WebAR : Creating Augmented Reality Experiences on Smart Glasses and Mobile Device Browsers

Pablo Mendigochea
HoloLeo Studios

Figure 1: WebAR example running in mobile browser without use of native app.

CCS CONCEPTS

• **Computing methodologies → Mixed / augmented reality**;

KEYWORDS

Augmented Reality, Webgl, WebAR, Virtual Reality, ARToolkit, Computer Vision, 3D Computer Graphics

ACM Reference format:
Pablo Mendigochea. 2017. WebAR : Creating Augmented Reality Experiences on Smart Glasses and Mobile Device Browsers. In *Proceedings of SIGGRAPH '17 Studio, Los Angeles, CA, USA, July 30 - August 03, 2017,* 2 pages. https://doi.org/10.1145/3084863.3107580

1 INTRODUCTION

Recently Augmented Reality (AR) - overlaying digital content on a real world view - has seen a sharp increase in popularity due to availability of commercial SDKs and capability of display devices such as tablets, phones, and smart glasses. Typically the AR will be developed by a proprietary SDK and requires user to download a native app. Each AR experience requires a separate app.

SIGGRAPH '17 Studio, July 30 - August 03, 2017, Los Angeles, CA, USA

ACM ISBN 978-1-4503-5009-9/17/07.
https://doi.org/10.1145/3084863.3107580

In contrast, WebAR, a brainchild of Jerome Etienne [Etienne 2017], allows AR experiences to run entirely within a browser. Users need only navigate to a URL, point their camera to a fiducial marker, and the experience begins. All tracking and rendering is done in HTML5/Javascript. WebAR is open source, standards based, and runs at acceptable frame rates.

This class will teach participants how to view and create AR experiences using WebAR. Students are encouraged to bring a laptop to author their own AR content. Students can view WebAR with a modern android phone or through Epson MOVERIO smart glasses. At end of class, the student?s WebAR experiences will be shared on social media.

2 METHOD

WebAR works in 3 steps. First a fiducial marker is placed in the real world environment. Second a webpage containing WebAR code is loaded by the user. Lastly, the user points the device camera to the marker.

Detection and tracking begins and the AR experience begins. Under the hood, WebAR works by using web standard technologies and Open Source libraries. Device camera is accessed using HTML5 getUserMedia. Marker detection and tracking is done by JSARtoolkit5. Rendering is powered by WebGL and rides on top of the three.js 3D library. Efficiencies have been achieved with WebRTC and WebWorkers. Finally content creation is simplified using A-Frame.

Designing WebAR experiences has specific considerations that need to be taken into account. Most important is marker detection and tracking. Continuous line of sight and shallow camera angles relative to marker are necessary. The environment should be well lit and marker printed on non glossy paper. Marker should be printed at large size (paper size) so detection is possible from a few feet away. Finally the device camera lens should be clean and have a wide FOV without distortion.

Content that can be augmented can be anything that can be rendered on an HTML5 canvas. Thus 2D images, 3D models, text, even animations can be programmed. Toolkits such as A-Frame can be used to rapidly prototype 3D content.

Considerations will be given to designing WebAR using HMDs such as Epson MOVERIO smart glasses. These glasses are ideal for WebAR since their optical see-through display allows for convincing AR effects.

3 BIOGRAPHY

Pablo Mendigochea is head of R&D at HoloLeo studios an AR/VR think tank. His interest in AR and VR spans a decade of investigation on pc, handheld, and head-mounted devices.

REFERENCES

Jerome Etienne. 2017. AR.js project homepage. https://github.com/jeromeetienne/AR.js/. (2017).