Vol.2012-MUS-97 No.1
2012/12/21

音楽演奏システムの分散ネットワーク化

中島武三志[†1]　　菅野由弘[†1]

電子楽器の研究では，演奏者の行為と MIDI などの演奏情報とのマッピングや演奏者に対する何らかのフィードバックの呈示をおこなう新たな演奏インターフェイスの制作がこれまでに数多くおこなわれてきた．しかし電子楽器はこうした演奏インターフェイスだけではなく音源やエフェクター，スピーカなど様々な機能が有機的に結びついた総体である．本研究では，複数の端末に分散されたこれらの機能をネットワークで結び，音楽演奏における演奏データやオーディオデータをリアルタイムで送受信するシステムと，ネットワーク構成を動的にコントロールするインターフェイスの制作について述べる．

Distributed Networking of Musical Performance System

MUSASHI NAKAJIMA[†1]　YOSHIHIRO KANNO[†1]

Numerous electronic musical devices have been innovated where as typically they map players' performances to musical information such as MIDI or offer some sorts of feedback to players. However, electronic instruments are related not only to performance interfaces but also are correlated to other various functions such as sound sources, effects or speakers. In this study, we propose a system, which conducts playout and uplink of performance and audio information, networking those functions on plural terminals, and at the same time, an interface that enables dynamic controls of the network's structure.

1. はじめに

電子楽器の研究では，演奏者の行為と MIDI などの演奏情報とのマッピングや，演奏者に対する何らかのフィードバックの呈示をおこなう新たな演奏インターフェイスの制作がこれまでに数多くおこなわれてきた．しかし電子楽器はこうした演奏インターフェイスだけでは演奏できない．実際には音を作り出す音源やエフェクター，スピーカなど様々な役割を持つ要素が有機的に結びつくことにより，ひとつの電子楽器が形作られる．

電子楽器とは異なる従来の生楽器においても，様々な役割を持つ要素同士が有機的に結びついている点では共通している．しかし単身での演奏が前提となっていることや物理的な制約により，生楽器の持つ各機能は連結，あるいは近接せざるを得ない．その結果楽器の各機能は一点に集中し、楽器は物理的に連続，あるいは近接したひとつの物体として認識される．この従来の楽器における制約から，楽器は“基本的に単身で演奏される，一点に集中した物体”というイメージが定着していると思われる．

電鳴楽器や電子楽器ではこうした制約は減少し，各々をケーブル等で連結することでひとつの楽器としての機能を形成している．しかしケーブル等で連結された各機能の総体がひとつの楽器として認識されていない場合も多く存在する．エレクトリックギターを例に挙げるならば，通常は楽器本体をアンプに接続して演奏されるが，アンプは必ずしも楽器の一部とは捉えられていない．エレクトリックギターは楽器であるが，アンプは楽器の音を増幅する別の装置として認識される．先に述べた“基本的に単身で演奏される，一点に集中した物体”というイメージから，電鳴楽器や電子楽器においては実際に演奏者が演奏をおこなう部分，言い換えればインターフェイス部分のみを“楽器”と捉え，その他の機能が物理的に分離されると，それらは別のものとして認識される．

音楽之友社，音楽中辞典によると楽器は広義には「音を出すための道具（音具）の総称」と定義されているが，この定義には曖昧な部分がある．そこでこうした楽器に対する従来の認識を拡張する概念として，演奏者の演奏が最終的に空気の振動としての音に変換され耳に届くまでの過程を媒介する機器の有機的な結びつきの総体を“音楽演奏システム”と呼ぶことにする．これにより空間的に一点であった従来の楽器の概念は，空間を占める 3 次元のネットワークとなり，単身による演奏という前提から解放される．

空間を占める 3 次元のネットワークに楽器を拡張することにより，空間内における音の位置操作という，従来の楽器ができなかった表現の可能性が生まれるほか，複数の人間が互いに協調ながら音楽を創りだす形態（Collaborative Music Composition[1]）を支えるしくみともなり得る．

また，従来の楽器においては各機能の結びつきは基本的に固定されている．先のエレクトリックギターの例では，一度楽器本体をアンプに繋いでしまえば，演奏中に別のアンプに繋ぎ替えることは通常おこなわれない．というのも演奏中にこうした各機能の連結を繋ぎ替えることの必要性があまりないことや，各機能の連結には手間がかかるためであると考えられる．

このような各機能の結びつきを構築する行為は，一般的に演奏という言葉で表される行為と異なるため，ここでは“メタ音楽演奏”と呼ぶことにする．従来の楽器では，メタ音楽演奏は手間がかかり，即時性に欠けることから，“演奏”の文字を当てるのはやや不適切で，“セッティング”と捉える方が自然であるが，演奏中に即時的に各機能の結びつきを組み替えられるようになれば，“セッティング”ではなく“演奏”と捉えることが可能となる．つまり音楽演奏システムの各機能の結びつきを動的に組み替えるメタ音楽演奏が可能になれば，これも音楽表現のひとつの方法となり得ると考えられる．

また，コンピュータ技術とともに発展してきたネットワーク技術が，電子楽器の演奏にも応用されるようになっている．その例として，ネットワークを介して MIDI と同様の音楽演奏データを送受信する通信プロトコルである RMCP[2]や OSC(Open SoundControl)[3]などが考案されている．本研究では、複数の端末に分散された電子楽器の機能をネットワークで結び，音楽演奏における演奏データやオーディオデータをリアルタイムで送受信するシステムと，ネットワーク構成を動的に構築するインターフェイスの制作について述べる．

2. 音楽演奏システム

2.1 音楽演奏システムの構成

音楽演奏システムは、広義には演奏者の演奏としての行為が最終的に演奏者自身または聴衆の耳に音として知覚される過程を媒介するもの全てを指すが，ここでは特に電子回路による演算をおこなう装置のみを主な対象とし，これらの各機能を大きく以下の５つのモジュールに分類する．

2.1.1 演奏インターフェイス

演奏インターフェイスは，演奏者の操作と直接関わる部分である．この部分で後述する各モジュールのパラメータを制御する．各モジュール用の演奏インターフェイスは，必要に応じて複数用意される．

2.1.2 シーケンサ

シーケンサは，演奏インターフェイスによって設定されたパラメータを用いて，以下の各モジュールに対して自動的に演奏データを生成する部分である．

2.1.3 音源モジュール

音源モジュールは，演奏インターフェイスやシーケンサから演奏データを受け取り，それをもとにオーディオデータ（リニア PCM）を生成する部分である．

2.1.4 エフェクトモジュール

エフェクトモジュールは，音源モジュールで生成したオーディオデータにエフェクトを施す部分である．

2.1.5 再生モジュール

再生モジュールは，音源モジュールまたはエフェクトモジュールからオーディオデータを受け取り，実際にスピーカから再生する部分である．各モジュールの関係図を図１に示す．

2.2 音楽演奏システムの分散ネットワーク化

これらのモジュールを必要に応じて複数の機器に分散化し，ネットワークを通じて機器同士の通信をおこなう．各機器は最低ひとつ以上のモジュールを持ち，ネットワーク通信機能を有するものとする．

それぞれのモジュールはネットワーク上で一意に識別できるアドレスを持ち，データ通信の相手先を，後述するメタ音楽演奏システムを用いることで演奏者が自由に決定できる．また通信方法は物理的制約の少ない無線 LAN を主に用いる．

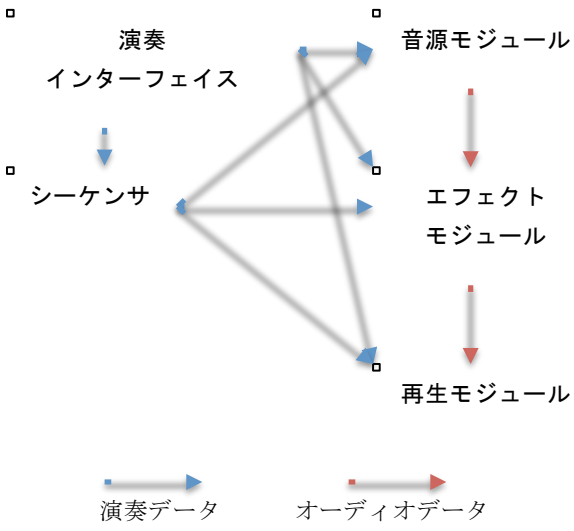

図 1 各モジュールの関係図

3. メタ音楽演奏システム

3.1 音楽演奏システムの動的構築

従来の楽器に関しては，楽器の持つ各機能を連結する行為は楽器の製作時か，あるいは演奏の前にあらかじめおこなわれ，演奏中は各機能の連結を組み替えることは通常おこなわれない．ここでは通常の演奏と並行して音楽演奏システムを動的に構築する行為をメタ音楽演奏と呼び，これをおこなうシステムをメタ音楽演奏システムと呼ぶことにする．

3.2 メタ音楽演奏システムの構成

音楽演奏システム内の各モジュールは，ここではソフトウェア上の架空の実体として存在するものとする．モジュールの生成とネットワークの動的構築をおこなうために，

†1 早稲田大学基幹理工学研究科表現工学専攻
The Department of Intermedia Art and Science, Waseda University

メタ音楽演奏システムを以下の４つの機能に分類する．

3.2.1 メタ演奏インターフェイス

メタ演奏インターフェイスは，ネットワーク内のモジュールを表すシンボルからなる．はじめに，後述するモジュールリストやモジュールマネージャ，モジュールコーディネータと連携してモジュールを表すシンボルを生成する．次にシンボル同士を何らかのかたちで連結させ，シンボルの連結情報を後述するモジュールコーディネータに送信することで，実際のモジュール同士の連結をおこなう．

3.2.2 モジュールマネージャ

モジュールマネージャは，端末内にあるモジュールの管理をおこなう部分である．モジュールマネージャは後述するモジュールリストに対し，自身の持つモジュール情報を送信する．モジュールリストからモジュールの生成依頼を受けると新たにモジュールを生成し，モジュールのアドレスをモジュールコーディネータに送信する．

3.2.3 モジュールリスト

モジュールリストは，各端末のモジュールマネージャから受信したモジュール情報を一覧表示し，メタ演奏インターフェイスと連携してモジュールマネージャに対しモジュールの生成依頼をおこなう．

3.2.4 モジュールコーディネータ

モジュールコーディネータは，生成されたモジュールのアドレスを管理し，メタ演奏インターフェイスから受け取ったシンボル同士の連結情報をもとに各モジュールに対してデータの送信先アドレスを送信する．メタ音楽演奏システムの構成図を図２に示す．

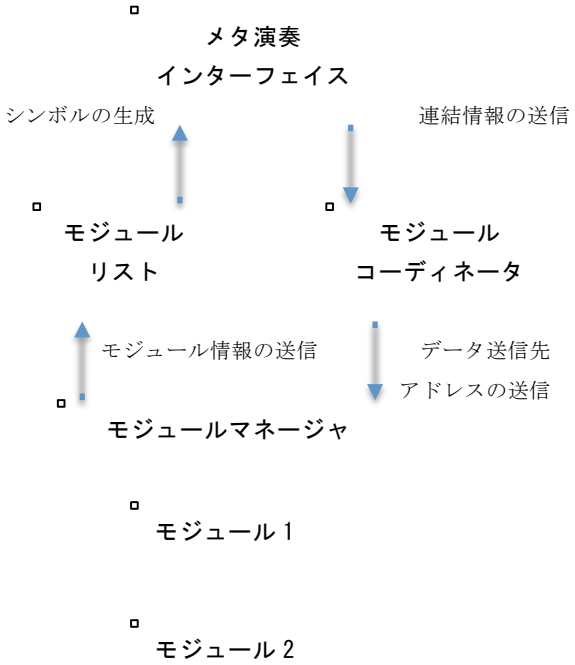

図 2 メタ音楽演奏システムの構成図

4. システムの実装

4.1 概要

音楽演奏システムの各モジュールを C++クラスとして実装した．またメタ音楽演奏システムのモジュールマネージャ，モジュールリスト，モジュールコーディネータを含む CUI のプログラムと，タイル型メタ演奏インターフェイスを制作した．

4.2 通信プロトコル

各モジュールでの演奏データ，オーディオデータのやり取りとして OSC を用いた．OSC はコンピュータやシンセサイザー, マルチメディア機器間の通信プロトコルであり，主に UDP/IP をベースに実装され電子楽器の制御やマルチメディア処理などの用途に用いられている．また OSC は URL 形式の命名規則が特徴である．リアルタイム性が必要とされるこのシステムにおいては発音の遅延は致命的であり，オーディオデータの少々の欠損はやむを得ないとしてこれに適した OSC を用いて通信をおこなうことにした．

実装の際には C 言語のライブラリ liblo[4]を使用し, OSC アドレスと端末の IP アドレスを組み合わせて各モジュールを一意に識別する．

4.3 各モジュールの実装

モジュールはメタ音楽演奏システムによって OSC アドレスを付与される．この OSC アドレスは”/Tile1/DAC”といった形式になっている. 各モジュールはこの下位に “/Data” と “/Stream” というアドレスを持ち，“/Data” では演奏データの受信処理を，“/Stream” ではオーディオデータの受信処理をおこなう．

モジュールが２つ以上の入出力をおこなうときは，入出力データの順序で区別する．データ順序と処理内容は各モジュールに依存する．

4.3.1 AudioClock モジュール

AudioClock モジュールは，オーディオデータ処理のタイミングを指定するタイマーである．実際には送信先のモジュールに対し，デフォルトで 256 フレームの空のオーディオデータを送信する．受信側モジュールは受信したタイミングでオーディオ処理を実行する．送信するオーディオデータのフレーム数は演奏インターフェイスを用いることで変更可能である．

4.3.2 DAC モジュール

DAC モジュールは, 他のモジュールから受信したオーディオデータを実際にスピーカから再生するモジュールである．ネットワークを介したストリーミング再生という性質上，受信側は常に一定のデータを受信できるとは限らないため，バッファリングをおこないオーディオデータが途切れないようにしてある．そのためネットワーク部分での遅延に加え，オーディオデータを受信してから実際にスピー

カから再生されるまでに遅延が生じる．このバッファの大きさは演奏インターフェイスを用いて 64 フレームから 4096 フレームまで変更可能である．またボリュームも演奏インターフェイスを用いて変更可能である．

実装では C 言語によるリアルタイムオーディオ処理ライブラリである PortAudio[5]を使用した．

4.3.3 ADC モジュール

ADC モジュールは，マイク入力を 44.1kHz, 16bit のリニア PCM データにサンプリングし，別のモジュールに送信するモジュールである．

実装では DAC モジュールと同様 PortAudio を使用した．

4.3.4 Sine モジュール

Sine モジュールは特定の周波数の正弦波を生成し，別のモジュールに送信するモジュールである．演奏モジュールを用いて周波数の変更をおこなえる．

4.3.5 Envelope モジュール

Envelope モジュールはアタック,ディケイ,サステイン,リリースの４つのパラメータからなり，他のモジュールから受信したオーディオデータのボリュームを制御する．

4.3.6 Delay モジュール

Delay モジュールは他のモジュールから受信したオーディオデータに対してディレイ効果を施すモジュールである．処理が施されたオーディオデータは別のモジュールに送信される．

4.3.7 タッチディスプレイ型演奏インターフェイス

演奏データを送信するための演奏インターフェイスを Apple 社 iPhone (iPod Touch, iPad)用アプリケーションとして実装した．ディスプレイ上にタッチされた指の座標を送信するモジュールとなっている．

図 3 第３世代 iPod Touch，第４世代 iPod Touch，iPad 2

4.4 メタ音楽演奏システムの制作

4.4.1 タイル型メタ演奏インターフェイス

半透明のアクリル筐体の中に電子回路を収めたタイル型のインターフェイス（以下タイル）を制作した．タイルの側面にはマグネットが取り付けられており，タイル同士が磁力で連結するようになっている．タイルの左隅にはフルカラーLED が取付けられており，タイルに割当られたモジュールを表す色が点灯する．

モジュールを表すタイルの側面はモジュールの入出力部を表す．

- 上部 … 演奏データ入力
- 下部 … 演奏データ出力
- 左部 … オーディオデータ入力
- 右部 … オーディオデータ出力

またタイルの側面左部と上部には赤外線受光素子が，側面右部と下部には赤外線 LED が取り付けてあり，タイル同士が連結されると赤外線 LED に繋がった磁気スイッチが入り，タイルの個体識別番号（以下タイル ID）が赤外線で送信される．受信側は赤外線を受信すると信号を解析し，連結されたタイルのタイル ID，自身のタイル ID，連結している面の 3 つの情報をモジュールコーディネータに送信する．

赤外線送信には Microchip 製マイクロコントローラ PIC12F625 を使用した．赤外線の受信と解析には Atmel 製マイクロコントローラ AVR ATMEGA 328P-PU を使用した．データ送信は ZigBee 規格による無線通信モジュールである XBee Series2 を使用した．

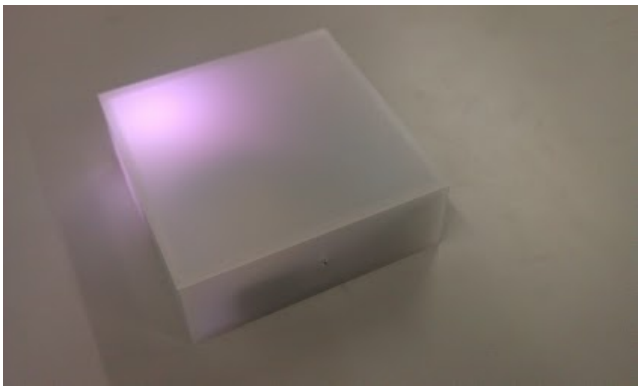

図 4 タイルの外観

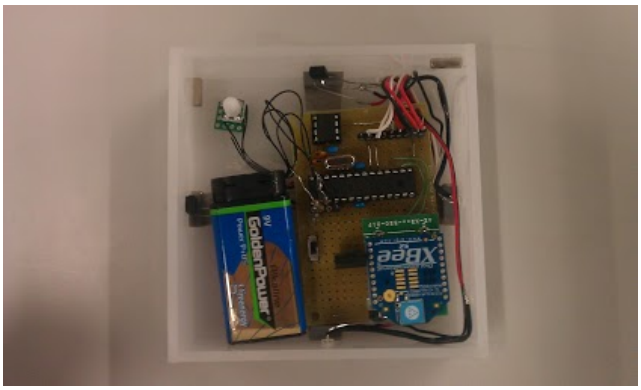

図 5 タイルの内部

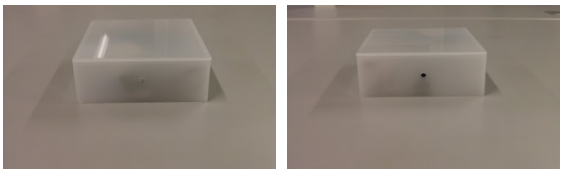

図 6 タイル側面の様子
（左: 赤外線送信部　右: 赤外線受信部）

4.4.2 タイル型メタ演奏インターフェイスシミュレータ

タイルの動作をシミュレートする Mac OSX 用アプリケーションもあわせて制作した．画面内をクリックするとタイルが新たに生成される．シミュレータではタイルの表すモジュールはあらかじめ決められており，タイルとモジュールのマッピング作業はおこなわない．タイル同士を連結させると，連結しているタイル同士のタイル ID と連結している面の情報がモジュールコーディネータに送信される．

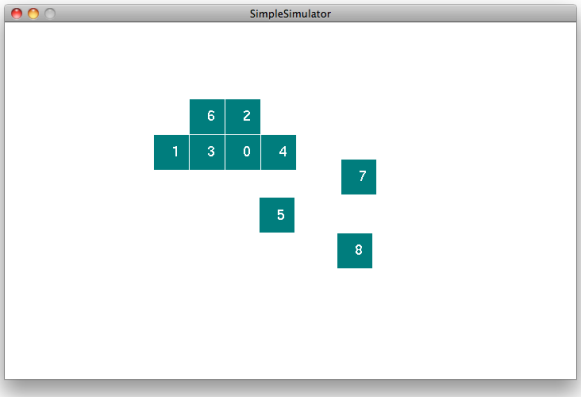

図 7 タイル型メタ音楽演奏インターフェイス
シミュレータ実行画面

4.4.3 モジュールリスト

モジュールリストのコマンドライン用プログラムを実装した．言語は C++を使用した．またタイルとの接続部として，赤外線受光解析部もあわせて制作した．赤外線の受信と解析には Arduino Duemilanove を使用した．

モジュールリストはネットワーク内に複数存在でき，モジュールマネージャから受信したモジュール情報の表示およびモジュールとタイルのマッピングをおこなう．プログラムの具体的な動作は以下の通りである．

1. はじめに赤外線受光部とタイルを接続し，タイルから送信される赤外線を受信し，タイル ID を取得する．
2. 赤外線受光部で取得したタイル ID はシリアル通信でパソコン側プログラムに送られる．モジュールリストは OSC アドレス”/ModuleList”でモジュール情報を待ちうける．
3. 一方モジュールマネージャは起動時にネットワーク内に存在する全てのモジュールリストに対してモジュール情報を送信する．具体的には OSC アドレス”/ModuleList”への OSC パケットを UDP ブロードキャスト送信する．
4. モジュール情報を受信したモジュールリストはモジュール情報をテキスト表示する．
5. 表示されているモジュール情報のうち，モジュール識別番号（以下モジュール ID）を入力すると，そのモジュールを持つ端末に対してタイル ID を添えたモジュール生成依頼を OSC で送信する．

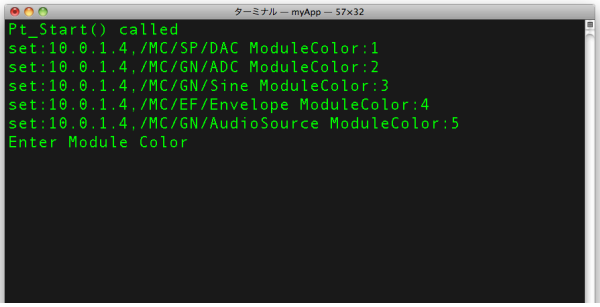

図 8 モジュールリストのプログラム実行画面

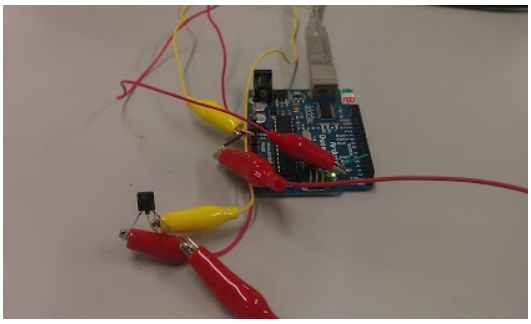

図 9 赤外線受信部

4.4.4 モジュールとタイルのマッピング

以下の手順でモジュールとタイルのマッピングをおこなう．

1. モジュールコーディネータはネットワーク内にただひとつ存在し，タイルと無線通信をおこない連携をとる．はじめにタイルのふたを開け，スイッチを入れるとタイルはモジュールコーディネータに対し自身のタイル ID を定期的に送信する．モジュールコーディネータはタイルから送られてきたタイル ID とタイルの ZigBee アドレスを取得する．
2. モジュールリストは，ユーザがモジュール ID を入力するとそのモジュールを持つモジュールマネージャに対し，接続されているタイルのタイル ID を添えて新規モジュールの生成依頼を OSC で送信する．
3. モジュールマネージャは，モジュールリストからタイル ID の添えられたモジュール生成依頼を受信した

のち，タイル ID とモジュール名からなる OSC アドレスを持つモジュールを新たに生成する．タイル ID が 1 でモジュール名が”DAC”であれば，新たに生成されるモジュールの OSC アドレスは”/Tile1/DAC”となる．さらに”/Data”と”/Stream”という下位アドレスが付加され，”/Tile1/DAC/Data”では演奏データを，”/Tile1/DAC/Stream”ではオーディオデータを受信することになる．次にモジュールマネージャはモジュールコーディネータに対し，モジュールの OSC アドレス（先の例では”/Tile1/DAC”），タイル ID，モジュール ID の登録依頼として OSC アドレス”/Coordinator”への OSC パケットを UDP ブロードキャスト送信する．

4. モジュールコーディネータはモジュールマネージャからの登録依頼を受信したのち，これに添えられたタイル ID と一致するタイルにモジュール ID を ZigBee で送信する．
5. モジュール ID を受信したタイルはモジュール ID に対応した色を点灯し，これによりタイルとモジュールのマッピングが完了する．

4.5 システムの運用

タイル型メタ演奏インターフェイスシミュレータを用いて実際にシステムを運用した．使用したモジュールは以下の通りである．

- DAC モジュール
- ADC モジュール
- Sine モジュール
- Envelope モジュール
- タッチディスプレイ型演奏インターフェイス（iPod Touch 用アプリケーション）

２台のコンピュータ(iMac)で各モジュールを含むプログラムを実行させ，別のコンピュータ(MacBook Air)でメタ音楽演奏システムのプログラムを実行させた．各モジュールはいずれも仕様通り動作した．タイル同士の連結によって実際にモジュール同士が連結され，正しく演奏データやオーディオデータが送受信されていることが確認できた．

同じ端末内に含まれているモジュール同士の通信は問題なかったが，別々の端末に含まれるモジュール同士のオーディオデータ通信の際に，無線 LAN の状況によってデータが途切れ，ノイズが発生することがあった．

5. おわりに

本稿では，従来の電子楽器に対する認識について考察し，この概念をさらに広げるために，電子楽器を構成する各機能のネットワークである音楽演奏システムと，音楽演奏システムの動的構築をおこなうメタ音楽演奏システムという捉え方を提案したが，実際のところメタ音楽演奏を従来の音楽演奏と区別して捉えるべきか，音楽演奏のひとつのスタイルとして捉えるべきかは難しい議論である．今後このような音楽演奏システムの動的構築をおこなうしくみが発展していくうちに演奏という行為に関するさらなる議論がおこなわれることを期待している．

音楽演奏システム内の各モジュールに関しては次々と新しいモジュールを追加してゆく予定である．演奏インターフェイスについては特に充実させてゆかねばならないと考えている．メタ音楽演奏システムについてはより有効な仕組みについて検討中である．特にモジュールのネットワークを再現するシンボルとなるメタ演奏インターフェイスは検討する余地が多く残っている．

最後に，音楽演奏システムの分散ネットワーク化とメタ音楽演奏システムによって音楽の演奏がより豊かになることに少しでも貢献できれば幸いである．

謝辞

本研究を進めるにあたって電子回路の制作を手伝っていただいた三枝英一氏に感謝する．

参考文献

[1] Niklas K, Marc F, Georg G, Florian E,: An Approach to Collaborative Music Composition, Proceedings of the International Conference on New Interfaces for Musical Expression, pp. 32–35, 2011
[2] 後藤 真孝, 根山 亮, 村岡 洋一, : RMCP : 遠隔音楽制御用プロトコルを中心とした音楽情報処理, 情報処理学会論文誌 40(3), 1335-1345, 1999-03-15
[3] Wright, M., Freed, A.,: Open Sound Control: A New Protocol for Communicating with Sound Synthesizers, International Computer Music Conference, Thessaloniki, Greece, 1997.
[4] liblo: Lightweight OSC implementation http://liblo.sourceforge.net/
[5] Ross B., Phil B.,: PortAudio - an Open Source Cross Platform Audio API, International Computer Music Conference Proceedings, 2001