Introducing D$: An Interactive 3D Audio Rapid Prototyping and Transportable Rendering Environment Using High Density Loudspeaker Arrays

Ivica Ico Bukvic
Virginia Tech
SOPA, DISIS, ICAT
ico@vt.edu

ABSTRACT

With a growing number of multimedia venues and research spaces equipped with High Density Loudspeaker Arrays, there is a need for an integrative 3D audio spatialization system that offers both a scalable spatialization algorithm and a battery of supporting rapid prototyping tools for time-based editing, rendering, and interactive, low-latency manipulation. D$ library aims to assist this newfound whitespace by introducing a Layer Based Amplitude Panning algorithm and a collection of rapid prototyping tools for the 3D time-based audio spatialization and data sonification. The ensuing ecosystem is designed to be trans- portable and scalable. It supports a broad array of con- figurations, from monophonic to as many as hardware can handle. D$’s rapid prototyping tools leverage ocularcen- tric strategies and spatially rendering multi-dimensional data and offer an array of new approaches to time-based spatial parameter manipulation and representation. The following paper presents unique affordances of D$’s rapid prototyping.

1. INTRODUCTION

The history of Western music can be seen as a series of milestones by which the human society has emanci- pated various dimensions of aural perception. Starting with pitch and rhythm as fundamental dimensions, and moving on to their derivatives, such as homophony, and polyphony, every component was refined until its level of importance matched that of other already emancipated dimensions. In this paper the author posits that the observed maturity or the emancipation of these dimensions is reflected in their ability to carry structural importance within a musi- cal composition. For instance, a pitch manipulation could become a motive, or a phrase that is further developed and varied and whose permanence independently drive the structural development. The same structural importance can be also translated into research contexts where a sig- nificant component of the data sonification if not its en- tirety can be conveyed under an emancipated dimension. With the aforesaid definition in mind, even though timbre plays an important role in the development of the Western music, particularly the orchestra, its entirety use as a struc- tural element does not occur until the 20th century. Indeed, the 20th century can be seen as the emancipation of tim- bre. Similarly, while the audio spatialization has played a role throughout the history of music, with occasional spikes in its importance, including the Venetian cori spezz- zati [1] or the spatial interplay among the orchestral choirs, its structural utilization is a relatively recent phenomena. Today, the last remaining dimension of the human aural perception yet to undergo its emancipation is spatialization. From augmented (AR) and virtual reality (VR), and other on- and in-ear implementations, to a growing num- ber of venues supporting High Density Loudspeaker Ar- raies (HDLAs), 21st century is poised to bring the same level of emancipation to the spatialization as the 20th cen- tury did to timbre. Similarly, data sonification and soni- cation using primarily spatial dimension are relatively new but nonetheless interesting research areas whose full potential is yet to be realized [2].

In this paper HDLAs are defined as loudspeaker configu- rations of 24+ loudspeakers capable of rendering 3D sound without having to rely solely on virtual sources or post- processing techniques. This definition suggests there are multiple layers of loudspeakers spread around the listen- ing area’s perimeter.

Apart from the ubiquitous amplitude panning [3], con- temporary audio spatialization algorithms include Ambi- bisonics [4], Head Related Transfer Function (HRTF) [5], Vector Based Amplitude Panning (VBAP) [6], Depth Based Amplitude Panning (DBAP) [7], Manifold-Interface Amplitude Panning (MIAP) [8], and Wave Field Synthesis (WFS) [9].

There is a growing number of tools that leverage the aforesaid algorithms. This is of particular interest because the lack of such tools makes it particularly cumbersome to integrate algorithms, and whose permutations independently drive the structural development. The same structural importance can be also translated into research contexts where a sig- nificant component of the data sonification if not its entire- tity can be conveyed under an emancipated dimension. With the aforesaid definition in mind, even though timbre plays an important role in the development of the Western music, particularly the orchestra, its entirety use as a struc- 

desirable features an algorithm coupled with time-editing tools ought to deliver in order to foster a more widespread adaption and with it standardization:

• The support for irregular High Density Loudspeaker Arrays;
• Focus on the ground truth with minimal amount of idiosyncrasies;
• Leveraging the vantage point to promote data compre- hension;
• Optimized, lean, scalable, and accessible, and
• Ease of use and integration through supporting rapid prototyping time-based tools.

D$ is a new Max [10] spatialization library that aims to address the aforesaid whitespace by:

1. Introducing a new lean, transportable, and scalable Layer Based Amplitude Panning (LBAP) audio spa- tialization algorithm capable of scaling from mono- phonic to HDLAs environments, with particular focus on advanced perimeter-based spatial manipulations of sound that may prove particularly useful in arts- tic, as well as auralization and sonification scenarios, and
2. Providing a collection of supporting rapid prototyp- ing time-based tools that leverage the newfound audio spatialization algorithm and enable users to effi- ciently design and deploy complex spatial audio im- ages.

D$’s Layer Based Amplitude Panning (LBAP) algorithm groups speakers according to their horizontal layer and cal- culates point sources using the following series of equa- tions applied to the four nearest neighbors:

Below layer:

\[
B_{\text{amp}} = \cos(2\pi \frac{\text{BL}_{\text{distance}}}{\text{BL}_{\text{radius}}} + \text{BL}_{\text{azim}}) \times \cos(2\pi \frac{\text{BL}_{\text{elev}}}{\text{BL}_{\text{radius}}} + \text{BL}_{\text{azim}}) (1)
\]

Above layer:

\[
A_{\text{amp}} = \cos(2\pi \frac{\text{AL}_{\text{distance}}}{\text{AL}_{\text{radius}}} + \text{AL}_{\text{azim}}) \times \cos(2\pi \frac{\text{AL}_{\text{elev}}}{\text{AL}_{\text{radius}}} + \text{AL}_{\text{azim}}) (3)
\]

In the aforesaid equations B stands for the nearest layer below the point source’s elevation and A the nearest layer above. BL stands for the nearest left speaker on the be- low layer, BR for nearest right speaker on the below layer, AL for the nearest left speaker on the above layer, and AR for the nearest right speaker on the above layer. Amp refers to the amplitude expressed as a decimal value between 0 and 1. Distance reflects the normalized distance between two neighboring speakers within the same layer expressed as a decimal value between 0 and 1. LBAP focuses on the use of a minimal number of speakers. For point sources it can use anywhere between one to four speakers. Arguably its greatest strength resides in its ability to accommodate just about any speaker con- figuration (from monophonic as many loudspeakers as the hardware can handle) for perimeter-based spatializa- 

Copyright ©2016 Ivica Ico Bukvic et al. This is an open-access article distributed under the terms of the Creative Commons Attribution License 3.0 Unported, which permits unrestricted use, distribution, and reproduc- tion in any medium, provided the original author and source are credited.
Introducing D4: An Interactive 3D Audio Rapid Prototyping and Transportable Rendering Environment Using High Density Loudspeaker Arrays

Ilica Ico Bukvic
Virginia Tech
SOPA, DISIS, ICAT
lico@vt.edu

ABSTRACT

With a growing number of multimedia venues and research equipped with High Density Loudspeaker Arrays, there is a need for an integrative 3D audio spatialization system that offers both a scalable spatialization algorithm and a battery of supporting rapid prototyping tools for time-based editing, rendering, and interactive low-latency manipulation. D4 library aims to assist this newfound whitespace by introducing a Layer Based Amplitude Panning (LBAP) algorithm and a collection of rapid prototyping tools for the 3D time-based audio spatialization and data sonification. The ensuing ecosystem is designed to be transportable and scalable. It supports a broad array of configurations, from monophonic to as many as hardware can handle. D4’s rapid prototyping tools leverage oculocentric strategies to import and spatially render multi-dimensional data and offer an array of new approaches to time-based spatial parameter manipulation and representation. The following paper presents unique affordances of D4’s rapid prototyping tools.

1. INTRODUCTION

The history of Western music can be seen as a series of milestones by which the human society has emancipated various degrees of human aural perception. Starting with pitch and rhythm as fundamental dimensions, and moving onto their derivatives, such as homophony, and polyphony, each component was refined until its level of importance matched that of other already emancipated dimensions. In this paper the authors posit that the observed maturity or the emancipation of these dimensions is reflected in their ability to carry structural importance within a musical composition. For instance, a pitch manipulation could become a motive, or a phrase that is further developed and varied and whose permutation and combination independently drive the structural development. The same structural importance can be also translated into research contexts where a significant component of the data sonification if not its entirety can be conveyed in the emancipated dimension. With the aforesaid definition in mind, even though timbre plays an important role in the development of the Western music, particularly the orchestra, its strict use as a structural element does not occur until the 20th century. Indeed, the 20th century can be seen as the emancipation of timbre. Similarly, while the audio spatialization has played a role throughout the history of music, with occasional spikes in its importance, including the Venetian cori spezzati [1] or the spatial interplay among the orchestral choirs, its structural utilization is a relatively recent phenomenon. Today, the last remaining dimension of the human aural perception yet to undergo its emancipation is spatialization. From augmented (AR) and virtual reality (VR), and other on- and in-ear implementations, to a growing number of venues supporting High Density Loudspeaker Arrays (HDLAs), 21st century is poised to bring the same kind of emancipation to the spatialization as the 20th century did to timbre. Similarly, data sonification and spatialization using primarily spatial dimension are relatively new but nonetheless artists research areas whose full potential is yet to be realized [2].

In this paper HDLAs are defined as loudspeaker configurations of + loudspeakers capable of rendering 3D sound without having to rely solely on virtual sources or post-processing techniques. This definition suggests there are multiple layers of loudspeakers spread around the listening area’s perimeters.

Apart from the ubiquitous amplitude panning [3], contemporary audio spatialization algorithms include Ambisonics [4], Head Related Transfer Function (HRTF) [5], Vector Based Amplitude Panning (VBAP) [6], Depth Based Amplitude Panning (DBAP) [7], Manifold-Interface Amplitude Panning (MIAP) [8], and Wave Field Synthesis (WFS) [9].

There is a growing number of tools that leverage the aforesaid algorithms. This is of particular interest because the lack of such tools makes it particularly cumbersome to integrate algorithms, and also highlights the need for a special spectral adjustment or per-loudspeaker algorithms, its positioning is driven primarily by the azimuth and elevation, leaving it up to user to provide more advanced time-based editing and playback. Others focus on plugins for digital audio workstations (e.g. Sound Ports [14], Meyer’s Cuestation [15], Zirkonium [16], and Sound Emotion’s Wave 1 [17]. The fact that a major- ity of these tools have been developed in the past decade points to a rapidly developing field. A review of the existing tools has uncovered a whitespace [18], a unique set of desirable features an algorithm coupled with time-editing tools ought to deliver in order to foster a more widespread adoption and with it standardization:

• The support for irregular High Density Loudspeaker Arrays;
• Focus on the ground truth with minimal amount of idiosyncrasies;
• Leveraging the vantage point to promote data comprehension;
• Optimized, lean, scalable, and accessible;
• Ease of use and integration through supporting rapid-prototyping time-based tools.

2. D4

D4 is a new Max [10] spatialization library that aims to address the aforesaid whitespace by:

1. Introducing a new lean, transportable, and scalable Layer Based Amplitude Panning (LBAP) audio spatialization algorithm capable of scaling from monophonic to HDLAs environments, with particular focus on advanced perimeter-based spatial manipulations of sound that may prove particularly useful in artis- tic, as well as sonification and sonification scenarios, and
2. Providing a collection of supporting rapid prototyping time-based tools that leverage the newfound audio spatialization algorithm and enable users to effi- ciently design and deploy complex spatial audio im-

D4’s Layer Based Amplitude Panning (LBAP) algorithm groups speakers according to their horizontal layer and calculates point sources using the following series of equations applied to the four nearest speakers:

\[
B_{\text{amp}} = \frac{\cos(\text{BL}_{\text{distance}} \cdot \pi/2) \cdot \cos(\text{BR}_{\text{amp}} \cdot \pi/2)}{(1) \\
B_{\text{amp}} = \frac{\sin(\text{BL}_{\text{distance}} \cdot \pi/2) \cdot \cos(\text{BL}_{\text{amp}} \cdot \pi/2)}{2) \\
A_{\text{amp}} = \frac{\cos(\text{AL}_{\text{distance}} \cdot \pi/2) \cdot \cos(\text{AR}_{\text{amp}} \cdot \pi/2)}{3) \\
A_{\text{amp}} = \frac{\sin(\text{AL}_{\text{distance}} \cdot \pi/2) \cdot \cos(\text{AR}_{\text{amp}} \cdot \pi/2)}{4)
\]

In the aforesaid equations B stands for the nearest layer below the point source’s elevation and A the nearest layer above. BL stands for the nearest left speaker on the below layer, BR for nearest right speaker on the below layer, AL for the nearest left speaker on the above layer, and AR for the nearest right on the above layer. Amp refers to the amplitude expressed as a decimal value between 0 and 1. Distance reflects the normalized distance between two neighboring speakers within the same layer expressed as a decimal value between 0 and 1. LBAP focuses on the use of a minimal number of speakers. For point sources it can use anywhere between one to four speakers. Arguably its greatest strength resides in its ability to accommodate just about any speaker configuration (from monophonic as many loudspeakers as the hardware can handle) for perimeter-based spatialization with minimal CPU overhead. Like most other algorithms, its positioning is driven primarily by the azimuth and elevation values, with the ear level being 0° elevation and 0° azimuth being arbitrarily assigned in respect to venue’s preferred speaker orientation. The algorithm is further described in greater detail in [18].

Similar to VBP’s Source Spread, LBAP also offers Ri-
dius option that accurately calculates per-speaker ampi-
ditude based on spherical distance from the point source. The Radius distance is expressed in spherical degrees from the center of the point source and loudspeaker position. It also introduces a unique feature called Spatial Mask (dis-
cuss below). When coupled with the D4 library further is enhanced by a series of unique affordances, in-
cluding Motion Blur (also discussed below) and a battery of time-based editors that leverage oculocentric user inter-
facing for generating, importing, and manipulating multi-
dimensional data.

D4 library focuses on mostly open source (MOSS) lean implementation that leverages maximum possible amount of built-in Max objects while introducing only two new java-based objects, namely the main spatialization object D4 and Jitter-based based matrix object. This de-
sign choice introduces new challenges, like the lack of graceful handling of determinacy within the Max’s mul-
tithreaded environment (e.g. using a poly object for dy-
namic instantiation of a mask calculating ab-
stractions). It also provides opportunities for the user to build upon and expand library’s functionality, thus min-
imizing the limitations typically associated with closed (a.k.a. blackbox) alternatives.

Other features aimed at addressing the aforesaid whites-
paces include the support for a broad array of speaker configurations, dynamic reconfigurability of the speaker setup, user-editable loudspeaker configuration syntax, focus on perimeter-based spatialization without the need for a special spectral adjustment or per-loudspeaker processing beyond amplitude, and focus on leveraging real-time-friendly operation (in tests the system was able to render stable audio output with 1ms latency at 48KHz 24bit sampling rate built-on 28 bytes processors), built-in audio bus system per each audio source designed to promote signal isolation and streaming line editing, independent layers (e.g. sub arms), and focus on leveraging real-world acoustic conditions where vantage point is treated as an asset rather than a hindrance. LBAP does not aim to compensate for vantage point perceptual variances. This is in part be-
cause such an implementation mimics real-world acoustic conditions, and is therefore seen as offering opportunities for broadening of cognitive bandwidth by cross-pollinating different modalities (e.g. spatialization (aural per-
ception), and also in part because it minimizes the need for idiosyncrasies that may limit system’s scalabil-
ity and transportability, and/or adversely affect its overall CPU overhead. D4’s lean design promotes optimization and scalability, as well as easy expansion, with the ulti-
mate goal of promoting transportability. The library can serve as a drop-in replacement for the mainstream spatial-
3. UNIQUE AFFORDANCES

3.1 Spatial Mask

Spatial Mask (SM) is one of the unique features of the D4 ecosystem. Akin to its visual counterpart LBAP, Spatial Mask considers the entire perimeter space to have the default mask of 1. This means wherever the point source is and whatever its radius, it will populate as many loudspeakers as its computed amplitude and radius permit based solely on its calculated amplitude curve. The spatial mask, however, can be changed with its default resolution down to 0.5° horizontally and 1° vertically, giving each loudspeaker a unique maximum possible amplitude as a float point value between 0 and 1. As a result, a moving source’s amplitude will be limited by loudspeaker’s corresponding mask value as it traverses the said loudspeaker. This also allows a situation where a point source with 180° radius that emanates throughout all the loudspeakers can now be dynamically modified to map to any SM, thus creating complex shapes that go well beyond the traditional spherical.

3.2 Time-Based Editing Tools

SM implementation leverages Jitter library and its affordances, making it convenient to import and export SM snapshots and automate time-based alterations. Like a single channel video, D4’s SM editing tools allow for SM translation and can couple azimuth, elevation, as well as up-ramp and down-ramp data into a single collated video file that is accompanied by matrices corresponding with each keyframe. The library can then interpolate between those states at user-specified resolution both in real-time and via batch rendering, allowing for time-stretching and syncing with content of varying duration.

The entire D4 ecosystem is virtual audio bus aware and utilizes widgets (where appropriate) can be easily reconfigured to monitor and/or modify properties of a specific bus. Where applicable, leaving the bus name blank will revert to monitoring main outputs. Apart from the D4.calc abstraction that can play data rendered by the D4.mask.player and feed it into the target D4.calc (a.k.a. bus).

The entire library is envisioned as a modular collection of self-standing, yet mutually aware widgets. User can customize their workspace as they deem fit. The same widgets can be also embedded as GUI-less abstractions in their own patches by leveraging the annotated inlets and outlets, as well as included documentation and examples. In addition, due to their MOSS design the widgets themselves can be further enhanced (e.g. by altering the default speaker configuration that is preloaded within each D4.calc, adding custom filters to specific outputs, or by introducing new and more advanced ways of processing SM matrices). The resulting community enhancements that prove particularly useful may be eventually merged into the future upstream releases.

The ability of each widget to be utilized independently from others is limited only by context. For instance, editing SM makes no sense unless the bus being edited actually exists. Likewise, storing SM is impossible without having a renderer monitoring the same bus. To maximize the possible number of meaningful configurations it has required widgets to carry redundant implementations. For instance, the editor if used solely to alter Mask on a particular bus without the intent to store it (e.g. for a real-time manipulation), requires D4 mask calculator abstraction that is also present within the renderer. Consequently, to minimize the redundancy and the ensuing CPU overhead in situations where abstraction is already part of the same pipeline, the library has a framework to autodetect such a condition and minimize the redundancy by disabling the calculation within the editor and forwarding the editor data directly to the renderer.

3.3 Helper Abstractions

In addition to the aforesaid widgets, D4 also offers a collection of helper abstractions designed to streamline library’s utilization in more complex scenarios. D4, D4.dac, and D4.cell are abstractions used for the dynamic creation of the bus outputs, as well as to prevent CPU intensive real-time calculation. To aid in this process the system offers tools for playback of prerend- ered spatial data thereby making its playback resolution limited only by the per-loudspeaker amplitude crossfade values whose primary purpose is to prevent clicks while also enabling novel features like the Motion Blur.

4. CONCLUSIONS AND FUTURE WORK

D4 is an actively maintained production ready Max library designed to address the limited transportability of spatial audio using HDLAs in artistic and research contexts. It does so by coupling a new Layer Based Amplitude Pan- ning algorithm with a battery of supporting time-based tools for importing, editing, exporting, and rendering spatial data, including real-time low-latency HDLA scenar- ios. The newfound affordances, such as the Radius, Spatial Mask, and Motion Blur, when combined with Jitter-based editing tools, offers opportunities for exploring new ap- proaches to audio spatialization. These include scientific research that furthers the understanding of human perception and more importantly leveraging the ensuing knowledge for the purpose of emancipating spatial audio dimension both within the artistic and research scenarios while providing a scalable and transportable way of dis- seminating HDLA content.

Given D4’s expanding feature set, it is unclear whether the current MOSS approach as a Max library will prove an environment conducive of creativity it aims to promote, particularly in respect to a battery of tools and widgets that in their current form defy more traditional approaches to user interfaces commonly associated with DAWs and other time-based editing tools. Based primarily on user de- mand, it is author’s intention to continue investigating op- timal ways of introducing timeline-centric features within the existing implementation and expanding to other frame- works, including potentially a self-standing application.

5. OBTAINING D4

D4 can be obtained from http://ico.bukvic.net/ main/d4/.
ization alternatives that rely on azimuth and elevation parameters. Furthermore, D4 tools promote ways of retaining time-based spatial configuration in its original, editable format that can be used for real-time manipulation. The same can be also used to render time-based data for different speaker configurations and later playback that bypasses potentially CPU intensive real-time calculation. To aid in this process the system offers tools for playback of prerendered spatial data thereby making its playback resolution limited only by the per-loudspeaker amplitude crosstalk values whose primary purpose is to prevent clicks while also enabling novel features like the Motion Blur.

3. UNIQUE AFFORDANCES

3.1 Spatial Mask

Spatial Mask (SM) is one of the unique features of the D4 ecosystem. Akin to that of its visual counterpart LBAP considers the entire perimeter space to have the default mask of 1. This means wherever the point source is and whatever its radius, it will populate as many loudspeakers as its computed amplitude and radius permit based solely on its calculated amplitude curve. The spatial mask, however, can be changed with its default resolution down to 0.5° horizontally and 1° vertically, giving each loudspeaker a unique maximum possible amplitude as a float point value between 0 and 1. As a result, a moving source’s amplitude will be limited by loudspeaker’s corresponding mask value as it traverses the said loudspeaker. This also allows a situation where a point source with 180° radius that emanates throughout all the loudspeakers can now be dynamically modified to map to any SM, thus creating complex shapes that go well beyond the traditional spherical sources.

3.2 Time-Based Editing Tools

SM implementation leverages Jitter library and its affordances, making it convenient to import and export SM snapshots and automate time-based alterations. Like a single channel video, D4’s SM editing tools use grayscale 2D matrix to calculate the ensuing per-loudspeaker mask. As of version 2.1.0, the time-based editing tools allow for SM translation and can couple azimuth, elevation, as well as up-ramp and down-ramp data into a single coll-formatted translation and can couple azimuth, elevation, as well as included documentation and examples. In addition, due to their MOSS design the widgets themselves can be further enhanced (e.g. by altering the default speaker configuration that is preloaded within each D4 calc , adding custom filters to specific outputs, or by introducing new and more advanced ways of processing SM matrices). The resulting community enhancements that prove particularly useful may be eventually merged into the future upstream releases.

The ability of each widget to be utilized independently from others is only limited by context. For instance, editing SM makes no sense unless the bus being edited actually exists. Likewise, storing SM is impossible without having a rendering monitoring the same bus. To maximize the possible configurations of SM, Jitter has provided abstractions that can carry redundant implementations. For instance, the editor if used solely to alter Mask on a particular bus without the intent to store it (e.g. for a real-time manipulation), requires D4 mask calculator abstraction that is also present within the renderer. Consequently, to minimize the redundancy and the ensuing CPU overhead in situations where the editor and the renderer are not present within the same pipeline, the library has a framework to autodetect such a condition and minimize the redundancy by disabling the calculator within the editor and forwarding the editor data directly to the renderer.

4. CONCLUSIONS AND FUTURE WORK

D4 is an actively maintained production ready Max library designed to address the limited transparency of spatial audio using HDLAs in artistic and research contexts. It does so by coupling a new Layer Based Amplitude Panning algorithm with a battery of supporting time-based tools for importing, editing, exporting, and rendering spatial data, including real-time low-latency HDLA scenarios. The newfound affordances, such as the Radius, Spatial Mask, and Motion Blur, when combined with Jitter-based editing tools, offers opportunities for exploring new approaches to audio spatialization. These include scientific research that furthers the understanding of human spatial perception and more importantly leveraging the ensuing knowledge for the purpose of emancipating spatial audio dimension both within the artistic and research scenarios while providing a scalable and transportable way of disseminating HDLA content.

Given D4’s expanding feature set, it is unclear whether the current MOSS approach as a Max library will prove an environment conducive of creativity it aims to promote, particularly in respect to a battery of tools and widgets that in their current form defy more traditional approaches to user interfaces commonly associated with DAWs and other time-based editing tools. Based primarily on user demand, it’s author’s intention to continue investigating optimal ways of introducing timeline-centric features within the existing implementation and expanding to other framework components, including potentially a self-standing application.

5. OBTAINING D4

D4 can be obtained from http://ico.bukvic.net/main/d4/.
6. REFERENCES


[Akinoi Ito] Tokyo University of Technology
akinoi@ed.tue.u.ac.jp

[Kengo Watanabe] Watanabe-DENKI Inc.
kengo@wdsk.co.jp

[Ken’ichiro Ito] Tokyo University of Technology
itoken@stf.tue.ac.jp

ABSTRACT

isSuperColliderKit (abbr. iSCKit) has been improved in the aspect of productivity and maintainability. In this version, we implemented 3 features: smart initialization without declaring as a shared instance, file reading, avoiding necessity to hande pointers in objective-C. The features become easier to embed due to re-organizing the project template and build setting.

1. CONCEPT

1.1 The Original Motivation of this Project

There have already been some sound API or game sound middleware[1][2] have the features of modifying the sampled data: changing tempo, transpose per source file, dy- namic filtering and mixing. However, it is still difficult to make some musical variations like “virtual improvisator” on iOS. If programmers would develop some kind of ap- plications, they have to develop their own algorithmic composition features and combine the low-level MIDI API. In the meantime, numerous platforms have been proposed to bring synthesis to iOS. libPd[3], AudioKit[4] and MoMu[5] are widely used in iOS developer community. However, they are suitable for building synthesizer or ef- fector, not for playing multiple musical series and chang- ing musical element dynamically. urMus is full fledged factor, not for playing multiple musical series and chang- ing musical element dynamically. urMus is full fledged factor, not for playing multiple musical series and chang- ing musical element dynamically. urMus is full fledged fac-


1.2 Problem of previous work

Based on the concept and target, we run the develop- ment[9] and take it public[10]. At that time, iSCKit was still unstable and inconvenience to use for other developers. The engine had to be instantiated as a shared instance and developers had to manage handle pointer. Further, there was no SuperCollider file handling function in the OSC code increased in length. It caused to some difficulties to manage the long and multiple layered music descriptions. In this article, we report how to improve iSCKit.

2. SUMMARY OF PREVIOUS WORK

To clarify the points of the improvements, we summarize our previous work.

2.1 Replacement the 32bit ARM NEON code

At the beginning of this trial, there were many codes 32bit ARM NEON architecture in the past version that we referred on GitHub repository[11], especially in SC_VFP11.h, I0G0En.cpp, SC_CoreAudio.cpp. The ”vin_next_a(), vfill(), vcopy()” are the functions for 32bit version NEON architecture of standard “vin_next_a(), vfill(), vcopy()”. However, these functions caused many of errors for the latest build environment. Therefore, we replaced and many of

2.2 Adapting to ARC Programming Style

In the iOS programming, the memory management mech- anism “Automatic Reference Counter” was supported from Xcode2. In accordance with it, the “AutoRelease- Pool” used before became deprecated. Therefore, we de- leted all autorelease statement, added release, and many of corresponding dealloc.