

テーマ名：ソースコード編集ツールとしての音楽制作ソフトウェアの設計

申請者名：松浦知也

## 【提案テーマ詳細説明】

### なにをつくるか

本企画では、近年の音楽プログラミングを用いた表現の動向や、プログラミングを用いた生成的表現の支援ツールの発展などを背景にして、楽譜としてのソースコードをテキストとGUIで双方向に編集可能な音楽制作ソフトウェア、またそのための音楽プログラミング言語の設計を行う。

Avid社のProtoolsやSteinberg社のCubaseに代表されるDAWソフトウェアは歴史を振り返るとマルチトラック・レコーダーやミキサー、シーケンサーといったハードウェア機器を仮想的にソフトウェアで再現した構造を持つものが多い。とりわけ特徴的な機能として捉えられるのが、仮想的なエフェクターやシンセサイザーをサードパーティ製のものも含め使用できるプラグイン機能と、その中の個別のパラメータを時間軸に沿って変化させる「オートメーション」と呼ばれる機能である。現代の音楽制作においてオートメーションカーブはMIDIのような音価にも値する、楽曲を左右するものになりつつあり、また時間軸上で変化するパラメーターという意味では本質的には録音やシンセサイザーで生成される音の波形とソフトウェア内では同等のものとも言える。スマートフォンにおけるインターフェースが実物の模倣からフラットデザインに変化したように、メタファを超えたインターフェース設計が求められている。

他方、音楽を扱うプログラミング言語の発展を概観すると、オーディオ処理とイベント処理という時間スケールの異なる処理を境目なく記述できるChuck(<http://chuck.stanford.edu/>)やすべての処理を外部ライブラリに頼ること無く一つの言語で完結できるFaust(<https://faust.grame.fr/>)、その2つをどちらも実現しているExtempore(<https://extemporelang.github.io/>)などがこれまで開発されている。またこれらの言語は近年のコンピューターの処理能力向上もあいまって比較的複雑な処理も高速にコンパイルができる他、ライブコーディングというリアルタイムに処理を書き換え即時反映できる機能もある。こうした音楽プログラミング言語はこれまで主に実験的な音楽の文脈の中だけで用いられてきたが、近年ではAlex McleanがはじめたAlgoraveというライブコーディングでダンスミュージックを演奏するイベントが世界各地で行われるようになってきている[1](筆者も一度出演し、ChuckとFaustを用いて演奏した経験がある)。その中ではMcleanがAlgoraveのために開発したTidalCycles(<https://tidalcycles.org/>)のほか、FoxDot(<http://foxdot.org/>)やSonic Pi(<https://sonic-pi.net/>)など、親しみやすい音楽をリアルタイムに生成することを射程に入れた言語も登場している。

こうした背景のなかで筆者は一般的なDAWと音楽プログラミング環境の中間的位置づけの音楽制作ソフトウェアの必要性を感じている。自身の音楽制作の経験からも、近年はプログラミングで生成した波形サンプルやMIDIパターンをDAWに取り込み後から編集するような作業工程が増えており、1つのソフトウェアで制作が完結することは殆ど無い。

そこで本企画では、DAWと音楽プログラミング環境の中間的位置づけとしてリアルタイム実行可能な音楽プログラミング言語と、そのソースコードの編集ツールとしてDAWの様なグラフィカル・ユーザーインターフェース ( GUI ) を持った音楽制作ソフトウェアの設計を行う。

これは視覚表現の分野ではSVG画像のソースをテキスト編集とダイレクト・マニピュレーション(マウスによる直接編集)を行うRavi ChughらによるSketch-n-Sketch[2]や、生成的表現のダイレクト・マニピュレーションを実現するJenifer JacobsのPara[3]といったソフトウェアがあり、Adobe Illustratorの様なベクター画像編集ソフトと、Processing(<https://processing.org/>)のような画像を生成するプログラミングの中間的位置づけと捉えることができる。

本企画で提案するソフトウェアはこうした視覚表現での先例を音表現の分野で実行するものである。

具体的には、

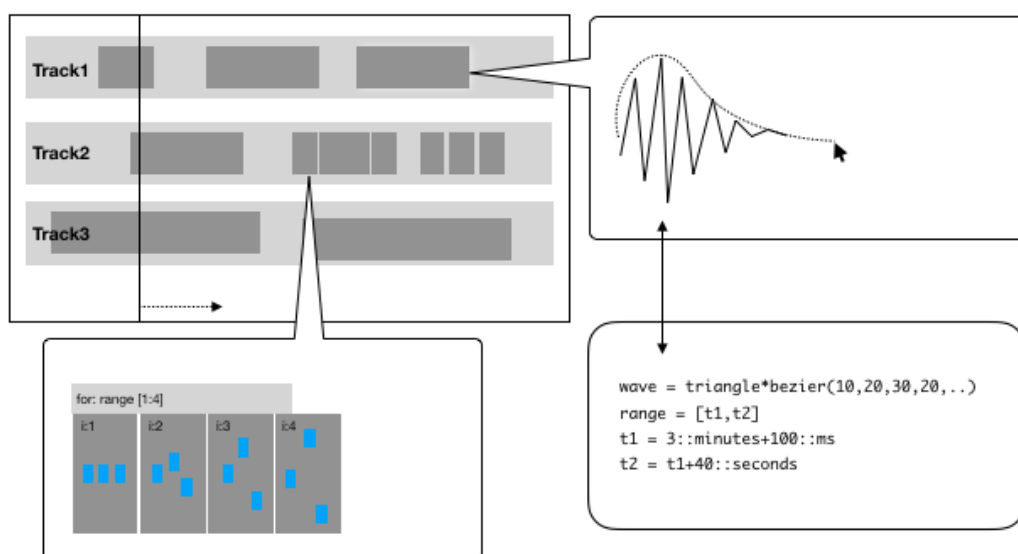
1.音楽プログラミング言語とその実行環境

2.その言語をグラフィカルに編集するソフトウェアの開発

の2つを行う。

前者はFaustのように全てのオーディオ処理を他言語に頼ること無く実現でき、ChuckKのようにごく短い時間から長い時間を垣根なく記述でき、また冗長性の排除よりもテキストとしての可読性を高め、ソースコードが楽譜のようにも機能することを目指す。

後者は一般的なDAWのように複数のトラックが存在し、その中に時間範囲を指定してソースコードを配置、編集できるようにする。ソースコードの編集にはマウスを用いた波形生成、アルゴリズムミックなシーケンス生成をテキスト入力/数種類のGUIで生成支援するGUIを用いて行う。そして最終的に音声ファイルやMIDIパターンとして書き出しする機能を持つ物を目指す。



## どんな出し方を考えているか

オープンソースのソフトウェアとして出すことを検討している。またソフトウェアのデザインは提案者の博士研究の一環として行われ、New Interfaces for Musical ExpressionやUIST、CHIといった国際学会やComputer Music Journalといった論文誌での発表を目指す。

## 斬新さの主張、期待される効果など

本ソフトウェアが、プログラミングを用いて行う音表現と大衆音楽的な音表現の橋渡しになることを期待している。

既存のDAWは既存の音楽の構造に大きく依拠しているうえにユーザー自身がカスタマイズ/ハックすることがほとんど出来ないことからそこから生まれる表現自体も既存の音楽の再生産にならざるを得ない現状がある。

例外的に Ableton Live(<https://www.ableton.com/ja/live/>) は音楽プログラミング環境Max(<https://cycling74.com/>)を内部で用いるMax for Liveという機能を持つてはいるが、上位版のみで使えることからあくまで拡張機能という位置づけである。本ソフトウェアは予めプログラミング的表現を行う事を前提として全てを設計することが大きく異なる。

いずれは音楽制作をするのに与えられた/買ったツールを使うのではなく、自身でカスタマイズ/ハックした音楽制作を行う用になることこそがコンピューターを用いた表現のあるべき姿だと信じている。そうした意味ではArdour(<http://www.ardour.org>)やRadium(<http://users.notam02.no/~kjetism/radium/>)といった既存のオープンソースの音楽ソフトウェアは、オープンソースではあるもののデベロッパーとユーザーを分ける構造になっており、発展性には欠けると言える。

本ソフトウェアが言語の設計と操作ソフトウェアという二つのレイヤーに分けて開発をするのは、同じ言語を用いてソフトウェア自体は自分用に異なるバージョンを制作するような発展性を考慮に入れてのことである。

またもう一つの観点として音楽アーカイブの視点がある。DAWは現代の、特に商業的な音楽制作の現場で広く普及しており、そのプロジェクトファイルは曲に対して録音した音ファイルの情報や、細かなオーディオ処理などのパラメーターが全て保存されていることから、音楽学の分野で楽譜を参照して研究を進めるように将来の考古学的資料的位置づけになりうる事は大いに考えられる。実際に渡辺はリュック・フェラーリという電子音楽の作曲家の作品分析をProtoolsのプロジェクトファイルを元に、オートメーションカーブを見ながら分析するという試みをしている[4]。

しかし、そのプロジェクトファイルのフォーマットは基本的に各ソフトウェアごとに互換性がなく、

また殆どがバイナリファイルであり可読性はない。またソフトウェアの機能の増加とともにこうしたファイルフォーマットの仕様が更新されることもある。(ptf→.ptxなど)後方互換性が保たれてアップデートされることが普通ではあるが、使用されなくなることでそのプロジェクトで具体的に何が行われていたか分からなくなる可能性がある。またプロジェクトファイルそのものが開けても、プロジェクト内で使用されていたプラグインがOSやDAWソフトウェア、またはプラグイン自体の更新に伴って開けなくなることも予測される。提案者はこういった歴史的資料としてのプロジェクトファイルの損失という自体を大きく危惧している。

そうした中で、コンピューターを用いて音を生成するという環境において音楽プログラミング言語のソースコードは現代の楽譜にも相当するものになりうる。

本ソフトウェアはDAWを置き換えることを目的とするわけではないが、楽譜としてのソースコードの編集ツールとしての音楽制作ソフトウェアは、今後の音楽文化をより作家がクリエイティビティを発揮でき、後世にその創作のプロセスが引き継がれるものにするための第一歩として大きな意義があると考えられる。

## 具体的な進め方と予算

開発は筆者の主に利用している、また音楽制作者にユーザーの多いmacOSを用いて行う。

開発は主に提案者の自宅と、所属する九州大学大橋キャンパスを中心に行う。

使用言語、ツールについては、

1. プログラミング言語の実行環境についてはC++を使用する。これはオーディオ処理のためにパフォーマンスが求められることと、LLVMを使用することで異なるハードウェアやプラットフォームでの実行が可能になるからである。
2. ソースコードの編集ソフトウェアとしては、フロントエンドをWebブラウザやElectronのようなアプリケーションとして実装し、1.の実行環境と内部的に通信を行う形を検討している。これはインターフェースのプロトタイピングと検証をなるべく早く手軽に行えることが重要だと考えているからであり、もしパフォーマンス的な問題がある場合はその後JUICEやQtといったC++で用いられるGUIライブラリに移行する可能性もある。

開発時間には1440時間を提案する。

スケジュールとしては始めに言語仕様の策定と、筆者がまだ経験のないプログラミング言語自体の設計のための技術的な調査を行うと同時に、ソースコード編集ツールの方は早いうちにユーザーインターフェースの可能性としてどんなことができるかのプロトタイプを多く作成する。

時間に余裕のできる8月から言語、ソースコード編集ツールともに本格的な実装を並行して進める（お互いの仕様が互いに依存するため、本格的な開発は同時に行う必要があると考えている。）

11月末をめどにβバージョンの公開を行い、広く意見を集めたい。フィードバックをもとにして改め

て改善を行う。

2020年1月末には国際学会New Interfaces for Musical Expressionの投稿があるので、それに向けて論文執筆も行う。

基本的に1日8時間稼働する計算で、

6月24日~8月31日 で計51日

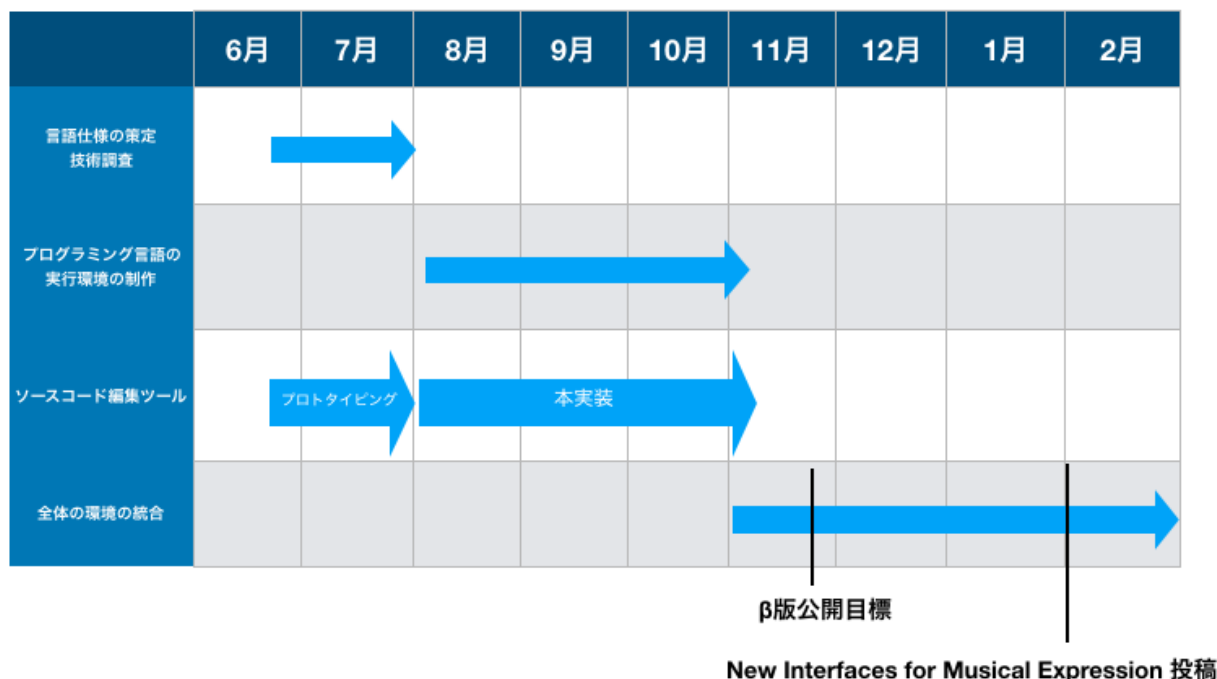
9月は現在既に決まっている予定があるため18日、

10、11、12月で計66日間 23+21+22

1月はNew Interfaces for Musical Expressionの投稿のための論文執筆があるので18日

2月~3月頭で27日間

稼働し、180日\*8=1440時間を開発に費やす。



## 提案者（たち）の腕前を証明できるもの

開発者は2015年から2017年に掛けて株式会社チームラボでサウンドエンジニア/プログラマとして、インスタレーション展示のためのMaxを用いたインタラクティブミュージックのプログラミングを多数手がけた。

また2017年には日本ではじめて音声処理プログラミング言語Faustのワークショップを行ったり、2018年8月にはChuckK言語とFaust言語を用いたAlgorave Tokyoでのライブパフォーマンスを行うなど、一つの言語に頼らず複数の音楽プログラミング環境についての実際の仕様に基いた知見を持つ。

2018年にはNYにあるSchool for Poetic Computationでopenframeworks(C++製のクリエイティブ・コーディング向けライブラリ)を中心にプログラミングを用いた表現について学んだほか、C++に変

換されるFaustをOpenframeworksと連携するためのAddonの製作にも取り組んでいる ( <https://github.com/tomoyanonymousofxFaust> )。

ほか、Webのプログラミングに関しては自信のWebサイトや大学の年次発表展Webサイト製作を通じて独習し、近年ではRiot.jsなどのモダンなライブラリを用いたWebアプリケーション形式での実装も行っている(<http://onkansotsuten.com/2017/>)。

## プロジェクト遂行にあたっての特記事項

本ソフトウェアの制作は提案者の博士後期課程(進学予定)の研究の一環として行われる予定である。指導教員になる九州大学大学院芸術工学府、城一裕先生の承諾は得ている。

## ソフトウェア作成以外の勉強、特技、生活、趣味など

学部で東京藝術大学在籍中から、サウンドアーティストとしてのインスタレーション作品の製作や、ライブパフォーマンスを繰り返し行ってきている。

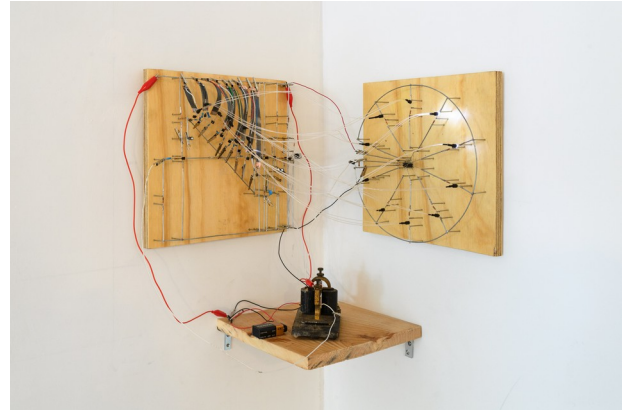
卒業制作のサウンドインスタレーション「送れ|遅れ/post|past」([https://matsuuratomoya.com/works/post-past\\_sotsuten/](https://matsuuratomoya.com/works/post-past_sotsuten/))はアジアデジタルアート大賞2017学生・インスタレーション部門で入賞、第23回学生CGコンテストで入賞した(以下写真)。音響遅延戦メモリというコンピューター黎明期に使われていた音波を用いるメモリ装置をスピーカーとマイクロフォンを用いて再構成し、今日におけるデジタルデータの存在を問い直すものである。



近年ではオーディオフィードバックを用いた電子楽器Exidiophone(<https://matsuuratomoya.com/works/exidiophone/>)の開発を行っており、それを用いたパフォーマンスを年間10回以上行ったり、ポップ・アンサンブルの蓮沼執太フルフィルでの演奏も行っている(下記写真左)。また、2018年秋にはNYのアーティスト・ラン・スクールであるSchool for Poetic Computationで留学し、音を出すためのマスタークロックを持たないコンピューターとしての回路彫刻作品「Electronic Delay Time Automatic Calculator」(<https://matsuuratomoya.com/works/edtac/>)の製作などを行っている(下記写真

右)。

こうした製作活動は音を今日コンピューターで生成する意義についての考察からスタートしているものであり、今回のソフトウェアの製作もこうした活動の中で感じたことが起点になっている。



## 将来のソフトウェア技術について思うこと・期すること

アーティストとしてコンピューターを用いた表現を探求する中で感じることは、巨人の肩に乗ることと同じくらいに巨人の肩を一度降りてみる事の大切さである。

特にプログラミングについてはコンピューターを利用する事を単なるデベロッパーとユーザーという2つに分けてしまうのではなく自らがデベロッパーになることでこれまで存在しなかったものが作れるようになる重要性がある。

しかし現在（特に表現に関わる分野であれば）において、例えば2019年現在はVRや機械学習といったいわゆる最新技術を追いかけることは、仕組みのわからないブラックボックスのライブラリをただ使用するだけの、再びデベロッパー/ユーザー、提供する側/享受する側の構造になってしまうようにも見える。

(ただこれは、人が作ったものを手軽に再利用できるからこそ自分ひとりでは作れない規模のものが作れるということとトレードオフの関係でもある。)

新しいソフトウェアを作るために必ずしも最新の何かが必要なわけではないし、枯れた技術が新しい価値を生み出すことも大いに有り得ると思う。

ソフトウェアを作る行為は自由を獲得する行為と同時に新たな制約を生み出すことでもあり、広義の政治的行為であるということを、今後は技術者も自覚を持っていく必要が出てくると考えている。

## 参考文献

[1] Collins, N. and A. McLean (2014). Algorave: A survey of the history, aesthetics and technology of live performance of algorithmic electronic dance music. In Proceedings of

the International Conference on New Interfaces for Musical Expression, London, United Kingdom, pp. 355–358. Goldsmiths, University of London.

[2] Brian Hempel and Ravi Chugh(2016). [Semi-Automated SVG Programming via Direct Manipulation \(UIST 2016\)](#)

[3]Jennifer Jacobs(2017),Dynamic Drawing: Broadening Practice and Participation in Procedural Art.PhD Thesis, Media Arts and Sciences, Massachusetts Institute of Technology, Cambridge, MA.

[4]渡辺愛(2015).リュック・フェラーリの電子音響作品における逸話の構造,博士論文,東京藝術大学