ISMAR 2025 Paper Title: An Open Testbed for Mixed Reality Precise Rotation Guidance: Comparative case study of Arrow, Gestalt and Magnifier Cues
Supplemental Materials: Research Guide

1. Implications and Reusability of the Testbed

The MR Open Testbed for Precision Rotation Guidance is an adaptable and scalable platform that enables researchers to conduct precise rotational tasks within MR environments. With its open-source, modular design, the testbed is a versatile tool for a broad range of research areas, including human-computer interaction (HCI), motor skill learning, and perceptual studies.

The testbed's open-source architecture allows for extensive customization. Researchers can adjust key parameters, such as rotation thresholds, cue types (e.g., Arrow, Gestalt, Magnifier) and target generation, making it suitable for a wide variety of experimental setups. This adaptability supports studies ranging from motor skill development to testing visual cues, positioning the testbed as an ideal resource for researchers exploring precision guidance task scenarios. The testbed is designed to be extensible, with the potential to integrate emerging technologies such as adaptive multimodal feedback systems, collaborative MR tasks or customized user interfaces. Future work could explore adaptive systems that respond dynamically to user performance. By integrating machine learning algorithms that adjust cue parameters in real-time based on individual user behavior, the testbed could facilitate research into adaptive systems that improve user performance through personalized guidance.

The comprehensive documentation, including setup guides and hardware specifications, ensures that the testbed can be easily replicated across various research labs. Its low-cost hardware and open-source Unity framework provide broad accessibility, making it a valuable tool for all and fostering global collaboration and knowledge sharing.

2. Open Source Materials

Apart from the research guide, the repository is organized to facilitates access:

- **Unity_Project**: Contains the full Unity testbed, including scene files, visual cue prefabs (Arrow, Gestalt, Magnifier), and parameterized scripts for rotation guidance and automatic performance logging.
- **Arduino_Sketch.ino**: Includes the necessary Arduino code to interface with the rotary encoder, complete with signal processing libraries and detailed calibration instructions.
- **UnityScriptReadEncoder.cs**: A set of Unity scripts that process rotary encoder data, allowing for precise rotation angle calculations and error tracking within the Unity environment.
- **Hardware_Lists.pdf**: A comprehensive hardware list, specifying each component's specifications to ensure ease of replication.
- **3D_Printed_Parts.STL**: STL files for custom 3D-printed components, such as mounts for the encoder and controllers, designed for a straightforward assembly process.
- **Video_Presentation.mp4**: A concise video demonstrating the functionality and setup of the testbed, highlighting key features and guiding users through the platform's capabilities.
- **Raw_Data.csv**: Includes raw data from our study, providing performance metrics (e.g., accuracy, elapsed time) of experimental conditions for comparison and validation.

3. Testbed Usage Instructions

The following procedure outlines the steps necessary to configure the hardware and software for the MR Open Testbed for Precision Rotation Guidance, enabling precise rotational tasks within mixed reality environments.

1. 3D Printing the Mouth Components

Start by 3D printing the mouth component using the STL file provided in the repository (3D_Printed_Parts.STL). Select a compatible 3D printer and appropriate material (e.g., PLA or ABS) to print the component, ensuring the quality and structural integrity of the printed part.

2. Assembling the Encoder to the Mouth

Once the mouth component is printed, assemble the rotary encoder onto it. This process is visually demonstrated in the Video Presentation available in the repository. The video provides detailed instructions to ensure proper encoder placement, alignment, and secure attachment, which are critical for accurate rotational measurements.

3. Cloning the Repository

Clone the repository to your local machine to obtain all the necessary files for the testbed setup. Execute the following command to download the Unity project, Arduino sketches, and documentation.

4. Installing the ESP32 Driver

Install the ESP32 drivers in the Arduino IDE. Open File > Preferences and add the following URL in the "Additional Boards Manager URLs" field: Arduino Copia https://dl.espressif.com/dl/package_esp32_index.json

Next, open Tools > Board > Boards Manager, search for ESP32, and install the package to enable communication between the ESP32 and the IDE.

5. Uploading the Arduino Sketch to the ESP32

After installing the driver, open the Arduino_Sketch.ino file from the repository, connect the ESP32 via USB, and select the correct board and port in the Arduino IDE. Upload the sketch to the ESP32, which will configure the microcontroller to interact with the Unity project. Once the upload is complete, close the Arduino IDE to avoid conflicts when running the Unity project.

6. Opening the Unity Project

Open the Unity project by launching Unity Hub, selecting Open, and navigating to the cloned repository. This will load the MR environment, where the testbed is set up for precision rotation tasks. In the Unity editor, locate and open the "01_scene" from the Assets directory. Press the Play button to begin the testbed simulation, which will execute the rotational tasks using the encoder input. Data can be collected for analysis and further experimentation. The data is comparable with provided our experiment "raw data" .