公立はこだて未来大学 2024 年度 システム情報科学実習
グループ報告書

Future University Hakodate 2024 Systems Information Science Practice
Group Report

プロジェクト名

生体信号を利用した身体拡張インターフェース〜ASHURA〜

Project Name

Body augmentation interface using biologic signals 〜ASHURA〜

グループ名

グループ C

Group Name

Group C

プロジェクト番号/Project No.

14-C

プロジェクトリーダ/Project Leader

池田考太　Kouta Ikeda

グループリーダ/Group Leader

渥美星汰　Shota Atsumi

グループメンバ/Group Member

渥美星汰　Shota Atsumi
佐々木勇仁　Hayato Sasaki
清水蒼太　Sota Shimizu

指導教員

櫻沢繁　髙木清二　山田恭史　山田浩　辻義人

Advisor

Sigeru Sakurazawa　Seiji Takagi　Yasufumi Yamada　Hiroshi Yamada　Yoshihito Tsuji

提出日

2025 年 1 月 21 日

Date of Submission

January 1, 2025

概要

演奏者には自身が理想とする演奏を実現できることを望んでいる．そのために，演奏者は様々な演奏表現を行っている．例として，エフェクターを用いた楽器の音の変化や，照明演出といったライティングがある．本グループは，演奏者が理想とする演奏と実際の演奏の再現率の高さに起因する高揚感を，vibration という単語の（人・物・場所から受ける）感じ，雰囲気という意味からそのまま，ノリや雰囲気といった意味で使われる言葉であるバイブスという言葉として定義した．本グループの目的はこのバイブスを上げる，つまり，演奏者が期待する演奏と，実際の演奏の再現率を高めることことである．そして，再現率を上げるには，演奏者の演奏の能力を向上させる必要がある．その補助は，本グループは音色（エフェクター）と照明（ライティング）の，演奏者自身が演奏中に操作することは難しいという問題を解決することで，演奏者が可能な演奏表現を拡張することで可能だと考えた．これはフロー体験という挑戦と能力が高いところで一致しているときに起こる没頭状態に着目した結果だ．バイブスが演奏者が理想とする演奏と実際の演奏の再現率の高さに起因する高揚感であるなら，フロー体験に至る状態は適していると考えられる．また，フロー体験には挑戦の難易度が上昇した際に，能力の向上を図ることで，再度フロー体験に至ろうとするため，フロー体験は新しいレベルの挑戦と能力を発展させる働きがある．しかし，能力を向上させるためには，新しい能力を学習する必要がある．本グループは演奏表現の拡張により，新しい能力を学習しやすくすることで，演奏者は能力を発展させようとするフロー体験の働きを受けやすくなり，かつ演奏者は可能な演奏表現が増えているため，より高いレベルの演奏を実演できるようにし，バイブスを上げやすくなると考えた．そして，本グループは筋肉が活動する際に計測ができる筋電位という生体信号の一種である電気信号を用いることで，演奏者の演奏動作をエフェクターやライティングの操作に用いることが可能ではないかと考えた．そのために，演奏者の筋電位を電極や計測回路を通して計測し，Arduino がシリアル信号に変換する．そうして計測した筋電位の値に応じて，プログラムがモーターの回転角度と LED の色を決定する．モーターはエフェクターのつまみを操作するという動作機構を製作し，実際に使用して演奏する実験を行った．実験では両前腕と肩から首元にかけての 3 か所から筋電位を計測し，エフェクトはディレイ，ディストーション，オーバードライブの 3 種類にそれぞれ適用させた．実験は動作を抑えて演奏した場合と動作を加えて演奏した場合で行った．結果は，動作の有無に応じて音の変化具合と LED の色に違いは確認できたが，実験の途中から筋電位を計測していない場合でもサーボモーターが回転してしまうことがあったため，筋電位によるエフェクトのかかり具合の調整ができているか確認しづらくなった．また，LED の色も筋電位が最大値を計測していても，本来なる筈の赤にならない場合があった．しかし，実験に協力してくれた演奏者からは今回の実験に協力してくれた 1 名の演奏者からは荒削りな技術だが，普段のギター演奏では有り得ない音に，この技術の可能性を感じたという意見を得た．これらの問題点はより緻密なモーター制御を行い，LED の色の変化の条件を更に正確にできれば解決できると考えられる．よって，一か所から計測し，一種類のエフェクトを制御することは可能でも，複数箇所から計測し，それらを利用してエフェクターを制御することは，より緻密なモーター制御を行う必要があることが分かった．しかし，複数での完璧なモーター制御を実現できた場合は，筋電位によるエフェクター，ひいてはエフェクトの制御は不可能ではないことも分かった．そもそも演奏することができないということもなかった．そのため，上記の問題点を改善することができれば，筋電位によるエフェクターとライティングの制御は不可能ではなく，演奏者に更なる演奏表現をもたらすことは可能だと考えられる．また，問題点を改善したうえで実際に演奏に利用し，演奏者が期待する演奏を実現する補助がどれほど可能なのかを調べる必要があると考える．

キーワード　エフェクター, バイブス, フロー体験, 筋電位

（※文責: 渥美星汰）

Abstract

Performers want to be able to realize their own ideal performance. To achieve this, performers use a variety of performance expressions. Examples include changes in the sound of instruments using effectors and lighting effects. This group has defined the sense of elation resulting from the high reproducibility between the performer's ideal performance and the actual performance as "vibes," a word used in the sense of "groove" or "atmosphere," directly from the word vibration, which means "feeling" or "atmosphere" (received from a person, thing, or place). The purpose of this group is to increase the vibes, that is, to increase the reproduction rate between the performer's expected performance and the actual performance. In order to increase the reproducibility, it is necessary to improve the performer's performance ability. The group believes that this can be aided by solving the problem of timbre (effectors) and lighting (lighting), which are difficult for performers to manipulate during their own performances, and by expanding the range of performance expressions available to them. This is the result of focusing on the flow experience, a state of immersion that occurs when challenge and ability are highly aligned. If vibes are a feeling of elation caused by the high reproducibility between the performer's ideal performance and the actual performance, the state that leads to flow experience is considered to be suitable. In addition, the flow experience has the function of developing a new level of challenge and ability, because the performer tries to reach the flow experience again by improving his/her ability when the difficulty of the challenge increases. However, in order to improve one's ability, it is necessary to learn a new ability. The group believes that by facilitating the learning of new abilities through the expansion of performance expression, performers will be more receptive to the flow experience that tries to develop their abilities, and since the number of possible performance expressions has increased, performers will be able to perform at a higher level and raise their vibes more easily. The group considered that it would be possible to use the performer's performance movements to manipulate effectors and lighting by using electrical signals called myoelectric potentials, a type of biological signal that can be measured when muscles are active. For this purpose, the performer's myopotential is measured through electrodes and measurement circuits, and converted into a serial signal by Arduino. The program then determines the rotation angle of the motor and the color of the LEDs according to the measured Myoelectric potential values. The motor operates the knob of the effector, and an experiment was conducted in which the motor was actually used to play the music. In the experiment, electromyograms were measured from three locations on both forearms and from the shoulders to the neck, and three types of effects (delay, distortion, and overdrive) were applied to each. Experiments were performed with and without motion. The results showed that there were differences in the degree of sound change and the color of the LEDs depending on the presence or absence of movement. However, during the experiment, the servomotors sometimes rotated even when the Myoelectric potential was not measured, making it difficult to confirm whether the effects of the Myoelectric potential effects could be adjusted. In addition, the LED color sometimes did not turn red as it should have even when the maximum myopotential value was measured. However, one performer who cooperated in the experiment commented that although the technique was rough, he felt that the sound, which was not possible with ordinary guitar playing, showed the potential of this technique. These problems could be solved with more precise motor control and more accurate conditions for LED color changes. Thus, while it is possible to measure from a single location and control a single type of effector, it is necessary to measure from multiple locations and use them to control the effector, which requires more precise motor control. However, we also found that if perfect motor control could be achieved at multiple locations, controlling effectors and lighting using myopotentials would not be impossible, and would allow the performer to express his or her performance even more. Therefore, if the above problems can be improved, it is not impossible to control effectors and lighting using myoelectric potentials, and it is possible to bring about further performance expression to performers. It is also necessary to investigate how much assistance is possible to realize the performance expected by the performer by actually using the system in the performance after improving the problems.
Translated with DeepL.com (free version)

Keyword Effector unit, Vibes, flow experience, Myoelectric potential

（※文責: 渥美星汰）

目次

第 1 章　はじめに

1.1　背景

演奏者は自身が理想とする演奏ができることを望んでいる．そして，演奏者は演奏表現を変化させることで，個々に食い違いはあれど聞き手にニュアンスの違いを伝えることができるとされている [1]．例として，ギターを用いた演奏において，演奏者は感情によって演奏動作を区別し表現していることが示唆されている [2]．また，エレキギターで演奏をするときに，演奏者はエフェクターという装置を用いることでギターの音を変化させ，更なる表現を可能としている．エフェクターを用いた音の変化をエフェクトといい，ギターの音波を過大増幅させてクリッピングするものや，再生を元の音波から一定の時間遅らせた複製音波を再生ことで, やまびこのように知覚できる反響音を演出するものがある．本グループは，これらの演奏動作を区別することや演奏表現を変化させる取り組みは，演奏者自身が求める演奏を実現するために行われていると考えた．バイブスとは，vibration という単語の（人・物・場所から受ける）感じ，雰囲気という意味 [3] からそのまま，ノリや雰囲気といった意味で使われる言葉である．本グループは，受ける感じや雰囲気を，対象に抱く期待から生まれるものと解釈し，バイブスを期待したイメージと実際の再現率の高さに起因する高揚感と定義した．そして，演奏によるバイブスの向上，つまり演奏者が期待する演奏と，実際の演奏の再現率を高めることを目的とした．

実際の演奏の再現率を高くするために，本グループはフロー体験に着目した．フロー体験とは，挑戦することが自身の能力でちょうど達成できることと釣り合ったときに感じる、深い集中と充足感を伴う没頭状態を指す [4]．この挑戦することと自身の能力の釣り合いは，高いところで一致していなければフロー体験にならないとされる [4]．フロー体験が挑戦と能力が高いところで一致しているときに起こる没頭状態とすると，バイブスが上がる状態とはフロー体験が起きる状況と類似していると考えられる．よって，フロー体験の維持はバイブスの向上に繋がると考えられる．また，挑戦の難易度が高いならば新しい能力を学習すること，つまり能力を向上させることで，能力が高い場合は挑戦の難易度を上げることでフロー体験に戻れるために，フロー体験は新しいレベルの挑戦と能力を発展させるとされる [4]．即ち，フロー体験になる状態の維持は，演奏者の演奏の能力の向上につながり，バイブスが上がる状態に至りやすくなるとも考えられる．しかし，演奏者や聴衆が求める演奏のレベルが，演奏者の能力に対して高いということが起こり得る．挑戦の難易度が高い場合，人々は新たなスキルを獲得することで再びフロー体験に至れる状態に戻れる [4]．しかし，人が能力を学習する等で自身の能力を向上させ続けるには当然限界がある．そのため，本グループは演奏者が可能な演奏表現を増やし，演奏表現を拡張することで，演奏者の能力に対し演奏のレベルが高く，フロー体験を得ることができない場合を解消しやすくする．つまり，フロー体験に至りやすくなることで演奏者は，能力を発展させようとするフロー体験の働きを受けやすくなり，かつ演奏者は可能な演奏表現が増えているため，より高いレベルの演奏を実演できるようにし，バイブスを上げやすくすると考えた．

より豊かな音楽表現を行う方法として，音色（エフェクター）によるアプローチ，照明（ライティング）によるアプローチが考えられる．現代の商業・娯楽音楽の制作現場では，エフェクターによる積極的な音づくりを行うことが当たり前となっており，更には本来想定されていた使い方

を超えた手法によって新たなサウンド／音楽ジャンルが生み出される状況にもなっている [5]．また，有彩色とテンポには速い順に，赤，橙，緑，青緑，青，紫と関係があり，同じ感情が想起されることが示唆された [6]．その際に，エフェクターは主につまみを回して音の変化具合を調整しており，ボタンを押すことでオンオフの切り替えをしている．よって，演奏者がエフェクターを操作する場合、エフェクターによる楽器の音の変化は予め設定したものに限られ，できる操作は主に足でのオンオフの切り替えのみとなる．その操作も演奏中では演奏動作に制限がかかる．ライティングに関しても，演奏者自らが演奏中に照明の色を変更するといった操作をすることは困難である．こういった問題を解決することで演奏表現を拡張し，今までの演奏者にはできない演奏ができるようにならないかと考えた．そして，エフェクターやライティングに関して動作が制限される，そもそも演奏者単独では操作できないという問題は，エフェクターやライティングを，演奏者の演奏表現と連動させることで解決できるのではないかと考えた．

そのために本グループは次に身体の拡張に着目した．身体拡張は，暦本（2021）によると，ヒューマンオーグメンテーション（人間拡張）のひとつの領域として定義することが可能である [7]．ヒューマンオーグメンテーションとは人間の能力をテクノロジーによって自由に増強・拡張させる技術を扱う学問領域であり，拡張の対象として，知的な能力に加え，身体能力や人間の存在などを包含する [7]．ヒューマンオーグメンテーション分野では，身体拡張の事例として，外骨格のように構造的に身体能力を増強するもの，機能性電気刺激（FES）によって筋肉を駆動するもの，義手・義足のように身体機能を補綴するものなどが例に挙げられる [7]．また，上田（2011）は「外界の対象を自分の身体として認識すること」を「身体拡張」としている [8]．このように，身体拡張とはテクノロジーによって増強・拡張された能力を自身の身体のように認識・扱うことができることと定義できる．こういった拡張技術を用いた取り組みは，音楽分野でも行われている [9]．こうした拡張技術の中でも、本グループは筋肉が活動する際に計測ができる筋電位という生体信号の一種である電気信号を用いることで，演奏者の演奏動作をエフェクターやライティングの操作に用いることが可能ではないかと考えた．

よって本グループは，筋電位という生体信号の一つである，筋肉が活動しようとする際に生じる電気信号を利用し，エフェクターや LED テープの光の色の変化を演奏者の動きで制御させることで，エフェクターの操作のために足を使う必要をなくし，既存のエフェクターの動きの制限をなくす．そして，音や光の変化を演奏中でも演奏者のみでも調整可能にし，演奏者が演奏中に可能な表現の幅を広げ，フロー体験に至りやすくし，演奏者の演奏能力の向上を補助する．そうすることで，バイブスを上げやすくすることを試みた．

（※文責: 渥美星汰）

1.2　課題

本グループは，バイブスが上がりやすくすることを目的とした．そのために，演奏者が自身の演奏動作によって音や光を調整可能にし，演奏表現を拡張する．その方法として筋電位を利用する．

その課題として，筋電位をエフェクターや光の色の変化の制御に利用するうえで，どの箇所の筋電位をどのように利用して，どのようなエフェクターの制御を行うことで，演奏者が演奏表現の幅が拡張されたと感じるかを調べることがある．

（※文責: 渥美星汰）

1.3　製作するデバイスの構想・その重要性

本グループは，演奏者の動きで音や光の色の制御を可能とすることで，演奏者のみでも可能な演奏表現を増やし，演奏の能力を向上しやすくする．バイブスを上げやすくすることを目的とした．そのために，演奏者の動作を筋電位で計測するための回路を製作し，計測された筋電位によってサーボモーターを任意の角度まで回転させるプログラム，計測された筋電位によって LED の色を変化させるプログラム，エフェクターによる音の変化の強弱を制御するためのつまみ部分をサーボモーターで操作するためのアタッチメント，サーボモーターを固定するための装置，LED テープを演奏者に装着するための装置，エフェクターを自身で操作していることを強調するための手を模倣した装飾を構想した．これらが機能することで，演奏者が演奏者のみでエフェクターや LED の色を制御することができるようになり，演奏者のみでも可能な表現を増やすことができる．演奏者自身が望んだ演奏の実現や演奏の能力の向上を補助したことにより，バイブスが上がりやすくなると考える．

（※文責: 渥美星汰）

第 2 章　実装方法と関連した知識や技術

2.1　用いた方法とその関連知識

2.1.1　筋電位

筋電位とは，筋肉が収縮する際に生ずる活動電位である．脳や脊髄からの筋肉収縮命令によって運動神経が興奮状態になり，神経軸索から神経筋接合部にかけて伝わったことで，筋細胞内外のイオンの異なる分布により発生している電位差が，わずかに正に向かうことで，細胞膜内外でイオンの交換を行うことができるチャネルたんぱく質の活動で，その電位差が逆転することで生じる電位変化のことである．この際に，筋繊維内に通ずる T 管に伝播することでカルシウム小胞からカルシウムイオンが放出され，トロポニンが受容することでアクチン－ミオシンの相互抑制作用が解除され，アクチン - ミオシンの滑り運動により，筋収縮が起こる．これは筋繊維一つ一つ毎に様々な間合いで生じる．

（※文責: 佐々木勇仁）

2.1.2　筋電位計測

筋電位の計測方法として，表面筋電位を計測する手法を用いた．この手法では，対象の筋肉の上の皮膚表面に電極を貼り，電極周辺で発生する筋電位の合算値を計測する．ヒト皮膚電気抵抗の高さにより計測される電位の値が小さいことや，計測環境にある周辺機器の電源による電磁誘導の影響を受けやすいなどの問題点がある．この問題点を解決するために、アクティブ電極，差動増幅器，ハイパスフィルタ，半波整流器，ローパスフィルタ，非反転増幅器を構成要素とする回路を構築した．

（※文責: 佐々木勇仁）

2.1.3　アクティブ電極

本来、ヒトの皮膚表層から電極を用いて筋電位を計測しようとすると、皮膚の電気抵抗の高さによって筋電位信号が微弱化される。アクティブ電極とは，電極側の電気抵抗をおよそ無限大に保つことで信号の微弱化を防ぐエミッタフォロワ回路が搭載された電極のことである．増幅率は１で，入力電圧をそのまま出力する．

（※文責: 佐々木勇仁）

2.1.4　差動増幅器

差動増幅器とは，与えられた２入力の差を増幅する回路である．この２入力の電位が同じときは，０を出力する．これにより，計測環境下にある周辺機器の電源による電磁誘導により発生し

た．電位差が測定対象の電位差にノイズとして混入してしまう問題を解消した．今回の場合は，皮膚表面に貼った２つの電極からの入力を対象としている．

（※文責: 佐々木勇仁）

2.1.5　ハイパスフィルタ

ハイパスフィルタとは，設定した遮断周波数より高い周波数の交流電圧の入力を減衰させず，低い周波数の交流電圧の入力を減衰させる性質を持つフィルタ回路の一つである．

（※文責: 佐々木勇仁）

2.1.6　半波整流器

半波整流器とは，交流を直流に変換し正の電圧のみ出力する回路である．今回の場合では，入力を Arduino で扱えるようにするために用いる．Arduino は低電圧の直流回路を前提に設計されているため，交流では動作が安定せず，破損の恐れがある．

（※文責: 佐々木勇仁）

2.1.7　ローパスフィルタ

ローパスフィルタとは，設定した遮断周波数より低い周波数の交流電圧の入力を減衰させず，高い周波数の交流電圧の入力を減衰させる性質を持つフィルタ回路の一つである．包絡検波に用いる回路であり，今回の場合では入力された筋電位の振幅を出力するために用いられる．

（※文責: 佐々木勇仁）

2.1.8　非反転増幅器

非反転増幅器とは，入力と出力の極性を変えずに増幅を行う回路のことである．今回の場合では，ハイパスフィルタやローパスフィルタの出力の増幅に用いた．

（※文責: 佐々木勇仁）

2.1.9　LED リボン

体に装着した LED ライトの光の変化を利用して感情を表現する試みを行った．複数チャンネルで計測した筋電位の中から，振幅が最も大きいものを入力として採用し，その値を計算によって RGB の数値指定に適した形式に変換した．この数値計算および色変換の指示は C 言語で記述し，Arduino に書き込んだ．そして，Arduino の出力を基に LED テープの色を変化させる仕組みを構築した．なお，LED テープはそのままでは体への装着が困難であったため，リボンに縫い付けることで装着可能な形態とした．

（※文責: 佐々木勇仁）

2.1.10　つまみ制御用モーター

エフェクターのつまみをモーターで回転させることで音色を変化させる機構を採用した．筋電位の振幅値を測定し，その値をもとに Arduino で C 言語で記述した計算式を用いて回転秒数を求め，360°サーボモーターを回転させた．

（※文責: 佐々木勇仁）

2.1.11　モーター固定用アクリルボックス

図 2.1　アクリルケース

エフェクターのつまみや 3D ハンドモデルを回転させる際に使用するモーターは，固定しないと回転軸だけでなくモーター本体も回ってしまい，つまみを回す動作が行えない．この問題を解決するため，アクリル板で固定具を作成した．具体的には，エフェクターにかぶせるアクリル製の箱を作り，その箱にモーターのサイズに合った穴を空けてモーターを固定した．

（※文責: 佐々木勇仁）

2.1.12　3D プリントによるパーツの製作

本製作物では，モータの回転でエフェクターのつまみを制御するためのパーツが必要であった．そこで，CAD を用いて 3D モデルを製作し，3D プリンタを用いて印刷し，製作した．また，本制作物の動作が使用者の身体に結びついているという認識を向上させるために，人間の手の形のパーツを製作し，エフェクターに用いた．

（※文責: 清水蒼太）

2.1.13　3D ハンドモデル

エフェクターのつまみが回転していることを視覚的に分かりやすくするため，また作成物の見栄えをよくする目的で，3D プリンターで物をつまんでいる人間の手のモデルを作成した．この手のモデルをモーターに取り付け，エフェクターのつまみの回転と連動させる仕組みにした．

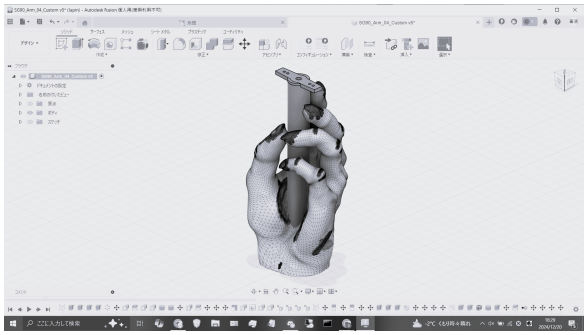

図 2.2　3D ハンドモデル

（※文責: 佐々木勇仁）

2.1.14　Arduino とシリアル通信

本製作物では，筋電位計測回路によって処理された信号をデジタル信号に変換し，エフェクターを制御する必要があった．また，デジタル信号をエフェクターに送信する必要があった．そこで，A/D 変換を行うことができ，シリアル通信が可能な Arduino Uno を用いた．シリアル通信とは，デジタルデバイス間でデータを送受信するための基本的な通信方法である．シリアル通信は，データを 1 ビットずつ順次送信する方式であり，シンプルで効率的なデータ交換が可能である．PC と Arduino 間でのシリアル通信プロトコルには，UART（Universal Asynchronous Receiver/Transmitter）が用いられている．

（※文責: 清水蒼太）

2.1.15　Python と PyAudio によるリアルタイム音声処理

Python は，その単純さとライブラリの豊富さから，様々な分野で広く使用されているプログラミング言語である．本プロジェクトでは，Python の利点を活かして，筋電位の変化を測定し，リアルタイムで音声にエフェクトをかけるシステムを構築した．その中で，PyAudio は音声データの入出力を効率的に行うためのライブラリとして利用された．また，Serial は Arduino とシリアル通信するためのライブラリとして利用された．PyAudio は，Python で音声データを扱うためのライブラリであり，PortAudio というクロスプラットフォームの音声処理ライブラリの Python バインディングである．PyAudio を用いることで，音声の入出力，録音，再生を簡単に行うことができる．特に，音声データのリアルタイム処理が可能であり，低遅延での音声エフェクトの適用ができる点が特徴である．

（※文責: 清水蒼太）

2.1.16　制御プログラム

また，筋電位計測回路の信号を Arduino で認識し，その信号電圧の大きさを Python のコードを実行しているコンピュータにシリアル通信で送信するプログラムを作成した．

（※文責: 清水蒼太）

2.1.17　モーターの制御プログラム

本製作物では，筋電位の値に連動して回転するモータを製作する必要があった．そこで，筋電位計測回路の信号を Arduino で認識し，Servo ライブラリを用いてその信号電圧の大きさを回転角に変換するプログラムを作成した．本制作物では速度制御式の無限回転サーボモータを疑似的に角度制御するためのプログラムを製作した．

（※文責: 清水蒼太）

2.1.18　LED の制御プログラム

本製作物では，筋電位の値に連動して LED の色を変化させる必要があった．そこで，筋電位計測回路の信号を Arduino で認識し，AdafruitNeoPixel ライブラリを用いてその信号電圧の大きさを RGB 値に変換し，LED テープの色を制御するプログラムを製作した．

（※文責: 清水蒼太）

2.2　エフェクターとエフェクト

エフェクターとはエフェクトを調整するものである．そしてエフェクトとは，音波を意図的に変形させるものである．例えば，ディストーションというエフェクトが存在する．ディストーションは適切に使用された場合，聴衆の感情的反応や関心が高まることが確認されている [8]．特にロックやメタルなどの分野では，ディストーションが曲の活力や動的要素を強調し，リスナーに強い印象を与えることが示されている [8]．

（※文責: 清水蒼太）

2.2.1　クリッピングによる歪み知覚の原理とその利用法

クリッピングとは，音波の振幅が一定のしきい値を超えた際にその部分を切り落として平らにすることである．これにより音波の形状が変形し，結果として音に歪みが生じる．図 2.3 の上が元の音波，下がクリッピング後のものである．

ディストーションは音波を過大増幅させてクリッピングすることによって歪みを得るエフェクトである．歪みは音波の波形が変化することによって，音割れのように音色が変化することを指す．具体的には，クリッピングが発生するほど音波を増幅すると元の音波から異なる形状，例えば矩形波に近い形に変わる．図 2．3 は矩形波と周波数の関係を表したものである。その結果，非線形変形するため新たに基本周波数の整数倍の周波数成分が生成される．聴衆は基本周波数と共に新たに生成された倍音成分も感知する．それによって聴覚的に歪んだと感じる．

オーバードライブというエフェクトも音波を過大増幅させてクリッピングすることによって歪みを加えるエフェクトである．ディストーションと同様に音波をクリッピングするが，その増幅の仕方や歪みの程度に違いがある．オーバードライブは，音波を軽く圧縮し，過度に歪ませることなく温かみのある音色を生成することを目的とする．オーバードライブでは，音波がクリッピングに達

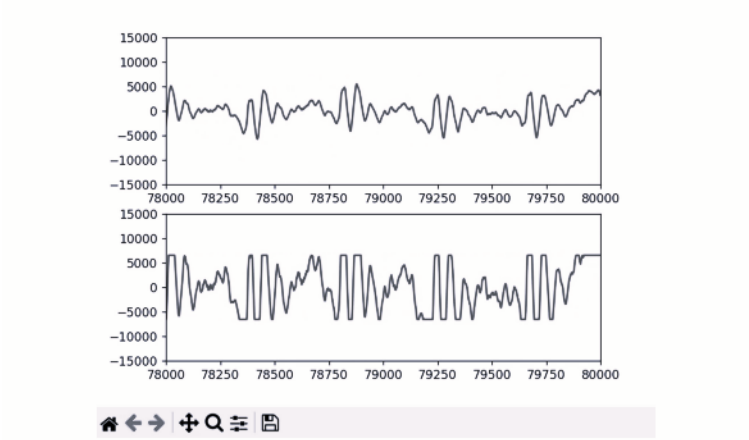

図 2.3　クリッピング

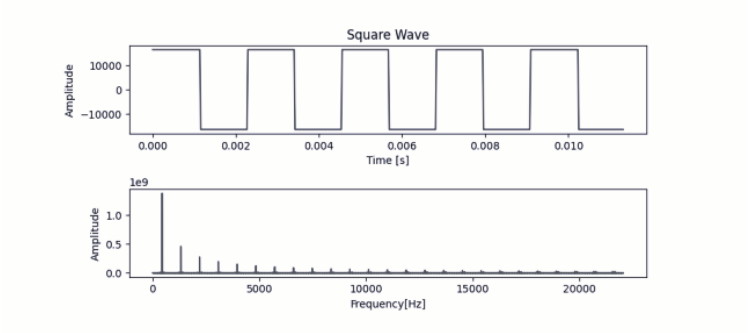

図 2.4　矩形波

する手前の段階で歪みが発生し，音に豊かな倍音成分が加わる．これは，元の音波が高い振幅で増幅されることにより，波形が部分的に切り取られ，波形が変形することで発生する．オーバードライブは完全な矩形波にはせず，元の音に対して穏やかな歪みを加えるだけで，音が過度な倍音成分の混入は行わない．このため，オーバードライブはギターなどの楽器において，音程の知覚に重要な基本音成分を相対的に支配的に残しつつ，音に豊かな響きを加えるために使用される．

（※文責: 清水蒼太）

2.2.2　ディレイの仕組み

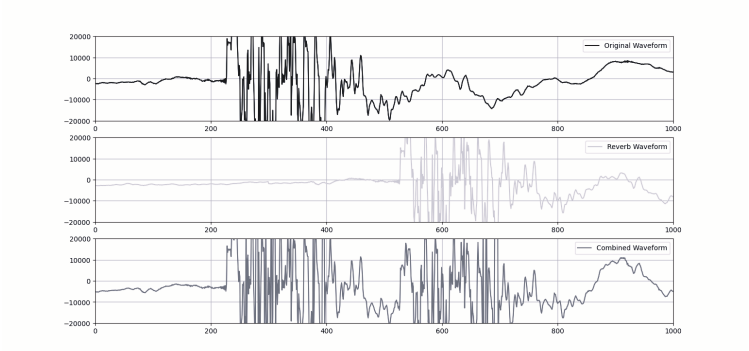

図 2.5　ディレイ

ディレイはもともとの音波と，一定時間遅延させた複製音波を再生することによって，空間的な

広がりやリズム感を強調するエフェクトである．具体的には，音波の再生を元の音波から一定時間遅らせることで，やまびこのように知覚できる反響音を演出することを可能にする．この音の重なりにより，リズムの強調や深みが加わり，聴覚的に広がりを感じることができる．さらに，遅延信号を繰り返し再生するフィードバック処理を調節することで音の反復回数が増し，反響音が持続するような効果を得ることができる．その結果，音の質感や印象を大きく変化させる．

（※文責: 清水蒼太）

2.3 技術

2.3.1 Arduino によるモーター制御

Arduino を用いて、筋電位の値に応じてモーターの回転角度を制御した．また，使用したモーターは速度制御サーボモーターであったため，疑似的に角度制御サーボモーターのように制御するためのプログラムを制作した．具体的な手順を以下に示す．

(1) 筋電位計測回路で処理された信号を Arduino Uno に入力し，その信号を A/D 変換によってデジタル化する．これにより，0V から 5V の筋電位アナログ信号を 0 から 1023 の範囲のデジタル値に変換する．

(2) デジタル値に基づいてサーボモーターの回転角度と回転時間を決定する．LED を 6 色で表現するため，サーボの回転角度制御の範囲を 6 等分し，それぞれの範囲に対応する角度を 6 段階で変換する．具体的には，エフェクターのつまみ部分は約 270 度回転させることが可能である．したがって，1023/6 は 170．5 であり，270/5 は 54 であるため，次のように条件を設定した．

- デジタル値が 0 から 169 の時，角度は 0 度.
- デジタル値が 170 から 340 の時，角度は 54 度.
- デジタル値が 341 から 510 の時，角度は 108 度.
- デジタル値が 511 から 681 の時，角度は 162 度.
- デジタル値が 682 から 851 の時，角度は 216 度.
- デジタル値が 852 から 1023 の時，角度は 270 度.

(3) エフェクター制御用のサーボモーターについて実験の結果，細かな制御ではなく，大きな変化を伴う制御が適していることが分かった．したがって，エフェクター用サーボモーターは 3 段階で分割し，それぞれの分割閾値および回転角度は，実際の演奏者の筋電位を見ながら調整した．

(4)Arduino では回転前の角度を記録しており，回転前後の角度の情報から回転方向と回転時間を決定する．具体的には，回転前の角度と回転後の角度の差を計算することで，何度回転するかを求める．計算結果の正負の符号から時計周りか反時計回りのどちらに回転するかを決定する．また，計算結果に 300/180 を掛け算することで回転時間を決定する．これは，180 度回転するのに約 300 ミリ秒の時間を要するためである．

(5) 上記の計算から求められた回転方向と回転時間の通りに回転させることで，計算結果と同様の角度を回転することができ，疑似的に角度制御サーボモーターのような制御を行うことが出来る．

詳細なコードについては付録に記載する．本制御により，速度制御サーボモーターを疑似的に角度制御サーボモーターのように扱うことが可能となった．また，筋電位の値に基づいてサーボモーターを制御することも実現した．この制御をモーターごとに繰り返すことで，複数のモーターを個別に制御することが可能となる．

（※文責: 清水蒼太）

2.3.2 Arduino による LED 制御

Arduino を用いて，筋電位の変化に応じて LED テープライトの色を制御するプログラムを制作した．具体的な手順を以下に示す．

(1) 筋電位計測回路で処理された信号を Arduino Uno に入力し，その信号を A/D 変換によってデジタル化する．これにより，0V から 5V の筋電位アナログ信号を 0 から 1023 の範囲のデジタル値に変換する．

(2) 複数箇所で計測した筋電位のデジタル値のうち，最も大きい値を参照する．本製作物では，三箇所から筋電位を測定し，最も高いデジタル値に基づいて LED の色を変化させる．

(3) 筋電位のデジタル値に応じて LED の色を変化させる．これを基に，0 から 1023 のデジタル値を 6 等分し，それぞれに対応する LED の色を設定した，具体的には，1023/5 は 204．6 であるため，次のように条件を設定した．

- デジタル値が 0 の時，LED の色は紫．
- デジタル値が 204 の時，LED の色は青．
- デジタル値が 409 の時，LED の色は青緑．
- デジタル値が 613 の時，LED の色は緑．
- デジタル値が 818 の時，LED の色は橙．
- デジタル値が 1023 の時，LED の色は赤．

(4) 各色への遷移が滑らかに行われるように制御を行った．例えば，デジタル値が 0 から 204 に変化する際，LED は紫から青へ滑らかに遷移する．

詳細なコードについては付録に記載する．本制御により，複数の LED テープライトを同時に同じ色で点灯させることが可能となる．

（※文責: 清水蒼太）

第 3 章　プロジェクトの結果

3.1　成果物

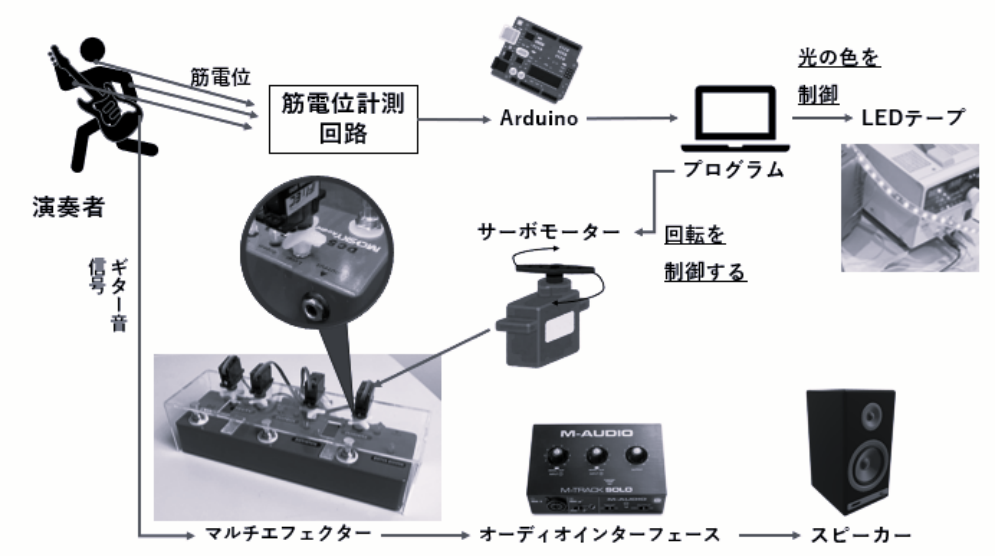

図 3.1　全体のシステム図

本グループは，既存の演奏では困難な表現を，演奏者が演奏に伴って行う動作を楽器の音と演奏者に装着させた LED テープの光の色に反映させることで表現可能になると考えた．そして，演奏者の演奏表現を拡張したことによるフロー体験の維持から，演奏者の能力の向上を補助し，バイブスを上げやすくすることを目的とした．

そのために，演奏者の筋電位を計測するための回路を製作した．また，計測した筋電位をシリアル信号に変換し，プログラムで利用できるようにするために Arduino を導入した．そして，計測した筋電位によってサーボモーターの回転や LED の色を制御するためのプログラムも制作した．これらの動作機構は演奏者の筋電位を電極や計測回路を通して計測し，Arduino がシリアル信号に変換する．そうして計測した筋電位の値に応じて，プログラムがモーターの回転角度と LED の色を決定する．モーターはエフェクターのつまみを操作する役割である．これらの関係を図 3.1 のシステム図に図示した．

3.2　実験

本グループは，成果物を用いることで演奏表現の拡張が可能かを，実際に演奏して実験をした．実験で制御することになったエフェクトはディレイ，ディストーション，オーバードライブの 3 種類である．これらは，事前の準備段階で行った手動によってエフェクターのつまみを回し，ディレイは従来の使用方法ではできないであろう音を出すことができる，ディストーションとオーバードライブは歪みのあるなしにより音の変化が分かりやすいため選択した．計測箇所は，下半身では筋電位の計測に用いられた電極と回路間のケーブルを踏んでしまう危険性があったため，上半身に限定した．その中でも，電極の張り付けがしやすい両前腕，肩から首元が候補として挙がった．演奏

者の任意のタイミングで制御したいという要望から，両前腕ではエレキギターの演奏中に筋電位を計測させないことが難しいため，左肩をディレイに対応させた．そして，左前腕をディストーション，右前腕をオーバードライブに対応させた．LED の色の制御は 3 か所から計測した筋電位で最も大きいものを反映させた．そして，演奏者には動きを抑える場合と動きを加える場合の 2 通りで演奏してもらった．

3.3　実験の結果

動きを抑えて演奏してもらった場合では，3V 未満の筋電位を計測した時間は全体 22 秒に対し約 20 秒であった．そして，LED の色が青から緑であった時間の割合は全体が 22 秒に対し、約 20 秒だった．緑だった場合を除くと 14 秒だった．また動きを加えて演奏してもらった場合では，3V 以上の筋電位を計測した時間は全体 44 秒に対し，約 42 秒だった．LED の色は，黄（緑から橙にかけて変化する場合の筋電位を計測するとなる場合がある）から赤であった割合が，全体 44 秒に対し，約 17 秒だった．実験の初期時点では，エフェクトのかかり具合にも差は確認できた．しかし，実験の途中から筋電位を計測していない場合でもサーボモーターが回転してしまうことがあったため，筋電位によるエフェクトのかかり具合の調整ができているか確認しづらくなった．また，LED の色も筋電位が最大値を計測していても，本来なる筈の赤にならない場合があった．今回の実験に協力してくれた 1 名の演奏者からは荒削りな技術だが，普段のギター演奏では有り得ない音に，この技術の可能性を感じたという意見を得た．

（※文責: 渥美星汰）

第 4 章　考察

4.1　プロジェクトの結果から

今回の実験では，不安定な挙動が多かった．モーターが筋電位を計測していないときでも動いてしまうのは、速度制御サーボモーターをプログラムの制御で角度制御モーターのように扱う必要があったが，その制御方法では，複数の筋電位計測回路とパソコンをつないだことによって生じた電気的ノイズの影響を受けてしまったためと考えられる．LED の色が本来なる筈の色に変化しなかったのは，プログラム側の条件で筋電位が最大値になると赤にするとしていたが，実験時の回路によって計測される最大値が，想定した最大値に僅かに届かず，最大値を計測しても赤にならないということが考えられる．

プロジェクトの結果から，一か所から計測し，一種類のエフェクトを制御することは可能でも，複数箇所から計測し，それらを利用してエフェクターを制御することは，より緻密なモーター制御を行う必要があることが分かった．しかし，複数での完璧なモーター制御を実現できた場合は，筋電位によるエフェクター，ひいてはエフェクトの制御は不可能ではないことも分かった．また，エフェクトをプログラムで再現する場合に存在した音の遅延はなく，そもそも演奏することができないということはなかった．そのため，上記の問題点を改善することができれば，筋電位によるエフェクターとライティングの制御は不可能ではなく，演奏者に更なる演奏表現をもたらすことは可能だと分かった．そのためには，より緻密なモーター制御を行い，LED の色の変化の条件を更に正確にする必要がある．また，問題点を改善したうえで実際に演奏に利用し，演奏者が期待する演奏を実現する補助がどれほど可能なのかを調べる必要があると考える．

付録 A　プログラムリスト

A.1　サーボモーター制御

```
#include <Servo.h>
#include <Adafruit_NeoPixel.h>

/*
#筋電位関係の変数
VoltReadPin：アナログピンの番号
value：電圧を 0 から 1023 のデジタル値として扱う
volt：デジタル値を 0.00 から 5.00 のに変換して扱う
*/
int VoltReadPin1 = 1;
int value1 = 0;
float volt1 = 0;
int VoltReadPin2 = 2;
int value2 = 0;
float volt2 = 0;
int VoltReadPin3 = 3;
int value3 = 0;
float volt3 = 0;
int VoltReadPin4 = 4;
int value4 = 0;
float volt4 = 0;

/*
#筋電位関係の変数
ServoPin：デジタルピンの番号
beforeAngle：角度制御のための変数
afterAngle：角度制御のための変数、これを更新していく
lotateTime：時間制御のための変数、これを更新していく
stopAngle：回転が停止する数値、90 前後で個体差あり
beforeAngle と afterAngle は、lotateAngle が初期位置
*/
//エフェクター用サーボ
const int ServoPin1 = 7;
int beforeAngle1 = 270;
int afterAngle1 = 270;
```

```
int lotateTime1 = 0;
int stopAngle1 = 93;
const int ServoPin2 = 6;
int beforeAngle2 = 270;
int afterAngle2 = 270;
int lotateTime2 = 0;
int stopAngle2 = 92;
const int ServoPin3 = 4;
int beforeAngle3 = 270;
int afterAngle3 = 270;
int lotateTime3 = 0;
int stopAngle3 = 90;
const int ServoPin4 = 5;
int beforeAngle4 = 270;
int afterAngle4 = 270;
int lotateTime4 = 0;
int stopAngle4 = 94;
//装飾用サーボ
const int ServoPin5 = 10;
int beforeAngle5 = 0;
int afterAngle5 = 0;
int lotateTime5 = 0;
int stopAngle5 = 88;
const int ServoPin6 = 11;
int beforeAngle6 = 0;
int afterAngle6 = 0;
int lotateTime6 = 0;
int stopAngle6 = 90;

int lotateAngle1 = 270;
int lotateAngle2 = 180;



Servo servo1;
Servo servo2;
Servo servo3;
Servo servo4;
Servo servo5;
Servo servo6;

//テスト用
char val;
```

```
void setup() {
  Serial.begin(9600);
  servo1.attach(ServoPin1, 500, 2400);
  servo2.attach(ServoPin2, 500, 2400);
  servo3.attach(ServoPin3, 500, 2400);
  servo4.attach(ServoPin4, 500, 2400);
  servo5.attach(ServoPin5, 500, 2400);
  servo6.attach(ServoPin6, 500, 2400);
  servo1.write(stopAngle1);
  servo2.write(stopAngle2);
  servo3.write(stopAngle3);
  servo4.write(stopAngle4);
  servo5.write(stopAngle5);
  servo6.write(stopAngle6);
}

void loop() {
  //筋電位の数値をサーボモータと LED の操作のために変換
  value1 = analogRead(VoltReadPin1);
  value2 = analogRead(VoltReadPin2);
  //value3 = analogRead(VoltReadPin3);
  value4 = analogRead(VoltReadPin4);

  //デジタル値を 0 から 1023 を 0.00 から 5.00 に変換
  volt1 = value1 * 5.0 / 1023;
  volt2 = value2 * 5.0 / 1023;
  volt3 = value3 * 5.0 / 1023;
  volt4 = value4 * 5.0 / 1023;

  //サーボモータの回転角と回転時間に変換
  //エフェクター用サーボ
  if (1023 * 0 / 6 <= value1 && value1 < 1023 * 3 / 6) {
    afterAngle1 = lotateAngle1 * 0 / 5;
  } else if (1023 * 3 / 6 <= value1 && value1 < 1023 * 4 / 6) {
    afterAngle1 = lotateAngle1 * 3 / 5;
  } else if (1023 * 4 / 6 <= value1 && value1 < 1023 * 6 / 6) {
    afterAngle1 = lotateAngle1 * 5 / 5;
  }
  lotateTime1 = (afterAngle1 - beforeAngle1) * (300.0 / 180);
  /*
  if (1023 * 0 / 6 <= value2 && value2 < 1023 * 1 / 6) {
```

```
    afterAngle2 = lotateAngle1 * 0 / 5;
  } else if (1023 * 1 / 6 <= value2 && value2 < 1023 * 2 / 6) {
    afterAngle2 = lotateAngle1 * 1 / 5;
  } else if (1023 * 2 / 6 <= value2 && value2 < 1023 * 3 / 6) {
    afterAngle2 = lotateAngle1 * 2 / 5;
  } else if (1023 * 3 / 6 <= value2 && value2 < 1023 * 4 / 6) {
    afterAngle2 = lotateAngle1 * 3 / 5;
  } else if (1023 * 4 / 6 <= value2 && value2 < 1023 * 5 / 6) {
    afterAngle2 = lotateAngle1 * 4 / 5;
  } else if (1023 * 5 / 6 <= value2 && value2 <= 1023 * 6 / 6) {
    afterAngle2 = lotateAngle1 * 5 / 5;
  }
  lotateTime2 = (afterAngle2 - beforeAngle2) * (300.0 / 180);
*/

  if (1023 * 0 / 6 <= value2 && value2 < 1023 * 3 / 6) {
    afterAngle2 = lotateAngle1 * 0 / 5;
  } else if (1023 * 3 / 6 <= value2 && value2 < 1023 * 4 / 6) {
    afterAngle2 = lotateAngle1 * 3 / 5;
  } else if (1023 * 4 / 6 <= value2 && value2 < 1023 * 6 / 6) {
    afterAngle2 = lotateAngle1 * 5 / 5;
  }
  lotateTime2 = (afterAngle2 - beforeAngle2) * (300.0 / 180);

  /*
  if (1023 * 0 / 6 <= value3 && value3 < 1023 * 1 / 6) {
    afterAngle3 = lotateAngle1 * 0 / 5;
  } else if (1023 * 1 / 6 <= value3 && value3 < 1023 * 2 / 6) {
    afterAngle3 = lotateAngle1 * 1 / 5;
  } else if (1023 * 2 / 6 <= value3 && value3 < 1023 * 3 / 6) {
    afterAngle3 = lotateAngle1 * 2 / 5;
  } else if (1023 * 3 / 6 <= value3 && value3 < 1023 * 4 / 6) {
    afterAngle3 = lotateAngle1 * 3 / 5;
  } else if (1023 * 4 / 6 <= value3 && value3 < 1023 * 5 / 6) {
    afterAngle3 = lotateAngle1 * 4 / 5;
  } else if (1023 * 5 / 6 <= value3 && value3 <= 1023 * 6 / 6) {
    afterAngle3 = lotateAngle1 * 5 / 5;
  }
  lotateTime3 = (afterAngle3 - beforeAngle3) * (290.0 / 180);
*/
  if (1023 * 0 / 6 <= value4 && value4 < 1023 * 1 / 6) {
    afterAngle4 = lotateAngle1 * 0 / 5;
```

```
  } else if (1023 * 1 / 6 <= value4 && value4 < 1023 * 2 / 6) {
    afterAngle4 = lotateAngle1 * 1 / 5;
  } else if (1023 * 2 / 6 <= value4 && value4 < 1023 * 3 / 6) {
    afterAngle4 = lotateAngle1 * 2 / 5;
  } else if (1023 * 3 / 6 <= value4 && value4 < 1023 * 4 / 6) {
    afterAngle4 = lotateAngle1 * 3 / 5;
  } else if (1023 * 4 / 6 <= value4 && value4 < 1023 * 5 / 6) {
    afterAngle4 = lotateAngle1 * 4 / 5;
  } else if (1023 * 5 / 6 <= value4 && value4 <= 1023 * 6 / 6) {
    afterAngle4 = lotateAngle1 * 5 / 5;
  }
  lotateTime4 = (afterAngle4 - beforeAngle4) * (290.0 / 180);

  //装飾用サーボ
  if (1023 * 0 / 6 <= (value1 + value2) / 2 && (value1 + value2) / 2 < 1023 * 1 / 6) {
    afterAngle5 = lotateAngle2 * 0 / 5;
  } else if (1023 * 1 / 6 <= (value1 + value2) / 2 && (value1 + value2) / 2 < 1023 * 2 / 6)
    afterAngle5 = lotateAngle2 * 1 / 5;
  } else if (1023 * 2 / 6 <= (value1 + value2) / 2 && (value1 + value2) / 2 < 1023 * 3 / 6)
    afterAngle5 = lotateAngle2 * 2 / 5;
  } else if (1023 * 3 / 6 <= (value1 + value2) / 2 && (value1 + value2) / 2 < 1023 * 4 / 6)
    afterAngle5 = lotateAngle2 * 3 / 5;
  } else if (1023 * 4 / 6 <= (value1 + value2) / 2 && (value1 + value2) / 2 < 1023 * 5 / 6)
    afterAngle5 = lotateAngle2 * 4 / 5;
  } else if (1023 * 5 / 6 <= (value1 + value2) / 2 && (value1 + value2) / 2 <= 1023 * 6 / 6)
    afterAngle5 = lotateAngle2 * 5 / 5;
  }
  lotateTime5 = (afterAngle5 - beforeAngle5) * (300.0 / 180);

  if (1023 * 0 / 6 <= (value3 + value4) / 2 && (value3 + value4) / 2 < 1023 * 1 / 6) {
    afterAngle6 = lotateAngle2 * 0 / 5;
  } else if (1023 * 1 / 6 <= (value3 + value4) / 2 && (value3 + value4) / 2 < 1023 * 2 / 6)
    afterAngle6 = lotateAngle2 * 1 / 5;
  } else if (1023 * 2 / 6 <= (value3 + value4) / 2 && (value3 + value4) / 2 < 1023 * 3 / 6)
    afterAngle6 = lotateAngle2 * 2 / 5;
  } else if (1023 * 3 / 6 <= (value3 + value4) / 2 && (value3 + value4) / 2 < 1023 * 4 / 6)
    afterAngle6 = lotateAngle2 * 3 / 5;
  } else if (1023 * 4 / 6 <= (value3 + value4) / 2 && (value3 + value4) / 2 < 1023 * 5 / 6)
    afterAngle6 = lotateAngle2 * 4 / 5;
  } else if (1023 * 5 / 6 <= (value3 + value4) / 2 && (value3 + value4) / 2 <= 1023 * 6 / 6)
    afterAngle6 = lotateAngle2 * 5 / 5;
  }
```

```
  lotateTime6 = (afterAngle6 - beforeAngle6) * (300.0 / 180);

  /*
  //テスト用
  if (Serial.available() > 0) {
    val = Serial.read();
    if (val == '1') {
      afterAngle1 = 0;
    } else if (val == '2') {
      afterAngle1 = 270;
    } else if (val == '3') {
      afterAngle2 = 0;
    } else if (val == '4') {
      afterAngle2 = 270;
    } else if (val == '5') {
      afterAngle5 = 0;
    } else if (val == '6') {
      afterAngle5 = 270;
    } else if (val == '7') {
      afterAngle6 = 0;
    } else if (val == '8') {
      afterAngle6 = 270;
    } else if (val == 'a') {
      //紫
      k1 = 0;
      k2 = LightLimit * 1 / 2;
    } else if (val == 'b') {
      //青
      k1 = 0;
      k2 = LightLimit;
    } else if (val == 'c') {
      //青緑
      k1 = LightLimit * 1 / 2;
      k2 = LightLimit * 1 / 2;
    } else if (val == 'd') {
      //緑
      k1 = LightLimit;
      k2 = 0;
    } else if (val == 'e') {
      //橙
      k1 = LightLimit * 1 / 2;
      k2 = 0;
```

```
    } else if (val == 'f') {
      //赤
      k1 = 0;
      k2 = 0;
    }
  }

  //afterAngke の後に lotateTime がないと動かない
  lotateTime1 = (afterAngle1 - beforeAngle1) * (320.0 / 180);
  lotateTime2 = (afterAngle2 - beforeAngle2) * (320.0 / 180);
  lotateTime3 = (afterAngle3 - beforeAngle3) * (320.0 / 180);
  lotateTime4 = (afterAngle4 - beforeAngle4) * (320.0 / 180);
  lotateTime5 = (afterAngle5 - beforeAngle5) * (320.0 / 180);
  lotateTime6 = (afterAngle6 - beforeAngle6) * (320.0 / 180);
*/

  //サーボモータの制御
  //エフェクター用
  if (lotateTime1 > 0) {
    servo1.write(180);  //エフェクターの正方向に回転
    delay(lotateTime1);
  } else if (lotateTime1 < 0) {
    lotateTime1 = lotateTime1 * -1;
    servo1.write(0);  //エフェクターの負方向に回転
    delay(lotateTime1);
  }
  servo1.write(stopAngle1);
  beforeAngle1 = afterAngle1;

  if (lotateTime2 > 0) {
    servo2.write(180);  //エフェクターの正方向に回転
    delay(lotateTime2);
  } else if (lotateTime2 < 0) {
    lotateTime2 = lotateTime2 * -1;
    servo2.write(0);  //エフェクターの負方向に回転
    delay(lotateTime2);
  }
  servo2.write(stopAngle2);
  beforeAngle2 = afterAngle2;

  if (lotateTime3 > 0) {
    servo3.write(180);  //エフェクターの正方向に回転
```

```
    delay(lotateTime3);
  } else if (lotateTime3 < 0) {
    lotateTime3 = lotateTime3 * -1;
    servo3.write(0);  //エフェクターの負方向に回転
    delay(lotateTime3);
  }
  servo3.write(stopAngle3);
  beforeAngle3 = afterAngle3;

  if (lotateTime4 > 0) {
    servo4.write(180);  //エフェクターの正方向に回転
    delay(lotateTime4);
  } else if (lotateTime4 < 0) {
    lotateTime4 = lotateTime4 * -1;
    servo4.write(0);  //エフェクターの負方向に回転
    delay(lotateTime4);
  }
  servo4.write(stopAngle4);
  beforeAngle4 = afterAngle4;

  //装飾用
  if (lotateTime5 > 0) {
    servo5.write(0);  //時計周りに回転
    delay(lotateTime5);
  } else if (lotateTime5 < 0) {
    lotateTime5 = lotateTime5 * -1;
    servo5.write(180);  //反時計周りに回転
    delay(lotateTime5);
  }
  servo5.write(stopAngle5);
  beforeAngle5 = afterAngle5;

  if (lotateTime6 > 0) {
    servo6.write(180);  //反時計周りに回転
    delay(lotateTime6);
  } else if (lotateTime6 < 0) {
    lotateTime6 = lotateTime6 * -1;
    servo6.write(0);  //時計周りに回転
    delay(lotateTime6);
  }
  servo6.write(stopAngle6);
  beforeAngle6 = afterAngle6;
```

```
  /*
  //テスト用
  Serial.print("volt1:");
  Serial.println(volt1);
  Serial.print("volt2:");
  Serial.println(volt2);
  Serial.print("volt3:");
  Serial.println(volt3);
  Serial.print("volt4:");
  Serial.println(volt4);
  Serial.print("valueMax3:");
  Serial.println(valueMax3);
  Serial.print("afterAngle1:");
  Serial.println(afterAngle1);
  Serial.print("beforeAngle1:");
  Serial.println(beforeAngle1);
  Serial.print("lotateTime1:");
  Serial.println(lotateTime1);

    Serial.print("afterAngle:");
  Serial.println(afterAngle4);
  Serial.print("beforeAngle:");
  Serial.println(beforeAngle4);
  Serial.print("lotateTime:");
  Serial.println(lotateTime4);
  Serial.println("------------------------------");
*/
  delay(300);
}
```

A.2 LED 制御

```
#include <Servo.h>
#include <Adafruit_NeoPixel.h>

/*
#筋電位関係の変数
VoltReadPin：アナログピンの番号
value：電圧を 0 から 1023 のデジタル値として扱う
*/
```

```
int VoltReadPin1 = 1;
int value1 = 0;
int VoltReadPin2 = 2;
int value2 = 0;
int VoltReadPin3 = 3;
int value3 = 0;
int VoltReadPin4 = 4;
int value4 = 0;

int valueMax1 = 0;
int valueMax2 = 0;
int valueMax3 = 0;

/*
#LED 関係の変数
LED_PIN：デジタルピンの番号
LED_COUNT：光らせる LED の個数
LightLimit：明るさの最大値
g,b：筋電によって RGB 値を制御するための変数
s：全 LED の等分数
*/
int LED_PIN = 8;
int LED_COUNT = 10;
int LED_PIN2 = 9;
int LED_COUNT2 = 20;
int LightLimit = 200;
int g = 0;
int b = 0;
int s = 1;

Adafruit_NeoPixel led = Adafruit_NeoPixel(LED_COUNT, LED_PIN, NEO_GRB + NEO_KHZ800);
Adafruit_NeoPixel led2 = Adafruit_NeoPixel(LED_COUNT2, LED_PIN2, NEO_GRB + NEO_KHZ800);

//テスト用
char val;

void setup() {
  Serial.begin(9600);
  led.begin();
  led2.begin();
}
```

```
void loop() {
  //筋電位の数値をサーボモータと LED の操作のために変換
  value1 = analogRead(VoltReadPin1);
  value2 = analogRead(VoltReadPin2);
  value3 = analogRead(VoltReadPin3);
  value4 = analogRead(VoltReadPin4);

  //電圧の最大値の判定
  valueMax1 = max(value1, value2);
  valueMax2 = max(value3, value4);
  valueMax3 = max(valueMax1, valueMax2);

  //色の決定
  if (1023 * 0 / 5 <= valueMax3 && valueMax3 < 1023 * 1 / 5) {
    //紫から青
    g = 0;
    b = (LightLimit * 1 / 2) + (LightLimit * 1 / 2) * ((valueMax3 - 1023.0 * 0 / 5) / (1023
  } else if (1023 * 1 / 5 <= valueMax3 && valueMax3 < 1023 * 2 / 5) {
    //青から青緑
    g = 0 + (LightLimit * 1 / 2) * ((valueMax3 - 1023.0 * 1 / 5) / (1023.0 * 1 / 5));
    b = LightLimit - (LightLimit * 1 / 2) * ((valueMax3 - 1023.0 * 1 / 5) / (1023.0 * 1 / 5)
  } else if (1023 * 2 / 5 <= valueMax3 && valueMax3 < 1023 * 3 / 5) {
    //青緑から緑
    g = (LightLimit * 1 / 2) + (LightLimit * 1 / 2) * ((valueMax3 - 1023.0 * 2 / 5) / (1023
    b = (LightLimit * 1 / 2) - (LightLimit * 1 / 2) * ((valueMax3 - 1023.0 * 2 / 5) / (1023
  } else if (1023 * 3 / 5 <= valueMax3 && valueMax3 < 1023 * 4 / 5) {
    //緑から橙
    g = LightLimit - (LightLimit * 1 / 2) * ((valueMax3 - 1023.0 * 3 / 5) / (1023.0 * 1 / 5)
    b = 0;
  } else if (1023 * 4 / 5 <= valueMax3 && valueMax3 <= 1023 * 5 / 5) {
    //橙から赤
    g = (LightLimit * 1 / 2) - (LightLimit * 1 / 2) * ((valueMax3 - 1023.0 * 4 / 5) / (1023
    b = 0;
  }


  //LED の制御
  for (int i = 0; i < LED_COUNT / s; i++) {
    for (int j = 0; j < s; j++) {
      led.setPixelColor(i + j * LED_COUNT / s, led.Color(LightLimit - g - b, g, b));  // 赤、
緑、青の値を設定
      led.show();
```

```
    }
  }
  for (int i = 0; i < LED_COUNT2 / s; i++) {
    for (int j = 0; j < s; j++) {
      led2.setPixelColor(i + j * LED_COUNT2 / s, led2.Color(LightLimit - g - b, g, b));  // 赤
緑、青の値を設定
      led2.show();
    }
  }


  delay(300);
}
```

参考文献

[1] Senju, Mariko, and Kengo, Ohgushi. How Are the Player’s Ideas Conveyed to the Audience?. Music Perception: An Interdisciplinary Journal, 1987, Vol.4, No.4, pp.311–323. https://www.jstor.org/stable/40285377

[2] 三戸勇気. モーションキャプチャを用いた演奏動作計測. 日本音響学会誌，2021，77(9), pp.580-586. https://doi.org/10.20697/jasj.77.9_580

[3] 野村恵造，花本金吾，林龍次郎．オーレックス英和辞典　第 2 版新装版．株式会社旺文社，2016.

[4] M. チクセントミハイ，大森弘（訳）: フロー体験入門．世界思想社，2010，pp.42,45.

[5] 藤沢望．小特集「音楽制作を彩る音づくりの技術”エフェクタ”」にあたって (＜小特集＞音楽制作を彩る音づくりの技術”エフェクタ”)．日本音響学会誌，2012，68(7)，pp.343 － 344. https://doi.org/10.20697/jasj.68.7_343

[6] 星野喜久三．色とテンポの関係に関する研究．北海道學藝大學紀要. 第一部，1959，10(1), pp.233-239. https://doi.org/10.32150/00000602

[7] 暦本純一．人間拡張が築く未来．東京大学大学院情報学環紀要 情報学研究，2021，(100), pp.19-45. https://www.iii.u-tokyo.ac.jp/about/bulletin/journal

[8] 上田祥代．身体拡張に関わるメカニズムの検討－受動的／能動的刺激入力による差異、および、ミラーシステムとの関連性－．人間文化創成科学論叢 ，2011，14，pp.217-226. https://teapot.lib.ocha.ac.jp/record/39410/files/23_217-226.pdf

[9] 三浦寛也，浜中雅俊．Deep Augmented Performers: メロディモーフィングと身体機能の融合によるアンサンブル演奏システム．研究報告音楽情報科学（MUS），2021，132(4)，pp.1-8. http://id.nii.ac.jp/1001/00212722/

[10] Jan-Peter, Herbst. Empirical Explorations of Guitar Players Attitudes Towards Their Equipment and the Role of Distortion in Rock Music. Current Musicology,2020, (105). https://doi.org/10.7916/cm.v0i105.5404