Audio in the UCSB CNSI AlloSphere

The UCSB AlloSphere is a joint effort of the California NanoSystems Institute (CNSI) and the graduate program in Media Arts and Technology (MAT) at the University of California Santa Barbara (UCSB). It is currently under construction, with completion scheduled for the first half of 2006. The AlloSphere is designed as an immersive computational interface for 10 to 20 users, featuring surround-sound data sonification and immersive visualization (i.e., 3D audio and video projection) on a spherical surface. It will provide interactive control by the means of microphone arrays, cameras, and mechanical, and magnetic input tracking. The actual shape of the AlloSphere can be described as two hemispheres with 16-foot radii pulled 8 feet apart, placed in a 3-story anechoic chamber. A 7-foot-wide bridge runs across the center, supporting the users. This document describes the requirements for the audio component of the AlloSphere, introduces the three prevalent spatial sound processing technologies in use today, and outlines the AlloSphere audio input and projection design and implementation plan, from low-level transducer elements to high-level network protocols.

Contents

Introduction

The UCSB AlloSphere will be an immersive multimedia computational interface; as such, it must support very high resolution audio and video projection, and a diverse array of multi-modal input sensors. Our goal is to provide what we envision will be the computational and interface capability of a computer ten years in the future.

We will use the sphere to develop next-generation applications and modalities of man-machine interaction that are impossible today due to technological and methodological limitations, but will be widely available to scientific, engineering and artistic community in the future. This is a unique opportunity to use a “time machine” to design and test these technologies while contributing to leading-edge research at CNSI and to the wider communities in each of our fields of expertise.

Many-channel spatialized sound (also referred to in the literature as surround, plurisonic, or periphonic sound) is a central component to the AlloSphere multimedia computational interface. Spatial sound representation, processing, and projection has been one of the core research and development topics at UCSB’s CREATE Center for over a decade, with the Creatophone concert instrument, CSL library and CRAM tool set projects forming the core of a coordinated effort to develop playback systems and distributed computing frameworks capable of supporting many moving sources and hundreds of channels of output (Pope 2000; Pope et al. 2001a).

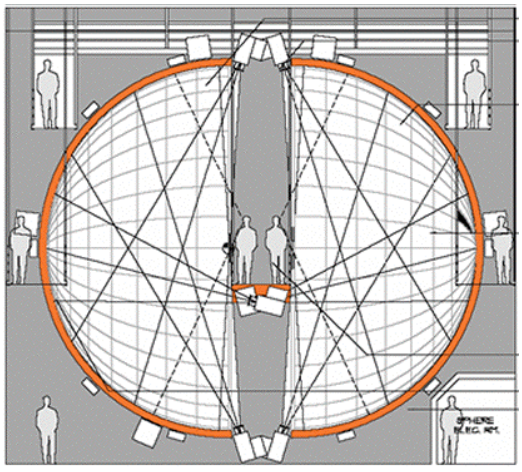

Figure 1: Side view of the CNSI AlloSphere showing the video projectors at the top and bottom and users on the bridge in the center.

In this document, we present the audio output requirements of the UCSB AlloSphere, and then discuss the current and near-term future technologies we intend to use to meet those requirements. We introduce the three primary techniques used for spatial sound performance (vector-based panning, ambisonic representations, and wave field synthesis), and detail the role that they will play in the AlloSphere audio framework (Appendix 1). The core of this working paper presents the output hardware (loudspeaker array) and computational infrastructure that we are designing for use in the AlloSphere.

The CNSI and the AlloSphere

The California NanoSystems Institute (CNSI) is a dual-site research institute, with large facilities at UCLA and UCSB. The UCSB home of CNSI (http://www.cnsi.ucsb.edu) will provide 61,994 square feet of interdisciplinary research laboratories for sophisticated imaging, spectroscopy, and bio-nanofabrication; digital media research laboratories; and conference and multipurpose facilities. Collectively, these laboratories, including a state-of-the-art clean room, represent the hub of the institute's cross-disciplinary and collaborative research program.

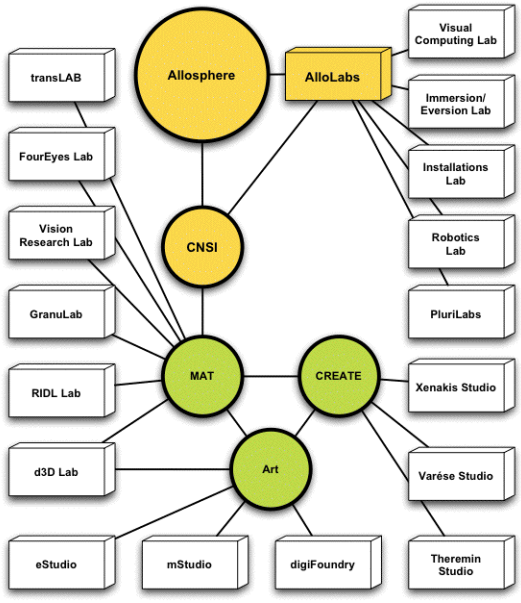

Figure 2: CNSI partner lab relationships

The schematic figure above shows the relationships of the CNSI partner labs, with the AlloSphere's staging labs on the upper right and the MAT core departments in the lower half. The fully exploded view would include the CNSI scientific partners in chemistry, materials research, and bio-informatics.

The AlloSphere is situated at one corner of the CNSI building, surrounded by a suite of machine rooms and staging areas for media researchers. Figure 3 below shows the media research labs in the CNSI, with the AlloSphere at the top and its associated labs (motion capture, robotics, interactive installation, distributed systems, media post-production, etc.) below it.

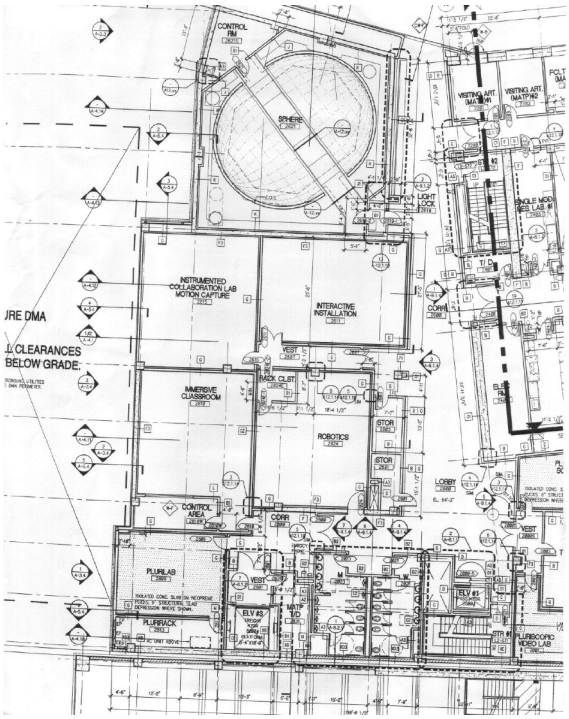

Figure 3: AlloSphere and related staging areas (media workshops) at the north end of the CNSI building at UCSB

The AlloSphere space consists of a 3-story empty cube that is treated with extensive sound absorption material (4-foot foam wedges on almost all inner surfaces) making it one of the largest anechoic chambers in the world. Standing inside this chamber are two 16-foot-radius hemispheres constructed of perforated aluminum that are designed to be optically opaque (and have low optical scatter) and acoustically transparent.

Figure 4 below is a detailed drawing showing the AlloSphere as seen from above. One can see the sound treatment (dotted pattern around the walls), the machine rooms (top-left), and the bridge through the center of the sphere. Figure 5 is another view cutting across the middle of the AlloSphere in the horizontal plane; here again, control

and machine rooms are visible, as is the sound treatment and user bridge. Figure 6 is a photograph of the room at mid-construction, with the bridge and upper ring truss (steel support structure) visible.

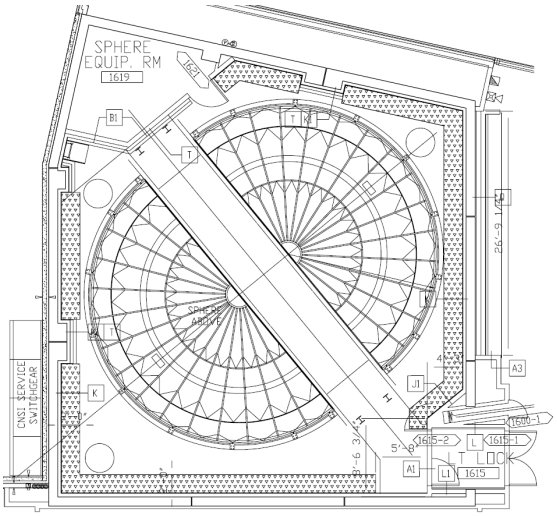

Figure 4: AlloSphere detail (looking down from above) showing the sound treatment, two hemispheres, and bridge

There are to be 14 high-resolution video projectors mounted around the seam between the two hemispheres, giving eye-limited resolution on the entire inner surface. Other documents describe the design of the video computation, rendering, and projection infrastructure.

The loudspeaker array (400-500 individual speaker elements plus sub-woofers) will be suspended behind the aluminum screen, hung from the steel infrastructure in rings of varying density. These speakers are connected to multiple Gigabit Ethernet LAN fibers, driven from the server farm running custom-developed spatial sound projection software.

The requirements placed on the AlloSphere audio output system are the subject of the next section, after which we present our current design to satisfy these requirements.

Requirements for AlloSphere Audio

Our goal for the UCSB AlloSphere is to build an immersive multimedia computational interface that provides "sense-limited" resolution in both the audio and visual domains. This means that we have to achieve visual resolution that is nearly as good as the spatial resolution of our eyes (meaning on the order of 150 million pixels distributed on the surface of a sphere), and that the spatial resolution for the audio output must allow us to place virtual sound sources at arbitrary points in space with convincing synthesis of the spatial audio cues used in psychoacoustical localization. Complementary to this, the system must allow us to simulate the acoustics of measured or simulated spaces with a high degree of accuracy.

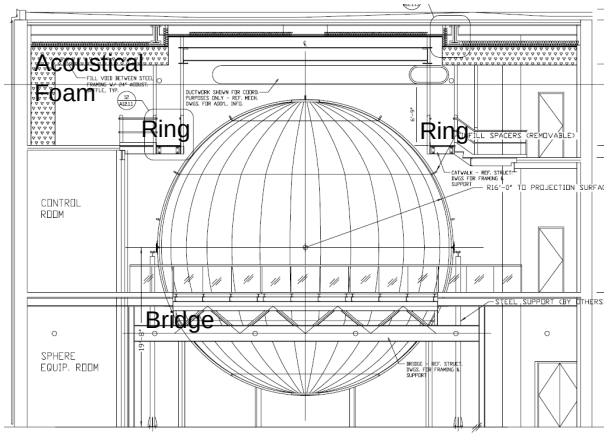

Figure 5: AlloSphere horizontal perspective with the user bridge running below the center and the support structure shown

The actual number of loudspeakers required for this, and their placement, are matters of some discussion, and will be derived carefully below. The positions are assumed to be some approximation of a symmetrical arrangement with respect to rotation around the up/down axis of the sphere.

Acoustical Requirements

In order to provide for "ear-limited" dynamic, frequency, and spatial extent and resolution, we require the system to be able to reproduce in excess of 100 dB sound pressure level near the center of the sphere, to have acceptable low- and high-frequency extension (-3 dB points below 80 Hz and above 15 kHz), and to provide spatial resolution on the order of 3 degrees in the horizontal plane (i.e., 120 channels), and 10 degrees in elevation. To provide hi-fidelity playback, we require audiophile-grade audio distribution formats and amplification, so that an effective signal-to-noise ratio exceeds 80 dB, with a useful dynamic range of > 90 dB.

To be useful for data sonification and as a music performance space, the decay time (the "T60 time") of the AlloSphere must be less than 0.75 seconds from 100 Hz to 10 kHz. This is primarily an architectural feature related to the properties of the aluminum screen and the sound absorbing

treatment in the anechoic chamber. With the exception of the spatial resolution, these values are typical of high-end surround-sound reinforcement systems such as found in modern recording studios or theaters.

Figure 6: AlloSphere construction status as of Summer, 2005, showing the bridge across the bottom and "halo" ring truss at the top

Sound Synthesis, Processing, and Spatialization Software

Since the AlloSphere is to foster the development of integrated software for scientific data sonification and "auditory display" as well as artistic applications, it is essential that the software and hardware used for audio synthesis, processing, control, and spatial projection be as flexible and scalable as possible.

We require that the audio software libraries support all popular synthesis and processing techniques, that they be easily combined with off-the-shelf audio software written using third-party platforms such as Csound, Max/MSP, and SuperCollider, and that they support flexible control via (at least) the MIDI and Open Sound Control (OSC) protocols.

Due to the sophistication of the audio synthesis and processing techniques used in AlloSphere applications, and the expected very large number of final output channels, we require that the core audio libraries support easy inter-host streaming of large numbers of channels of high-resolution (24-bit, 96 kHz) audio, probably using both the CSL/RFS and SDIF networked audio protocols.

Spatial Sound Processing

There are three techniques for spatial sound reproduction used in current state-of-the-art systems: (1) vector-based amplitude panning, (2) ambisonic representations and processing, and (3) wave field synthesis. We describe each of these techniques in more detail in Appendix 1; the AlloSphere speaker count and configuration should support the use of any of them for sound spatialization. This implies high speaker density (on the order of one source per square yard of surface, or about 380 channels), and a semi-regular and relatively symmetrical speaker layout.

Audio Input

For input, it should be possible to gather audio input from users on the bridge. Several microphone plugs must be provided to plug in near-field microphones on the bridge, and a small microphone array (a ring of 16 or so microphones) should be installed behind the AlloSphere's surface structure.

Computational Infrastructure

The basic requirements given above lead us to believe that off-the-shelf computing and interface solutions will prove to be inadequate. The premise of surround video/audio and real-time interaction does not simply imply the linear scaling of number of loudspeakers, pixels and polygons, but demands an exponential increase in I/O bandwidth and processing power in order to process the underlying data, interactively, to the degree of complexity required by the emergent modes of human-computer interaction.

The requirements in terms of processing power and interconnect bandwidth are clear; AlloSphere applications will require a server farm consisting of 20-50 machines dedicated to video and audio processing, and a low-latency interconnection fabric so that data can be processed on multiple computers (in a variety of topologies) in real time.

For the audio output component, we require a distributed computational cluster capable of providing low-latency FIR/FFT convolution for sound rendering to each of 512 or more loudspeakers. Actually, most of the algorithms work in terms of pairs of FFT-based convolutions per channel, meaning 1024 FFTs. Assuming that the current FFT implementation requires about 1 MFLOP, we see that the basic requirement for audio processing is on the order of 1 GFLOP.

The minimum required I/O bandwidth should permit uncompressed audio playback of 512 channels of 24-bit/96 kHz high-resolution audio in the

sphere (at the same time as the video). This adds up to about 1.3 GBps (ignoring the protocol overhead, which you can never ignore). Ideally, would also want to stream many channels between the synthesis and spatialization servers on the same LAN, leading to the eventual requirement of several times the above estimate.

Audio Projection System Design

We assume below that the reader is familiar with the spatial sound rendering techniques presented in Appendix 1, and with Appendix 2's derivation of the number of channels desired and optimal speaker array geometry (speaker placement); we now turn to the actual projection hardware and driver software for audio in the UCSB CNSI AlloSphere.

The surface of the AlloSphere itself is a perforated aluminum projection screen. The loudspeakers will be mounted behind the screen, and for the purposes of acoustics, we assume that the screen material is like a normal speaker's grill cloth (i.e., acoustically inert). The room that the AlloSphere is standing in is a 3-story anechoic chamber.

The two hemispheres that comprise the spherical structure are supported on tubular-beam legs, and there is an outer tubing structure supporting the projection surface. The heavy-duty supports include the upper ring truss "halo" above the sphere, which will serve as the mounting scaffolding for the upper hemisphere's loudspeakers.

It has been a major project to derive the optimal speaker placements and speaker density function for use with mixed-technology many-channel spatialization software (see App. 2). Our driver placement design comprises between 400 and 500 speakers arranged in several rings around the upper and lower hemispheres, with accommodations at the "seams" between the desired equal and symmetrical spacing and the requirements of the support structure.

We use an iterative design technique to suggest an optimal placement and then compare it to an actual (possible) placement that is a compromise to accommodate the screen's mounting framework and the "seam" between the 2 hemispheres (with the doors).

We envision densely packed circular rings of speaker drivers running just above and below the equator (on the order of 100-150 channels side-by-side), and 2-3 smaller and lower-density rings concentrically above and below the equator (See the figure above). We assume that the main loudspeakers will have limited low-frequency extension, in the range of (down to) 200-300 Hz. To project frequencies below this, one or more large sub-woofer(s) will be installed, possibly mounted on the underside of the bridge.

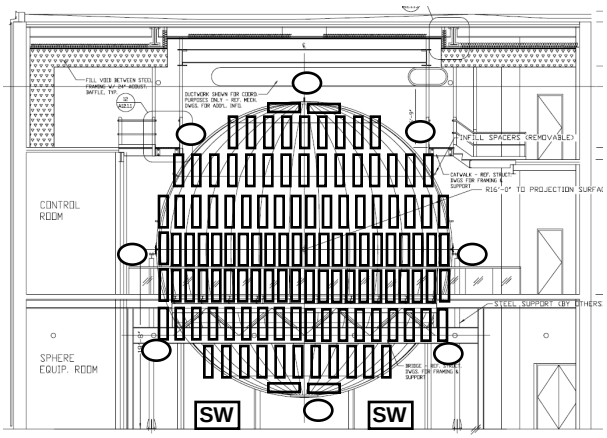

Figure 7: AlloSphere speaker layout (side view) showing dense rings of transducers (narrow rectangles), DAC/distribution amplifiers (ovals), and 1 or more sub-woofers (blocks)

The (passive) speaker elements will be wired to a set of 8-16 networked interface, digital-to-analog convertor (DAC), amplifier boxes, each of which supports in the range of 32-128 channels and has a Gigabit Ethernet interface.

The following sections discuss each of these facets of the design in sequence.

Loudspeaker Technologies

Several distinct technology families (and a plethora of design options) are in common use in modern loudspeaker design. Building a speaker consists of choosing a core driver technology (dynamic, electrostatic, ribbon, piezo-electric, etc.) for one or more driver elements, designing a set of cross-over filters, and constructing a speaker enclosure. It is commonplace today to find multi-driver (i.e., woofer, mid-range, and tweeter) mixed-technology (i.e., dynamic woofer and ribbon tweeter) systems, although there are still proponents of minimalist designs (e.g., "1-way" Lowther speakers and full-range electrostats).

To specify a speaker enclosure, one must take into account the speaker driver, the "motor board" on which it is mounted, handling (absorption or redirection) of the speaker's back-wave radiation, crossover circuit board mounting, and other considerations.

Based on our decade of experience with the "Creatophone" many-channel spatial sound playback system, and the scope of the desired 500-channel system, we started to look for alternatives to traditional multi-way dynamic-driver-based "box" speakers. (Considerations included weight, size, vibration, power consumption, reliability, and cost.)

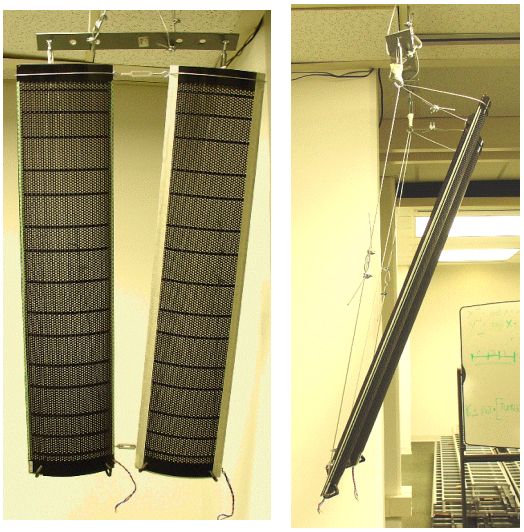

Figure 8: 9-by-32-inch curved electrostatic panels made by MartinLogan, Inc. and suspended by a cable and truss assembly

After investigating several other options (e.g., ribbon tweeters or actuator panels), we began experimenting with medium-sized electrostatic loudspeaker (ESL) panels (see the figure above). ESL function like large sandwiches where a thin foil (membrane) is stretched between 2 stiff planes of metal (stators) carrying a high electrical charge; when the charge between the stators is changed, the membrane moves. ESL panels are lightweight and require no box and only modest damping; they radiate in a "figure-8" pattern, meaning front- and rear-ward acoustical waves with relatively low output to the sides (an advantage over most other designs, which have a wider and more uneven radiation pattern). (For an introduction to ESL technology, see http://martinlogan.com/esl_technology.html.)

ESLs come in a variety of sizes, with the mid-range extending from 8 inches to 1 foot in width and 2 to 3 feet in height; they are easy to mount or hang (4 lbs each); they have a frequency response extending to over 20 KHz in the treble, and down to 400-450 Hz in the lower midrange for smaller panels, down to 200 Hz for larger ones. The driver's frequency response is important because it means that there need be no cross-over filter in the mid-range audio frequencies, leading to better sound and simpler electronics. We have been using several sets of ESL panels made by MartinLogan, Inc. of Lawrence, Kansas.

The down-side of ESLs is electrical: they have very low impedance at high frequencies (down to 1.5 Ω at 20 KHz, meaning that they need very high-current amplifiers), and the stator panels on either side of the thin moving membrane must be charged to a very high voltage (on the order of 5 kV), requiring step-up transformers. This circuitry need not be bulky, however, and has no heat sink. The figure below shows an example, the cross-over (large inductors to the left), step-up transformer (middle), and stator power circuit board (upper right) from a MartinLogan reQuest speaker.

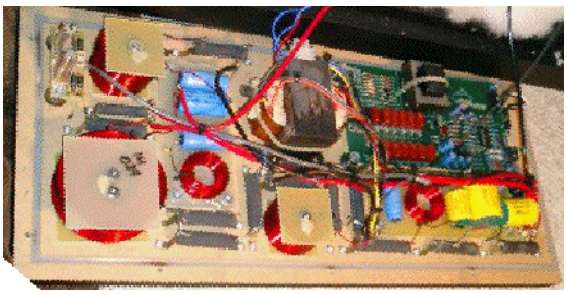

Figure 9: ESL cross-over and step-up transformer from a MartinLogan, Inc. loudspeaker

Depending on the size of the ESL panels used, the system might require mid-woofers (100 - 400 Hz or so) for good spatialization, since most sub-woofers extend up to the range of 150-200 Hz at the maximum. If required, these mid-range speakers could be less closely spaced, e.g., as 80 channels distributed over the sphere. A separate active dynamic sub-woofer cabinet(s) will be mounted on the floor under the Sphere (or attached to the underside of the bridge).

Hanging ESL Panels

Assuming a medium-sized ESL panel, we can proceed to design a framework to hang them around the outside of the sphere, attaching the mounting framework to the room's main ring truss assembly, rather than to the sphere surface's mounting framework (at least for the upper hemisphere). The panels can be hung ("flown") from an assembly of cables and cross-members behind the sphere surface. The figures above show a pair of panels (taken from MartinLogan "Aerius i" model speakers) and their mounting cables; below is a detail of the cable suspension. Because of the rela-

tively low weight (relative to powered speaker boxes), it is straightforward to scale this mounting approach to ESL panels of different dimensions and aspect ratios, and to hang this network from the halo ring truss for the upper hemisphere and the sphere's "girdle" for the lower hemisphere in the CNSI structure.

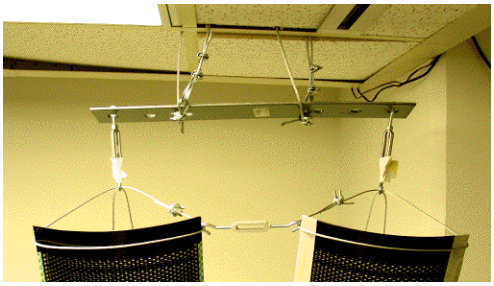

Figure 10: Detail of the lightweight cable/truss suspension of ESL panels

Alternatively, these panels could be clamped onto a fixed frame at their edges (see their use in commercial products, e.g., http://martinlogan.com/aeon_i_speaker.html). For rear clearances, while the ESL panels are less deep than "box" speakers, the cable/truss mounting framework will require approx. 24" rear working space where possible.

Remember that ESLs radiate in a figure-8 pattern; for loud levels, we must assure that the space's acoustical treatment can handle (absorb) the panels' back-wave.

While the panels are quite light, the distribution amplifiers are larger and heavier (max 100 lbs, 2 cubic ft.), and must be mounted near the speaker groups on the support framework. Each of these will require up to 5 KW of clean power (and a fiber gigabit Ethernet input, of course).

Loudspeaker Interconnection

Groups of (passive) speaker elements will be wired to custom-built interface boxes, which consist of a Gigabit Ethernet interface, digital/analog convertor, power amplifier, and step-up transformer. The bulk of the circuitry is based on a design developed at the CNMAT Lab at UC Berkeley (Freed 2005) for their 120-channel loudspeaker array.

The required network bandwidth leads us to plan a system with multiple 1000BaseT trunks, (or a 10KBaseT trunk, when available). Future versions of FireWire, running non-IP protocols, might be another audio-LAN distribution solution. Given current high-resolution standards, we can calculate that 256 channels would require guaranteed 620 MBit/sec throughput (256 * 24-bit * 96-KHz ≈ 620 MBit/sec), which would flood a single 1000BaseT LAN due to the overhead imposed by the IP protocol (or any other).

Each of the network interface boxes will use a semi-private LAN interface and support between 32 and 128 channels; it should consist of:

- 1000BaseT (and/or FireWire1600) interface
- Control logic, input buffering, etc.
- 32-128 channels of:
  - Digital-to-analog convertor
  - Crossover/compensation filter
  - 50 watt (at 2 Ω) Class D amplifier
  - Step-up transformer (for ESL voltages)

The control logic (implemented in a custom-programmed Xylinx FPGA chip) handles the network protocol (probably SDIF audio format wrapped in the CSL RFS protocol over UDP), some digital signal processing (FIR filter), and the demultiplexing of the in-coming sample stream into separate buffers for the DACs.

Recent research has advanced the state of the art in class D switching amplifiers for hi-quality audio. These have the advantage of high efficiency (up to 97%) and small size. To our knowledge, though, these have yet to be used with ESL panels, and custom step-up transformers will be required to provide a reasonable load to the amplifier and drive the panel very low hi-frequency impedance.

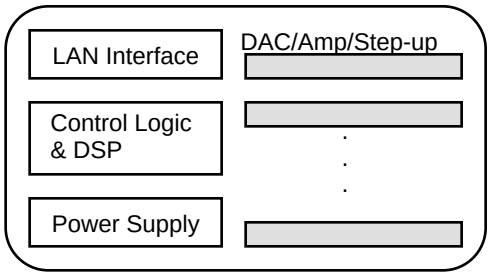

Figure 11: Interface/driver hardware system configuration in a simple card cage

Each DAC/amplifier/transformer board will measure approx. 4 * 8 inches (about the size of a PCI plug-in card) and will handle 8-16 channels. The components of the interface/amplifier system are shown in the figure above. The main chassis will support the interface and logic motherboard, and 4-16 output cards. Most of the components of this circuit are well understood and can be integrated off-the-shelf. The main question relates to the size of the output step-up transformer (i.e., how small can they be made).

Software and Computer Infrastructure

[NB: This section is incomplete, largely because the software described (CRAM, CSL, OSG, GestureSensors, DRIVE) has already been implemented and is in-use. The reader is referred to the web sites of these projects at CREATE and MAT.]

The CNSI's computational infrastructure consists of (1) a traditional vector supercomputer running (largely FORTRAN) numeric/simulation applications and using MPI software framework for parallel computation, (2) a large Linux cluster running (modern cluster-oriented) scientific application based on MPI and other cluster/grid application managers, and (3) the multimedia processing cluster (media IO farms) we're designing.

A typical multi-modal AlloSphere application will integrate services running on multiple hosts on the LAN that implement a distributed system composed of:

- input sensing (camera, sensor, microphone),
- gesture recognition/control mapping,
- interface to a remote (scientific, numerical, simulation, data mining) application,
- back-end processing (data/content accessing),
- output media mapping (visualization and/or sonification), and
- A/V rendering and projection management.

We're concentrating on the audio output in this discussion, but application developers will also use APIs or LAN-distributed application services in AlloSphere system integration.

Given the requirements described above, we decided (long ago) to develop the audio software infrastructure and application-level support ourselves using mainstream cross-platform languages and APIs. AlloSphere audio applications consist of a hierarchy of sound synthesis, spatialization, and output convolution servers, mostly written in C++ running on Macintosh or Linux servers.

Using the DSCP design pattern (see below), our systems use a 3-tier architecture that consists of sensing and gesture mapping, data mapping and sound synthesis, and spatial processing and output services, with separate application management tools, databases, and protocols. The spatialization/output software generally involves several processors running encoders or mapping routines, and a bank of output driver servers doing large numbers of FFTs on multi-channel sample streams.

As we note in App. 2, the general cases involve either large numbers of small buffers (fewer channels populated), or smaller numbers of large buffers (more channels populated).(We obviously try to avoid cases where the encoding algorithm produces many fully-populated buffer streams.) As discussed elsewhere, we estimate that the current design will require 2-8 GFLOPS of CPU power and 2-8 GBps of LAN streaming.

Application Development

We provide several kinds of APIs and standalone spatialization servers for use in building audio applications for the AlloSphere. Most applications end up using a mix of C++, Supercollider, and/or Smalltalk and PD. The CSL C++ library includes Doug McCoy's implementation of VBAP in 2 versions (separate classes) that present different APIs to the developer. A separate application (Ventriloquist) provides a GUI for recording and editing spatial trajectories, and playing multiple streams back through a given configuration loaded from a formatted file. Florian Hollerweger, Graham Wakefield, and Jorge Castellanos implemented CSL classes for simple and higher-order Ambisonic encoders and decoders; which can easily be distributed and stream samples over a LAN. There is also a good open-source version of Ambisonic coders in SuperCollider. Lastly, we are currently porting the Wonder wave field synthesis software (Baalman 2004) to CSL for our use.

To support the new generation of applications, many of our controllers and servers provide for OSC continuous control and position streaming, and CSL has been extended to have custom buffer classes for use in advanced spatialization systems (i.e., buffer + geometry object, or partially populated many-channel buffers).

We expect that extending these packages as needed for the AlloSphere (e.g., adding weighting maps and better geometric searching to VBAP, or porting open-source code for fast multipole methods to solve our 3D WFS equations) will be an exciting and challenging task.

Software for Running the AlloSphere

AlloSphere application software uses a set of object-oriented design patterns we call "distributed sensing, computation, and presentation" or DSCP service architecture (see the figure below). The component design patterns or application service groups are:

- back-end application models are scientific/numerical/simulation services;
- multimodal multiuser sensing/control and tracking/mapping farms;

Applications on Compute Servers	DSCP Infrastructure	I/O
Protein Structure Prediction	Media Streaming	Sensors (head-tracking, glove, suit)
Gene Sequence Mapping	Scheduling/Start-Stop	Other input devices
Sound Composition	Run-time Monitoring	Vision-based Input
User-Aware Spaces	Pluggable Sensor Framework	Many-channel Video/Audio Projection
	Visualization and Aural Rendering	

Figure 12: Software architecture of AlloSphere applications with end-user apps. at the left, DSCP middleware at the center, and sensing and rendering on the right

- application model = sensing/tracking policies + output data mappings;
- presentation/interaction via AlloSphere and LAN/WAN streaming; and
- Databases for configurations, resources, and media content (renderers).

Stand-alone AlloSphere audio applications are provided to interface with software sound synthesis packages such as SuperCollider, Pd, Max/MSP, and custom-written (CSL, Supercollider, and Max) synthesis and spatialization software running on a farm of 16-64 processors for rendering and streaming. The output mapping, convolution, and scheduling servers are currently written in CSL/C++ or SuperCollider.

It is yet to be determined just what kind of haptic and displayed interface will be presented to the user on the bridge, but there must be a simple front-end to running multimedia applications, interacting with stored presentations, and giving short demonstrations of the AlloSphere's capacity. We envision a front-end layer written in a GUI-centric rapid prototyping programming system such as Smalltalk,; this front-end will send messages to a CRAM system manager for starting apps, and to global control services for control of overall volume, etc.

In other documents, we have given mock-ups of the user interface for running the Allosphere that incorporates a control panel with a few knobs and sliders with a touch-sensitive LCD screen. I/O ports for USB, MIDI, and LAN hook-ups should also be provided on the bridge (outside of the scope of this document).

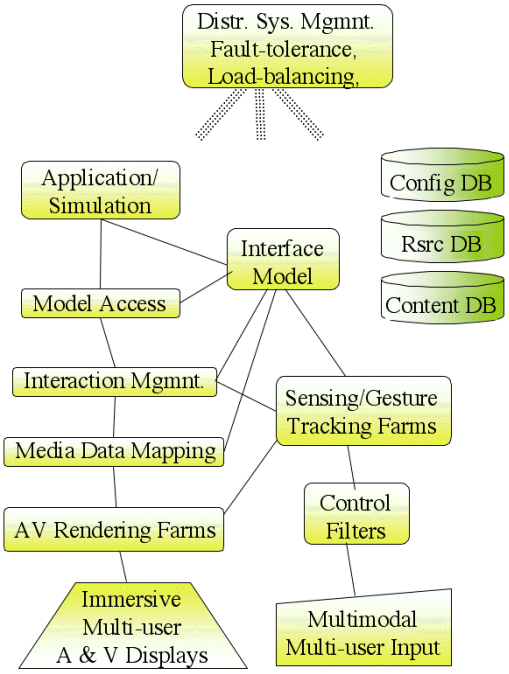

Figure 13: Distributed sensing, computation, and presentation (DSCP) application configuration and service intercommunication

Software Management

A distributed real-time software system such this requires an application management layer such as those found in grid computing or large-scale client-server applications. We use the CREATE real-time applications manager (CRAM, see Pope et al 2001b; Pope Engberg, and Holm 2001) distributed system management software (described elsewhere) for start/stop and monitoring of multi-host audio applications. CRAM uses several databases that describe how end-user applications are configured (what service runs on what machine), and what network resources are available.

Network Audio Streaming

There are two different protocols in use for LAN audio streaming: the CNMAT/IRCAM Spectral Data Interchange Format (SDIF), and the CSL RemoteFrameStream or RFS protocol. Both of these can be hosted on top of UDP/IP or TCP/IP, and drivers could be written for future networking standards such as IPv6, RTSP, or FireWire sharing protocols.

As discussed in App. 2, there are configurations of each of the spatialization methods we use that are CPU-bound, and others that are LAN-bound in terms of their scalability and multi-host distribution. We can easily project that a suitably complex system would require more than 1000BaseT-range LAN bandwidth.

There are several alternative network interfaces for streaming large-volume audio data over LANs. Yamaha's mLAN (http://www.yamaha.co.jp/tech/1394mLAN) is based on FireWire, while Gibson's "Media-accelerated Global Information Carrier" (MaGIC, http://www.gibsonmagic.com) and Axia's Livewire system (http://www.axiaaudio.com) use CAT-5 or CAT-6 cable. These networks use proprietary (licensed) protocols and provide APIs for application development; we believe they warrant further study and possible collaboration.

Lastly there is the question of how we are supposed to clock multiple output servers to provide sample synchronous output (as required by the spatialization software). Luckily, a group at UCSB has been working on this problem, and is able to achieve multi-host synchronization on the order of 100 nsec or better (i.e., smaller than the sample rate of high-resolution audio) (Butner and Vahey 2002).

Steps to Get There

Having outlined the requirements and then proposed a hardware/software design for audio in the UCSB CNSI AlloSphere, we can decompose the task list down to the point where we can estimate the level of effort (LOE) to implement each step of the system. We present a rough (simplified) task list below.

AlloSphere Audio Hardware

- Choice of driver elements; supplier contact
- Design of speaker suspension framework
- Placement process: compilation of "forbidden zones" from sphere structure supplier, geometry iteration steps
- Design of LAN/DAC/Amp hardware (based on UCB 120-channel system) mother card, daughter card, Xylinx firmware
- Sub-woofer specification and selection
- Sub-woofer mounting design
- Design of additional acoustical treatments for the bridge, doors, and seam
- Evolution of current test-bed lab (see App. 3) to AlloSphere in 2 stages over 12 months
- Design 16-channel circular microphone array

Software Development

- Port open-source 2-D WFS code to CSL
- Study 3-D WFS solutions
- Better geometric search for VBAP
- Integrate VBAP and CSL spatial reverberator
- Mixed- and higher-order Ambisonic coders
- Support app. developers in system integration
- CRAM interfaces for CSL apps.
- Application and server farm DBs for CRAM
- Audio input, spatial speaker ID

Challenges

If we rank the risks to this implementation plan by their cost, there are high-order risks in each of the main task areas. We outline them below.

- Funding and project management
- Mixed-mode spatial representation
- Controlling multiple sources in real-time
- AlloSphere run-time management, main menu
- Environmental control
- Recording and playback of "performances"
- Will 2.5-D WFS work?
- How do we handle the output transformers if we use ESL panels?
- How to the speaker elements themselves effect the T60 time of the room?

Conclusions

The UCSB AlloSphere has been designed to serve as the next-generation immersive multimedia computational interface. This document is a technical working paper for the audio component of the AlloSphere's multimedia interaction framework. We are looking for industry partners to contribute latest-generation (or perhaps yet unreleased) hardware and software with the benefit of receiving feedback and real-world usage data as well as exchanging IP in the area of emergent applications.

Acknowledgments

This white paper is based on contributions by Stephen Pope, Florian Hollerweger, Alexandre

Kouznetsov, Xavier Amatriain, Marcos Novak, Curtis Roads, and JoAnn Kuchera-Morin. We wish to acknowledge the contributions of the rest of the CREATE spatial sound team and MAT and IGERT students and faculty.

Several of the figures above were taken from the references, and others are used by permission of the CNSI's architects Auerbach, Pollock, and Friedlander.

References

Baalman, M. A. J. 2003. "Application of Wave Field Synthesis in Electronic Music and Sound Art." *Proc. 2003 International Computer Music Conference* (ICMC).

Baalman, M. A. J. 2005. "Updates of the WONDER software interface for using Wave Field Synthesis." *Proc. 3rd International Linux Audio Conference* (LAC2005). April 2005, Karlsruhe, Germany.

Jens Baluert. 2001. *Spatial Hearing*. MIT Press.

Carlile, S. 1996. *Virtual Auditory Space: Generation and Applications*. Chapman & Hall Publishers.

Berkout, A.J., D.de Vries, and P. Vogel. 1993. "Acoustic control by wave field synthesis." *J. Acoust. Soc. Am.* 93(5): 2764-2778.

Butner, S. E. and S. Vahey. 2002. "Nanosecond-scale Event Synchronization over Local-area Networks." *Proceedings of the 2002 IEEE Conference on Local Computer Networks* (LCN 2002), Tampa, Florida, Nov 2002, pp. 261-269.

Cooper, D.H. and T. Shiga, 1972. "Discrete-Matrix Multichannel Stereo." *J. Audio Eng. Soc.* (JAES), 20: 346-360.

Freed, A., 2004. *Bi-directional AES/EBU Digital Audio and Remote Power over a Single Cable.* CNMAT R&D Report. http://www.cnmat.berkeley.edu/AES1999/docs/wTriaxAE.pdf.

Freed, A. 2005. *Design of a 120-channel loudspeaker array*. Unpublished CNMAT report.

Freed, A. and D. Wessel, 1998. "Communication of Musical Gesture using the AES/EBU Digital Audio Standard," *Proc. 1998 ICMC.*

Gerzon, M. 1973. "Periphony: With-Height Sound Reproduction." *JAES* 21: 2-10.

Gumerov, N., and R. Duraiswami. 2004. *Tutorial: The FMM for 3D Helmholtz Equation*. http://www.cscamm.umd.edu/tutorials.

Gumerov, N., and R. Duraiswami. 2005. *Fast Multipole Methods for the Helmholtz Equation in Three Dimensions*. Elsevier Publ.

Hollerweger, F., 2005a. *An Introduction to Higher-Order Ambisonics.* CREATE working paper, April, 2005. http://create.ucsb.edu/wp/FH_HOA.pdf

Hollerweger, F., 2005b. *Designing a Periphonic Sound Spatialization Engine for the UCSB AlloSphere.* CREATE working paper, May, 2005. http://create.ucsb.edu/wp/FH_AlloEngine.pdf

Hollerweger, F., 2005c. *Sound Spatialization for the UCSB AlloSphere: Progress Report.* CREATE working paper, June, 2005. http://create.ucsb.edu/wp/FH_SphereAudio.pdf

Malham, D. G., and A. Myatt. 1995. "3-D Sound Spatialization using Ambisonic Techniques." *Computer Music Journal (CMJ)* 19(4): 58-70.

Martens, W. L., and W. Woszczyk. 2003. "Guidelines for Enhancing the Sense of Presence in Virtual Acoustic Environments" *Proc. 9th Int. Conf. on Virtual Systems and Multimedia*, pp. 306-313.

McCoy, D. 2004. *Ventriloquist: A Performance Interface For Real-Time Gesture-Controlled Music Spatialization.* Master's Degree Thesis, UCSB/MAT, 2004. http://mat.ucsb.edu/~d.mccoy/thesis/VentriloquistThesis.pdf

Pope, S. T. 2000. *The State of the Art in Sound Spatialization.* Presented at the SoundInSpace Symposium, UCSB, March, 2000. http://create.ucsb.edu/~stp/PostScript/SiS2k.slides.pdf

Pope, S. T. 2001. "Music and Sound Processing Using Siren." In M. Guzdial and K. Rose, eds. *Squeak: Open Personal Computing and Multimedia*. Prentice-Hall, pp. 377-421.

Pope, S. T., et al. 2001a. *Research on Spatial and Surround Sound at CREATE.* CREATE white paper, March, 2001. see http://create.ucsb.edu/wp.

Pope, S. T., et al. 2001b. "The Distributed Processing Environment for High-Performance Distributed Multimedia Applications." *Proc. 2001 IEEE Multimedia Technology and Applications Conference* (MTAC), U. C. Irvine.

Pope, S. T., A. Engberg, and F. Holm. 2001. "The Real-time (Multimedia) Interface Description Language: RIDL." *Proc. 2001 IEEE MTAC.*

Pope, S. T., and C. Ramakrishnan. 2003. "The CREATE Signal Library: Design, Issues, and Applications." *Proc. 2003 ICMC.*

Pulkki, V. 1997. “Virtual Sound Source Positioning Using Vector Base Amplitude Panning.” *JAES* 45(6): 456-466.

Pulkki, V. and T. Hirvonen 2005. “Localization of Virtual Sources in Multi-Channel Audio Reproduction.” *IEEE Transactions on Speech and Audio Processing* 13(1): 105-119.

Rabenstein, R., S. Spors, and P. Steffen. 2005. “Wave Field Synthesis Techniques for Spatial Sound Reproduction.” to appear in: E. Haensler, G. Schmidt, eds. *Selected methods of Acoustic Echo and Noise Control*. Springer Verlag.

Spors, S., H. Teutsch, and R. Rabenstein. 2002. “High-Quality Acoustic Rendering with Wave Field Synthesis.” in *Proc. Vision, Modeling, and Visualization Workshop*, pp. 101-108.

Teutsch, H., et al. 2003. “An integrated real-time system for immersive audio applications.” P*roc. 2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, NY.

Wright, M., et al. 1999. *Audio Applications of the Sound Description Interchange Format Standard.* CNMAT R&D Report. See http://cnmat.berkeley.edu/SDIF.

Wright, M., A. Freed, and A. Momeni. 2003, “OpenSoundControl: State of the Art 2003.” *Proc 2003 International Conference on New Interfaces for Musical Expression*, Montreal, Canada pp. 153-160.

Appendices

App. 1: Reproduction of Spatial Sound

We use the spatial localization of sound sources every day to help us cope with the complexity of our sonic worlds, for example, it helps us differentiate between road noise and the sound of our car radio when driving, and it helps us disambiguate the voices of several speakers in a room (the so-called “cocktail party effect”) (Blauert 2001).

The history of sound recording and playback has seen continuous progress in the area of spatial sound, beginning with the monophonic format, followed in the third quarter of the last century by stereophonic records and later stereophonic broadcasting. The 1970s saw the development and promulgation of several consumer-oriented four-channel formats (all now defunct), followed by multiple formats for 5.1-channel surround sound, first in theaters and then (in the 1990s) in homes. More recently, many theaters have begun the move to 10.2 or more playback channels, and 7.1-channel surround sound is gaining popularity as a consumer format. The largest system known to the authors is a 198-channel cinema in Ilmenau, Germany (www.iosono-sound.com/cinema.html) built for large-scale wave field synthesis (see figure).

Figure 14: 198-channel cinema in Ilmenau, Germany; along the walls are the closely spaced loudspeakers for wavefield synthesis.

In parallel with these developments, composers and performers of electroacoustic music have been using many-channel playback systems since the 1950s, and eight or more playback channels are commonly found in festivals of contemporary art music. As examples of this development, the figures below show an installation of the UCSB Creatophone for a concert in the year 2000 (or so) in which 15 channels are used to create the frontal stereophonic spatial image alone, and, below that, an “orchestra of loudspeakers” installed at the French radio studio on Paris in 1995.

In another application domain, researchers in the field of auditory display have used sound spatialization as an important property in “data sonification” (so-named to explicitly imply the parallel

to "visualization") systems starting in the 1970s. It is for both the scientific and the artistic applications that we are planning high-spatial-fidelity sound playback for the UCSB AlloSphere.

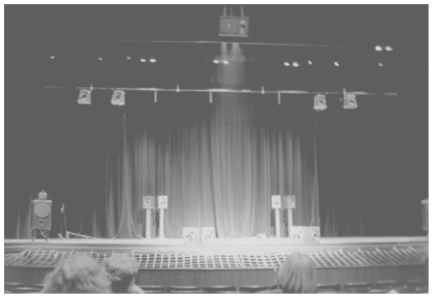

Figure 15: Creatophone installation (ca. 2000) in UCSB's Lotte Lehmann Hall, 15 channels are used for the frontal spatial image

In theory, to synthesize a spatial sound field, one requires a monophonic sound stream and a source location (possibly changing over time for the case of moving sound sources). Some systems are also able to simulate room acoustics, which means that they require a room model (i.e., the geometry of the listening space to be simulated) and the geometrical position of the listener (Carlile 1996; Martens and Woszczyk 2003).

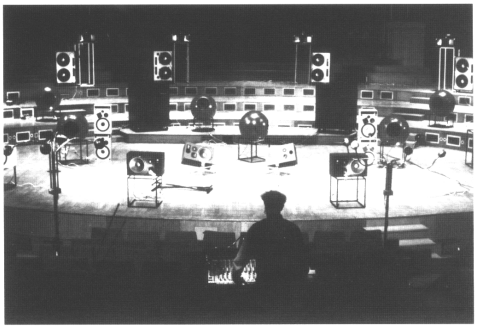

Figure 16: "Orchestra of loudspeakers" in concert at the French radio studio, Paris, 1995.

As we mentioned above, there are three methods in common use today that can each take this kind of information and render a spatial sound field. Each of these techniques has certain applications where they excel, and others that pose greater problems. The three techniques we'll present are:

- vector-based amplitude panning (VBAP);
- the ambisonic representation; and
- wave field modeling and synthesis.

(For the purposes of this discussion, we will ignore the techniques that are primarily aimed at playback over headphones, such as the use of the head-related transfer function or HRTF.) We will discuss each of these techniques in sufficient detail to serve as the basis for our design discussion.

Vector-based Amplitude Panning

As the simplest case of spatialized sound, we are familiar with stereophonic sound panning, where perceived position of a sound source between the left and right speakers is determined by the balance of the signal sent to each speaker. Figure 16 below shows the geometry for this case. Given a desired source position (angle Θ), one can apply an arbitrary cross-fade weighting function to derive the factors for the 2 output channels. Note that the assumption is that the speakers are equidistant from the listener, and that panning only allows us to move the source position along the arc between the speakers (i.e., source distance must be simulated by volume and reverberation cues).

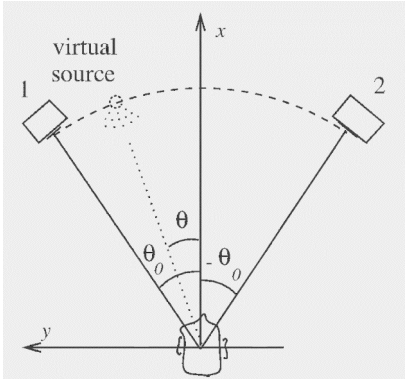

Figure 17: The geometry of stereophonic panning with a virtual source shown at angle Θ

If we extend this idea to a larger number of speakers and to the 3-dimensional case, we end up with a system in which the speakers are distributed on the surface of a sphere centered at the listener's position. The geometry for this looks like the simplified representation shown in the next figure (see Pulkki 1997). For any desired source position (shown as "*p*" in the Figure), the software simply needs to determine which three speakers define the smallest triangle that includes *p*, and what the contribution of energy from each of these will be to simulate a source at position *p*. These gain factors can be cached, as they only change if the source moves.

Practical VBAP systems (e.g., Pulkki and Hirvonen 2005) allow interactive performance with multiple moving sound sources, which are mapped and played back over medium-scale projection systems. VBAP has been mainly promulgated by R&D groups in Finland and France and is used effectively in 8-32-channel CAVE virtual environments.

The drawbacks of VBAP are that it does not directly answer the question of how to handle distance cues (relatively easy to solve for distant sources and low Doppler shift), and that it provides no spatialization model for simulating sound sources *inside* the sphere of loudspeakers. This is a grave problem for our applications, but also a worthy topic for our research. The question boils down to how to spread a source over more than 3 speakers without limiting the source position to the edges of the surface described by the chosen set of speakers (i.e., having it "collapse" to the edge of a speaker surface).

Software Implementation

The VBAP algorithm involves a search among a database of the geometrical speaker triangles that define the playback configuration, and then some simple matrix math to calculate the relative gains of each of the three chosen speakers. There are several open-source implementations of VBAP that support multiple sources (with some interactive control over their positions), and flexible speaker configurations up to about 32 channels.

One of our graduate students (Doug McCoy) implemented an system in which the user can move and direct a number of independent sound sources using a data glove input device, and play back sound files or streaming sound sources through VBAP, playing out to a variable number of loudspeakers (the configuration is read in at start-up) (McCoy 2004). This system was developed using the CREATE Signal Library (CSL) programming framework in C++.

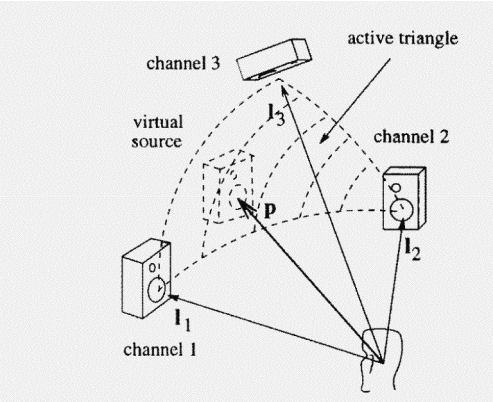

Figure 18: Geometry of vector-based amplitude panning (VBAP) where a source at position p is distributed to at most 3 speakers

VBAP can be integrated with a spatial reverberator software, allowing early reflections from a reverberator to be individually panned, though this gets computationally very expensive with many sources, complex room simulations, or rapid source (or listener) motion.

Scalability and Distribution

Because VBAP is so simple, most implementations are monolithic 1-piece packages. This is obviously unacceptable for our purposes, so we need to consider both (1) how the VBAP system scales to large numbers of sources, rapid source motion, and many output channels, and (2) how such a scaled-up application can best be distributed to a peer-to-peer server topology streaming data over a high-speed LAN.

The scalability of VBAP encoding software is excellent, since the block-by-block processing is very simple, and the computation of new output weights for new or moving sources can be accelerated using well-understood geometrical search techniques. For the case of many sources or rapid source or listener motion, VBAP scales linearly, because each source is encoded into 3 channels, meaning that many mappers each write 3 channels into a many-channel output buffer. Alternatively, if the servers are distributed, each mapper sends 3 channels over the LAN to its output server. If the output servers are themselves distributed (each taking over a subset of the sphere's surface), then most encoding servers will stream to a single output server.

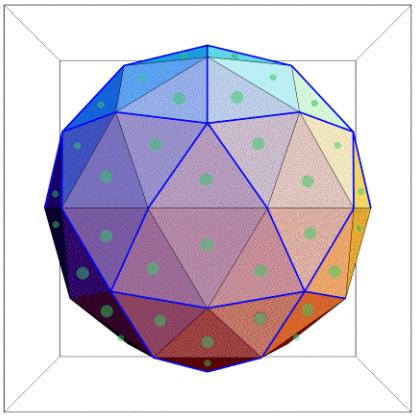

Figure 19: Uniform 80-way tesselation of a sphere (visualized in Mathematica)

A VBAP-based spatial reverberator is more difficult to distribute, since by definition the individual reflections are not localized to a small number of channels; indeed, if you calculate a reasonable number of reflections (e.g., 64 or more) for a complex room model, you can assume that the reflections will approximate an even distribution among all channels, leading us back to a monolithic output server topology. We look forward to attacking

this scalability and partitioning issue in the real system.

Loudspeaker Configuration

The assumptions of the speaker elements and system configuration for playing VBAP are that the elements be identical full-range speakers, and that they be placed in triangles of more-or-less equal size in all directions. The speaker density can be made a function of height, however, leading to somewhat poorer spatialization accuracy above (and possibly below) the listener.

All that being said, since VBAP makes so few assumptions about the constructed wave, it supports non-uniform speaker distributions quite well. Directional weighting functions to compensate for an uneven distribution of speakers can be built into the VBAP amplitude matrix calculations, and the fidelity of spatial impression is a directional function of both the speaker density and regularity of spacing.

In our earliest designs for the sphere, we ran a set of programs to tessellate spherical surfaces, leading to the 80-channel configuration shown in the figure above. Note the two regular rings above and below the equator; one can rotate the upper hemisphere by 1/2 the side length to form a zig-zag pattern here (which handles VBAP better) Continuing this process (as described in the next Appendix), we can design and evaluate further regular subdivisions of a sphere, as illustrated in the configuration shown in the figure below.

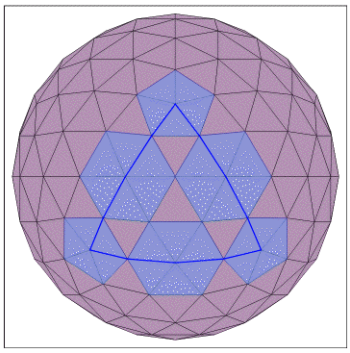

Figure 20: Finer-grained spherical distribution

The Ambisonic Representation

Like VBAP, the ambisonic model (Malham and Myatt 1995) also has its roots in the earliest days of stereophony. Soon after engineers started recording in stereo using microphone pairs, it became clear that there were problems related to controlling the "width" of stereo recordings (i.e., increasing or decreasing the difference between the left and right channels in a naturally recorded signal).

To overcome this, the "mid/side" (M/S) recording technique was developed. An M/S stereo recording has two channels, but, rather than corresponding to the left and right channels, they represent the sum and difference signals, i.e., one monophonic signal (the "mid"), and one that captures the left/right difference (the "side").

M/S stereo can be recorded live using a central directional microphone for the mid signal, and a bidirectional "figure-8" microphone to capture the L/R side signal (see Figure, which shows the two overlapping directional responses of the microphones used in M/S recording). The side signal can be added to or subtracted from the mid signal to derive the left and right signals, or can be mixed in variable amounts to give a control over the width of the stereo field. Several manufacturers produce analog signal processors that take M/S-encoded stereo and mix it to typical L/R format with control over the stereo field width.

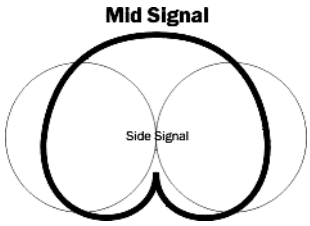

Figure 21: Mid/Side (M/S) stereo microphone (polar) directional response diagrams

In the early 1970s, two groups in England independently developed the idea of extending this model to 3-dimensional representations, and of using spherical harmonics (a notation taken from geometry) to describe the spatial weighting of the various signals that constitute the multi-channel signal (see Cooper and Shiga 1972 and Gerzon 1973). In the simplest ambisonic representation (called B-format), four channels represent the monophonic signal and the difference signals in each of the three spatial dimensions (x, y, and z). These directional patterns are shown in the figure below (see also http://www.ambisonic.net).

One can think of this as a simple extension of the M/S technique to three dimensions. A sound source's position can be captured in terms of its contribution to the monophonic signal and each of the difference signals. As with M/S encoding, a custom signal processor/mixer can take a 4-channel B-format ambisonic recording and decode it

for various speaker configurations (including stereo, [2-D or 3-D] quadrophonic, octophonic, 5.1-channel surround, etc.).

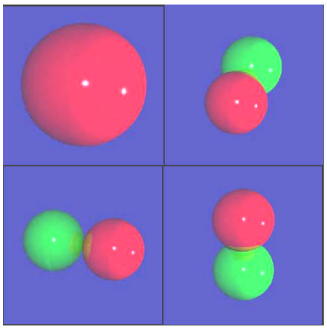

Figure 22: The first-order spherical harmonics as used in the ambisonic representation.

This representation can be extended for higher spatial fidelity simply by adding more channels using the higher-order spherical harmonics (see Hollerweger 2005a). This results in 9- or 16-channel representations (for the 2nd- and 3rd-orders, respectively) that can be decoded to give excellent spatialization with very large speaker arrays (e.g., a decoder that takes the 9-channel 2nd-order format and produces 32-channel output). The figure below shows the surfaces that describe the directionality of the five channels of a 2nd-order system; below that, you can see the directional patterns for several of the 10th-order harmonics, along with their geometrical classifications.

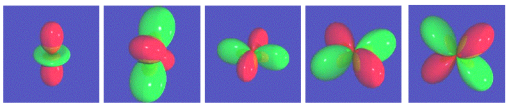

Figure 23: The 2nd-order spherical harmonics

The general relationship between harmonic order, encoding, and meaningful decoding is

$$L >> N = (M + 1)^2$$

where:

- M = Ambisonic order
- N = number of channels in *M*th-order Ambisonic representation
- L = number of decoder output channels

Software Implementation

To implement the Ambisonic representation, one needs to write an encoder that takes a monophonic signal and a virtual source position, and generates a fixed-format multi-channel signal whose format and size depend on the order of encoding chosen. the mathematics for this involves relatively simple trigonometric operations on the input signal buffers. The decoder takes the encoded n-channel signal and a list of output speaker positions. For each speaker, it uses complementary trigonometric equations to determine the contribution of each harmonic signal to the output at the given speaker's location.

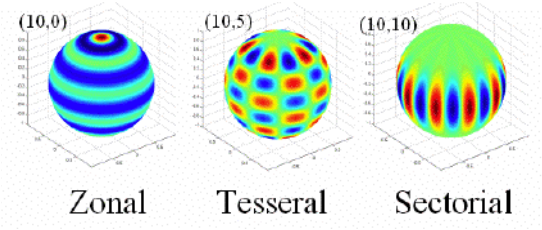

Figure 24: Examples of several 10th-order spherical harmonics showing the detail of the spatial encoding possible

As with VBAP, MAT graduate students (Graham Wakefield and Jorge Castellanos with visiting research associate Florian Hollerweger) have implemented higher- (up to 11th-) order ambisonic processing and decoding in C++ using the CSL framework. The encoder and decoder are separate classes, and utility classes exist for processing (e.g., rotating the axes of) Ambisonic-encoded sound. There are also open-source implementations in both SuperCollider and PD.

Scalability and Distribution

Ambisonic encoders and decoders are all relatively simple, and can be decoupled from one another. For a simple scaled-up system, multiple 3rd-order encoders would run on machines in the server farm, each of them streaming a 16-channel signal to the output driver(s). These signal busses can be summed and then distributed to one or more output decoders. the scalability to higher orders is well understood, and scales with the number of channels required by the representation.

One of the main benefits of the Ambisonic representation is that it scales very well for large numbers of moving sources, because the encoding is based on the order of the representation used. The decoding scales well to large numbers of speakers because decoders are independent of one another, each receiving the same set of inputs; there are no obvious scalability limits, either in terms of CPU processing or LAN bandwidth requirements.

Loudspeaker Configuration

Ambisonic decoders work best with a regular and symmetrical loudspeaker configuration; software and hardware decoders for 2, 4, 8, etc. channels are readily available. There is no way in the

processing algorithms to compensate for irregular speaker placement. What's interesting is the fact that very large speaker arrays can especially benefit from higher-order ambisonic processing, using ever-higher orders of spherical harmonics to encode the sound field, and then decoding it using these factors to play out over a (regular and symmetrical) many-channel speaker array. The figure below shows the Birmingham ElectroAcoustic Sound Theatre or BEAST system, which uses a circular speaker layout with an additional 48-channel overhead tweeter array.

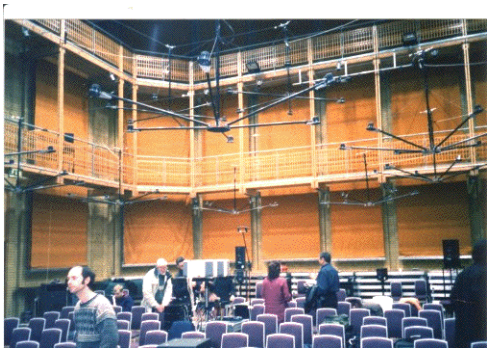

Figure 25: Birmingham, UK BEAST system with its 48-channel overhead speaker array and circular base projection system

The next figure shows the listening room at the Institute of Electronic Music and Acoustics (of the Univ. of Music and Dramatic Arts) in Graz, Austria, an example of a symmetrical 32-channel hemisphere playback system that uses Ambisonic decoders written in PD running on Linux servers.

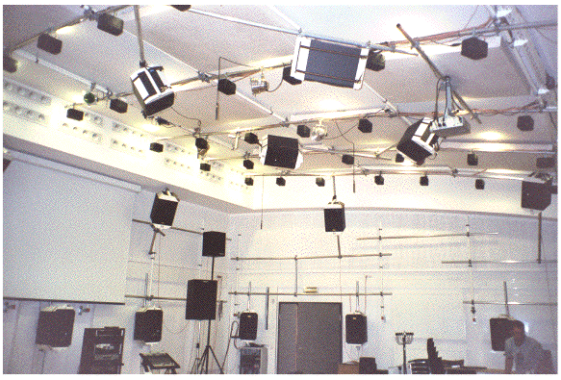

Figure 26: 32-channel hemispherical playback system at the Institute of Electronic Music and Acoustics of Graz, Austria

Wave Field Synthesis

Wave field synthesis (WFS) is a sound recording and playback technique based on the physical principle of superposition (taken from optics). In WFS, the sound of a positional source is simulated by a dense array of loudspeakers summed together in space (i.e., they are superimposed) to create the wave front, as shown in the figure below.

In the figure below, the sound of the imaginary source (placed a bit above the boundary of the speaker array) is simulated by the signals emanating from the array of speakers around the listening area in the middle of the figure. By controlling the signal sent to each of these speakers (primarily using delays and amplitude scaling), virtual source positions, and even moving sources, can be simulated (Berkout, de Vries, and Vogel 1993; Rabenstein, Spors, and Steffen 2005; Spors, Teutsch, and Rabenstein 2002; Teutsch et al. 2003).

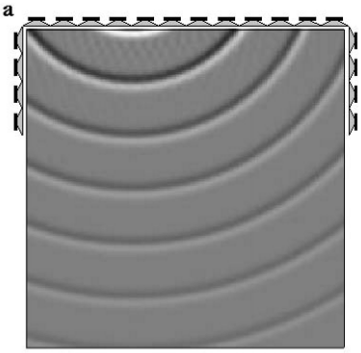

Figure 27: Wave field synthesis: superposition of the signals from an array of loudspeakers

WFS playback spaces generally involve a circle or square of 32-200 loudspeakers. Due to the speed of sound, one can calculate the speaker spacing that will be required to reconstruct wave fronts of a given upper limit frequency. In the general literature, to achieve good performance up to 1 kHz, the speakers should be spaced around a circle at distances of half of the wavelength of 1 kHz, or about 6 inches apart. The next figure shows a small-scale 24-channel WFS playback system; the figure below that is a circular 48-channel system; both are at the Univ. of Erlangen-Nuremberg, Germany.

Figure 28: Small-scale wave field synthesis playback configuration

There is a corresponding wave field recording technique that uses microphone arrays, and the synthesis of wave field signals from monophonic streams and geometry data is well-understood.

If you think of WFS as a 2-dimensional field with cylindrical wave-fronts in 3-D space, you quickly notice that low-frequency compensation will be required, and that distance cues will be distorted and must be exaggerated.

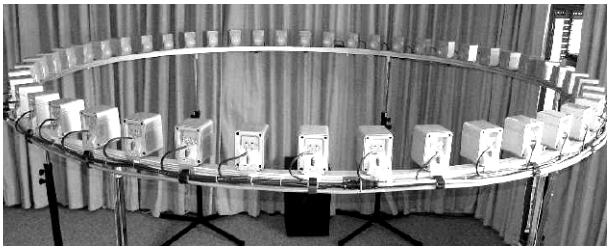

Figure 29: 48-channel circular WFS system

Software Implementation

Synthesis or processing of wave field signals involves solving an equation called the Kirchhoff-Helmholtz integral[1], which is done using sophisticated (and computationally costly) differential equation approximation techniques. There have been several recent advances from the R&D groups in Holland and Germany that make this possible, for example, up to 128 channels can be decoded on a current-model PC dual-processor server.

There are existing open-source implementations of WFS available to us (e.g., Baalman 2003), and we are in the process of porting these to work within the CSL framework.

As alluded to above, WFS also requires compensation for speaker characteristics and room effects, but this process is well understood and computationally tractable (adding a level of FIR filters to the large convolutions involved in the implementation).

Scalability and Distribution

Basic WFS processing for single sources and medium-range numbers of output channels can be implemented on a single processor. The scalability for the case of multiple sources is thought to be poor, requiring multiple processing servers all of whose output channels are mixed. Similarly, nobody has ever tried to partition a WFS server onto multiple hosts.

The processing of WFS math for large 2-D and 3-D systems is still a complex problem requiring efficient solutions to a set of matrix equations using techniques referred to as fast multipole methods (FMM), which is an active are of research in computational mathematics (Gumerov and Duraiswami 2004, 2005). Implementing a large-scale (> 200 speakers) 2-D system is straightforward, though we have yet to investigate the scalability and distribution issues. The development of distributed FMM-based 3-D solutions is to be a MAT thesis project in the near future.

Loudspeaker Configuration

The requirements on the loudspeaker configuration for WFS are simple; one needs the densest-possible packing of speakers around the circumference of a circle, preferably less than half of the wavelength of the lowest frequency in the signal. The limits on speaker spacing for WFS are simple and physical. It is generally assumed that we do not perceive the spatialization for sounds below about 200 Hz, since the wavelength is so much larger than the distance between our ears. In order to get any subtlety in the spatial impression generated using WFS, we need the speakers to be spaced at intervals on the same order as the longest wavelength we desire to reconstruct with spatial accuracy. These two facts give us upper and lower bounds on the speaker spacing interval; if we want to have at least a few octaves where we get accurate spatial wavefront reconstruction, speaker spacing on the order of 1 foot or less is a requirement.

Figure 30: An array of 8-channel multi-actuator panels for wave field synthesis

1. For those interested, it reads,

$$P_A = \frac{1}{4\pi}\oint_S \left[\left(P\frac{1+jk\Delta r}{\Delta r}cos\phi\frac{exp(-jk\Delta r)}{\Delta r}\right) + \left(j\omega\rho_0 V_n\frac{exp(-jk\Delta r)}{\Delta r}\right)\right] dS$$

Recent research has lead to the development of speaker panels with multi-actuator arrays driving them. The figure ablve shows a system using a string of four 8-actuator panels in a portable 32-channel WFS system.

App. 2: Speaker Count and Placement

Given the requirement that the AlloSphere provide high-fidelity spatialization of a large number of sources over a large region near the center of the space, and that it support (at a minimum) all three of the spatialization techniques introduced above, we can proceed to derive the number of channels and loudspeakers that will be required. The reference (Hollerweger 2005b) presents this discussion in more detail.

The AlloSphere's geometry is that of two hemispheres pulled slightly apart, as shown in the figure below. The loudspeaker configuration will approximate a sphere circumscribed around the projection screen. This means that it would be optimal for the speakers to be set back behind the surface near to the seam between the 2 hemispheres, and to be closer to the surface at the midpoint of each hemispherical section. The compromises will be with the AlloSphere's own mounting framework, the video projectors around the sphere's seam, and the spacing of the surrounding acoustical foam. The basic measurements and geometry are summarized below.

Effective radius of 16 ft. (4.75 m)
Effective circumference 115 ft. (35 m)
Effective surface area 3920 ft^2 (364.25 m^2)

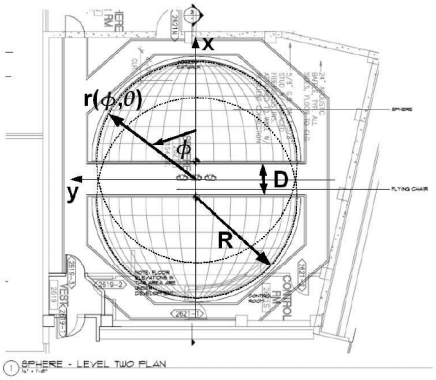

Figure 31: AlloSphere geometry

We designed a process and evaluation framework for comparison of speaker configurations for VBAP, Ambisonics, and WFS (see figure below). It allows us to weigh the compromises implicit in various layout solutions. The goal was to approximate a uniform and symmetrical configuration and express it as an optimal speaker density as a function of the Z (vertical) axis. The design process involves coming up with a desired configuration, and then taking into account the limits of the actual space (expressed geometrically as "forbidden areas" and "forced positions"). In the quality evaluation stage, the modified placement list is checked against the spacing and symmetry requirements of VBAP, Ambisonic, and WFS methods to yield the quality evaluation.

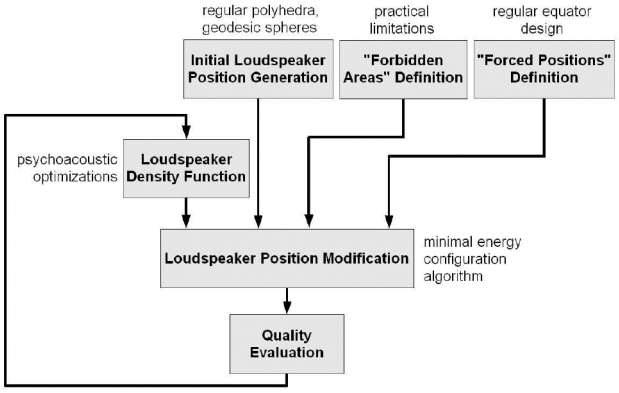

Figure 32: Loudspeaker layout design process

For VBAP, equal and symmetrical spacing is somewhat preferred, and horizontal resolution on the order of 10 degrees is required. (In some designs, the spacing is tighter near the equator, where our spatial perception is most acute.) For the AlloSphere's dimensions, this reduces to a density of approx. 1 speaker per square meter, leading us to an initial estimate that 364 channels (equally distributed on 1 m centers) would suffice for hi-fidelity VBAP playback.

There are few systems optimized for higher-order ambisonic decoding and projection. Medium-scale systems built to-date (mostly in England and Austria) generally consist of hemispheres (upper half only) with between 12 and 32 speakers arranged in concentric rings (see Figures 25 and 26 above). Using higher-order Ambisonics as we plan, it is estimated that speaker counts of 256 and higher would be effective for up to 11th-order representations.

According to the standard wave field synthesis design criteria (equal spacing at half of the shortest wavelength of interest, meaning 6-inch centers), it would require 9100 channels to construct a 3-dimensional WFS system with imaging accuracy above 1 kHz in the AlloSphere. As a compromise, we envision a pair of dense 2-D rings

running just above and below the equator, and lower density rings above/below these (see the next figure). This solution will allow us to experiment with 3-D solutions of the Kirchhoff-Helmholtz integral (using fast multipole methods) without needing fully 3-D WFS density. We hope that a future system will support fully 3D WFS in the AlloSphere.

By merging all these considerations, we arrive at a configuration that combines close spacing along the equator (2.5-dimensional WFS configuration) with semi-regular spacing above and below. The speaker density function is not necessarily symmetrical in the vertical dimension (i.e, denser above the equator than below it).

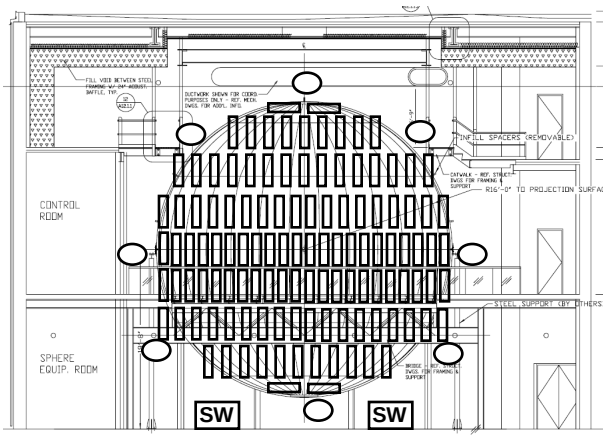

Figure 33: AlloSphere speaker layout (side view) showing dense rings of transducers (narrow rectangles), DAC/distribution amplifiers (ovals), and sub-woofers (blocks)

The two main rings should have 128-150 speakers, about 1-foot spacing, with monotonically lower density above and below. Depending on the exact dimensions of the ESL panels used (if they are all to be the same size at all), we can project a second ring with 80 speakers above and 64 below, and third rings of 40 and 32 elements, respectively. The remaining top and bottom circles can be covered with approx. 16 panels each.

To arrive at a precise speaker placement design, we must consult with the architects again after the finalization of the AlloSphere's support structure (expected very soon), then compile new lists of forbidden and recommended speaker positions. Based on the mounting constraints and speaker element geometry, we can then proceed with the placement design process.

App. 3: Current Development Test-bed

The current development test-bed at the MAT Lab at UCSB consists of a switched Gigabit Ethernet LAN with a variety of servers handling input and gesture mapping, signal synthesis and processing, output spatialization and reverb, and projection. The figure below shows schematically what's involved. The LAN is streaming both OSC/UDP and CSL/RFS protocols.

The input services are generally written in low-level C running on MS-Windows or Macintosh computers. They take gesture input from our sensors (EBeam, Matrix, OverTone Keyboard, FOBirds, DGlove, P5, etc. see http://mat.ucsb.edu/594O) and send Open Sound Control (OSC) messages out over the network to one or more clients. We use the CREATE Signal Library (CSL) C++ software library to write the mapping/synthesis/spatialization applications (SuperCollider and PD are also supported).

The output servers incorporate the software implementations of VBAP, Ambisonics, and WFS for pluriphonic playback, and run on a Macintosh G5 computer. The 18-channel speaker array is built from powered Mackie studio monitors in 3 rings (4/8/4) with one top channel and a sub-woofer.

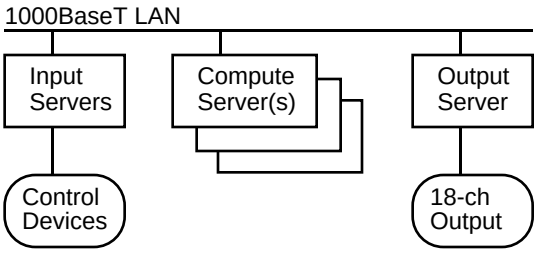

Figure 34: Current (Summer, 2005) development test-bed

For the next revision, we expect to move to a 64-channel system using two output servers.

App. 4: Unique Aspects of ESL Panels

We mentioned above that electrostatic loudspeaker panels have a couple of unusual electrical: and acoustical properties; first, they have very low impedance at high frequencies, which places special demands on the amplifiers driving them. The figure below (taken from the *Stereophile* magazine review of the MartinLogan Aerius, see http://stereophile.com/loudspeakerreviews/312) shows that the absolute impedance (the solid line, scale to the left in Ohms) falls below 2 Ω above 10 kHz (lower right of the figure). Luckily, the phase

response (dashed line, scale in degrees on the right) is near 0 degrees and well-behaved.

Note that this does not mean that it's inherently difficult to drive these speakers, just that the amplifier used must serve as a good current source. Audio amplifiers are typically measured with 8 Ω loads, the rated output power being the level that delivers 1% total harmonic distortion (THD) into a (simulated or real) 8 Ω load. For the ideal amplifier, this power would double into half the load, meaning that a nominally 100 W amplifier would deliver 200 W into 4 Ω and 400 W into a 2 Ω load. In many cases, audio amplifiers have neither the power supply nor the heat sinks necessary to scale to low load impedances, but on the other hand, several audiophile manufacturers (e.g., Krell and Mark Levinson) pride themselves on this, and even specify their amplifiers' performance driving 1 Ω loads.

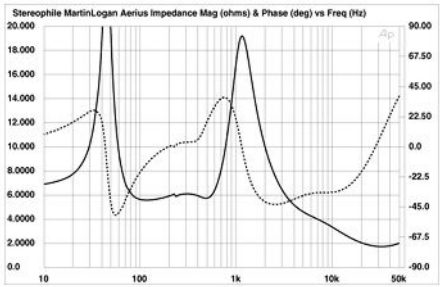

Figure 35: Electrical impedance and phase of a representative ESL speaker (with cross-over), note the low impedance at high frequencies

For our application, the bipolar (figure-8-shaped) radiation pattern of ESL patterns is good news; it reduces the extraneous acoustical energy within the AlloSphere. The plot in the figure below (from the same review as the previous one) shows how the speaker deviates from its nominal (flat) frequency response as you move laterally (to the right or left); each slice of the plot represents a different angle (see the legend at the right of the figure); note the steep fall-off of the spectrum for angles that are not right in front of (or behind) the ESL panel.

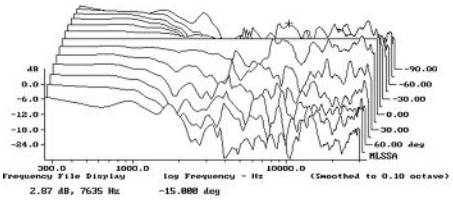

Figure 36: Lateral radiation pattern plots, showing the on-axis response in the middle slice, and off-axis responses around it

Contact Information

Stephen Travis Pope
Email: stp@mat.ucsb.edu