Audio Engineering Society

Convention Paper

Presented at the 117th Convention
2004 October 28–31 San Francisco, CA, USA

This convention paper has been reproduced from the author's advance manuscript, without editing, corrections, or consideration by the Review Board. The AES takes no responsibility for the contents. Additional papers may be obtained by sending request and remittance to Audio Engineering Society, 60 East 42nd Street, New York, New York 10165-2520, USA; also see www.aes.org. All rights reserved. Reproduction of this paper, or any portion thereof, is not permitted without direct permission from the Journal of the Audio Engineering Society.

A Host-Based Real-Time Multichannel Immersive Sound Playback and Processing System

Ramy Sadek[1]

[1] Institute for Creative Technologies / Univ. of Southern California, Marina Del Rey, California 90292, USA
sadek@ict.usc.edu

ABSTRACT

This paper presents ARIA (Application Rendering Immersive Audio). This system provides a means for the research community to easily test and integrate algorithms into a multichannel playback/recording system. ARIA uses a host-based architecture, meaning that programs can be developed and debugged in standard C++ without the need for expensive, specialized DSP programming and testing tools. ARIA allows developers to exploit the speed and low cost of modern CPUs, provides cross-platform portability, and simplifies the modification and sharing of codes. This system is designed for real-time playback and processing, thus closing the gap between research test-bed and delivery systems.

1. INTRODUCTION

Multichannel-audio researchers have limited options in choosing a computing test-bed. Developing modules for commercial audio playback software carries a steep learning curve and their implementation is time-consuming. Conversely, technical computing packages (e.g. MATLAB and Simulink[1]) provide convenient test-beds for multichannel DSP development, but they are not well-suited to use as playback systems.

Traditionally, practitioners have had to choose between ease of implementation and ease of deployment. For example, when developing a new pan-pot law, it is easy to experiment in a technical computing package. However, once the desired pan-pot formulation is found, testing it with experimental subjects often requires redevelopment, especially if the tests involve interactive applications.

The virtual reality (VR) community also suffers from a lack of multichannel playback systems. While the videogame industry provides multichannel audio solutions for standard 4.1, 5.1 and 7.1 configurations, many VR setups use high-end or custom audio hardware setups for which gaming solutions are unsuitable. Consider a 10.2 system, or a hemispherical speaker array to be used for interactive VR applications; gaming hardware solutions lack sufficient channels, and have

[1] The Mathworks

Report Documentation Page	*Form Approved* *OMB No. 0704-0188*

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE **OCT 2004**	2. REPORT TYPE	3. DATES COVERED **00-00-2004 to 00-00-2004**

4. TITLE AND SUBTITLE **A Host-Based Real-Time Mutlichannel Immersive Sound Playback and Processing System**	5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of Southern California,Institute for Creative Technologies,13274 Fiji Way,Marina del Ray,CA,90292**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR’S ACRONYM(S) 11. SPONSOR/MONITOR’S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**6**	

hard-coded pan-pot laws that assume a planar array. Additionally, the SNR performance on consumer-grade hardware components is too low for many such applications.

We present an Application Rendering Immersive Audio (ARIA) as a solution to both scenarios. With this system, the audio researcher can easily develop, integrate and test her algorithms while allowing the VR research team to employ processing techniques required for their experiments (e.g. configuration-appropriate pan-pot laws, room equalizations, acoustic simulations, etc). ARIA achieves this flexibility by leveraging the low cost and rapid development of current computer architectures. ARIA can be distributed across several computers, allowing easy scalability with affordable computer systems.

2. DESCRIPTION AND MOTIVATION

ARIA is a general-purpose host-based multichannel streaming DSP system intended for real-time processing, rendering and recording. In particular, this system is useful to members of the Virtual Reality, multichannel and immersive audio research communities. The ARIA design is guided by two premises.

(i) Commodity CPUs are powerful enough to handle substantial DSP computations.

(ii) For collaborative research applications, C++ code is preferable to DSP assembly code because it is readily manipulated by practitioners with various backgrounds. Conversely, DSP assembly programming is abstruse and highly specialized.

Premise *(i)* implies that the modern CPUs available to researchers, graduate students and developers can perform extensive signal processing operations on numerous signals. Premise *(ii)* bears further consideration.

It is an often-lamented fact in software engineering that software codes often have a longer lifespan than intended by their original authors. Because of this phenomenon, clarity, cleanliness and simplicity are extremely important if software is to have long-term use. The world of research software is particularly sensitive to these issues because researchers often reuse software to build upon the work of one another. Also, optimizing compilers are often better able to produce efficient assembly code than are their human counterparts. Additionally, the wide array of tools for C++ programming significantly eases the development process. Finally, DSP assembly is often incompatible between DSP chips. This often means that a lab cannot scale performance of their existing DSP software by purchasing faster hardware. The reverse is true for software written in high-level languages. For these reasons, disciplined C++ programming is preferable to DSP assembly for research applications.

These sentiments are well known to the audio community, yet labs continue to target DSP systems (e.g. Lake's Huron) for developing research methods. High-end DSP systems are well-suited to final deliverables, but due to their rarity, inaccessibility and high cost, they often function poorly as collaborative research platforms.

ARIA addresses these issues by exploiting the power of modern computer architectures and by leveraging ubiquitous commodity hardware.

3. PREVIOUS WORK

Much of the existing work in multichannel audio development kits has been driven by the videogame industry (e.g. DirectSound3D, Miles, FMod, OpenAL, etc.). Numerous audio research groups have also created systems to drive their own experiments [4] [10], but very few have made systems intended for general use by other groups. The most notable system intended for use by other groups is the Virtual Audio Server [2] (VAS), discussed in Section 3.2.

The need for a general stream-processing system is demonstrated by the large number of papers in which a processing/playback system has been developed for particular research tasks.

3.1. Game Audio Systems

The videogame industry has focused on providing fully-featured audio subsystems. Indeed, PC games are able to create rich audio environments. However, these production-oriented systems do not provide the flexibility needed for research [2].

Three-dimensional game audio systems like OpenAL and DirectSound3D [2], combined with the environmental effects of EAX [3], create effective auditory scenes. Unfortunately, these development kits are not meant for easy extension; one cannot easily develop and load a novel panning/processing module into these systems.

For instance, a VR group looking to test the efficacy of a periphonic pan-pot law would need to do a great deal of programming beyond implementing the pan-pot. Although OpenAL, for example, is easy to use, it does not allow easy access to the underlying channel buffers. Therefore, the implementors would need to use a lower-level API (e.g. DirectSound), which is a difficult task.

Likewise, the environmental effects of EAX work nicely in gaming. Unfortunately, the hardware that supports it lacks the appropriate number of output channels [8] and SNR for high-end VR. Similarly, VR research groups interested in testing the efficacy of high-resolution audio during source localization, for example, will find that game audio hardware is unable to support the appropriate sample rates and bit-depth required for their experiments.

3.2. Virtual Audio Server (VAS)

The Virtual Audio Server (VAS) [2] is a tool for creating rich virtual acoustic environments. VAS is available to the public [9], but is implemented in an SGI-specific way, limiting the number of sources to 16 [2]. Additionally, while VAS is flexible, several types of processing do not fit into its design. For example, it is unclear how a room-equalization process such as the one described in [7] would fit the VAS paradigm. The goals of ARIA and VAS differ greatly. Like ARIA, VAS is designed for use as a flexible C++ development toolkit for audio processing; however, unlike VAS, ARIA is not tailored specifically to VR applications. Rather, ARIA is capable of more general processing than is VAS.

4. DESCRIPTION OF ARIA

Easy portability is an extremely important part of ARIA's design. Many research groups and end-users, in disciplines ranging from DSP algorithm development to computer music, are in need of multichannel audio solutions. The varied needs of these disparate groups leads to widely differing computing platforms between them. To support such diversity, the main components of ARIA have been designed and implemented in a platform-independent manner.

Five sub-modules comprise ARIA. These are:

1) Processing core
2) Backend
3) TCP/IP network layer
4) GUI
5) Client Application Library

These modules are loosely coupled to one another and each module is easily modified or replaced. For example, one may develop a new GUI module without modifying or re-compiling the other modules.

4.1. Processing Core

The processing core applies an arbitrary list of processes to signals, and can be parallelized. To maintain flexibility, ARIA enforces no hard timing constraints. This flexibility allows a trade-off between performance and process complexity, often unavailable in DSP and real-time systems. In such systems, processes are allotted time quanta irrespective of their complexity. Hard constraints on timing complicate development and debugging: for example, printing to output or using stepping debuggers is often impossible. In ARIA, excessive computations during processing leads to audio drop-outs, an acceptable compromise during the develop/debug cycle. Untimely processing results in a warning, rather than in a system-crash or lock-up.

ARIA readily supports signal-level parallelism. Parallelization of the entire computation depends on the independence of the processes applied to each signal. If, for example, the desired computation requires a convolution of a large impulse response for each channel of a 5.1 system, ARIA can run one convolution process per CPU, if available. These CPUs need not be in the same machine; ARIA sessions can be distributed across multiple computers by synchronizing their audio hardware [4]. Therefore, tuning and optimization of the final playback session is controlled by the user. ARIA places no restrictions on either the number of processes applied to each signal or the complexity of those processes.

[2] Microsoft

[3] Creative Labs

[4] High-end audio hardware currently supports sample-accurate synchronization at high-resolution sample rates.

Developers writing processing modules (*Processor* classes) for ARIA may also exploit any process-level parallelism in their own implementations. This feature allows for the precise tuning of algorithms and their scheduling.

In short, the processing core is comprised of a list of signal sources and a list of processes applied to each signal. By inheriting from the *Source* base class, developers can easily create sources that perform synthesis operations based on MIDI input, read audio data from a network socket, etc. Such extensions are trivial by design; ARIA requires minimal overhead to create and register a new class. This simplicity holds both for *Processor* and *Source* classes.

4.2. Backend

The backend handles the transport of audio data from main memory to the output hardware for D/A conversion. Because the backend is modular, ARIA easily ports to different operating systems, hardware devices and driver configurations. Thus, collaboration between research teams is not hindered by computing platform differences between their labs.

4.3. TCP/IP Network Layer

The TCP/IP network layer provides an abstraction between the host and the client applications it serves. This abstraction allows ARIA to be controlled remotely and to be distributed across multiple machines. Via the network layer, client applications send commands and data to control the various system components. Similarly, components can send data via this layer, sparing developers the burden of implementing network communications.

The network layer allows developers to easily send commands and/or data to the modules they create. (e.g. Musical synthesizers, voice-communications for remote VR, etc).

4.4. GUI

The GUI and the network layer are closely related in that the GUI is not directly attached to ARIA; rather, it runs as its own process, sending commands via the network. Hence, ARIA ports without concern for graphical systems. The provided reference GUI is written in Java to ensure that it can be run on any display-capable hardware configuration with a Java Virtual Machine (JVM).

4.5. Client Application Library

Finally, the application library provides an API for client applications to control ARIA. For high-power applications, this library uses network communications to send commands (optionally audio data as well), so that ARIA may utilize an entire system for computations. For less demanding applications, the library and ARIA can run on the same host.

4.6. Performance

System performance is governed by computing power available on the host, the latency of output hardware, and the latency of network communications. There is a trade-off between computing power and hardware latency; decreasing hardware latency increases demands on the host CPU.

Because ARIA uses commodity hardware, upward hardware scalability is inexpensive. At time of writing, a computer system from a leading manufacturer with a 2.4GHz x86 CPU and 512MB of RAM can be readily purchased for less than $500 (US). We have tested a similar (3.06GHz) computer running two processes (a 5-channel pan-pot [3] process and a volume control process, no compiler optimizations enabled) with up to 175x5=875 mixed streams [5] without drop-outs.

If ARIA is running on a remote host, the network contributes additional latency; however, on current networks (i.e.100Mb/s, 1Gb/s) this latency can be sub-millisecond. When the latency of the network communications is sufficiently below that of the output hardware, the network latency does not add to the total output latency. Current output hardware latencies are as low as 1.5ms.

5. EXAMPLE APPLICATIONS

ARIA is useful in many diverse scenarios. Here, we describe applications in virtual reality and computer music.

5.1. Virtual Reality with Room Equalization

Consider a research team using an elaborate virtual reality setup consisting of a projection screen and an ITU 5.1 audio configuration. Using the systems listed in Section 3, the team would be unable to perform room

[5] 32bits at 48KHz. Hardware output buffers were 1024 samples (21 ms) long.

equalization algorithms without additional hardware. In order to create a convincing acoustic environment, it is vital to compensate for the effect of the screen and ceiling. With ARIA, it is simple to add such a room equalization module as a final software process after virtualization and before output.

Upgrading the setup to utilize, for example, a twenty-channel periphonic audio system creates further problems for previously existing systems. In particular, the systems listed in Section 3 are unable to create and manage the channels required to drive the loudspeakers discretely [6].

Finally, imagine that the research team seeks to measure the effect of fine-granularity ITDs in source-localization via high-resolution audio vs. that of CD-quality audio. The team would need to upgrade its audio data and its output hardware. ARIA handles these upgrades without modification.

5.2. Computer Music

In addition to its research functionality, ARIA has creative applications. Large loudspeaker array music has long been an area of interest to composers and musicians in many genres. Of particular note are the complicated delivery mechanisms constructed by Edgard Varése and Karlheinz Stockhausen for the Brussels Worlds' Fair and EXPO 70, respectively [7]. Their tape-based compositions and complex designs made it difficult to control and modify these installation-pieces. Computer-based electro-acoustic compositions are more readily edited than tape-based compositions. Therefore with the numerous output channels in current audio hardware, ARIA could drive these setups, allowing easily modified content, routing and panning.

Interest in spatial elements of music continues in popular trends today. For example, the popular band U2 is working to develop "an Audio Spotlight [8] system for rock concert applications" [13]. An ARIA installation could manage the live audio streams and control of the spotlight hardware, receiving commands remotely over the network, and responding appropriately. In turn, one could attach input hardware (e.g. mixing boards or joysticks) to another computer which would translate inputs into network commands to drive ARIA. One can imagine passing wireless game-console controllers to the audience, allowing them to pan sounds themselves. Unconventional systems are often difficult to implement using conventional hardware and software. However, because of its high-level design, ARIA is easily extended to accommodate such novel applications.

6. CONCLUSION AND FUTURE WORK

The ARIA framework continues to evolve. While the system is fully-functional and ready for use, there are a number of conveniences we would like to provide. Examples include efficient FFT methods with selectable windowing schemes; convolution methods in both the time domain and the frequency domain; and an abstraction layer for threads, allowing developers to implement parallel methods without direct system calls. In short, our current and future efforts will make ARIA even more user-friendly. Additionally, we would like to include platform support for small computers such as hand-held devices and game consoles. Further information will be available on our website: *http://www.ict.usc.edu/sound/*. We look forward to suggestions from the community.

7. ACKNOWLEDGEMENTS

This paper was developed with funds of the Department of the Army under contract number DAAD 19-99-D-0046. Any opinions, findings and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the Department of the Army.

Thanks are due to Chris Kyriakakis and Tomlinson Holman for their assistance in this project. Additional thanks to Aaron Isaksen and John DeWeese for their detailed feedback. Thanks as well to Kumar Iyer, Farah Kidwai and David Sachs for extensive proof-reading, editing and support.

8. REFERENCES

[1] Moorer, J., "48-bit Processing Beats 32-bit floating-point for Professional Audio Applications".

[6] To clarify, several systems offer an arbitrary number of channels (e.g. DirectSound and CoreAudio) but they do not offer platform independence or easily replaced panners.

[7] A detailed description of these multichannel systems is beyond the scope if this paper. We refer the curious reader to [11] and [12].

[8] Holosonic Research Labs

Presented at the AES107th Convention, New York, USA, September 24 – 27 1999. Preprint Number 5038.

[2] Fouad, H., Ballas, J., Brock, D., "An Extensible Toolkit for Creating Virtual Sonic Environments". Proceedings of the 2000 Intl. Conf. on Auditory Display (Atlanta, USA, May 2000).

[3] Sadek, R., Kyriakakis, C., "A Novel Multichannel Panning Method for Standard and Arbitrary Loudspeaker Configurations", Presented at the AES117th Convention, San Francisco, USA, 2004 October 28 – 31.

[4] V. Pulkki, "Virtual Source Positioning Using Vector Base Amplitude Panning", J. Audio Eng. Soc., vol. 45, no. 6, pp. 456-466, 1997 June

[5] Bharitkar, S., Hilmes, P., and Kyriakakis, C., "Robustness of Multiple Listener Equalization with Magnitude Response Averaging", AES 113th Convention, Los Angeles, 2002 October.

[6] Bharitkar, S., Hilmes, P., and Kyriakakis, C., "Robustness of Spatial Averaging Equalization Methods: A Statistical Approach", Proc. 36th IEEE Asilomar Conference on Signals, Systems, & Computers, Nov. 2002. Pacific Grove. CA

[7] Bharitkar, S., and Kyriakakis, C., "Perceptual Multiple Location Equalization with Clustering", Proc. 36th IEEE Asilomar Conference on Signals, Systems, & Computers, Nov. 2002. Pacific Grove, CA

[8] Tsingos, N., "Perceptual Audio Rendering of Complex Virtual Environments", Proc. ACM SIGGRAPH 2004.

[9] *http://www.icg.seas.gwu.edu/research_VR.htm*

[10] Teutsch, H., Spors, S., Herbordt, W., Kellerman, W., Rabenstein, R., "An Integrated Real-Time System for Immersive Audio Applications", IEEE WASPAA 2003, New Paltz, NY.

[11] Ouellette, Fernand. *Edgard Varèse*. New York: The Orion Press, 1968

[12] Kurtz, Michael. *Stockhausen: A Biography*. London and Boston: Faber and Aber, 1992.

[13] *http://www.holosonics.com/customers.html*