Immerse: Game Engines for Audio-Visual Art in the Future of Ubiquitous Mixed Reality

Matthew Hughes[1], Jaime A. Garcia[1], Felicity Wilcox[1], Robert Sazdov[1], Andrew Johnston[1] and Andrew Bluff[1]

[1] University of Technology Sydney, Sydney, Australia
matthew.d.hughes@student.uts.edu.au
jaime.garcia@uts.edu.au
felicity.wilcox@uts.edu.au
robert.sazdov@uts.edu.au
andrew.johnston@uts.edu.au
andrew.bluff@uts.edu.au

Abstract. Through the context of the mixed-reality performance work 'IMMERSE', this paper discusses the use of game engines as a practical medium for the creation of audio-visual art. The game engine's metaphorical and spatial consistency with reality, flexible deployment options and capacity for high-fidelity graphics make them a perfect tool for constructing the artworks of the twenty-first century.

Keywords: mixed reality, audio visual art, composition, game engines

Introduction – The science fiction of a mixed reality

Nested in the footer of designer and film-maker Keiichi Matsuda's website sits a randomly selected quote. Today it presents a gem from J.G. Ballard:

> Everything is becoming science fiction. From the margins of an almost invisible literature has sprung the intact reality of the 20th century.

It accompanies Matsuda's concept works that explore a future where technology is immersive, pervasive and ubiquitous. His 2016 short film 'Hyper-Reality' (Matsuda 2016) transports the audience to a point-of-view bombarded by mixed-reality media (figure 1). Virtual objects are planted inside the day-to-day and appear as integrated as any physical object. It is not clear what technology is rendering the holographic imagery, but as Ballard noted for his own century, science-fiction literature is charged with the task of discussion – scientists will find the necessary techniques to transform the fiction into reality.

Much of what is described in 'Hyper-Reality' is materialising itself right now. Computer-generated cat-ears already crown our heads and 'Pokémon' roam our streets. Mobile and headset-based augmented reality is transforming the once immovable window to the virtual world into an interactive viewport. CAVE environments and AR headsets provide researchers and artists with mixed-reality play-spaces, and numerous technologies are sure to follow that continue to liberate us from the flat-panel.

To researchers, the concept of creating a ubiquitous computing environment is becoming increasingly important – and to artists, these developments in technology offer an exciting opportunity; a new medium in which to produce work, entangled with questions of how to present the imagined and the synthesised among us.

In this paper, we present 'IMMERSE' – an audio-visual work that combines live music and spatialised sound design with audio-reactive visuals in a CAVE-like (Cruz-Neira et al. 1992) environment. The work wraps the performance area in 360-degree projections, with the audience onlooking from inside the cylindrical space. In this mixed-reality performance piece, musicians interact in real-time with the virtual environment that surrounds them. At the core of the system's architecture is a video game engine, housing virtual objects manipulated by the music in real time.

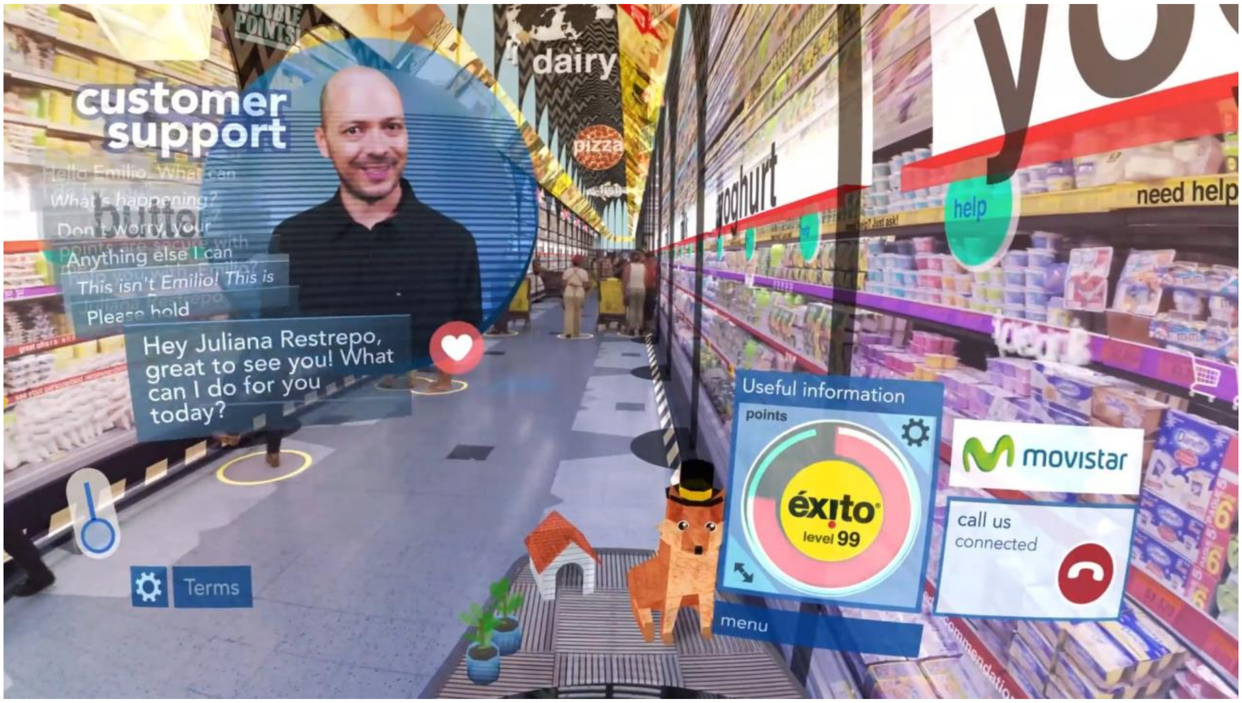

Figure 1. Caption: A still from Keiichi Matsuda's short film 'Hyper-Reality' (2016).

The game engine is becoming a key tool for carving out the immersive, pervasive and ubiquitous future of computing described by Matsuda and his science-fiction contemporaries – its place in our media is stretching far beyond what its name suggests, and the features of a game engine make it a versatile medium for those exploring the nexus of art and technology. This paper will explore the benefits of using the game engine as a medium, and through the context of our live performance piece 'IMMERSE', seeks to demonstrate that the object-based, spatially consistent, and widely deployable nature of game engines make them a perfect tool for constructing the interactive artworks of the 21st century.

Usual systems for audio-visual art

There are several approaches used by artists to create audio-visual works, although one of the most commonly used methods is to construct the system using a combined multimedia programming environment such as Max or its open-source cousin Pure Data (Pd). Max is a graphical environment in which applications are "patched" together by visually drawing connections between independent nodes (figure 4). Its component libraries MSP and Jitter contain nodes to construct and manipulate sound and visuals respectively. By combining audio and graphical development in the same environment, an artist who uses this type of system can easily achieve a high level of integration between the audio and the visuals. Visualisations of audio can be constructed by feeding sound directly into graphical nodes, and sonification of visuals can be achieved by doing the opposite (Jones & Nevile 2005).

The Cosm toolkit (Wakefield & Smith 2011) uses Max to design for CAVE environments, and the authors chose the highly integrated system over alternatives to ensure tight coupling between modalities. They do concede however, that the power of Jitter's graphical engine is not able to match that of a game engine – a point which is echoed by audio-visual composer Tadej Droljc (2017). Droljc uses Max to produce his compositions, and praises its immediacy, but the lack of fidelity available in its graphics engine forces him to use Max primarily as a compositional tool. For finalising visuals in high-quality, his animations are rendered offline using more traditional 3D animation techniques. He presents many pain points when using a system based on Max, including limited support for processing 3D meshes and inconsistent surface lighting, and also implies that a high degree of technical and programming skill is necessary, describing the process of importing usable 3D models into Max as “very complicated”.

Acknowledging these factors, audio-visual artists can combine the strong audio engines of Max/PD or alternative digital audio workstations with environments focused on graphics. VVVV and TouchDesigner are two examples of popular software environments used to create graphics for interactive artworks, and some artists even create their own graphical engines using languages like C++ along with creative coding libraries such as OpenFrameworks and Cinder (Johnston 2013; Correia 2015).

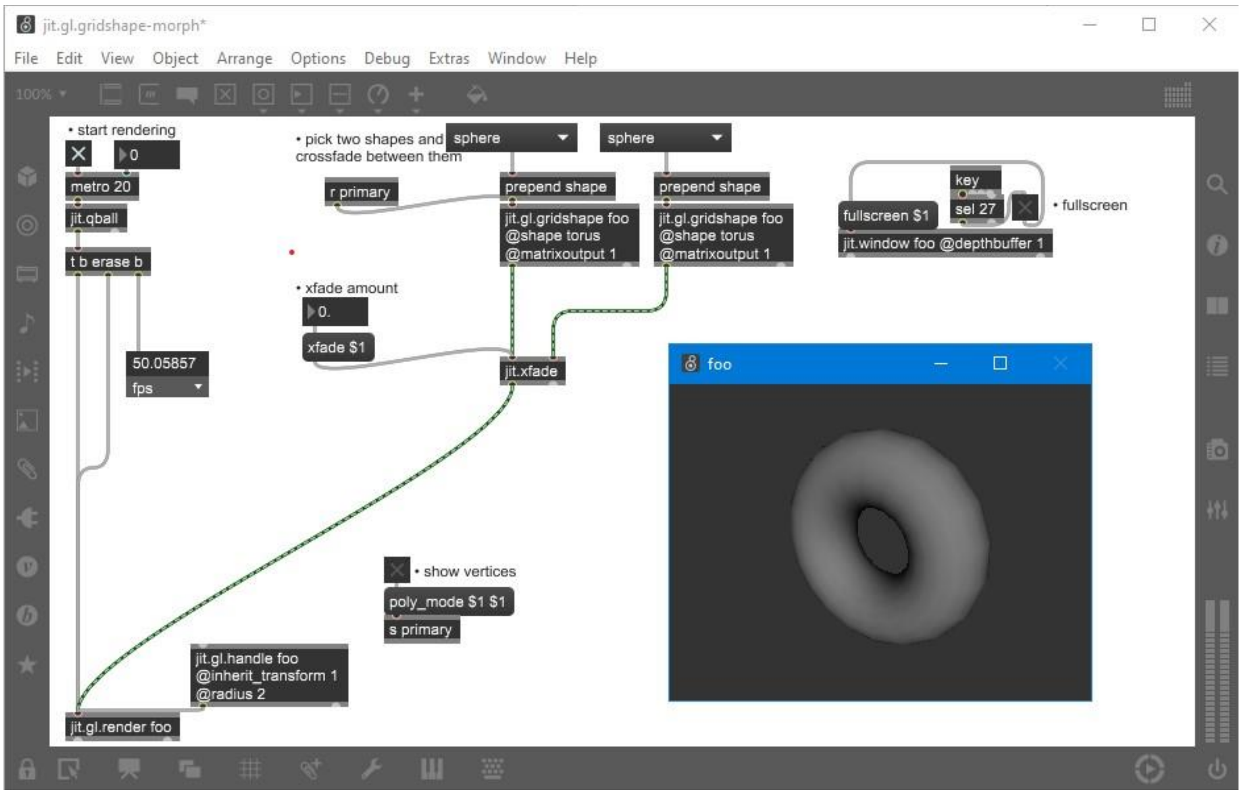

Figure 2. Cycling ‘74’s Max software environment and its ‘patcher’ interface.

Benefits of a game engine

A video game engine, in comparison to the methods previously outlined, offers artists a high fidelity, real-time graphics pipeline without needing to write low-level code – perhaps even no code at all.

Today, the video game industry is more profitable than the film and music industries combined (LPE 2018), and consequently, a great deal of high-quality tooling has been developed in the form of feature-rich game engines such as Unity and Unreal. These engines provide the user with high level control over meshes, lighting, physics, particle systems, fluid and cloth simulations, with huge communities (and corporations) furthering the state-of-the-art in these technologies and their interfaces. Without diminishing the welcoming communities of the alternative software environments, it is worth noting the company behind the free-to-use Unreal game engine, Epic Games, earned $3 billion in profits for 2018 (Kain 2018) – money which can be seen in the numerous features added in each of the engine’s frequent new releases.

Indeed, commercial game engines offer a higher-quality and more flexible graphics pipeline out-of-the-box as opposed to the alternatives audio-visual artists often use – but the most vital change is not this upgrade in fidelity. Game engine’s present a conceptual advantage to the process of audio-visual composition through their strong focus on the concept of 'game objects'.

The vision of the future painted in Matsuda's 'Hyper-Reality' is not one where creators design for the limitations of screen-space. You won't find a flat screen throughout his entire short. In this future, the existence of the virtual is bound to the citizen's own world-space, where each digital experience – be it a game, an advertisement, or in our case a work of art – exists as a 3D entity inside the real world.

In a game engine, everything presented within the environment exists as an object made from *attributes* – representing the object's current state – and *behaviours* – defining how that state should change over time and in response to events (Gregory 2017). These objects are typically referred to as entities, agents, or 'game objects', and can be easily manipulated within a 3D scene that is analogous to our real world.

As mixed-reality experiences become more prevalent and the science-fiction future emerges in front of us, the world-space canvas presented to game designers along with its game object metaphor is highlighting itself as a practical medium for a variety of real-time media – not just gaming. Game engines have always been designed to resemble the world of our own, and modern game engines include an 'editor' that allows for manipulation of game objects in ways that remain within this world/object metaphor. The metaphorical consistency with our own experience is important, because it is notably absent or convoluted in some of the alternative software environments used by audio-visual artists.

As an example of this metaphorical consistency between the world in a game engine and our reality, let's take this paper you're reading (and foolishly assume you're not doing so on a screen). You're probably reading it inside a room – or perhaps you're enjoying it outside in the sun – either way, it exists, and it does so inside a unique environment. You can place this paper down in front of you, inside your unique environment very easily. Now if you have a second paper, you could easily take that and stack it on top of this one. With a game engine, the process is analogous. If you were to load a model of this paper as a game object, you could insert it into the game world and stack more on top by adding additional "paper" objects into the scene – moving them about as you please by dragging them within the editor's interface.

If you attempted this same feat in Pd's GEM, the process is less intuitive. You would connect a "gemhead" node (which represents the rendering world) into a 3D model node. Immediately this metaphor is at odds with reality, because the connection order insists we connect the world to the object and not vice-versa. Adding a second 3D model, you'd need to create a "separator" to tell GEM the objects must exist in separate locations. Max's Jitter environment is closer to reality. You can create a world node and populate it with object nodes. Each object can be moved about the virtual world independently, but being able to manipulate the object's position interactively isn't available to us without additional programming. These details may seem insignificant, but the game engine's ability to navigate, populate and visualise a 3D scene with its editor – whilst being able to manipulate its contents intuitively – demonstrates the medium's immediacy and suitability for creating works that exist in a world of mixed-reality.

Audio-visual artworks that utilise video game engines for graphics typically don't use the game engine for audio as well though. The maturity of Max and Pd, and their unquestionably greater focus on sound manipulation make them far better candidates for handling audio than anything built into a game engine itself. Therefore, it is common for Max, Pd or similar software to be used as an audio backbone bound to a game engine handling the graphics. The most common way to unify the two engines – facilitating the binding of sound and game object attributes – is with an internal network connection, as demonstrated in (Dolphin 2009), and often through use of the Open Sound Control (OSC) protocol (Hamilton 2011).

This type of system that introduces a network layer in order to bind two separate engines has been denounced by some creators due to its implicit looser coupling between modalities (Wakefield & Smith 2011), but projects like Chunity (Atherton & Wang 2018) and Libpd (Brinkmann n.d.) offer solutions to embed the external audio engines inside the game engine itself, allowing for shared memory and simultaneous event handling without introducing the latencies of a network. Additionally, at the time of writing, Epic Games' latest Unreal engine release has enabled by default a new built-in audio engine – one that could potentially rival external alternatives, and as a first-party integration, may offer the most unified approach moving forward (Lerp 2019) .

Figure 3. A dress-rehearsal for 'IMMERSE' inside the data arena.

IMMERSE

Our project, entitled 'IMMERSE', is a mixed-reality audio-visual performance work that combines chamber music, spatialised sound design, and interactive visualisations. It was presented as part of the 2019 Sydney Festival inside the University of Technology Sydney's 'Data Arena' – a CAVE-like environment that projects images in 360-degrees, and houses a sixteen-channel surround sound system (figure 3).

Like many of our contemporaries, we chose to build a hybrid system that couples a game engine with an external audio engine over a local network (figure 4). Live audio from two performers, on viola and cello respectively, is fed through wireless microphones into Pd, where it is combined with spatialised electronic sound compositions controlled by a third performer. The Pd patch performs spectral analysis, pitch detection, and transient detection on both live audio sources, as well as for the 16-channel electronic sounds. The electronic soundscapes feature the surround speaker system heavily, swirling noise around the audience seated in a ring facing the performers in the centre. The volume of each of the 16 speaker channels is also monitored, and the location of the loudest speaker is calculated in terms of its position on the room's circumference, so that the aforementioned swirling can be tracked.

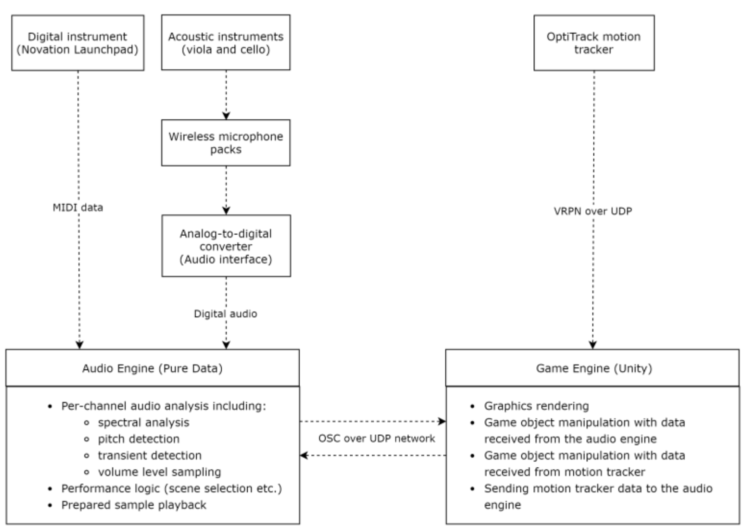

Figure 4. The system architecture of `IMMERSE`.

Each of these data streams created by Pd is piped to an instance of the Unity game engine running on the Data Arena's graphics server, which houses an array of graphics cards and is connected to six projectors. Game object attributes in Unity are bound to the incoming network messages, allowing the performers direct manipulation over the virtual environment through their musical gestures. For one scene, an OptiTrack motion tracking system is also utilised, and the position of some game objects are set to follow the viola player as he bounds throughout the performance space and around the audience.

A recurring pattern throughout the work is a motif we call the "connective tissue", which is composed of a particle system that reacts to abstract electronic sounds. Spectral data is used to animate the particle system's attributes, and the position of the particle system is guided by the surround sound position of the audio. Figure 5 shows this visualisation inside the Unity editor, alongside a representation of the physical performance area in proportional scale, where one spatial unit in Unity is equivalent to one metre in real life. The black circle represents the floor of the Data Arena, and the pink cubes are markers which orient the visualisations during the design process. The game engine's editor – being easily navigable and spatially consistent with reality – allows the designer to comprehend the experience of the Data Arena without needing to be inside the space itself. In fact, the principal designer of 'IMMERSE' was able to generate a series of game objects like this one on a laptop from the opposite side of the world, and be confident that they would present themselves as intended inside the Data Arena. Of course, the impact of the Data Arena's scale does not translate to a laptop screen, but the fact this virtual object is transferrable to a highly specific environment highlights another advantage to composing artworks in a game engine – flexibility over the technological system used for presentation. Unity can be used to build experiences for PCs, consoles and mobile. It can target VR headsets or AR glasses, and all that's necessary for us to deploy to the Data Arena is to insert a virtual "camera" suitable for our projector configuration.

As we close in on Matsuda's "hyper real" future, numerous techniques are being bred that allow the virtual to coexist in our spaces, but there isn't a singular presentation mechanism that's assumed for mixed reality artworks. CAVE environments like our Data Arena are fundamentally site specific. Headsets are – at this point in time – expensive and unergonomic, and their intermediary – the smartphone – requires users to fumble with screens in front of their face.

Holograms as depicted in Hollywood are a far-cry from reality, and it's unlikely there will ever only be one technology to account for all mixed-reality presentations. By composing audio-visual artworks within a scene of component game objects, the presentation technology is distanced from the composition itself. 'IMMERSE' has been designed so that the audience is encircled by visualisations – quite obviously because the Data Arena is constructed this way – but now the work is completed, it would be trivial to present the same production inside a VR headset because it is offered by Unity as a deployment target. We could build our show's scenes for a different CAVE space at another institution, or we could tell Unity to build for iPhone and deploy it as an augmented reality app. Because the game object's attributes and behaviours, along with the world's scale remains consistent no matter the deployment target, each experience would still effectively showcase our artistic intentions. When designing for mixed reality – a new artistic medium whose mechanisms for presentation are not standardised – the technological agnosticism provided to us with game engines can simplify development, allow exploration in multiple presentation methods, and in theory extend the lifetime of the artwork itself.

Future development

In future works of this nature – contrary to how we approached the 'IMMERSE' system – an architecture that allows the audio engine to be embedded within the game engine would certainly be an improvement. Instead of networking with an external audio software, an embedded approach (like some mentioned in section 3) would allow us to bind sonic parameters more directly to game objects. The improved audio engine present in Epic Game's new Unreal engine releases would simplify this even further – so we are watching with interest to see if it presents a competitive first-party alternative. For 'IMMERSE', the networked approach didn't present itself as an obstacle per se – and we are perfectly satisfied with the level of audio-visual integration we have been able to achieve – but in order to further explore the game engine's use as an artistic medium, focusing on integrating modalities within the game objects themselves seems like a logical path forward.

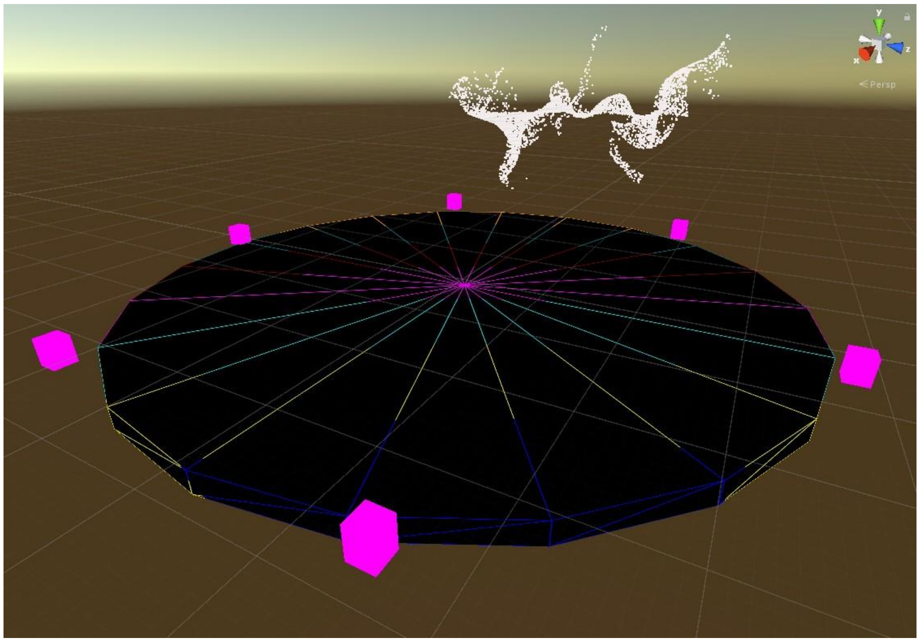

Figure 5. The “connective tissue” visualisation inside the Unity editor. The black circle is the floor space of the Data Arena, and each pink cube marks the centre of a projection screen.

Conclusion

In this paper, we started by describing a future of mixed reality. An imagined future indeed – but as is the nature of science fiction, time transforms the author’s work into something less resembling fiction as it passes. A recent surge of interest in mixed reality – along with advancements in technologies that seek to make its presentation possible – is set to deliver us a virtually augmented world sooner than one might think. Compared to the alternative approaches

in audio-visual software, game engines offer a flexible approach to high fidelity development, whose metaphorical consistency with our own world make them apt for mixed reality works. Through discussing the development of ‘IMMERSE’, we hope to have demonstrated this suitability, and propose that game engines – as tools long used to prepare virtual worlds – are a means of content creation particularly suited for the future of mixed reality upon us. We implore

References

Atherton, J. and Wang, G. 2018. Chunity: Integrated Audiovisual Programming in Unity. Proceedings of the International Conference on New Interfaces for Musical Expression 2018. 102–107

Correia, N.N. 2015. Prototyping Audiovisual Performance Tools : A Hackathon Approach. In Proceedings of the International Conference on New Interfaces for Musical Expression 2015, 12–14.

Cruz-Neira, C., Sandin, D.J., DeFanti, T.A., Kenyon, R. V, and Hart, J.C. 1992. The CAVE: audio visual experience automatic virtual environment. Communications of the ACM 35, 6, 64–73

Dolphin, A. 2009. Compositional applications of a game engine: Creative practical applications in sound art and music composition. In Proceedings of the 1st International IEEE Consumer Electronic Society’s Games Innovation Conference, 213–222

Droljc, T. 2017. Using Max and Live as an environment for audiovisual composition. Presentation at the Electric Spring, Max/MSP Power Users Symposium 2017. Retrieved from https://www.youtube.com/watch?v=qgLtp9WNpuo

Gregory, J. 2017. Game Engine Architecture. AK Peters/CRC Press. MA, USA

Hamilton, R.K. 2011. UDKOSC : An immersive musical environment. In Proceedings of the International Computer Music Conference 2011. 717–720

Johnston, A. 2013. Fluid Simulation as Full Body Audio-Visual Instrument. In Proceedings of the International Conference on New Interfaces for Musical Expression 2013, 132–135

Jones, R. and Nevile, B. 2005. Creating Visual Music in Jitter: Approaches and Techniques. Computer Music Journal 29, 4, 55–70

Kain, E. 2018. ‘Fortnite’ Creator Epic Games Reportedly Earned $3 Billion In Profits in 2018. Forbes online. Retrieved from https://www.forbes.com/sites/erikkain/2018/12/27/fortnite-developer-reportedly-earned-3-billion-in-profits-in-2018

League of Professional Esports. 2018. The Video Games’ Industry is Bigger Than Hollywood. Retrieved from https://lpesports.com/e-sports-news/the-video-games-industry-is-bigger-than-hollywood

Lerp, V. 2019. Unreal Engine 4.22 preview release summary. https://forums.unrealengine.com/unreal-engine/announcements-and-releases/1583659-unreal-engine-4-22-preview

Libpd. Peter Brinkmann. http://libpd.cc/

Matsuda, K. 2016. Hyper-Reality. Fractal, Colombia. Retrieved from http://hyper-reality.co/

Max, Cycling 74. https://cycling74.com/

Pure Data, Miller Puckette. https://puredata.info/

Unity. Unity Technologies. https://unity.com/

Unreal Engine. Epic Games. https://www.unrealengine.com/

Wakefield, G. and Smith, W. 2011. Cosm : a Toolkit for Composing Immersive Audio-Visual Worlds of Agency and Autonomy. In Proceedings of the International Computer Music Conference 2011