音楽情報科学 18－4
・（1996．12．14）

「広義の楽器」用ツールとしてのMIDI活用

長嶋洋一
イメージ情報科学研究所

楽器演奏情報プロトコルとしてのMIDIを離れて、広い意味での音楽情報処理ツールのための汎用プロトコルとしてのMIDI活用について、実例とともに紹介する。具体的な作品やシステムでの適用事例、専門的技術を従来ほど必要としないカードマイコンを利用したオリジナルMIDIツールの開発、Javaによる開発環境の検討などの紹介とともに、汎用インターフェースとしてのMIDIのメリットとデメリットについて考察・検討する。

MIDI applications for computer music as flexible tools

Y.Nagashima
Laboratories of Image Information Science and Technology

This paper reports many applications with MIDI, not as standard musical instruments but as flexible tools for computer music. We always produce many original MIDI tools: Sensors, Interfaces, Mergers, Switchers, Displays, Generators, Converters, etc. We discuss the merits and demerits of MIDI as universal interface and tools.

1. はじめに

コンピュータ音楽(Computer Music)創造環境の構成要素として、これまで色々な要素技術や概念について検討するとともに、具体的な作品として実験的な応用を試みてきている[1-8]。ここでは、世界的な音楽演奏情報のための統一規格であるMIDIを活用して、市販の電子楽器などの機器のために用いるだけでなく、オリジナル開発した各種の機器そのものにMIDIを組み込み、開発環境"MAX"によって効率的かつ有効な研究開発を行っている。本稿では、このような状況について紹介するとともに、音楽情報科学におけるMIDI活用の可能性と課題について報告とともに検討する。

2. MIDI システム化の実例

楽器メーカから提供される電子楽器や音源を用いて、ソフトハウスから提供されるシーケンスソフトによって音楽演奏情報を記述する、という形態のコンピュータ音楽は一般的なものであるが、これはビジネス領域でもホビー領域でも広く一般に行われているものであり、本研究ではこのような環境はあまり採用していない。たとえば、「ステージ上のPerformerが自由に身振りによって音響と映像を駆動したようなパフォーマンスを実現したい」というような目標を持った場合、そのような市販のセンサや制御機器は非常に特殊で高価な製品が海外にあるものの、結局は民生の電子楽器と同様に、提供される機能仕様に限定された使い方しか実現できないために、作品を創造する、という立場からは常に欲求不満と妥協の連続となってしまう。このような状況に対して、よりアーティストにフリーハンドを提供する統合された創造支援環境を実現していきたい、という目的が本研究の重要な原動力になっている。

たとえば図1は、1996年7月13日に日本コンピュータ音楽協会「コンピュータ音楽の現在'96」(ジーベックホール)で発表した、インタラクティブ・マルチメディアcomputer music作品"Asian Edge"のシステム図である。ここでは、身振りを検出するセンサを身に付けたPerformerの動きは音響サンプルをトリガしたりCGをリアルタイムに動かしたり、ビデオカメラによる実映像や映像作家の創った背景映像をリアルタイムにスイッチングすることで音楽を進行させていく。このような作品に満足に使用できる機器が存在しないために、たいていの機器やソフトウェアをオリジナル開発するところから作曲が始まる。

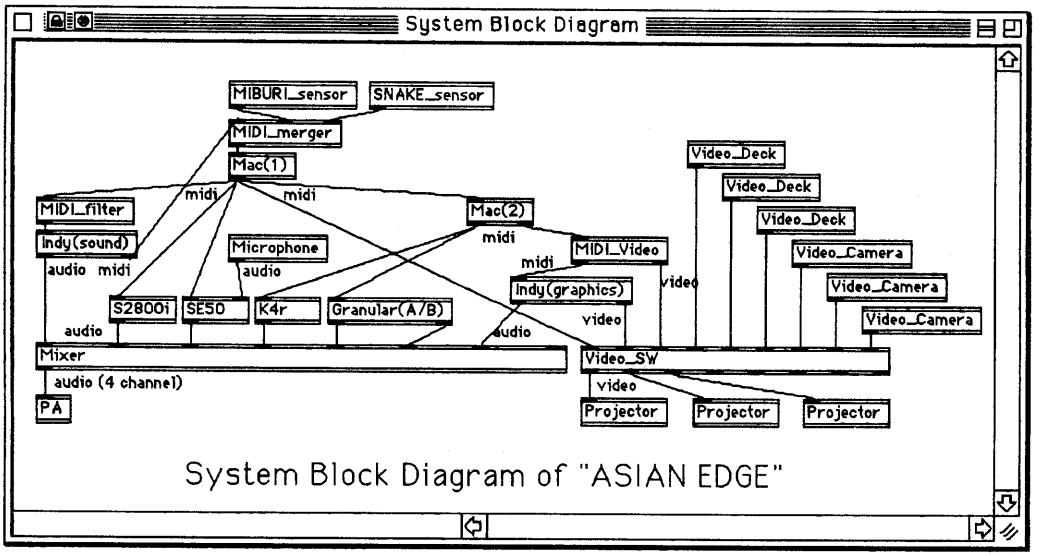

図1

また、図2は1996年10月19日に「京都メディア・アート週間」(関西ドイツ文化センター)で発表した、インタラクティブ・マルチメディア作品"Johnny"のシステム図である。ここでも、ステージ中央でダンスPerformanceを行うための各種センサは全てオリジナル開発するとともに、映像系・CG系・センサ情報のパターン認識処理・音楽系・全体の制御系なども全てMIDIベースで統合され、環境"MAX"によって柔軟にシステムのチューニングを行っている。

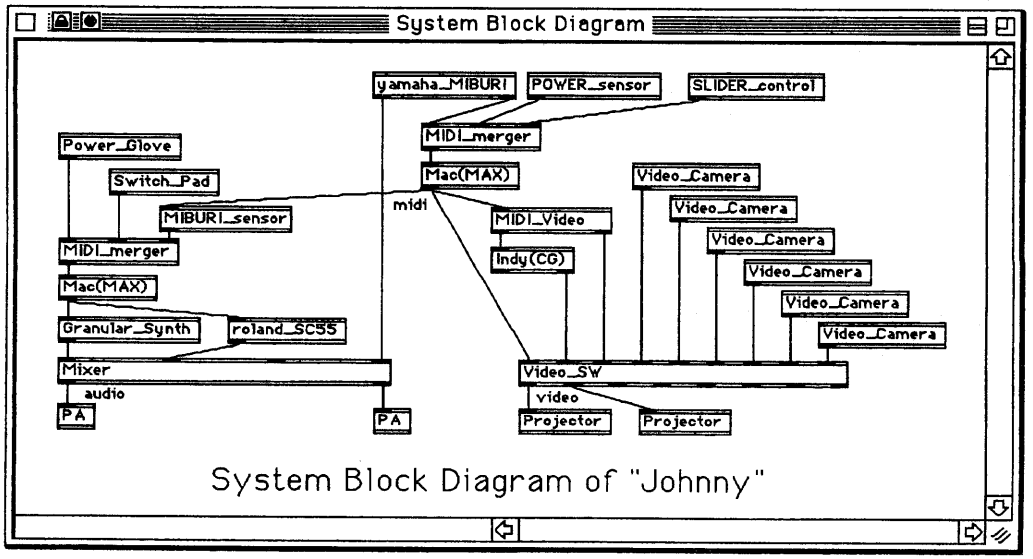

図2

3. オリジナルMIDIツールの例

筆者がこれまでに開発したオリジナルMIDIツールの一部としては、以下のようなものがある。

パワーグローブ	ファミコン用PowerGloveの超音波センサでX-Y軸の座標をMIDI出力
ワイヤレスPowerGlove	ファミコン用PowerGloveの指曲げ情報を無線でMIDI出力
MIDIジョイスティック	ゲーム用JoyStickのX-Y軸の座標をMIDI出力
赤外線ビームセンサ	赤外線ビームを遮断する速度をMIDI出力。閾値もMIDIで設定可能
身振りセンサ	手首・肘・肩の曲げセンサをMIDI出力
衝撃センサ	16箇所の超小型ピックアップで衝撃をMIDI化(prepared piano)
MIDIストロボ	MIDIを受けて16個の使い捨てカメラのストロボを光らせる
MIDIテストパターン生成	世界各地のビデオ基準信号やカラーパターンをMIDIで選択
MIDIマージャ	2-4系統のMIDI入力情報をマージする
ビデオスイッチャ	8入力8出力のビデオマトリクススイッチをMIDIでリアルタイム制御
MIDIディスプレイ	所定のMIDIコードで与えられた3桁の10進数を巨大LEDで表示
Indy用インターフェース	不要なMIDI入力をプログラマブルにカットする多機能フィルタ
MIDIエコーマシン	入力されたMIDI情報を自由な設定によって時間的に多重化
汎用A/D・D/A変換	32チャンネルのアナログ入力と出力をMIDIと変換(IRCAMに納品)

4. MIDIツールの開発環境

このようなMIDIツールを実現するためには、従来は汎用CPUの搭載されたいわゆるマイコンボードを製作し、ここにMIDI信号のためのシリアル通信LSIを載せて、必要なCPUプログラムをアセンブラで記述する必要があった。このような方法を取らない場合には、MIDIツールごとにパソコンを1台ずつ割り当てる必要があるが、設備投資と手間だけでなく機動性に欠ける方法であった。

しかし最近のカードマイコンの登場によって、このような状況は非常に改善されてきている。次ページの図3は、筆者が愛用しているMIDIツールのもっとも基本となるシステムの回路図であり、これだけで上記のMIDIツールのかなりのものがカバーされている、いわばエッセンスとなっている。この図はパソコン通信Nifty-Serveから電子メイルとしてFAXを宛先に発信したものであり、受信したFAX機をプリンタとして利用した。回路図が特殊なCADでなくplain textとして表現できるほどに単純であることに注意されたい。

この回路図は、本研究会の前身である任意団体「音楽情報科学研究会」メンバーによるインターネット上のメイリングリストjmacs-ML(現在は消滅)との情報交換を基本としてNifty-ServeのMIDIフォーラム内に開設された「音楽情報科学」会議室に筆者が紹介したものであり、この電子会議室では音楽家・演奏家・愛好家などが情報交換や議論を行っている。筆者の紹介したこの回路図をもとに、エレクトロニクスの専門家でないアマチュアが実際にこのシステムを製作し、筆者の提供したCPUプログラムとともに現実にオリジナルMIDIセンサとして完成させ、自作のセンサを接続して活用している実績があり、最近はさらに「脳波センサ」「心拍センサ」などの話題に広がってきている。

回路図にある「AKI-80」とは、秋葉原にある秋月電子のオリジナルのカードマイコンであり、東芝製の多機能8ビットCPUを中心に置いた、低価格高性能のボードである。このキットをいわば「部品」として使うことで、従来のマイコンシステムに比べて一桁ほど手間がかからずに簡単にハードウェアを製作できるようになった。そして、AKI-80の周辺にはたった3個のICがあればMIDI入出力が実現でき、さらにたった1個のA/D変換チップによって、同時に8チャンネルのアナログ電圧をリアルタイムにMIDI情報として利用できる、オリジナルMIDIツールが完成してしまう。

発信元　長嶋　洋一
(NBD03033)

題名 ー 音楽情報科学研究会発表原稿用テキスト図

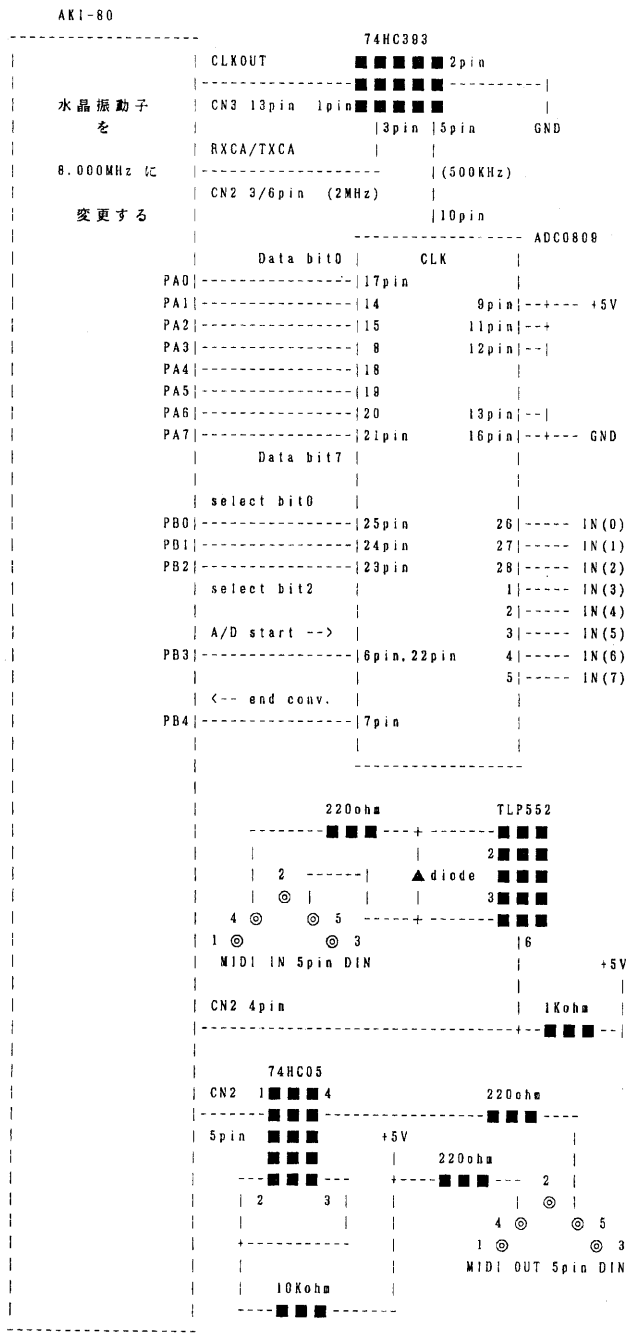

図3

電子会議室でこの記事を参考に実際に製作した人からのリクエストで、筆者が提供したCPUのプログラムは以下の図4のようなリストである。これは「インテルHEX形式」と呼ばれる、EPROMをプログラムするための「ROMライタ」にRS-232-Cで転送するためのものであり、これが得られれば、専門外の人であればこのデータを得るためのアセンブラプログラミングについては省略してしまうことが可能となる。このようなボランティアの協力は、同じ音楽情報科学愛好者として共にオリジナルMIDIツールによる新しい可能性を求めていきたい「仲間」にとっては自然な協力であり、世界のcomputer musicコミュニティでも日常的な文化となっている。

手元のサンプルプログラムを紹介します。これは既に作ったものそのままなので、仕様としては
・A/D入力のIN(0)-IN(5)までの6チャンネルを出力する
・MIDI入力のうち、ノートオン／オフだけを、全てMIDIチャンネルを1チャンネルに書き換えて出力しつつ、チャンネルプレッシャー出力としてA/D入力情報をマージして出力する
・PIOのポートBのビット5のLEDを点滅させる
という動作をします。以下の==========で挟まれた部分だけを切り出したファイルを、「****.HEX」という名前のインテルHEXファイルとして、ROMライタに転送するとOKです。オフセットアドレスは0番地にして下さい。

```
==========
:0700000031FF9FF3C368000C
:1C002000220008D9ED5B00903E80B2676BDB187713CB9AED530090D908FBED4DDF
:20006600ED452100803EA0360023BC30FA3ECFD31D3EFFD31D3E07D31D3ECFD31F3E10D30B
:200086001F3E07D31F3E18D31B3E04D31B3EC4D31B3E01D31B3E00D31B3E02D31B3E20D3E8
:2000A6001B3E18D3193E04D3193EC4D3193E01D3193E10D3193E05D3193E68D3193E03D387
:2000C600193EC1D319ED5EFBDB183E00320F90D31EAF321090CD1A02CD9A01CDEC00CD2263
:2000E60001CDD00118F2ED5B06902A0490A7ED52C83E00D319DB19CB57C83E88B2676B7E37
:20010600D31813CB9AED530690C9ED5B04903E88B2676B7013CB9AED530490C9ED5B029052
:200126002A0090A7ED52C83E80B2676B4613CB9AED530290CB78281E78FEF8D0FEF038058D
:20014600AF320890C978E60F320A9078E6F0320890AF320990C93A0890FE00C8FEC0C8FEA7
:20016600D0C83A0990FE0020093C32099078320B90C9AF3209903A0890FE90C078320C90F2
:200186000690CD10013A0B9047CD10013A0C9047CD1001C93A0D903C320D90FE00C03A0E3F
:2001A600903CE67F320E90FE1E2805FE2D280EC93A0F90E601F620320F90D31EC93A0F908B
:2001C600E601E6DF320F90D31EC93A18903C321890FE32D8AF321890DB1ECB67C8DB1CCBA4
:2001E6003F4F3A10906F2600111190197EB92815797732179O3A1090F6D047CD10013A17DE
:200206009047CD10013A10903C321090FE062004AF3210903A0F90E6204F3A1090B1D31EE8
:0D022600F608D31E00E6F7D31E320F90C974
:00000001FF
==========
```

図4

5. C言語による開発環境

AKI-80のようなカードマイコンでオリジナルMIDIツールを開発する場合のネックとしては、ソフトウェア開発言語としてのアセンブラの必要性も大きな壁となる。従来のパソコン上でのMIDIツール開発の場合には、ローランドのMPUボードなどによって比較的高速処理の必要なMIDI信号周辺のプログラミングが吸収されるために、C言語のような重い言語でも記述できたが、MIDI周辺のハードウェアを直接に制御する必要があるAKI-80などの場合には、C言語では処理しきれない問題点となっていた。

そこで筆者は、Cは知っているがアセンブラに深入りしたくないというプログラマのためのAKI-80ライブラリを試作してみた。使用したのは秋月電子の2,500円のCコンパイラ体験版であり、このコンパイラがCソースをアセンブラソースに変換する機構を解析して、MIDI処理の高速部分についてはオリジナルのライブラリを用意することによって、低速な部分やユーザインターフェース部分を図5のようにC言語で記述できるようにしたものである。

```
#include   "library.c"      /* Common Tools for AKI-80 */

initial(){
     lcd_module_setting();      /* using Port[A] */
     timer_0_setting();         /* about 10msec */
     midi_port_setting();       /* sio_a */
     enable_interrupt();        /* EI */
     sio_dummy_read();          /* omajinai */
}

main(){
     int timer,t;
     t = 0;
     initial();
     disp_string( 0, "MIDI =      ");
     while(1){
          tx_midi_check();
          lcd_disp_check();
          midi = rx_midi_check();
          if( midi < 256 ){
               tx_midi_set( midi );
               midi_display( midi );
          }
          if( ram_get(timer_flag) != 0){
               ram_put( timer_flag, 0 );
               timer++;
               if( timer > 50 ){
                    timer = 0;
                    line_disp(t++);
               }
          }
     }
}
```

図5

6. Java言語による開発環境

C言語に代わってプログラミング言語の主役になろうとしているJava言語でMIDIツールを開発する、という発想は誰にも自然な流れである。メインストリームとしては、現在アナウンスされている「Javaチップ」が多種大量の民生機器に使用されて、それがJavaチップ版のAKI-**としてカードマイコン化されたものを使う、という方向である。しかしこれにはまだ時間がかかるために、筆者は「JavaでAKI-80のMIDI処理をプログラミングしてみる」という実験を行ってみた。

このためには、まずsunのサイトから公開されているJava言語仕様と仮想マシン仕様(VMSPEC)を理解し、さらに具体的なJavaプログラム(バイトコード)の構造を解析してみる必要があった。結局、Javaアプレットを解析して逆コンパイルするためのオリジナルコンバータを製作してみることで、JavaからCへのコンバータというツールの試作版を開発することができた。次ページの図6は、一例としてJavaで記述したアプレットのプログラムであり、ブラウザ上の画面としては図7のようになっている。このバイトコードをそのままオリジナルのコンバータで変換して、このために作ったライブラリとともにCコンパイラでコンパイルすることで、最終的に「マウスによって画面のスライダが動く」というJavaアプレットでエミュレーションした動作が、「アナログ入力をMIDI出力する」マシンとして実現できた。

```
import java.awt.*;
import java.applet.*;
import java.lang.*;
import java.io.*;

public class miditx extends Applet implements Runnable{
    int py[],oy[],data[],flag,channel;
   Graphics g;
   private Thread flow;
   Font font;

   public void init(){
     channel = 8;  system_initial();
   }

   public void run(){
     try {
       while(true){
         Thread.sleep(100);  input_check();
         if( flag != 0 ){
           midi_transmit();  repaint();
         }
       }
     }
     catch(Exception e) {}
   }

   public boolean mouseDrag(Event e, int x, int y){
     event_check(x,y);  return true;
   }

   void event_check(int x, int y){
     int z;
     for(int i=0;i<channel;i++){
       z = 40 + 40*i;
       if( (x>z-10) && (z+10>x) && (9<y) && (191>y) ){
         oy[i] = py[i]; py[i] = y;  flag++;
       }
     }
   }

   void system_initial(){
     setBackground(Color.black);
     font = new java.awt.Font("TimesRoman", Font.PLAIN, 16);
    py = new int[channel];  oy = new int[channel];  data = new int[channel];
    for(int i=0;i<channel;i++) py[i] = 190;
    flag = 0;
   }
}
```

図6

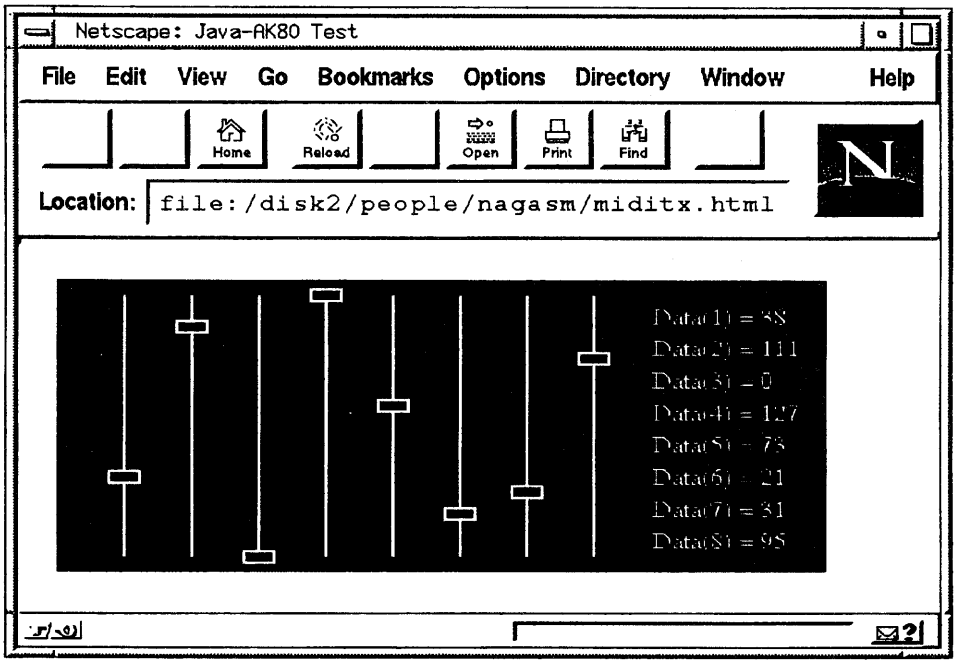

図7

7. おわりに

MIDIによるオリジナルツールの開発について報告と検討を行った。ZIPIやRMCPなど新しいプロトコルの提案も世界的に続いているが、DATやMDがあってもカセットテープが消滅しないように、もっとも基本的なプロトコルとしてのMIDIは今後も活用されていくものと思われる。今後もさらに新しい実験のために必要なマシンを開発していくとともに、音楽情報科学コミュニティとして共有できる情報はNiftyの電子会議室やホームページなどの場を通じて公開し、共有していきたい。

参考文献

[1] Y.Nagashima, H.Katayose, S.Inokuchi : PEGASUS-2: Real-Time Composing Environment with Chaotic Interaction Model. Proceedings of ICMC, pp.378--390, 1993.

[2] 長嶋洋一：マルチメディアComputer Music作品の実例報告. 情報処理学会研究報告 Vol.94,No.71 (94-MUS-7), pp.39--44, 1994.

[3] 長嶋洋一, 由良泰人 : Multimediaパフォーマンス作品 "Muromachi". 京都芸術短期大学紀要［瓜生］第17号1995年, pp.39--43, 1995.

[4] 長嶋洋一, 片寄晴弘, 由良泰人,井口征士：画像情報と統合化されたコンピュータ音楽創造環境の構築. 情報処理学会平成7年度前期全国大会講演論文集I, pp.363--364, 1995.

[5] Y.Nagashima : Multimedia interactive art: system design and artistic concept of real-time performance with computer graphics and computer music. Proceedings of HCI International, Yokohama, 1995.

[6] 長嶋洋一, 片寄晴弘, 由良泰人,井口征士：画像情報と統合化されたコンピュータ音楽創造環境の構築. 情報処理学会平成7年度前期全国大会講演論文集I, pp.363--364, 1995.

[7] Y.Nagashima, H.Katayose, S.Inokuchi : A Compositional Environment with Interaction and Intersection between Musical Model and Graphical Model --- "Listen to the Graphics, Watch the Music" ---. Proceedings of 1995 International Computer Music Conference, pp.369-170, 1995.

[8] 長嶋洋一, 由良泰人, 藤田泰成, 片寄晴弘, 井口征士：マルチメディア・インタラクティブ・アート開発支援環境と作品制作・パフォーマンスの実例紹介. 情報処理学会研究報告 Vol.96,No.75 (96-MUS-16), pp.39--46, 1996.