動画情報に基づく環境音楽実時間生成ソフトウェアの設計
―音楽の自動生成部分を対象として―

2015SE051 宮崎翼　2015SE084 鳥居基槻
指導教員：沢田篤史

1　はじめに

近年, 動画配信サービスの拡大により多くの人々が手軽に動画配信することができるようになった. 音楽を動画と同時に再生し BGM として利用されることが増加した. 映像に伴った音楽は場の状況や雰囲気を演出する効果がある. 「音」は環境を構成する重要な要素であり, 環境と音楽の関わりはその重要性を増している [1]. ここで環境音楽は聴くための音楽ではなく, 自然に耳に入る音楽を指す. 音楽に対してその雰囲気にあったミュージックビデオを自動で生成するといった, 音楽に対して映像を出力するソフトウェアは存在するが, 我々の調査した範囲では映像に対して音楽を出力するソフトウェアは実用化されていない. 映像から音楽を生成するためには前提として音楽の知識が必要であり, 映像と音楽の対応関係を見極め, 生成された音楽が映像の雰囲気に合っているかを検証しなければならない. 動画を基に音楽を生成することを目的とした研究は山本ら [2] や清水ら [3] が行っており, 動画から音楽に変換する際には共通してバッチ処理が行われている.

山本らの研究の中では音楽心理学に基づいた自動作曲を行っている. 動画全体を先に読み込み, 色調や明度から音楽情報を決定する方法をとっているので曲の構成をより細かく指定することができるという利点がある. 山本らの研究や清水らの研究では音楽を生成する際に, 映像解析したデータを一括で詳細に音楽に変換しているので, 処理に時間を要しリアルタイム性の実現はされていない. また, 現在動画を基に音楽をリアルタイムに生成する研究は行われていない.

本研究の目的は, 動画映像からリアルタイムに環境音楽を生成するシステムの実現である. このシステムは, 映像を解析して特徴量を抽出する映像解析部分と, 特徴量からそれに整合した環境音楽を生成する音楽生成部分に分かれる. 我々は, システム全体のアーキテクチャ設計に基づいて音楽生成部分を設計・実装した.

このアーキテクチャ設計は Pipes and Filters アーキテクチャに基づき実施した. 映像解析部分から渡されるデータストリームをフィルタに通すことで音楽を生成する. 映像解析部分で読み取った映像から, 音楽心理学に基づき色調の占める割合や明度を抽出し, フィルタ処理により MIDI 形式に変換し, 音楽生成部分へパイプで受け渡しを行う. 受け取ったデータをフィルタに通し音楽を生成していく. なお, 我々の考えるシステムではリアルタイム性を重視するので, 既存の研究で行っている曲の構成を考慮したシステムの設計は行わず, ごく単純な音楽を生成する.

2　背景技術

2.1　自動作曲

作曲は高度な芸術活動の一つであり, 多くの専門的知識と優れた感性を必要とする特殊な分野である. 作曲の過程を全て自動化することは極めて困難であるが, その内部にある機械的作業の部分について自動化の可能性を探ることは意義のある行為である [4].

自動作曲はコンピュータが全て自動で作曲を行うものではなく, あらかじめ書いたプログラムに則した計算結果が作曲である. 1957 年には Lejaren Hiller と Leonard Isaacson がイリノイ大学のコンピュータ ILLIAC I を用いて「イリアック組曲」を作曲した. これは世界で初めてコンピュータを作曲の自動計算に利用した例であった.

2.2　山本らの自動作曲システム

山本らのシステムでは決まった長さの動画全体を先に読み込み, 動画の長さに合わせた音楽を MIDI ファイルとして出力する. 概念図を図 1 に示す.

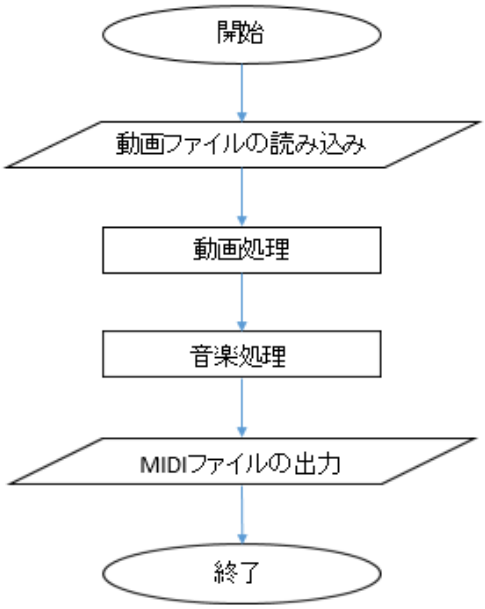

図 1　山本らのシステム

最初に動画ファイルの読み込みを行い, 動画処理を実行する. 読み込んだ動画から全フレームをビットマップとして切り出し, 各フレームにおける全ピクセルに対して RGB 成分を抽出する. 抽出した RGB 成分は 2 次元配列構造体

に格納する．また, 今回はより人間の知覚に基づいた HSB カラーモデルを,RGB カラーモデルと共に使用する．RGB 成分を HSB 成分に変換し, 先程と同様,2 次元配列構造体に格納する．これらの色情報を用いて音楽処理を行う．

つぎに, 動画の長さに応じて MIDI データの長さを決定し, 動画と曲の長さを対応させる．また, 曲のテンポと動画のフレーム数を対応させる．動画の中で大きく画面が変化するタイミングを場面転換と見なし,HSB 各値の変化量の合計が閾値を上回ったときとする．場面転換の後, 次に場面転換するまでの区間を一つのブロックとする．各ブロックに含まれるフレームの色情報をもとにして, 音楽心理学を参照しながら, ブロック毎にコード進行, リズム等の音楽情報を決定する．最後に RGB 成分からコード進行,HSB 変化量からリズム, 色相割合から伴奏, 明度からメロディラインの決定を行い,MIDI ファイルを生成し出力する．

2.3 バッチ処理とデータストリーム処理

バッチ処理とはあらかじめ決めておいた一連の手順を一括で処理する技術である．一定期間で発生したデータを一括処理するので大規模なデータを効率的に処理することができる．

データストリーム処理は継続的に発生するデータをリアルタイムに処理・分析する技術である．新しいデータが絶えず流れてくるため, 処理が終わったデータは一定時間メモリに保存されるが, 時間が経つと消去される．

それぞれの概念図を図 2, 図 3 に示す．バッチ処理は一定量のデータが蓄積してから処理し, データストリーム処理はデータが流れてくる度に処理するのでデータストリーム処理はバッチ処理に比べて遅延が小さくなる．本研究では映像データを即座に音楽データに変換する必要があるので, 遅延の小さいデータストリーム処理を行うべきである．

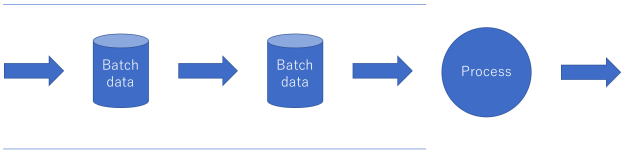

図 2 バッチデータ処理

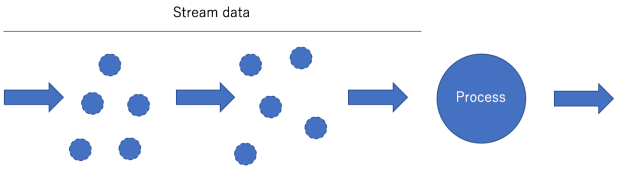

図 3 データストリーム処理

2.4 Pipes and Filters アーキテクチャ

Pipes and Filters はデータストリームを扱うシステムのための構造である．データは隣り合ったフィルタ間をパイプを通じて渡される．パイプとフィルタの組み合わせを変えることにより, システムを柔軟に変化させることができる．

Pipes and Filters は以下の

- データソース
- パイプ
- フィルタ
- データシンク

の 4 つの要素で構成されており, データソースからパイプとフィルタを通じて最終的にパイプからデータシンクへとデータが流れて行く．以下の図 4 のようにフィルタで処理が行われ, パイプによってデータの受け渡しが行われる．

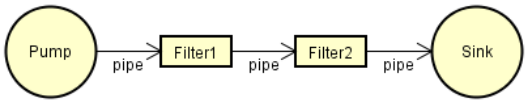

図 4 Pipes and Filters

3 環境音楽自動生成システムの設計

3.1 システムの概要

図 5 にシステムの概念図を示す．カメラ等から読み込んだ映像を色調や明度の情報を解析し, それらを基に音楽情報に変換した後, 映像と音楽を同期して再生するシステムである．映像解析システムでは, 読み取った映像のある瞬間の色調が占める割合や明度を解析し, リアルタイムに MIDI 形式のデータを生成, 更新する．音楽生成部分では, 一定時間おきに読み取った MIDI 形式のデータをもとに音楽生成する．生成された音楽は, 同期のために一定時間バッファリングされた映像とともにプレイヤーで再生される．音楽自動生成部分は 1 小節単位で譜面の生成, 再生を行う．何もしなければ必ず 1 小節分の遅延が発生するので同期処理が必要となる．映像解析部から解析結果の MIDI データのほかに未処理の映像データを受け取り, バッファストレージに一時保存する．音楽の BPM に応じた 1 小節分の時間が経過した後映像を出力する．

3.2 音楽生成部分のアーキテクチャ設計

音楽生成部分では, 映像解析部分から受け取った MIDI 信号をきっかけに環境音楽を自動生成するアーキテクチャ

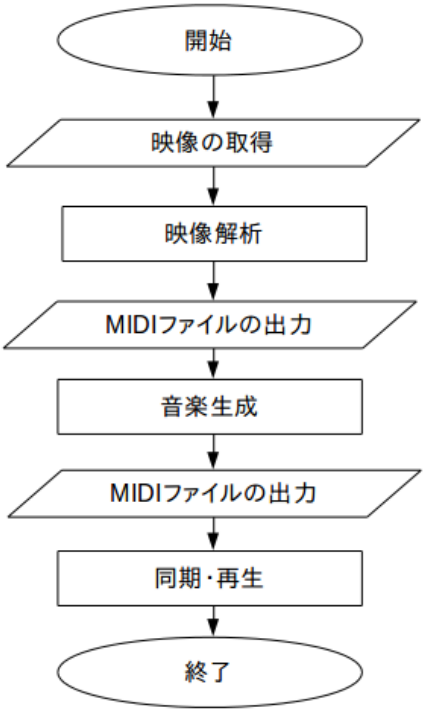

図 5　環境音楽自動生成システム

を,Pipes and Filters を適用し設計する．本研究では映像と音楽が同期するリアルタイム性について焦点を置き, そのためにストリーミングデータを扱うアーキテクチャを採用する必要がある．本研究では音や映像などのデータストリームを扱い, それらの扱いに適したアーキテクチャスタイルである Pipes and Filters アーキテクチャを適用した．Pipes and Filters アーキテクチャはデータをストリームとして扱うシステムのための構造を提供するので本研究に適していると考える．

図 6 に音楽生成部分のアーキテクチャを示す．

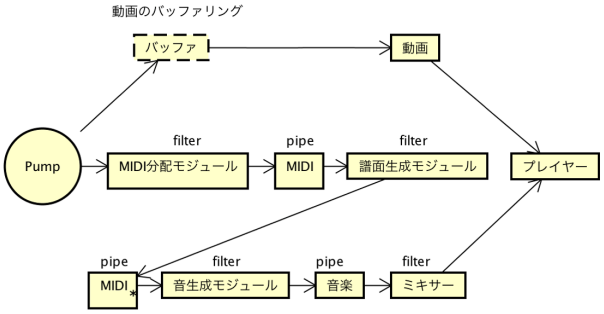

図 6　音楽生成部分のアーキテクチャ

音楽生成部分の構成と, コンポーネントごとの役割について説明する. 音楽生成部分は

- MIDI 分配モジュール
- 複数の譜面生成モジュール
- 複数の音生成モジュール
- ミキサー

によって構成される．MIDI 分配モジュールは, 映像解析部分から受け取った MIDI 信号をあらかじめ定められた値に対応した譜面生成モジュールへ送る．映像解析部分は複数の値の MIDI 信号を同時に出力するので, このような分配モジュールが必要である．譜面生成モジュールは音生成モジュールが演奏する譜面データを生成, もしくはあらかじめ制作したものを出力する．出力するきっかけとなる MIDI 信号を受信すると譜面データを MIDI 形式で出力する．音生成モジュールは譜面データをもとに演奏をし, 人が聞く音を出力する．ミキサーは複数の音生成モジュールから出力された音を入力し, 各音のレベルとパンニングを設定し, 一つの音楽データとして出力する．

本システムは, 映像解析を除いた全てを, オブジェクト指向プログラミング音楽用言語である MAX を用いて実装する．実装の大部分は BEAP という MAX 内でモジュラーシンセサイザの構築環境をシミュレートしたオブジェクトを用いる．BEAP を用いる理由は負荷が小さくタイムラグが発生しにくいので, よりリアルタイム性を追求できるからである．

3.3　MIDI 信号解析

本研究では MAX を使用する都合上, 最も互換性が高く音質劣化の起こらない MIDI を扱う．映像解析部分から 0.3 秒ごとにメロディ生成のきっかけとなる信号, コード進行情報を含んだ信号をそれぞれ MIDI 形式で MIDI 分配モジュールが受け取り, 各譜面生成モジュールへ渡す．受け取る MIDI 信号はメロディパート, コードパートとパーカッションパートで解析の方法が変わる．メロディパートとパーカッションパートは,MAX 上ですべて自動生成を行うので, 映像解析部分から受け取る MIDI 信号の役割はあくまで生成のきっかけとなるトリガーである．それに対してコードパートは, 映像解析部分が画面の色彩が占める割合に応じてコード進行情報を生成するので,MAX ではコード進行を解析し, コードの構成音を基にコードパートを生成する．

3.4　音楽の自動生成

MIDI 分配モジュールによって和声楽器, 旋律楽器, パーカッションなどに分けられた MIDI 信号を基にそれぞれ実際の音を生成する．各楽器パートごとに生成された音声は, それ単体では心地よい環境音楽としては不十分である．それを解決するためにはミキシングを行う必要があり, ミキサーモジュールで BEAP を用いて各楽器パートごとの音量とパンニングを適切な設定にする．ミキシング後生成された音楽はプレイヤーで動画とともに再生される．

動的構造を図 7 に示す. 以下の順に処理が行われる．

1. 映像解析部分は動画データをバッファストレージに一時的に保存

2. 映像解析部分から MIDI 分配モジュールへ解析データを転送し分配

3. MIDI 分配モジュールから譜面生成モジュールへ MIDI データを転送し譜面生成

4. 譜面生成モジュールから音生成モジュールへ MIDI データを転送し音生成

5. 音生成モジュールからミキサーへ MIDI データを転送し音楽の生成

6. ミキサーからバッファストレージへ音楽の生成通知

7. 音楽の生成通知を受け取り, バッファストレージからプレイヤーへ動画を転送し再生

8. ミキサーからプレイヤーへ音楽を転送し動画と同時に再生

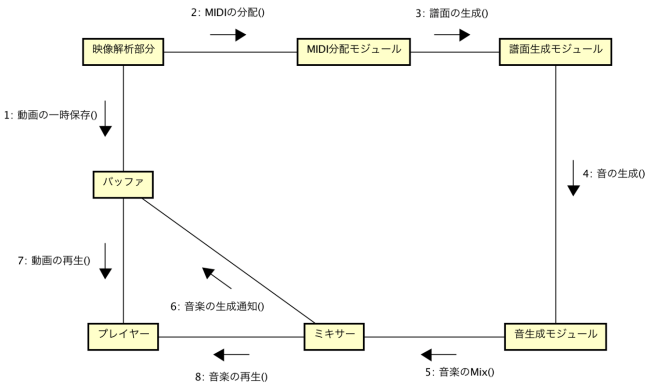

図 7　音楽生成部分の動的構造

4　考察

4.1　システムの実用性と課題

今回我々が設計, 実装したシステムは, 映像の色調が占める割合をもとに和音を生成し, 乱数を用いてメロディを自動生成する, あくまで基礎的なものである．実用性を高めるためには, 機械学習等を用いて映像をより詳細に解析し, 適した曲調を判断する必要があると考える．

音楽生成部分は, 譜面生成や音生成など役割ごとにモジュールを分割したアーキテクチャ設計を行っているので独立性が高く, 今後新たな機能を搭載するさいの拡張性に優れている．抽出する動画特徴量やパイプとフィルターを増やすことによってより複雑な音楽が生成できるようになると考える．メロディを除いてあらかじめ用意したパターンを組み合わせて音楽生成するので, 生成される音楽が似通ってしまう．よって, 様々なバリエーションの音楽が生成されることを期待するものにとって, 本システムは実用的とは言えない．また, 本システムは本格的な音楽理論やアレンジメント技法に則っていないので, 製品レベルの音楽が生成されることを期待する者にとっても, 実用的ではないと言える．課題として, 音楽の自動生成の方法, つまりアーキテクチャ中の譜面生成モジュールをより実際の環境音楽の特徴を反映させたものを実装する必要がある．さらに, 製品レベルの音楽作品を生成できるようにするために, 音生成モジュールではより音楽的な音色を生成できるようにする必要があると考える．

4.2　関連研究との比較

山本らの研究や清水らの研究では音楽を生成する際にバッチ処理を行い, 動画全体と同じ長さの音楽を生成するので変換に一定の時間を要する．我々の音楽生成部分では映像解析したデータの読み込みと, それらの音楽への変換を随時行うのでリアルタイム性が向上すると考える．実際に変換までの時間は検証できていないので今後行う必要がある．

山本らの研究は, 音楽心理学に基づき, 詳細に曲構成を決定しているのに対し, 我々の研究ではリアルタイム性に重点を置き, 詳細な曲構成の決定を重視していない．一方, 一部で乱数を用いている点は我々の研究と類似している．

5　おわりに

本研究では, 環境音楽を自動生成するソフトウェアのアーキテクチャを設計した．また, 全体として映像と音楽の同期を行うために実時間性を考慮したアーキテクチャを設計し, それに基づき実装を行った．

本システムのアーキテクチャを設計するにあたって, ストリーム処理に適した Pipes and Filters アーキテクチャを適用することによって再生される映像に同期した音楽を再生することが可能になった．また, 実装するにあたって BEAP を用いることでリアルタイム性を重視したシステムを作成することができた．さらに改良を加えるべき点としては, 製品レベルの環境音楽を生成できるよう, 実際の環境音楽の特徴や, 本格的な音楽理論やアレンジメント技法に則った実装がある．

参考文献

[1] 畑公也, “「環境音楽」、または「環境」と「音楽」,” 神戸薬科大学研究論集 : Libra = The journal of Kobe Pharmaceutical University in humanities and mathematics 16, 2016.

[2] 山本敏生, 宝珍輝尚, 野宮浩揮, “ 動画をもとにした自動作曲, ” 平成 21 年度情報処理学会関西支部大会, 2009.

[3] 清水柚里奈, 菅野沙也, 伊藤貴之, 嵯峨山茂樹, 高塚正浩, “ 動画特徴量からの印象推定に基づく動画 BGM の自動生成, ” 情報処理学会第 78 回全国大会, 2016.

[4] 保坂務, “ Java による自動作曲, ” 国際短期大学紀要第 20 号, 2005.