

ピアノ演奏に連動したリアルタイム映像制御システムの構築

福知山公立大学 情報学部情報学科

32245100 矢野 瑞季

指導教員 橋田光代 准教授

提出日 2026年1月30日

— 目次 ——

| | | | | | |
|-------|--------------------------------|---|-------|-----------------------|---|
| 1 | はじめに | 1 | 12 | 完成映像 | 7 |
| 2 | 関連研究 | 1 | 13 | USB フットペダルの配置イメージ | 7 |
| 3 | システムの概要 | 1 | 14 | 演奏者が使用している楽譜 | 7 |
| 3.1 | システム全体構成 | 1 | 15 | フットペダルの役割（「春よ、来い」の場合） | 7 |
| 3.2 | 使用機材およびソフトウェア | 2 | 16 | 実演風景 | 8 |
| 3.3 | 映像出力およびパフォーマンス環境 | 2 | | | |
| 4 | 画像生成システム | 2 | | | |
| 4.1 | 入力インターフェース | 2 | | | |
| 4.2 | 画像生成処理 | 3 | | | |
| 4.3 | 生成画像の表示および保存 | 3 | | | |
| 5 | TouchDesigner を用いた映像制作 | 3 | | | |
| 5.1 | TouchDesigner と Processingとの比較 | 3 | | | |
| 5.2 | 制作した映像システムの全体構成 | 4 | 5.2.1 | 入力処理部 | |
| 5.2.2 | 映像生成部 | | 5.2.2 | 映像合成部 | |
| 5.2.3 | 映像合成・出力部 | | 5.2.3 | 映像合成・出力部 | |
| 5.3 | 生成画像・歌詞データの利用方法 | 6 | 5.3.1 | 生成画像の表示 | |
| 5.3.2 | 歌詞の表示 | | 5.3.2 | 歌詞の表示 | |
| 5.4 | 映像の最終合成および出力 | 6 | | | |
| 6 | 足キーボードを用いた映像制御について | 7 | | | |
| 7 | 実演について | 8 | 7.1 | 公開実演 | 8 |
| 7.1 | 研究室内デモ発表およびテスト演奏 | 8 | 7.2 | 評価結果と考察 | 8 |
| 7.2 | | | 7.3 | | 8 |
| 8 | 課題と今後の発展について | 8 | | | |
| 9 | まとめ | 9 | | | |

— 図目次 ——

| | | |
|----|-------------------------|---|
| 1 | システムの概要図 | 2 |
| 2 | 歌詞・イメージ入力インターフェース | 2 |
| 3 | 指定フォルダに保存 | 3 |
| 4 | テキストファイル内 | 3 |
| 5 | TouchDesigner 内のシステム構成図 | 4 |
| 6 | 打鍵によって光る鍵盤 | 5 |
| 7 | 鍵盤から出る光・炎エフェクト | 5 |
| 8 | 桜の花びらが散るエフェクト | 5 |
| 9 | 煙エフェクト | 6 |
| 10 | 元の画像と加工後の画像 | 6 |
| 11 | 歌詞の表示 | 6 |

1. はじめに

近年、デジタル技術の発展により、音楽と映像を組み合わせた表現が多様化している。特に、ライブパフォーマンスや舞台演出において、音楽の進行に応じて映像が変化する演出は、観客に対して強い没入感を与える手法として注目されている。このような音楽と映像の連動表現は、事前に制作された映像を再生するだけでなく、演奏中の音楽的要素をリアルタイムに反映させることが求められている。

しかし、映像制作や映像演出には多くの時間と手間、ならびに専門的な知識が必要であり、音楽演奏と並行して映像を操作することは容易ではない。そのため、映像演出は演奏とは切り離され、別の操作者によって制御される場合が多いのが現状である。また、演奏者自身が演奏中に映像を制御することを前提としたシステムに関する研究や事例は少なく、音楽表現と映像表現を演奏行為の中で統合する手法は研究事例が限られており、現在も発展途上の段階にある。

さらに、演奏中の操作が複雑である場合、演奏そのものに支障をきたす可能性がある。このため、演奏者の負担を最小限に抑え、演奏の流れを妨げることなく直感的に映像を制御できる仕組みが求められている。このような課題は、音楽表現と視覚表現の一体化を実現する上で重要な検討事項である。

そこで本研究では、ピアノ演奏に基づいて映像をリアルタイムに制御・生成するシステムの構築を目的とする。具体的には、MIDI キーボードから取得したノート番号およびベロシティ情報、ならびに USB フットペダルからの入力を用いて、演奏内容に応じた映像表現を実現する。映像の制御および制作には TouchDesigner を用い、演奏中の入力情報を即時に映像パラメータへ反映させることで、演奏と映像が連動した表現を可能とする。

また、映像表現の一要素として生成 AI を用いた視覚表現を取り入れる。歌詞を含む楽曲に対しては、Python を用いて入力した歌詞データをもとに画像生成を行い、楽曲内容に対応した映像素材を生成する。一方、歌詞を持たない楽曲に対しては、曲の雰囲気や印象を表す情報をもとに生成 AI を用いて視覚表現を生成する手法を採用している。

さらに、本システムは生成された画像を静止画として用いるだけでなく、エフェクトなどの動画素材を映像表現として利用することも可能な構成となっている。これにより、楽曲の特性に応じた映像表現を柔軟に選択しながら、演奏者自身が演奏中に映像演出へ直接関与することが可能となる。以上のことから、本研究は、演奏行為の中に映像操作を組み込んだ、新たなライブパフォーマンス表現の可能性を示すものである。

2. 関連研究

近年、音楽と映像を統合したリアルタイム映像制御・生成に関する研究が複数報告されている。これらの研究は、本研究が目指す「演奏中の映像制御」を支える背景技術および表現の広がりを示している。

まず、ライブパフォーマンスにおけるリアルタイム映像生成と視覚体験の関連性に関する研究として、Erdmann らによる mixed reality (複合現実) を用いた音楽ビジュアライゼーションの評価研究が挙げられる。この研究では、リアルタイムの音響特徴量分析に基づき映像を制御することで、視聴者の美的体験が向上することを示した。このような実装例は、音楽情報のリアルタイム処理に基づいた映像制御の有効性を示す重要な先行研究である [1]。

加えて、リアルタイムの音楽ビジュアライゼーションシステムの実装と評価に関する研究も存在する。Graf らは、音響特徴の抽出とその視覚表現へのマッピングを行うシステムを提案しており、ライブ音楽と視覚表現が同期することで、観客の体験値が向上すると報告している。これにより、音響データの抽出・可視化が演奏と映像表現の融合において有効であることが示されている [2]。

国内の研究としては、ライブ映像に映像・音響表現を付与することで音楽体験を向上させる手法が報告されており、一視点から撮影されたライブ映像に対して視覚表現を付加することで、臨場感や没入感が高まる可能性が示されている。これは、映像と音響が統合された表現が音楽体験に好影響を与えることを示しており、本研究における映像制御の有効性を裏付けるものである [3]。

これらの先行研究は、音楽演奏と視覚表現の統合が観客体験や表現価値に寄与する可能性を示していると同時に、リアルタイム性や演奏同期の技術的課題が研究対象となっていることを示している。これらの流れを背景に、演奏者自身が演奏中に映像制御を行うインタラクティブな実装に焦点を当て、従来の手法とは異なる演奏-映像統合のアプローチを提示するものである。

3. システムの概要

3.1 システム全体構成

提案するシステムは、ピアノ演奏および生成 AI によって作成された映像素材を用いて、演奏内容に応じた映像表現をリアルタイムに制御・出力することを目的としている。本システムは、演奏情報および映像素材生成のための入力を受け取り、TouchDesigner 上で映像制御を行い、最終的にスクリーンへ映像を出力する構成となっている。

システム全体の流れは大きく二つに分かれる(図 1)。一つは、Python で構築した歌詞・イメージ入力画面から生成 AI を用いて映像素材を生成し、その結果を TouchDesigner に

取り込む流れである。もう一方は、MIDI キーボードおよび USB フットペダルから取得した演奏情報を TouchDesigner へ入力し、映像演出を制御する流れである。これら二つの情報が統合されることで、多様な映像表現に対応可能なシステムの構築を目指している。

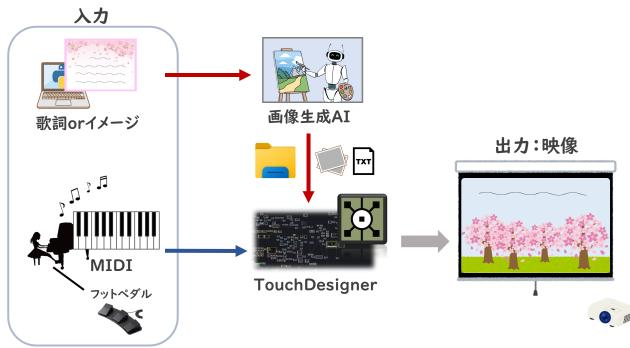


図 1: システムの概要図

3.2 使用機材およびソフトウェア

今回使用する主な機材は、MIDI キーボード、USB フットペダル [4]、および処理を行う PC である。MIDI キーボードはピアノ演奏情報の入力装置として使用し、USB フットペダルは演奏中の映像および歌詞の切り替え操作を行うために用いている。

使用ソフトウェアとしては、映像制御および映像制作に TouchDesigner を使用する。また、歌詞や曲のイメージ情報の入力には Python を用い、映像素材の生成には OpenAI の API を利用している。

3.3 映像出力およびパフォーマンス環境

本システムによって制御された映像は、スクリーンに対して一画面で投影される。ライブパフォーマンスでの使用を想定し、演奏者のピアノ演奏と映像が同期して提示される構成としている。これにより、音楽の進行や演奏表現に応じて視覚的な変化が生じ、観客に対して音楽と映像が一体となった表現を提示することが可能となる。

パフォーマンス中には、演奏者が USB フットペダルを用いて映像や歌詞の切り替えを行う。フットペダルによる操作を採用することで、手を使わずに映像制御を行うことができ、演奏動作への影響を最小限に抑えている。また、演奏者は演奏の流れに合わせて必要なタイミングで映像を切り替えることができるため、ライブパフォーマンスにおける即時性にも対応可能である。

このような構成により、演奏行為を中心としながらも、演奏者自身が映像表現に直接関与できる環境を実現している。結果として、音楽表現と映像表現が相互に影響し合うライブパフォーマンスが可能となり、演奏行為と同期した視覚表現を実現している。

4. 画像生成システム

ピアノ演奏と連動した映像演出に使用する映像素材を生成するため、生成 AI を用いた画像生成システムを構築した。画像システムは Python 上で動作する GUI アプリケーションとして実装されており、OpenAI の画像生成 API および GUI ライブリである Tkinter を用いて構成されている。

ユーザインターフェースでは、最大 20 行までのテキスト入力が可能であり、各行ごとに「歌詞」または「インスト」を選択することができる。歌詞を含む楽曲の場合には歌詞をそのまま入力し、歌詞を持たない楽曲の場合には楽曲の雰囲気やイメージを文章として入力することで、生成 AI(DALL-E 3[5])による画像生成を行う。生成された画像は PNG ファイルとして指定フォルダに保存され、TouchDesigner に取り込むことで映像演出に利用される。

4.1 入力インターフェース

入力インターフェースは、Tkinter を用いて実装されており、左側に入力欄、右側に処理状況を表示するテキストエリアを配置した構成となっている。入力欄は最大 20 行まで用意されており、各行にはテキスト入力欄とともに、「歌詞」または「インスト」を選択するためのプルダウンメニューが設けられている。



図 2: 歌詞・イメージ入力インターフェース

利用者が歌詞を任意のタイミングで分割して入力する運用を想定している。各歌詞フレーズは一つのセクションとして扱われ、入力したフレーズに対応する映像素材は、演奏中に USB フットペダルを操作することで切り替えられる。そのため、歌詞の進行と映像の切り替えを同期させるには、歌詞の入力時に想定したタイミングでフットペダルを踏む必要がある。

本研究では、システムの動作確認および評価のためのテスト曲として松任谷由実の「春よ、来い」[6]を使用し、歌

詞を複数のフレーズに分割して入力した(図 2)。生成された画像および入力された歌詞等の処理については、後の節にて詳細に説明する。また、歌詞を含まない楽曲や曲中インストロや間奏等、歌詞がない部分については、「インスト」を選択し生成したい映像のイメージを文章として入力することも可能である。

4.2 画像生成処理

入力されたテキストのうち、空でない行のみを対象として、生成 AI による画像生成を行う。各行は順番に処理され、OpenAI の画像生成 API を用いて 1 行につき 1 枚の画像を生成する。今回は、高解像度の映像演出を想定し、生成される画像サイズを横 1792 ピクセル、縦 1024 ピクセルに設定している。

生成された画像データは、Base64 形式で受信され、Python 上でデコード処理を行った後、画像ファイルとして保存される。「春よ、来い」では、1 番の歌詞に加えて前奏および間奏を含めた構成を想定し、楽曲全体を 7 つのセクションに分割した。

図 2 に示す画像生成画面において「生成開始」ボタンを押すと、入力された各行に対応する画像が順に生成され、合計 7 枚の画像が作成される仕組みとなっている。また、画像生成処理の進行状況やエラー情報は GUI 上に逐次表示され、ユーザが処理の状態を確認できるようになっている。

4.3 生成画像の表示および保存

画像は 1 枚ずつ PNG ファイルとして指定フォルダに保存され、TouchDesigner 側で画像ファイルとして読み込むことで、映像演出に利用される(図 3)。また、入力されたテキストのうち、「歌詞」として指定された行のみを抽出し、テキストファイルへ保存される機能を実装している(図 4)。このテキストファイルは TouchDesigner 側で読み込まれ、映像演出時に一行ずつ順番に表示されることを想定している。一方で、「インスト」が選択された行についてはテキストファイルには保存されず、空行となり、映像上にも歌詞として表示されない仕様となっている。これにより、歌詞を持たない楽曲に対しても、映像演出を行うことが可能となっている

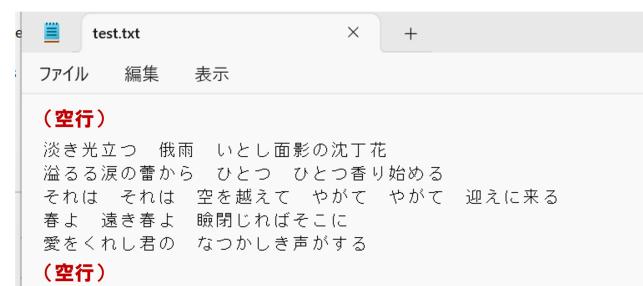


図 4: テキストファイル内

5. TouchDesigner を用いた映像制作

TouchDesigner[7] は、Derivative 社によって開発された、リアルタイム映像処理およびインタラクティブコンテンツ制作のためのビジュアルプログラミング環境である。ノードベースの操作体系を特徴としており、映像、音声、センサ入力、MIDI などの多様なデータを組み合わせて処理・制御することができる。そのため、ライブパフォーマンスやインスタレーション、舞台演出など、リアルタイム性が求められる表現分野で広く利用されている。

TouchDesigner では、映像処理を行う TOP (Texture Operator)、数値や制御信号を扱う CHOP (Channel Operator)、三次元空間での描画を行う SOP (Surface Operator) や COMP (Component) など、複数のオペレータを接続することで処理フローを構築する。これにより、プログラムコードを直接記述することなく、視覚的に処理内容を設計できる点が特徴である。

本研究では、TouchDesigner を映像制御および映像制作の中核として用い、MIDI キーボードや USB フットペダルから取得した演奏情報、ならびに生成 AI によって作成された画像およびテキストデータを統合的に処理する。これにより、ピアノ演奏の内容に応じた映像表現をリアルタイムに制御・出力することを可能としている。

TouchDesigner は初めて使用したツールであり、主に YouTube や公式ドキュメント、参考サイトなどの資料を参照しながらシステム構築を行った。構築過程では試行錯誤を要したもの、ノードの接続によって処理の流れを視覚的に把握できる点は、複雑な映像制御を伴うシステム構築において有効であると判断した。

5.1 TouchDesigner と Processing との比較

当初は、映像制作および演奏運動システムを Processing[8] で構築していた。Processing とは、MIT メディアラボを起源とする Java をベースとしたプログラミング環境であり、簡潔な記述によって図形描画やアニメーション、入力処理を行うことが可能であるため、ビジュアル表現やインタラクティブアートを中心に広く利用されているソフトウェアである。しかし、映像表現が複雑化し、複数の映

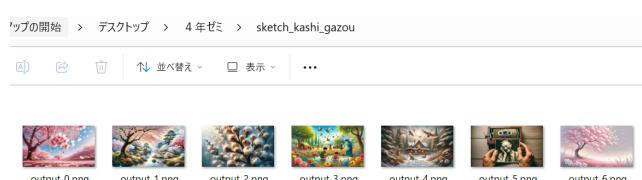


図 3: 指定フォルダに保存

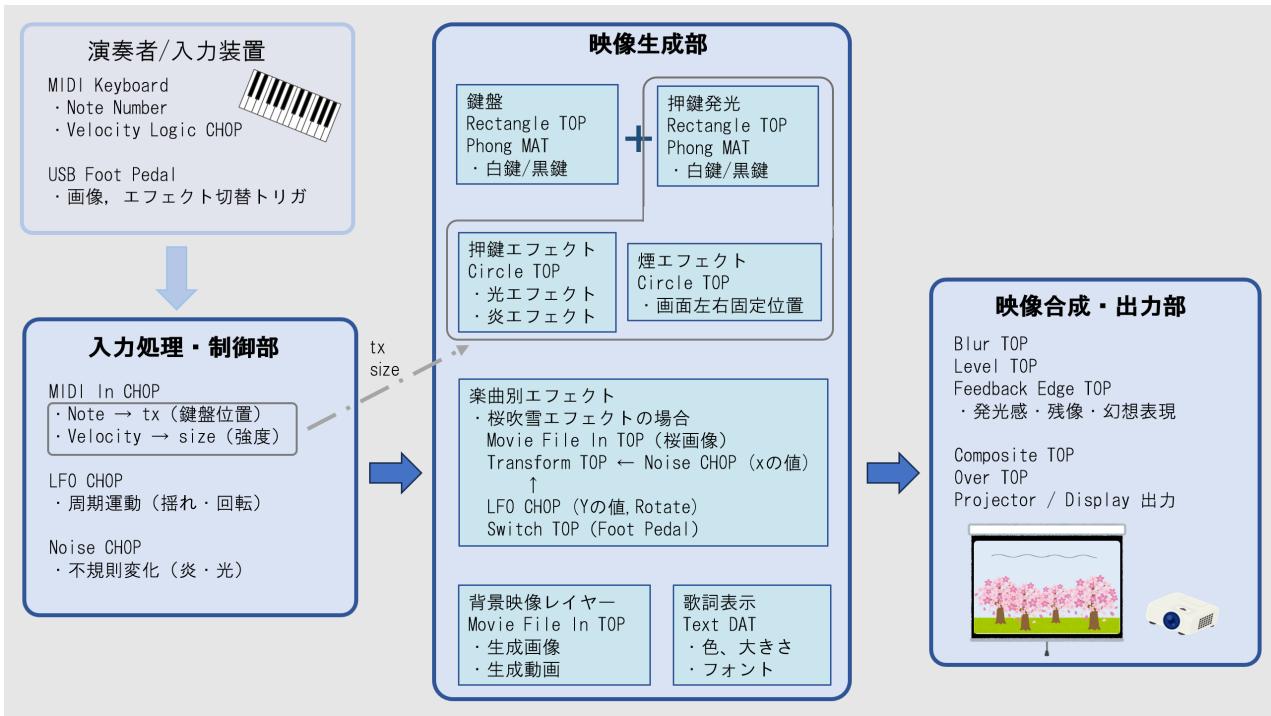


図 5: TouchDesigner 内のシステム構成図

像素材やエフェクトを同時に扱うようになるにつれて、描画処理および入力処理の負荷が増加した。その結果、映像の表示に遅延が生じるなど、リアルタイム性の面で課題が見られた。その点、TouchDesigner は、GPU を活用したりアルタイム処理に優れており、Processing で問題となっていた映像表示のタイムラグを軽減できる点が大きな利点であった。

また、Processing による映像表現は、主に図形描画を中心とした構成となっており、表現の変更やエフェクトの追加にはソースコードの書き換えを要する。一方、TouchDesigner による映像表現では、複数の映像素材やエフェクトを組み合わせた構成が可能であり、色彩表現や動きのバリエーションにおいて、より高いデザイン性を実現している。そして、ノードベースの操作体系により、映像の合成やエフェクト処理を視覚的に設計・調整できるため、デザイン性の自由度が高く、表現内容に応じた柔軟な映像制作が可能である。

この比較から、TouchDesigner は演奏内容に応じた視覚的に豊かな映像演出を柔軟に設計できるだけでなく、MIDI 入力や外部デバイスとの連携も容易であり、ピアノ演奏情報やフットペダル操作を映像制御に直接反映できることが確認できる。よって、演奏と映像をリアルタイムに統合する制作において、TouchDesigner は有効な制作環境であると判断した。

5.2 制作した映像システムの全体構成

本システムにおける映像制作は、TouchDesigner を用い

たノードベースの構成によって実装した。図 5 に、制作した映像システム全体のネットワーク構成を示す。制作した映像システムは、大きく分けて「入力処理部」「映像生成部」「映像合成・出力部」の三つの要素から構成されている。

5.2.1 入力処理部

入力処理部では、MIDI 信号などの外部入力に加え、LFO CHOP や Noise CHOP を用いて周期的またはランダムに変化する制御信号を生成し、それらを映像の動きやエフェクト強度の制御に利用している。

MIDI 信号を入力として、各鍵盤に対応するノート番号を水平方向の位置情報 (tx) に変換し、これを基準として鍵盤から発生する光エフェクトの x 座標を決定している。また、ベロシティの値はスケール値 (size) に変換され、鍵盤から発生する光や炎の大きさに加え、煙状エフェクトの強度にも用いられる。さらに、LFO (CHOP) や Noise (CHOP) を利用することで、入力画像の動きや桜吹雪、炎の揺らぎといった時間変化を伴う映像表現を実現している。

5.2.2 映像生成部

映像生成部では、TOP および Geometry COMP を用いて鍵盤および各種エフェクトの描画を行っている。鍵盤表現は、まず一つの矩形 (Rectangle TOP) を基に白鍵を複製して配置し、その上に位置を調整した黒鍵用の矩形を重ねる構成とした。

さらに、押鍵時の視覚的フィードバックを表現するため、白鍵および黒鍵それぞれに対して、押された場合のみ発光する矩形オブジェクトを別途用意し、MIDI 入力に応じて表示・非表示を制御している。この構成により、演奏

中に実際に押された鍵盤のみが発光する仕組みを作っている(図 6)。

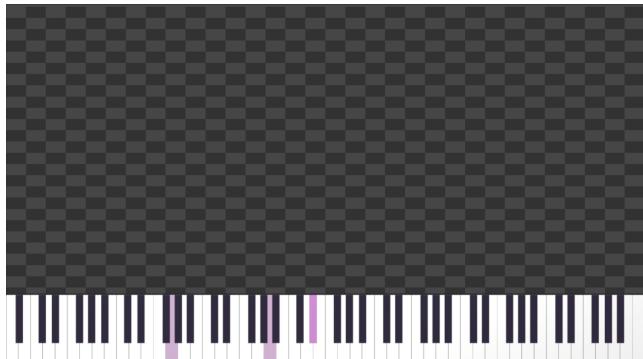


図 6: 打鍵によって光る鍵盤

図 7 に示すように、ピンク色の鍵盤から飛び出す光および鍵盤上で発光する炎のエフェクトは、Circle TOP を基に構成している。これらのエフェクトは、押鍵時に生成され、鍵盤が押されている時間に比例して縦方向に伸びる構造とした。これにより、鍵盤を長く押し続けるほど、より長い光が飛び出す視覚表現となる。

光および炎の出現位置は、各鍵盤に対応する位置情報(tx)によって制御されており、エフェクトの大きさや強さは MIDI ベロシティ値を変換したスケール値 (size) によって決定される。また、炎の揺らぎや光の不規則な動きには Noise CHOP や LFO CHOP を用い、時間的に変化する制御信号を付加している。

さらに、Blur TOP, Level TOP, Feedback Edge TOP の各パラメータを調整することで、円形オブジェクトの輪郭を拡散させ、発光感や残像を強調し、光や炎のように見える表現となるよう工夫している。



図 7: 鍵盤から出る光・炎エフェクト

楽曲「春よ、来い」では、桜の花びらが舞い落ちる情景を表現するため、桜吹雪エフェクトを映像演出として用いている(図 8)。桜の花びらの画像は Movie File In TOP により読み込み、複数の Transform TOP を用いて複製・配置した。

各花びらの移動には LFO CHOP で生成した周期的な数

値を用い、Translate X および Y に適用することで、左右に揺れながら落下する動きを再現している。また、Rotate パラメータにも LFO による制御を加えることで、花びらが回転しながら降ってくるような視覚表現とした。

さらに、各 Transform TOP に適用する LFO の位相や値をわずかにずらすことで、花びら一枚一枚の動きを変化させ、単調さを抑えた自然な桜吹雪の表現を実現している。

桜吹雪エフェクトの表示・非表示は Switch TOP により制御しており、指定した USB フットペダルが押された時のみ花びらが散る演出が発生する構成とした。この仕組みにより、演奏者が任意のタイミングで情景演出を切り替えることが可能となっている。なお、本手法は桜の花びら以外の画像にも容易に適用することができる。なお、本エフェクトは使用する画像を差し替えるだけで他の楽曲にも容易に適用できる構成としており、楽曲「きらきら星」では桜の花びらの代わりに星の画像を用いて、星が降り注ぐ映像演出を行った。

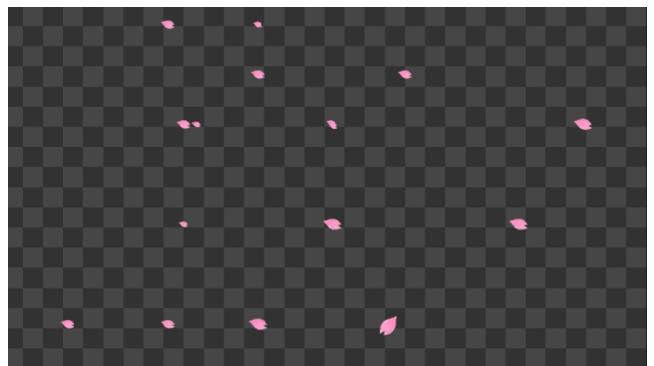


図 8: 桜の花びらが散るエフェクト

煙エフェクトは、USB フットペダルの指定位置が踏まれた際に発生する演出として実装している(図 9)。このエフェクトでは、Circle TOP を用いて煙の基本形状を生成し、MIDI 信号から取得したベロシティ情報をスケール (size) の制御にのみ使用している。発射位置には鍵盤位置に対応する tx は用いず、画面の右端および左端に固定した位置から煙が噴き出す構成とした。

煙は画面上部から下方向へ発射され、白色を基調とした表現によって煙の質感を表現している。この仕組みにより、鍵盤を強く打鍵した場合には煙のサイズが大きくなり、弱い打鍵では控えめな煙が生成されるなど、演奏の強弱が視覚表現に直接反映される演出を目指している。

5.2.3 映像合成・出力部

映像合成・出力部では、Composite TOP や Over TOP を用いて、入力処理部および映像生成部で作成した各映像要素を統合する構成としている。鍵盤、光や炎、煙、桜吹雪などの映像はそれぞれ独立したレイヤーとして管理され、Over TOP によって段階的に合成される。

この構成により、各映像要素を個別に調整・制御しなが

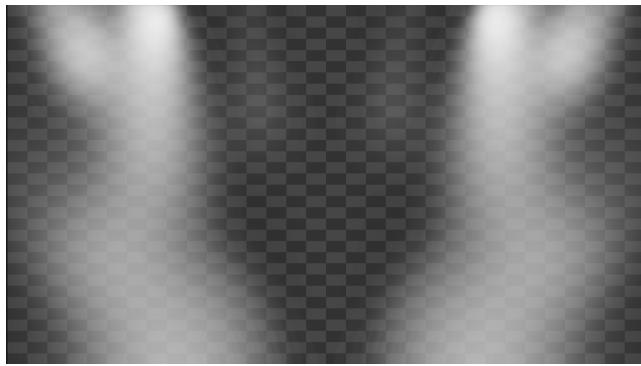


図 9: 煙エフェクト

ら、全体として一つの映像としてまとめることが可能となっている。最終的に統合された映像は 1 番最後に配置した Over TOP 出力され、外部モニターやスクリーンへの表示を前提とした映像信号として扱われる。

5.3 生成画像・歌詞データの利用方法

5.3.1 生成画像の表示

生成画像の表示には Movie File In TOP を用い、生成 AI によって作成した画像ファイルを TouchDesigner に読み込んでいる。読み込まれた画像は Transform TOP を通して拡大・縮小および位置調整を行い、Scale X および Y に LFO CHOP から生成した周期的な数値を適用することで、画像が緩やかにズームイン・ズームアウトを繰り返す動きを付加している。これにより、静止画像であっても映像としての動きを持たせることができるようになっている。

また、画像には Level TOP を用いて全体をやや暗くする処理を加えている(図 10)。これは、鍵盤から発生する光や炎のエフェクト、および歌詞テキスト表示を視覚的に強調することを目的としており、背景画像としての役割を明確にするためである。

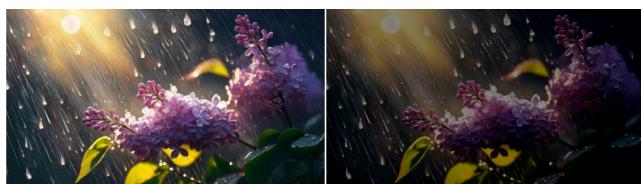


図 10: 元の画像と加工後の画像

生成画像の切り替えは、USB フットペダル操作によって行われる。フットペダルからの入力値をトリガーとして、Python によるスクリプトを実行し、指定したフォルダ内に保存された画像ファイルを順に Movie File In TOP に読み込む構成とした。これにより、演奏中でも演奏者自身が足操作のみで表示画像を切り替えることが可能となっている。

画像切り替え時には、LFO CHOP の位相をリセットする処理を行い、ズーム動作が毎回同じ状態から開始されるようにしている。これにより、画像切り替え直後の映像の動

きが安定し、視覚的な違和感を抑える工夫を行っている。

5.3.2 歌詞の表示

歌詞の表示には Text DAT および Text TOP を用いている。画像生成システムによって事前に作成された歌詞テキストファイルを Text DAT として読み込み、各行を 1 フレーズの歌詞として扱う構成とした。

歌詞の切り替え処理は、USB フットペダルからの入力をトリガーとして実行される。フットペダル操作により Count CHOP の値を更新し、その値を現在表示する歌詞行のインデックスとして利用している。Count CHOP の値が歌詞の総行数を超えた場合には、インデックスを 0 に戻すこと、歌詞表示が先頭に戻るよう制御している。

表示する歌詞は、取得したインデックスに対応する Text DAT の行を参照し、その内容を Text TOP に反映することで画面上に描画される(図 11)。この処理により、歌詞を一行ずつ順番に切り替えて表示することが可能となっている。なお、歌詞の切り替えタイミングは、生成画像が切り替わるタイミングと同一のトリガーによって制御されており、歌詞表示と生成映像が常に同期するよう設計されている。これにより、画像に対応した歌詞が表示されるようになっている。

また、Text TOP では文字の色、大きさ、フォントを調整し、背景映像やエフェクトとの視認性を考慮したデザインとしている。これにより、映像演出の一要素として歌詞を表示させている。



図 11: 歌詞の表示

5.4 映像の最終合成および出力

最終合成段階では、複数個の Over TOP を用いて各レイヤーの重なり順を調整し、重要な演奏情報に基づくエフェクトが視覚的に埋もれないよう構成した(図 12)。鍵盤に連動するエフェクト、背景映像、歌詞表示など、複数の映像要素を一つの画面として整理・統合することを目的としている。

具体的には、鍵盤や光・炎などの演奏に直接対応する要素を前面に配置し、桜吹雪や生成 AI による画像を背景として配置することで、画面全体の情報量と視認性のバランス

スを取っている。完成した映像は 1 番最後に配置した Over TOP へ出力され、ライブパフォーマンスにおける最終的な映像表現としてモニターまたはスクリーンに提示される。



図 12: 完成映像

6. 足キーボードを用いた映像制御について



図 13: USB フットペダルの配置イメージ

演奏中に映像演出を直感的に制御する手段として、3 入力の USB フットペダルを用いた足操作による映像制御システムを導入した(図 13)。手を使用せずに映像演出を操作することで、ピアノ演奏への影響を最小限に抑えつつ、演奏と映像を同期させることを目的としている。

フットペダルは、歌詞入力画面において区切られたタイミングと同じタイミングで操作する必要があるため、演奏中に自然に踏める位置として、ピアノの下部・左足側に設置した。一方で、ピアノ本来の足ペダル(ダンパーペダル等)は通常通り自由に使用できるようにし、演奏表現を干渉しない構成とした。

演奏時のフットペダルを踏むタイミングを間違わないようにするため、楽譜には踏むタイミングを事前に記載し、該当箇所にシールを貼ることで視覚的に把握できるよう工夫した。演奏者はこの目印を基に、演奏中に適切なタイミングでフットペダルを踏むことで、映像演出を制御する。

3 つのフットペダルにはそれぞれ異なる役割を割り当てている。「春よ、来い」の演奏で用いた楽譜とシールの色に

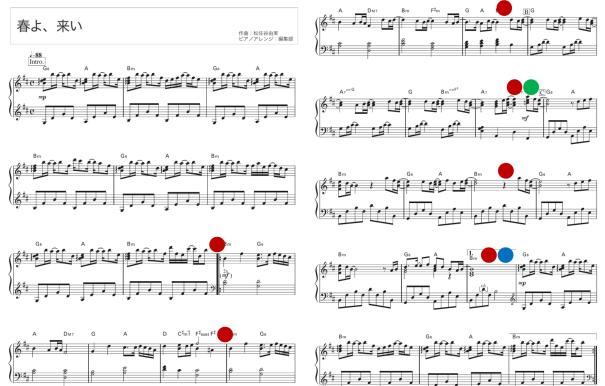


図 14: 演奏者が使用している楽譜



| シールの色 | 赤 | 緑 | 青 |
|----------|-----------|--------|--------|
| 役割 | 歌詞・画像切り替え | 煙エフェクト | 桜エフェクト |
| 割り当てた文字列 | Q | Z | T |

図 15: フットペダルの役割(「春よ、来い」の場合)

について図 14 と図 15 を用いて説明する。

図 15 に示すフットペダルには、識別を容易にするため、それぞれ異なる色のシールを貼付している。一番左のペダルには赤色のシールを貼り、踏み込み操作に応じて特定の文字列(Q)を TouchDesigner へ送信することで、歌詞および生成画像の切り替えを行う。この操作により、楽曲の進行に合わせて映像および歌詞の表示内容を更新する。中央のペダルには緑色のシールを貼り、対応する文字列(Z)をトリガとして、桜の花びらエフェクトの表示・非表示を制御する。これにより、楽曲の雰囲気に応じた情景演出の切り替えが可能となっている。一番右のペダルには青色のシールを貼り、ペダル操作によって送信される文字列(T)を TouchDesigner 側で受信・判定することで、煙エフェクトを発生させる。このエフェクトは、演奏の盛り上がりに応じた視覚的アクセントとして機能する。以上のように、各ペダルは色分けによって役割を明確化しており、楽譜上に示された同色のシールのタイミングでペダルを踏むことで、ピアノ演奏と映像表現を同期させた制御が可能となっている。

このように、フットペダルを用いることで、演奏者が自身の感覚に基づいて任意のタイミングで映像を切り替えることが可能となっている。従来のように、あらかじめ用意した映像を時間の流れに沿って再生する手法や、押鍵数や拍数を基準に映像を制御する手法では、テンポや演奏の正確性が強く求められる。一方、本手法ではそうした制約を演奏者が意識する必要がなく、演奏の流れや表現に合わせ

て柔軟に映像演出を操作できる。そのため、即興的な演奏や、楽譜を用いない耳コピによる演奏に対しても対応可能であり、演奏と映像がより密接に結びついた自由度の高い表現を実現している。

7. 実演について

今回構築した映像システムについて、実際の演奏環境における有効性を検証するため、複数の場面で実演および簡易的な評価を行った。

7.1 公開実演

2025 年 12 月 20 日、福知山市新町商店街で開催された「まちなかで音のとびらを開く会」というイベントにおいて実演を行った(図 16)。実演では、筆者自身によるピアノ演奏として「春よ、来い」および「secret base～君がくれたもの～」[9] を演奏し、演奏に連動した映像演出を提示した。また、ゼミ所属の友人に協力を依頼し、「人生のメリーゴーランド」[10] を演奏してもらい、このシステムが筆者以外でも使用可能であることを確認した。さらにこの曲では、静止画ではなく動画素材を背景映像として採用している。動画は生成 AI である Sora [11] により生成し、約 5 秒の動画をループ再生する構成とした。実演においても、映像のループ再生による違和感や動作不良は確認されず、本システムにおいて生成動画を背景映像として問題なく利用可能であることが示された。



図 16: 実演風景

7.2 研究室内デモ発表およびテスト演奏

さらに、研究室でのデモ実演回においても本システムを用いた演奏を行った。また、ゼミ生 8 名を対象に、簡易的なテスト演奏を実施した。テスト曲には「きらきら星」を用い、演奏者全員に対してその場で楽譜を配布した。

楽譜上には、フットペダル操作のタイミングを示すため、青色のシールを貼付しており、演奏者にはその位置で USB フットペダルの一番右のペダルを踏むよう指示した。これにより、演奏と映像切り替えを同時に行う操作が、初見の演奏者にも可能であるかを検証した。

これらの実演を通して、生成 AI による画像、歌詞表示、鍵盤入力に連動したエフェクト、およびフットペダルによる映像切り替えが、実際の演奏環境においても安定して動作することも確認した。

7.3 評価結果と考察

テスト演奏の結果、全員が概ねシールで示したタイミングに合わせてフットペダルを踏むことができており、本システムの操作方法が直感的であることが確認された。一方で、一部の演奏者においては、フットペダルを一度の操作で二回踏んでしまい、意図せず画像が二度切り替わるといった操作ミスも見られた。

また、ピアノ演奏に慣れている演奏者は、演奏を中断することなくスムーズにフットペダル操作を行えていたのに対し、演奏に不慣れな演奏者の場合、ペダル操作の直前で演奏が一瞬詰まる様子が確認された。しかし本システムは、ピアノ演奏に一定の習熟を持つ演奏者が、ステージ上で演奏と映像操作を同時にを行うことを想定して設計されており、ピアノ経験者による演奏では、想定どおり、演奏と足操作を無理なく両立できていた。これらの結果から、足による操作と演奏の同時進行には演奏経験が影響し、主にピアノ経験者を対象としたパフォーマンス用途において有効であると考えられる。

システムを利用した被験者からは「テンポを厳密に気にしなくてよい」、観察者からは「音楽と映像が自然に合っていると感じた」といった評価が得られた。これは、映像切り替えを時間や拍数に依存させるのではなく、演奏者自身の判断で行えるというシステムの特徴によるものと考えられる。

以上の実演および評価を通じて、フットペダルを用いた映像制御手法は、公開演奏および複数の演奏者による使用においても有効に機能することが確認された。また、テンポの正確性に縛られず、演奏者の表現や即興性を損なわない点において、従来の映像同期手法に対する有効な代替手法となる可能性が示されたと考える。

8. 課題と今後の発展について

本研究では、ピアノ演奏に連動した映像制御システムを構築し、実演および複数の演奏者によるテストを通じて、その有効性を確認した。一方で、実運用や評価結果から、いくつかの課題も明らかとなった。テスト演奏において、一部の演奏者がフットペダルを一度の操作で二回踏んでしまい、意図せず画像が二度切り替わるといった誤操作が確認された。これは、ペダル操作に対する入力判定が敏感であることや、演奏に集中することで足操作への意識が分散されることが原因であると考えられる。の課題に対しては、ペダル入力にクールタイムを設ける、あるいは一定時間内の連続入力を無効化するなど、ソフトウェア側での誤

操作防止処理を導入することで改善が可能であると考えられる。

また、評価結果から、ピアノ演奏に不慣れな演奏者では、フットペダル操作の直前に演奏が一瞬滞る傾向が見られた。このことから、演奏と足操作を同時に行うことには一定の演奏経験が必要であり、本システムは主にピアノ経験者を対象とした構成であることが明らかとなった。一方で、もともとステージ上でのパフォーマンス用途を想定して設計されており、ピアノ経験者による演奏では、演奏と映像操作を無理なく両立できるという設計意図に沿った結果が得られている。今後は、初心者や演奏経験の浅い演奏者にも対応可能なモードとして、操作回数を減らした簡易制御方式や、自動切り替え機能との併用などを検討する余地がある。

そして生成 AI による静止画像を中心とした映像素材を用いたが、映像表現としてはまだ発展の余地がある。今後は、生成画像をもとにしたリアルタイムエフェクトの変形・合成処理を強化することで、より音楽表現と密接に連動した映像演出が可能になると考えられる。また、現在は歌詞や楽曲イメージをテキストとして入力しているが、楽曲構造（A メロ・B メロ・サビ）や演奏の強弱、テンポ変化などを解析し、それらを映像生成や制御に反映させることで、演奏内容により適応的な映像表現を実現できる可能性がある。

以上の課題を踏まえ、今後、ピアノ演奏を中心としたステージパフォーマンスにおいて、演奏者自身が映像表現を主体的に制御するための表現基盤として発展させることが可能であると考えられる。音楽表現と映像表現を分離するのではなく、演奏行為そのものに映像制御を組み込むことで、より没入感の高いライブパフォーマンスの実現が期待される。

9. まとめ

本研究では、ピアノ演奏に基づいて映像をリアルタイムに制御・生成する演奏運動型映像システムの構築を行った。MIDI キーボードから取得したノート番号およびベロシティ情報、ならびに USB フットペダルによる足操作を用いることで、演奏内容と映像演出を同期させ、演奏者自身が演奏中に映像を制御できる環境を実現した。

映像制作および制御には TouchDesigner を用い、鍵盤に連動した光や炎のエフェクト、煙や桜吹雪といった情景演出、生成 AI によって作成した画像および歌詞表示を統合的に扱うシステムを構築した。各映像要素をレイヤーとして管理し、合成順を調整することで、演奏に直結する視覚情報を前面に、背景表現を背面に配置するなど、ライブパフォーマンスに適した画面設計を構築した。

また、生成 AI を用いた画像生成システムを Python 上に実装し、歌詞や楽曲のイメージを入力することで、楽曲内

容に対応した映像素材を事前に生成する手法を提案した。これにより、歌詞を持つ楽曲だけでなく、歌詞がない楽曲に対しても柔軟な映像演出が可能となっている。

実演として、商店街イベントにおける公開演奏および研究室内でのデモ発表を実施した結果、本システムが実際の演奏環境において安定して動作することを確認した。特に、フットペダルを用いた映像切り替え操作は、演奏のテンポや拍数に厳密に縛られることなく、演奏者の判断によって柔軟に行える点で高い評価を得た。一方で、演奏経験の差によって操作のスムーズさに違いが見られ、足操作と演奏の同時進行には一定の演奏習熟が影響することも明らかとなった。

これらの結果から、映像制御システムはピアノ演奏に一定の習熟を持つ演奏者が、ステージ上で演奏と映像操作を同時に行うパフォーマンス用途において有効であることが示された。演奏者自身が映像表現に直接関与することで、音楽と映像が相互に影響し合う一体化の表現を実現できる点は、本研究の成果である。

提案した手法は、従来のようにあらかじめ用意した映像を時間の流れに沿って再生する演出とは異なり、演奏中に演奏者自身が映像を操作することを可能とするシステムである。これにより、演奏テンポや演奏技術の正確性に強く依存することなく、演奏者の感情や表現意図を映像演出に直接反映させることができる。今後、操作性や表現手法を拡張することで、より自由度の高い演奏表現や、多様なパフォーマンス形態への展開が可能になると考えられる。

参考文献

- [1] Erdmann, M., von Berg, M. and Steffens, J.: Development and Evaluation of a Mixed Reality Music Visualization for a Live Performance Based on Music Information Retrieval, *Frontiers in Virtual Reality*, Vol. 6 (online), DOI: 10.3389/frvr.2025.1552321 (2025).
- [2] Graf, M., Opara, H. C. and Barthet, M.: An Audio-Driven System For Real-Time Music Visualisation (2021).
- [3] 小川剣二郎, 中村聰史: 1 視点固定型ライブ映像における映像・音響表現自動付与による音楽体験拡張, 情報処理学会論文誌, Vol. 66, No. 12, pp. 1715–1724 (2025).
- [4] USB3 連フットペダルスイッチ マウス操作対応 [RIFP3BK] 有限会社 ルートアール (Route-R) パソコンパーツ・周辺機器の輸入卸売、輸入代行、OEM の事なら全てルートアール (Route-R) にお任せ下さい。
- [5] DALL-E 3, <https://openai.com/ja-JP/index/dall-e-3/>.
- [6] 松任谷由実: 松任谷由実 - 春よ、来い (2019).
- [7] 株式会社ボーンデジタル: TouchDesigner.
- [8] Processing: Welcome to Processing!, <https://processing.org/>.
- [9] MUSIC Liverary: ZONE「secret Base ~君がくれたもの~」MUSIC VIDEO (2021).
- [10] Joe Hisaishi Official: Joe Hisaishi - Merry-Go-Round (from 'Howl's Moving Castle') (2020).
- [11] Sora, <https://openai.com/ja-JP/sora/>.