"Wrapping" X3DOM around Web Audio API

Andreas Stamoulias[1], Eftychia Lakka[1,2], Athanasios G. Malamos[1]

[1]Department of Informatics Engineering Technological Educational Institute of Crete Heraklion, Greece, GR 71004,
[2]Faculty of Computing, Engineering and Science, University of South Wales, Treforest, UK

***Abstract* — Spatial sound has a conceptual role in the Web3D environments, due to highly realism scenes that can provide. Lately the efforts are concentrated on the extension of the X3D/ X3DOM through spatial sound attributes. This paper presents a novel method for the introduction of spatial sound components in the X3DOM framework, based on X3D specification and Web Audio API. The proposed method incorporates the introduction of enhanced sound nodes for X3DOM which are derived by the implementation of the X3D standard components, enriched with accessional features of Web Audio API. Moreover, several examples-scenarios developed for the evaluation of our approach. The implemented examples established the achievability of new registered nodes in X3DOM, for spatial sound characteristics in Web3D virtual worlds.**

***Keywords* — Spatial sound, X3D, X3DOM, Web Audio API, Web3D, Real-time, Realistic 3D, 3D Audio.**

I. Introduction

THE three Dimensional (3D) visualization plays a vital role in Computer Graphics. In the last few years, there is a growing interest in the technologies which have been designed to present Web 3D scenes. This effort has started since 1995, when a text based meta-language, Virtual Reality Modeling Language (VRML), was design with the influence of HTML [1]. Following this, the Web Graphics Library (WebGL) introduced in 2011 by the Kronos Group [2] and attracted considerable attention by Eicke et al. [3], due to the effective suggested method for Web 3D representation, without any particular hardware and software requirements. The so-called extensible 3D (X3D), is one of the latest architectures that has come up as an extension of VRML [4]. The use of the Extensible Markup Language (XML) in X3D had as a consequence increased flexibility and maturity in comparison to VRML [5]. However, the drawback of plugins installation was remaining. This problem was solved by the proposed X3DOM framework, as reported by Behr et al. [6]. For this reason, the last five years almost 70% of the X3D specification has been realized into the X3DOM framework, but the spatial sound attributes are remaining an open issue and not implemented yet, despite the fact that it is a critical issue for the 3D scene.

Moreover, the literature on 3D sound shows that it has attracted much attention from research teams. The group of Garbe [7] and Ding et al. [8], through the enrichment of their X3D audio nodes, succeed in better Web 3D representation. In the meantime, the most interesting approach to this issue has been proposed by the Web Audio API. This describes a high-level JavaScript API [9] that provides a group of new and reinforced audio properties in Web applications [10]. Wyse et al [11] and Pettersson et al. [12] have developed a novel implementation by using Web Audio API in order to achieve their goals for spatial sound. Most of the 3D applications support spatial sound. For example, the position and the intensity of a sound source could be managed by the user and accordingly, could be changed depending on the movement of the user. All the above are important for realistic scene. Consequently, spatial sound is an equally decisive issue for both, the 3D and the Web 3D applications.

However, while X3D ISO [13] includes the specification of the spatial sound components, it has not been implemented in X3DOM yet. Moreover, there is no efficient implementation of spatial sound in X3D in general. Besides that, few publications can be available in the literature that address the issue of the X3D sound extension [7] [14] but they work only with a very limited number of sophisticated sound properties and methods. Therefore, we demonstrate an innovative method to improve the spatial auralization in X3DOM, through the X3D specification and the structure of Web Audio API. In detail, we make an effort to combine the benefits of X3DOM, such as the open technology and the lack of plugins use, with the flexible HTML5 and the efficient Web Audio API. Accordingly, our work intends to include 3D sound in a declarative web 3D scene, with the ability to run natively. Likewise, the paper does not only focus on the implementation but also on the evaluation through a variety of scenarios.

The rest of the paper is organized in 4 sections: Section II describes the technologies that are used, Section III analyses the implementation. The experimental results of the evaluation are presented in Section IV and Section V concludes the paper.

II. Background

This section presents background information for the X3D/X3DOM and for the 3D Audio (Web 3D Audio - Web Audio API - X3D/X3DOM Audio) as well.

A. Web 3D - X3D/X3DOM

The X3D has been proposed by Web3D consortium, a nonprofit organization which is a combination of business companies, government agencies, academic institutions, and individual professionals. It has been formally approved by the International Standards Organization (ISO) as ISO/IEC 19775, since 2004 [15]. X3D is modern descendant of VRML, it not only includes the capabilities of VRML, but also predominates in several aspects. Particularly, it uses XML in order to express the geometry and integrates plainly with other applications [16]. Moreover, it is organized into logical groupings of functionality (components) [17], which could be expanded and enriched with new ones. Likewise, X3D applications are reliable and predictable; similarly X3D binary format provides encryption and compression [18]. For the aforementioned reasons, the X3D has become more attractive and effective than the VRML.

On one hand, X3D has been used already in HTML5 in order to declare the web 3D worlds. On the other hand, it does not address the problem of connection between the web-browser frontends and the X3D backends. X3DOM comes to overcome this issue [19].

DOI: 10.9781/ijimai.2015.355

TABLE I
OUR IMPLEMENTATION FOR THE REGISTRATION OF X3D SOUND NODES (ATTRIBUTES) INTO X3DOM.

Node	X3D	X3DOM	OUR IMPLEMENTATION	
	Attributes		Web Audio API	Enhanced X3D Nodes
X3DSoundSourceNode: AudioClip	SFString description			AudioSource
	SFBool loop	✔	AudioBufferSourceNode	AudioSource
	SFNode metadata	✔		AudioSource
	SFTime pauseTime		AudioBuffer AudioContext	AudioSource
	SFFloat pitch		AudioBufferSourceNode	AudioSource
	SFTime resumeTime		AudioBuffer AudioContext	AudioSource
	SFTime startTime		AudioBufferSourceNode	AudioSource
	SFTime stopTime		AudioBufferSourceNode	AudioSource
	MFString url	✔		AudioSource
	SFTime duration_changed			
	SFTime elapsedTime			
	SFBool isActive			
	SFBool isPaused			
	✘	SFBool enabled		
X3DSoundNode: Sound	SFVec3f location		PannerNode	PannerNode
	SFVec3f direction		PannerNode	PannerNode
	SFFloat maxBack		PannerNode	PannerNode
	SFFloat maxFront		PannerNode	PannerNode
	SFFloat minBack		PannerNode	PannerNode
	SFFloat minFront		PannerNode	PannerNode
	SFNode metadata	✔		
	SFFloat intensity		GainNode	
	SFFloat priority		DynamicsCompressorNode	
	SFNode source	✔		AudioSource
	SFBool spatialize		TRUE	TRUE

It is a framework, which incorporates a set of technologies such as X3D, WebGL, HTML, CSS and JavaScript. The main objective is the development of an X3D scene by the use of the HTML DOM and the 3D content management through the DOM elements and without extra plugins [20]. Equally important is the capability of X3DOM, as a JavaScript framework, to allow three interactions; the first is an event in the scene that causes a behavior in the scene. The second is an event in the scene that causes a behavior in the HTML5. The last is an event in HTML5 that causes a behavior in scene [21].

The present research efforts are focusing on the development of Web 3D interactive virtual worlds, based on the X3DOM. However, some studies draw attention to the X3DOM node extension. Stamoulias et al. [22] suggest the implementation of the rigid body physics component in the X3DOM environment. A concept for the improvement of the shadow representation for X3DOM is provided by the Kuijper's group [3]. Additionally, Kapetanakis et. al [23] present an approach to extend the adaptation methods of X3DOM by adding a mechanism to perform dynamic adaptation and achieve HD video delivery in 3D Virtual Reality (VR) worlds. All the above have the same aim; to increase the level of realism in the Web 3D scene, through the enhancement of the X3DOM structure. The aim of the current work, in accordance to the previous examples, is to open up the field of the spatial sound in a web 3D scene, beyond the limits of the X3D-X3DOM technology.

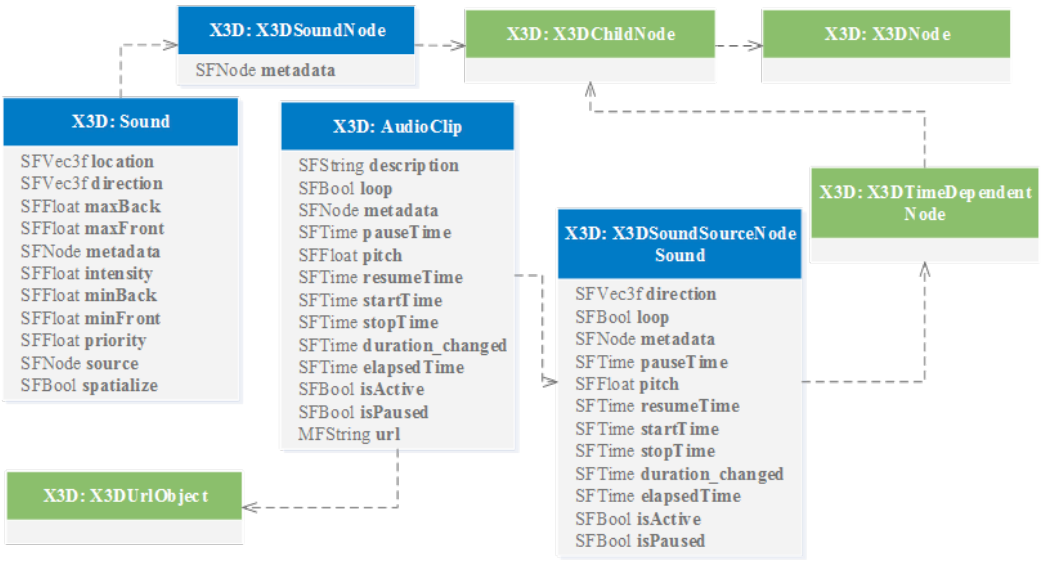

Fig. 1. Inheritance diagram of Audio nodes in X3D.

B. 3D Audio

The sound comprises a fundamental part of real life applications and can be produced from a plenty of audio sources. Every sound has a specific direction and can be easily identified, due to its distinctive characterizations and the familiarity that we have developed with sounds [24]. Additionally, a characteristic of the human hearing system is the ability to perceive the 3D sound. General, the meaning of 3D sound lies in the way that the listener receives the incoming sounds from all the directions. The concept of 3D sound is the same when it is simulated by a computer [25]. In other words, a listener can recognize meaningful spatial cues from a sound source, for example the direction, the distance and the spaciousness [26].

According to the above, the 3D sound is important to be included in virtual worlds, in order to improve the realism and the sense of the immersivity in the scene. In this case, the user can understand any sound source which is included in the virtual 3D scene through the combination of visual and auditory sense. Adding natural spatial sound to interactive 3D immersive applications, more sophisticated information is conveyed, in conjunction with using other modalities (e.g., vision) [27].

1) Web 3D Audio

Many approaches for Web 3D applications aimed at realistic visualization of the scene. The most often overlooked point is the sound, even though if it can offer further details to a 3D graphic world. Specifically, a high immersion level is accomplished and the natural interaction is increased, on the ground of that the graphic scene simulates the real world in the best possible way [23]. Equally important is the spatial audio and the rendering of spatial attributes of each auditory objects, in a Web 3D world. These attributes involve perceived directions, distances and spatial extends of the auditory objects; furthermore these attributes should be conceived in the same way as they are recognized in the real world [28].

For all the above reasons, considerable attention has been paid to introduce sound in web application. The first attempt took place via the

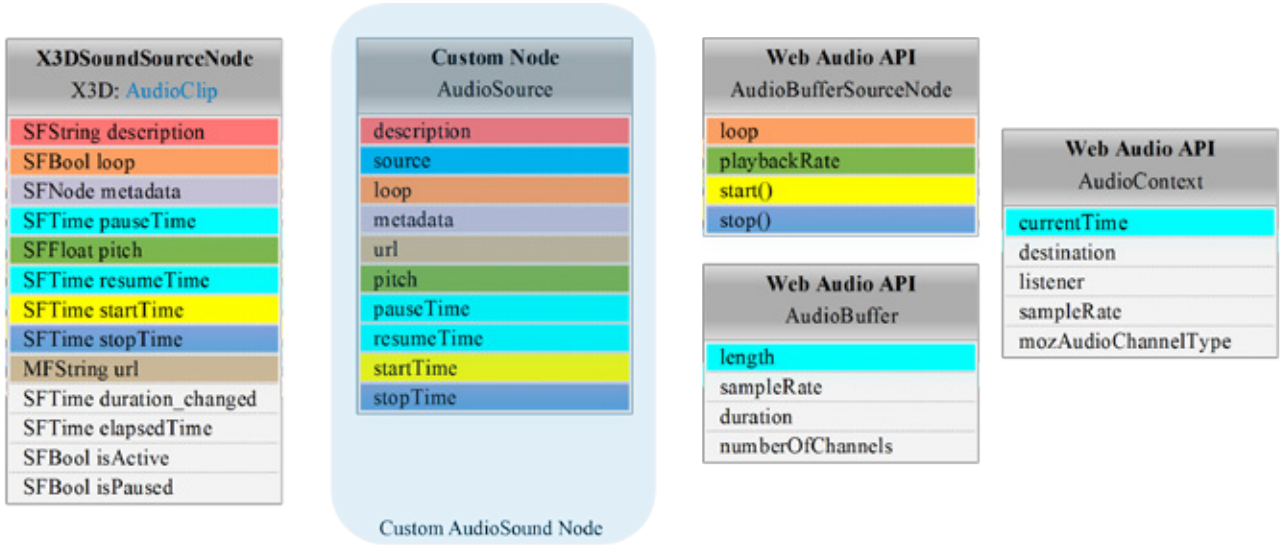

Fig. 2a. This component diagram presents our proposal for spatial sound in X3DOM. The attributes of AudioClip (X3D node) have been registered in X3DOM with the use of new enhanced X3D nodes and Web Audio API nodes. Each color expresses which fields combine in order to create the respective components in X3DOM.

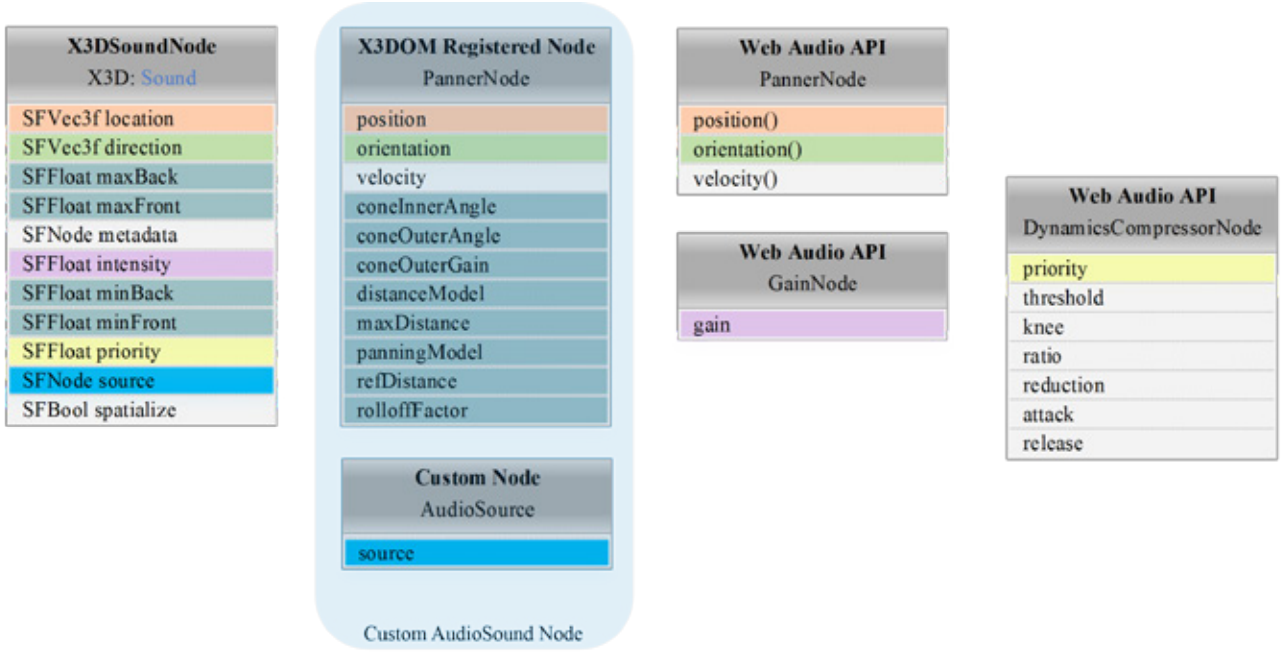

Fig. 2b. This component diagram presents our proposal for spatial sound in X3DOM. The attributes of Sound (X3D node) have been registered in X3DOM with the use of new enhanced X3D nodes and Web Audio API nodes. Each color expresses which fields combine in order to create the respective components in X3DOM.

<bgsound> tag, in which only background music could be contained in a web page and was available for specific browsers. After that, flash was the first cross-browser way of audio on the Web, but plugins were required. The focus of recent research was concentrated on the element <audio> in HTML5, which could avoid the plugins, but was not designed for sophisticated and ambitious developments [29], [30]. Specifically, the element <audio> is inferior to apply filters to the sound signal and access the raw PCM data. Moreover, it does not include the concept of position and direction of sources and listeners. Lastly, it does not afford low-latency precise-timing model, which is very important for interactive applications for the need of fast auditory response to user actions [31]. Thus, it is not adequate for a 3D interactive web environment with demanding sound design.

Under these circumstances, several alternatives have been proposed, in order to establish an effective API, which attends to overcome the most of these limitations. One of the most interesting approach to this issue is Web Audio API, which has been proposed by Mozilla Foundation. Predecessor of Web Audio API was the Audio Data API. This API provided a distinct structure for writing audio callbacks in JavaScript, but it did not provide specifications for lower level, native, pre-compiled agent to be included in browsers [32]. As a result, Mozilla Foundation resulted in the adoption of the Web Audio API, which was endorsed by the other browsers.

2) Web Audio API

Web Audio API is a high-level JavaScript API, which can be used to synthesize audio in web applications. Additionally, it is characterized as an extremely powerful tool for controlling audio in the browser and tends to become a de facto standard in modern browsers [33].

Until now, 3D audio rendering engines were utilized, such as OpenAL (Open Audio Library) [34], FMOD library, in order to develop 3D auralization. Even thought that the existing solutions afford friendly interfaces, they have a number of limitations. Namely, they confined to the rendering of static audio sources, in contrast with the Web Audio API [35].

Moreover, the significance of Web Audio API can be obvious from the fact that a number of web audio libraries and APIs have been developed, in order to use/handle it. Specifically, Three.js, uses the Web Audio API to play the sound and determine the correct volume. It is a JavaScript library, which offers a simple way programming WebGL directly from JavaScript, with the view to create and animate 3D scenes [36], [37]. Moreover, the webaudiox.js library disposes a set of helpers for using the Web Audio API and the howler.js library supports automatic caching for Web Audio API [38], [39]. Furthermore, the pedalboard.js is an open-source JavaScript framework which develops audio effects through the Web Audio API [40]. Another one is the Wad library that helps to simplify manipulating audio using the Web Audio API [41]. Lastly, the Fifer Javascript library is a lightweight conductor for the Web Audio API with Flash Fallback. [42].

As mentioned above, the Web Audio API becomes attractive for the spatial audio in web environments. Except the previous reasons, it also distinguished for extra benefits. In detail, it is open source and be supported from the most browsers. Furthermore, multi-channel audio is available and is integrated with Web Real-Time Communications (WebRTC). Also, high-level sound abilities as filters, delay lines, amplifiers, spatial effects (such as panning) are offered. At the same time, audio channels can have 3D distribution according to the position, speed or direction of the viewer and the sound source [43]. Additionally, the Web Audio API is characterized by compositionality, using audio node structure, which can be linked together in order to form an audio routing graph. Besides that, it is much faster since it is written in C++, rather than it has been written in JavaScript. However, Web Audio API also provides a node (ScriptProcessorNode) that allows the web developers to manage audio using JavaScript. All the above assets make the Web Audio API to break new ground for the web sound synthesis [10].

Indeed, it was not a coincidence that Web Audio API has drawn much attention from research teams in the last two years. In particular, the literature demonstrates a variety of approaches which utilize Web Audio API, in order to accomplish the sound in web environment [44]-[47]. Furthermore, many researchers have proposed various methods of adjusting the Web Audio API in their application [48], [49], [50].

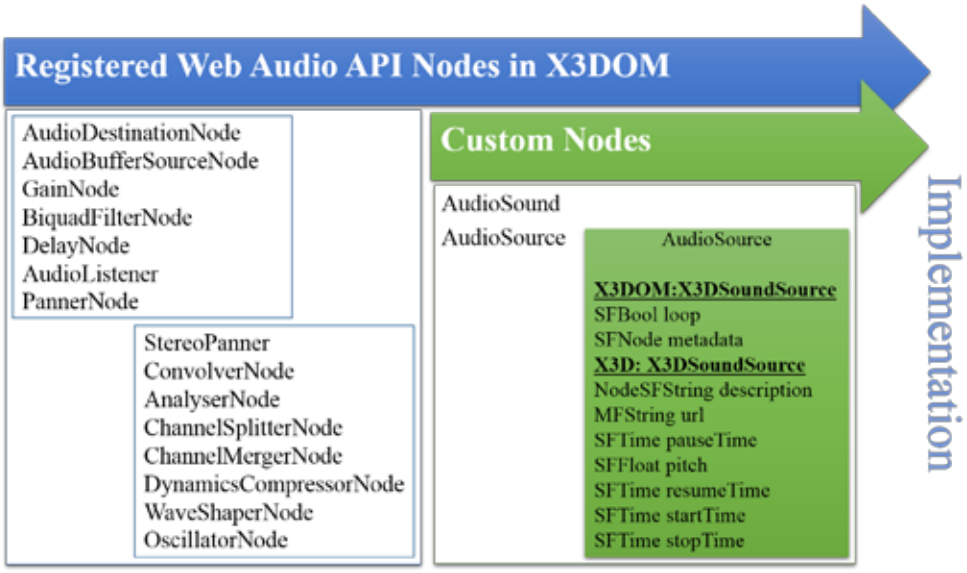

Fig. 3. Registered nodes for the paper implementation

In either case, the approach of Web Audio API is based on the concept of audio context, which presents the direction of audio stream flows, between sound nodes. In every node, the properties of the sound can be adapted and changed, depending on the application requirements.

The included nodes are possible to be sorted by type: a) Source nodes (audio buffers, live audio inputs, oscillators and JS processors), b) Modification nodes (filters, convolvers and panners), c) Analysis nodes (analyzers and JS processors), d) Destination nodes (audio outputs and offline processing buffers) [29].

3) X3D/X3DOM Audio

The current literature shows that insufficient efforts have been done in order to incorporate the spatial sound in X3D. Particularly, two types of nodes are included in the X3D, the first is about the sound description and the other one for the sound source. Specifically, the first node is the X3DSoundNode, which is an abstract node for all sound nodes. It is minimalist with only one attribute, metadata, which expresses important information for the significance, appearance and the proposed role of the model [51]. The second node is the X3DSoundSourceNode, which is the abstract node for each node that is used to emit sound and it has a number of common fields with the TimeSensor, for example the loop, the startTime, the stopTime, the pauseTime and resumeTime (Figure 1).

The third node is the Sound, which is derived from the X3DSoundNode. It is designed for the description of the X3D scene sounds. Specifically, it determines both the location and the behavior of the sound. Additionally, the geometry describes that the sound can be directed and be emitted in an elliptical pattern. Two ellipsoids constitute the pattern, which specifies the borders for level of loudness of the sound. Also, ellipsoids can be reshaped in order to provide more or less directional focus from the location of the sound [52]. Consequently, the sound node is intended to recognize the source and is related to the direction, the location, the priority and general, the spatial features of the sound source (Figure 1) [53].

The forth node is the AudioClip, which is derived from the X3DSoundSourceNode. It specifies audio data that can be referenced by Sound nodes. Basically, it loads an external audio file with a view to handle playing, stopping and starting. As regard the attributes, AudioClip has a number of fields in common with TimeSensor, because it is an X3DSoundSourceNode and implements the X3DTimeDependentNode abstract type. Basically, the fields of the sound nodes and their interrelation are presented in Figure 1 by an interpretive diagram.

Besides that the X3DOM is a descendant of X3D, the only thing that it has been implemented for sound is the registration of X3DSoundNode, X3DSoundSourceNode, AudioClip and Sound nodes, but without the most of the properties of the respective X3D nodes. Specifically, an X3DOM scene could include an audio file for playing, without any spatial characteristics. Only the attribute “SFBool enabled” has been extra added to X3DOM in comparison with X3D, which specifies whether the clip is enabled or not. The first two columns of Table 1 illustrate the attributes of sound nodes that are transferred to X3DOM from X3D.

C. Motivation

The X3DOM framework comprises a fundamental part in the 3D web development, because it provides an approach for the integration of declarative 3D in HTML5 [19]. However, the implementation of spatial sound in X3DOM is still lacking, even thought that the spatial sound should be an integral part of an immersive 3D application. In order to overcome this drawback, we present an innovative solution of the spatial sound in X3DOM framework that based on a combinational methodology. Specifically, we suggested the enrichment of X3DOM with spatial sound features, using both the X3D sound nodes and the structure of Web Audio API. We selected this combination for the reason that both the X3D and the Web Audio API has been influenced by OpenAL. Particularly, the Web Audio API has borrowed many concepts from OpenAL (position and orientation of sources and listeners, parameters associated with the source audio cones, relative velocities of sources and listeners) [29]. In the same time, the structure of OpenAL has been used for X3D sound nodes extension, by the group of Garbe [7] and several authors [54], [55], [18] have proposed the implementation of 3D virtual environment with the cooperation of X3D (for 3D scene) and OpenAL (for 3D sound). The compatibility which stems from the common ground that the API and X3D has, is the main asset that we are taking advantage in order to incorporate the spatial sound to X3DOM framework.

A measurement of the impact of our contribution is the number of web applications in X3DOM platform that may benefit from the

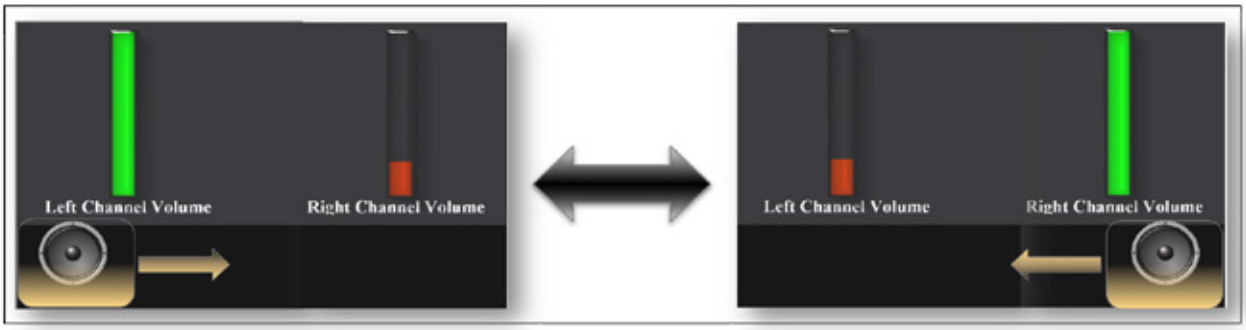

Fig. 4. The second example: spatial sound with a sound source which moves in the scene (right-left).

spatial sound components we introduce. Teleconferencing, gaming, immersive internet and entertainment are some of the areas that achieve a truly immersive listening experience, through 3D sound of X3DOM.

Although Web Audio API has been coupled with WebGL for spatial sound in the past [36], these efforts are custom implementations that use API's methods natively and are still away of becoming a platform or even more a language. Our effort was not just to customize an environment but instead to fully integrate Web Audio API into X3D language, by introducing new sound nodes and incorporate them in X3DOM API. In other words, with this approach the complex sound design and implementation is becoming transparent to the programmer and moreover the applications are independent of the sound libraries are employed.

Fig 3 presents the new nodes which have been registered in X3DOM according to our proposal (a combination of Web Audio API nodes and Custom Nodes). Table 1, Fig. 2a and Fig. 2b depict the matching of X3D and Web Audio API attributes, which have been integrated in order to create the respective components in X3DOM.

III. Implementation

This Section is devoted to the implementation of the proposed design. In the first place is the registration of Web Audio API and custom components into X3DOM framework. All these nodes are used by HTML DOM, which is essentially an X3D scene, directly from the X3DOM and no through JavaScript structure. On the other hand, JavaScript code, which is indicated as JavaScript Controller, accords all the indispensable instruction of nodes design and role. It interacts with the HTML file, for the purpose of parsing the 3D scene and being updated on any potential change in the scene. In the following sections, a more detailed description is presented of which nodes are entered on the X3DOM core in order to be recognized and be used as X3DOM element, as well as the process of JavaScript Controller.

A. Web Audio API/Enhanced X3D nodes registration into X3DOM

The attributes of AudioClip (X3D node) and Sound (X3D node) have been re-introduced in X3DOM with the use of:

- Two enhanced X3D nodes
  1. AudioSound
  2. AudioSource
- An added value set of Web Audio API nodes.

The first stage of our work is the registration of new components in X3DOM core. Besides that, it mutually exchanges information with the AudioBufferSourceNode component of Web Audio API. For visual representation of the implementation structure, the reader is referred to Fig 3.

1) Registered Web Audio API nodes in X3DOM

A set of Web Audio API nodes were registered in the X3DOM, in order to implement the proposed work. To enumerate, we utilized the AudioDestinationNode and AudioBufferSourceNode, which represent the final audio destination and the audio source consisting of in-memory audio data. Likewise, the GainNode affects the loudness of a sound and the BiquadFilterNode supports all of the commonly used second-order filter types. Furthermore, the DelayNode causes a delay between the arrival of an input data and its propagation to the output. The AudioListener represents the position of a person listening to an audio source in 3D space and each source can be passed through a PannerNode, which spatializes the input audio. Based on the relative position of the sources and the listener, the correct gain modifications can be computed by this method. Also, we registered the StereoPannerNode in order to pan an audio stream left or right. A further register node is the ConvolverNode which is effectively a very complex filter (like the BiquadFilterNode), but rather than selecting from a set of effect types, it can be configured with an arbitrary filter response. The AnalyserNode is intended to provide real-time frequency and time-domain analysis information. The ChannelSplitterNode was utilized to separate the different channels of an audio source and it often used in conjunction with its opposite the ChannelMergerNode. Lastly, we recommended to insert the DynamicsCompressorNode which was used for the compression of effects, the WaveShaperNode for the representation of a non-linear distorter and the OscillatorNode for the rendition of a periodic waveform [9].

It is noteworthy that the nodes StereoPannerNode, ConvolverNode, WaveShaperNode and OscillatorNode have been registered as extra nodes, in order to provide further sound features in the web 3D scene and succeed a higher dimension of realism in the 3D environment.

2) Enhanced X3D nodes

The enhanced X3D nodes, which were developed for the needs of the proposed work, are the AudioSound and AudioSource. The first one is the "parent" element of any other new component and the second is a combination of the development node in X3DOM, X3DSoundSourceNode and set of attributes of the corresponding X3D node. Specifically, the registration of AudioSound has been developed with the follow structure:

Registration of AudioSound Node in X3DOM

```
x3dom.registerNodeType("AudioSound"){
this.addField_SFNode("transform",x3dom.nodeTypes.Transform),
this.addField_SFNode("source", x3dom.nodeTypes.AudioSource),
this.addField_SFNode("panner", x3dom.nodeTypes.PannerNode),
this.addField_SFNode("filter",x3dom.nodeTypes.BiquadFilterNode),
```

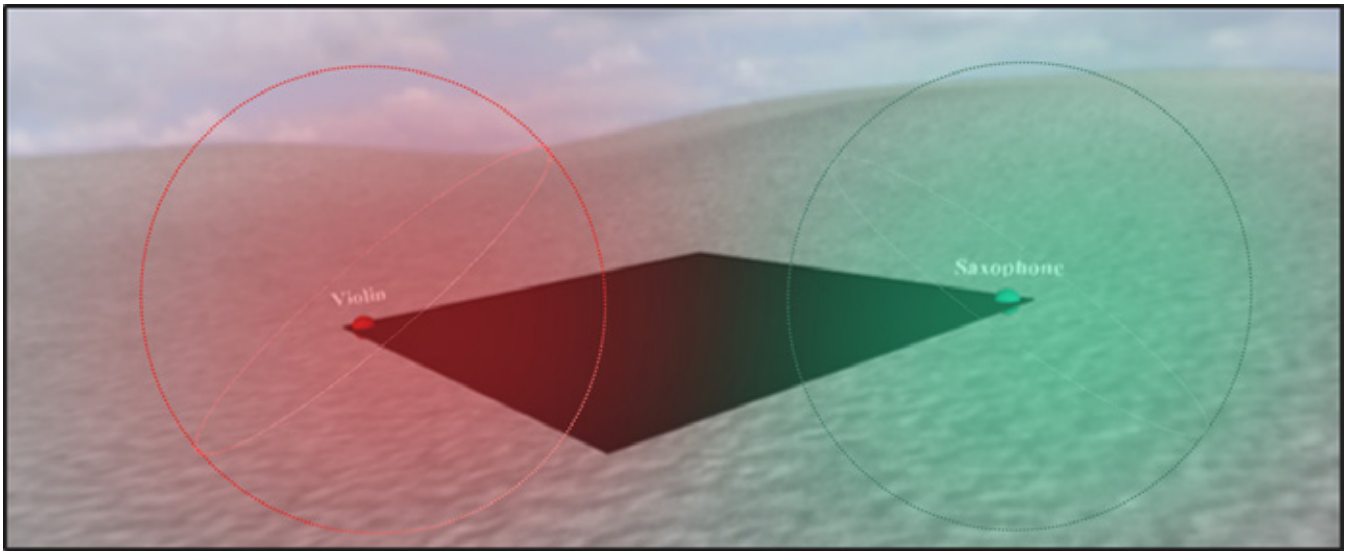

Fig. 5. Forth example: spatial sound through camera animation in an X3D scene with two sound sources.

```
    this.addField_SFNode("delay", x3dom.nodeTypes.DelayNode),
    this.addField_SFFloat("playbackRate", 1),
    this.addField_SFNode("metadata",x3dom.nodeTypes.X3DMetadataObject)
}
```

As it is mentioned above, the AudioSound includes transform field which is linked with the Transform type node, in order to connect them using the DEF/USE attribute. Additionally, the source attribute is to determine the sound source, through the AudioSource enhanced X3D node. Moreover, the panner attribute links with the PannerNode, which is reliable for the representation of audio source position and behavior. After that, the filter is intended to insert a simple low-order filter. Lastly, the attributes delay and playbackRate are used for delay and the rate of input sound data.

Furthermore, AudioSource have been registered with the follow structure:

Registration of AudioSource Node in X3DOM

```
x3dom.registerNodeType("AudioSource"){
this.addField_SFString("description", ""),
this.addField_SFBool("loop", !1),
this.addField_SFNode("metadata",
x3dom.nodeTypes.X3DMetadataObject),
this.addField_MFString("url", []),
this.addField_SFTime("pauseTime", 0),
this.addField_SFFloat("pitch", 1),
this.addField_SFTime("resumeTime", 0),
this.addField_SFTime("startTime", 0),
this.addField_SFTime("stopTime", 0) }
```

The description field specifies a textual description of the audio source, the loop attribute arranges for the repetition of sound source. The url field points to the location of the sound source of interest and the pitch is a multiplication factor applied to sound sampling and playback. Lastly, in X3D structure (pauseTime, startTime, stopTime) had the defaults value (0, 0, 0). In our implementation, the respective attributes was implemented with the default values (-1, 0, -1), for the case that pauseTime and stopTime do not need to be used.

B. JavaScript Controller

The role of JavaScript Controller is to emulate the ScriptProcessorNode node of Web Audio API, which provides the ability of web audio synthesis and process, directly in JavaScript. Equally important is the fact that one or more AudioSound components, which incorporates AudioSource, PannerNode, BiquadFilterNode, DelayNode and Transform nodes, could be included in HTML. Accordingly, the structure of JavaScript Controller starts with a repetitive detection in the HTML DOM, until find an AudioSound node (or an Inline node. In this case, the scene is an X3D file and all nodes are loaded by Inline node via the "url" of X3D file.), essentially until recognize an X3DOM audio element. Once this being achieved, a new 3D Sound Object will be created. Then, registered nodes are invoked, following certain order, AudioSourceNode, AnalyserNode, WaveShaperNode, BiquadFilterNode, ConvolverNode, GainNode, PannerNode and AudioDestinationNode. Those of the above nodes will be recognized, they will be appended in an array, in order to be available. Together with that, 3D Sound Object was initialized and was updated, through the exchange of data with HTML DOM. In addition to, ArrayBuffer and XMLHttpRequest are used, for the purpose of to load and play sound. In case that, visual and aural conditions will be changed, HTML DOM will be respectively updated.

IV. Evaluation

In this section the results of our implementation are given. In order to verify the validity of our method, we carried out several experiments – examples, in which new registered nodes were used (http://medialab.teicrete.gr/minipages/x3domAudio/index.html). Our tests have been investigated with the use of Google Chrome 41.0, Firefox Mozilla 36.0 and Opera 28.0. This choice was based on the fact that these browsers support the Web Audio API [56]. However, in our development has anticipated the fact that the browser cannot support spatial sound for any reason (for example not support Web Audio API). In this case, the implementation was adapted and produces the result without sound spatiality. Additionally, the browser Opera cannot load .mp3 files, in this case, our work controls this condition and if it is true, returns warning message. Next, a description is following, which presents these examples in detail.

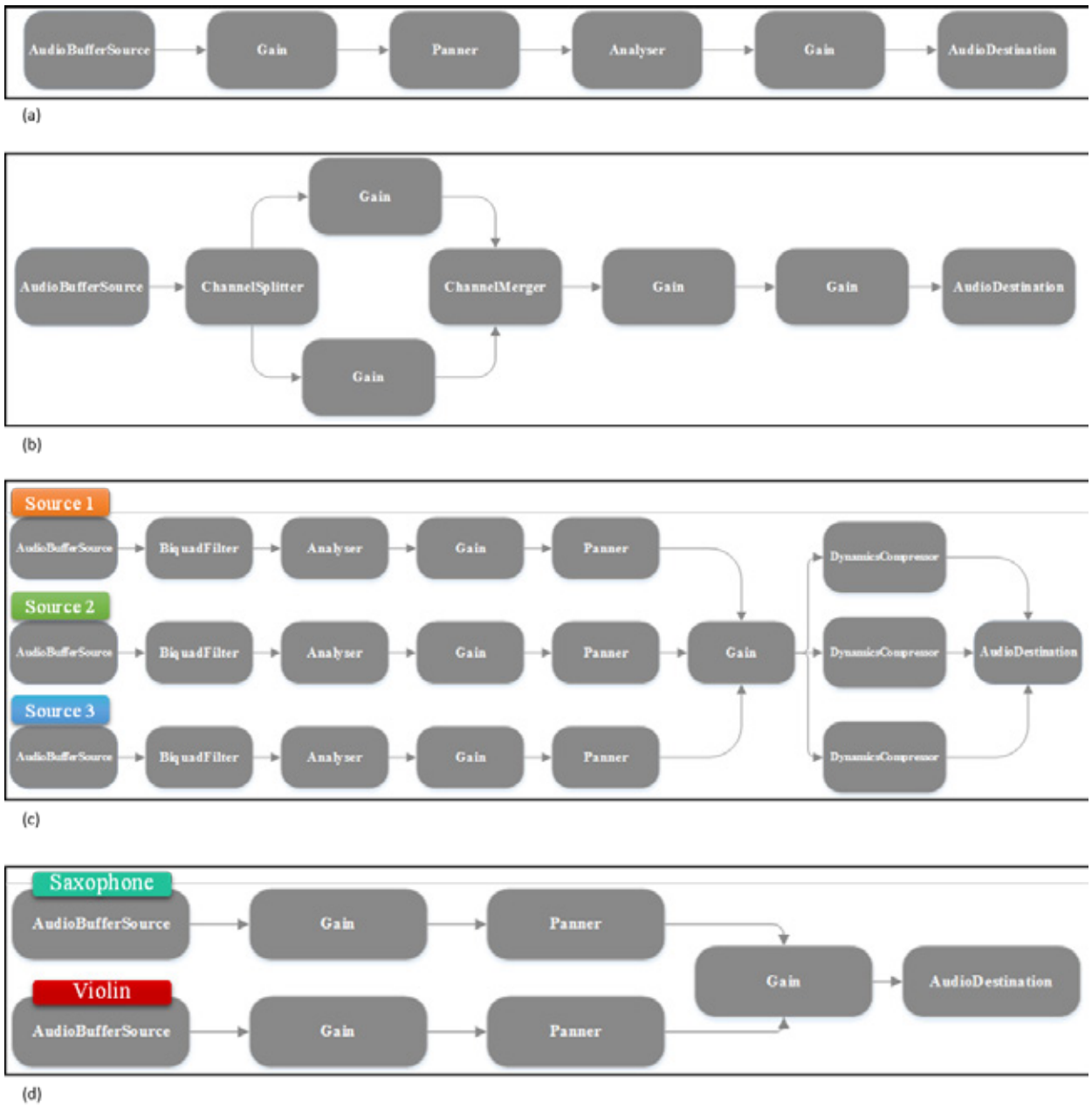

Diagram 1: (a) The Node Diagram of Single Audio Source Example (First Example). (b) The Node Diagram of Split Channels Example (Second Example). (c) The Node Diagram of Spatial Sound Effects and Filters Example (Third Example). (d) The Node Diagram of Web3D Spatial Audio Camera Animation Example (Forth Example).

A. Examples-Experiments

1) Single Audio Source

The first example1 evaluates the attenuation of one sound source, in a web 3D scene, through the X3DOM. Specifically, it includes the implementation of a single sound source, which is represented by a 3D object. The spatiality of the sound is expressed by a process, in which when the user approaching nearby to the sound source the volume is increased and accordingly when removed therefrom is reduced. In addition to this and depending on the side of the sound source that the user observes, the sound is emitted from the corresponding speaker. Apart from the 3D scene, we have also added an analyser slider. The analyser gives the possibility to receive real-time generated data, without any change from the input to output sound information. Through this process we achieved the audio visualization of the sound source.

In order to achieve what is described in the first example, a subset of new nodes (registered by us in X3DOM framework) were utilized in HTML DOM. Specifically, AudioSound is used as the parent element of any other new node. Transform, PannerNode and AudioSource constitute the children of AudioSound. The role of Transform node is to offer multiple DEF/USE copies of X3DOM in the new registered AudioSound node.

Moreover, PannerNode spatializes the input audio based on the relative position of the source and the listener. Finally, the AudioSource is responsible to load the sound file.

The Diagram 1a summarizes the background of implementation. Particularly, the graph illustrates how the nodes are connected and in

1http://www.medialab.teicrete.gr/minipages/x3domAudio/singleAudio.xhtml

which sequence they are implemented for this specific example.

2) Split Channels

The second example2 assesses the capability of the audio channels split, through our implementation (Fig 4). This X3D scene includes a simple sound source which can be moved right and left. Depending on the position of the sound source, the user can hear the produced sound from the corresponding output speaker. In this case, the new registered sound nodes of the X3DOM are called directly from the HTML file. In particular, we started with the AudioSound, which is the parent element of any other new node, as mentioned earlier. Transform, PannerNode and AudioSource which can represent different kinds of filters, comprise the children of the AudioSound. Accordingly, there is a source that can be passed through a PannerNode for the spatialization of the input audio.

Also, Diagram 1b outlines the nodes connection of the second example through the implementation.

3) Spatial Sound Effects and Filters

In the same way, a third example3 is investigated during the evaluation process, in order to introduce effects and filters in a X3DOM environment. This example includes an X3D scene with three sound sources. Each of them is visualized by a 3D object (in our case is a sphere) that depicts the sound effects. Specifically, we have added filters through of them we are able to manage the different sound effects in an impressive way. Filters can be composed of a number of attributes, frequency, detune, gain and the factor quality which also known as Q.

Basically, the filters are classified in some specific types, depending on the sound effects that produce. In detail, there is the Low-pass filter which can create more muffled sound. Another one is the High-pass filter, which is used to generate tinny sound. Equally important is the Band-pass filter, which cuts off low and high frequencies and passes through only these within a certain range. On the contrary, the Notch filter has exactly the opposite operation of the Band-pass filter. Then is the Low-shelf filter, its role is to change the amount of bass in a sound, as a result the frequencies that are lower than the current frequency get a boost, while them that are over it remain unchanged. Next, the High-shelf filter is responsible for the quantity of treble in a sound. Moreover, Peaking filter is used in order to handle the amount of midrange in a sound. Lastly, there is the All-pass filter, whose role is to introduce phased effects.

In order to implement the said to our example it has been utilized the nodes of the previous example in a similar way (the new registered sound nodes of the X3DOM are called directly from the HTML file). Consequently an AudioSound node for each sound source is used. AudioSound is comprised of Transform, PannerNode, AudioSource and BiquadFilterNode. Also, an analytical menu is provided in the web page, so the user can change the parameters of the filters which are described above.

The Diagram 1c shows in which way the registered nodes in X3DOM created and used in this scenario-example.

4) Web3D Spatial Audio Camera Animation

In the last example4 (Figure 5), we evaluate the attenuation of two different sound sources, while the camera (the user) is moving in the 3D scene. Through the immersion in the X3D scene the user could attend a rational navigation. Whenever the camera moves in the direction of an existing sound source, the sound strength of this source increases, while the sound strength of the other (the second one) decreases and vice versa. Through this process, great realism of the scene is achieved, since it emulates the spatial sound in real world.

In a like manner as in the previous examples, the new registered sound nodes of the X3DOM are called directly from the HTML file. Therefore, an AudioSound node (which contains Transform, PannerNode and AudioSource, as its children) for each sound source is used.

Furthermore, Diagram 1d represents the new registered nodes flow of this example.

Consequently, the experiments indicated that our implementation corresponded as it was expected, without any major problem. Furthermore, some tests repeated with the use of a considerable number of sound sources at the same time, with effectual results.

2 http://www.medialab.teicrete.gr/minipages/x3domAudio/splitChannels.xhtml
3 http://www.medialab.teicrete.gr/minipages/x3domAudio/filters.xhtml
4 http://www.medialab.teicrete.gr/minipages/x3domAudio/spatialAudioCamera.xhtml

V. Conclusions

Even though the X3DOM advantages in comparison with X3D, it does not require plugins, the X3DOM could not adequately handle the spatial sound. This was a major drawback for the design of realistic and interactive 3D scenes. Our proposal is a sufficient way to solve this problem and adds the spatial sound in X3DOM, ensuring that the quality of 3D scenes will be increased.

Substantially, the implementation of already existing X3D sound nodes was insufficient. For this reason, we included the structure of Web Audio API nodes and we adapted them in the X3DOM framework.

Lastly, based on the examples-experiments, it can be concluded that interactive Web3D scene can be composed with the use of new registered nodes and the results confirm our methodology.

References

[1] Raggett, D. (1995). Extending WWW to support platform independent virtual reality. Techical Report.

[2] KHRONOS GROUP. Retrieved from https://www.khronos.org/news/press/khronos-releases-final-webgl-1.0-specification [accessed May 2015].

[3] Eicke, T. N., Jung, Y., & Kuijper, A. (2015). Stable dynamic webshadows in the X3DOM framework. Expert Systems with Applications. Retrieved from http://dx.doi.org/10.1016/j.eswa.2014.11.059

[4] Daly, L., & Brutzman, D. (2007). X3D: Extensible 3D Graphics Standard. 24, pp. 130 - 135. IEEE. doi:10.1109/MSP.2007.905889

[5] McIntosh, P., Hamilton, M., & Schyndel, R. v. (2005). X3D-UML: Enabling Advanced UML Visualisation Through X3D. Association for Computing Machinery, Inc.

[6] Behr, J., Jung, Y., Keil, J., Drevensek, T., Zoellner, M., Eschler, P., & Fellner, D. (2010). A Scalable Architecture for the HTML5/ X3D Integration Model X3DOM. Web3D '10 Proceedings of the 15th International Conference on Web 3D Technology. doi:10.1145/1836049.1836077

[7] Garbe, K., & Herbst, I. (2008). Extending X3D with Perceptual Auditory Properties. Virtual Reality Conference, 2008. VR '08. IEEE. doi:10.1109/VR.2008.4480787

[8] Ding, H., Schwarz, D., Jacquemin, C., & Cahen, R. (2011). Spatial audio: graphic modeling for X3D. Web3D '11 Proceedings of the 16th International Conference on 3D Web Technology. doi:10.1145/2010425.2010427

[9] W3C. (2013). Web Audio API. Retrieved from http://www.w3.org/TR/webaudio/ [accessed May 2015].

[10] Wyse, L. (2014). Interactive Audio Web Development Workflow. ACM. doi:10.1145/2647868.2655064

[11] Wyse, L., & Subramanian, S. (2013). The viability of the web browser as a computer music platform (Vol. 37). (C. M. Journal, Ed.)

[12] Dibra, D., Otero, N., & Pettersson, O. (2014). Real-Time Interactive Visualization Aiding Pronunciation of English as a Second Language. IEEE. doi:10.1109/ICALT.2014.131

[13] Extensible 3D (X3D) ISO/IEC 19775-1:2008. Retrieved from http://www.

web3d.org/documents/specifications/19775-1/V3.2/Part01/Architecture.html [accessed May 2015]
[14] Ding, H., Schwarz, D., Jacquemin, C., Cahen, R. (2011). Spatial audio: graphic modeling for X3D. Web3D '11 Proceedings of the 16th International Conference on 3D Web Technology
[15] Parisi, T. (2014). Programming 3D Applications with HTML5 and WebGL. O'Reilly Media, Inc
[16] Spala, P., Malamos, A. G., Doulamis , A., & Mamakis, G. (2011). Extending MPEG-7 for efficient annotation of complex web 3D scenes (Vol. 59). Multimedia Tools and Applications, Springer US. doi:10.1007/s11042-011-0790-5
[17] Stewart, J. A., Dumoulin, S. J., & Noël, S. (2006). X3D Conformance Testing Factors in Creating Aviable Test Suite. Electrical and Computer Engineering, 2006. CCECE '06. doi:10.1109/CCECE.2006.277506
[18] Bouras , C., Panagopoulos , A., & Tsiatsos, T. (2005). Advances in X3D multi-user virtual environments. Multimedia, Seventh IEEE International Symposium. doi:10.1109/ISM.2005.28
[19] Behr, J., Eschler, P., Jung, Y., & Zollner, M. (2009). X3DOM: a DOM-based HTML5/X3D integration model. Web3D '09 Proceedings of the 14th International Conference on 3D Web Technology. doi:10.1145/1559764.1559784
[20] X3DOM. Retrieved from http://www.x3dom.org/ [accessed May 2015]
[21] Baglivo, A., Ponti, F. D., Luca, D. D., & Fanini, B. (2013). X3D/X3DOM, Blender Game Engine and OSG4WEB: open source visualisation for cultural heritage environments. Digital Heritage International Congress (DigitalHeritage).
[22] Stamoulias, A., Malamos, A. G., Zampoglou, M., & Brutzman, D. (2014). Enhancing X3DOM declarative 3D with rigid body physics support. Proceedings of the Nineteenth International ACM Conference on 3D Web Technologies. ACM.
[23] Mora-Lumbreras , M. A., Flores-Pulido, L., González-Contreras, B. M., & Portilla , A. (2012). Incorporating 3D Sound in Different Virtual Worlds. Web Congress (LA-WEB). doi:10.1109/LA-WEB.2012.17
[24] Kapetanakis, K., Panagiotakis, S., Malamos, A. G., & Zampoglou, M. (2014). Adaptive Video Streaming on top of Web3D A bridging technology between X3DOM and MPEG-DASH. Telecommunications and Multimedia (TEMU), 2014 International Conference, IEEE. doi:10.1109/TEMU.2014.6917765
[25] Wu, J.-R., Duh, C.-D., Ouhyoung, M., & Wu, J.-T. (1997). Head Motion and Latency Compensation on. Lausanne Switzerland: ACM VRST '97.
[26] Lee, H. C., Kim, H. B., Lee , M. J., Kim , P. M., Suh, S. W., & Kim, K. H. (1998). Development of 3D sound generation system for multimedia application. Computer Human Interaction. IEEE.
[27] Cowan, B., & Kapralos, B. (2013). Spatial sound rendering for dynamic virtual environments. Digital Signal Processing (DSP). IEEE.
[28] Laitinen, M. V., Pihlajamäki, T., Erkut, C., & Pulkki, V. (2012). Parametric time-frequency representation of spatial sound in virtual worlds. ACM Transactions on Applied Perception (TAP). doi:10.1145/2207216.2207219
[29] Smus, B. (2013). Web Audio API. O'Reilly Media, Inc
[30] Spuy, R. v. (2015). Sound with the Web Audio API. In Advanced Game Design with HTML5 and JavaScript. IEEE.
[31] Smus, B. Retrieved from Developing Game Audio with the Web Audio API: http://www.html5rocks.com/en/tutorials/webaudio/games/ [accessed May 2015]
[32] Roberts, C., Wakefield, G., & Wright, M. (2013). The Web Browser As Synthesizer And Interface. NIME.
[33] Walker, W., & Belet, B. (2015). Birds of a Feather (Les Oiseaux de Même Plumage): Dynamic Soundscapes using Real-time Manipulation of Locally Relevant Birdsongs.
[34] openAL. Retrieved from http://openal.org./ [accessed May 2015]
[35] Seo, B., Htoon, M. M., Zimmermann, R., & Wang, C.-D. (2010). Spatializer: A Web-based Positional Audio Toolkit. ACE. ACM.
[36] Dirksen, J., Learning Three.js – the JavaScript 3D Library for WebGL, (2015)
[37] Three.js Retrieved from http://threejs.org [accessed May 2015]
[38] GidHub, webaudiox, Retrieved from
[39] https://github.com/jeromeetienne/webaudiox [accessed May 2015]
[40] GidHub, howler.js. Retrieved from https://github.com/goldfire/howler.js [accessed May 2015]
[41] GidHub, pedalboard.js. Retrieved from
[42] http://dashersw.github.io/pedalboard.js/ [accessed May 2015]
[43] GidHub, wad, Retrieved from https://github.com/rserota/wad [accessed May 2015]
[44] GidHub, fifer, Retrieved from https://github.com/f5io/fifer-js [accessed May 2015]
[45] DevBattles. Learn Web Audio API. Retrieved from http://www.devbattles.com/sand/post-69-Learn-Web-Audio-API [accessed May 2015]
[46] Toyosak, A., Flahive, L., & Diaz, J. (2015). Music Story 2: a Motion-Graphic-based Music Video Creator.
[47] Mahadevan, A., Freeman, J., & Magerko, B. (2015). EarSketch: Teaching computational music remixing in an online Web Audio based learning environment. Web Audio Conference.
[48] Paradis, M., Clarke, R. G., & Melchior, F. (2015). VenueExplorer, Object-Based Interactive Audio for Live Events. WAC. Paris
[49] Wyse, L. (2015). Spatially Distributed Sound Computing and Rendering Using the Web Audio Platform. 1st Web Audio Conference. Paris.
[50] Pendharkar, C., Bäck, P., & Lonce, W. (2015). Adventures in scheduling, buffers and parameters: Porting a dynamic audio engine to Web Audio.
[51] Schnell, N., Saiz, V., Barkati, K., & Goldszmidt, S. (2015). Of Time Engines and Masters An API for Scheduling and Synchronizing the Generation and Playback of Event Sequences and Media Streams for the Web Audio API. WAC. Paris.
[52] Clark, C., & Tindale, A. (2014). Flocking: A Framework for Declarative Music-Making on the Web. ICMC|SMC. Athens.
[53] Zampoglou, M., Spala, P., Kontakis, K., Malamos, A. G., & Ware, J. A. (2013). Direct Mapping of X3D Scenes to MPEG-7 Descriptions. Web3D '13, ACM. doi:10.1145/2466533.2466540
[54] Pohja, M. (2003). X3D and VRML Sound Components.
[55] Extensible 3D (X3D) Sound component. Retrieved from http://www.web3d.org/documents/specifications/19775-1/V3.3/Part01/components/sound.html [accessed May 2015]
[56] Ahmad, L., Boukerche, A., Hamidi, A., Shadid, A., Pazzi, R. (2008). Web-based e-learning in 3D large scale distributed interactive simulations using HLA/RTI. Parallel and Distributed Processing, IEEE
[57] Repplinger, M., Loffler, A., Schug, B., Slusallek, P. (2009). Proceedings of the 14th International Conference on 3D Web Technology, ACM
[58] Browsers support Web Audio API. Retrieved from http://caniuse.com/#feat=audio-api [accessed May 2015]

Andreas Stamoulias was born in Athens, Greece, in 1987. He received a B.Sc. in Applied Informatics and Multimedia in 2011 and a M.Sc. in Informatics Engineering, in 2014, both at the Technological Educational Institute of Crete. Since 2011, he has been a Researcher at the Multimedia Content Laboratory, dept. of Informatics Engineering, T.E.I. of Crete. His main interests are 3D graphics and multimedia web applications.

Eftychia Lakka was born in 1982. She received her BSc degree in Computer Science from the University of Ioannina, Greece (2006) and her M.Sc. in Synthesis of Images and Graphic Designs from the University of Limoges, France (2011). She has been as Research Associate at the Advanced Knowledge, Image & Information Systems Laboratory, dept. of Informatics, T.E.I. of Athens (2010-2014) and at the Multimedia Content Laboratory, dept. of Informatics Engineering, T.E.I. of Crete (2015-now).

Athanasios G. Malamos was born in 1969. He received his BSc degree in Physics from the University of Crete (1992) and his PhD from the Technical University of Crete in 2000. From 1997 to 2002 he was a Research Assistant and a researcher in the ICCS National Technical University of Athens. Since 2002 he is with the Technological Educational Institute of Crete at the Department of Informatics Engineering as an Assistant Professor (2002-2006) and as an Associate Professor (2006 until present). He is the coordinator of the Multimedia Content Lab, and has supervised many research projects granted with EU and National research funds. He has served as program committee member and reviewer for several international conferences and workshops. Dr. Malamos is a reviewer for IEEE, Springer as other international journals. He is member of the IEEE Computer Society, ACM SIGGRAPH and the WEB3D consortium. His research interests include multimedia semantics, AI and graphics