Vol.2014-GN-90 No.5
Vol.2014-CDS-9 No.5
Vol.2014-DCC-6 No.5
2014/1/23

ローカル・ラグ制御機能を持つ音楽同期演奏システムの提案

松長 尚徳[1]　李 榮宰[1]　大島 義博[1]　片桐 滋[1]　大崎 美穂[1]

概要：遠隔コラボレーション支援システム「t-Room」上で動作する，ユーザが所持する音楽をネットワークを介して同期的に再生（同時に再生）するシステムを提案、開発した．システムは，現在開発が進んでいる Linux 版 t-Room 用の音響サーバ上に実装した．また，その性能向上のために，リアルタイム Linux を導入することによって，土台となる既存の音響サーバの高速化を行った．さらに，このリアルタイム Linux 上の音響サーバの，高負荷時の動作の安定性や速度の調査も行った．その結果，かなりの高負荷条件下においても平均 23ms 程度の遅延量に留まり，安定に動作することを確認することができた．

キーワード：遠隔合奏，Linux，t-Room，音響サーバ

A proposal of music synchronously playing system having local-lag control

Hisanori Matsunaga[1]　Lee YeongJae[1]　Yoshihiro Oshima[1]　Shigeru Katagiri[1]　Miho Ohsaki[1]

Abstract: We developed a new system that enabled one to synchronously play shared music data over computer networks on a remote collaboration support system called "t-Room". The system was implemented on a Linux-version sound server that we developed for t-Room. To increase the performances of the proposed system, we also sped up the sound server, which was a basis of our system, by adopting Real-time Linux. Then, we investigated such performances as stability and speed of the sound server on Real-time Linux under a heavy load condition. Analyses showed that the sound server on Real-time Linux stably ran only with the delay of about 23 ms.

Keywords: Remote ensemble，Linux，t-Room，Sound server

1. はじめに

テレビ会議システムなどの遠隔地とのコミュニケーションを支援するシステムが広く普及しており，その研究開発が盛んに行われている（[1], [2] など）．しかし多くの既存システムでは，人や物同士の位置関係や方向，距離関係などを共有するための視聴覚メディアの対称性が成立しておらず，同室にいる者と交わすコミュニケーションほどに効果的な遠隔コミュニケーションは行われていない．このメディアの対称性の崩れの克服を目指した，遠隔コラボレーション支援システム「t-Room」が提案されている [3]．t-Room は視聴覚メディアの対称性を確保するために，大型のディスプレイとカメラ，スピーカー，マイクによって構成されるモノリスによって囲まれた部屋構造を持ち，遠隔地にある同じ構成の部屋とネットワーク接続をすることで，視聴覚情報を共有，すなわちメディアの対称性の確保を目指す．この t-Room を実際に利用する際，様々な用途が考えられる．例えば遠隔地にいる人とバンドの打ち合わせをする，楽器を教えるなど音楽に関わるミーティングを t-Room を通して行うこともその候補の一つである．そして，そのようなミーティングの最中にユーザが所有する音楽データを t-Room 経由で遠隔地のユーザと一緒に聴きたいという要

[1] 同志社大学大学院 理工学研究科
Graduate School of Engineering, Doshisha University

求も十分起こりえる．しかし，現在の t-Room にはファイルとして存在する音楽データを遠隔地のユーザと共に聴く機能は存在しない．著者らは，音楽データを遠隔地のユーザと共に聴きたいという要求に応えるために，t-Room 上で動作する同期演奏システムを提案，実装した．提案するシステムはユーザが所持する音楽を遠隔地の t-Room に送信し，自身の部屋と遠隔地の部屋とで同期演奏を行うことで同期作業を支援する．また，本システムは音楽の同期演奏を行うと同時に，停止や一時停止，再生，早送りや巻き戻しなどの楽曲の操作も可能とし，操作が行われると遠隔地の t-Room にも同様の操作が反映される．さらに，同期演奏された楽曲ファイルは，同期演奏が終了した後もハードディスクに保存されるようにする．これは楽曲データを受信する側のユーザが音楽ファイルを加工することや，過去に何の音楽が同期演奏されたかを参照できるようにするためである．

現在の t-Room は Microsoft 社が提供する OS である Windows 上で動作している．しかし，OS のバージョン・アップに伴うメディア処理遅延速度の増加や，DirectSound のバージョン更新によって音データが伝送できなくなることなど，Windows 上で t-Room の開発やメンテナンスを続けるコストが増え続けている．そこで，オープンソースの OS である Linux 上で t-Room の開発が行われている [4][5]．Linux 版 t-Room では映像と音の伝送を，先行研究で開発を進めている映像サーバ [4] と音響サーバ [5] で行っている．本研究で提案するシステムは，この音響サーバ上に実装し，動作確認を行う．また，先行研究の音響サーバ [5] では，処理遅延が 35ms 存在する．先行研究の音響サーバを用いて二地点での同期演奏を行う時，通信遅延を除外した場合でも，データの送信元と受信元で 35ms のズレが生じることとなる．このズレによりユーザが音のズレを知覚し，t-Room 利用時の快適性を喪失させることは十分に考えられる．著者らは，リアルタイム Linux を導入することで，音響サーバ上で同期演奏をする時の処理遅延をより短くすることを試みた．また，これまで精査されていなかった音響サーバの負荷耐性についても分析し，負荷と処理遅延の関係について既存の音響サーバとリアルタイム Linux の比較を行った．

2. 関連研究

2.1 Linux 版 t-Room

先述したように，現在 Linux 版 t-Room の開発が進んでいる [4], [5]．Linux 版 t-Room は，従来の Windows 版 t-Room と同様に，大型のディスプレイとカメラ，スピーカー，マイクによって構成されるモノリスによって囲まれた部屋構造を持つが，音伝送部は先行研究で開発された音響サーバ [5] を，映像伝送部は映像サーバ [4] を用いており，それらを統合し，モノリスとして機能させるデバイス制御サーバによって部屋を構成している [8]．

2.2 音響サーバの概要 [5]

音響サーバは，先行研究 [5] では遠隔合奏時の音データを送受信するサーバとして設計されており，マイクロホンから音データを入力する機能と，ネットワークを介した音データの送受信機能，音響サーバ間の遅延を計測する機能，ラグを追加しつつ音データを出力する機能などを持つ．OS は Red Hat Enterprise Linux のクローン OS である CentOS を用いており，音響サーバは CentOS 上で最大のスレッド優先度で動作する．これにより，OS が管理しているデータ I/O に伴うブロッキング制御を円滑に動作させ，従来の音響サーバ [6],[7] よりも安定して動作することが確認されている．音響サーバは多地点との接続が可能になっており，通信遅延量を考慮した上で各音響サーバが同期を取り，音データを同時に再生する．通信遅延による地点間での音データ再生時のズレに対処し，円滑な同期再生を行うために，音響サーバはローカル・ラグ法を用いている．ローカル・ラグ法の詳細は次節に示す．

2.3 ローカル・ラグ法 [9], [10]

以下の図 1 に，ローカル・ラグ法を用いて遠隔合奏を行う時の音データの流れを示す．図 1 では local 側が音データの送信元で，remote 側が音データの受信元であることが示されている．通信遅延を考慮せずに local 側から送られたデータを remote 側で再生した場合，通信遅延分だけ遅れて再生される．これは同期性が重視される遠隔合奏などで大きな問題となる．ローカル・ラグ法とは，通信遅延分のズレを緩和するため，通信遅延時間を計測し，その時間分音データの送信元で音楽再生を遅らせることで同期演奏を実現する手法である．通信遅延時間は，local 側から remote 側へのパケット送信時と remote 側から local 側へのパケット受信時の差分を出し，2 で割ったラウンドトリップディレイタイムにより算出する．

3. ローカル・ラグ制御機能を持つ音楽同期演奏システム

3.1 概要

提案するシステムの構成例を以下の図 2 に示す．

図 2 中，ユーザが t-Room を通して共有したい音楽が音響サーバ A に入っているものとする．この音楽を音響サーバ B と音響サーバ C との間で同期演奏する時，音響サーバ AB 間の通信遅延と，音響サーバ BC 間の通信遅延が，それぞれ 40ms と 20ms であるとする．この場合，楽曲を同期演奏する時に，音響サーバ A に 40ms のスリープを，音響サーバ C に 20ms のスリープを加える．これにより，全ての地点で同時に楽曲を再生することができる．

楽曲の再生中は，いずれの音響サーバからの一時停止と

Vol.2014-GN-90 No.5
Vol.2014-CDS-9 No.5
Vol.2014-DCC-6 No.5
2014/1/23

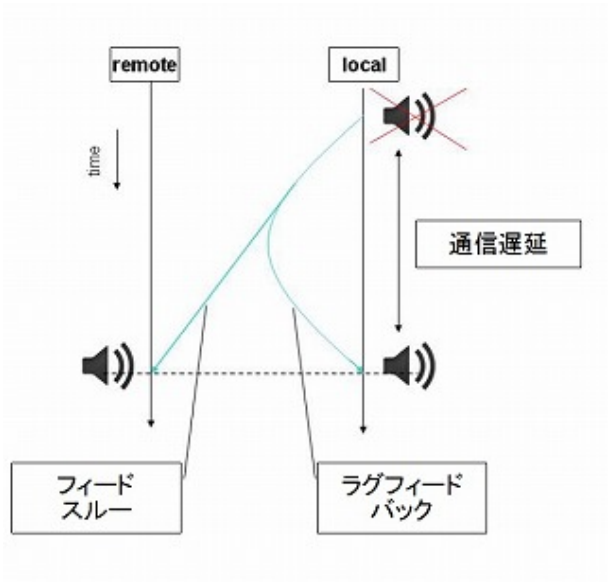

図 1　ローカル・ラグ法の概要
Fig. 1　About Local-lag.

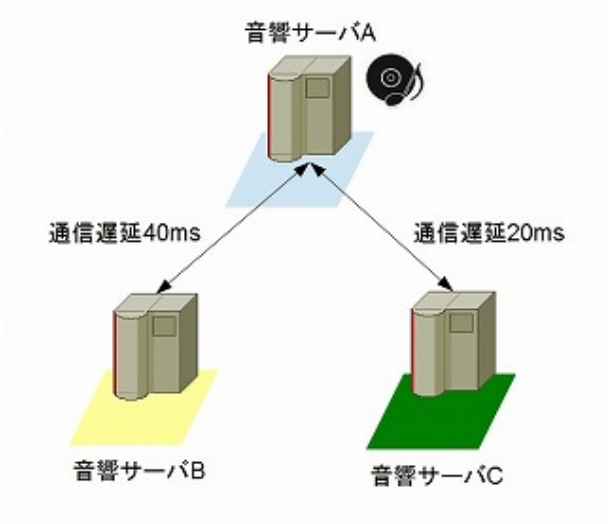

図 2　提案するシステムの構成
Fig. 2　Structure of music synchronously playing system.

停止，再生，シークなどの操作が可能であり，またこれらの操作は同期されているすべての音響サーバにも反映される．例えば，このような操作のコマンドが楽曲データを送信する音響サーバ A で入力された場合，その入力されたコマンドは，音響サーバ B と音響サーバ C に送られ，音楽再生時と同様に，音響サーバ A 上では 40ms のスリープが，音響サーバ C 上では 20ms のスリープが加えられた後に実行される．

また，操作コマンドが楽曲データを受信する側である音響サーバ B あるいは音響サーバ C で入力された場合は，まず楽曲データの送信元である音響サーバ A にコマンドが送信され，続いて音響サーバ A から改めて他の音響サーバにコマンドが送信される．後の手続きは，元々音響サーバ A からコマンドが発せられる場合と同じである．この場合，音響サーバ B から音響サーバ A への通信遅延 40ms が追加でかかるが，三地点で同時にコマンドを実行することができる．これにより，楽曲の操作を全ての音響サーバで同様に行うことができる．

同期演奏が終了した時には，楽曲データは転送先に保存される．これにより，転送先に同期演奏された音楽データの加工を許し，タイムスタンプなどを利用して，いつ楽曲を共有したかなどのヒストリー機能の実現が可能になる．伝送中で音楽ファイルが不完全なままで同期演奏が終了した場合はについては，そのファイルはそのまま破棄される．

3.2　実装と動作確認

提案するシステムの開発と実行のための環境を以下の表 1 に示す．

表 1　提案するシステムの開発と実行のための環境．

Linux kernel version	2.6.35.6
Programming Language	C++
Compiler	GCC 4.4.7
Audio library	ALSA 1.0.22

Linux 版 t-Room に導入される音響サーバは Linux ディストリビューションの一つである CentOS を基に実装されており，提案するシステムも CentOS 上で実装，動作確認を行う．システムの実行は CentOS に標準で搭載されているシェルの bash 上で行い，楽曲の操作も bash 上でコマンドを入力することで実現させる．ネットワーク環境は，通信速度が最大 1Gbps のギガビットイーサネットの LAN を用いた．

表 1 の環境を用いて，提案したシステムがどれほどの精度で動作するかを調べた．今回は二地点の音響サーバで同期演奏を行い，ローカル・ラグによりどれほど二地点間での音楽再生のタイミングのズレが軽減されているかを計測する．実験環境を以下の図 3 に示す．

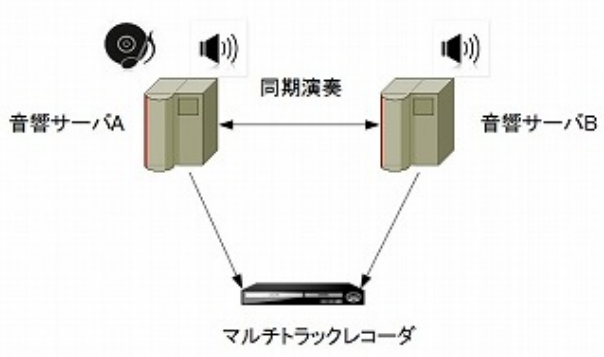

図 3　実験環境
Fig. 3　Experiment environment.

図 3 は，音響サーバ A に音楽が入っており，音響サーバ B と同期演奏を行う場面を想定している．音響サーバ A と

音響サーバ B の同期演奏がどの程度の精度で行われているかを調べるために，マルチトラックレコーダを外部に配置する．マルチトラックレコーダは音響サーバ A と音響サーバ B に接続されており，レコーダに記録される音響サーバ A で再生される音データの波形と，音響サーバ B で再生される音データの波形を比べることで，音楽再生のタイミングのズレを定量的に調べることができる．この実験環境を用いて，実際に音響サーバ間で同期演奏を行った時の音響サーバ A と音響サーバ B から出力される音データの波形を以下の図 4 に示す．

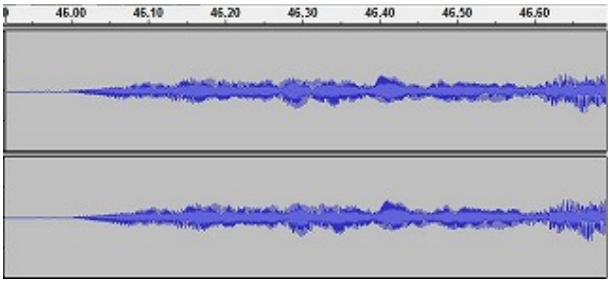

図 4　音響サーバ A および B から出力される音データの波形
Fig. 4　Wave of sound data from Acoustic Server A and B.

波形の分析の結果，音響サーバ A と音響サーバ B のズレは 10ms 程度であった．これは先行研究 [5] での遠隔合奏時の二点間のサーバのズレとほぼ合致する．これにより，提案したシステムが正常に動作することが確認できた．

4. 音響サーバの改良

4.1 音響サーバの処理遅延の改善

提案するシステムは既存の音響サーバ上で動作する．この音響サーバは，先行研究 [5] では時間経過によるシステムの処理遅延の変動を抑えることに成功しているが，依然として 35ms 程度の遅延が発生している．この処理遅延はネットワークを通して音データを受信し，スピーカーから再生する時に発生するので，ネットワークの通信遅延を除外しても，データの送信元と受信元で 35ms のズレが生じることになる．関連研究 [11] では，二地点から音楽を同時に再生する時，40ms 程度のズレは十分知覚できるとされている．そのため，既存の音響サーバの 35ms の処理遅延はユーザに音のズレを知覚させ，快適な協働作業を妨げる可能性があると言える．この処理遅延に対応するために，著者らはリアルタイム OS を基盤とした音響サーバを構築し，処理遅延の測定を行った．リアルタイム OS は，組み込みシステムなど一定周期内でタスクを実行することに特化したものが多いが，本研究では Linux カーネルのタイマ割り込み頻度の増加や Real-time パッチの導入を行ったリアルタイムカーネルの利用により低レイテンシのメディア処理を実現した Ubuntu Studio を用いる．本実験に使ったコンピュータの仕様を以下に示す．

表 2　実験に用いたコンピュータの仕様．

機種	Dell Optiplex 9010
CPU	Intel®Core™i-7-3770 Processor 3.40GHz
メモリ	4GB DDR3 SDRAM
サウンドカード	ctxfi Creative Labs SB X-Fi
コア数	4

また，構築した実験環境を以下の図 5 に示す．

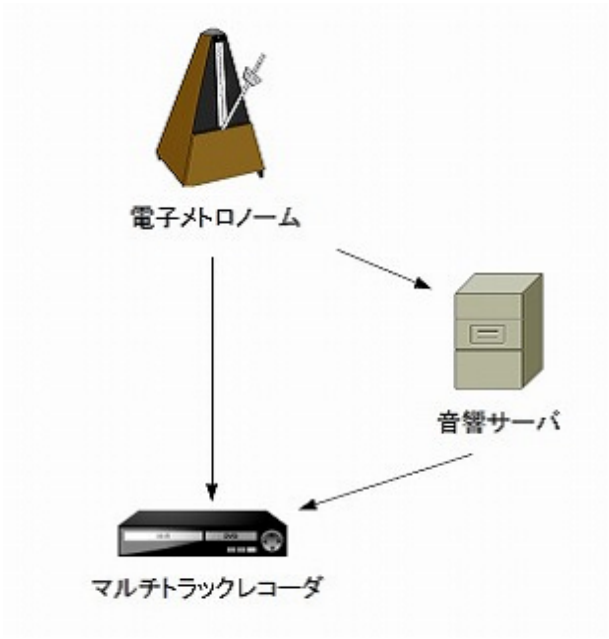

図 5　実験環境
Fig. 5　Environment of the experiment.

実験では，120BPM の電子メトロノームを音源とし，電子メトロノームから直接出力される音と，音響サーバを経由して出力される音とを，マルチトラックレコーダに入力，録音し，それら両者の音の時間的ズレを計測することで遅延時間を割り出した．構築した環境で 60 分間システムを動作させ，実験で得られた 2 つの録音ファイルの 0 分（ファイルの 1 音目）から 60 分までのデータに対し，5 分ごとにメトロノームから直接出力される音データと音響サーバ A から出力される音データのズレを計測した．

まず，表 1 の仕様のコンピュータで既存の音響サーバを動作させた時の実験結果を以下の図 6 に示す．また，リアルタイム Linux 上で音響サーバを動作させた時の実験結果を図 7 に示す．図の縦軸は，メトロノームから直接出力される音と音響サーバ A から出力される音データのズレである．図の横軸は 5 分ごとの計測時点を示している． 計測結果から，既存の音響サーバが常に 38ms 程度の処理時間がかかっていることに対して，リアルタイム OS 上での処理時間は，これも安定的に 23ms 程度であることがわかる．これは，動作中のシステムのプロセスが，より高い頻度でカーネルへの割り込みを行っており，より長い時間プロセスを動作することができていることから，結果として

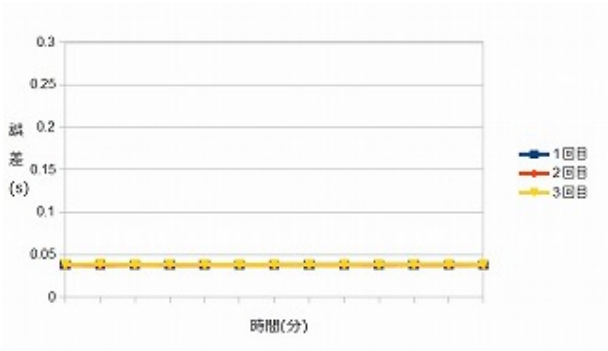

図 6　既存の音響サーバの処理遅延
Fig. 6　Processing delay of the old Acoustic Server.

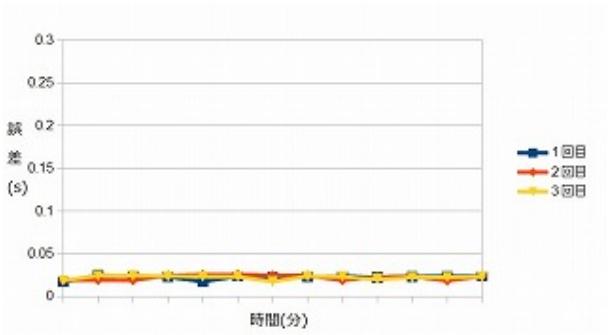

図 7　リアルタイム Linux 上の音響サーバの処理遅延
Fig. 7　Processing delay of the Acoustic Server on real-time Linux.

処理遅延が減ったためだと考えられる．また，60 分の計測中に処理遅延の急激な変動も見られず，既存の音響サーバと同程度に安定して，かつ高速にシステムが動作していることが確認できた．

4.2　音響サーバの高負荷耐性の調査

4.2.1　一定の負荷を加えた場合の負荷耐性

Linux 版 t-Room では，多地点とのセッションを行うことが考えられる．しかし，これまでは音響サーバがどれほどの負荷に耐えうるかの精査がなされていなかった．そこで，先行研究の音響サーバと著者らが処理遅延の軽減のために用いたリアルタイム OS を基盤にした音響サーバとで，高負荷をかけながら音楽伝送をどれだけ安定して行うことができるかを調査した．実験で用いるコンピュータは先述した図 2 の仕様のものを用いた．また，コンピュータに負荷を加えるため，以下のコマンドを用いた．すなわち，y

表 3　負荷を加える bash コマンド．

```
yes >> /dev/null
```

を無限に出力するコマンドである yes を，与えられたデータをすべて破棄するデバイスノードである /dev/null に投げ続けることでコンピュータに高い負荷を加える．このコマンドを一つ実行すると，CPU の 1 スレッドがコマンドの処理のために占有され，スレッドの使用率が限りなく大きくなる．今回の実験で用いたコンピュータの CPU は 4 コア 8 スレッドで動作するため，本コマンドを 8 スレッド分動作させることにより CPU の全スレッドの使用率を限りなく高くし，負荷をかける．その上で音響サーバを動作させ，処理遅延を計測する．まず，既存の音響サーバでの計測結果を以下の図 8 に示す．

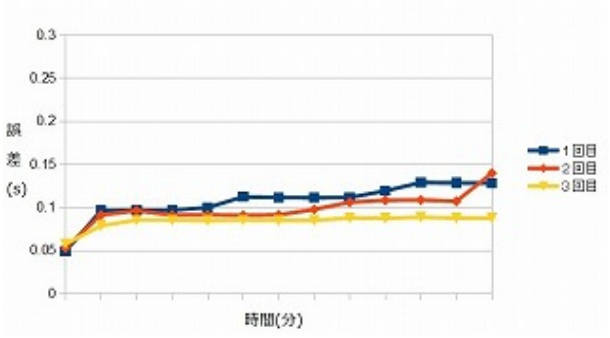

図 8　高負荷時の処理遅延（旧音響サーバ）
Fig. 8　Processing delay for preceding sound server under a heavy-load condition.

既存の音響サーバでは，負荷を掛けていない時の平均の処理遅延が 35ms だったのに対して，図 8 に示すように，最初は 50ms 程度の遅延があり，時間が経つにつれ遅延が 100ms 程度まで増加し，その後は安定して動作を続けていた．

次に，リアルタイム OS での実験結果を図 9 から図 13 に示す．Ubuntu Studio ではサウンドサーバデーモンとして JACK を利用している．JACK サウンドサーバでは，音の入出力時のデータサイズを変更できる．本実験ではデータサイズを 256, 512, 1024, 2048, 4096 と変化させて処理遅延を計測した．図の縦軸は，メトロノームから直接出力される音と音響サーバ A から出力される音データのズレである．図の横軸は 5 分ごとの計測時点を示している．

計測結果を見ると，図 9，図 10 のデータサイズ 256，512 の時では処理遅延が不安定に遷移していることが分かる．これは小さいデータサイズで音の入出力を行った場合に，

図 9　高負荷時の処理遅延 (データサイズ 256)
Fig. 9　Processing delay at heavy load (Data size 256).

Vol.2014-GN-90 No.5
Vol.2014-CDS-9 No.5
Vol.2014-DCC-6 No.5
2014/1/23

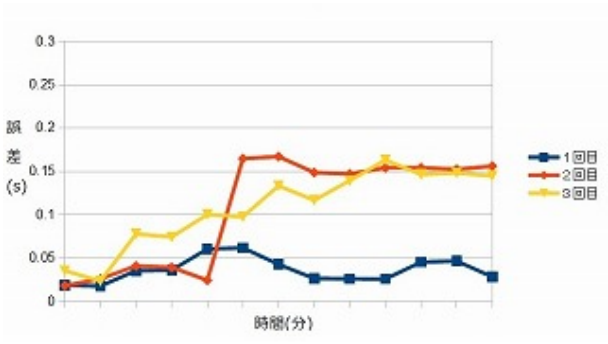

図 10　高負荷時の処理遅延 (データサイズ 512)
Fig. 10　Processing delay at heavy load (Data size 512).

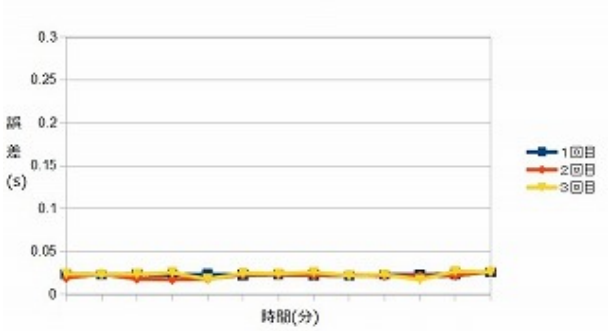

図 11　高負荷時の処理遅延 (データサイズ 1024)
Fig. 11　Processing delay at heavy load (Data size 1024).

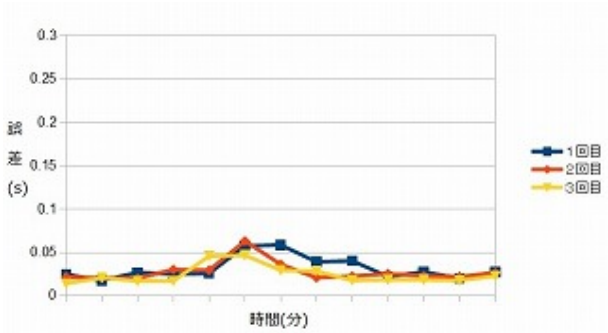

図 12　高負荷時の処理遅延 (データサイズ 2048)
Fig. 12　Processing delay at heavy load (Data size 2048).

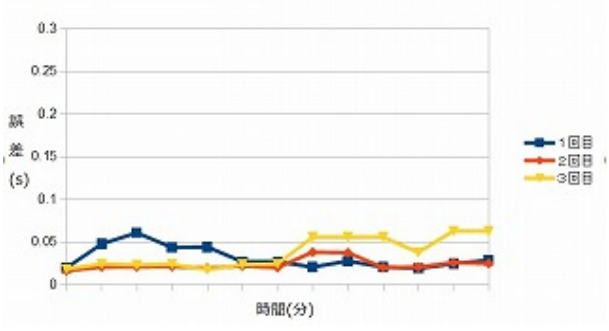

図 13　高負荷時の処理遅延 (データサイズ 4096)
Fig. 13　Processing delay at heavy load (Data size 4096).

よりバッファの書き込み，読み出しの頻度が増えるために処理の負担がかかり，高負荷がかかった時に処理を一定周期に収められなくなるからだと考えられる．また，図 12，図 13 のデータサイズ 2048，4096 の時ではデータサイズが 256，512 の時よりは安定しているものの，時折処理遅延が 60ms 程度かかることがあった．これは，データサイズを大きくとると処理の負担は小さくなるが，バッファからのデータの入出力のタイミングが遅くなるため，その時間分だけ処理遅延として結果に表れるためだと考えられる．結果として，処理の負担が少なく，またバッファからのデータの入出力のタイミングが適切である図 11 のデータサイズ 1024 の時が最も安定して動作することが分かった．処理遅延の平均値は 23ms であり，これは図 7 での実験結果とほぼ合致する．これにより，リアルタイム Linux 上で適切な JACK データサイズを設定することで，高負荷がかかった時でも安定して，かつ平常時と同様の処理遅延で音響サーバが動作することが確認できた．

4.2.2　不規則な負荷を加えた場合の負荷耐性

上述の実験では，CPU の全スレッドの CPU 使用率が限りなく高くなるようにコマンドを実行し，計測を行った．しかし，実際に音響サーバを t-Room 上で利用する時，何台の音響サーバと同時に接続するかは不確定であり，どの程度の負荷がかかるかも不定である．そこで，不規則な負荷が音響サーバに加えられた時に処理遅延がどう推移するのかを実験，計測する．実験に一意性を持たせるために，負荷は 5 分ごとにあらかじめ決められた負荷量に変えるものとする．事前に決めた負荷量を以下の図 14 に示す．

時間(分)	0	5	10	15	20	25
負荷コマンドの数	8	4	16	0	2	6
時間(分)	30	35	40	45	50	55
負荷コマンドの数	10	14	8	2	6	8

図 14　音響サーバへの負荷量
Fig. 14　Load amount to sound server.

先行研究での音響サーバでの実験結果を図 15，リアルタイム Linux 上での実験結果を図 16 に示す．リアルタイム Linux 上での実験は，先述した固定負荷での実験において最も安定かつ処理遅延が短かった JACK データサイズ 1024 で行った．

図 15 を見ると，既存の音響サーバでは，固定負荷を加えた時と同様に開始時に 50ms 程度の処理遅延がかかっている．それからしばらくは 50ms 程度の処理遅延でシステムが動作していたが，10 分時の負荷コマンドを 16 個並列に動作させた時に処理遅延が 200ms 程度まで急激に上昇した．そこから，15 分時には負荷コマンドを 0 個，20 分時には負荷コマンドを 2 個並列動作と，負荷があまりかからない状況でも処理遅延が高いままであり，負荷コマンドの数にかかわらず 200ms 程度の処理遅延が計測終了時まで続いた．一方，図 16 のリアルタイム Linux 上の音響サーバの結果を見ると，負荷コマンドの数に関わらず計測終了時

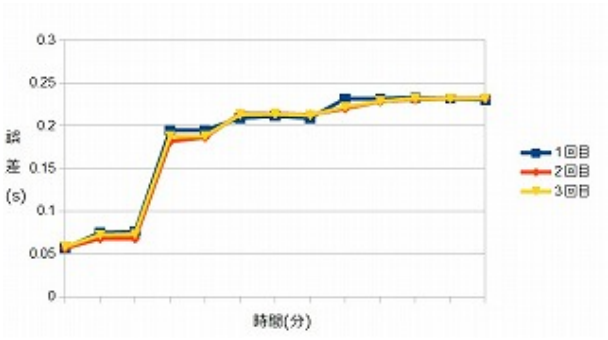

図 15　不規則な負荷を加えた時の処理遅延 (旧音響サーバ)
Fig. 15　Processing delay for preceding sound server under altering heavy load.

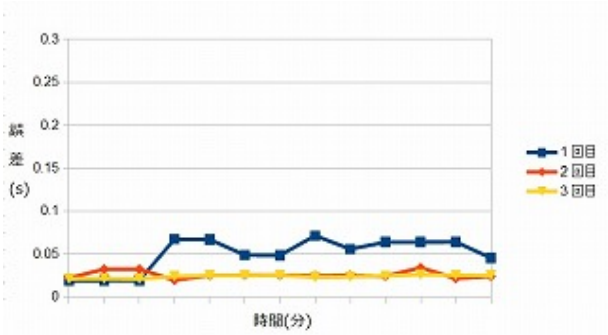

図 16　不規則な負荷を加えた時の処理遅延 (リアルタイム Linux)
Fig. 16　Processing delay for sound server on real-time Linux under altering heavy load.

まで 25ms 程度の処理遅延でシステムが動作していることがわかる．しかし，1 回目の計測では，既存の音響サーバと同様，10 分時に 60ms まで処理遅延が増大しており，それから低負荷時には 50ms 程度まで処理遅延が下がったが，50ms より遅延が短くなることはなく，計測終了まで 50ms から 60ms の間で遅延が推移した．これにより，短期間に急激な負担がかかった時に，既存の音響サーバとリアルタイム Linux 上の音響サーバとで同様に処理遅延が急激上昇し，負荷がなくなっても処理遅延が大きいままシステムが動作し続けることがあり得ることが分かった．しかし，従来の音響サーバに比べれば，リアルタイム Linux 上のサーバは平均的には安定的に動作することも明らかになった．

5.　実験結果と考察

以上の実験により，リアルタイム Linux 上での音響サーバが既存の音響サーバよりも短い処理遅延でシステムを実行でき，また一定の負荷を加えた状況でも，安定して 25ms 程度の処理遅延でシステムが動作することが確認できた．しかしながら，短期間に急激な負荷がかかった場合には既存の音響サーバとリアルタイム Linux 上での音響サーバの両方においてシステムの処理遅延が急激に増大し，負荷がなくなった時にも処理遅延が大きいままであった．実際に t-Room を利用する時は，何台の音響サーバが同時に接続するかは不定であり，どれほどの負荷がかかるかも不明である．その中で，短期間で高い負荷がかかった時にシステムの処理遅延が大きくなったまま遅延が小さくならないのでは，t-Room を利用する時の同質感を損ねることに繋がりかねない．この問題について，既存の音響サーバでは負荷がなくなった時に処理遅延が全く小さくならないが，リアルタイム Linux 上の音響サーバでは 10ms 程小さくなっているため，Linux カーネルのタイマの割り込み周期や，スケジューリングの優先度などが関係しているのではないかと推測できる．よって，音響サーバにとって最適な Linux カーネルのタイマ割り込み周期を計測することや，より高度なタイムスケジューリングが可能なリアルタイム OS の利用，CPU コアを個別にリアルタイムカーネル上で動作させることのできる ART-Linux[12] の導入などを行うことで，負荷がかからなくなった後に，処理遅延をより小さくできるのではないかと考える．また，OS のカスタマイズによる負荷耐性を向上させるだけでなく，コンピュータに高い負荷がかかった時に伝送する音データの質を下げるなど，高負荷時に柔軟に対応できるようシステムを改良することで，より安定した処理遅延でシステムを実行できるのではないかと予想できる．

6.　まとめ

Linux 版 t-Room 上で動作する，ユーザが持つ楽曲を同期演奏するシステムの提案，開発を行った．その結果，開発したシステムが音響サーバ上で正常に動作することが確認できた．また，システムの処理遅延を短くするためにリアルタイム Linux の導入を行い，従来の音響サーバとのシステムの処理遅延の比較実験を行った．加えて，今まで行われていなかった音響サーバの負荷耐性を計測し，リアルタイム Linux 上の音響サーバと従来の音響サーバとで，一定の負荷を加えた時のシステムの処理遅延と，負荷を不規則に加えた時のシステムの処理遅延の変化を明らかにした．

謝辞　本研究を進めるにあたり，システムの設計から開発に至るまで数多くの御助言を頂きました山口毅氏に深く感謝致します．

参考文献

[1] Bly, S. A., Harrison, S. R. and Irwin, *S. Media spaces: bringing people together in a video, audio, and computing environment*, Communications of the ACM, 36, 1, pp.28-46 (1993).
[2] 森川治: 超鏡: 魅力あるビデオ対話方式を目指して, 情報処理学会誌, 41, 3, pp.815-822 (2000).
[3] Hirata, K., Harada, Y., Takada, T., Aoyagi, S., Shirai, Y., Yamashita, N., and Yamato, J.: *The t-Room: Toward the Future Phone*, NTT Technical Review, Vol. 4, No. 12, pp. 26-33 (2006).
[4] 村上昴, 飯田卓也, 片桐滋, 大崎美穂: 遠隔コラボレーション支援のための映像伝送システムの開発とその遅延評価, 信学会 信学技報 ITS2011-56, IE2011-132, pp.325-330

(2012).
[5] 岩原正典, 竹森幸輝, 前田佳奈, 片桐滋, 大崎美穂: ローカル・ラグ制御機能と同期機能を持つ音響サーバの性能評価, 信学会 信学技報 ITS2011-56, IE2011-132, pp.325-330 (2012).
[6] 竹森幸輝, 前田佳奈, 岩原正典, 片桐滋, 大崎美穂: ローカル・ラグ制御機能とログ同期機能を持つ音響サーバの開発, オーディオビジュアル複合情報処理研究会 (2012.02).
[7] 前田佳奈, 竹森幸輝, 岩原正典, 片桐滋, 大崎美穂: ローカル・ラグ機能を持つ音響サーバを用いた遠隔合奏の評価, オーディオビジュアル複合情報処理研究会 (2012.02).
[8] 荻野裕也，杉本直也，片桐滋，大崎美穂 : *Linux* 版 *t-Room* の開発：デバイス制御システムの設計と実装, 情報処理学会研究報告 (2014)(掲載予定).
[9] 入江洋介，青柳滋己，高田敏弘，平田圭二，梶克彦，片桐滋，大崎美穂 : *t-Room* のための遠隔合奏支援システムの構築, 情報処理学会研究会報告, 2009-GN-73 (23), (2009).
[10] Dane Stuckel and Carl Gutwin: *The effects of locallag on tightly-coupled interaction in distributed groupware*, Computer Supported Cooperative Work, pp. 447-456 (2008).
[11] Massimiliano Zampini, Timothy Brown, David I. Shore, Angelo Maravita, Brigitte Roder, Charles Spence *Audio-tactile temporal order judgments* Acta Psychologica 118, pp.277-291 (2005).
[12] 加賀美 聡, 石綿 陽一, 西脇 光一, 梶田 秀司, 金広 文男, 尹 祐根, 安藤 慶昭, 佐々木 洋子, Simon THOMPSON, 松井 俊浩 *ART-Linux* の複数コア利用機能によるソフトウエアディペンダビリティー向上　第 12 回計測自動制御学会システムインテグレーション部門講演会論文集, pp.728–731, 京都大学, Dec., 2011.