Is Native Naïve? Comparing Native Game Engines and WebXR as Immersive Analytics Development Platforms

Butcher, Peter; Batch, Andrea; Saffo, David; MackIntyre, Blair; Elmqvist, Niklas; Ritsos, Panagiotis D.

IEEE Computer Graphics and Applications

DOI:
10.1109/MCG.2024.3367422

Published: 01/06/2024

Peer reviewed version

Is Native Naïve? Comparing Native Game Engines and WebXR as Immersive Analytics Development Platforms

Peter W. S. Butcher, Bangor University, Bangor, Gwynedd, LL57 2DG, UK
Andrea Batch, U.S. Bureau of Economic Analysis, Suitland, MD 20746, USA
David Saffo, JP Morgan Chase & Co., New York, NY 10017, USA
Blair MacIntyre, JP Morgan Chase & Co., New York, NY 10017, USA
Niklas Elmqvist, Aarhus University, Aarhus, 8200, Denmark
Panagiotis D. Ritsos, Bangor University, Bangor, Gwynedd, LL57 2DG, UK

Abstract—Native game engines have long been the 3D development platform of choice for research in mixed and augmented reality. For this reason they have also been adopted in many immersive visualization and immersive analytics systems and toolkits. However, with the rapid improvements of WebXR and related open technologies, this choice may not always be optimal for future visualization research. In this paper, we investigate common assumptions about native game engines vs. WebXR and find that while native engines still have an advantage in many areas, WebXR is rapidly catching up and is superior for many immersive analytics applications.

Immersive analytics [16] (IA) uses immersive technologies such as Virtual and Augmented Reality (VR/AR) to create visual reasoning environments for analyzing data. Because much of its origins and academics came from the VR/AR research communities, several practices have been drawn from those fields. One of these is the choice of technical platform: many VR/AR research projects tend to adopt native 3D game engines such as Unreal Engine or Unity, and this is also true of IA. For example, ImAxes [7], one of the first IA applications, was built on Unity, and so are the DXR [21] and IATK [6] toolkits.

However, there is an alternative way to build IA applications: using the web. Open web technologies have long led the standardization of interactive computing, and this is also true for the VR and AR fields. The W3C WebXR Device API standard[1] has existed since 2019 and is slowly gaining acceptance in the community as well as adoption among browser manufacturers. When combined with other web technologies such as WebGL, WebGPU, WebAudio, and WebRTC, it is helping the web to become a full-featured platform for creating and delivering immersive analytics applications. The recent VRIA [5] and Wizualization [2] systems demonstrate how these technologies can be used for IA. Such open and free technologies stand in contrast to walled gardens such as Unity, which—as recent unexpected changes to Unity's licensing model have shown—can be a perilous path to trod, even for academics. WebXR, on the other hand, offers an open and standards-based approach to creating IA applications. As visualization researchers, it is hard not to draw parallels to how our field eschewed native 2D graphics toolkits in favor of web-based technologies more than a decade ago. But is the situation different today?

There is currently significant technical debt to native 3D game engines in the IA community, with most efforts using Unity. This may be because building such XR experiences has traditionally been challenging, and opting to use widely established game engines may be a reasonable first step. Or, it may be due to the many persistent assumptions about the state of said game engines vs. WebXR for developing IA applications.

In this article, we enumerate some of these assumptions we have gathered from informal discussions with other members of the IA community, and investigate their legitimacy in the context of IA: (1) performance, (2) compatibility, (3) ease of learning, (4) component ecosystem, (5) rapid development, (6) deployment, (7) interoperability, (8) accessibility, and (9) enterprise integration. Although we focus on Unity, due to its widespread and almost exclusive adoption in IA, we feel these assumptions apply to other native game engines as well. We discuss evidence advocating or opposing these assumptions. We close the article with a future outlook for how WebXR can gain wider acceptance and adoption in the IA community.

Most of the assumptions we explore in this article favor native game engines: that they are fast, have superior hardware support, are easier to learn, and are better for rapid development. But is it true?

Assumption: Native Engines are Faster

The most common claim made by native engine proponents is that it is faster than comparable solutions for WebXR. Such claims echo earlier ones from a decade or more ago when the data visualization field was contemplating making the switch from native to web-based rendering platforms. The answer then, as now, is that the extra overhead from a higher abstraction level does come at the cost of decreased performance. However, for many applications, this decreased performance will be negligible and easily worth the extra convenience of having full access to the rich web technology ecosystem. Furthermore, for all but the most demanding 3D rendering, WebXR is sufficient.

WebXR is a thin abstraction layer on the underlying platform XR APIs, which are typically exposed via OpenXR.[2] It is designed to seamlessly integrate with WebGL and WebGPU, which are in turn thin layers on top of native APIs such as OpenGL, Vulkan, or Direct3D. The web-based 3D engines built on top of these APIs are written in JavaScript or Web Assembly, which have near-native compute performance on modern browsers. This means that, at least in theory, a WebXR application can come close to the same performance as a comparable native application.

To test this assertion, we conducted a small-scale experiment comparing rendering performance of Unity vs. WebXR (using the popular Three.js 3D graphics library) for an unlit 3D scene consisting of 24-vertex cubes. We used three rendering methods:

- Naïve (GameObject, THREE.Mesh);
- Instanced Meshes; and
- Merged Meshes (non-interactive).

Results from our benchmark tests are presented in Table 1 and indicate that Unity outperforms WebXR in each case.

In practice, of course, Unity and Unreal Engine are both highly optimized game engines designed for high-performance 3D games, and Three.js is not aggressively optimized for high performance, so achieving high performance rendering is clearly easier using them, especially when their more advanced graphical features are used. This benchmark currently uses the older WebGL API for graphics, not the newer WebGPU API, so we expect web-based performance to increase as WebGPU is more widely available.

ASSUMPTION *true*

Assumption: Native Engines have Better Hardware Support

Another common claim from native engine proponents is that they are a superior platform for AR and VR research due to their broad hardware compatibility. It is certainly true that many hardware manufacturers tend to prioritize Unity when creating device drivers for their devices. As a result, all of the major XR hardware interfaces have dedicated Unity support. Unity also features full support for the OpenXR[2] standard API.

WebXR, in contrast, is playing catch-up in this area; while significant manufacturers such as Oculus, HTC, Huawei, and Pico all support WebXR in some form or another, this support is typically dependent entirely on the affordances and permissions of browsers available for a given device. Sometimes it is even the browsers themselves that cause bottlenecks against full WebXR support; Google Chrome, Mozilla Firefox, Microsoft Edge, and Opera have decent (if at times buggy) support, but Safari currently has no WebXR support (something which will hopefully be addressed in the future, in light of the release of Apple Vision Pro, as its version of Safari supports WebXR). The lack of WebXR adoption on Safari was the reason one of us created the WebXR Viewer application on iOS at Mozilla, building on our past work on the AR-enabled Argon browser [13].

Similarly, the XR capabilities offered by WebXR are more limited than native engines because web

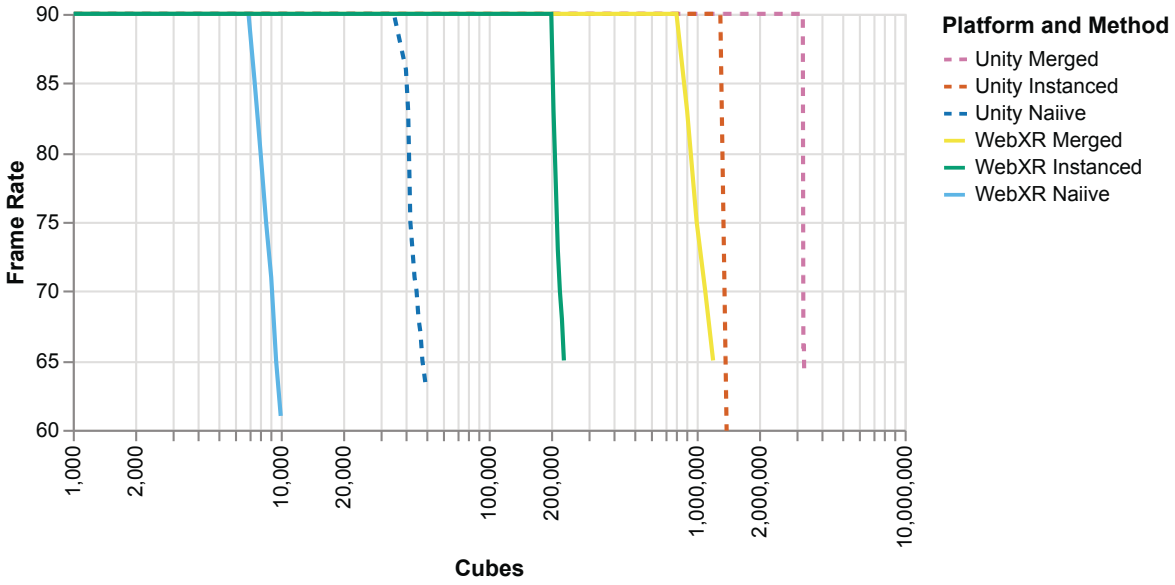

FIGURE 1. Unity versus WebXR (Three.js) benchmark results. 24-vertex cubes with 3 mesh rendering methods. Benchmarking was conducted on Windows 11 Enterprise with a Valve Index HMD. Hardware: 12th Gen Intel Core i9-12900KF 3.19GHz, 32.0GB RAM, NVIDIA GeForce RTX 3080 Ti. Software: Unity 2022.3.4f1, Google Chrome 119.

standards have a high emphasis on security and cross-platform compatibility. As a result, WebXR will not provide immediate access to unique features of new platforms, and may limit access to those that have security or user-privacy implications. However, IA applications may not want to use the latest features of each platform and may instead benefit from targeting a common set of features available across many platforms.

ASSUMPTION *true*

Assumption: Native Engines are Easier to Learn

Its supporters often argue that Unity and Unreal Engine are easier to learn due to the wealth of assets, guides, and interactive editors available in the user community and component ecosystem. The Unity Editor, in particular, is a sophisticated piece of software, and, when combined with its ecosystem of extensions, enables a developer to build a complete 2D or 3D game with only point and click and little to no programming. The step to building advanced immersive analytics tools using a similar interaction methodology is not far. Thus, if the developer is a novice to both web and game development, then Unity is probably easier to learn.

However, the visualization and broader data and business analytics communities have different needs than the game development community. For one thing, visualization researchers and data analysts are more likely to be proficient with web technologies rather than game engines, and have an existing ecosystem of tools and workflows that they can leverage. It will therefore be easier for them to delve into web-based IA. Moreover, the web has the potential to become the de-facto platform for dissemination of immersive visualizations as means of reporting analysis outcomes, similar to traditional visual analytics, which is still an open research opportunity for IA [20]. Furthermore, IDE tools such as PlayCanvas, along with editors for Babylon.js and Three.js and abstractions of these libraries into higher-level DOM components via libraries such as A-Frame,[3] are increasingly making it possible to build advanced WebXR experiences with little or no coding skill needed. Finally, the community is extending familiar, traditional (2D) web visualization tools such as D3 to support WebXR in implementations such as Anu,[4] as well as creating WebXR visualization grammars, such as VRIA [5].

ASSUMPTION *mixed (favoring native engines)*

Assumption: Native Engines have Better Component Ecosystems

The web and native engines such as Unity and Unreal Engine benefit from enormous communities of dedicated developers who create and share their components for others to use. The web community largely embraces the open source software philosophy, uploading components for free (often with nonrestrictive licenses) via services such as GitHub or NPM. Unity developers, on the other hand, have access to the Unity Asset Store, where user-submitted components can be purchased, passing some of the profit back to the creator. However, a large portion of components in the asset store are also available for free and many Unity developers also embrace the open source philosophy by uploading components for free on sites such as GitHub.

Unity currently has a more mature ecosystem of components for creating applications for immersive analytics, including toolkits like IATK [6] and DXR [21], applications such as ImAxes [7], and toolkits for broader VR and MR such as VRTK[5] and MRTK.[6] A number of recent IA systems for evaluating user session data, including MIRIA [4], ReLive [12], and MRAT [17], were all developed in Unity, which we think reflects the maturity and variety of the tools already available on the platform in light of their focus on the more narrow area of analyzing empirical user metrics.

As WebXR continues to mature, however, the ecosystem of components for immersive analytics has been growing as well. For example, Babylon.js now has its own implementation of MRTK, inspired by Unity and Unreal Engine.[7] Several IA libraries and frameworks already exist, including VRIA [5] and Anu.[4] Furthermore, modular systems with exportable components, such as Wizualization [2], have emerged from the IA community within the past year, along with open-source WebXR implementations for empirical studies, such as VRxD [19] and others [3], highlighting a growing body of work among IA researchers in deploying and freely sharing web-based IA tools.

[5] https://assetstore.unity.com/packages/tools/integration/vrtk-virtual-reality-toolkit-vr-toolkit-64131

[6] https://learn.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/mrtk3-overview/

[7] https://doc.babylonjs.com/features/featuresDeepDive/gui/mrtk

ASSUMPTION *true (but leveling out)*

Assumption: Native Engines are Better for Rapid Development Iterations

Web developers have become used to quickly iterating on their applications with modern tooling such as live reloading and hot module replacement. These tools can aid WebXR development; for instance, it is now possible to iterate on code and see the result immediately reflected in the headset without the need to re-enter WebXR or refresh the page. It is this tooling that can enable effective rapid prototyping and development workflows for WebXR projects. When building and shipping a WebXR application, developers can quickly and easily host and serve applications themselves. For enterprise production use-cases there exists a huge range of third-party scalable hosting options. These services can build and serve updates to your code-base in seconds, making your application available on all WebXR supported browsers almost immediately after a new build.

Many native engines such as Unity give developers a fully integrated GUI, scene viewer, and editor. Unity's play mode allows developers to make temporary changes to their applications at runtime, enabling experimentation and rapid prototyping. Changes to scene properties are immediately reflected in any connected XR headset, albeit temporarily until play mode is stopped. However, in our experience, what you see isn't always what you get in Unity: Under some circumstances, a built scene is required to get an accurate reflection of the finished experience. It is more difficult to quickly preview development iterations on several devices at once, requiring builds to be created and distributed to each device for testing. While Unity offers many build targets, including WebGL (supporting WebXR), building and shipping applications for other platforms is a process potentially involving submitting new versions to app stores and awaiting approval.

While iteration time does, of course, vary based on the relative familiarity of the developer with one set of tools or the other, Unity's iteration issues have long been a subject of complaint within the Unity development community to the extent that the Unity team has felt it necessary to address it on multiple occasions.[8,9] In fact, Unity developers are still writing

[8] https://forum.unity.com/threads/improving-iteration-time-on-c-script-changes.1184446/

[9] https://blog.unity.com/engine-platform/better-build-times-and-iteration-speed-for-quest

blog posts about iteration times to this day at the time of our writing.[10] Iteration time issues are so prevalent among the Unity developer community that numerous third-party tools and extensions have been released to address them, such as HotReload,[11] and the Unity team itself has also released patches and tools like the Unity Editor Iteration Profiler,[12] but even these have been insufficient to adequately address developers' complaints. Such misgivings are, in our experience, far less common in the WebXR development community.

ASSUMPTION *false*

ASSUMPTIONS FAVORING WEBXR

There are several aspects that favor WebXR: deployment, interoperability, accessibility, and industry practices. Here we investigate these assumptions in turn.

Assumption: WebXR is Easier to Deploy

Applications developed with WebXR and native engines inhabit almost diametrically opposite parts of the network architecture spectrum. The web is a mostly server-based environment with thin clients and loosely coupled components, whereas Unity and Unreal Engine clients are thick and full-featured with a centralized component ecosystem. This has direct implications when deploying applications. Almost every modern device in existence, including HMDs, has a web browser implementation (even if not all have WebXR support), and thus apps and content can be delivered in a platform-agnostic and dynamic fashion [19]. Running a Unity application, in contrast, requires downloading and installing a native app. Furthermore, the loose dependency structure of the web, while yielding its own set of challenges, means that interoperability and backwards compatibility is relatively easy; a web application can simply be sandboxed for a specific version of its components. Native engines, in contrast, tends to be centrally installed and updated on development machines, causing cascading maintenance effects to its dependencies.

ASSUMPTION *true*

[10] http://blog.s-schoener.com/2023-08-16-why-your-unity-project-is-slow/

[11] https://premortem.games/2023/02/24/new-tool-dramatically-improves-compiling-times-for-unity/

[12] https://forum.unity.com/threads/introducing-the-editor-iteration-profiler.908390/

Assumption: WebXR is More Interoperable

One of the most ambitious goals of immersive analytics, and its neighboring fields of ubiquitous analytics [1], [11] (e.g., anytime anywhere analytics [9]), is to weave data into the fabric of everyday life, spatially aligning physical and computer-generated objects along with their situated data [20], [22]. This concept aligns well with Wendy Mackay's less popular definition of AR [14], which describes an environment augmented through synergies of networked physical and virtual objects, converging with Weiser's ubiquitous computing [23]. For this vision to become reality, interoperability between the physical and the computer generated artifacts in our environment is key [18].

We already live in a highly connected, collaborative world through the Internet and the web, and any incarnation of IA will have to build on this unifying infrastructure [18]. WebXR, with its cross-device support and native synergy potential over the HTML DOM, and thus strong synergies with Internet-of-things infrastructures, offers great potential. More importantly, it makes the browser the single window through which we access a platform-independent cyber-physical world, as we transition between linked, IA-enriched places much like we do today with websites.

ASSUMPTION *true*

Assumption: WebXR Better Supports Accessibility

Accessibility has recently become a hot topic for visualization [10], [15], and there is cause for this in a field that has largely ignored the more than one billion people living with some form of disability [15] that still need to access large-scale data. These concerns are even more pronounced in AR and VR [8], where the bleeding edge of technology tends to prioritize the largest group of users: people with no disability.

There is so far little research on accessible immersive analytics—a shortcoming that must be addressed by future research—but it is pretty clear that IA will face the same, if not further aggravated, accessibility concerns that general AR and VR platforms already do. In this regard, WebXR application's browser-based nature allows leveraging many of the existing and widely used accessibility tools built for the browser. 3D web engines support standard web accessibility features either naively or through third party libraries by generating skeleton HTML DOM structures from the 3D scene graph. This, coupled with descriptive text and navigation techniques allows tools such as screen readers to be used to experience 3D scenes. Additionally, with relatively small effort on the part of a

developer, WebXR's cross-device support gives users the option to choose the platform that best suits their accessibility needs, including platforms that did not exist when the application was originally written.

In contrast, building native applications with engines such as Unity, would require developers to develop or integrate accessibility tools on their own, and limits the number of platforms an application can realistically be built for. In both of these cases, the burden for supporting accessibility is still largely on the developers. However, we believe that WebXR applications and the existing accessibility infrastructure on the web reduces the ask on developers significantly.

ASSUMPTION *true*

Assumption: WebXR Better Integrates with Industry Standards of Practice

Immersive analytics systems and visualizations need to be able to integrate with current industry practices if we hope to deploy real-world applications at any meaningful scale. Developing IA applications natively using game engines such as Unity often means users will need to access IA capabilities through a standalone application. However, a large number of data-driven applications already exist as web-based applications, and developing IA applications with WebXR can leverage this fact. For example, we envision a working environment where a user can jump from a document, discussing some data, to immersive depictions of said data accessible via a head-mounted display.

Additionally, the developers and practices for developing or integrating web applications within a company is likely to be well established, especially in large enterprises. As a result, many companies already have much of the talent and infrastructure they need to begin developing and deploying WebXR apps. In contrast, if a native engine such as Unity is not already established within a company, on-boarding, licensing, and staffing developer teams can become a roadblock to developing and deploying immersive applications.

ASSUMPTION *true*

Much can be said about the current state of web technologies, but it is at least clear that they are proven, battle-tested, and have stood the test of time (at least so far). While the past is not a reliable indicator of the future, we think that betting on the web and its software ecosystem to persist well into the future is rather safe.

FACTOR	LEADING
Performance	Native
Hardware Support	Native
Easy to Learn	Mixed
Component Ecosystems	Native
Rapid Development	WebXR
Deployment	WebXR
Interoperabiliy	WebXR
Accessibility	WebXR
Industry Standards	WebXR

TABLE 1. Summary. Key development criteria for immersive analytics and the currently favored platform.

In particular, the pervasiveness of web technologies across the software engineering industry means that web development skills are ubiquitous; many developers know at least basic web programming. In contrast, finding native game engine developers means plumbing the depths of the 3D game developer market, which is a niche and specialized part of the software engineering industry. While experience with 3D graphics and optimization surely is an asset for complex IA applications, the more general web foundation for WebXR makes it a more appealing technology stack for potential IA developers than native game engines.

Another lesson can be learned from the past success of the web as a technical platform: that open standards survive and flourish, while walled gardens often wither and die. The last year has been a turbulent one for Unity in particular, and while the platform still enjoys a massive market share—especially for games—and has made significant changes in response to recent missteps, the unanticipated licensing changes highlighted the risks of being dependent on a closed platform. And while the ecosystem of components for the platform is vast, the Byzantine licensing and versioning system used by Unity, Unreal Engine, and other native engines makes the availability and regular maintenance of any one component a risky proposition. Obviously, the situation may look different from an academic perspective, where there is a reduced need for long-term software maintenance, but the point still stands as to the longevity of the platform.

Nevertheless, with this Visualization Viewpoint, we issue a strong recommendation for the immersive and ubiquitous analytics communities to transition away from native engines such as Unity and Unreal Engine as technical platforms and to instead adopt WebXR and its associated web technology stack for future research. As our debunking (and confirmation) of persistent assumptions above show, there is a small and shrinking gap between native engines and WebXR

when it comes to pure performance metrics, especially in regard to data visualization. Contrary to the focus of the mainstream XR community on extracting maximum power and leveraging latest graphics features, the visualization community should focus on the significant advantages to adopting the web in terms of quality of life, convenience, compatibility, interoperability, and future potential. By focusing our efforts on the web, we can instead contribute to a common vision of an improved, more flexible, and more performant Document Object Model, maybe even one that better integrates with our surrounding cyberphysical reality.We urge researchers to come on in and join the WebXR movement for immersive analytics—the water is just fine.

Acknowledgments. We thank Piotr Fratczak for his help with Unity. This work was partly supported by Villum Investigator grant VL-54492 by Villum Fonden. This paper was prepared for informational purposes with contributions from the Global Technology Applied Research center of JPMorgan Chase & Co. This paper is not a product of the Research Department of JPMorgan Chase & Co. or its affiliates. Neither JPMorgan Chase & Co. nor any of its affiliates makes any explicit or implied representation or warranty and none of them accept any liability in connection with this paper, including, without limitation, with respect to the completeness, accuracy, or reliability of the information contained herein and the potential legal, compliance, tax, or accounting effects thereof. This document is not intended as investment research or investment advice, or as a recommendation, offer, or solicitation for the purchase or sale of any security, financial instrument, financial product or service, or to be used in any way for evaluating the merits of participating in any transaction.

1. S. K. Badam, A. Mathisen, R. Rädle, C. N. Klokmose, and N. Elmqvist. Vistrates: A component model for ubiquitous analytics. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):586–596, 2019.
2. A. Batch, P. W. S. Butcher, P. D. Ritsos, and N. Elmqvist. Wizualization: A "hard magic" visualization system for immersive and ubiquitous analytics. *IEEE Transactions on Visualization and Computer Graphics*, 30(1), 2024. to appear.
3. A. Batch, S. Shin, J. Liu, P. W. S. Butcher, P. D. Ritsos, and N. Elmqvist. Evaluating view management for situated visualization in web-based handheld AR. *Computer Graphics Forum*, 42(3):349–360, 2023.
4. W. Büschel, A. Lehmann, and R. Dachselt. MIRIA: A mixed reality toolkit for the in-situ visualization and analysis of spatio-temporal interaction data. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, New York, NY, USA, 2021. ACM.
5. P. W. S. Butcher, N. W. John, and P. D. Ritsos. VRIA: A web-based framework for creating immersive analytics experiences. *IEEE Transactions on Visualization and Computer Graphics*, 27(7):3213–3225, 2021.
6. M. Cordeil, A. Cunningham, B. Bach, C. Hurter, B. H. Thomas, K. Marriott, and T. Dwyer. IATK: an immersive analytics toolkit. In *Proceedings of the IEEE Conference on Virtual Reality*, pages 200–209, Los Alamitos, CA, USA, 2019. IEEE Computer Society.
7. M. Cordeil, A. Cunningham, T. Dwyer, B. H. Thomas, and K. Marriott. ImAxes: Immersive axes as embodied affordances for interactive multivariate data visualisation. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, pages 71–83, New York, NY, USA, 2017. ACM.
8. J. Dudley, L. Yin, V. Garaj, and P. O. Kristensson. Inclusive immersion: a review of efforts to improve accessibility in virtual reality, augmented reality and the metaverse. *Virtual Reality*, 2023.
9. N. Elmqvist. Data analytics anywhere and everywhere. *Communications of the ACM*, 66(12):52–63, Dec. 2023.
10. N. Elmqvist. Visualization for the blind. *Interactions*, 30(1):52–56, 2023.
11. N. Elmqvist and P. Irani. Ubiquitous Analytics: Interacting with Big Data Anywhere, Anytime. *Computer*, 46(4):86–89, 2013.
12. S. Hubenschmid, J. Wieland, D. I. Fink, A. Batch, J. Zagermann, N. Elmqvist, and H. Reiterer. ReLive: Bridging in-situ and ex-situ visual analytics for analyzing mixed reality user studies. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pages 24:1–24:20, New York, NY, USA, 2022. ACM.
13. B. MacIntyre, A. Hill, H. Rouzati, M. Gandy, and B. Davidson. The Argon AR web browser and standards-based AR application environment. In *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality*, pages 65–74, Los Alamitos, CA, USA, 2011. IEEE Computer Society.
14. W. E. Mackay. Augmented reality: Linking real and virtual worlds: A new paradigm for interacting with computers. In *Proceedings of the ACM Conference on Advanced Visual Interfaces*, pages 13–21, New York, NY, USA, 1998. ACM.
15. K. Marriott, B. Lee, M. Butler, E. Cutrell, K. Ellis, C. Goncu, M. A. Hearst, K. F. McCoy, and D. A. Szafir. Inclusive data visualization for people with

disabilities: a call to action. *Interactions*, 28(3):47–51, 2021.

16. K. Marriott, F. Schreiber, T. Dwyer, K. Klein, N. H. Riche, T. Itoh, W. Stuerzlinger, and B. H. Thomas, editors. *Immersive Analytics*, volume 11190 of *Lecture Notes in Computer Science*. Springer International Publishing, New York, NY, USA, 2018.
17. M. Nebeling, M. Speicher, X. Wang, S. Rajaram, B. D. Hall, Z. Xie, A. R. E. Raistrick, M. Aebersold, E. G. Happ, J. Wang, Y. Sun, L. Zhang, L. E. Ramsier, and R. Kulkarni. MRAT: The mixed reality analytics toolkit. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, page 1–12, New York, NY, USA, 2020. ACM.
18. J. C. Roberts, P. D. Ritsos, S. K. Badam, D. Brodbeck, J. Kennedy, and N. Elmqvist. Visualization beyond the desktop – the next big thing. *IEEE Computer Graphics and Applications*, 34(6):26–34, Nov. 2014.
19. D. Saffo, A. Batch, C. Dunne, and N. Elmqvist. Through their eyes and in their shoes: Providing group awareness during collaboration across virtual reality and desktop platforms. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*, pages 383:1–383:15, New York, NY, USA, 2023. ACM.
20. S. Shin, A. Batch, P. W. S. Butcher, P. D. Ritsos, and N. Elmqvist. The reality of the situation: A survey of situated analytics. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–19, 2023.
21. R. Sicat, J. Li, J. Choi, M. Cordeil, W.-K. Jeong, B. Bach, and H. Pfister. DXR: a toolkit for building immersive data visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):715–725, 2019.
22. B. H. Thomas, G. F. Welch, P. Dragicevic, N. Elmqvist, P. Irani, Y. Jansen, D. Schmalstieg, A. Tabard, N. A. M. ElSayed, R. T. Smith, and W. Willett. Situated analytics. In *Immersive Analytics*, volume 11190 of *Lecture Notes in Computer Science*, pages 185–220. Springer, 2018.
23. M. Weiser. The computer for the 21st Century. *Scientific American*, 265(3):94–104, 1991.

Peter W. S. Butcher is a Lecturer in the School of Computer Science and Engineering at Bangor University in Bangor, UK. His current research interests include data visualization, immersive analytics, and web-based XR. He received his Ph.D. in computer science from the University of Chester. Contact him at p.butcher@bangor.ac.uk.

Andrea Batch is an Economist at the U.S. Bureau of Economic Analysis in Suitland, MD, USA. Her current research interests include data visualization, immersive analytics, and human-computer interaction. She received the Ph.D. degree in information studies from University of Maryland, College Park. She is a member of the IEEE Computer Society. Contact her at andrea.c.batch@gmail.com.

David Saffo is a Senior Applied Research Associate at JP Morgan Chase in New York, NY, USA. His current research interests include data visualization, immersive analytics, and cross-reality collaboration. He received the Ph.D. degree in computer science from Northeastern University. He is a member of the IEEE Computer Society. Contact him at david.saffo@jpmchase.com.

Blair MacIntyre is Global Head of Immersive Technology and Spatial Computing Research at JP Morgan Chase in New York, NY, USA. His current research interests include human-computer interaction, cross-reality collaboration, and enterprise XR. He received the Ph.D. degree in computer science from Columbia University. He is a member of the IEEE Computer Society. Contact him at blair.macintyre@jpmchase.com.

Niklas Elmqvist is a Villum Investigator and Professor in the Department of Computer Science at Aarhus University in Aarhus, Denmark. His research interests include data visualization, human-computer interaction, and human-centered AI. He received the Ph.D. degree in computer science from Chalmers University of Technology. He is a Fellow of the IEEE and the IEEE Computer Society. Contact him at elm@cs.au.dk.

Panagiotis D. Ritsos is a Senior Lecturer in the School of Computer Science and Engineering at Bangor University in Bangor, UK. His current research interests include data visualization, mixed reality, and human-computer interaction. He received the Ph.D. degree in electronic systems engineering from University of Essex. He is a member of the IEEE Computer Society. Contact him at p.ritsos@bangor.ac.uk.