CLOUD-BASED AI FOR AUTOMATIC AUDIO PRODUCTION FOR PERSONALISED IMMERSIVE XR EXPERIENCES

R. G. Oldfield[1], M. S. S. Walley[1], B. G. Shirley[1] and D. L. Williams[2]

[1] Salsa Sound Ltd, UK and [2] BT, UK

ABSTRACT

In this paper we focus on the machine learning approach we have developed for automatic audio source recognition and mixing for the UK DCMS funded collaborative project called 5G Edge-XR. Leveraging GPU acceleration, we deployed innovative algorithms in the cloud so that content can be automatically mixed on-the-fly for a personalised, immersive and interactive experience for audiences. In particular we will describe the algorithms involved, the system architecture and how it has been implemented for immersive live boxing and also how we are using it to enhance a live in-stadium experience.

INTRODUCTION

In this paper we present work being carried out in the UK DCMS funded, 5G Edge-XR project. The project is led by BT and is a collaboration between several SMEs, The University of Bristol and a Dance school in the UK. The project is exploring how a combination of 5G connectivity and a GPU cloud capability at the network edge can affect the delivery and experience of immersive experiences including those based on augmented, virtual and mixed reality – collectively 'XR experiences', to consumer devices including headsets, smart glasses, phones and tablets. The project includes a particular focus on real-time experiences where the viewers can change the content viewpoint freely on an AR headset with the rendering being done live, in real-time in the cloud and delivered to the end-user over a 5G network.

The AV content used will depend on the use-case but will be made up of volumetric video, broadcast production content and additional data and video feeds which the user has control over the rendering of. Traditionally, it would not be possible to facilitate these experiences for the end-user without viewers having a fast internet connection and a powerful GPU capability in their consumer devices with the compute power to drive the bespoke audio and visual representation. In 5G Edge-XR we are exploring the use of 5G networks for the provision of high the bandwidth links required to send the live AV components to a cloud-based GPU capability at the network edge which performs computation and rendering in order to deliver the live personalised content to the end-user device over the 5G network.

This paper focuses on the audio system of the project and in particular the machine learning approach for automatic audio source recognition/extraction, composition and mixing. Leveraging GPU acceleration, we have deployed these innovative algorithms in the cloud so that content can be automatically mixed on-the-fly for personalised, immersive and interactive experience for audiences. The paper is structured as follows. We will begin by outlining the 5G Edge-XR project, its motivation, technical features, architecture and use

cases and proceed to discuss the (object-based) audio system in more detail, highlighting in particular the AI-driven audio source analysis, extraction and composition engine. Next, we will describe how the audio scene can be rendered in the Unity gaming engine to match the visual presentation and finish up with a discussion on the project outlook and conclusions.

5G EDGE XR PROJECT

5G Edge-XR will demonstrate that 5G networks, coupled with cloud graphics processing units, will enable users to view sporting events from every angle in a totally immersive experience. These experiences will be available on a range of devices including smartphones, tablets, AR and VR headsets and TVs. This approach will underscore the vision and potential of 5G networks and how it can transform content composition and delivery, bringing new experiences to many sectors

Project/system architecture

The end-to-end chain of the experiences in the 5G Edge-XR project can be represented as in Figure 1 below. Assets are generated, possibly in real-time from cameras and microphones, and are encoded and uploaded to the GPU processing system. The processor renders the assets into a scene and generates, using knowledge of the pose and orientation of the client device, a view onto that scene and renders both visual and audio streams to represent the view from that position. In this configuration the requirements of the client device are relatively simple; they are required to present pose and orientation requirements to the GPU processing and to be able to decode and present an AV signal. The client enjoys the privilege of then being able to choose from which to view the rendered scene.

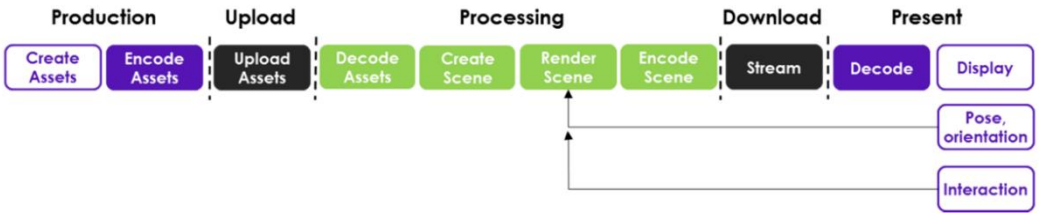

Figure 1 – End to end chain simplifying and generalising the process by which the scene the end user views and hears is generated.

The logical and physical location of the GPU affects the end-to-end delay budget. Internet based cloud resources typically add 20-30ms to the round-trip time. Thus, siting the GPU at the logical edge of the network (see Figure 2), prior to exiting the network via a peering node, we hope to reduce the end-to-end delay and expect this to result in a better, more responsive user experience.

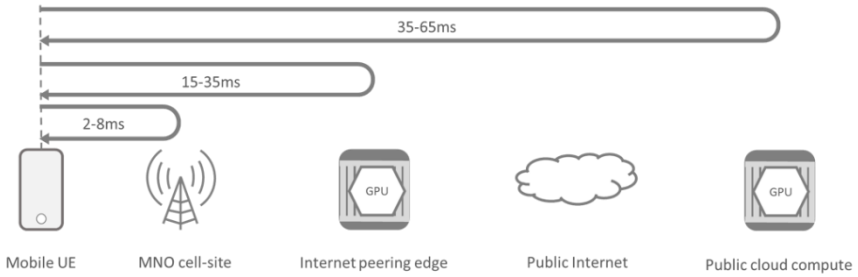

Figure 2 – Simplified network routing between the mobile device and the GPU cluster from where the experience is streamed, with associated estimated round-trip delays.

In some use cases the scene generated may be a near real-time render of a volumetric capture. In this work we are using a dance lesson and a boxing match as exemplar scenes that are being observed and captured volumetrically. In time, as the technology evolves the size of the scene captured may increase to include a whole theatre stage, a television studio, a tennis court or even football or rugby pitch.

Volumetric captures generate huge amounts of data, from cameras and depth sensors which must be analysed and encoded as point clouds and textures which are transmitted to the GPU cluster to be rendered, frame by frame, into a photorealistic hologram. The real-time challenge is to generate a virtual camera view onto the rendered scene which is sufficiently similar to the image formed from a real camera placed at the same position as the virtual camera within the scene. Achieving that structural similarity requires complex calculations well suited to GPUs, including real-time rendering, AI and deep learning to improve encoding methods.

Alongside the development of the visual point-of-view, another challenge, as discussed in this paper, is the task of generating a soundscape appropriate to the position and content of the virtual camera.

Use cases

There will be several use cases that will be demonstrated as part of this project. For the purpose of this paper, we will discuss only the boxing and in-stadium demonstrations.

Boxing

The boxing use case involves the generation of a photo realistic hologram which, using AR, may appear to be situated on viewers' coffee tables. This is powered by the volumetric video capture and composition by project partner, Condense Reality. In the concept it will be possible to observe the action from any angle, on smartphones, tablets and AR headsets. The presentation will be synchronised with the live TV Broadcast feed and will contain interactive elements allowing the viewer to personalise their presentation. For example, they may include the ability to select a replay from a list of replays or to interact with graphics to provide more information - for example to choose whether or not to include data panels providing real-time information about the bout and the pugilists etc.

Figure 3 – 5G Edge-XR boxing use case concept

In the concept being built, the experience will be editorially driven with commentary and automatic replays. Because viewers can walk around the hologram that appears on the coffee table it is obvious that the audio must change to match the changing perspective of the viewer. The viewers will experience an audio 'bed' with the main (background) audio feed but as they add/remove content and navigate within the scene they will also have overlaid a bespoke audio feed which will match the visual representation. For example, we will isolate each audio source and localise it to the correct place in the scene so that the sound of each punch and shout of each trainer can be panned to the correct location relative to the viewing angle of each viewer. The challenge from an audio

perspective therefore is to be able to extract each sound source/object and localise it in space in real-time within a noisy environment. As described later in this paper, we have approached this challenge using a real-time deep-learning technique based on the raw input audio. In addition, we will be leveraging the power of the cloud to perform speech-to-text and other metadata extraction routines to drive more content and personalisation for viewers.

In-stadium

Typically, there are over 700,000 fans attending stadiums for English football matches every week in the UK and as a result there is a drive by many clubs to reimagine and improve the perceived value of a matchday ticket. An important part of this is the use of appropriate technology to enhance the experience by providing unique access to in-stadium content an enable fans to get an enhanced/improved in-stadium experience. New content is also a way of mobile service/content providers promoting themselves above the competition.

The in-stadium, use case will hence demonstrate an AR football experience for sports fans within stadia. Again, viewable on AR headsets, phones or tablets, viewers will have the option to have stats/data overlaid onto the scene they can see. This may include player names, virtual off-side guides, gain lines, refereeing decisions etc. Also available will be text overlays synchronised to the commentary, replays shown on virtual jumbotron screen, video from other camera viewpoints and alternative partisan commentaries. Our audio engine will be used again to segment and analyse the audio feeds to drive graphics overlays and additional content. For example, we use the sound of the referee's whistle for accurate localisation and trigger a 3D speech bubble of their communications locked to their location for improved accessibility, additionally on-pitch sounds can be isolated and panned to the correct position relative to the viewer to bring the fans closer to the action

Figure 4 – 5G Edge-XR in-stadium use case concept

Other project use case

XR technologies have potential applications in almost all industries including engineering and architecture, live performance, medical imaging, education and retail. Use cases addressing some of these additional deployments will be explored within the 5G Edge-XR project.

Getting content to the edge

As with any cloud-based or remote live production a key element is the ability to get the captured content up to the cloud in an efficient manner with all content clocked and synced correctly. The 5G Edge-XR project will be using a 5G contribution network with the raw microphone signals sent to the edge directly. We will be using DANTE capture devices on premises with DANTE Domain Manager managing the network and a DANTE virtual Sound

card running on the server to pull the audio feeds in to the edge compute where the audio processing will be done as described in subsequent sections of this paper.

OBJECT-BASED AUDIO

One of the biggest challenges for the 5G Edge-XR project and other similar technologies is how to capture the scene in such a way that as much content and information about the content is preserved as possible while retaining well established broadcast production workflows. In addition to the audio content being captured, it is imperative that substantial metadata describing the content is also captured or extracted from the scene to enable correct, bespoke rendering at the user end. Consequently, the adoption of an *object-based audio* paradigm to be able to facilitate requisite personalised, immersive audio. An object-based paradigm is fundamentally different to traditional channel-based approaches as instead of mixing audio content for a target system, the audio components and descriptive metadata are retained as discrete assets right through the production chain for bespoke rendering at the user end.

Channel-based systems mix the audio content for a specific audio output format (stereo, 5.1 etc) using the available audio sources at the capture end. Once this content has been mixed it is not possible to manipulate it or personalise it at a later stage as all of the components have been ‘baked in’ to the audio content stream. An object-based paradigm differs in that the individual elements are composed at the capture end but are kept separate right through the broadcast/signal chain so at any point, sources can be added, moved or manipulated right up until the point of rendering. This has several advantages, both in terms of a format agnostic audio rendering (i.e., it can be played back over any output system) and also in that it enables sounds to be added, removed, altered or panned around etc., for personalised rendering. It is hence apparent that to enable adaptive and personalised rendering for the 5G Edge-XR project an object-based paradigm is essential. Fundamental to an object-based system are the ‘audio objects’ that we will define here as the discrete sounds in the scene, usually rendered at a specific point in space, and the ambient beds which make up the immersive background sound on to which the audio objects are overlaid.

Audio objects

In any audio scene there will be many discrete sound sources that can be described as coming from a specific location and that will have defined audio characteristics. Such ‘point’ type sources can be defined as *audio objects*. The term ‘audio object’ is also used to describe any audio asset that can be added or removed from a sound scene/composition. This might include the sound of a racket strike in tennis, the microphone feed of the umpire, the commentator and the PA for instance. Audio objects are typically discrete audio sources with accompanying metadata describing their location, type, duration and other attributes/signal statistics. Describing the audio scene in this way and keeping the assets separate right up until the end user, means that the end-user has full control of their audio mix and are able to add/remove/reposition and interact with any aspect of the mix (within the bounds allowed by the broadcaster).

Ambient ‘beds’

Aside from the discrete sound source/objects that are in a scene there is also the ambient/background sound that needs to be captured in a spatial sense such that a faithful reproduction of the scene can be rendered for the end user. It may be that there are several ambient beds that are used to describe and audio scene and confusingly these can even be

referred as ‘objects’ that can be interchanged/mixed together if necessary. For example, in a sports context there may be a base crowd sound, a home crowd sound and an away crowd sound and a user could have a choice over the feeds they add in to their final mix.

There are different approaches to capturing the ambient bed on to which the audio objects can be superimposed. It is possible to have a bed consisting of a 5.1 capture for example, or more complex descriptions of the overall sound scene can be included, utilising technologies such as ambisonics [Gerzon (1)] which is a means of describing an audio scene in terms of its spherical harmonic components such that the scene can be efficiently captured, transmitted and then decoded for reproduction. The more recent introduction of higher order ambisonics (HOA) brings increased spatial resolution to the format [Bertet (2)] and is the preferred audio format for several popular 360° video platforms. Interaction within an ambisonics scene is generally limited to rotation and, to some extent, zoom. For this project we have adopted a second order ambisonics capture for the bed.

Metadata

The audio system for the 5G Edge-XR project requires that the audio objects and sound field/bed descriptions are not only captured with a high level of accuracy but that the rendering be done on-the-fly, matching the viewer’s perspective of the scene. An important part of the system therefore is the metadata accompanying the audio data which will describe the scene and will be fed right through to the rendering engine to enable the manipulation of the audio output to match what the viewer is seeing in real-time. Perhaps the most important aspect of this metadata for immersive applications is the location of each audio object which has to be determined and in many cases is non-trivial as sounds cannot always be tracked and there are often high levels of background noise at e.g., live sports events.

S-ADM Stream

Once the requisite metadata parameters have be extracted from the scene, a stream needs to be composed that is linked with the audio components so the render can use the information to compose the final sound stage at the user end. The chosen metadata format for this project is the Audio Definition Model (ADM) (3). ADM is an ITU-R BS.2076-2 metadata specification that can be used to describe object-based audio, scene-based audio and channel-based audio. ADM allows the description of many audio and scene aspects such as the position, time and type of audio source as well as the input and output audio formats. It can even be included in BWF WAVE files or used as a streaming format in production environments so fits the purpose of the project very well. The ADM model is divided into two sections, the **content** part, and the **format** part. The content part describes what is contained in the audio, so will describe things like the language of any dialogue, the loudness, location and so on. The format part describes the technical nature of the audio so it can be decoded or rendered correctly. Some of the format elements may be defined before having any audio signals, whereas the content parts can usually only be completed after the signals have been generated. The content elements can edited/ authored at any point in the production chain.

In order to facilitate the real-time aspects, we will use serialised ADM (S-ADM) (4) as it is extremely versatile and well suited for object-based audio content. S-ADM sends consecutive ADM data frames, updating any changes in the audio scene from one frame to the next. Reading these ADM frames into the system in real-time enables the dynamic manipulation of the audio sources to match any type of control system, whether that be

automatic from the user's headtracking or manually via an audio scene authoring tool. S-ADM allows us to adapt the audio scene and rendering in real-time for a personalised, immersive rendering for all users and so has been chosen as the metadata format for this project.

THE AUDIO SYSTEM

In the past, real-time audio processing in the cloud has been difficult due to bandwidth limitations. These limitations could be somewhat reduced by compressing the audio; however, this is not possible if the processing is part of the main signal path due to the loss in audio quality that comes with many compression techniques. A method of bypassing these limitations, often used in mixing, is to keep audio processing local but to have an application that interacts and controls the processing through the cloud. A 5G network greatly improves the amount of bandwidth available to such a point that transmitting multi-channel uncompressed audio into the cloud is possible.

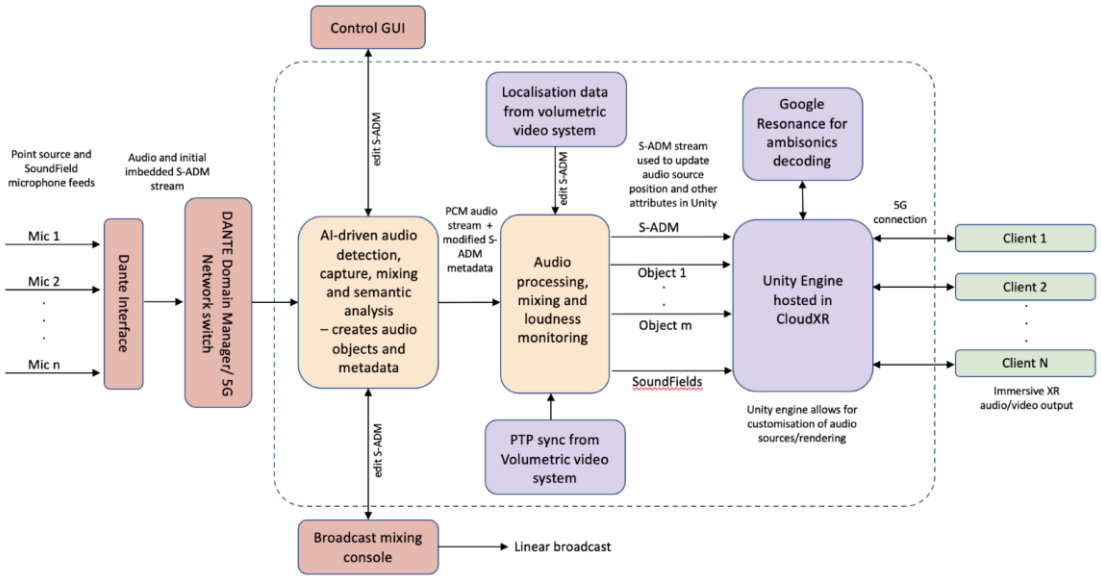

Figure 5 – 5G Edge-XR audio system diagram

The 5G Edge-XR audio system architecture is shown in Figure 5. The raw microphones are ingested at the event and are uploaded to the cloud/edge over the DANTE contribution network. Through the Dante virtual soundcard, these microphone feeds are input into the audio event extractor and semantic analysis engine which compiles an audio scene consisting of the objects, beds and associated metadata. The metadata stream and audio mix can be manipulated with the remote GUI and/or the broadcast mixing console. In addition, there we are running semantic analysis on some of the audio streams/objects to generate more metadata that can be used later on in the signal chain to drive the end-user experience. The metadata in the system can be edited by any of the processing/input blocks including the volumetric video capture which can update and add key positional data. The audio content is then synced with the video stream and the objects, sound field beds and S-ADM stream are fed into the Unity game engine for scene compilation. The Unity scene sits in the cloud and when clients connect to the server, they are delivered a bespoke audio and video representation to match their perspective and preferences.

AUTOMATIC AUDIO PRODUCTION IN THE CLOUD USING AI

Traditionally, the mixing would be done on the premises and an output mix will be created in an OB truck for a given output format/target system. For the 5G Edge-XR project, the AI-driven mixing engine has been ported from the premises to the cloud. One of the benefits of running the audio analysis, processing and mixing in the cloud is that the processor power can be greatly increased with GPU acceleration. This increases what can be done with audio analysis and can enable more complex processing tasks like real-time audio object extraction, localisation and semantic analysis on the incoming streams. Furthermore, the cloud-based audio mix engine enables automated content composition for different audiences.

Extracting Sound Sources

Capturing a scene in an object-based way presents some challenges and requires some alterations in the way that content is created/mixed to ensure that the individual sounds at an event e.g., the sound of a punch in boxing, the racket strike in tennis or the sound of a ball being kicked etc. are detected and extracted as separate sources. The way that this is done at the capture end depends somewhat upon the context and the audio extraction techniques will vary accordingly. Fundamentally however we employ machine learning techniques that analyse various representations of the audio signal to learn complex patterns that allow them to detect when specific audio events occur. The audio signal representations used are also context dependent since audio events have different characteristics which are presented through a range of features. The selection of these features is based upon which ones will show the most differences between the event and any other audio. This will help the AI pick up on patterns which will ultimately improve its

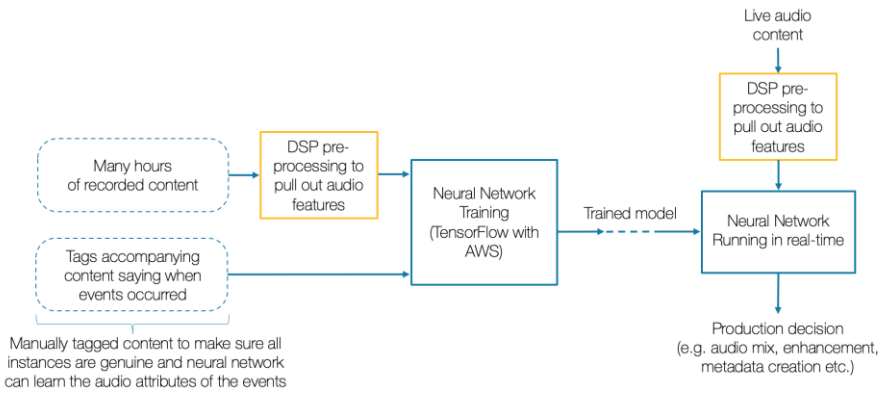

Figure 6 – AI system overview diagram

detection.

An overview of our AI approach to audio capture and extraction is shown in Figure 6. Audio templates are derived based on perceptual models of the salient sound sources in the current context and a neural network trained on content from the same context enables very accurate detection and classification of the audio events of interest in real-time. If several microphones capture the same audio event, the signals are triangulated using an efficient optimization algorithm, creating positional metadata to help automatically facilitate spatial and immersive mixes.

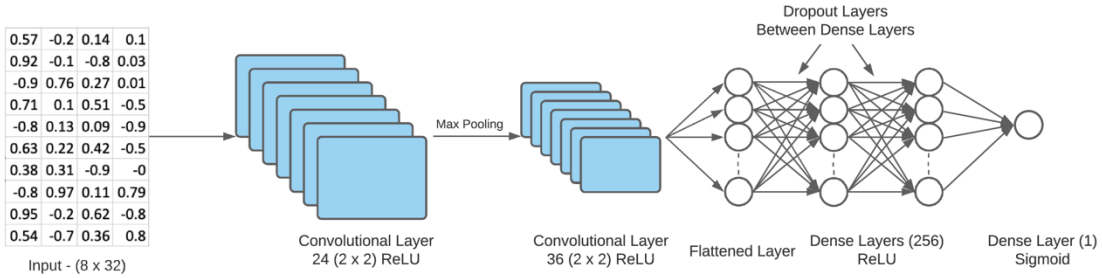

Figure 7 – Example convolutional audio network diagram

Figure 7 shows an example convolutional neural network model used for detecting highly transient audio events, such as punches in a boxing match. Here, a Mel spectrogram is used as the representation of the audio data. The neural network is fed eight consecutive spectrograms that were created from FFT windows of 1024 samples with a 256-sample overlap. This overlap is necessary due to the transient nature of the event. The data forms a two-dimensional grid which is normalised and then passed to the input layer of the neural net. The convolutional layers move through the data analysing each 2 x 2 window. These windows have filters applied to each value to attempt to recognise patterns in them. In this case, the size of the window and the nature of the data means that the first filter will be looking for patterns in a space of two Mel filters in two Mel spectrograms.

In between the two convolutional layers a max-pooling function reduces the size of the data by moving through it in windows of 2 x 2 and reducing that window to one value equal to the maximum value in that window. The data is then flattened to a single dimension and passed through a sequence of Dense layers. Each of these layers contains a number of nodes which are connected to every other node of the previous and next layer. Each of these connections has a weight which determines how important the first node is to the value of the second node. Data moves through these layers being multiplied by the weights to create the value of the single node in the last layer of the network. It is this value that represents how confident the network is that an event occurred. Two of these layers are dropout layers that will randomly zero nodes. This is useful during training to prevent overfitting, but is not applied during the normal running of the network.

With transient events, a second non-neural network detection approach can be applied in tandem to the neural net in the form of an onset detector. This can be used as a filter to limit the number of false positives produced by the network. This is useful in the case of real-time audio event detection since even with a low percentage of false positives, the network is being run so often (187 times a second at 48kHz with a 256-sample hop size) that many false positives can be produced in a short space of time. The onset filter is set up so that audio is only passed to the neural net if there was a significant transient detected. This can be extended to a multi-microphone set up by using the onset filter and neural network in one microphone to attempt to detect an event. If this triggers that an event was detected the onset detectors of the other mics can be bypassed for a certain amount of time. It is likely that the microphone which first picked up the transient is the closest microphone to the event and thus by bypassing the onset detectors of the other mics we can still pick up the event in multiple microphones without being limited to just those which are close enough to detect a transient.

The AI is written in C++ using TensorFlow so is cross-platform compatible but has been deployed in this case on the Linux edge compute server to make use of the GPU acceleration from the project's server. This GPU acceleration is vital for the use cases stated

above, since AI processing is computationally very expensive and needs to run on multiple audio channels simultaneously. GPU acceleration also allows for us to push our AI models further than would be possible with a CPU approach. Required computational power for the AI to perform increases with the amount of input data fed into it. Therefore, the GPU acceleration allows us to pass more audio signal representations to the AI to give it a better understanding of what is happening in the scene.

From the content analysis and extraction, we create content flags that can be used to create mix decisions (within a traditional framework) or to trigger pre-recorded content to enhance broadcast audio. Additionally, these flags can be used to localise the sound sources as described below. The process flow is shown in Figure 8. The individual 'audio objects' are packaged up with localisation and other metadata as a scene description which can be manipulated later in the production chain.

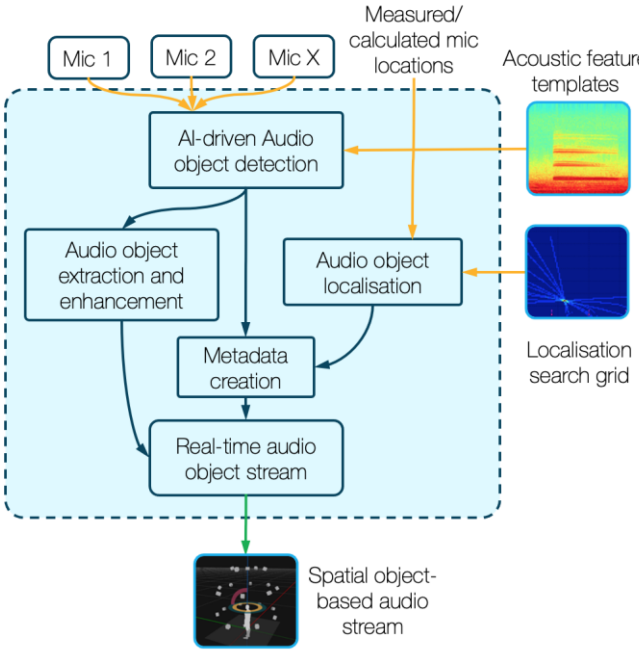

Figure 8 – process flow for the audio extraction system

Localising sound sources

To be able to create audio objects for immersive contexts it is important that individual sources are accurately localised in the scene and corresponding metadata authored. This means that as the viewer navigates their visual perspective on the content, it is possible to correctly move sound sources around so that they match the location of the visual sources.

To facilitate this, we have employed a triangulation routine. The triangulation methodology varies dependent upon the capture setup but typically is done using the time difference of arrival (TDOA) between signals of different microphone pairs picking up the same source. Knowing the location of these microphones allows the source to be positioned on a hyperbolic path between the two microphones. If the same source is picked up in additional microphones, there will be additional hyperbolae and the overlap between the resulting curves enables the accurate positioning/triangulation of the source. Determining the TDOAs can be a challenge using traditional algorithmic methods such as cross-correlation due to the high background noise of live events so we use our AI to extract the time stamp of source detection in each microphone to determine the TDOAs for a more robust approach.

Determining the overlap of the resulting hyperbolic paths from the triangulation can also be cumbersome and computationally inefficient so for a real-time approach we have adopted a least squares search grid approach where we look for the most likely source position based on a set of microphone TDOAs. For each square in the search grid, the amount of time it would take for a sound originating in this square to reach each microphone is recorded. These are then adapted to be relative to the closest microphone to each square which gives the relative TDOAs for that square. When a set of microphones all detect the same audio

event, the sample numbers at which each microphone detected the sound is recorded. These are then also adapted to be relative to the microphone which detected the event at the earliest point giving the relative TDOAs for this event. The relative TDOAs for this event can then be compared to the relative TDOAs for each square. The square in which there is the least difference between the two sets of values is the square in which the audio event occurred. The resolution of the search grid can be tailored to the context and available compute power.

A practical constraint of performing the localisation in a live broadcasting environment is the requirement of the on-site broadcast team to measure the microphone locations. This can be problematic without specific equipment which limits the practical application of this methodology. Using the power of the cloud for processing however we are able to determine the location of not only the sound sources but also the position of the microphones as well, using an iterative optimisation routine which takes multiple source detections.

Metadata

With an object-based audio approach, the content composition can be altered at any point in the signal chain by additional processing stages and systems. For example, in the 5G Edge-XR project, the volumetric video processing from Condense Reality is able to provide additional positional data of the main sources or a region of interest in the space which is used to correlate with audio source positions of the various objects in the scene. The mechanism by which this happens is the S-ADM metadata stream which can be edited within the signal chain either on the server or at the end-user device for a personalised experience.

AUDIO CONTENT RENDERING

Automatic mixing

When multiple, personalised mixes are required at the render end, it is imperative that there is an automated mixing stage within the system architecture. Automated mixing can take in the audio content, external location/tracking data and any individual viewing data to be able to compile an immersive and personalised mix.

With the ability to add and remove the various objects/sources within the scene, the relative balance of the various components needs to be automatically managed and the output loudness monitored and scaled correctly. At the output stage of the signal chain, signal loudness is measured and manipulated in accordance with EBU R128 and ITU BS.1770-3. The parameters of this processing are editable so that different loudness standards for different platforms can be achieved.

Rendering in a game engine for a personalised experience

The audio content needs to be streamed in to the Unity gaming engine as a scene composed of various audio objects and ambisonic sound field descriptions in real-time so that the output content can be composed and rendered in real-time in conjunction with the Condense Reality volumetric video components.

While Unity supports real-time audio input through its Microphone feature (typically used to allow users to input their voice into the game for voice commands and player to player communication), its capabilities are limited in two main aspects. Firstly, this feature only supports a single input audio channel and secondly this audio channel must come from a system audio input device leaving no way to stream audio from an external application or connection. These limitations are acceptable for player combination, however they become

problematic when all audio in a scene needs to be streamed into Unity in a multichannel format as is the case for this project. Unity's audio objects are usually connected to an abstract audio provider such as a microphone or an audio file. This makes implementing them fast and simple. The technique used to bypass these input limitations involves interfacing with Unity's audio engine at the lowest level. One of Unity's audio objects are created without a standard audio provider. Instead, a bespoke provider is implemented that receives audio data from an external stream and passes it to the audio object through a low-level call-back function. This call-back is usually used for applying effects to or analysing the real time audio passing between the object and provider. Since this call-back allows for the raw audio data to be read and manipulated, we are able to use it to pass our data into the audio engine.

Audio output options are also limited by Unity. Unity only supports up to 8 discrete output channels. Furthermore, channel positions in their configuration are not changeable. This means that if an 8-channel output format is used, the output mix is rendered based on the standard channel positions in a 7.1 audio format. This can be somewhat improved by using a third-party spatialisation plugin, however this makes only limited additions, the most useful of which for our purpose is a binaural output format. Ongoing work in this area is the creation of a bespoke audio rendering plugin for 5G Edge-XR.

OUTLOOK

The 5G Edge-XR project is an excellent test-bed for what is achievable using fast 5G networks that unlock the processing power GPUs in the cloud and opens the door for more applications of AI in the cloud. New, more advanced audio processing and mixing methodologies can be applied, facilitating not only future focussed immersive applications as described in this paper but also enhanced methods of producing higher quality content at a lower cost for current work-flows where there is a constant drive more content at lower budgets.

CONCLUSIONS

This paper has described work underway as part of the UK DCMS funded 5G Edge-XR project which aims to bring immersive XR experiences of live content to viewers on consumer devices by harnessing 5G networks and cloud/edge compute power. The paper has focussed on the audio engine, describing how an object-based audio system has been facilitated by employing real-time AI in the cloud for the real-time creation and rendering of audio objects. The object-based system allows for personalisation of the immersive content to match the visual rendering of the content performed in the cloud.

REFERENCES

1. Gerzon, M.A., 1985, Ambisonics in multichannel broadcasting and video. J. Audio Eng. Soc., 33(11): p. 859-871.

2. Bertet, S., et al., 2013, Investigation on localisation accuracy for first & higher order ambisonics reproduced sound sources. Acta Acustica united w. Acustica, 99(4): p.642-657

3. Recommendation ITU-R BS.2076-2 (10/2019), Audio Definition Model

4. Recommendation ITU-R BS 2125-0 (01-2019), A serial representation of the Audio Definition Model