DEIM Forum 2021 D24-3

関係データベースおよびMIDIデータを用いた演奏支援システム

渋川　大樹†　　横山　昌平††

† 東京都立大学 システムデザイン学部　〒 191–0065 東京都日野市旭が丘 6-6
†† 東京都立大学院 システムデザイン研究科　〒 191–0065 東京都日野市旭が丘 6-6
E-mail: †shibukawa-taiki@ed.tmu.ac.jp, ††shohei@tmu.ac.jp

あらまし　近年, COVID-19 の影響により, リモートで行う活動が盛んになってきた. その結果, 音楽分野でもネットワークを介した活動が行われるようになった. また, 音楽分野では楽曲などの製作時, DTM の使用が主流となっており, MIDI データなどコンピュータ上やネットワーク上で扱いやすいデータ形式のものが使用されるようになってきた. データとして扱いやすい状況にあるにも関わらず, 音楽をデータベースで扱っている研究は少なく, その中で演奏そのものを扱うような演奏者のための研究はさらに少ない. そこで本研究は関係データベースと MIDI データを用いた演奏支援の手法を提案する. 本研究では, 実証実験のために Web アプリケーションとしてルーパーを作成した. その結果, 関係データベースを用いることで, 遠隔ならびに 1 人で合奏することが可能になり, SQL 操作で音楽を制御することが可能になった.

キーワード　関係データベース, MIDI, SQL, 演奏支援

1　はじめに

近年, COVID-19 の影響により様々な経済活動が停滞することになった. その中でも特に音楽活動の分野においては多くの制限がかけられ, ライブ活動などを行うことが困難となっている. このような状況下で活動をするために, オンラインライブや SYNCROOM[1]のように, 遠隔地にいる人同士とリアルタイムで合奏ができるアプリケーションなどが開発される等, インターネットを介した表現方法が模索されている状況にある.

また, 最近の音楽制作は DTM（Desktop Music）であり, 昔のように楽譜に音符を書き込んだり, 磁気テープに録音したりするのではなく, パソコンを使用した方式が主流となっている. この流行は, 多くの種類の DAW（Digital Audio Workstation）ソフトが販売されていることからも分かる. DAW ソフトは MIDI（Musical Instrument Digital Interface）規格に則って制作されたデータを扱うことが多い. DAW ソフトを用いることで MIDI データを記録, 編集, 再生をすることが可能である.

MIDI データは音声情報のような波形データではなく, どのような音が発せられたかという音源や楽器へのメッセージであり, 演奏情報という. この MIDI データはデータサイズが小さく, かつ音楽情報の細部を容易に変更することが可能であるという特徴のため, 機器間で共有や転送がしやすい.

我々は, 音楽データを用いることでインターネットを介して, 複数人で遠隔演奏するシステムを構築する研究を進めている. その中で, 音楽データの 1 つである MIDI データに着目し, このデータを使用した複数人で遠隔演奏するシステムを構築する研究を現在行っている. このシステムを構築するためには, 複数人でデータにアクセスすることが可能であり, かつ同期的で一貫性を保て, ネットワーク越しで扱うことが可能であるという条件が必須となる. 複数人でデータを共有することに最適なシステムとして関係データベースがある. 関係データベースは e コマースや金融など, データの整合性が特に必要とされる分野においてもシステムの核となっており, 高度なトランザクション処理能力を有している.

本研究ではインターネットを介して, 遠隔のサーバで演奏データを管理することを目的としている. そのため, 社会における重要なデータを扱う十分な実績のある関係データベースを用いるということがこの研究のアイデアとなる. (図 1)

我々は、関係データベースと MIDI 機器を用いた遠隔演奏を目的とした実証実験のために, ルーパーという既存のエフェクターに着目した. ルーパーとは, 記録した演奏フレーズを重ね合わせ, 任意の小節で半永久的にループ再生するものである. 簡単に表現すると, 一人で多重録音をして合奏を行うことが可能なものである. フレーズをリアルタイムに保存し、同時に再生するルーパーを関係データベース上で実現することにより、ネットワーク越しの合奏環境を実現することが可能である.

本研究では, 演奏した MIDI データを関係データベースにただちに保存し, 複数人の演奏者で同期・共有するシステムを構築することを目指す. 目指すシステムの特徴としては, 共有・同期できることや, 保存と取り出しを同時に行えること, SQL 操作のみで音楽構成を制御できることにある. つまり, 関係データベースを使用することによって, 関係データベースの利点をそのまま利用できることになる. そのため, 特別な手法や処理を加えることなく, ネットワーク越しで音楽を複数人で扱うことが可能な点に提案手法の特徴がある. 今回は 1 人の演奏者が 1 人で合奏するシステムを作成し, 本論文ではこのシステムに関する説明をする.

関係データベースには MySQL[2]を使用し, MIDI データを入

1：https://syncroom.yamaha.com/

2：https://www.mysql.com/jp/

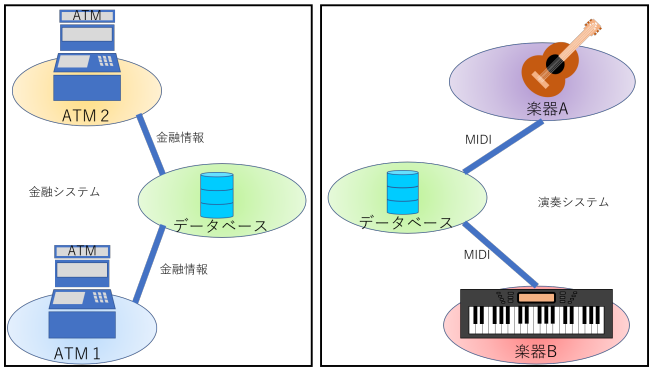

図 1　金融システムと複数人演奏の概略図

力する装置として Roland[3]の MIDI キーボード・コントローラーである A-300PRO を用い, 通信方法としては WebSocket を使用し, Web アプリケーションとしてルーパーを作成した. このシステムの利点としては, 目指すシステムの利点と同じように, SQL 操作で音楽を制御できることや, 関係データベースの容量がそのまま演奏を記録できる時間になることである.

本論文の構成は以下の通りである. 2 章では関連研究について述べる. 3 章では提案手法について述べ, 作成したシステムについて説明する. 4 章では遅延に関する評価実験を述べる. 5 章では作成したアプリケーションに関して説明する. 6 章では本研究のまとめと今後の課題, 展望を述べる.

2　関 連 研 究

本章では関連研究について述べる. まず, 遠隔地から音楽を扱う研究を紹介する. 後藤ら [1] は Open RemoteGIG という遅延を考慮した不特定多数による遠隔セッションシステムを開発した. このシステムは, 同一のコード進行の繰り返しを一定のテンポで演奏することを前提とし, スター状のネットワーク接続によって演奏情報を 1 つの場所に集約し, コード進行の 1 周期の整数倍遅延して送り返すことでセッションを可能としている. これにより遅延をなくすのではなく, 遅延を利用するという新しい方向性を示した.

野原ら [2] は WebRTC を用いた音楽表現の指導を配慮した遠隔指導支援システムを制作した. WebRTC により, ブラウザ間の直接通信ができ, リアルタイムに送受信を行えるようになった. さらにカメラやマイクが利用可能になった. この結果, DAW の遠隔指導において, 指導者がリアルタイム化された音響を聴いて, 対面と同様に問題把握をしながら指導できるようになった. しかし, リアルタイムで合奏を行う分には不十分という結果も得られた.

このように遠隔地から音楽を扱った研究は遅延に対する問題が付随する. 遅延を活用する手段や, 遅延を少なくする手段を用いることで, 遅延に対応することが必要となる.

次に音楽とデータベースを扱った研究を紹介する. James B. Maxwell ら [3] は音楽情報検索に利用可能な音楽情報データベース・検索システムを提案した. これにより, シンプルな 6 部構成の構文でクエリを受け付け, データベースに格納されている音楽形式のエンコードされた知識を利用して, さまざまなタイプの音楽情報を返すことが可能である.

橋田ら [4] はピアニストの演奏解釈を記述した演奏表情データベースの構築を行っている. このデータは, 音楽構造記述に焦点を当てており, 音符の発音時間のずれや継続時間など, 音楽表現に関する詳細データを提供する. これにより, 音楽演奏に関する研究にデータベースを活用できると考えている.

音楽とデータベースを扱った研究は, 音楽専用のデータベースを扱ったものが多く, 演奏者のためのシステムではないことが多い.

次に演奏支援に関する研究を紹介する. 佐竹ら [5] は自分の過去の演奏履歴を活用して合奏を行うシステムを提案した. Emacs 拡張のひとつである入力の繰り返しを自動化する Dynamic Macro を用いて, 過去の演奏を繰り返すことで, その演奏に合わせてさらに演奏を重ねられるものとなっている. そのため, 演奏中や演奏終了後にいつでも繰り返し再生を行い, 合奏を始めることが可能である. これにより, ルーパーにはない機能が実現された.

金杉ら [6] は, 演奏者の身体や楽曲に合わせて, 鍵の大きさをカスタマイズすることが可能なソフトウェア鍵盤楽器による演奏支援を提案した. 手動と自動の 2 種類のシステムを作成し, 手動では演奏者の意図を直接伝えられたため, 演奏がしやすくなった. 一方, 自動では変化があまりなく, 演奏のしやすさも変化がなかった. 結果として, 変形により演奏がしやすくなることは確認できた.

このように演奏支援の研究では, さまざまなアプローチで行われている. 関連研究と本研究の違いとしては, 関係データベースを用いたことにあり, さらにそれを演奏者のために使用することにある. 本研究では, 特別な手法や処理を加えることなく遅延を考慮し, 汎用性のあるデータベースを用いた演奏者のためのシステムを構築した.

3　提 案 手 法

本研究では関係データベースと MIDI データを用いた演奏支援システムの手法として, Web MIDI API [4]と WebSocket 通信を用いた Web アプリケーションとしてのルーパーを提案する. ここではシステムの概要, MIDI の規格, 関係データベースの設定に関してをそれぞれ説明する.

3.1　システムの概要

ここではシステムの構成と作成したルーパーの仕組みを説明する. システムを作成するうえで以下のものを用いた.

(1) Roland の A-300PRO（MIDI KEYBOARD CONTROLLER）

(2) MySQL 8.0（関係データベース）

3：https://www.roland.com/jp/

4：https://www.w3.org/TR/webmidi/

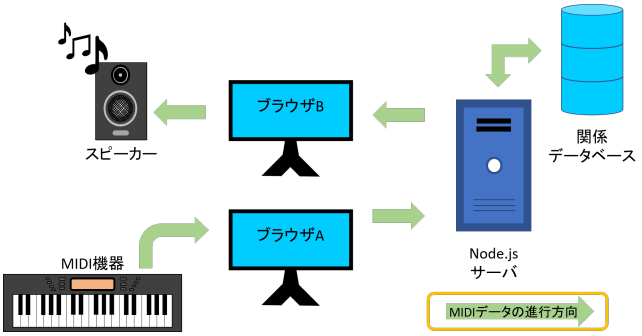

図 2　一人での遠隔演奏時のシステムの構成図

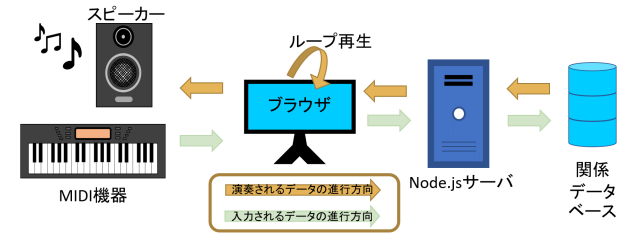

図 3　ルーパーの仕組み図

(3) Cakewalk by Bandlab[5]（音源）

これらと自作した Web アプリケーションで実験を行う．

3.1.1　システムの構成

まず，システムの構成について説明する．本アプリケーションは 3 つの構成に分けることが可能である．

(1) クライアント

(2) Node.js サーバ

(3) 関係データベース

クライアントは各種設定，制御，MIDI データの入出力を行う Web ページである．Node.js サーバは Web サーバであり，クライアントからのデータを受信，送信する．また，関係データベースへデータを送信，受信する役割も持つ．クライアントとの通信では WebSocket 通信を用いている．関係データベースでは，MIDI 機器から出力された MIDI データと時刻，何ループ目の演奏かという 3 つの情報を保存できるようになっている．

図 2 に一人での遠隔演奏時のシステムの構成図を示す．MIDI 楽器から出力された MIDI データは，ブラウザ A 側で出力された時刻とループ数が付加され，Node.js サーバに送られる．送られたデータはブラウザ A 側の指示により，Node.js サーバは関係データベースにデータを保存するのと同時にブラウザ B 側に送信するのか，保存せずにそのままブラウザ B 側へ送信するかを判断する．その後ブラウザ B 側に送信されてきたデータによって音を鳴らしたり，演奏されたりする仕組みになっている．

3.1.2　ルーパーの仕組み

ここでは作成したルーパーの仕組みについて説明する．本システムで作成したルーパーの仕組みを図 3 に示す．MIDI 機器から出力されたデータはサーバを中継して，関係データベースに保存される．関係データベースから送信されたデータはクライアント側で一度保存される．この保存したデータをループ再生することで演奏する．この演奏に合わせてさらに演奏することで，音を重ねて演奏データを記録することが可能である．この録音が終わるたびに関係データベースから新しくデータが送信され，クライアント側のデータが上書きされる．

5：https://www.bandlab.com/products/cakewalk

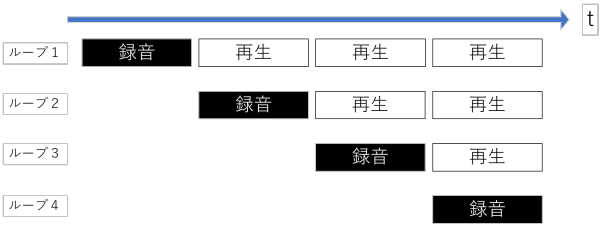

図 4　既存のルーパー

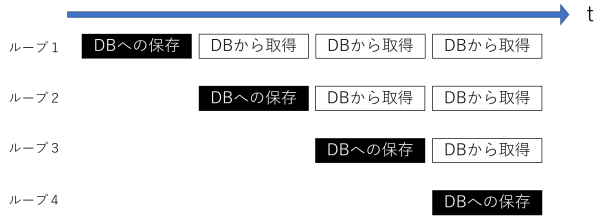

図 5　作成したルーパー

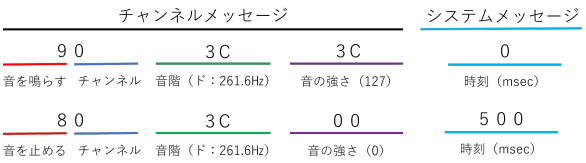

図 6　MIDI データの例

図 4 に既存のルーパーのループ再生の仕方，図 5 に作成したルーパーのループ再生の仕方を示す．このシステムの仕組みは，既存のルーパーで置き換えると，録音の部分は関係データベースにデータを保存すること，再生の部分は関係データベースからデータを取り出すことと表すことが可能である．

3.2　MIDI の規格

ここでは MIDI の規格について説明する．MIDI は演奏の情報を電子データとして転送・共有するために規格化された世界共通のものである．MIDI データはメッセージであり，シンセサイザーなどの音源にどのような演奏をするかを指示するものである．そのため，音を生成するものはシンセサイザーなどの音源であって，MIDI データではない．

図 6 に実際の MIDI データであるドの音が鳴らされた際および音を停止させた際の例を示す．MIDI データは大きく分けると 2 つあり，チャンネルメッセージとシステムメッセージとなる．チャンネルメッセージは 16 のチャンネルの 1 つを主に指定して送信するものであり，演奏情報そのものである．このメッセージにはチャンネルや音階，音を鳴らす，止めるの情報，音の強さの情報などが含まれており，16 進数で表すことが多い．

システムメッセージはチャンネルの指定に関わらず，システム全体に対するメッセージである．メーカー固有のデータのやり

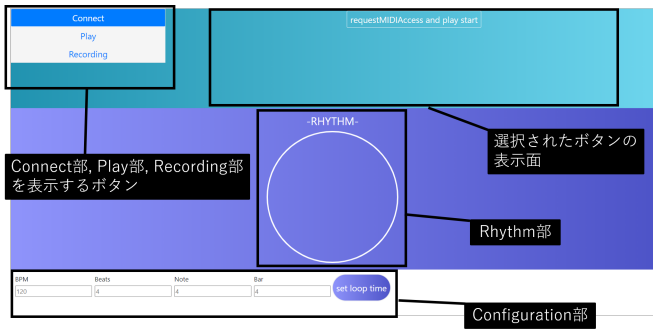

図 7　初期表示画面

とりをするメッセージや, 機器のコントロールをするメッセージがあり, 時刻などが当てはまる. これらのメッセージデータを Web ブラウザから使用するために Web MIDI API を用いた. また, そのデータを関係データベースで扱った.

3.3　関係データベースの設定

ここでは関係データベースの設定について説明する. 今回使用したデータベースのテーブル構造は表 1 のように設定した. カラムは 5 つ作成した. チャンネルと鍵盤が押されたか離されたかの情報を noteON に, 音階を notenumber に, 打鍵の強さを velocity に, MIDI データが作成された際の時刻を time に, ルーパーの何ループ目か (何回目の録音か) を loopnumber に設定している. Web MIDI API の timestamp の型が double 型となるため, time のみ double 型にし, そのほかは整数で表現できるため int 型としている. また, time の順番にデータが並ぶように Multiple key を設定した.

関係データベースに MIDI データを保存する際は, 1 個の MIDI データが作成されるたびに保存し, 取り出す際は, 全てを一度に取り出す方法と, いくつかに分割して取り出す方法の 2 種類を試した. これらの制御はサーバ側から SQL 操作にて行っている.

表 1　テーブル構造

Field	Type	Null	Key	Default	Extra
noteON	int	YES		NULL	
notenumber	int	YES		NULL	
velocity	int	YES		NULL	
time	double	YES	MUL	NULL	
loopnumber	int	YES		NULL	

4　アプリケーション

ここでは作成したアプリケーションの表示と機能に関して説明する. 図 7 が初期表示画面となる. 本アプリケーションの表示としては 5 つの部に分けることができ, Connect 部, Play 部, Recording 部, Rythm 部, Configuration 部となる. そのうち, Connect 部, Play 部, Recording 部の 3 つは, ボタンを押すことで表示を切り替えることが可能になっている. これらの 5 つの部に関してそれぞれ述べる.

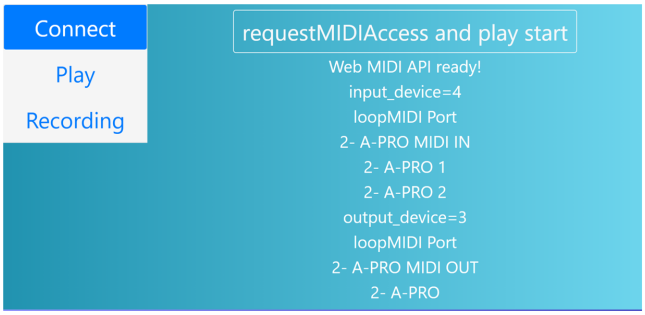

図 8　Connect 部

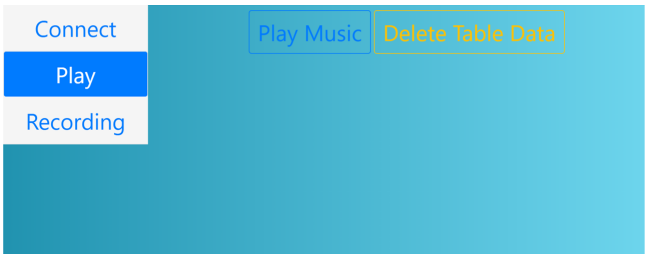

図 9　Play 部

4.1　Connect 部

図 8 が Connect 部の表示画面となる. ここでは MIDI 機器の接続と演奏の開始を制御する部分となる. “requestMIDIAccess and play start” ボタンをクリックすることにより, MIDI 機器との接続を行う. そのとき, 接続ポートを表示する. 接続ポートの表示ではインプットデバイスとアウトプットデバイスの 2 つを表示し, それぞれの検出されたポート数も表示する. この表示が出ることで演奏が可能となる. しかしながら, Web ページそのものには音源がないため, 音源を別に繋がなくては音が鳴らない.

Web ページに音源を付随させなかった理由として, 音源を持たせてしまった場合, 1 つの音源に固定されてしまうということがある. また, 複数人で合奏する場合, 将来的に MIDI データをある 1 つの機器に集め, そこから 1 つの音源に繋ぐことや, 機器それぞれが別々の音源を使用するといった設定を検討しているためでもある.

4.2　Play 部

図 9 が Play 部の表示画面となる. ここでは記録した演奏に関する制御をする部分となる. “Play Music” ボタンをクリックすることで記録した演奏を聴くことが可能である. このとき録音したものを聴くことが 1 回目なら, 関係データベースから取り出したデータで演奏する. 2 回目以降なら, 1 回目のデータをクライアント側で保存し, そのデータで演奏する. その結果, 同じ通信を何度もしなくて済むようになる. “Play Music” ボタンがクリックされた際の表示としては 1 回目なら ‘play music 1’, 2 回目以降なら ‘play music 2’ とした.

“Delete Table Data” ボタンは記録されている関係データベースにあるデータを削除する. このデータ削除は一括で行われ, クライアント側で保存されているデータも削除される.

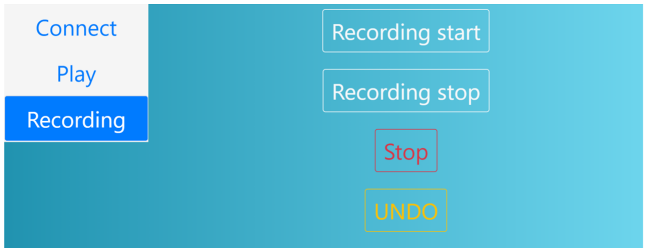

図 10　Recording 部

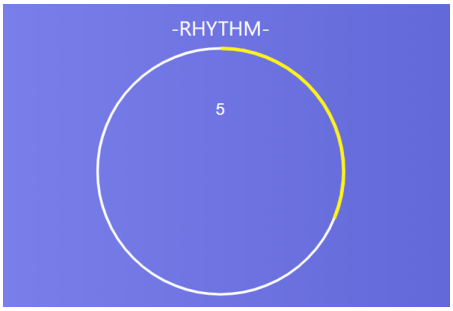

図 11　Rythm 部

4.3　Recording 部

図 10 が Recording 部の表示画面となる．ここでは演奏を記録でき，さらに記録した演奏を聴きながらその演奏に合わせて，新たな演奏を重ねることで合奏として演奏を記録することが可能である．“Recording start” のボタンをクリックすることで記録を開始する．“Recording stop” のボタンで記録を停止する．この瞬間から記録した演奏を半永久的にループ再生する．再び開始ボタンをクリックすることで自分の好きなタイミングで演奏を重ねて記録することが可能である．ループ再生する演奏を停止する際は “Stop” ボタンをクリックすることで停止する．記録に失敗した際は “UNDO” ボタンを押すことで 1 番最後に記録した演奏を削除することが可能である．

これらの機能は既存のルーパーにも存在する．1 回目の演奏データの記録開始を ‘スタート’，重ね録りとなる 2 回目以降の演奏データ記録開始を ‘オーバーダブ’，最後に重ねた演奏データを消去することを ‘アンドゥ’ という．これら機能を関係データベースを使用して，SQL 操作で行えるようにした．

4.4　Rythm 部

図 11 が Rythm 部の表示画面となる．ここでは視覚的に演奏のリズムや 1 ループが残りどのくらいあるか分かるインジケータを作成した．Recording 部の “Recording start” ボタンを押した際に，このインジケータが動き出す．また，“Recording stop” ボタンが押された際は，再び初めからインジケータが動き出す．このインジケータの周期やリズムは Configuration 部で変更できる．

インジケータの動きとしては 1 拍ごとに数字がカウントされていき，1 周期分がカウントされ終わると初期値 1 に戻る．また円形の部分は，1 拍ごとに黄色く色づいていき，1 周期経つと全て黄色くなる．この画面表示は表示が切り替えられてもそのまま表示される．

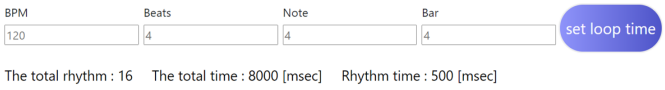

図 12　Configuration 部

4.5　Configuration 部

図 12 が Configuration 部の表示画面となる．ここではインジケータの 1 周期の時間やリズムを設定する．左から，BPM，1 小節の拍数，基準となる音符，小節数となっている．Connect 部の “requestMIDIAccess and play start” ボタンをクリックすることで 1 周期の拍数，1 周期の時間，1 拍の時間が表示される．初期設定は 4 分の 4 拍子基準の BPM120 で，4 小節分の長さとなっている．分かりやすく表現すると，0.5 秒に 1 回拍を打ち，1 周期 8 秒となる．この設定は自分の好きなタイミングで自由に操作できる．

自由に操作できるようにした理由としては，記録時間の柔軟性を求めたためである．既存のルーパーでは最初の記録時間が 1 ループの時間となってしまう．そこで本システムでは，1 ループの時間をいつでも変更できるようにすることで，最初の記録時間に制限されることなく自由に演奏できる．

5　評 価 実 験

本研究では，ネットワークを介して音楽を扱うため，音が鳴るまでに時間がかかり，遅延が発生する．特に合奏において遅延が大きいと，演奏のタイミングがずれてしまい，演奏ができなくなってしまう．

西堀ら [7] によると，演奏における遅延は 30 ms 以上で認識できるようになり，50 ms 以下でなら正しく演奏できるとある．このため，作成したシステムが十分な応答速度を有しているか検証するために，遅延時間の計測を行った．作成したシステムは 1 人の演奏者が 1 人で合奏するシステムのため，この計測はローカル環境で行ったものになる．また，ブラウザの環境としては Google Chrome バージョン: 87.0.4280.141 を使用した．以下に計測の方法を示す．

計測 1：鍵盤を押してから音が鳴るまでの計測
計測 2：クライアント側で保存したデータを取り出して演奏されるまでの計測
計測 3：関係データベースからデータを取り出して演奏されるまでの計測

3 つの計測パターンの評価実験についてそれぞれ述べる．

5.1　計測 1 の検証

図 13 に計測 1 の計測方法を示す．鍵盤を押してから音が鳴るまでの計測時間は，ブラウザ A が MIDI データを取得してからブラウザ B がデータを受信するまでの時間を往路時間とし，ブラウザ B がデータを受信してから受信したことを知らせるデータをブラウザ A に伝達し，ブラウザ A がそれを取得した時間を復路時間として，往路時間と復路時間の合計とする．実際には鍵盤を押してから音が鳴るまでの時間ではないが，計測す

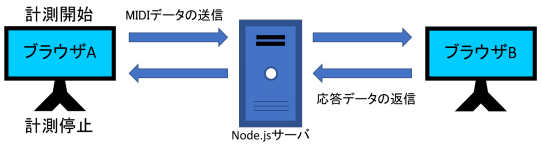

図 13　計測 1 の計測方法

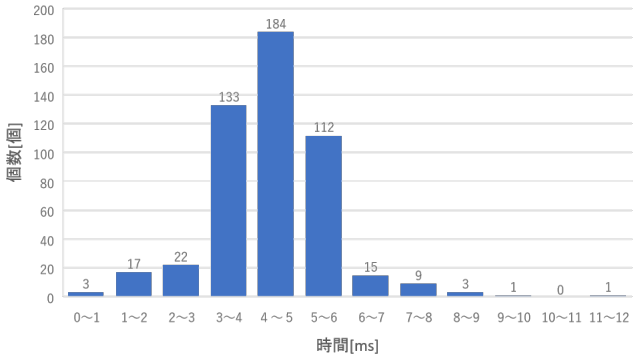

図 14　計測 1 の計測結果

る時間はこの時間よりも長い．したがって，計測する時間が十分に短ければ，鍵盤を押してから音が鳴るまでの時間も十分に短くなり，十分な応答速度を有しているか検証することが可能である．

500 個の MIDI データを計測した結果を図 14 に示す．図 14 からも確認できるように 4 ms から 5 ms の範囲が 184 個と一番多くなった．この範囲に続いて，3 ms から 4 ms の範囲，5 ms から 6 ms の範囲となり，3 ms から 6 ms の範囲が約 86 %を占めた．また，計測値の平均は約 4.38 ms で，最大値は約 11.51 ms となった．遅延が認識できる時間は 30 ms 以下であるため，十分な応答速度があると考えられる．

5.2　計測 2 の検証

クライアント側で保存したデータを取り出して演奏されるまでの計測時間は，音源にデータを送信するまでの時間であり，データを取得してから音源に送信されるまでにかかる内部処理の時間となる．MIDI データの個数を 10 個，100 個，1,000 個，10,000 個のときの各場合について 5 回ずつ計測をした結果を表 2 に示し，得られた各データの平均値を表 3 に示す．

表 2　計測 2 の結果

総データ数 [個]	1 回目 [ms]	2 回目 [ms]	3 回目 [ms]	4 回目 [ms]	5 回目 [ms]
10	0.509999925	0.375000061	0.359999947	0.400000019	0.355000142
100	0.385000138	0.390000176	0.529999845	0.379999867	0.405000057
1,000	0.364999985	0.409999862	0.379999867	0.539999921	0.404999824
10,000	0.370000023	0.405000057	0.435000053	0.365000218	0.4149999

表 3　表 2 で得た各データの平均値

10 個 [ms]	100 個 [ms]	1,000 個 [ms]	10,000 個 [ms]
0.400000019	0.418000016	0.419999892	0.39800005

各データ 1 ms 以上の遅延は起きておらず，全データの平均値としては約 0.412 ms である．その結果，内部処理にかかる遅延は演奏にとって影響が少ないと考えられる．また，演奏を重ねることを考慮すると，クライアント側でデータを保存したものを取り出して演奏することが，合奏に合っていると考える．

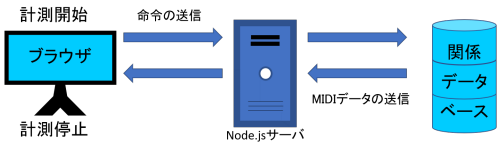

図 15　計測 3 の計測方法

5.3　計測 3 の検証

図 15 では計測 3 の計測方法を示す．関係データベースからデータを取り出して演奏されるまでの計測では，“Play Music” ボタンをクリックしてから，MIDI データが関係データベースからブラウザに送信されるまでの時間を計測する．

関係データベースから MIDI データの取り出し方としては，一度に全てのデータを取り出す方法と，いくつかに分割してデータを取り出す方法の 2 パターンで行った．

5.3.1　一度に全てのデータを取り出す方法

関係データベースから MIDI データの個数を 10 個，100 個，1,000 個，10,000 個取り出した際の各場合について 5 回ずつ計測をした結果を表 4 に示し，得られた各データの平均値を表 5 に示す．

表 4　計測 3 において一度に全てのデータを取り出した場合の結果

総データ数 [個]	1 回目 [ms]	2 回目 [ms]	3 回目 [ms]	4 回目 [ms]	5 回目 [ms]
10	5.145000061	24.67499999	20.02500021	27.81999996	28.46499998
100	11.13	26.01500018	30.21	22.93999982	29.69500003
1,000	32.3950001	42.12500015	36.4049999	35.85999995	35.66500009
10,000	102.6950001	119.6399999	122.7450001	127.6400001	109.4249999

表 5　表 4 で得た各データの平均値

10 個 [ms]	100 個 [ms]	1,000 個 [ms]	10,000 個 [ms]
21.22600004	23.99800001	36.49000004	116.429

データ数が増加するにつれて計測時間が大きくなり，遅延が大きくなることが分かる．データ数が少ない 10 個や 100 個の場合は，遅延を意識することなくその演奏に合わせて合奏することが可能であるが，10,000 個にもなると遅延が 100 ms を超えるようになり，合奏が困難になると考えられる．また，どのデータも初めの遅延が小さくなっているが，その原因は現在のところは不明である．

5.3.2　分割してデータを取り出す方法

関係データベースから 10,000 個の MIDI データを 1 個ずつ，10 個ずつ，100 個ずつ，1,000 個ずつに分割して取り出した際の各場合について 5 回ずつ計測をし，各場合初めに取り出された際の計測結果を表 6 に示す．また，得られた各データの平均値を表 7 に示す．

分割データ数が 1 個の場合は遅延が非常に大きくなり，重ね

表 6　計測 3 において分割してデータを取り出した場合の結果

分割データ数 [個]	1 回目 [ms]	2 回目 [ms]	3 回目 [ms]	4 回目 [ms]	5 回目 [ms]
1	264.4150001	296.325	323.1950002	319.385	301.92
10	32.60000004	72.08999991	84.35499994	73.52000009	51.82499997
100	8.900000248	38.26000029	34.73500023	38.63999993	36.14499979
1,000	22.64999971	45.88499991	34.78500014	35.66000005	44.79000019

表 7　表 6 で得た各データの平均値

1 個 [ms]	10 個 [ms]	100 個 [ms]	1,000 個 [ms]
301.0480001	62.87799999	31.3360001	36.754

られる演奏には適してないと考える．しかしながら，分割データ数が 100 個や 1,000 個の場合は一度に全てのデータを取り出す際よりも遅延が小さくなり，演奏として許容することが可能である遅延時間 50 ms よりも小さくなった．この結果から，データ数が 10,000 個など多い際は，関係データベースからある程度まとまりのある分割をすることで遅延を小さくすることが可能であると考える．また，分割してデータを取り出す際も一度に全てのデータを取り出す際と同様に，どの分割数でも初めの遅延が小さくなり，その原因は現在のところは不明である．

5.4　考　　察

ここでは 3 つの計測パターンの評価実験についての考察を行う．計測 1 の検証からも分かるように，演奏だけを行う分には遅延の影響はほぼないと考えられる．したがって，重ねる演奏としてもほぼ影響がないということが分かる．

計測 2 の検証の結果の通り，クライアント側で保存したデータを用いる際の遅延はほぼない．したがって，重ねられる演奏としては最も適していると考えられる．しかしながら，関係データベースからデータを取り出し，クライアント側で保存するという段階を踏む必要があり，遅延が生じることになる．そのため，演奏をループ再生させた際の 2 ループ目以降に用いる場合にしか適用できないという欠点がある．

計測 3 の検証からも分かるように，関係データベースからデータを取り出す際，データ数が多くなるごとに遅延も大きくなる．その結果，データ数が多い演奏を用いる際は，重ねられる演奏には適していないといえる．しかしながら，データを分割して取り出すことでデータ数が多い演奏だとしても，一度に全てのデータを取り出す場合よりも遅延を小さくすることが可能であると分かった．したがって，データ数が多い際は分割してデータを取り出した方が良いと考える．

本検証実験では個数で分割してデータを取り出した方が，実際に分割する場合は時間や小節で分割してデータを取り出す必要があると考える．合奏などになると短時間の中に多くのデータを持つことがあり，区切られたデータ数よりも多くなった際に音がずれるといった不具合が生じる可能性がある．演奏支援システムとして活用するためにも新たな検証が必要と考える．

6　おわりに

本研究では，関係データベースと MIDI データを用いた演奏支援システムを実装するにあたり，Web アプリケーションとしてのルーパーを作成した．このシステムは，一人で遠隔演奏ならびに合奏できるものとなった．関係データベースに演奏データを集めることで，1 つの楽曲を複数回に分けて作成することが可能になり，SQL 操作で音楽を制御できるようになった．しかしながら，再生中の演奏をリアルタイムに音程を変更することなどはできておらず，リアルタイム操作性に乏しい状態である．また，データ量が多くなるにつれて，遅延が大きくなるといった問題も見つかった．

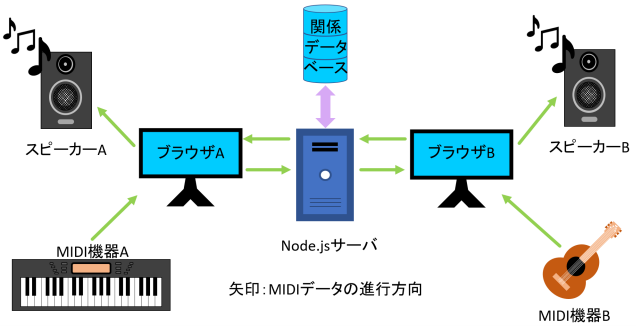

図 16　複数人対応のシステム構成図

今後の課題としては，リアルタイム処理の充実，遅延の改善が挙げられる．リアルタイム処理としては，関係データベースから MIDI データを取り出した後，時系列的に処理し，関係データベース内のデータの変更と分けることで，関係データベース内でデータを変更してから取り出すより遅延をなくすことが可能であると考える．また，遅延の改善では効率の良いデータの取り出し方を検討する．さらに現在は複数人で扱える環境ではないので，関係データベースに複数人が繋ぐことが可能な環境の整備を目指し，図 16 のようなシステムの構築をする．

今後の展望としては，デジタル楽器による MIDI の演奏データだけでなく，アナログ楽器によって演奏された波形データを本システムで扱うことを可能にしたいと考えている．

文　　献

[1] 後藤真考，根山亮，”Open RemoteGIG：遅延を考慮した不特定多数による遠隔セッションシステム，” 情報処理学会論文誌，Vol.43, No.2, pp.299-309, 2002.

[2] 野原祐一，辻靖彦，”WebRTC を用いた DAW 用遠隔指導支援システムの開発，” 情報処理研究学会報告 (MUS), Vol.2017, No.15, pp.1-6, 2017.

[3] James B. Maxwell, Arne Eigenfeldt, ”A MUSIC DATABASE AND QUERY SYSTEM FOR RECOMBINANT COMPOSITION，” In ISMIR, pp.75-80, 2008.

[4] 橋田光代，兼口敦音，中村栄太，古屋晋一，小川容子，片寄晴弘，”ピアニストの演奏解釈を記述した演奏表情データベースの構築，” 研究報告音楽情報科学 (MUS), Vol.2017, No.23, pp.1-6, 2017.

[5] 佐竹紘明，増井俊之，”Re:Piano: 演奏履歴を活用する楽器演奏支援システム，” エンタテインメントコンピューティングシンポジウム 2017 論文集，Vol.2017, pp.192-195, 2017.

[6] 金杉季実果，宮下芳明，” 演奏者と楽曲に合わせて変形可能なソフトウェア鍵盤楽器による演奏支援の検討，” エンタテインメントコンピューティングシンポジウム 2019 論文集，Vol.2019, pp.113-118, 2019.

[7] 西堀佑，多田幸生，曽根卓朗，” 遅延のある演奏系での遅延の認知に関する実験とその考察，” 情報処理学会研究報告音楽情報科学 (MUS), Vol.2003, No.127(2003-MUS-053), pp.37-42, 2003.