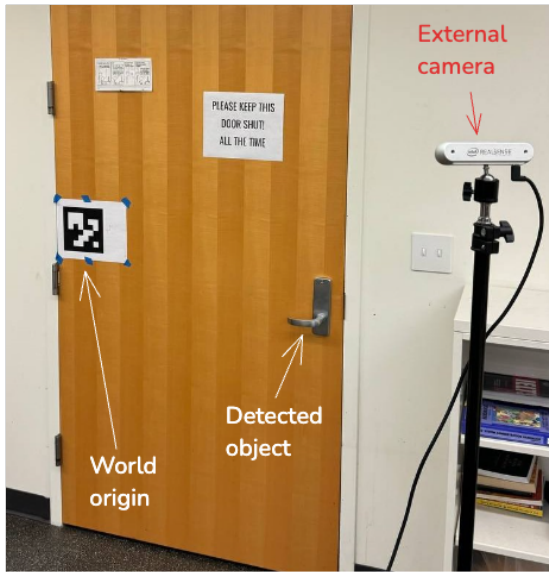


Figure 8: Setup for measuring the accuracy of the estimated 3D positions of detected objects.

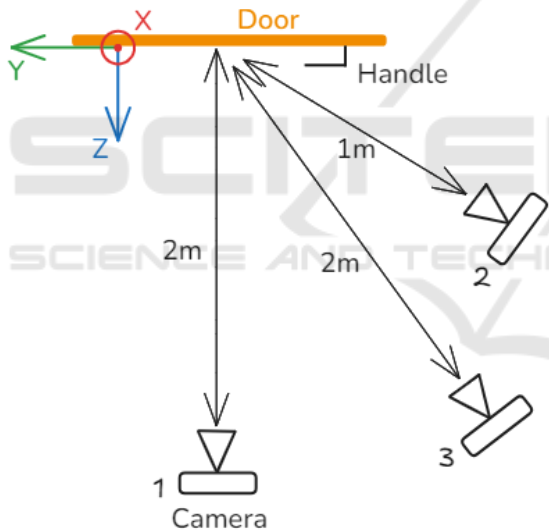


Figure 9: Top-down scheme of camera placements for measuring 3D position error.

Table 2: RMSE of detected handle’s 3D position in meters.

Camera Placement	X	Y	Z
1	0.020	0.010	0.012
2	0.029	0.003	0.016
3	0.013	0.052	0.057

5.2 Mobile Architecture

Second, we tested the mobile architecture described in Section 3.1. In this setup the hardware is the same as in Section 5.1 except that the camera is connected

to the Nvidia Jetson Orin and the images from it are sent over 5G WiFi to the AI backend for processing.

Again, we evaluated the performance of the AI Backend first

Table 3: AI Backend’s FPS in Mobile mode.

# Measure	Min	Max	Avg
1	22.38	39.22	30.20
2	30.45	41.64	37.25
3	27.56	36.73	32.44
4	22.38	39.22	30.20
5	25.88	33.51	30.28
6	28.69	42.79	36.99
7	29.03	41.66	34.35
8	24.25	50.12	40.01
9	29.03	41.66	34.35
10	22.65	38.85	28.93

Another important metric is the latency increase which is caused by having one additional device in the architecture that communicates over the network. To measure latency correctly we used PTP timestamps (IEEE 1588) to synchronize time between the Nvidia Jetson and the AI Backend. Each data package with data from the camera was timestamped and time difference was calculated in the backend comparing current synchronized time and the timestamp from the package. The measurements are aggregated over 1-second windows. The results in milliseconds are present in Table 4.

Table 4: Latency added by external camera reader device.

# Measure	Min	Max	Avg
1	69	80	74
2	70	91	77
3	68	87	77
4	73	110	79
5	71	92	73
6	73	89	81
7	65	81	75
8	68	91	76
9	72	96	82
10	76	103	81

6 CONCLUSIONS

This paper presented and evaluated a proof of concept for an AI-informed augmented reality task guidance system operational on a mass-market AR headset Oculus Quest Pro. Demonstrated through the use case of opening door handles, the system successfully provided real-time AR instructions based on the

perceived environment and user actions. The system architecture, comprising two major components—the AR Frontend and the AI Backend—was detailed for both stationary and mobile scenarios. A crucial synchronization mechanism between these components was proposed and tested.

The results underscore the potential of integrating AI and AR technologies to enhance task guidance systems, offering adaptive feedback and error-handling mechanisms in real-world applications. Technical challenges such as tracking accuracy, latency, and synchronization were addressed through calibration and stress testing under varying conditions, demonstrating the system’s robustness and adaptability.

Future work will focus on transitioning from a manually created state machine to an automated method for generating the task guidance graph based on natural language task descriptions. This generalization will necessitate the development of a new visualization framework capable of rendering AR elements aligned with real-world objects based on the generated graph. Moreover, we aim to explore advanced semantic scene understanding using AI-driven assistants. Specifically, instead of relying solely on fine-tuned YOLO models trained on predefined datasets, we plan to leverage visual large language models (vLLMs) that combine visual perception with contextual understanding. By advancing these areas, the integration of AR and AI technologies holds significant promise for improving task guidance across diverse applications, enhancing efficiency and reducing errors in user interactions.

ACKNOWLEDGEMENT

This publication was prepared by Oregon State University using Federal funds under award #07-79-07914 from the Economic Development Administration, U.S. Department of Commerce. The statements, findings, conclusions, and recommendations are those of the authors and do not necessarily reflect the views of the Economic Development Administration or the U.S. Department of Commerce.

SUPPLEMENTAL MATERIALS

Supplemental materials are available at <https://bit.ly/3P4CINS>

REFERENCES

- Atici-Ulusu, H., Özdemir, Y., Taskapilioglu, O., and Gunduz, T. (2021). Effects of augmented reality glasses on the cognitive load of assembly operators in the automotive industry. *International Journal of Computer Integrated Manufacturing*, 34:1–13.
- Becher, M., Herr, D., Muller, C., Kurzhals, K., Reina, G., Wagner, L., Ertl, T., and Weiskopf, D. (2022). Situated Visual Analysis and Live Monitoring for Manufacturing. *IEEE computer graphics and applications*, 42(2):33–44.
- Bellucci, A., Ruiz, A., Díaz, P., and Aedo, I. (2018). Investigating augmented reality support for novice users in circuit prototyping. In *Proceedings of the 2018 International Conference on Advanced Visual Interfaces, AVI '18*, New York, NY, USA. Association for Computing Machinery.
- Buchner, J., Buntins, K., and Kerres, M. (2022). The impact of augmented reality on cognitive load and performance: A systematic review. *Journal of Computer Assisted Learning*, 38(1):285–303.
- Castelo, S., Rulff, J., McGowan, E., Steers, B., Wu, G., Chen, S., Roman, I., Lopez, R., Brewer, E., Zhao, C., Qian, J., Cho, K., He, H., Sun, Q., Vo, H., Bello, J., Krone, M., and Silva, C. (2023). ARGUS: Visualization of AI-Assisted Task Guidance in AR.
- Chandan K. Sahu, C. Y. and Rai, R. (2021). Artificial intelligence (AI) in augmented reality (AR)-assisted manufacturing applications: a review. *International Journal of Production Research*, 59(16):4903–4959.
- Cheng, B., Schwing, A. G., and Kirillov, A. (2021). Per-Pixel Classification is Not All You Need for Semantic Segmentation. *CoRR*, abs/2107.06278.
- Collins, T. and Bartoli, A. (2014). Infinitesimal Plane-Based Pose Estimation. *Int. J. Comput. Vision*, 109(3):252–286.
- Dubey, A. (2024). The Llama 3 Herd of Models.
- Fiorentino, M., Uva, A. E., Gattullo, M., Debernardis, S., and Monno, G. (2014). Augmented reality on large screen for interactive maintenance instructions. *Computers in Industry*, 65(2):270–278.
- Funk, M., Mayer, S., and Schmidt, A. (2015). Using In-Situ Projection to Support Cognitively Impaired Workers at the Workplace. In *Proceedings of the 17th International ACM SIGACCESS Conference on Computers & Accessibility, ASSETS '15*, page 185–192, New York, NY, USA. Association for Computing Machinery.
- Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F., and Marín-Jiménez, M. (2014). Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280–2292.
- Han, F., Liu, J., Hoff, W., and Zhang, H. (2017). [POSTER] Planning-Based Workflow Modeling for AR-enabled Automated Task Guidance. In *2017 IEEE International Symposium on Mixed and Augmented Reality (ISMAR-Adjunct)*, pages 58–62.
- He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2018). Mask R-CNN.

- Henderson, S. J. and Feiner, S. K. (2011). Exploring the Benefits of Augmented Reality Documentation for Maintenance and Repair. *IEEE Transactions on Visualization and Computer Graphics*, 17:1355–1368.
- Herbert, B., Wigley, G., Ens, B., and Billinghamurst, M. (2022). Cognitive load considerations for Augmented Reality in network security training. *Computers & Graphics*, 102:566–591.
- Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., de las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L. R., Lachaux, M.-A., Stock, P., Scao, T. L., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. (2023). Mistral 7B.
- Jocher, G., Qiu, J., and Chaurasia, A. (2023). Ultralytics YOLO.
- Konin, A., Siddiqui, S., Gilani, H., Mudassir, M., Ahmed, M. H., Shaukat, T., Naufil, M., Ahmed, A., Tran, Q.-H., and Zia, M. Z. (2022). AI-mediated Job Status Tracking in AR as a No-Code service. In *2022 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, pages 1–2.
- Lapointe, J.-F., Molyneaux, H., and Allili, M. S. (2020). A Literature Review of AR-Based Remote Guidance Tasks with User Studies. In Chen, J. Y. C. and Fragomeni, G., editors, *Virtual, Augmented and Mixed Reality. Industrial and Everyday Life Applications*, pages 111–120, Cham. Springer International Publishing.
- Mendoza-Ramírez, C. E., Tudon-Martinez, J. C., Félix-Herrán, L. C., Lozoya-Santos, J. d. J., and Vargas-Martínez, A. (2023). Augmented Reality: Survey. *Applied Sciences*, 13(18).
- Moncur, B., Galvez Trigo, M. J., and Mortara, L. (2023). Augmented Reality to Reduce Cognitive Load in Operational Decision-Making. In Schmorow, D. D. and Fidopiastis, C. M., editors, *Augmented Cognition*, pages 328–346, Cham. Springer Nature Switzerland.
- Morales Méndez, G. and del Cerro Velázquez, F. (2024). Impact of Augmented Reality on Assistance and Training in Industry 4.0: Qualitative Evaluation and Meta-Analysis. *Applied Sciences*, 14(11).
- Ng, L. X., Ng, J., Tang, K. T. W., Li, L., Rice, M., and Wan, M. (2020). Using Visual Intelligence to Automate Maintenance Task Guidance and Monitoring on a Head-mounted Display. In *Proceedings of the 5th International Conference on Robotics and Artificial Intelligence*, ICRAI '19, page 70–75, New York, NY, USA. Association for Computing Machinery.
- Osti, F., Amicis, R., Sanchez, C. A., Tilt, A., Prather, E., and Liverani, A. (2021). A VR training system for learning and skills development for construction workers. *Virtual Reality*, 25:1–16.
- Park, K.-B., Choi, S. H., Kim, M., and Lee, J. Y. (2020). Deep learning-based mobile augmented reality for task assistance using 3D spatial mapping and snapshot-based RGB-D data. *Computers and Industrial Engineering*, 146:106585.
- Puladi, B., Ooms, M., Bellgardt, M., Cesov, M., Lipprandt, M., Raith, S., Peters, F., Möhlhenrich, S. C., Prescher, A., Hölzle, F., Kuhlen, T. W., and Modabber, A. (2022). Augmented Reality-Based Surgery on the Human Cadaver Using a New Generation of Optical Head-Mounted Displays: Development and Feasibility Study. *JMIR Serious Games*, 10(2):e34781.
- Simões, B., Amicis, R., Barandiaran, I., and Posada, J. (2019). Cross reality to enhance worker cognition in industrial assembly operations. *The International Journal of Advanced Manufacturing Technology*, 105.
- Simões, B., Amicis, R., Segura, A., Martín, M., and Ipiña, I. (2021). A cross reality wire assembly training system for workers with disabilities. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, 15:1–12.
- Stover, D. and Bowman, D. (2024). TAGGAR: General-Purpose Task Guidance from Natural Language in Augmented Reality using Vision-Language Models. In *Proceedings of the 2024 ACM Symposium on Spatial User Interaction*, SUI '24, New York, NY, USA. Association for Computing Machinery.
- Tarallo, A., Mozzillo, R., Di Gironimo, G., and Amicis, R. (2018). A cyber-physical system for production monitoring of manual manufacturing processes. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, 12.
- Thees, M., Kapp, S., Strzys, M. P., Beil, F., Lukowicz, P., and Kuhn, J. (2020). Effects of augmented reality on learning and cognitive load in university physics laboratory courses. *Computers in Human Behavior*, 108:106316.
- Truong-Allié, C., Paljic, A., Roux, A., and Herbeth, M. (2021). User Behavior Adaptive AR Guidance for Wayfinding and Tasks Completion. *Multimodal Technologies and Interaction*, 5(11).
- Van Krevelen, R. (2007). Augmented Reality: Technologies, Applications, and Limitations.
- Wan, E. and Van Der Merwe, R. (2000). The unscented Kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, pages 153–158.
- Westerfield, G., Mitrovic, A., and Billinghamurst, M. (2013). Intelligent augmented reality training for assembly tasks. In Lane, H. C., Yacef, K., Mostow, J., and Pavlik, P., editors, *Artificial Intelligence in Education*, pages 542–551, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Yang, Z., Shi, J., Jiang, W., Sui, Y., Wu, Y., Ma, S., Kang, C., and Li, H. (2019). Influences of Augmented Reality Assistance on Performance and Cognitive Loads in Different Stages of Assembly Task. *Frontiers in Psychology*, 10.