G-05

ローカルラグ制御機能を持つ音響システムの改良

Improvement of Acoustic Server Having Local Lag Control

竹森 幸輝† 入江 洋介† 前田 佳奈† 片桐 滋† 大崎美穂†
Koki Takemori Yosuke Irie Kana Maeda Shigeru Katagiri Miho Osaki

1. まえがき

遠隔コラボレーション支援を目指して，t-Room などの様々な支援システムの研究が精力的に行われている[1]．コラボレーション作業は一般に，音楽アンサンブルのような同時作業と会話のような交互作業に分類されるが，ネットワークや計算処理に基づく遅延をどうしても避けることができない遠隔コラボレーションの場合，同時作業を実施することは決して容易ではない．

しかし，省エネルギーや生産性向上を強く要請される今，遠隔地を結ぶ同時作業実現への期待も大きい．通信に不可避の遅延が存在するとき，可能な遅延対策の一つとして，作業者の知覚のレベルにおける遅延を制御することを考えることができる[2][3]．筆者らは，特に音メディアに着目し，遠隔地への音信号の送信遅延と同量の遅延をローカルな再生音に付加するローカルラグの，同時作業における効果の調査を行ってきた[3]．

上記の調査研究のために，筆者らは，遠隔合奏におけるローカルラグ量を制御する実験用音響システムを構築してきた．しかし，その操作には専門的な技術が必要とされ，かつ種々の制約もあった．音響的ローカルラグの研究をさらに進めるためには，その土台となる音響システムの性能向上は必須である．本稿は，この点に動機づけられて行った音響システムの改良結果を報告するものである．改良は，特に，これまで個別に実装されていた 3 種類のクライアントプロセス（音データの送受信や遅延制御を行う）の統合と，プロセスの実行を制御する条件設定ファイルの読み込み機能の追加，コマンドインタフェースの追加，そして，2 地点接続に制限されていた制約を排除した多地点接続機能の追加に関して行った．以下のページにおいて，順次その詳細を紹介する．

2. ローカルラグを用いる遠隔合奏支援手法

2.1 ローカルラグ

図 1 にローカルラグの原理を図解する．図 1 において，2 人の演奏者は別々の t-Room に配置されているとする．それぞれの t-Room における時間の経過は，縦の時間軸によって表現される．演奏者自身が聴く演奏音をローカルフィードバック音と呼び，通信過程を経由して合奏相手に届けられる演奏音を遠隔フィードスルー音と呼ぶ．

図 1 に示されている様に，ローカルラグは，演奏者自身が聴くローカルフィードバック音に通信遅延と同量の遅延を付加してから再生する手法である．

† 同志社大学大学院
Graduate School of Engineering, Doshisha University

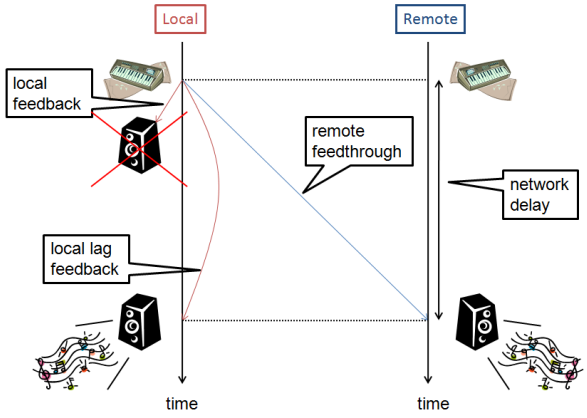

図 1　ローカルラグの原理.

遅延量が等しいため，ローカルフィードバック音と遠隔フィードスルー音とは同時にそれぞれの演奏者に届けられ，ここに同時性が成立する．本稿ではこれ以降，このラグが付加されたローカルフィードバック音をローカルラグフィードバック音と呼ぶ．

2.1 拡張ローカルラグ

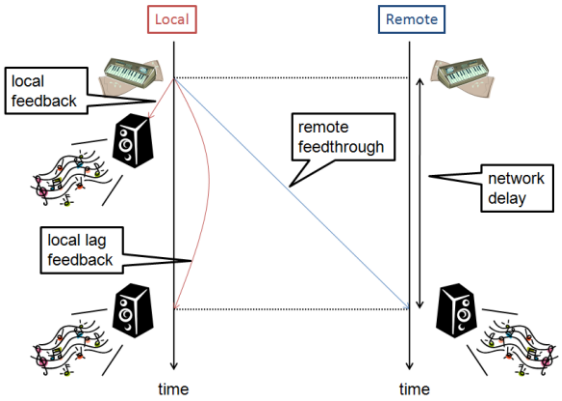

図 2　拡張ローカルラグの原理.

ローカルラグの導入によって，演奏者自身は，自分の打鍵に合わせた演奏音を聴くことは出来ずに，遠隔地にいる合奏相手と同時ではあるものの，遅れた演奏音しか聴くことができなくなってしまう．これは，演奏者に新たに負担を強いることになり，演奏をかえって困難にする恐れがある．その対策として，ラグのないローカルフィードバック音と，ローカルラグフィードバック音との両方を提示する方法を導入する．本稿ではこの手法を拡張ローカルラグと呼ぶ．図 2 に拡張ローカルラグの原理を図解する．

3. ローカルラグ制御機能を持つ音響システム

3.1 システムの構成

音響システムは，Fedora 8 を OS とする Linux PC を用いて構築した．そのハードウェア諸元は以下の通りである．

- CPU：Intel® Core™2 CPU 2.66GHz
- メモリ：3GB
- OS：Fedora 8
- サウンドカード：M-AUDIO Delta 1010LT
- 符号化方式：Linear-16
- 標本化周波数：44.1kHz
- チャンネル数：2（ステレオ）
- 入力デバイス数：4
- 出力デバイス数：4

3.2 通信方式

音響システムは，楽器音の入力や他地点への音データの送信（転送）を行うクライアントと受信した音信号を再生用にサウンドカード側に出力するサーバから構成される，クライアント・サーバ型のシステムである．このクライアント・サーバ間の通信では，その用途に応じて UDP と TCP の 2 種類のプロトコルが用いられている．

TCP はクライアントとサーバの間のセッションの確立に用いられる．クライアントは起動するとまず，サーバに対して TCP コネクションの確立を要求する．コネクションが確立したら，サーバは接続したクライアントの情報を保持するための準備を行う．これ以降，ここで確立したコネクション自体がクライアントを識別する際の識別子となる．より厳密に言うと，クライアントと接続しているソケットのソケットディスクリプタを識別子として用いている．クライアント情報を保持する準備が整うと，サーバはクライアントに準備が整った旨を伝える．それを確認したクライアントは，サーバに対し，UDP 受信の準備をするよう促す．UDP の用途については後述する．サーバ側の UDP 受信準備が整ったことを確認したら，クライアントはサーバ側で必要な自分の情報を送信する．このような流れでクライアント・サーバ間にセッションが確立される．

UDP は，以上のように確立したセッション上で，演奏音をやりとりする際に用いられる．

上記の流れを簡潔にまとめると，この音響システムでは，まず TCP を用いてセッションを確立し，確立したセッション上で UDP を用いて演奏音を転送する．

3.3 演奏音の流れ

2 地点間接続時の，各プロセス間での大まかな演奏音の流れを図 3 に示す．

ある地点の演奏者が発した演奏音は，まずその地点のクライアントに入力される．クライアントは，入力された演奏音に対してローカルラグなどの通信制御を適用した後，ローカルフィードバック音としてローカルのサーバに，遠隔フィードスルー音として演奏相手側のサーバに，それぞれ転送する．このとき，起動しているクライアントプロセスによって適用される通信制御方式が変わる．また，ラグ付加などの通信制御はローカルフィードバック音にのみ適用される．転送された演奏音をサーバが出力することで，各演奏者にそれぞれの演奏音が届くという仕組みである．

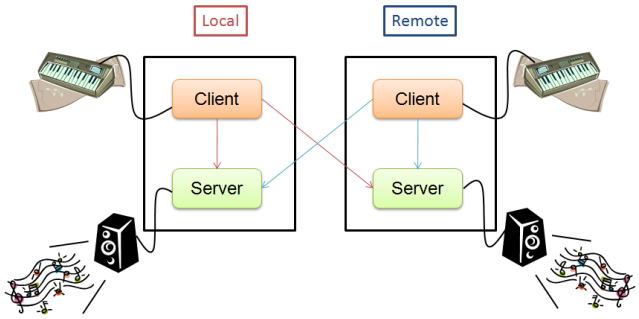

図 3　2 地点間接続時の演奏音の流れ．

3.4 通信制御方式の切り換え

1 部屋の t-Room には，1）ローカルラグを適用せず通常の通信方式を用いるクライアントと 2）ローカルラグを適用するクライアント，3）拡張ローカルラグを適用するクライアントの 3 種類のクライアントプロセスが用意されており，それらは個別のプロセスとして扱われる．適用したい通信制御を担うクライアントプロセスを起動することによって，意図した通信制御方式を適用することができる．

3.4 各プロセス内のスレッドの動作

音響システムは全部で 4 つのスレッドから構成されている．クライアントプロセスが 2 つ，サーバプロセスが 2 つ，それぞれスレッドを並列に動作させている．この節では，各スレッドの動作について述べる．図 4 に，2 地点間接続時の，各スレッド間での詳細な演奏音の流れを示す．

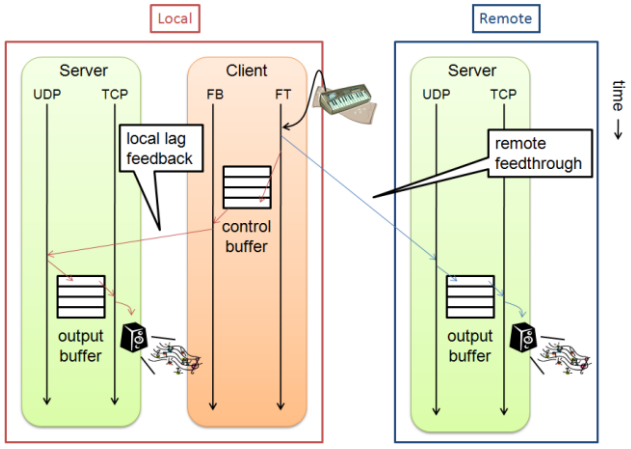

図 4　各スレッド間での演奏音の流れ．

図中の黒い縦の矢印は各プロセス内で並列に動作しているスレッドを表している．クライアントのスレッド間に示されている control buffer は，通信制御の際に用いられるバッファであり，制御バッファと呼ぶ．サーバの TCP

スレッドが持つ output buffer は，出力される演奏音が格納される出力用バッファであり，出力バッファと呼ぶ．

まず，サーバプロセスで並列に処理されている，UDP スレッドと TCP スレッドについて述べる．

UDP スレッドは，クライアントから転送された演奏音を受け取り，保持するスレッドである．

TCP スレッドは，UDP スレッドによって保持されている演奏音を出力バッファに受け取り，出力するスレッドである．TCP コネクションを用いたクライアント接続状況の管理も行っている．

続いて，クライアントプロセスで並列に処理されている，FT(FeedThrough)スレッドと FB(FeedBack)スレッドについて述べる．

FT スレッドは，演奏音が入力されるスレッドであり，遠隔フィードスルー音を扱うスレッドでもある．入力された演奏音は，同プロセス内の制御バッファと通信先サーバの UDP スレッドに，ほぼ同時に転送される．この時，例えばローカルラグ制御を行うクライアントプロセスだった場合，制御バッファに対して転送する演奏音にのみ，現在時刻に通信遅延量を足したものを時刻情報として付加する．拡張ローカルラグ制御を行うクライアントプロセスの場合，制御バッファが 2 つ用意されている．片方には現在時刻のみを付加した演奏音を格納し，もう一方には現在時刻と通信遅延量を足したものを付加した演奏音を格納する．通信制御を行わないクライアントプロセスの場合，制御バッファは存在せず，遠隔フィードスルー音を転送するのとほぼ同時に，直接サーバへとローカルフィードバック音を転送する．

FB スレッドは，ローカルフィードバック音を扱うスレッドである．このスレッドは通信制御を行うクライアントプロセスでのみ動作する．制御バッファ内から演奏音を取り出し，サーバに転送するのだが，その際，格納時に付加された時刻を経過するまで演奏音の取り出しを行わない．この待ち時間により、ローカルラグフィードバック音を作成する．拡張ローカルラグ方式の場合，ローカルラグフィードバック音とラグの無いローカルフィードバック音の 2 つを合成してサーバに転送する．

4. 追加機能の設計と実装

4.1 基本的な設計方針

今回追加実装したすべての機能の根底にある基本コンセプトは，実験環境の改良である．特に，実験効率の向上と実験内容の拡張を目指すものであり，具体的には，1）3 種類のクライアントプロセスの統合，2）設定ファイルの読み込み機能の追加，3）コマンドインタフェースの追加，そして 4）多地点接続機能の追加を行う．

4.2 3 種類のクライアントプロセスの統合

この節では，3 種類あるクライアントプロセスを 1 つに統合するための設計と実装を行う．

実装は，各クライアントプロセスから通信制御部のモジュールを抽出し，抽出した 3 つのモジュールを 1 つのクライアントプロセスに持たせるという方針で行った．どの制御方式を用いるかは，コマンドオプションで指定する．図 5 に，通信制御モジュール統合前と統合後，それぞれのプロセス構成の対比図を示す．

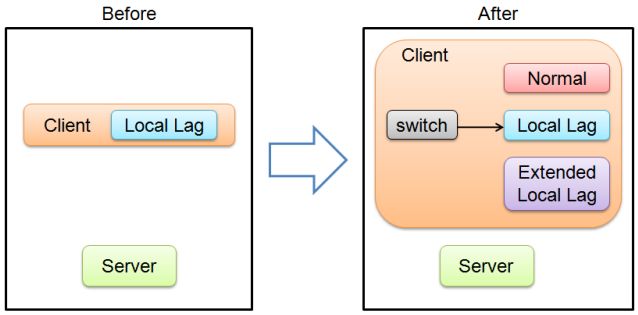

図 5　クライアント統合前と後のプロセス構成.

図中，Before は統合前の構成を，After は統合後の構成を表している．統合前は，適用したい通信制御方式に応じて，プロセス自体を切り換える必要がある．一方統合後は，同じクライアントプロセスの中で制御方式を切り換えることができるため，1 つのクライアントプロセスを扱うだけでよい．

4.3 設定ファイル読み込み機能

この節では，まず音響システム用の設定ファイルを定義する．さらに，その設定ファイルからパラメータを読み込む機能を設計・実装する．

4.3.1 設定ファイルの定義

設定ファイルは，XML をベースとした独自仕様の書式を持ち，大まかな項目を指定するための要素タグと，要素タグで指定した項目内の細かな設定情報を指定するための属性というタグから構成されている．要素タグは開始記号と終端記号で構成され，それらの記号の間に，その要素タグが持つ属性タグを記述する．図に，設定ファイルの記述例を示す．

```
<system>
connectnum = 2
</system>

<feedback>
host = localhost
spkindex= 0
</feedback>

<feedthrough>
host = 192.168.1.26
tcpport = 11000
udpport = 12000
spkindex = 1
</feedthrough>
```

この例では，ローカルな地点と，192.168.1.26 という IP アドレスを持つ地点の 2 地点間接続を想定している．また，リモート側はセッションを確立する際に用いる TCP のポートに 11000 番を使用し，演奏音の通信に用いる UDP ポートに 12000 番を使用するとしている．また，自身の演奏音であるローカルフィードバック音はスピーカ番号 0 番から，相手側の演奏音である遠隔フィードスルー音はスピーカ番号 1 番から出力されるように指定されている． 表 1 に要素タグの種類とその概要を示す．なお，特筆されていない制限は，実装仕様上設けられた制限である．

表 2 に system タグが持つ属性タグの種類と概要を，表 3 に feedback タグと feedthrough タグが持つ属性タグの種類と概要を示す．

この設定ファイルでは，「#」と「//」を 1 行コメント記号として定義している． 1 行コメント記号を入力すると，その記号以降の文字を行単位でコメントアウトすることができる．

表 1　要素タグ．

要素名	概要
system	システムの全般的な設定を行う．conf ファイルの最初に記述する必要があり，必ず 1 つ記述されなければならない．また，複数記述されてはならない．
feedback	ローカルフィードバック音の設定を行う．必ず 1 つ記述されている必要があり，複数記述することはできない．また，feedthrough タグの前に記述されていなければならない．
feedthrough	遠隔フィードスルー音の設定を行う．出力デバイス数との関係で，最大 3 つまでしか追加できない．

表 2　system タグが持つ属性タグ．

属性名	概要
connectnum	接続地点数を指定する．出力デバイス数との関係で，1 から 4 までで指定する必要がある．feedback タグと feedthrough タグの数を合計した数を指定しなければならない．これは多地点接続機能との関係で設けられた制限である．詳細は 4.4.1 で述べる

表 3　feedback タグと feedthrough タグが持つ属性タグ．

属性名	概要
host	通信先サーバのホスト名を指定する．
tcpport	TCP の通信に使用するポートを指定する．デフォルト値は 10000 である．
udpport	UDP の通信に使用するポートを指定する．デフォルト値は 10300 である．
spkindex	出力デバイスのインデックスを指定する．ここでいうインデックスとは，出力デバイスと対応付けされた識別番号のことを指す．

4.2.3　ファイル読み込み機能の実装

設定ファイルからパラメータを読み込むまでの大まかな処理の流れは以下の通りである．

1. パラメータ格納用変数の初期化
2. 設定ファイルを開く
3. タグ名を元にタグをパースする
4. パースした情報を格納し保持する

エラー処理を除く，パース処理の流れを図 6 に示す．

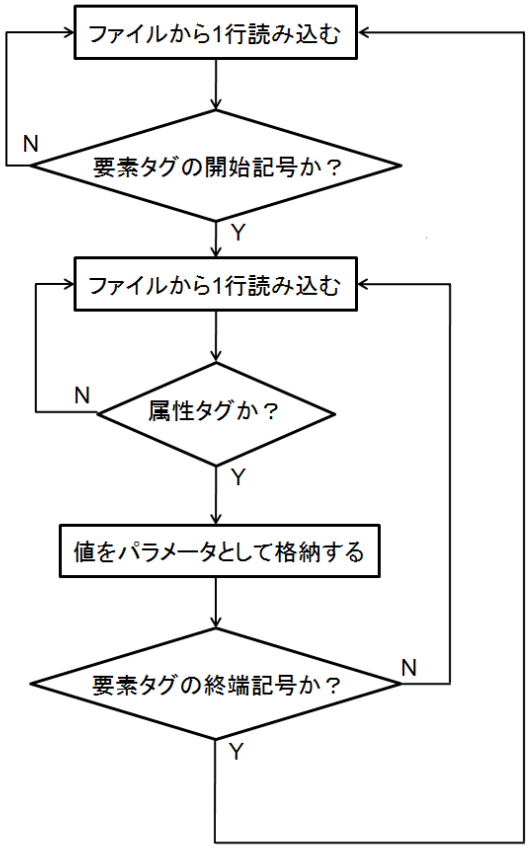

図 6　パース処理の流れ．

まず要素タグの開始記号を 1 行単位で検索する．開始記号を見つけたら，対応する終端記号を見つけるまで 1 行単位で属性タグのパースを行う．このとき，属性タグをパースして得られたパラメータは，パラメータ格納用変数に格納しておく．開始記号と対応する終端記号を見つけたら，次の要素タグの開始記号の検索を行う．以上の処理を繰り返すことで，設定ファイルからパラメータを読み込む．

4.4　コマンドインタフェース

この節では，プロセス起動時にコマンドライン引数としてパラメータを入力するための，コマンドインタフェースの設計と実装を行う．

4.4.1　コマンドオプションの定義

各プロセスの起動コマンドを以下に示す.

```
＜サーバプロセス＞
#./udpserver

＜クライアントプロセス＞
#./mic
```

これらの起動コマンドにコマンドオプションを付加することで，様々なパラメータをその都度入力できるようにする.

表 4 にクライアントに対して指定できるコマンドオプションを，表 5 にサーバに対して指定できるコマンドオプションを示す.

表 4　クライアントに指定できるコマンドオプション.

書式	概要
-L [latency]	ローカルラグ方式でサーバと通信する. latency に値を入力することでローカルラグフィードバック音の遅延量を指定することができる.
-E [latency]	拡張ローカルラグ方式でサーバと通信する. latency は-L オプションと同じ仕様を持つ. -L と-E オプションどちらも指定しなかった場合，通信制御を用いない通常の通信方式を用いる.
-c [filepath]	読み込む設定ファイルを指定する．filepath に読み込みたい設定ファイルのパスを記述する必要がある．デフォルト値は「mic.conf」である.

表 5　サーバに指定できるコマンドオプション.

書式	概要
-f [filename]	出力ファイル名を指定する．filname にファイル名を記述する必要がある.

4.4.3　コマンドオプション読み込み機能の実装

コマンドライン引数からパラメータを読み込むまでの処理の流れは以下の通りである.

1. コマンドライン引数を 1 つずつ読み込む
2. 読み込んだ引数の先頭文字が「-」なら次の手順へ
3. 「-」の次の文字でオプションを判別
4. オプションごとに定義されたパラメータを読み込む
5. 読み込んだパラメータを格納する

4.5　多地点接続機能

この節では，多地点間接続を想定した実験を行うための第 1 歩として，多地点間接続機能の設計・実装を行う.

4.5.1　多地点間接続機能の実装

図 7 に，3 地点間接続を想定したときの，演奏音の流れを図解する.

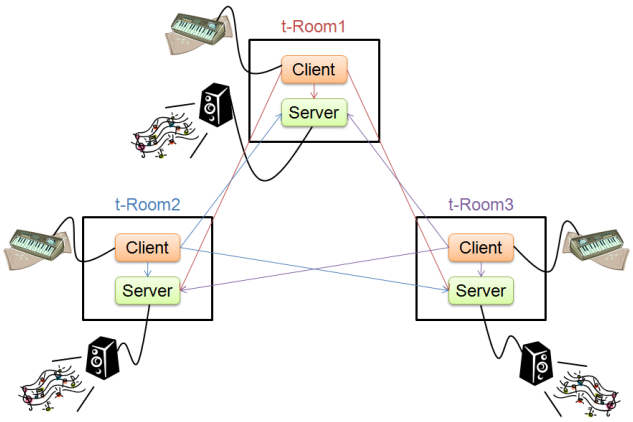

図 7　3 地点間接続時の演奏音の流れ.

3 地点接続を考えた場合，クライアントに入力された演奏音は，ローカル地点のサーバを含む 3 地点のサーバへと転送される．サーバは受け取った音を追加的な処理はなにも行わずにそのまま出力するだけなので，処理的には 2 地点間接続と変わらない．多地点間接続機能において重要なのはクライアントの挙動である．本稿では，図 7 に示されている様に，フィードスルーの転送処理を担当するスレッドにおいて，同じ演奏音を 1 度に各サーバに転送するという手法をとった.

多地点間接続を実装する上での，従来の音響サーバにおける問題点は，クライアント側で扱える TCP ソケットの数が 1 つしかないという点であった．TCP ソケットはサーバとのセッションを確立するのに必要であるため，多地点間接続を想定する場合，扱える TCP ソケットの数は接続するサーバの数だけ用意する必要がある.

また，それに合わせてサーバ側でも複数のセッションを扱えるように改良を施す必要がある.

4.5.2　接続地点数と出力デバイス数の関係

多地点間接続が可能になると，一度に接続できる上限数を設定する必要性が出てくる．この上限数について，本稿では 4 という値を設定した．これは出力デバイス数に基づいた数字であり，1 セッションに 1 デバイスが対応するように考えて定義した．TCP ソケットや UDP ソケットと，デバイスが対応すると言い換えることもできる．図 7 を例に考えたとして，今，t-Room1 に着目する．この地点のサーバが持つセッションの数は，ローカル地点のクライアントとのセッション，t-Room2 のクライアントとのセッション，t-Room3 のクライアントとのセッションの 3 つである．この時の接続地点数は 3 となり，各セッションに対して出力デバイスが割り当てられることになる.

5. システムの性能評価

実装した音響システムの実行には，ネットワークに由来する遅延に加えて，システムが稼働する PC とその周辺装置に起因する計算遅延が付随する．本音響システムを，ラグを制御する実験の土台とするには，この計算遅延等を含むシステム実行に伴う遅延全体を明らかにしておく必要がある．また，演奏実験結果を解析する際には，遠隔合奏の各演奏者の音を取り扱う複数台の音響システム間の同期，すなわち時計合わせもしておく必要がある．ここでは，これらの計算処理遅延の計測と時計合わせの結果を報告する．

5.1 処理遅延の計測

5.1.1 計測方法

今回計測した処理遅延は，演奏音が入力されてから出力されるまでの遅延である．今回実際に計測した計測方法を，通信制御を用いない場合を例として図 8 に図解する．

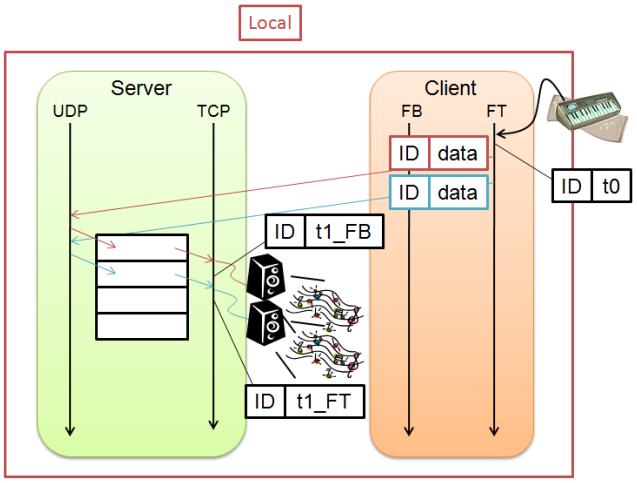

図 8　計測方法.

システム内に実装した計測手順を以下に示す.

1. 演奏音を取得後，現在時刻を標準出力に出力する
2. FT スレッド内のループカウンタを ID として設定する
3. 2 で設定した ID と現在処理中の演奏音を対応づける
4. ID は対応する演奏データにヘッダとして付加される
5. 4 で作成したパケットをサーバに転送
6. 出力時にヘッダとデータを分け，データを出力する
7. 演奏データ出力後，現在時刻を取得
8. 6 で分けたヘッダから ID を取り出し，7 で取得した現在時刻と同時に標準出力に出力する

図に示されているように，計測範囲は，1 地点のクライアントに対する入力から，同地点のサーバからの出力までである．t1_FB – t0 は，ローカルフィードバック音が入力されてから出力されるまでにかかる処理時間を表わし，t1_FT – t0 は，遠隔フィードスルー音が入力されてから出力されるまでにかかる処理時間を表す．ただし，本来遠隔フィードスルー音は，ネットワークを経由することによる転送遅延を伴う．そのため，同一環境のマシン同士で接続した場合，この方法で計測した処理遅延に，転送遅延を付加してやれば，理論上は遠隔フィードスルー音の入力から出力までの処理遅延が計測できる．なお，1 回で計測できる演奏データのサイズは 512byte である．

5.1.2 計測環境

以下に今回計測に用いた PC のハードウェアの諸元は以下の通りである．

- CPU：Intel®Core™2 Quad CPU Q6600 2.40GHz
- メモリ：3GB
- OS：Fedora 8
- サウンドカード：M-AUDIO Delta 1010LT
- 符号化方式：Lenear-16
- 標本化周波数：44.1kHz
- チャンネル数：2（ステレオ）
- 入力デバイス数：4
- 出力デバイス数：4

入力音には 120bpm の電子メトロノームを採用した．通信はローカル内のクライアントとサーバの間のみで行い，フィードバック音とフィードスルー音それぞれの，入力から出力までの処理遅延を計測した．通常方式，ローカルラグ方式，拡張ローカルラグ方式それぞれで行い，約 30 秒間計測を行った．この時，ローカルラグ方式，拡張ローカルラグ方式ともに，ローカルフィードバック音に 100ms のラグを付加するように設定した．約 30 秒間で約 11000 回分の入力があったため，方式ごとに得られた 11000 回分の計測結果を採用する．

5.1.2 計測結果

表 6 に得られた 11000 回分の計測結果の平均を，フィードバックとフィードスルーに記載する．

表 6　処理遅延計測結果の平均.

	フィードバック(msec)	フィードスルー(msec)
通常	5.071	4.345
ローカルラグ	104.400	5.708
拡張ローカルラグ	106.012	4.416

前述した通り，ローカルラグ方式と拡張ローカルラグ方式においては，フィードバックに 100ms のラグを付加するように設定している．そのため，ローカルラグと拡張ローカルラグのフィードバックに関して，理想的にはこの結果の平均から 100 を引いたものが，入力から出力までの処理遅延であると言える．しかし，ラグの付加精度に関しては未知であり，それを明らかにしない限りは，ローカルラグと拡張ローカルラグの結果の考察については推測の域を出ない．

通常方式の結果について，フィードバックの処理遅延のほうが大きいが，これは出力デバイスへの書き込み順が影響している．この音響システムにおいて，演奏音が

どの出力デバイスに書き込まれるかは，その出力デバイスに対応付けられたスピーカ番号によって判別される．実験時，フィードバックをスピーカ番号 1 から出力し，フィードスルーをスピーカ番号 0 から出力していた．実際の処理において，出力デバイスへの書き込みは，スピーカ番号の 0 から順番に行われる．以上のことから，フィードスルーの書き込み後にフィードバックが書き込まれているためにこのような結果になったと考えられる．

5.2 音響システム間の同期

実験結果からローカルラグの効果を定量的に検証するには，ローカルラグを用いた時と用いなかった時で各演奏者の演奏のずれを計測し，その差を調べればよい．この時，実際に比べなければならない音は，ローカル地点における演奏者自身が発した演奏音（ローカルフィードバック音）と，同じくローカル地点における遠隔地の演奏者が発した演奏音（遠隔フィードスルー音）である．そのため定量データは，各地点において，ローカルフィードバック音と遠隔フィードスルー音をそれぞれ録音する必要があり，この音響システムは上述した要件を満たす機能を持っている．

これらの録音データの比較分析を行う際には，それぞれのデータの送信源である音響システムの時刻，あるいは時間軸の基準点を合わせておく必要がある．しかし，PC の時計を正確に合わせることは必ずしも容易ではなく，なんらかの同期の基準の取り方の工夫が求められる．

5.3 音響システムにおける同期の取り方

図 9 に，音響システムにおける同期の取り方を図解する．

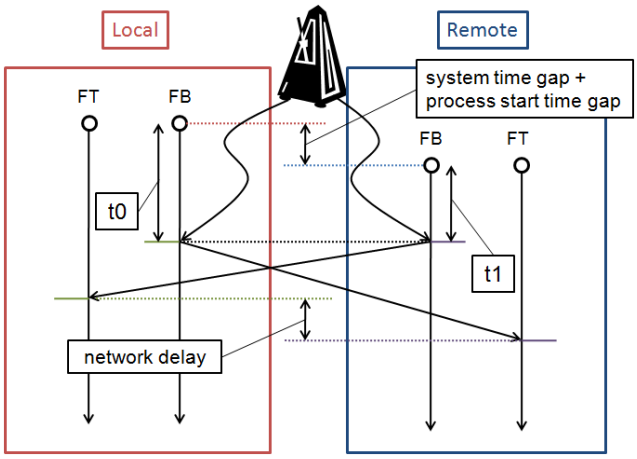

図 9 音響システムにおける同期の取り方．

図中，Local 内の FT，FB はそれぞれ，Local 側で録音された，Remote 側からのフィードスルー音と Local 側のフィードバック音のファイルである．同様に，Remote 内の FT，FB は，Remote 側で録音された，Local 側からのフィードスルー音と Remote 側のフィードバック音のファイルである．録音ファイルの作成タイミングは，サーバプロセスの起動タイミングによって異なるため，それを考慮して，各地点でファイルの作成タイミングがずれている．このずれは，図中，system time gap + process start time gap と表されている．これは，録音ファイル作成タイミングのずれが，システム時間のずれとプロセス起動時間のずれによって生じるものであるということを示している．

音響システムでは，1 つの電子メトロノームからの入力を同期の基準として用いる．1 つの電子メトロノームからの入力を各地点へと分配することで，入力タイミングの同期は確実に保証される．

実際にローカルラグの効果を検証するには以下のように行う．まず，比較対象となるファイルは Local 内の FB ファイルと FT ファイルとする．この理由については 5.2 で述べた．次に，FB ファイルの 1 音目と FT ファイルの 1 音目を合わせる．すなわち，この音を時間軸の基準点としてファイル全体をシフトさせる．この音を時間軸の基準点とすると，以降の音は全てその音を基準にして比較することができる．

さらに，入力タイミングが同時であることが保証されているので，ファイルを作成した詳細な時間をプログラム内で取得しておくことで，図中の t0，t1 の値がわかる．t0 と t1 の差をとれば，ファイル作成時間のずれの時間量がわかる．この時間量がわかれば， Local 側の FT ファイルと Remote 側の FT を比較するというように，別地点の録音ファイルを比較することも 可能になる．

5.4 音響システムの同期の取り方の精度

先程，比較対象ファイル（同一地点で録音された FB ファイルと FT ファイル）の 1 音目を揃えることで，そのファイル全体としての比較が行えると述べた．しかし，本当に 1 音目を揃えるだけでファイル全体が比較できるのか，また，比較できたとしてどの程度信用できるのかは，実際に比較し，その精度を確かめなければ判断することはできない．

そこでこの節では，実際に 1 音目を揃えるだけでファイル全体の比較が行えるのかどうか，また，できたとしてどの程度の精度があるのかを調査する．調査手順は以下の通りである．

1. 図 9 のように，各地点に接続された 1 つの電子メトロノームの音のみを入力として，録音を行う
2. Local 側 FB ファイルと FT ファイルの全ての音について波の立ち上がりの時間を求める
3. Local 側 FB ファイルの 1 音目の波が立ち上がる瞬間の時間から，Local 側 FT ファイルの 1 音目の波が立ち上がる瞬間の時間を引いた値を求める
4. 2 で求めた時間の内，FB ファイルの時間それぞれから 3 で求めた値を引く
5. 4 で得られた値の平均をとる

まず，手順 2 でフィードバックとフィードスルーの，録音ファイル内での時間差を計測する．次に，手順 3 で FB ファイルと FT ファイルの 1 音目の差をとり，ずらす量を求める．手順 4 では，手順 3 で求めた量だけ FB ファイルの時間をずらし，手順 5 では，手順 4 で得られた値を統計的なデータに変換している．

以上の手順で構成される調査を，通常方式，ローカルラグ方式，拡張ローカルラグ方式それぞれで 1 度ずつ行っ

た．調査結果を表 7 に示す．計測は，3.1 で示されている環境と同じ環境を持つ PC2 台を用いて，それらを LAN 上で接続した．

表 7　調査結果.

	考慮しなければならないずれ(msec)
通常	0.029
ローカルラグ	0.037
拡張ローカルラグ	0.020

調査結果から，5.3 で述べた方法で Local 地点における，各演奏者の演奏音（この場合は 1 つの電子メトロノームからの入力）のずれの比較を行うと，全ての方式において誤差 0.04ms 未満という調査結果を得た．この誤差は，この同期の取り方を用いて録音ファイルを比較した場合の，不可避の遅延であるとみなすことができる．不可避の遅延はあるものの，その値はどの方式も 0.04ms 未満の誤差でしかないため，この同期の取り方を用いれば，十分な精度でローカルラグの効果の検証を行えることが期待できる．

6. まとめ

本稿では，音響的ローカルラグの研究を促進させるため， 研究の土台となる音響システムの性能向上を目指して，その改良を行った．

具体的には，これまで個別実装されていた 3 種類のクライアントプロセスを統合し，プロセスの動作を制御する設定ファイルの定義とその読み込み機能の追加を行った．さらにコマンドインタフェースの追加と，多地点接続機能も追加した．

今後のさらなるシステムの改良を考慮し，システムの性能を評価するための手法を提案し，実装した．さらに，実験から得られるデータの精度を向上させるため，新たな同期基準を定め，その原理について紹介した．

システムの性能評価において，今回はローカル地点のみの処理遅延の計測しかできなかった．今回計測した処理遅延は，確かにシステム単体の性能評価に用いることはできるが，本来の実験環境は 2 地点間以上であり，その場合の全体の性能評価については適用できない．そのため，今後は 2 地点間以上を接続したときの評価尺度を探求し，さらなるシステムの性能向上に繋げたい．

また，音響システムの同期の取り方について述べ，その方法を用いて録音ファイルを比較したときの比較精度について計測した．計測結果から，この同期の取り方を用いて録音ファイルの比較を行えば，ローカルラグの効果を検証するにあたって十分な精度を得られることがわかった．今回は同一地点で録音したフィードバック音とフィードスルー音のみを比較したが，別々の地点のファイルを比較することで得られる知見が何かあるかもしれない．そのため，今後は，各録音ファイルの関係性について深く考察してみる必要があると考えられる．

謝辞：本研究は，t-Room に関する研究の一環として行われた．NTT コミュニケーション科学基礎研究所の平田圭二氏や青柳滋己氏をはじめとする t-Room 研究グループの皆様には多くのご助言を頂いた．著者一同，改めて御礼を申し上げる次第です．

参考文献

[1] K. Hirata, et al.; “The t-Room: Toward the Future Phone,” NTT Technical Review, Vol. 4, No. 12, pp. 26-33 (2006).
[2] D. Stuckel, et al.; “The Effects of Local-lag on Tightly-Coupled Interaction in Distributed Groupware,” Computer Supported Cooperative Work, pp. 447-456 (2008).
[3] 入江洋介, 他: “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, Vol. 2009-GN-73, No. 3 (2009. 11).