The state-of-art platforms and tools which can be used to develop immersive analytical applications

Walid Chtioui[*]
Computer Science M.Sc. Student

Ed Grimley[†]
Computer Science M.Sc. Student

Abstract

This paper presents an overview of the latest state-of-the-art platforms and tools which can be used to develop immersive analytics (hereafter IA) applications. The term platform refers to programs that provide a high-level application programming interface (API) along side a set of extensive XR features which allow developers to create complex XR experiences. Initially, an overview of what we believe are the major extended reality (hereafter XR) development platforms is provided after which a table of, what we think are, important-to-have features and by which platforms they are supported is presented.

Index Terms: HEYHEY

1 Introduction

This template is for papers of VGTC-sponsored conferences which are ***not*** published in a special issue of TVCG.

2 Platforms

2.1 Unity

Unity has a wide adoption in the world of XR thanks to its unified workflow and support for various XR platforms - build once, run everywhere -. Unity supports an extensive set of XR vendor-specific software development kits (hereafter SDK) including: Apple's ARKit, Google's ARCore, Microsoft's HoloLens and OpenXR. Following the announcement of Apple's mixed reality (MR) headset Vision Pro in Apple's Worldwide Developers Conference (WWDC) 2023, Unity was announced to provide native support for Vision's Pro operating system VisionOS [1]. Unity also provides a set of XR packages that are built on top of these vendor plugins to add application-level development tools [2]. For instance, AR Foundation is an industry-standard framework that provides support for various AR features such as: object tracking and plane detection.

2.2 Unreal Engine

2.3 Comparison

Figure **??** provides a comparison between the previously discussed platforms in terms of support for vendor-specific SDKs and a set of features.

3 Toolkits

3.1 DXR Toolkit

Mr. X and al. proposed an IA toolkit built on top of the Unity game engine. DXR allows fast prototyping and iteration for non-experienced users; i.e. users with no or little programming knowledge in XR and Unity. Alongside the data input, DXR takes a specification file written in JavaScript Object Notation (hereafter JSON) from which visualisations are created. The specification file is described in Vega-Like declarative grammar (only what should be achieved has to be provided, not how) making it suitable for users with no programming experience to rapidly realise immersive visualisations. This file can be edited in a separate text editor or through the use of a GUI with pre-configured set of parameters. DXR also provides built-in specification templates for common visualisations such as: Scatter plots and bar charts. This even more extends the scope of users to include those without any technical experience. Although the authors claim that DXR provides suitable flexibility, the scope of that flexibility seems to be limited, among other things, to providing custom graphical markers, custom encoding channels (a channel is a visualisation parameter affected by some data attribute, such as object size affected a quantitative value in data) and other visualisation-type specific properties. That limits users to a templated and common set of visualisations such as Scatter plots, bar charts and radial bars. There is also no mention of real-time data support thus limiting the use-case of DXR to offline data only. As the authors have explicitly mentioned, DXR is meant for prototyping and exploring designs, it is not designed to handle visualisations of large data-sets. On HoloLens, for datasets with more than approximatly a 1000 item, sub-optimal; i.e. less that 60 frames per second (hereafter FPS) performance has been observed. Nonetheless, the author argue that DXR can still be useful for quickly and cheaply prototyping large dataset designs before moving to specialized, optimized and detailed implementations.

	Platform	
Strain	Unity	Unreal Engine
ARCore	X	X
ARKit	X	X
Magic Leap	X	
Microsof HoloLens	X	
OpenXR	X	X
Oculus	X	X
WebXR		
VisionOS	X	

Table 1: Per-platform supported SDKs and XR features.

4 Prototypes

4.1 Uplift

5 Collaboration Tools and Techniques

6 Interaction Tools and Techniques

7 Navigation Tools and Techniques

8 Conclusion

TO BE WRITTEN AT THE VERY END

Acknowledgments

TO BE WRITTEN AT THE VERY END

[*] e-mail: walid.chtioui@ensi-uma.tn
[†] e-mail: ¡EMANIL HERE ACHREF!¿

REFERENCES

[1] Apple. *Bring your Unity VR app to a fully immersive space*. `https://developer.apple.com/videos/play/wwdc2023/10093/`. Last accessed 03 December 2023. 2023.

[2] Unity3D. *XR packages*. `https://docs.unity3d.com/Manual/xr-support-packages.html`. Last accessed 03 December 2023. 2023.