情報処理学会論文誌　Vol. 49　No. 12　3918–3932 (Dec. 2008)

インタラクティブに音楽を検索・発見できるソフトウェア「音楽夢想」

古山真之[†1]　大村廉[†2]　今井倫太[†2]

現在，広く利用されている音楽ファイルの再生・管理のソフトウェアにおける，キーワードベースの検索インタフェースでは，ユーザの曖昧な検索の要求を実現することは困難である．この要求を満たすためには，音楽情報の関連性に基づいた発見支援が必要である．しかし既存研究では，(1) 様々な音楽情報を組み合わせた情報の提示，(2) ユーザの操作に基づいた動的な情報提示，(3) 音楽情報間の関連性の強弱の調整，(4) 曲の直感的なグルーピング，といった 4 つの要求を十分に満たしていなかった．そこで本研究では，これらの要求を実現するために，インタラクティブに音楽を検索・発見できるソフトウェア「音楽夢想」を提案する．音楽夢想は，の 4 つの要求を「分類スタッキング」「芋づるサーチ」「集合コンバージェンス」「一筆書きグルーピング」機能によりそれぞれ実現する．音楽夢想の動作例，アンケートによるユーザ評価により，音楽夢想の各機能が有効に動作し，ユーザの曖昧な検索の要求を実現することが示された．

Ongaku Musou: Interactive Software for Searching and Discovering Music

Masayuki Furuyama,[†1] Ren Ohmura[†2]
and Michita Imai[†2]

The keyword base search interface implemented in widely-used software to play and manage music files are hard to meet demand of user's ambiguous search. To meet this demand, discovery support of music based on relations among music information is needed. However, previous works are difficult to meet following four demands: (1) showing information by combining various music information, (2) showing information based on user's operation, (3) controlling strength of relations among music information, (4) grouping music intuitively. To meet above four demands, we propose "Ongaku Musou", the interactive software for searching and discovering music. Ongaku Musou meets these demands by "Layer Stacking", "Dragging Out Search", "Element Set Convergence", and "Drawing Grouping" function. The operation example and the impression evaluation of Ongaku Musou showed that the each functions of Ongaku Musou work effectively and meet the demand.

1. はじめに

近年，音楽の分野ではパーソナルコンピュータ（PC）の記憶容量の増加，音声圧縮技術・インターネットの発達により，ユーザが大量の曲を PC に保存できるようになった．このため，ユーザは膨大なライブラリの中から曲を選択する必要が生じ，音楽ライブラリの検索・発見技術が重要となっている．

現在広く用いられている Windows Media Player [1)]，iTunes [2)] といった音楽ライブラリの再生・管理のためのソフトウェアでは，キーワードによる検索機能を提供し，結果をリスト形式で表示するという手法が主流になっている．この技術はユーザにとって目的の曲が明確であり，かつ曲に付随するキーワードが既知の場合は効率的に目的の曲を発見することが可能である．一方で，実際に曲を聴く場面では，「アーティストの名前を忘れたが，出身地や曲の雰囲気からアーティストを検索したい」というように明確なキーワードを思い出せない場合や，「好みに近い曲を漠然と検索したい」というように，そもそも明確な目的がなく曲を選択しようとする場合が多く存在する．キーワードベースのインタフェースでは，このようなユーザの曖昧な要求を満たすことは困難である．

この問題を解決するアプローチの 1 つは，音楽情報（音楽ファイルに付随するアルバムやアーティストなどのメタ情報，曲の雰囲気といった曲そのものから抽出された音響特徴量）の関連性に基づいた検索・発見支援を行うことである．音楽情報の関連性に基づいて類似した属性を持つ曲を近くに配置させることができれば，曲の集合の効果的な可視化およびブラウジングをすることが可能になる．これによって，明確な検索要求を持たないユーザの曲検索を支援できる．

しかし，明確な検索要求を持たないユーザの曲検索を支援を実現するためには，類似した属性を持つ曲を近くに配置するだけでなく，さらに以下に述べる要求を満たす必要がある．ユーザごとに検索・発見に用いたい音楽情報は異なるので，ユーザに対し効果的な発見支

†1 UBS 証券会社
UBS Investment Bank
†2 慶應義塾大学理工学部
Faculty of Science and Technology, Keio University

援を行うためには，「(1) 様々な音楽情報を組み合わせた情報の提示」を満たす必要がある．また，ユーザが注目している音楽情報にのみ基づいて情報を提示した場合，提示される音楽情報の範囲は限られてしまう可能性がある．したがって，ユーザが注目していない音楽情報に関しても提示する必要がある．さらに，ユーザの現在の興味に応じて，曲の関連性や類似する曲の集合に対するとらえ方は変化するため，曲の配置を動的に変化させる必要がある．すなわち，「(2) ユーザの操作に基づいた動的な情報提示」を実現する必要がある．さらに，ユーザの興味の範囲は曖昧性をもっており，比較的関連の薄い曲でも有用な検索結果となる場合も生じる．したがって，ユーザが音楽情報間の関連性を自由に調整して，関連の薄い曲，強い曲を選択しやすくする必要がある．すなわち「(3) 音楽情報間の関連性の強弱の調整」を満たす必要がある．また，膨大な音楽ライブラリ内から複数の曲を効果的に選択するためには，前述の (1)〜(3) を満足しつつ，曲の関連性の強弱に基づく「(4) 曲の直感的なグルーピング」を実現する必要がある．以上をまとめると，音楽情報の関連性に基づいた曲の発見支援のためには以下の 4 つの要求を満たす必要がある．

(1) 様々な音楽情報を組み合わせた情報の提示
(2) ユーザの操作に基づいた動的な情報提示
(3) 音楽情報間の関連性の強弱の調整
(4) 曲の直感的なグルーピング

2 章で後述するように，音楽情報の関連性に基づく楽曲検索，発見支援に関する研究は，これまでいくつかなされてきたが，これらの 4 つの要求を同時に満たすものは提案されてこなかった．

そこで本研究では，(1)〜(4) の要求を満たすために，インタラクティブに音楽を検索・発見できるソフトウェア「音楽夢想」を提案する．音楽夢想では，音楽情報に基づく分類をレイヤとして表現して，レイヤを自由に組み合わせる「分類スタッキング」機能により「(1) 様々な音楽情報を組み合わせた情報の提示」の要求を満たす．音楽夢想では，ユーザが選択した要素（曲やその曲の音楽情報）と関連する要素を次々に引き出し様々な要素の集合を出現させる「芋づるサーチ」機能により「(2) ユーザの操作に基づいた動的な情報提示」の要求を満たす．情報の密度や広がり方を直感的に調整する「集合コンバージェンス」機能により「(3) 音楽情報間の関連性の強弱の調整」の要求を満たす．そして「分類スタッキング」「芋づるサーチ」「集合コンバージェンス」機能により出現した曲の集合を，効果的にグルーピングするためにユーザが図形を描いて曲の集合をグルーピングする「一筆書きグルーピング」機能を提供し，「(4) 曲の直感的なグルーピング」の要求を満たす．音楽夢想では，これら 4 つの機能を，各レイヤ上の要素の位置をバネモデルにより決定し，レイヤ上の要素と関連のある他のレイヤ上の要素を連動して移動させる機構により実現する．本研究では，この位置決定機構を **CMSL（Connected Multilayer Spring Layout）**と呼ぶ．

本論文の構成は以下のとおりである．2 章で関連研究について述べる．3 章で CMSL の概要と音楽夢想の 4 つの機能について述べる．4 章で音楽夢想における CMSL の実装，音楽夢想の GUI について述べる．5 章で動作例，6 章で音楽夢想のユーザ評価，7 章で結論を述べる．

2. 関 連 研 究

ユーザが新たな音楽情報を発見することを目的とした可視化インタフェースとして，Musicream[3]，MusicRainbow[4] がある．Musicream では，現在注目する曲のアイコンをドラッグし，これを振り回すことで類似の音楽情報を持つ曲をくっつけ，関連性のある曲のグルーピングを行うインタフェースを提供している．これにより，直感的に関連のある曲の検索・発見を行うことができる．MusicRainbow は，曲の類似度に基づいて，円上にアーティストをマッピングすることで，ユーザの直感的な音楽ライブラリのブラウジングを実現している．これらの研究は，ユーザの直感的な操作により具体的なキーワードを用いることなく，曲の選択を行うことができる．しかし Musicream や MusicRainbow は，アーティストと雰囲気の情報をユーザが自由に組み合わせるといった「(1) 様々な音楽情報を組み合わせた情報の提示」の要求に関しては特に考慮されていない．また「(3) 音楽情報間の関連性の強弱の調整」の要求に関しても十分に満たされていない．

音楽ライブラリの可視化に重点を置いた研究として，van Gulik らの artist map[5),6)]，SIMT（Search Inside the Music）[7]，MeltingSound[8]，Island of music[9] がある．artist map は，アーティスト間の類似度に基づいて，アーティスト間の関係を 2 次元平面上でバネモデルに基づいて可視化を行うことができる．SIMT は曲を類似度に基づいて 3 次元空間上に可視化し，ユーザが聴いている曲と似た曲を含むアルバムをその曲の近くに配置することで，ユーザの注目に合わせて音楽情報を提示するシステムである．MeltingSound は，ユーザが音や画像を用いて直感的に曲を検索することができるシステムである．Island of Music は，類似した音楽情報を持つ曲をグループ化し，風景情報と対応させて可視化することにより，ユーザに対して曲の発見を支援することができるシステムである．これらの研究では，具体的な属性の提示を避けつつ曲の関係を空間上に表すことで，ユーザの曲への興味を空間表現に対する興味に置き換えている．これにより，ユーザは実際の曲をあまり意識

することなく，関連性のある曲を複数選択することを可能としている．しかし artist map，SIMT，MeltingSound，Island of music は，いずれも可視化する音楽情報の分類数の組合せに制限があり，「(1) 様々な音楽情報を組み合わせた情報の提示」の要求を満たすことは困難である．またこれらの研究では，ユーザの注目に合わせて情報構造全体の変化をとらえることは難しく，「(2) ユーザの操作に基づいた動的な情報提示」「(3) 音楽情報間の関連性の強弱の調整」の要求を十分に満たしているとはいえない．

音楽情報の可視化インタフェースには，自己組織化マップ（SOM: Self-Organization Map）を利用して曲をクラスタリングして可視化し，曲の発見を支援を目指す Neumayer らの PlaySOM[10]，Knees ら[11]，Pampalk ら[12] の研究がある．しかし，PlaySOM，Knees らの研究で提案されているインタフェースは，いずれも SOM により生成されたマップに基づいた可視化であるため「(2) ユーザの操作に基づいた動的な情報提示」「(3) 音楽情報間の関連性の強弱の調整」の要求を満たすことは難しい．

プレイリスト作成に関する研究として，梶らのプレイリスト推薦システム[13] がある．梶らの研究は，ユーザが音楽情報にアノテーション情報を付加することで，ユーザの嗜好と状況に合った曲のプレイリストの作成を実現している．しかし梶らの研究は，プレイリスト作成について「(4) 曲の直感的なグルーピング」を満たすことは考慮されていない．

グラフ構造を用いた可視化研究として，KeyGraph[14]，Vizster[15] がある．KeyGraph は単語間の関係をグラフ構造で可視化することにより，ユーザの発見支援を実現する研究である．Vizster は，コミュニティ内のユーザをノードで表し，各ノードの位置をユーザ間の関係の強さに基づいてばねモデルによって可視化してコミュニティ・ネットワーク内でのノードのグループの発見支援を実現するシステムである．KeyGraph や Vizster は単語やユーザといった情報要素間の関係を可視化する点では非常に優れているが，情報の分類が増えた場合の拡張性については考慮されておらず，「(1) 様々な音楽情報を組み合わせた情報の提示」の要求を満たすことは困難である．

CMSL のように複数のレイヤを組み合わせた可視化手法として，VisLink[16] がある．VisLink は，2 次元平面で可視化された画面を 1 つのレイヤとして扱い，それら複数のレイヤ上の各要素間の関係をリンクにより可視化する研究である．しかし VisLink では，リンク数が増えすぎた場合に，ユーザがレイヤ間の関係をすぐに理解することは難しくなるため，「(1) 様々な音楽情報を組み合わせた情報の提示」を十分に満たすことは難しい．

巨大なグラフ構造を可視化する際には，Clustered Graph[17] を用いる場合が多い．Clustered Graph では，代表的なノードのみを表示し，ユーザの操作に合わせてノードを展開して表示することで，ユーザの注目に合わせて情報を整理して表示することができる．しかし，Clustered Graph は，クラスタを閉じてしまった場合，クラスタに属するノードを混ぜ合わせたり，グルーピングしたりすることが難しくなるため，「(3) 音楽情報間の関連性の強弱の調整」「(4) 曲の直感的なグルーピング」を満たすためには適していない．

ユーザの注目した情報と関連のある情報を引き出すことに着目した可視化手法の研究として，Tominski らの研究[18] で提案されている Bring Neighbor Lens がある．Bring Neighbor Lens は，ユーザが選択したノードと関連があるノードを自動的に近くに移動させ，ユーザの注目に合わせて情報を引き出すことができる．しかし，ノードの移動によるグラフ構造全体の変化の可視化は考慮されておらず，「(2) ユーザの操作に基づいた動的な情報提示」を十分に満たすことは困難である．

以上のように，「(1) 様々な音楽情報を組み合わせた情報の提示」「(2) ユーザの操作に基づいた動的な情報提示」「(3) 音楽情報間の関連性の強弱の調整」「(4) 曲の直感的なグルーピング」の 4 つの要求を同時に満たすシステムは今まで提案されてこなかった．音楽夢想は，これらの 4 つの要求を同時に満たすシステムである．

3. 音 楽 夢 想

音楽夢想では，ユーザは自分の興味ある音楽情報の分類レイヤを「分類スタッキング」機能により重ね，「芋づるサーチ」機能により要素を引きずり出し，キャンバス上で次々と新しい曲の集合を出現させる．そしてユーザは「集合コンバージェンス」機能により情報の関連度合いの強弱を調整しつつ，「一筆書きグルーピング」機能により，任意のタイミングで出現した曲の集合をグルーピングしてプレイリストを作成する．この一連の流れにより，音楽夢想は 1 章の (1)〜(4) の要求を満たす直感的かつ統一的な曲の発見支援インタフェースを実現する．

「分類スタッキング」「芋づるサーチ」「集合コンバージェンス」「一筆書きグルーピング」機能は，CMSL という機構上で実現される．**図 1** に CMSL において雰囲気レイヤ，曲レイヤ，作曲家レイヤを重ねた例を示す．CMSL ではレイヤごとに要素を保持しており，各要素間の関係性を関連リンク（図 1 中の要素間の線分）として保持している．ユーザは，図 1 中の重ねたレイヤを上から下向きに見ることにより，2 次元平面上で要素を操作する．以降，「分類スタッキング」「芋づるサーチ」「集合コンバージェンス」「一筆書きグルーピング」機能と CMSL の関係について触れながら，各機能について説明する．

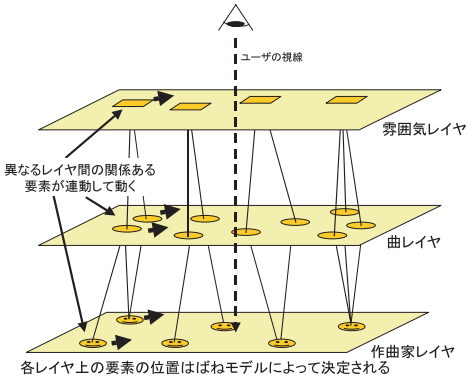

図 1　CMSL（Connected Multilayer Spring Layout）の構成．雰囲気レイヤ，曲レイヤ，作曲家レイヤを重ねた場合．各要素の位置はバネモデルで決定され，関連リンクで（図中の線分）で結ばれた要素が連動して動く

Fig. 1　Structure of CMSL (Connected Multilayer Spring Layout). Stack atmosphere layer, music layer, composer layer. Each elements' position are determined by spring-model and elements which connected to each other move relative to one another.

3.1　分類スタッキング

音楽夢想では，アーティスト，アルバム，雰囲気といった音楽情報をそれぞれ 1 つの分類レイヤとして扱い，分類レイヤを自由に組み合わせる「分類スタッキング」機能により，「(1) 様々な音楽情報を組み合わせた情報の提示」の要求を満たす．分類スタッキング機能は，CMSL においてレイヤを重ねる作業に対応する．分類スタッキング機能により，ユーザの興味に合わせて複数の音楽情報を，数の制限なしに，直感的に組み合わせることができる．

3.2　芋づるサーチ

音楽夢想では，ユーザが分類レイヤ上のある要素（A とする）を選択すると，その要素と関連のある要素を接近させ，要素 A と関連のある要素群を要素 A に接近させたまま一緒に移動させる．したがって，ユーザが複数の要素を引きずり出して固定していくと，引きずり出された要素の位置関係に基づいて，新しい要素の集合を出現させることができる．音楽夢想では，この「ユーザが選択した要素と関連する要素を次々に引き出し，新しい要素の集合を出現させる」仕組みを「芋づるサーチ」機能と呼び，これにより「(2) ユーザの操作に基づいた動的な情報提示」の要求を満たす．芋づるサーチ機能は，CMSL 上の同一レイヤ，異なるレイヤに属する要素を問わず，ある要素と関連リンクで結ばれた要素が連動して動く仕組みにより実現される．芋づるサーチ機能は，特に複数のレイヤを重ねたとき

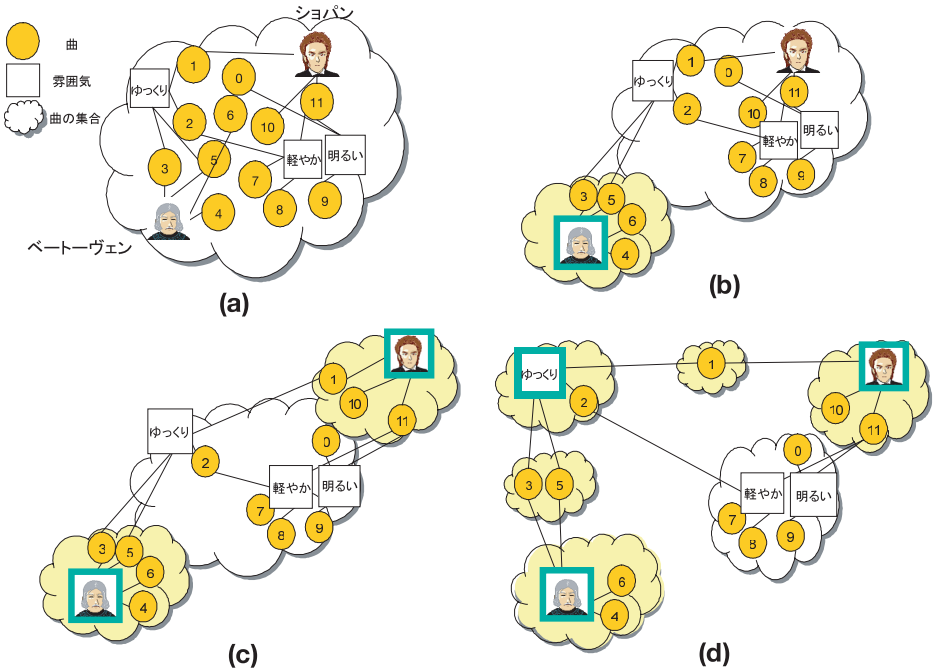

図 2　芋づるサーチにより曲の集合が出現する過程：(a) 曲，作曲家，雰囲気レイヤを重ねる．(b)「ベートーヴェン」の作曲した曲の集合を引きずり出す．(c)「ショパン」の作曲した曲の集合を引きずり出す．(d)「ゆっくり」な雰囲気の曲の集合を引きずり出す

Fig. 2　Process of emerging music sets: (a) music, composer and mood layers are Stacked. (b) set of music composed by Beethoven is extracted. (c) set of music composed by Chopin is extracted. (d) sets of music which have relations with "slow" on mood layer.

に，ユーザがある程度知識を持っている音楽情報（分類レイヤ）だけでなく，あまり知識のない音楽情報（分類レイヤ）について，様々な新しい要素の集合が出現させ，ユーザの気付かなかった音楽情報間の関係を提示することができる．**図 2** に芋づるサーチ機能による「(2) ユーザの操作に基づいた動的な情報提示」の実現例を示す．図 2 (a) では，分類スタッキングによりユーザが曲レイヤ・作曲家レイヤ・雰囲気レイヤが重ねた状態を表しており，大きな 1 つの曲の集合を形成している．ユーザが作曲家「ベートーヴェン」を選択し移動させると，「ベートーヴェン」の作曲した曲の集合を芋づるサーチにより引きずり出すことができる（図 2 (b)）．濃い縁取りの要素は，ユーザがその要素を検索キーとして利用するため

に固定したことを示している．続いてユーザが「ショパン」を選択し移動させると，「ショパン」の作曲した曲の集合を芋づるサーチ機能により引きずり出すことができる（図 2 (c)）．ここで，ユーザが「ゆっくり」な雰囲気の要素を選択し移動させると，「ベートーヴェン」「ショパン」それぞれの「ゆっくり」な曲の集合が現れる（図 2 (d)）．

3.3　集合コンバージェンス

音楽夢想では，要素の集合の広がり方を調整する「集合コンバージェンス」機能により，「(3) 音楽情報間の関連性の強弱の調整」の要求を満たす．集合コンバージェンス機能は，CMSL におけるバネモデルのバネの自然長を調整することにより実現される．**図 3** に集合コンバージェンス機能の動作例を示す．図 3 は，要素「ベートーヴェン」「ショパン」「ゆっくり」がユーザにより引きずり出されている状態を図示したものである．図 3 (a) 中の左下には，「ベートーヴェン」の作曲した曲の集合図 3 (a) 中の右上には，「ショパン」の作曲した曲の集合が提示されているが，全体的に要素の集合が広がっており，ユーザは特定の情報に偏らない，周辺の曲を含んだ曲の集合を得ることができる．図 3 (b) は，図 3 (a) の状態から集合コンバージェンス機能により集合が小さくまとめられ，「ベートーヴェン」の「ゆっくり」した曲，「ショパン」の「ゆっくり」した曲，「ゆっくり」だが作曲家情報のない曲，「ベートーヴェン」の「ゆっくり」以外の曲，「ショパン」の「ゆっくり」以外の曲，「軽やか」で「明るい」曲の集合が細かく分かれて提示される．

3.4　一筆書きグルーピング

音楽夢想では，ユーザは 2 次元平面上で要素を操作することになる．したがって，「分類スタッキング」「芋づるサーチ」「集合コンバージェンス」機能を用いた結果出現した曲の集合に対して，曲数やユーザが重視する情報を直感的に調整し，グルーピングするためには，重ねられたレイヤ上で，ユーザが図形を描いて曲の集合をグルーピングする方法が効果的である．音楽夢想ではこのグルーピングの方法を「一筆書きグルーピング」機能と呼ぶ．一筆書きグルーピング機能によりグルーピングされた曲をプレイリストとして利用することで，「(4) 曲の直感的なグルーピング」の要求を満たす．一筆書きグルーピング機能は，集合コンバージェンス機能と組み合わせることでより効果的に曲の集合のグルーピングを行うことができる（図 3）．図 3 (a) では，ユーザは「ベートーヴェン」の近くにある曲を直感的に囲むだけで，「ベートーヴェン」の曲を中心として，周辺の曲も含まれたプレイリストを作成することができる．一方，図 3 (b) では，小さい曲の集合に分かれており，ユーザは「ベートーヴェン」の作曲した曲のうち「ゆっくり」でない曲だけを明確に指定してグルーピングすることができる．

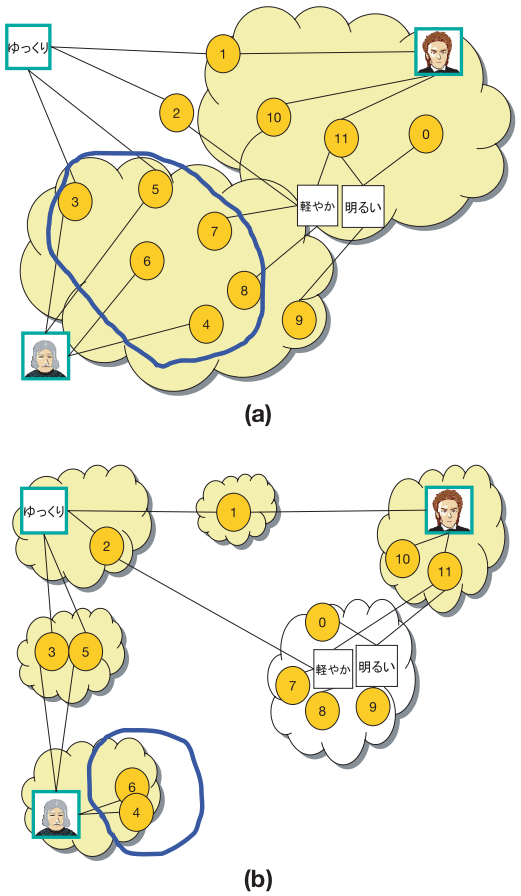

図 3　集合コンバージェンスと一筆書きグルーピングの組合せ：(a) 要素が全体的に広がっている状態で，ベートーヴェン付近の曲を「一筆書きグルーピング」により選択した場合．(b)「集合コンバージェンス」により要素間の距離を縮め，ベートーヴェン付近の曲を「一筆書きグルーピング」により選択した場合

Fig. 3　Combination of Element Set Convergence and Drawing Grouping: (a) The case of grouping music around the element “Beethoven” by “Drawing Grouping” when the elements are scattered. (b) The case of bringing music closer together by “Element Set Convergence”, and grouping music around the element “Beethoven” by “Drawing Grouping”.

4. 実　　装

4.1 音楽夢想における CMSL の実装

はじめに CMSL 上の要素の関連付けの方法について述べる．CMSL 上の要素は必ず 1 つの分類レイヤに属しており，分類レイヤでは音楽要素，作曲家，雰囲気などの分類によって各要素をまとめる．各要素は，他の要素と関連がある場合，関連リンクという形で関連する要素の情報を，要素ごとに保持する．CMSL の関連リンクは，同一レイヤ，異なるレイヤを問わず，すべての要素間で自由に結ぶことができる．ただし音楽夢想で利用する際には，データの性質上，異なるレイヤに属する要素間でのみ関連リンクを結んでいる．たとえば，作曲家「ベートーヴェン」が「交響曲第 5 番」を作曲したという関係を表現する場合，作曲家レイヤ上の要素「ベートーヴェン」と音楽レイヤ上の要素「交響曲第 5 番」の間に関連リンクを結ぶ．また「交響曲第 5 番」の雰囲気が「荘厳」であるという関係を表現する場合は，音楽レイヤ上の要素「交響曲第 5 番」と雰囲気レイヤ上の「荘厳な曲」と「交響曲第 5 番」が関連リンクで結ばれる．

続いて CMSL で用いたバネモデルについて述べる．CMSL はバネモデルを用いたグラフのレイアウトアルゴリズムである Eades の “Spring Embedder” [19] を参考にして設計した．“Spring Embedder” は，ノード間の接続性を考慮し，接続されているノード間には式 (1) に示す引力 F_s，接続されていないノード間では式 (2) に示す斥力 F_r をはたらかせる．ただし c_1，c_2，c_3 は定数，r はノード間の距離である．

$$F_s = c_1 \times \log(r/c_2) \tag{1}$$

$$F_r = c_3/\sqrt{r} \tag{2}$$

これにより，ノード間の重なりを避けつつ，ノード間の接続関係を位置を用いて分かりやすくユーザに提示することに成功している．

CMSL は，Spring Embedder のノード間の接続関係に基づいて力を計算し要素の位置を決定する方法を基にした．CMSL では，特に要素間の距離を調整しやすくするため，実在のバネの力学モデルとした，式 (3) に示す引力 F_s，および式 (4) に示す斥力 F_r を用いる．ただし d はバネの自然長，K_s，K_r はバネ定数にそれぞれ相当する変数である．

$$F_s = \begin{cases} K_s|r-d| & (d \geq r) \\ 0 & (d < r) \end{cases} \tag{3}$$

$$F_r = K_r|r-d| \tag{4}$$

同一レイヤ上の各要素間に式 (4) に示される斥力をはたらかせることにより，各レイヤにおいて要素どうしがある一定の距離を保つようにする．一方，関連リンクの存在するお互いの要素は式 (3) に示される引力によって近くに配置する．CMSL では，d を要素間の理想距離を調整するための，バネの自然長に相当する変数として扱う．また K_s，K_r の値を要素間の力の強さを調整するための，バネ定数に相当する変数として扱う．そして d および K_s，K_r の値を後述のように (A)，(B)，(C) それぞれ個別に設定する．

次に，具体的な CMSL の位置更新の手順について述べる．CMSL では異なるレイヤ間の要素の重なりは考慮せず，同一レイヤ内でのみ要素間に斥力をはたらかせる．全体の要素数が多い場合，すべての要素について斥力をはたらかせると，各要素が均一に散らばってしまいユーザが要素間の関連性を把握することが難しくなるためである．CMSL のこの仕組みによって，ユーザは作曲家や雰囲気など現在選択している要素についての距離に集中することができ，ユーザの視認性・操作性を両立させることができる．

CMSL では各要素ごとに位置更新の計算を行う．CMSL における 1 つの要素の位置更新の流れを以下 (A)〜(D) に示す．

(A) 同一レイヤ上の要素どうしが重ならないようにするための，位置更新対象要素が同一レイヤ上の他の要素に対して理想距離内に入った場合に発生する斥力による速度ベクトル $\boldsymbol{v}^A_{u,t}$ の計算

(B)「芋づるサーチ」「集合コンバージェンス」機能を実現するための，位置更新対象要素が関連リンクで結ばれた要素から受ける力による速度ベクトル $\boldsymbol{v}^B_{u,t}$ の計算

(C) 要素が広がりすぎないようにするために設定した固定点 C から，位置更新対象要素が受けるバネの力により生じる速度ベクトル $\boldsymbol{v}^C_{u,t}$ の計算

(D) 上記 (A)〜(C) の速度ベクトルを合計し，要素の移動速度が速くなりすぎないようにするために，最高速度を制限して位置を更新

ただし位置更新対象の要素が固定されている場合とユーザにより選択されている場合は，ユーザが要素を自由に動かせるようにするために，要素の位置更新は行わない．

CMSL の位置更新を数式を用いて表すため，必要な変数を以下のように定義する．

- t：要素の位置更新を行う時刻
- E_i：CMSL 上の要素番号 i の要素（所属レイヤは問わない）
- E^l_i：レイヤ l 上の要素
- u：位置更新対象の要素番号
- $\boldsymbol{e}_{i,t}$：時刻 t における要素 E_i の位置ベクトル

- $\boldsymbol{r}_{i,j,t}$：時刻 t における E_i から E_j へのベクトル

以下 E_u^l の時刻 $t+1$ における位置 $\boldsymbol{e}_{u,t+1}$ を求めるための式を (A)〜(D) の流れに沿って導出する．

(A) での速度ベクトル $\boldsymbol{v}_{u,t}^A$ は式 (5) により決定する．

$$\boldsymbol{v}_{u,t}^A = \sum_{\{i|E_i^l \in A_u^l\}} K_{u,i}^A \left(|\boldsymbol{r}_{u,i,t}| - d_{u,i}^A\right) \frac{\boldsymbol{r}_{u,i,t}}{|\boldsymbol{r}_{u,i,t}|} \tag{5}$$

ただし，$d_{u,i}^A$ は E_u^l と E_i^l 間の理想距離，集合 A_u^l は，E_u^l と同一レイヤ上に属する要素のうち要素間の距離が理想距離よりも近い要素の集合，すなわち，$A_u^l = \{E_i^l | i \neq u, |\boldsymbol{r}_{u,i,t}| \leq d_{u,i}^A\}$ である．また $K_{i,j}^A$ は要素間の斥力の強さを調整するための変数である．

(B) での速度ベクトル $\boldsymbol{v}_{u,t}^B$ は式 (6) により決定する．

$$\boldsymbol{v}_{u,t}^B = \sum_{\{i|E_i \in B_u\}} K_{u,i}^B \left(|\boldsymbol{r}_{u,i,t}| - d_{u,i}^B\right) \frac{\boldsymbol{r}_{u,i,t}}{|\boldsymbol{r}_{u,i,t}|} \tag{6}$$

ただし，B_u は E_u^l と関連リンクで結ばれた要素（レイヤ間，および同一レイヤ上の要素も含む）の集合，$d_{u,i}^B$ は E_u^l と E_i 間の理想距離，$K_{u,i}^B$ は要素間の力を調整する変数である．

(C) での速度ベクトル $\boldsymbol{v}_{u,t}^C$ は式 (7) により決定する．

$$\boldsymbol{v}_{u,t}^C = -K^C \left(|\boldsymbol{e}_{u,t}| - d^C\right) \frac{\boldsymbol{e}_{u,t}}{|\boldsymbol{e}_{u,t}|} \tag{7}$$

ただし d^C は要素 E_u^l と固定点 C との理想距離，K^C は固定点 C と要素 E_u^l の間の力を調整する変数である．

(D) では，E_u^l の位置ベクトル $\boldsymbol{e}_{u,t}$ に (A)〜(C) で求めた速度ベクトルを合計する．この際に，合計された速度ベクトルの大きさがある閾値 V 以上であれば，V になるように大きさを調整する．

以上から CMSL における要素 E_u^l の位置は，$\boldsymbol{v}_{u,t} = \boldsymbol{v}_{u,t}^A + \boldsymbol{v}_{u,t}^B + \boldsymbol{v}_{u,t}^C$ とすると，式 (8) により決定される．

$$\boldsymbol{e}_{u,t+1} = \boldsymbol{e}_{u,t} + \begin{cases} \boldsymbol{v}_{u,t} & (|\boldsymbol{v}_{u,t}| \leq V) \\ \dfrac{\boldsymbol{v}_{u,t}}{|\boldsymbol{v}_{u,t}|} V & (|\boldsymbol{v}_{u,t}| > V) \end{cases} \tag{8}$$

以上 (A)〜(D) の位置更新を，すべての要素について行い，CMSL 上での要素の位置更新を実現する．

表 1　初期状態における変数の関係
Table 1　Relation between values in initial state.

ステップ	バネの強さの調整変数	理想距離
(A)	$K^A = K_{\text{init}}$	$d^A = d_{\text{init}}$
(B)	$K^B = K_{\text{init}} \times 20$	$d^B = d_{\text{init}}/2$
(C)	$K^C = K_{\text{init}}/2$	$d^C = d_{\text{init}}$

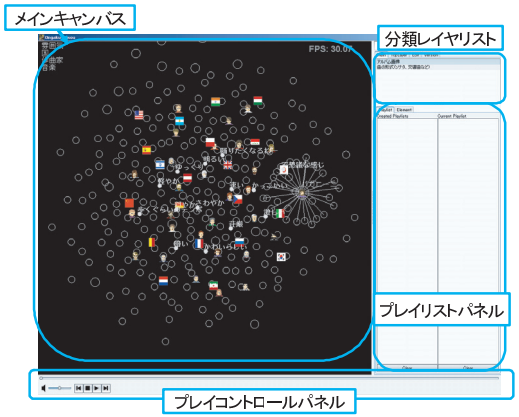

図 4　音楽夢想の GUI
Fig. 4　GUI of Ongaku Musou.

なお，初期状態における変数 K^A，K^B，K^C，d^A，d^B，d^C 間の値の関係を**表 1** に示す．関連リンクで結ばれた要素を初期状態で強く引き寄せるために，K^B の値を他の 2 つの場合よりも大きく，d^B の値を他の 2 つの場合よりも小さくしている．音楽夢想では，K_{init}，d_{init} を，それぞれ 0.01，1,000 とした．これらの値は，各レイヤに約 100 個の要素を保持しているレイヤを数レイヤ重ねたときに，要素が振動せず，ユーザが適切に操作できる値を選んだ．

E_u^l がユーザによって選択または固定されている場合，この要素を目立たせるために，$d_{u,i}^A$，$K_{u,i}^A$ を大きい値に設定して斥力を強くする．これにより E_i^l を E_u^l から遠くに弾き飛ばし，ユーザの選択した要素 E_i^l が目立つようにする．また，芋づるサーチでは，要素 E_u^l が選択または固定されている場合に，$d_{u,i}^B$ を小さくするとともに $K_{u,i}^B$ の値を大きくする．これにより，要素 E_i（$E_i \in B_u$）が選択された要素の近くに強く引き寄せて実現する．また集合

表 2　メインキャンバスの操作方法
Table 2　Manner of operation on Main Canvas.

操作	実行方法
レイヤ回転	キーボードのスペースバーを押す
画面全体の拡大・縮小	ホイールを回転
画面全の移動	要素を選択しないで左ドラッグ
要素の選択	マウスポインタを要素に合わせる
要素の固定	要素を左クリック
要素の固定解除	要素を右クリック
要素の移動	固定した要素を左ドラッグ

コンバージェンスは，バネの自然長 d^A，d^B，d^C の値を，表 1 の (A)〜(C) 間の関係式を満たしながら変化させることによって実現される．

4.2　音楽夢想の GUI

音楽夢想の GUI の構成を**図 4** に示す．音楽夢想の GUI は，分類レイヤを重ね，レイヤに属する要素が可視化される「メインキャンバス」（操作方法は**表 2**），曲の再生・停止・シークを行うプレイコントロールパネル，メインキャンバス上に重ねる分類レイヤを保持している「分類レイヤリスト」，メインキャンバス上で一筆書きグルーピングにより作成されたプレイリストを管理するための「プレイリストパネル」からなる．分類レイヤリストからメインキャンバス上へ分類レイヤをドラッグすることで，ユーザはメインキャンバス上でレイヤを組み合わせることができる．組み合わせたレイヤ名はメインキャンバスの左上に表示される．音楽夢想の実装には Java SDK 6 を用いた．

5. 動　作　例

本章では，音楽夢想により 1 章で述べた 4 つの要求がいかにして満たされ，音楽情報の関連性に基づいた発見支援が実現されるかを示す．本章で示す動作例では，クラシックの曲データに基づいた**表 3** のデータを利用した．

まず「(1) 様々な音楽情報を組み合わせた情報の提示」の例について示す．ユーザは，特定の曲を聴きたいわけではなかったので，曲を選択する際のヒントとして「作曲家」「曲の雰囲気」「曲の形式」の情報を組み合わせて曲を探したいと思った．そこでユーザは，分類スタッキング機能により，メインキャンバスに作曲家レイヤ，音楽レイヤ，雰囲気レイヤ，曲の形式レイヤを重ねた．このとき，音楽要素は各レイヤ上の要素の各関連性に基づいて自動的にクラスタ化される．このようにして，音楽夢想では音楽情報に関する複数の関連性を，同時に扱うことができる．

表 3　動作例で用いたデータ
Table 3　Data used in operation example.

レイヤ名	表示方法	要素数	主に関係のあるレイヤ
アルバム	画像	64	音楽
曲の形式	音楽	12	音楽
雰囲気	文字列	14	音楽
音楽	円	287	国以外の分類レイヤ
作曲家	画像	29	国，音楽
国	画像	21	作曲家

続いて「(2) ユーザの操作に基づいた動的な情報提示」の例について示す．ユーザは，ランダムに何人かの作曲家を選択して曲を聴くために，芋づるサーチ機能により何人かの作曲家の要素を引きずり出した．ユーザは引きずり出した作曲家の中から偶然目にとまった作曲家「ショパン」に興味を持ち（**図 5** (a)），「ショパン」の作曲した曲を 1 曲聴いた（図 5 (b)）．ユーザはこの曲が気に入り，この曲のように「速い」曲が聴きたいと思った．そこでユーザは，この曲の近くにあった雰囲気レイヤの要素「速い」を選択して移動させた（図 5 (c)）．するとショパンの「速い」曲の集合だけでなく，ユーザの予期していなかったユーザが最初に引き出しておいたショパン以外の作曲家の「速い」曲の集合が出現した（図 5 (d)）．このように，音楽夢想では，ユーザの操作に応じて動的に音楽要素のクラスタの様子が変化し，ユーザに提示される．

次に「(3) 音楽情報間の関連性の強弱の調整」「(4) 曲の直感的なグルーピング」の例について示す．ユーザは曲の集合をはっきりと切り分けるために，集合コンバージェンス機能で集合を小さくまとめ（**図 6** (e)），出現した曲の集合の中から，作曲家の「チャイコフスキー」の「速い」曲を選択して 1 曲聴いた（図 6 (f)）．ユーザはこの曲が気に入ったので，この曲を含むようにプレイリストを作成しようと思った．しかし「チャイコフスキー」の「速い」曲の集合だけでは曲数が少なかったので，その周辺の曲も含めたプレイリストを作成して聴くことにした．ユーザは集合コンバージェンス機能を使って，要素の集合を広げて周囲の集合と混ざるようにした．そして一筆書きグルーピング機能により，「チャイコフスキー」の「速い」曲の集合を中心として，周辺の曲も含むように囲んでプレイリストを作成した (図 6 (g))．最終的にユーザは，「チャイコフスキー」の「速い」曲と同時に，ユーザが最初に引き出しておいた作曲家（パガニーニ）の曲が含まれたプレイリストを作成することができた（図 6 (h)）．このように，音楽夢想では，関連性の強弱を調整しつつ，直観的に複数の

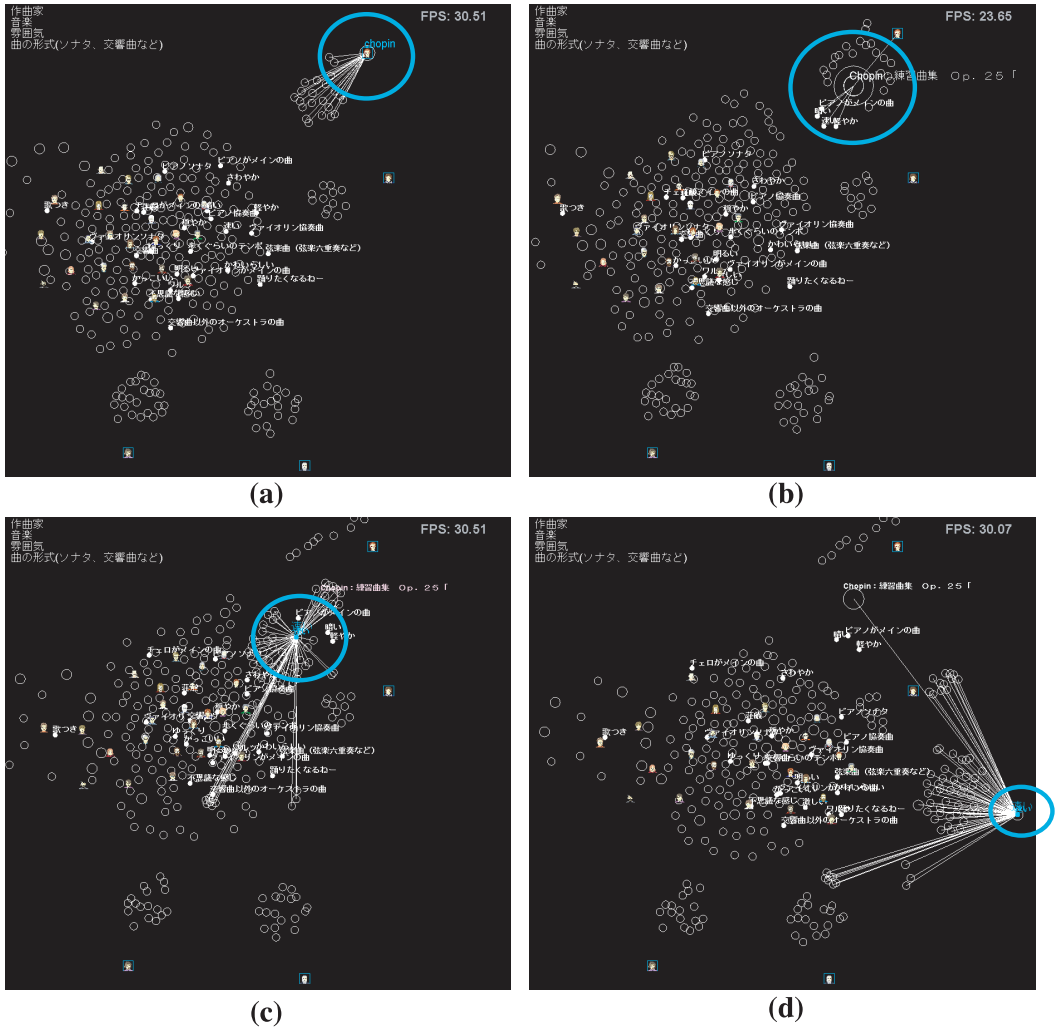

図 5　(a) ユーザが何人かの作曲家を引き出し，その中から「ショパン」を選択する．(b) ユーザがショパンの作曲した曲の集合の中の 1 曲を聴く．(c) ユーザは自分が聴いている曲に関連した雰囲気のうち「速い」を選択する．(d) ユーザが「速い」雰囲気の要素を引きずり出すと，ユーザが予期していなかった複数の曲の集合が現れる

Fig. 5　(a) User drags out some composers and selects Chopin from them. (b) User listens to music in set of music composed by Chopin. (c) User selects element "fast" related with music to which user is listening on mood layer. (d) When user drags out element "fast", unanticipated music sets are emerged.

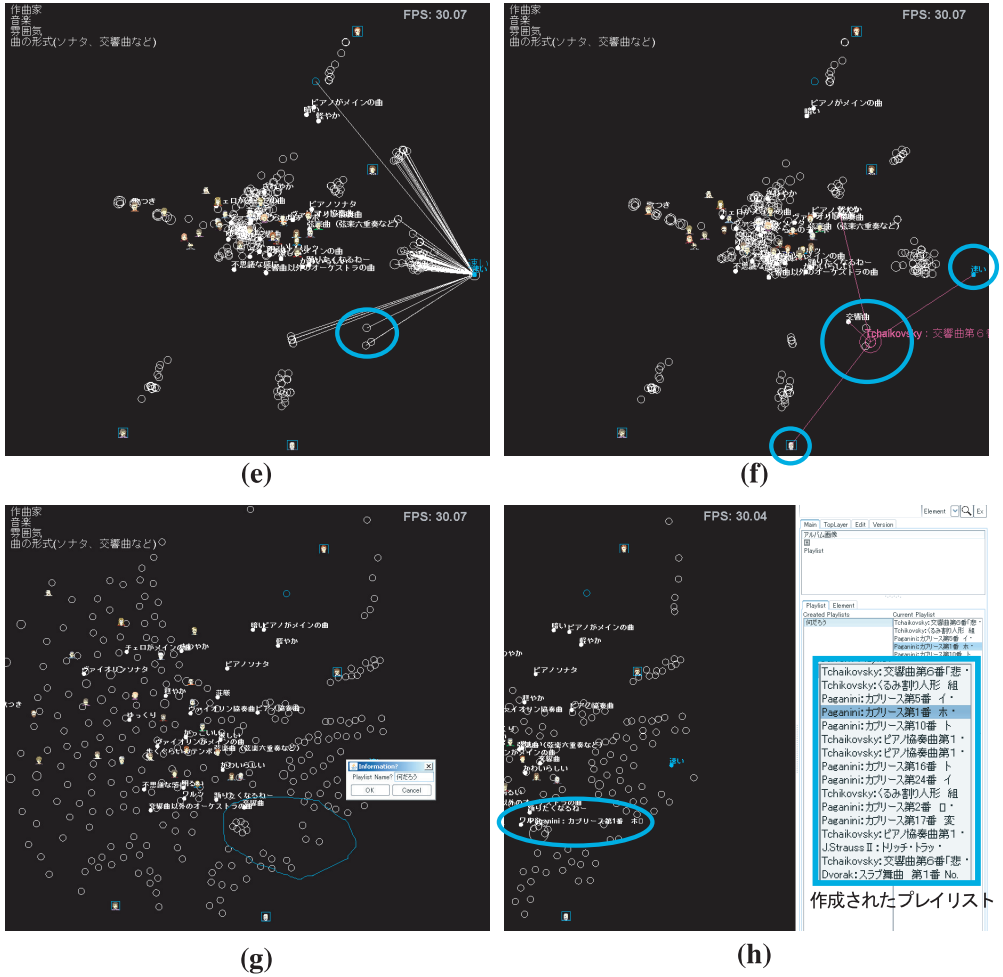

図 6　(e) ユーザは曲の集合をはっきりさせるため集合コンバージェンス機能により，要素の集合を小さくまとめる．(f) ユーザはチャイコフスキーの速い曲を含む曲の集合の中の 1 曲を聴く．(g) 集合コンバージェンス機能により要素の集合を広げ，一筆書きグルーピングによりプレイリストを作成する．(h) プレイリストには，チャイコフスキーの速い曲だけでなく，他の作曲家の速い曲も含まれている

Fig. 6　(e) User changes size of element sets smaller by Element Set Convergence function to clear up boundary of element sets. (f) User listens to one music in set of Tchaikovsky's fast music. (g) User spreads distance between elements by Element Set Convergence function, creates playlist by Drawing Grouping function. (h) Playlist contains not only Tchaikovsky's fast music but also other composers' fast music.

音楽要素をグルーピングすることができる．

6. ユーザ評価

音楽夢想の各機能が有効に動作し，音楽夢想が音楽情報の関連性に基づいた検索・発見支援を実現することを確認するために，2 つのイベント会場（イベント 1，2 とする）でアンケートによるユーザ評価を行った．ユーザ評価における設定を**表 4** に，評価項目を**表 5**，アンケートの内容を**表 6** および **表 7** に示す．ユーザ評価には，動作例で用いた表 3 のデータを用いた．なお，以下の各ユーザ評価において，年齢の差異による有意な差は特に認められなかった．

6.1 各機能の 5 段階評価

各機能の有用性を検証するため，アンケートにおいて，各機能について被験者に 5 段階で評価をしてもらった．アンケートには表 6 に示す文面を用いた．評価は 1 を最低，5 を最高とし，4 と 5 と回答した人を「この機能を有用だと評価した人」と見なした．各機能の 5 段階評価の結果を**図 7** に示す．

図 7 より，分類スタッキングは 72%，芋づるサーチは 89%，一筆書きグルーピングは 83%，集合コンバージェンスは 83%の人が有用だと評価しており，各機能が高く評価されていることが確認された．

6.2 既存のソフトウェアとの比較

音楽夢想での関連性に基づいた曲の検索の有効性について，既存のソフトウェアとの比較を行った．音楽夢想の「(A1) 分類スタッキングと芋づるサーチによる曲の検索」，「(A2) 集合コンバージェンスと一筆書きグルーピングによるプレイリスト作成」と既存のソフトウェアの「(B1) キーワードによる曲の検索」「(B2) 一曲ずつ選択してプレイリストを作成する方法」について，表 7 に示す文面を用いてその有用さを 5 段階評価で被験者に評価してもらった．

ここで，既存のソフトウェアは，アンケート内で被験者に普段使っているソフトウェアを尋ね，回答されたソフトウェアを既存ソフトウェアとした．そして被験者には，これらのソフトウェアについて，各々設問に答えてもらった．既存ソフトウェアとしては，Windows Media Player，iTunes，フォルダを直接開いて聴く，その他のソフトウェアを利用するという回答が得られた．特に Windows Media Player と iTunes が回答の約 7 割を占めたが，既存ソフトウェアの種類と本評価との間には特に相関は見られなかった．

「関連情報を用いた曲の検索：(A1) と (B1)」「直感的なプレイリストの作成：(A2) と (B2)」の組について 5 段階評価の平均値に差が出るかどうかを t 検定により検定した．それぞれ有意水準 5%で 5 段階評価の平均値に有意差があることを確認した（**表 8**，**表 9**）．

表 4　ユーザ評価の設定
Table 4　Setup of impression evaluation.

データ	動作例で用いた表 3 のデータ		
被験者	年齢	イベント 1	イベント 2
	10 代未満	1	0
	10 代	13	0
	20 代	14	12
	30 代	4	7
	40 代	0	4
	50 代	3	1
	60 代	1	0
	計	36	24
手順	(1) 音楽夢想の概要を説明する．(2) コンピュータの前に座ってもらい，ソフトウェアの操作を説明する．(3) 5 分ほど自由にソフトウェアに触ってもらう．被験者から操作途中で質問があれば適宜答える．(4) アンケートに回答してもらう．		
その他	評価で用いたマシンにはスピーカを接続し，被験者が聴きたい曲を実際に聴くことができるようにした．		

したがって，「関連情報を用いた曲の検索」に関して，音楽夢想による「(A1) 分類スタッキングと芋づるサーチによる曲の検索」が既存のソフトウェアよりも優れていることが確認された．また「直感的なプレイリストの作成」に関しても，「(A2) 集合コンバージェンスと一筆書きグルーピングによるプレイリスト作成」が，既存のソフトウェアよりも優れていることが確認された．

6.3 曲の発見に関する評価

音楽夢想によって「音楽情報の関連性に基づいた発見支援」が実現できているかどうかを調べるため，アンケートには曲の発見がなされたかどうかの設問を儲けた．被験者に対して特に「曲を発見してほしい」とは言及せず，音楽夢想に自由に触ってもらった後，「いくつかの要素を自由に固定していったとき，あなたの予期していない曲の集まりが現れてきましたか（発見できましたか）？」という文面を用いて調査した．被験者には，「発見できた」「発見できなかった」の 2 択で回答してもらった．曲の発見に関する評価の結果を**図 8** に示す．

今回の評価実験では被験者が音楽夢想を操作する時間が限られていたが，図 8 から，約 7 割の人が曲の発見を実現できたと回答したことが分かる．曲を発見できたと答えた被験者

表 5　ユーザ評価における評価項目
Table 5　Evaluation items in impression measurement.

節	目的	方法	対象者
6.1	分類スタッキング，芋づるサーチ，集合コンバージェンス，一筆書きグルーピングの有用性を調べる．	各機能を被験者に 5 段階評価（1 を最低，5 を最高）で評価してもらった．またそれぞれの機能について，4 と 5 と回答した被験者を「この機能を有用だと評価した」と見なし，その割合を評価した．	イベント 1 の被験者全員．
6.2	音楽夢想と既存のソフトウェアを比較し，音楽夢想による関連性に基づいた曲の検索の仕方の有効性を調べる．	音楽夢想の「(A1) 分類スタッキングと芋づるサーチによる音楽の検索」「(A2) 集合コンバージェンスと一筆書きグルーピングによるプレイリスト作成」と，既存のソフトウェアの「(B1) キーワードによる音楽の検索」「(B2) 一曲ずつ選択してプレイリストを作成する方法」について，被験者に有用さを 5 段階評価で評価してもらった．「関連情報を用いた曲の検索：(A1) と (B1)」「直感的なプレイリストの作成：(A2) と (B2)」の組について 5 段階評価の平均値に差が出るかどうかを t 検定により検定した．	イベント 2 の被験者のうち既存のソフトウェアを使って曲を聴くと回答した 22 人．
6.3	「音楽情報の関連性に基づいた発見支援」が実現できるかどうかを調べる．	被験者には特に曲を発見して欲しいとは言及せず，音楽夢想を操作してもらい，被験者の行動を観察した．また，操作の後，アンケートによって被験者が予期していない曲が現れたかどうかを「発見できた」「発見できなかった」の 2 択で回答してもらい，その割合を評価した．	イベント 2 の被験者全員．

表 6　各機能の 5 段階評価（6.1 節）のアンケート文面
Table 6　Contents of questionnaire of five grade evaluation.

機能	文面
分類スタッキング	音楽夢想では，複数のレイヤを重ねて表示していますが，この方法は便利だ（見やすい）と思いましたか？
芋づるサーチ	音楽夢想では，あるレイヤの要素を引っ張ると他のレイヤ上の関連のある要素も連動して動くことで，要素間の関係を知ることができます．この機能は便利だと思いましたか？
集合コンバージェンス	音楽夢想では，要素間の距離を変化させることで要素をまとめることができますが，この機能は便利だと思いましたか？
一筆書きグルーピング	音楽夢想ではキャンバス上で絵を描くようにして，プレイリストを作成することができますが，この機能は便利だと思いましたか？

表 7　既存ソフトウェアとの比較評価（6.2 節）のアンケート文面
Table 7　Contents of questionnaire for comparison with existing software.

比較内容	文面
(A1) 分類スタッキングと芋づるサーチによる検索 (B1) キーワードによる検索	「ベートーヴェン（作曲家）の曲を探し，見つかった曲から明るい雰囲気の曲を探し出す」といった曲の関連情報に基づいた検索をする際，どの程度やりやすいと感じますか？
(A2) 集合コンバージェンスと一筆書きグルーピングによるプレイリスト作成 (B2) 一曲ずつ選択してプレイリストを作成	曲を選んでプレイリストを作成する際，どのくらい手間がかかる（簡単）と感じますか？

の音楽夢想の操作のパターンとしては以下のようなものが見られた．

- 「国，雰囲気，作曲家，音楽レイヤを重ねる」→「雰囲気レイヤから雰囲気の要素をいくつか引きずり出す」→「国または作曲家レイヤから興味のある要素を引きずり出した際に被験者の興味のある雰囲気の曲で，被験者が意図していなかった国やアーティストに属する曲の集合が現れ，曲を発見する」
- 「知っている曲 A を選択」→「曲 A の作曲家 B が接近」→「作曲家 B を選択」→「作曲家 B の作曲した曲が接近」→「あらかじめ引き出されていた雰囲気や国の要素の影

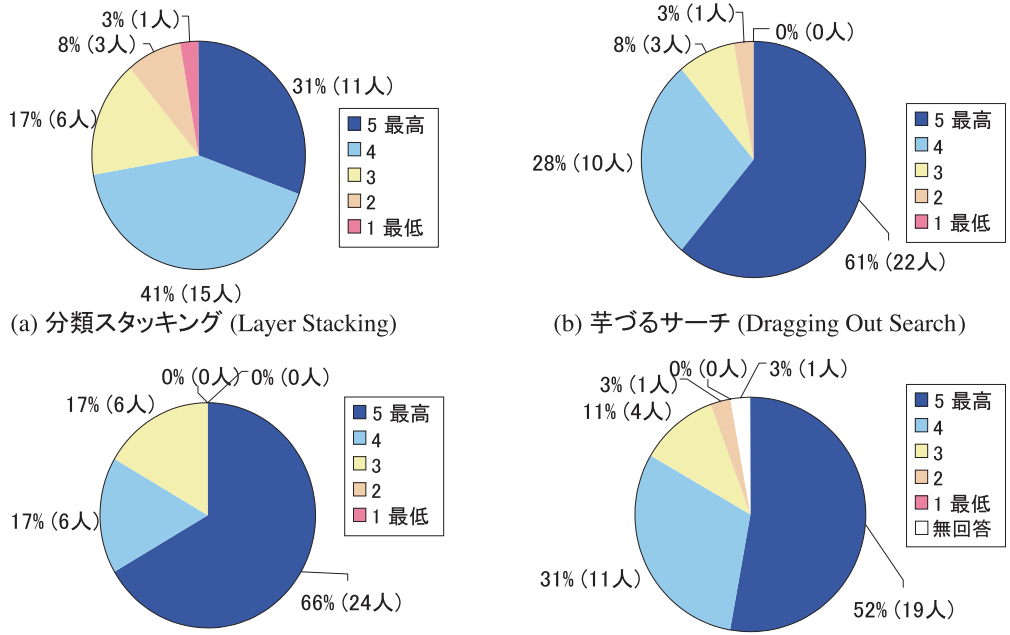

図 7　音楽夢想の機能に関するユーザ評価の結果
Fig. 7　Results of user evaluation about Ongaku Musou's functions.

表 8　曲の検索に関する 5 段階評価の比較
Table 8　Comparison test of search music.

機能	平均	有意確率
(A1) 音楽夢想	4.22	0.023
(B1) 既存方法	3.45	

表 9　プレイリスト作成に関する 5 段階評価の比較
Table 9　Comparison test of creating playlists.

機能	平均	有意確率
(A2) 音楽夢想	4.05	0.00037
(B2) 既存方法	2.72	

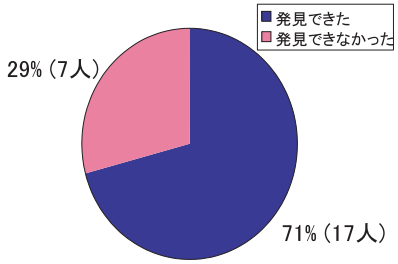

図 8　曲の発見に関する評価
Fig. 8　Evaluation of discovering music.

響を受けて，作曲家 B の曲がいくつかの部分集合に分かれて出現する」→「作曲家 B の今まで気付かなかった雰囲気の曲を発見する」

また，被験者は特に明確な目的も持たず，国や雰囲気の要素を選択してどんなアーティストや曲の集合が現れるのかを期待して操作するという場面も多数見られた．このように，音楽夢想によって曲の発見支援が行われる様子や，発見を促す様子が多く確認された．

逆に曲を発見できなかった被験者から，自由回答で得た意見として「音楽夢想をクラシック以外のジャンルで使いたい」という意見が多数寄せられた．これらの被験者は，クラシックに対する興味があまりなかったため，曲に付随した国や作曲家を選ぶという手順で興味のある要素を選ぶことが難しく，曲を発見できる割合が低下したと思われる．曲のジャンルが被験者にとって興味のあるものであれば，新たな曲の発見が行われた可能性が高いと考えられる．

これらの結果により，音楽情報の関連性に基づいて類似した属性を持つ曲を近くに配置する音楽夢想の機能が，明確な検索要求を持たないユーザの「音楽情報の関連性に基づいた発見支援」に関して有効であるといえる．

7. 結　　論

本研究では，インタラクティブに音楽を検索・発見できるソフトウェア「音楽夢想」を提案した．ユーザの曖昧な検索の要求を実現するためには，音楽情報の関連性に基づいた発見支援が必要である．音楽夢想は，既存研究では十分に満たされていなかった，音楽情報の関連性に基づいた発見支援のために必要な 4 つの要求である，(1) 様々な音楽情報を組み合わせた情報の提示，(2) ユーザの操作に基づいた動的な情報提示，(3) 音楽情報間の関連性の強弱の調整，(4) 曲の直感的なグルーピングを「分類スタッキング」「芋づるサーチ」「集合コンバージェンス」「一筆書きグルーピング」機能により実現した．音楽夢想の動作例，ユーザ評価から，音楽夢想の各機能が有効に動作し，(1)〜(4) の要求を満たすことが示された．

今回は筆者らが独自に用意した音楽・アーティスト情報を用いたが，今後は，オンラインデータベースの情報を利用して曲名，アーティスト，アルバム，ジャンルの情報を取得し，レイヤや要素のデータを構築する予定である．また今後は音楽夢想で提案したインタフェースを，動画配信サイトやオンラインの百科事典といった様々な分野で利用することを考えている．

参 考 文 献

1) Microsoft: Windows Media Player. http://www.microsoft.com/windows/windowsmedia/
2) Apple: iTunes. http://www.apple.com/itunes/
3) Goto, M. and Goto, T.: Musicream: New Muisc Playback Interface For Streaming, Sticking, Sorting and Recalling Musical Pieces, *Proc. 6th International Conference on Music Information Retrieval* (*ISMIR 2005*), pp.404–411 (2005).
4) Pampalk, E. and Goto, M.: MusicRainbow: A New User Interface to Discover Artists Using Audio-based Similarity and Web-based Labeling, *Proc. 6th International Conference on Music Information Retrieval* (*ISMIR 2006*) (2006).
5) van Gulik, R., Vignoli, F. and van de Wetering, H.: Mapping Music in the Palm of Your Hand, Explore And Discover Your Collection, *Proc. 5th International Conference on Music Information Retrieval* (*ISMIR 2004*), pp.409–414 (2004).
6) van Gulik, R. and Vignoli, F.: Visual Playlist Generation on the Artist Map, *Proc. 6th International Conference on Music Information Retrieval* (*ISMIR 2005*), pp.520–523 (2005).
7) Lamere, P. and Eck, D.: USING 3D VISUALIZATIONS TO EXPLORE AND DISCOVER MUSIC, *Proc. 8th International Conference on Music Information Retrieval* (*ISMIR 2007*), pp.173–174 (2007).
8) 神原啓介，安村通晃：MeltingSound：なめらかなオーディオブラウジング，HIS2003 論文集，pp.817–820 (2003).
9) Pampalk, E.: Islands of music: Analysis, organization and visualization of music archives, Master's thesis, Vienna University of Technology (2001).
10) Neumayer, R., Dittenbach, M. and Rauber, A.: PlaySOM and PocketSOMPlayer: Alternative Interfaces to Large Music Collections, *Proc. 6th International Conference on Music Information Retrieval* (*ISMIR 2005*), pp.618–623 (2005).
11) Knees, P., Schedl, M., Pohle, T. and Widmer, G.: An Innovative Three-Dimensional User Interface for Exploring Music Collections Enriched with Meta-Information from the Web, MM'06, pp.513–516 (2006).
12) Pampalk, E., Rauber, A. and Merki, D.: Content-based Organization and Visualization of Music Archives, *Proc. 10th ACM international conference on Multimedia*, pp.570–579 (2002).
13) 梶　克彦，平田圭二，長尾　確：コミュニケーションメディアとしてのプレイリストを目指して For Realization of Playlist-Mediated Communication，情報科学技術フォーラム（FIT）(2005).
14) Ohsawa, Y., Benson, N.E. and Yachida, M.: KeyGraph: Automatic Indexing by Co-occurrence Graph based on Building Construction Metaphor, *Proc. Advances*

in Digital Libraries Conference, pp.12–18 (1998).
15) Heer, J. and Boyd, D.: Vizster: Visualizing Online Social Networks, *IEEE Symposium on Information Visualization 2005* (2005).
16) Collins, C. and Carpendale, S.: VisLink: Revealing Relationships Amongst Visualizations, *Trans. Visualization and Computer Graphics*, Vol.13, No.6, pp.1192–1199 (2007).
17) Li, W., Hong, S.-H. and Eades, P.: A Framework for Visualising Large Graphs, *Proc. 9th International Conference on Information Visualisation* (*IV'05*), pp.528–535 (2005).
18) Tominski, C., Abello, J., van Ham, F. and Schumann, H.: Fisheye Tree Views and Lenses for Graph Visualization, *Proc. Information Visualization* (*IV'06*), pp.17–24 (2006).
19) Eades, P.A.: A heuristic for graph drawing, *Congressus Numerantium*, Vol.42, pp.149–160 (1984).

(平成 20 年 3 月 24 日受付)
(平成 20 年 9 月 10 日採録)

古山　真之（正会員）

2006 年慶應義塾大学理工学部情報工学科卒業．2008 年慶應義塾大学大学院理工学研究科開放環境科学専攻修士課程修了．現在，UBS 証券会社（株）に勤務．

大村　　廉（正会員）

1999 年慶應義塾大学理工学部電気工学科卒業．2004 年慶應義塾大学大学院理工学研究科開放環境科学専攻後期博士課程修了．同年（株）国際電気通信基礎技術研究所（ATR）に入社．2007 年より慶應義塾大学理工学部情報工学科助教．現在に至る．オペレーティング・システム，センサ・ネットワーク，実世界情報処理等の研究に従事．電子情報通信学会，ACM，IEEE 各会員．博士（工学）．

今井　倫太（正会員）

1992 年慶應義塾大学理工学部電気工学科卒業．1994 年慶應義塾大学大学院計算機科学専攻修士課程修了．同年 NTT ヒューマンインタフェース研究所入社．1997 年 ATR 知能映像通信研究所へ出向．2002 年慶應大学大学院理工学研究科開放環境科学専攻後期博士課程修了．博士（工学）．現在，慶應大学理工学部情報工学科准教授および ATR 知能ロボティクス研究所研客員究員，科学技術振興機構さきがけタイプ研究員．人型ロボットとのインタラクションの研究に従事．電子情報通信学会，日本人工知能学会，日本認知科学会，ヒューマンインタフェース学会，ACM，IEEE 各会員．