AR-Collaborative – Augmented Reality Spring 2025 University of Arkansas

https://github.com/SynysterNght/AR-Collaborative

This repository contains the documentation and code for a project that explores the use of collaborative design between mobile phones and head-mounted displays. This project aims to demonstrate and simulate an environment where access to expensive equipment such as a head-mounted display is limited to one device per design group

Project Structure

The codebase is organized within the Assets folder of the Unity project. Key directories include:

- **Scenes:** Contains the project scenes, including Launcher and GeneralRoom.
- **Scripts:** Holds the C# scripts responsible for the project's logic and AR interactions.
- **Packages:** Contains imported packages, including the Meta XR SDK and Photon PUN.

Comprehensive Documentation

This section provides detailed steps to replicate and run this project.

Prerequisites

- **Unity:** 2022.3 LTS
- **Android Development Environment:** Ensure you have the Android SDK and NDK installed and configured within Unity.
- **Meta XR SDK:** Required for Meta Quest device support.
- **XR Plugin Management:** Unity's package for managing XR plugins.
- **Photon PUN (Photon Unity Networking):** For real-time networking features.
- **Android Device:** For testing AR functionality.
- **VR Headset (Optional):** For testing VR functionality.
- **USB Debugging Enabled:** On both your Android phone and VR headset (if applicable).
- **PUN Account:** A Photon Engine account to obtain an App ID.

Steps to Replicate the Project

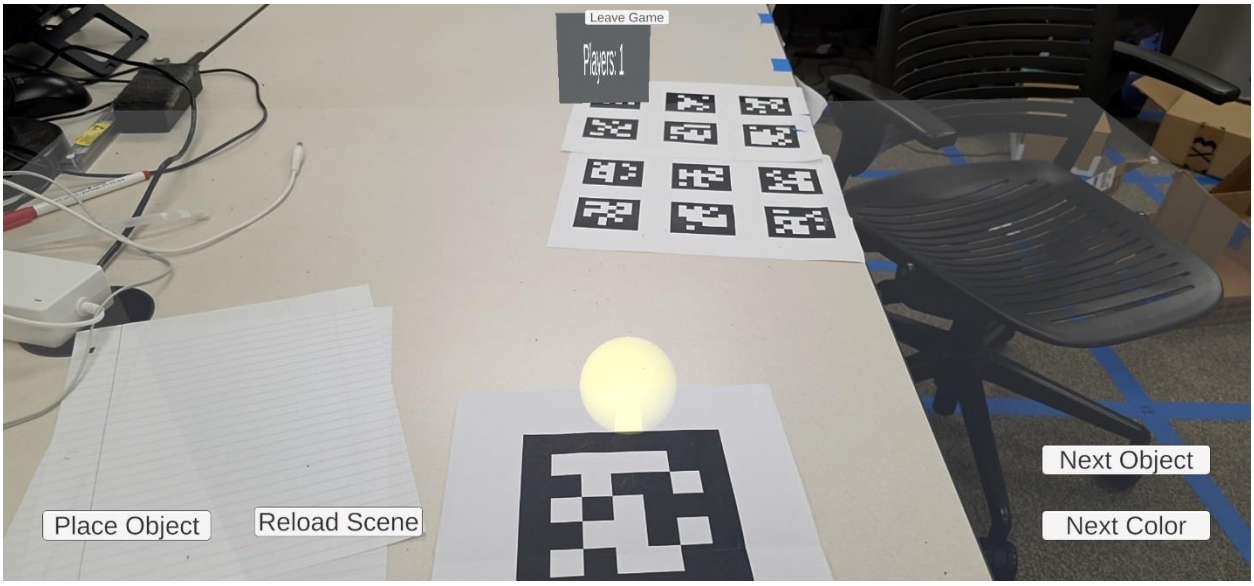
1. **Create a New Unity Project:**
    - Open Unity Hub and create a new project using the **Universal 3D Core** template.
2. **Import Final Package:**
    - Import the Final.unitypackage located in the root of this repository.
3. **Install XR Plugin Management:**
    - Navigate to **Window -> Package Manager**.
    - Search for and install **XR Plugin Management**.
4. **Install Meta XR Core SDK:**
    - Follow the instructions provided by Meta to install the **Meta XR Core SDK** through the Package Manager (usually via "Add package from git URL..." or by adding it through the OpenXR feature sets). Refer to the official Meta developer documentation for the most up-to-date installation method.
5. **Install Meta XR Interaction SDK:**
    - Similarly, install the **Meta XR Interaction SDK** through the Package Manager, following Meta's installation guidelines.
6. **Select "Use Open XR Hand new":**
    - Navigate to **Edit -> Project Settings -> XR Plugin Management**.
    - Under the **General** tab, ensure that **OpenXR** is selected as an active Loader.
    - Within the OpenXR settings (usually accessible by clicking on OpenXR under XR Plugin Management), enable the **"Use Open XR Hand new"** interaction profile.
7. **Restart Unity:**
    - Unity will likely prompt you to restart after making changes to the XR Plugin Management settings. Click **Restart**.
8. **Build Settings - Platform Switching:**
    - Go to **File -> Build Settings**.
    - **Switch Platform to Android:** Select **Android** from the Platform list and click **Switch Platform**.

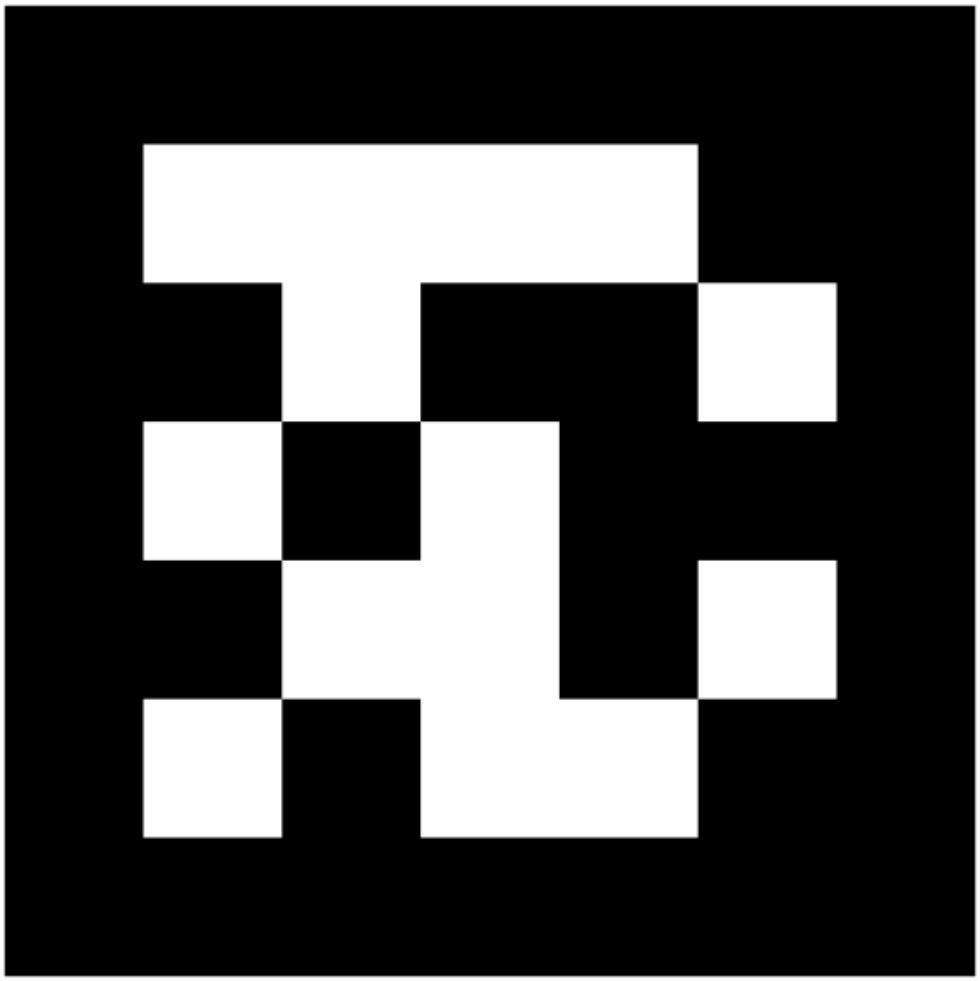
9. **Add Scenes:**
    - In the Build Settings window, under "Scenes In Build," add your project scenes:
        - Drag and drop your **Launcher** scene (should appear as index 0).
        - Drag and drop your **GeneralRoom** scene (should appear as index 1).
10. **Project Validation:**
    - In the Build Settings window, click on the **Project Validation** button (usually at the bottom).
    - Click **Fix All** to resolve any identified issues related to your current build settings.
11. **XR Plugin Management Configuration (Android):**
    - Go to **Edit -> Project Settings -> XR Plugin Management**.
    - Select the **Android** tab.
    - Ensure **ARCore** is checked.
    - Ensure **OpenXR** is checked.
12. **Create a PUN Account:**
    - Go to the Photon Engine website: https://www.photonengine.com/
    - Follow the registration process to create an account and obtain your **App Id PUN**.
13. **Update PUN App ID:**
    - In your Unity project, navigate to **Assets/Photon/PhotonUnityNetworking/Resources/**.
    - Select the file **PhotonServerSettings.asset**.
    - In the Inspector window, locate the **App ID PUN** field and replace the existing value with your newly obtained App ID.

Running on Android Phone (AR)

- **Prerequisites:**
    - Android phone with ARCore support.
    - USB debugging enabled on your Android device.
- **Project Settings (Android for AR):**
    - Go to **Edit -> Project Settings -> XR Plugin Management**.
    - Select the **Android** tab.
    - Ensure **ARCore** is checked.
    - **Uncheck OpenXR**.
    - **Ignore Issues:** You may encounter warnings related to OpenXR. You can typically ignore these for a pure AR build:
        - "OpenXR must be added to the XR Plugin active loaders"
        - "There can only be one camera"
        - "Enabled OpenXR Features require OpenXR to be selected as the active loader of this platform"
- **Player Settings (Android Graphics API):**
    - Go to **Edit -> Project Settings -> Player**.
    - Select the **Android** tab.
    - Navigate to **Other Settings -> Rendering -> Graphics APIs**.
    - Change the order to have **OpenGLES3** listed first, and **Vulkan** listed second. You can achieve this by dragging the 3-lines "=" around in the list.
- **Build and Run:**
    - In the **Build Settings** window, ensure **Android** is selected as the platform.
    - Connect your Android device via USB.
    - Click **Build And Run** to build the application and install it on your connected device.

- **Inside the App**
    - Wait until the session is created by the Headset, then start the application.
    - The phone must first scan the QR code included in this document to orient its position inside the collaborative space.
    - Once inside the room, the phone will have different functionalities:
        - Next Object: Cycle through available shapes. (Depends on how many phone users are present in the room)
        - Next Color: Cycle through Red, Green and Blue colors for the shapes.
        - Place Object: Places the displayed object onto the environment which is the table so the Headset user can grab them.
        - Reload Scene: Clears everything on the screen and have to scan the QR Code again.





Running on VR Headset

- **Prerequisites:**
    - Compatible VR Headset (e.g., Meta Quest).
    - USB debugging enabled on your VR headset.
- **Project Settings (Android for VR):**
    - Go to **Edit -> Project Settings -> XR Plugin Management**.
    - Select the **Android** tab.
    - **Uncheck ARCore**.
    - Ensure **OpenXR** is checked.
- **Player Settings (Android Graphics API):**
    - Go to **Edit -> Project Settings -> Player**.
    - Select the **Android** tab.
    - Navigate to **Other Settings -> Rendering -> Graphics APIs**.
    - Change the order to have **Vulkan** listed first, and **OpenGLES3** listed second.
- **Build and Run:**
    - In the **Build Settings** window, ensure **Android** is selected as the platform.
    - Connect your VR headset via USB.
    - Click **Build And Run** to build the application and install it on your connected device.

- **Inside the App**
    - Start the application to create the session.
    - Once inside the room, the headset will have different functionalities available:
        - ScoreBoard: Displays the timer and current score of the game.
        - Start Game: A poke button that starts the game with a randomized objective and a timer of 2 minutes when pressed.
        - Reload Arena: Clears the table and reset the position.
        - Grab Interaction: The Headset user can grab the transparent table and move it according to its convenience.
        - Legend Board: Shows available and possible objects with possible colors.