SUVSTAR - README

Last updated: June 2019 by Khoi Le and Kyle Qian

Github: https://github.com/dragonkhoi/suvstar/

Description

Stereoscopic Untethered Video See-Through Augmented Reality (SUVSTAR) was developed as the final project for EE267 - Virtual Reality at Stanford University with Dr. Gordon Wetzstein. It uses a separate ARCore-enabled device for each eye to enable standalone outdoor, anchored augmented reality. It was created as a proof-of-concept to showcase use cases for AR HMDs of the future that users wear at all times, in any lighting, and without being tethered to a separate computation hub.

Dependencies

Using third-party libraries and software from:

- ARCore 1.9.0 (https://github.com/google-ar/arcore-android-sdk/releases)
- Unity 2019.1.4f1 (https://unity3d.com/get-unity/download/archive)

Using code based off of:

- EE267 adaptation of Google Cardboard SDK v0.6 (https://drive.google.com/file/d/1JiZHSX6oHcyx4dcg-cLvgg-PgsBha0qi/view)

Runs on:

- Samsung Galaxy S9 phones running Android 9.0 (Pie) (the code is hard-coded to run for this phone's aspect ratios)

To use:

- Open scene: SUVSTAR_MAIN
- Click on Cameras→ PostRenderCamera and check "isLeft" on the SUVSTARPostRender component
- Build to a Samsung Galaxy S9 phone (this will be the left eye)
- Uncheck "isLeft" on the SUVSTARPostRender component
- Build to a 2nd Samsung Galaxy S9 phone (this will be the right eye)
- Slot both phones in the housing head down so the camera are facing outward and are the middle of the headset near the wearer's eyes
- Make sure both phones are exactly centered in the headset, all the way down on the harness, and as close together as possible

- When inside the headset, take a bit of time to verge on an arms-length object in order to adjust
- Use the control panel sticking out at the tops of the phones to adjust render depth and camera feed positioning (make sure the changes are symmetrical across both phones)
- If AR image anchors were uploaded to the build (see SUVSTARImageController.cs), simply bring into view the corresponding real life images in order to track them

Implementation

The basic principles are outlined here:

- Use ARCore to get a lower latency camera feed in the background; and display virtual objects (ARCore Device prefab)
- Send the entire ARCore camera feed (virtual objects and video background) to a Unity RenderTexture (First Person Camera has a Target Texture of size 2960x1440 -- the resolution of the S9)
- Display this RenderTexture on a Quad (ARCoreCameraFeedDisplay has aspect ratio 18.5:9 -- the aspect ratio of the S9)
- Use an orthographic camera shifted back to compensate for spatial offset and depth distortion; this simulates a virtual reprojection by scaling the camera feed down (EyeCamera)
- Send the orthographic camera feed to a RenderTexture
- Use the Google Cardboard algorithm to post-process this RenderTexture into a barrel distorted form and display on a realtime generated mesh (SUVSTARPostRender.cs on PostRenderCamera)
- Shift IPD depending on whether building for left or right phone (see `bool IsLeft` in SUVSTARPostRender.cs)
- Use control panel to help adjust depth for different users, and to center the camera feed for each lens

Scripts:

SUVSTARPostRender.cs

Modeled after the CardboardPostRender.cs from the Unity Stereo SDK provided by the EE267 teaching team, which is based off the old Cardboard SDK. Placed on an orthographic camera to take the RenderTexture of the view from the eye and display the barrel distorted version.

SUVSTARProfile.cs

Modeled after the CardboardProfile.cs from the Unity Stereo SDK provided by the EE267 teaching team. Placed on an object in the scene and shows device parameters necessary for distortion calculations.

SUVSTARImageController.cs
Modeled after the AugmentedImageController.cs in the Google ARCore Augmented Images sample scene. Placed on an object in the scene and manages tracked image life cycles, recognition, and tracking.

RaycastUI.cs
Script attached to the First Person Camera (the main AR camera) to trigger the expansion of the HUD when gazed at.