# オーバーレイマルチキャストを利用した擬似ライブ録音システムの試作

2004MT020 橋口 雅一　　2004MT042 川井 一希
指導教員　後藤 邦夫

## 1　はじめに

現在，通信速度の増加により，ネットワークを通しての音楽の販売やリアルタイムで音楽やライブ映像の視聴が可能となっている [4]．しかし，リアルタイムで演奏をやり取りできるシステムは存在しない．これは送受信時に起こる遅延やパケットロスが原因である．このような中で，離れた場所にいても音楽の演奏，視聴を共有できるシステムがあると良いと考え，複数人に同時に音楽を配送できるオーバーレイマルチキャストを用いた遠隔擬似ライブシステムを試作する．

本研究では，2006 年卒業論文 [2] として近藤，中野が考案した複数人でネットワークを介し，リアルタイムで演奏データを一人ずつ送受信し，違う場所にいてもそれぞれの音楽を聞きながら演奏できる擬似ライブ録音システムを改良する．

改良点は，オーバレイマルチキャストを用いて複数人に同時に演奏データを送信，受信させることである．さらに，演奏をリアルタイムに視聴できるリスナーという新たな役割を追加する．また，このアプリケーションを使い易くするために，ライブラリ Qt[3] を用いて，GUI を追加した．Qt については第 3 節のシステムの実現で，詳しく説明する．

システムの構築，考案は共同で行い，そのなかでも特に，橋口は Qt での GUI プログラミングを，川井は既存録音システムのプログラム改良部分を担当した．

## 2　システムの概要

この節では，既存の録音システム及び，本研究で提案するシステムを説明する．

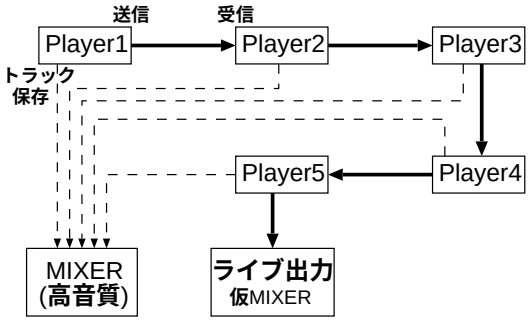


図 1　既存録音システムの送信形態

既存の遠隔擬似ライブ録音システムは，C++ で実装しており，録音や送受信はスレッドを用いて同時に行われる．player1 はリズム楽器を担当し，演奏を録音すると同時に player2 に送信する．２人目以降は受信と送信を同時に行い，受信した音源を再生しつつ演奏し，自ら演奏している音も乗せて次の演奏者に送る．この作業をくり返し，最終的に全ての演奏がミキサーに送信される．このように，別々の場所で，同時間帯に演奏をし，順に音を重ねることで，曲の録音が可能となるシステムである．我々は，このシステムを拡張する．

### 2.1　オーバーレイマルチキャスト

既存のシステムをさらに便利にするため，複数人に配信する機能を追加する．これは，同時間帯において，このシステムに参加できる人数を増やすことを可能にする．既存の IP マルチキャストでは，グループアドレスに送信することで，ルータが自動的に参加者全員にデータを送信する．しかし，ネットワークがマルチキャストに対応する必要があり，誰もが簡単に利用はできない．そこで，アプリケーションレベルで明示的にアドレスを複数与えることで，複数人に配信が可能となる．送信経路は，演奏データの送信順序となるため，参加者間で互いに情報を交換し，事前に打ち合せができる環境があるとして経路を設定する．

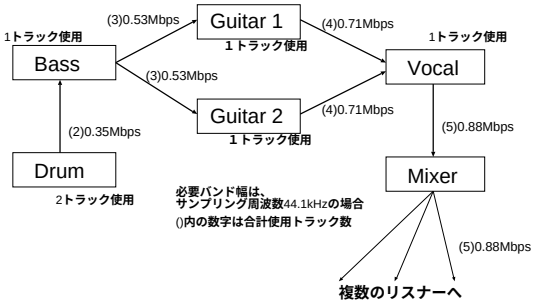


図 2　本研究の送信形態

本研究のシステムでは，同じ時間に違う演奏者が同じ演奏データを聴きながら演奏，録音することを可能にする (図 2)．例えば，Drums と Bass までの録音は，既存のシステムの送信形態をとり，そこからは，オーバーレイマルチキャストを利用し，同時に複数に送信する．そのデータを受信した複数のギター演奏者がそれぞれ，Drums と Bass の音を聴きながら演奏，録音をする．そして，この複数の演奏をふたたびひとつに合流させ，ひとつの曲として出力ができるようにする．これにより，既存のシステムにはない，新しい楽しみ方ができる．

### 2.2 書き込み速度の遅れへの対処

さらに，録音機材によっては，データを書き込む際に遅れが生じ，演奏にズレが起きてしまう．そこで，その遅れを考慮し，録音のズレをなくす処理を追加した．

### 2.3 GUI の追加

使いやすさを得るために，録音システムのインタフェース部を実装する．既存の遠隔擬似ライブ録音システムは，コマンドベースのアプリケーションであり，スレッドの開始や終了などはすべて手で入力する必要がある．また，必要なパラメータはあらかじめファイルに書き込んでおく必要がある．本研究では，システムの操作，パラメータの更新を容易に可能にするインタフェース部を作成し，これがシステムのコントロールを行う．GUI ツールキットである Qt を使用し，使いやすいインタフェース部を作成する．

## 3 録音システム

ここでは，既存の遠隔擬似ライブ録音システムに追加，または改良した部分について説明する．ここでの録音システムとは，GUI 部とは別のアプリケーションである．

### 3.1 送信データ

演奏データは UDP 通信方式を取り，ペイロードは 1400byte 程度であるとし，1 トラックは 100byte とすることで，1 パケットに最大 14 トラックまでの演奏データを格納できる．送信データは，ヘッダ部とデータ部に分かれている (図 3)．

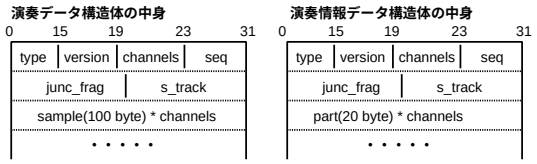


図 3 送信データの構造体の中身

ヘッダ部には，再生時にはチャネルを指定する必要があるため，channels を持つ．パケット番号 seq は，パケットロスや入れ換わりへの対処，書き込みのズレ修正のために使われる．本研究では，合流を行うため，そのデータが分岐後のデータであることを示す `junc_frag` を追加する．`s_track` は，分岐後に使用した使用トラック数である．

データ部である sample には，各トラックの演奏データを格納する．part には，各トラックはどの楽器が使用しているかというパート情報を格納する．演奏データパケットの場合は sample を，演奏情報パケットの場合は part を使用し，type によってそれを識別する．このデータ部は union で宣言し，sample と part は同じ領域である．

### 3.2 送受信処理

同時に複数人に演奏データを送信するために，sendto() を用いて，データの送信の度に宛先を指定する．演奏を録音し送信する際，一人目に送信，二人目に送信を続けて繰り返す．これにより，アプリケーションで自動的に複数同時送信が可能となる．

複数から同時に受信する際には，recvfrom() を用いて，受け取った相手を識別し，それぞれ別のバッファにためる．ある程度たまったら，それぞれのパケット番号を比較し，番号が同じである場合は演奏データの合成をして保存する．パケット番号の比較は，2 人から受け取る場合，両方からのパケットをある程度ためた後に行うため，片方からのデータがまったくこない場合は，もう片方をさらにためるよう指定する処理を追加した．

演奏データの合成処理は，ふたつのパケットの演奏データの中身を合成したい場合，パケットひとつ目の空きトラック領域に，ふたつ目の演奏データを挿入することで実現した．

### 3.3 書き込み速度の調整

PC の性能や使用する録音機材によって，録音の書き込みにかかる時間は異なる．機材によっては，しっかりと再生した音に合わせて演奏していても，録音した演奏データを聞くと，大きなズレが生じていることがある．マイクに音を入れてから，実際にデータが書き込まれるまでに時間がかかる．そこで，再生したパケットをためておき，書き込まれる瞬間のタイミングで録音することで，ズレを修正できる (図 4)．1 パケットは，4.5 ミリ秒の演奏データを持っているため，4.5 ミリ秒単位で調整ができる．

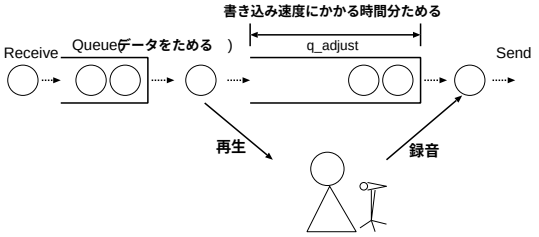


図 4 書き込み補正の流れ

本システムは，オーディオ装置から 200 サンプルずつ書き込み，読み込みをする．そこで，再生した音を直接録音し，そのサンプル番号の差をとることで，ズレのサンプル数がわかる．それをサンプリング周波数で割ると，ズレ時間が測定できる．

## 4 GUI アプリケーション

この節ではライブラリ Qt を用いて作成した GUI アプリケーションについて説明する．Qt とは Trolltech 社により開発された C++ で記述された GUI ライブラリ群であり，様々なアプリケーションの開発に使われている．Qt で作るアプリケーションの主な役割は，動作設定の変更，録音と送受信の開始，終了，ボリューム変更，エフェクト設定などをわかりやすく操作できるようにすることである．

### 4.1 動作設定の変更，録音と送受信の開始，終了

まず，録音システムと通信をするために，実行されている録音システムのプログラムのあるパソコンの IP アドレスと録音システム起動時に設定したポート番号を入力し，‘Connect‘ボタンを押すことで，録音システムと接続する．

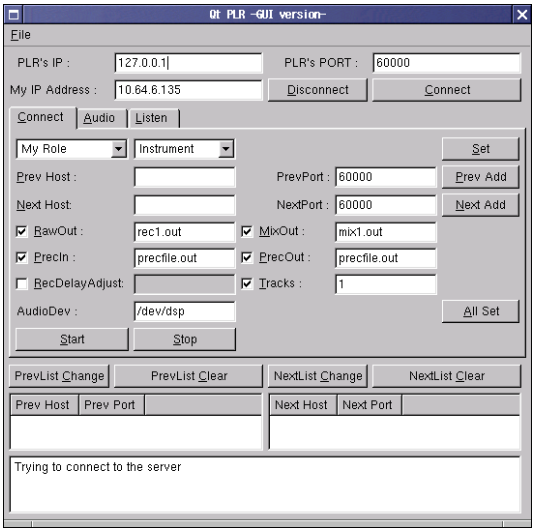


図 5　GUI 画面の例

動作設定は，録音システムの動作に必要な値 (自分の役割やデバイス設定) の変更を行う場合に，それらの値を録音システムに送信する．必要な項目を GUI 画面から入力したあとに ‘ALL Set‘ボタンを押すと，MyRole : ORIGIN ; Instrument : Drum ;.. という様に，一行の文字列で録音システムへ送信する．送信方法は，TCP Socket を使い，TextStream 形式で送信する．録音や送受信のスタート，ストップは，それぞれボタンを作成し，ボタンが押されたときに合図を送信する．

### 4.2 再生時のボリューム，エフェクト関係

ボリューム変更は，スライダー (つまみ) を作成し，スライダーが離された時に変更内容を録音システムのプログラムに送信する．各トラック (パート) を指定して，音量やエフェクトをそれぞれに変更を可能にするため，ラジオボタンを用いることで，選択されているトラックに限定して調整が可能である．さらにミュートボタンを作成し，これをチェックすることで，ボリュームを 0 にすることができる．エフェクト設定は，チェックボタンを作成し，チェックがついている場合は’ON’，外されている場合は’OFF’ とする．’OFF’ の場合は，トーンコントロール，リバーブが変更できないように制限をする．トーンコントロール，パン，リバーブはダイアル式のつまみとしてレイアウトし，使い易さを考慮する．

### 4.3 現在の演奏情報の表示

録音システムでは，どのトラックがどの楽器の演奏かを次の演奏者に知らせるために，演奏データとは別に演奏情報データを 9 秒に 1 回送信する．この情報を録音システムが受信した場合，GUI アプリケーション側で，’Info Add’ ボタンが押されたときに，その情報を GUI 画面に表示させることができる (図 6)．

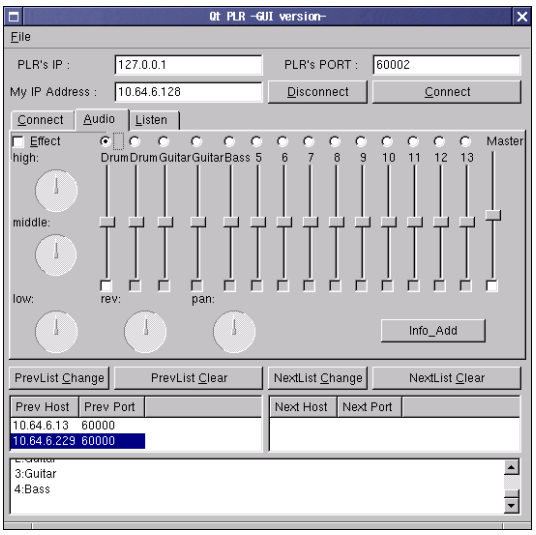


図 6　再生時のコントロール

### 4.4 設定値のセーブとロード

動作設定の際に設定した値を，ファイルとして保存可能にする機能を追加した．’Save’ を選択し，ファイル名を指定し，そのとき持っている値を保存することができる．また，’Load’ でファイルを開くことも可能である．これにより，よく使う設定などを保存しておくことで，値の入力の手間を最小限にしている．

### 4.5 リスナー

演奏には参加しないが，視聴のみできるリスナーについて説明する．リスナーに必要な情報としては，オーディオデバイスの設定，受信時のポートの設定である．この二つを入力し，あらかじめ作成しておいたリストファイルを開くことにより，宛先リストが録音システムへ送信される．

## 5 実験・評価

ここでは，作成したアプリケーションの実験と，その評価を行う．

### 5.1 実験方法

実験方法として，ノート PC2 台，デスクトップ PC3 台，合計 5 台を使用して実験を行った．演奏に使用した楽器はドラムマシン，ベース，ギター，マイクの 4 つである．実ネットワークを模倣するため，デスクトップ PC3 台のうち 1 台は，遅延・ロスを起こすネットワークエミュレータ GINE2R0.7[1] として動作し，他のコンピュータは，ネットワークエミュレータを経由して通信を行う (図 7)．すべての参加者が Qt アプリケーションから録音システムを操作するとし，すべての操作を Qt アプリケーションから行うとする．

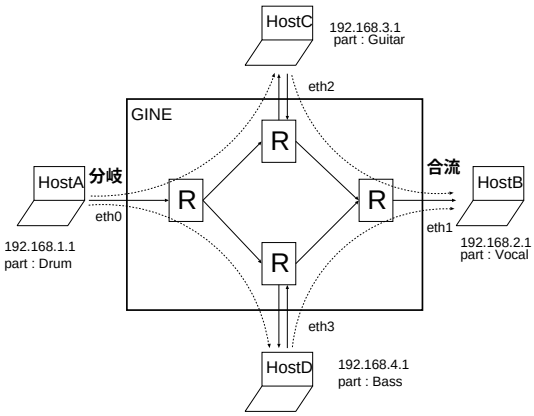


図 7　実験構成

### 5.2　評価方法

評価のポイントは以下のとおりである．

1. 演奏データの複数同時送信ができる
2. 演奏データの合流ができる
3. 書き込むタイミングの調整ができる
4. パケットロスの対処
5. 遅延の対処
6. 演奏情報パケットの送信と受信，保存ができる
7. 以上の操作を含め，すべての操作を Qt アプリケーションから行える

### 5.3　結果

各 Host の演奏が HostB で聞けたので，分岐，合流はできているとわかる．HostB で視聴する際，音のずれが生じたので，書き込むタイミングを調整したところ，音のずれも解消された．また，演奏情報を GUI 画面上で表示できたので，演奏情報パケットの送信と受信，保存ができるとわかる．これにより 1，2，3，6，7 は成功している．次に，同じ実験構成でロス率を 1% から 20% に変化させて，実験を行った．結果は以下の通りである．

表 1　ロス率の比較

| ロス率 | リズム | 音質 |
|---|---|---|
| 1% ～ 2% | 乱れない | ほとんど普通に聞ける |
| 3% ～ 8% | 若干乱れる | 多少雑音が聞こえる |
| 10% ～ 13% | 乱れる | 音がかなり乱れる |
| 14% ～ 17% | 乱れる | 悪い |

表 2　テンポによる比較

| 演奏速度 | ロスがある場合のリズム |
|---|---|
| ♪ 120 | リズムがとりにくくなる |
| ♪ 60 | 問題なし |

上記の結果より，ロス率が 1，2% 程度であればリズム，音質ともに問題なくシステムが実行される．10% 以上の場合はリズムも乱れ，音質も悪く，演奏が困難になった．また，書き込むタイミングの調整を行い，録音のズレは解消された．外付け USB オーディオ装置では，約 0.10～0.15msec 程度の遅れがあり，調整値は 30 前後でズレを解消することができた．

片道のみの遅延，両方の遅延のいずれの場合もデータを受信，合成することができた．ただ，100msec 以上の遅延がある場合は，合成されるまでの時間が非常に長くなった．これは，片方の Host からしかデータが送られて来ない場合には，もう一方の Host からデータが送られてくるまで，データを溜め続けるからである．国内の遅延は 50msec 程度，国外の遅延は 300msec 程度なので，実験結果から，国内，国外問わず問題なく利用できることがわかる．

## 6　おわりに

本研究では，オーバーレイマルチキャストを用いて，複数人に同時に演奏データを送信，受信が可能となり，途中で分岐しても，再び合流することにより 1 つの曲を作成できたり，分岐した状態でアレンジの異なる二つの曲を作成することが可能となった．また，Qt により，GUI アプリケーションから録音システムを操作することができるようになった．基本的な動作設定をしたり，ボリュームやトーンのコントロールをリアルタイムで変更することが可能となった．設定値のセーブやロードも可能となり，目標である使い易いインタフェースが実装できた．しかし，現時点では録音システムの方ではリスナーの処理は実装していない．また，自分の IP アドレスや START，STOP の合図をプレイヤー間でやりとりするための，チャット機能を組み込むことも有効であるだろう．

今後このアプリケーションを改良するとすれば，上記のことを追加すると，さらに便利なアプリケーションになると考えられる．

## 参考文献

[1] Ihara,A., Murase,S. and Goto,K.: IPv4/v6 Network Emulator using Divert Socket., Proc. of 18th International Conference on Systems Engineering(ICSE2006)., Coventry, UK, pp.159-166(Sep.2006).

[2] 近藤 美希, 中野 好絵: 音楽の遠隔擬似ライブ録音システムの試作, 卒業論文, 南山大学数理情報学部情報通信学科 (2006).

[3] trolltech 社: Qt Reference Documentation(accessed May 2007). http://doc.trolltech.com/4.0.

[4] Online Recording Community: digital musician net(accessed April 2007). http://www.digitalmusician.net/.