Vol.2010-MUS-88 No.10
2010/12/5

ボクセルに基づく音楽生成のための 3 次元データマッピングインタフェース

濵 野 峻 行[†1] 岡ノ谷 一夫[†1,†2,†3] 古 川 聖[†4]

本稿では，ボクセルを操作単位とする音楽生成のための 3 次元グラフィカルインタフェースの開発について述べる．デザインとアフォーダンス，プログラミングパラダイムなどのキーワードを基に，美学的観点と技術的観点の双方について吟味しながら設計，柔軟なデータマッピングシステムを備える統合創作環境を実現させた．

Voxel-based 3-D Data Mapping Interface for Music Creation

Takayuki Hamano,[†1] Kazuo Okanoya[†1,†2,†3]
and Kiyoshi Furukawa [†4]

This paper describes a development process of a three dimensional graphical interface for music creation whose manipulative unit is voxel. Based on some keywords such as design, affordance, and programming paradigm, the system is planned examining both of aesthetic and technological viewpoints. Consequently, an integrated creation environment with a flexible data mapping system was realised.

†1 JST-ERATO 岡ノ谷情動情報プロジェクト
JST-ERATO Okanoya Emotional Information Project
†2 理化学研究所
RIKEN Brain Science Institute
†3 東京大学
The University of Tokyo
†4 東京芸術大学
Tokyo National University of Fine Arts and Music

1. はじめに

本稿で紹介するシステム“VDAM”(Volumetric Digital Art Mixer) は，画像処理技術の音楽生成への適用を目指す数々の試みの中で，2008 年から著者により開発が始められた．VDAM は，市販の音楽制作ソフトウェアのような特定のニーズに応えるべく開発されたものではなく，アーティストが試行錯誤をする中で，より幅広く新しい芸術表現の可能性を模索できるような環境となることを目標として設計されている．更に最終的には，データを 3 次元上の仮想オブジェクトとして表現することによって，音楽のみならず，楽器，自然現象，統計情報，ユーザインタラクションなどの様々なエンティティをシームレスに接続する媒介役となることを目指している (図 1)．これを実現するため VDAM では，“ボクセル”によるデータ表現の視覚化を原則として採用した．ボクセルを用いることで，ユーザはあたかも彫刻を彫ったり粘土をこねたりするような感覚で，3 次元で表されたデータを直感的に操作することができる．

VDAM の開発をする上で最根底となるモチベーションは，「如何に本質的に新鮮なアート作品の創られ得る技術的環境を整備できるか」という究極的な疑問にある．それは別の見方をすれば，「如何により多くの創作的可能性をアーティストに想起させる環境を作り得るか」という疑問に発展できる．これらの疑問を紐解く上で，無限とも言えるアートの多様性を一般化しようとするのはあまりにも挑戦的である．従って今回の開発では，アート制作の幾つかの側面のみに注目し，アーティストにとって作品制作の模索を行い易い環境を作るという点に限定して，システムの構築を試みた．

本稿ではまず第 2 章にて，概略的な在来のアート制作の過程についての考察と，デジタルインタフェースを活用した創造性の追求方法についての議論をする．その後に第 3 章で，VDAM で実装した技術の詳細について述べる．また，VDAM は多彩なスタイルのアートに適用されることを意図しているものであるため，第 4 章で応用的な適用例を列挙する．

2. VDAM 開発のアプローチ

2.1 美学的観点から

2.1.1 アートにおけるメディア

アートとは一面において，情動情報を循環させる行為であると言える．どのようなアート分野にあっても，人から人へ情動情報を伝えるためには必ず媒介が介在する．ここではそれを“メディア”と呼ぶ．例えば音楽におけるメディアには，楽器や楽譜，さらに広義では演

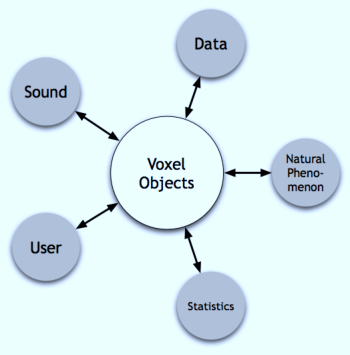

図 1　システムの目標概念図

奏音を媒介する空気までもが含まれ得る．メディアは「芸術的結果を生み出すべく表現すること」を手助けするものであり，それ自体が十分な潜在的能力を持っている必要がある．加えて，メディアの利用に熟達すればするほど，ユーザ自身の制御性が向上してより効率的に結果を生むことができるのであれば，尚望ましいことであろう．また楽譜のように，社会で情報を共有するという役割がメディアによって果たされるためには，ある程度の情報の記号化も必要となってくる．

以上のようなメディアのメタファーが芸術的創作をする上で意義を持ち始めるのは，それらが我々の想像に影響を与え得るという事実においてである．別の言い方をすれば，メディアのデザインが創作者のアフォーダンスを決定するとも言える．VDAM は，このようなアートにおけるメディアとしての役割を果たすものである．

2.1.2　データマッピング

データマッピングは，VDAM の設計において美学面での中核的概念となるものである．近年最新技術を活用しようとするアーティストは多くいるが，より重要なのは技術を単純にアートに適用するだけでなく，認知可能な感覚を如何に作り出すかということであると言える．[7] データマッピングはそれを可能にする，アート作品制作における 1 つの手法となるものである．

我々人間は何らかの新しい価値や，日常気付くことのなかった側面を発見した時に喜びを感じるものである．例えば，DATA FLOW というデザインブックには，データを様々なマッピング方法によって視覚化したイラストの例が沢山載っている．[5] そのように同じデータをオリジナルとは異なる方法で再提示することで，そのデータの新しい特徴的側面を発見しようとするのが，アートにおけるデータマッピングの意図するところである．広義には，自然界における現象の一面を切り取って拡大提示し認知可能な形にすることであり，アート手法の一つに成り得るものであろう．例えば，海洋に泳ぐ魚の群れの動きから音楽を作り，聴覚的にダイナミクスを感じ取ることができるとしたら，大変興味深いものになることが想像出来る．

またデータマッピングからの派生手法としては，ランダムな情報を意図的に付加する“ランダムネス”，観察により分析したモデルを再現する“シミュレーション”，既存の手法に則った上で逸脱した手法へと展開する“デビエーション”などが考えられる．このような各種のデータマッピングの手法は，創作者に発想を与えながら，新しい知見を作品に活用することを促すものである．VDAM はこのようなデータマッピングの体験を提供するものとなっている．

2.2　技術的観点から

デザインという言葉は一般に，視覚的な意匠の設計について用いられる言葉であるが，ソフトウェア工学においてはプログラミングパラダイムやデータ処理の仕方までに意味が及ぶ．一例として，オブジェクト指向プログラミングパラダイムは，ソフトウェアの開発形態に大きな変革をもたらしたものである．オブジェクト指向でのデザインパターンとは，まさにプログラムの再利用性と開発効率を高めるために，プログラムのデータとプロセスをクラスの関係として表現するものである．本節ではこの例のような技術的観点からのデザインについて，以下で検討する．

2.2.1　グラフィカルユーザインタフェース

今日我々が使っている多くのパーソナルコンピュータのグラフィカルユーザインタフェース (以下，GUI) は，WIMP(Window，Icon，Menu，Pointer) と呼ばれる概念を基にしている．この方式が長く使われて勝ち残ってきたことを見れば，汎用的な環境において直感的にデジタルデバイスに命令を伝達する ための優れた方式の一つであることがわかる．

これと類似する概念に，オブジェクト-アクションインタフェースモデルが挙げられる．[4] このモデルは GUI の操作方法について，データを仮想的実体 (=オブジェクト) とし，それに対して命令 (=アクション) を加えることにより処理が達成されるという様に捉える，二項構造の概念である．この考え方によれば逆に，UNIX コンソールに代表されるようなキャラクタユーザインタフェース (CUI) は，命令を先に入力した後に対象を指定するので，アクション-オブジェクトインタフェースモデルとして捉えることができる．両者どちらも今

日に至るまで使われ続けていることを考えれば，操作性と効率性の面で，両者にそれぞれの特徴から生じる利点があることが想像できる．VDAM では理想的環境の設計を追求した結果，内部構造を両者のモデルに対応させることにより柔軟なオブジェクト操作を可能にした．

2.2.2 ビジュアルプログラミング

ビジュアルプログラミング環境を開発する上で問題になるのは，データとプロセスを如何に視覚的に表すかという点に集約される．そこには常に，数々あるプログラミングの概念 (=abstractions) の内どの範囲までを取り入れるかというジレンマが付きまとう．

例えば Max/MSP[6] は一見するとオブジェクト指向的表現をしているが，実際は抽象クラスや継承といった概念を犠牲にすることによって視覚化に成功している．また別種の例として挙げられるのはタンジブルインタフェースであるが，これは情報を現実に触れることのできる物体で表現し，且つ直接操作できるようにしたものである．[3] この場合は現実世界の物理的制約を逆手に取り，情報のコントロールに積極的に活用している．

以上の例の通り，起こり得る概念の衝突を回避しながら概念の取捨選択をすることが，スマートな環境を開発する上で肝要と言える．今回 VDAM では，タンジブルではないビジュアルな環境を作ることを採択した．それは現実の物体では不可能な表現を可能にするためと，ソフトウェアの持つポータビリティを優先したためである．また，現在の VDAM の基礎となった前バージョン[2] においては，オブジェクト間を線で繋いで方向付き関係をプログラムできる仕様であったが，当バージョンでは廃止された．それは後述のように，当バージョンにおけるボクセルと解釈オブジェクト群の，二項構造のシンプルさを追求した結果である．

2.2.3 ボクセル

以上の考察を踏まえ，当システムはデータ表現の方法として“ボクセル”(=volumetric pixel) の概念を取り入れた．これまで一般的なコンピュータグラフィクスの市場では，パフォーマンスとデータ量の制約故に，頂点と面で構成される“ポリゴン”が多くのレンダリングエンジンに採用されてきた．一方ボクセルは，3 次元のグリッド上に置かれる膨大なブーリアンセル (=点) の集合として立体データを表現する方法である (図 2)．

ボクセルの特徴として以下のような事項が挙げられる．

- 無数の分子で構成されている現実世界の物体と同じように，ボクセルは物体の形だけではなく体積を表すことができる．また物体の重さを予測することも比較的容易である．
- ポリゴンで不得意とされてきたブーリアン演算での変形が容易である．

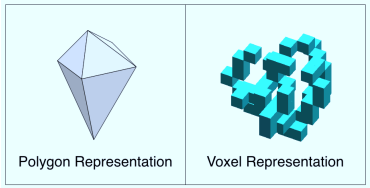

図 2　3 次元情報表現におけるポリゴンとボクセルの違い

- 一つ一つのセルはカラー情報を持つため，テクスチャの概念が無い．

ボクセルは，近年の記憶デバイスの大容量化とグラフィックプロセッサの高速化によって，益々注目を浴びている技術である．VDAM では，システム内で表現されるデータの操作単位としてボクセルの基礎概念部分を取り入れ，画像処理技術の点においては擬似的に再現した．これにより，ボクセルオブジェクトとそれに対する操作という様に，先に述べたオブジェクトとアクションの対比モデルがシステムに適用でき，わかりやすく且つ柔軟なデータマッピング環境を実現できた．

3. VDAM の設計と実装

3.1 開発の背景

今回開発した VDAM の元となったのは，著者による音楽の楽譜を仮想の 3 次元空間上に表現する作品“Saccade”である．[2] その作品のために開発したシステムのアイディアを拡張し，より広範な用途に使えるシステムを新たに制作できないかと考えたのが，VDAM 開発の始まりである．

3.2 VDAM の基本設計

図 3 に VDAM のアーキテクチャダイアグラムを示す．本システムの構成は，入力部，VDAM 本体の処理部分，出力部の 3 つより成り立っている．入出力は様々な外部装置の利用を想定しており，マウスやペンタブレット，外部 MIDI コントローラなどによっても操作可能である．また音楽用通信プロトコルである，OpenSoundControl を用いて入出力することも可能である．入力されたデータは通常，システム内で 3 次元オブジェクトとして表現され，ユーザによる操作の対象物となる．視覚化のための 3 次元グラフィックの描画は，OpenGL によって行われる．更に VDAM 処理部には各種の 3 次元オブジェクト処理の他，SuperCollider のプロセスも内包しており，3 次元視覚オブジェクトから音への変換をシー

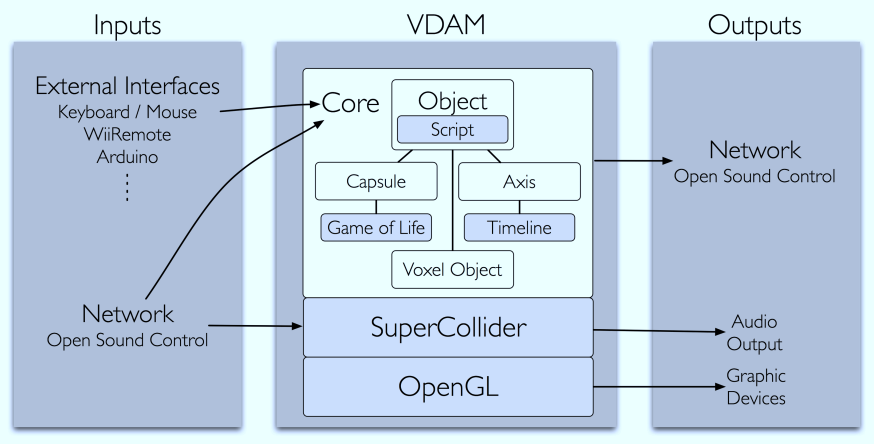

図 3　VDAM のアーキテクチャダイアグラム

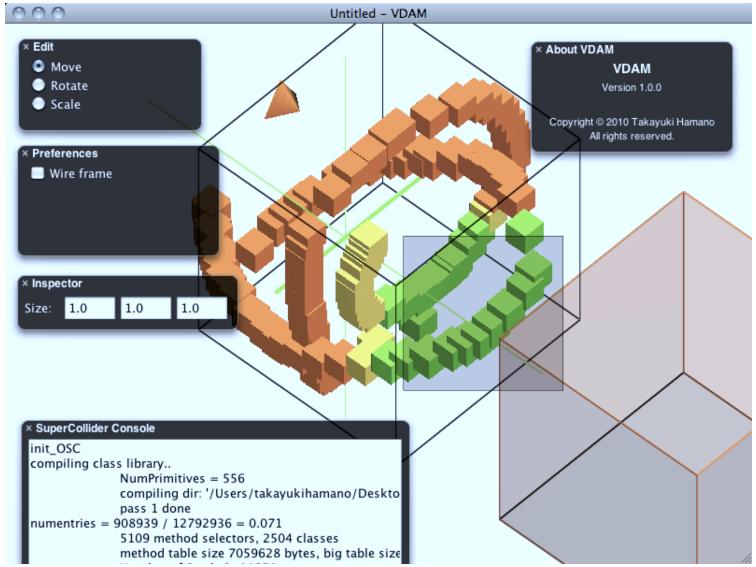

図 4　VDAM のスクリーンショット

ムレスに実現する統合開発環境となっている．尚，後述するスクリプト内に SuperCollider のコードを記述することも可能である．

3.2.1　VDAM の操作

図 4 は，VDAM を起動したときの様子である．メインのウィンドウは 3 次元のキャンバスを持ち，そこにユーザが各種のオブジェクトを配置，更にその上に表示されたパネルでオブジェクトの設定や編集操作ができるようになっている．他の一般的な音楽生成ソフトウェアと異なり，VDAM は固定されたタイムラインやスコアなどを持たない．また VDAM は 3 次元モデリングソフトのような感覚で，切り取りや貼りつけ，回転や変形等の編集作業を行える．

3.2.2　3 次元オブジェクトの内部構造

3 次元オブジェクト群に関するプログラムの内部構造は，オブジェクト指向的構造で設計されている．オブジェクトは階層構造を持ち，下層のオブジェクトは最上階のオブジェクトより継承される．上層オブジェクトは，下階のオブジェクトに用いられる共通の機能を定義し，自身はインスタンス化され得ない． また全てのオブジェクトは，他のオブジェクトとコミュニケートする能力と， 新規にオブジェクトを生成する能力を持つ．後述するボクセルオブジェクトや解釈オブジェクトは本システムの代表的なオブジェクトであり，これらのオブジェクトがコミュニケートすることで複雑な結果を生み出す．

3.3　オブジェクトの種類

VDAM でデータの基本操作単位となる，各 3 次元オブジェクトの種類と役割を以下示す．

3.3.1　ボクセルオブジェクト

ボクセルオブジェクトは座標値とカラー値を保持する．このオブジェクトによってボクセルの表示が実現する．これらのボクセルは複雑な集合を成して表示する事が可能である．ボクセルオブジェクトは，下記の解釈オブジェクトにより分析される．

3.3.2　解釈オブジェクト

解釈オブジェクトは，ボクセルオブジェクトの解釈のためのオブジェクトである．現時点で 3 種類の解釈オブジェクトが用意されており，各々の解釈を返すオブジェクトである．以下にそれぞれのオブジェクトを説明する．

(1)　メジャーオブジェクト— メジャーオブジェクトは，自身とボクセルオブジェクトとの距離を単純に算出して結果を返すオブジェクトである．

(2) 軸オブジェクト— 軸オブジェクトは，空間内で自由に軸を変形させるアイディアを基に作られた．ボクセルオブジェクトとこのオブジェクトの直交する点を相対的座標の x 値として取る．それに応じて y，z 値も同時に計算される．複数の軸オブジェクトを組み合わせることで，異なるパラメータ同士の補間を行うこともできる．

(3) タイムラインオブジェクト— タイムラインオブジェクトは，軸オブジェクトの発展形である．このオブジェクトは特別に，音楽シーケンサでの時間軸上の位置を表すバーに相当する平面を持つ．プレイボタンを押すことでその平面は軸に沿って移動する．平面がボクセルオブジェクトと衝突した時をトリガーとして，軸オブジェクト同様の方法でボクセルオブジェクトの相対的座標位置を計算し，結果として出力する．タイムスケールとバーのスピードは変更可能である．

3.3.3 カプセルオブジェクト

カプセルオブジェクトは特殊なタイプのオブジェクトで，ボクセルオブジェクトをカプセル化して，より複雑なボクセルの操作をするためのオブジェクトである (図 4 中，大きな立方体の表示されている部分)．カプセルオブジェクトは任意の容積を持ち，カプセル内のオブジェクトの動作範囲を制限することができる．また後述するスクリプトによって，カプセルの内包するオブジェクトを動かしたり色を変えたりと，様々な影響を与えることが可能である．

3.4 スクリプティング

全てのオブジェクトは，自身の挙動を定義するスクリプトを持つ．カプセルオブジェクトの場合は，カプセルに内在する他オブジェクトを操作するためのスクリプトも持つことができる．スクリプトは Java の文法に従って書かれているため，全てのオブジェクトにおいて，スクリプトの継承が可能である．また，視覚化のスクリプトは純正な OpenGL のコードによって記述される．

スクリプトはシステム内のソースコードエディタでスクリプトを手軽に編集，コンパイル，動的リロードをすることができる．またユーザの操作の負担を軽減させるために，全てのスクリプトはプログラムを起動した時，或いはシステム内のエディタで編集した際に随時自動的にコンパイルされる．これにより，ユーザは他環境にオブジェクトを移植する際，ソースコードファイルをコピーだけで済ませる事ができる．

4. VDAM の活用例

VDAM は幅広い用途で柔軟に利用できるように設計されている．以下に，VDAM を使って制作した実用的なシステムの例をいくつか紹介する．VDAM では全てのデータはボクセルとして表されるので，それらの各例を複数組み合わせて使うことも設計概念上可能である．

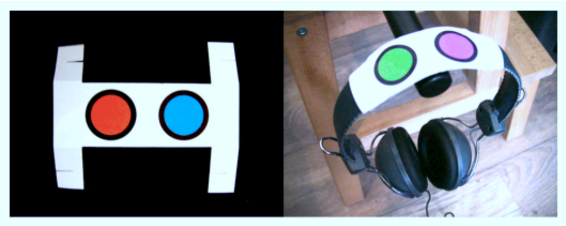

図 5 ヘッドトラッキング用マーカーとヘッドホン

4.1 ライブパフォーマンス

最初の活用例は，リアルタイムに音楽演奏を行うライブパフォーマンスである．パフォーマーは入力機器で VDAM の 3 次元オブジェクトの操作を行い，SuperCollider にパラメータを送信することで音楽を生成できる．筆者が制作した作品“Concert Étude”では VDAM の柔軟な操作性を示すべく，通常のマウスと同等の機能しか持たないペンタブレットのみを入力インタフェースとして使った．一種類の入力方式のみによっても，パフォーマは操作に熟達するほどに柔軟なデータマッピングを実現することができ，より複雑な表現を一度に操ることが可能となる．その点で，通常のライブパフォーマンスのように入力コントローラとパラメータを 1 対 1 で対応させる方法よりも，高度で応用的である．

4.2 ヘッドトラッキング

通常我々が特定の場所から聴こえる音源をよりクリアに聴こうとする場合，音源の方向に耳を傾けようとするだろう．ヘッドトラッキング機能はこの体験を仮想的に実現するものである．ユーザは図 5 のようなマーカーの付属したヘッドホンを装着する．ユーザの頭の動きは頭上に設置したビデオカメラにより検出される．検出された結果はヘッドオブジェクトとしてキャンバス中央に表示される．それに応じて，キャンバス内に配置された音源を表すオブジェクトとの相対的な距離と方向関係が瞬時に計算される．結果として，現実世界での音場を仮想的に体験できるというものである．この機能の利用法としては，音楽作品やインスタレーション作品に用いられることが期待される．

4.3 画像の音声化

画像の音声化 (Sonification) は，入力した視覚情報を聴覚情報に変換するものである (図

Vol.2010-MUS-88 No.10
2010/12/5

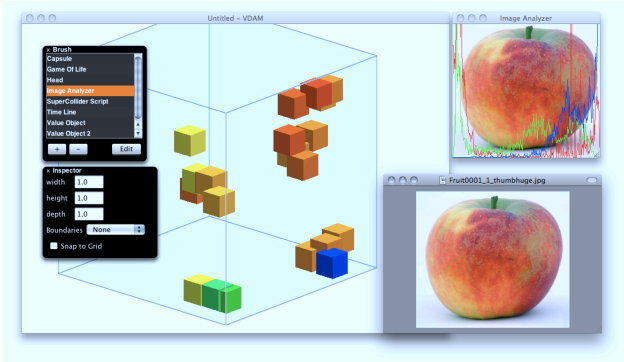

図 6　画像分析の画面

6)．VDAM ではこの機能が以下のように実装されている．まずスクリーンの特定の領域をリアルタイムにキャプチャし，画像の視覚的構成を分析する．画像は内部処理によって HSB のヒストグラム情報に変換され，特徴点が求められる．抽出された特徴点はボクセルオブジェクトとして空間上に表現される．入力は静止画と動画の両方に対応しており，カプセルオブジェクトの派生として実現されている．この機能を作曲に利用することで，興味深い結果が得られると思われる．またボクセルに変換した画像データを，音声のみならず他の様々な用途にパラメータを用いることも可能である．

4.4　Game of Life シミュレーション

アルゴリズミックコンポジションのフィールドにおいては，古くからセルラーオートマトンについて多くの関心が注がれてきた．コンウェイのライフゲーム (Conway's Game of Life)[1] はその中でも有名な一例であるが，通常 2 次元で実施されるこのシミュレーションを，VDAM では 3 次元空間に拡張して適用した．応用例としては，ボクセルで表された他のデータと組み合わせてランダムな挙動を与えるなど，拡張的な作曲技法として用いることもできるだろう．

5. ま と め

本稿では，ボクセルを基礎概念として用いた，音楽生成のための 3 次元データマッピングインタフェース VDAM について解説した．そして VDAM が様々なエンティティをシームレスに接続し，アートメディアとして媒介役の機能を果たすことを，応用例で紹介した．アートのためのソフトウェア環境制作においては，デザインとアフォーダンスを吟味することが必須である．本システムの開発ではそれを踏まえて設計，結果的にアーティストが試行錯誤を行う環境の提供を可能にした．将来的な課題として，活用実績の拡大，開発用 SDK の公開と開発コミュニティの形成などが挙げられよう．

参 考 文 献

1) Gardner, M.: MATHEMATICAL GAMES: The fantastic combinations of John Conway's new solitaire game "life", *Scientific American*, Vol.223, pp.120–123 (1970).
2) Hamano, T.: MC (2008). http://www.takayukihamano.com/projectsworks.htm.
3) Ishii, H. &Ullmer, B.: Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms, *Proceedings of Conference on Human Factors in Computing Systems (CHI '97), ACM*, Atlanta, ACM, pp.234–241 (1997).
4) Khella, A.: Objects-Actions Interface model (2002). from University of Maryland, Department of Computer Science website, http://www.cs.umd.edu/class/fall2002/cmsc838s/tichi/oai.html.
5) Klanten, R., e.a.: *Data Flow*, Gestalten (2008).
6) Puckette, M.: Max at Seventeen, *Computer Music Journal*, Vol.26, No.4, pp.31–43 (2002).
7) Ryan, J.: *MuViz: Visualization Notes*, NAi Publishers (2003).