平成12年度 デザイン学部長特別研究費
研究成果報告書

1. 研究の概要

名称　　ネットワーク支援によるデザインシステムの研究

研究者　　松原季男・長嶋洋一

概要　この研究は、平成12年度後期デザイン学部長特別研究として、コンピュータ技術およびインターネットを活用した新しいデザインシステムを模索し研究したものである。

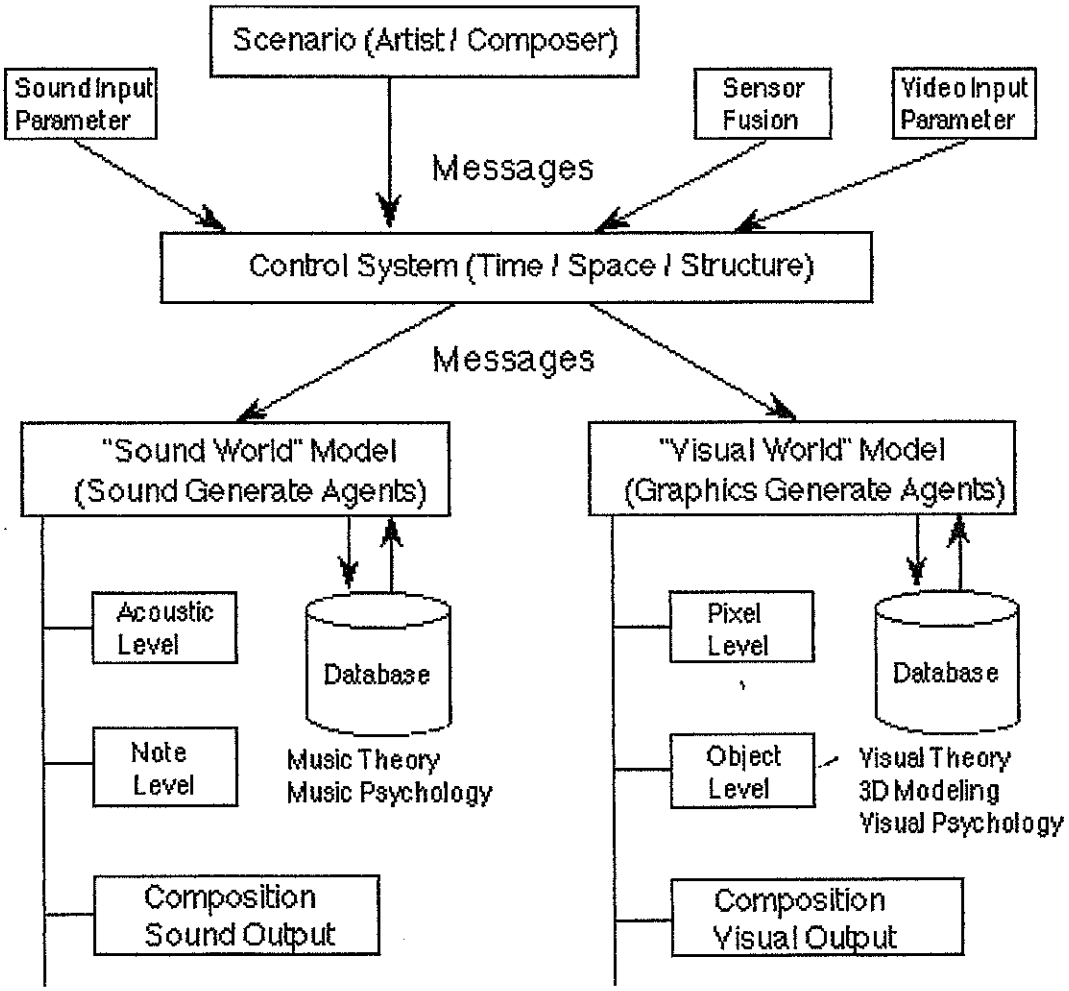
2. 研究の背景と目的

効果的で芸術的なインターネット・ホームページを制作する際に、専門的なプログラミング技術を必要としないでコンテンツの内容に専念できるための、ホームページ・コンテンツ制作支援環境の開発を行うことを目的とした。具体的には、

(1)視覚的なコンテンツと聴覚的なコンテンツを相互に有機的に結合したマルチモーダル環境の構築

(2)遠隔地のデザイナーが協力してコンテンツ制作を行えるコラボレーション支援環境

という2つのテーマがあり、これらは両輪として有効な制作環境を実現する。なお、従来のコンテンツ制作では、画像(動画／静止画)と音響とを個別に扱う環境しか存在しないが、本研究においては「目で聴き、耳で観る」という基幹コンセプト(下図)がユニークなものとなる。



インターネット・ホームページによる情報発信や個人領域の拡大、さらにビジネス応用の展開において、現在もっとも不足しているのが効果的で芸術的なマルチメディア・コンテンツの制作環境である。本研究が目的としたシステムは、このような有効なホームページ・コンテンツの制作をヒューマンインイーフェースの視点から支援するとともに、インターネットで結び付いた遠隔地のデザイナーやアーティスト等のコラボレーションによる制作を支援することで、在宅勤務やバーチャルカンパニーという産業振興上の新しい可能性の発展にも寄与することを目的とした。

3. 研究結果の報告

3-1. インターネットの状況と考察

インターネットの普及とともに、WWW(World Wide Web)を単に「見る」だけのネットサーフィンの時代は過去のものとなり、個人、組織、企業などがホームページによって積極的に情報発信を行うことが主流となってきた。ところが、HTML(HyperText Markup Language)を基盤とするホームページは、プログラミング技術など専門的な知識を必要とするために、デザイナやビジュアル／サウンド等のアーティストにとって敷居の高いものとなっていた。この一方で、工学系のエンジニアの制作するホームページにはセンスの乏しい画一的なものが多く、ホームページ制作に対する需要と、コンテンツ制作スタッフの供給とのアンバランスは解消されるどころか拡大している状況にある。

ここに、最近は単純に画像ファイルを並べるだけでない、インタラクティブ(対話的)な機構を盛り込む、CGI・Java・VRML・JavaScript等の各種の技術も登場したが、デザイナやアーティストにとっては、ますます専門的技術の壁が高くなってきた。この一方で、ホームページ制作を支援するツールとして、素材を配置すればそのままHTMLファイルを自動生成する各種の「お手軽ツール」が発表されてきたが、いずれも素材を単純に配置するだけのものであり、画像・音声・テキスト・リンクなどのコンテンツを希望する意図のもとに有機的に結合するためにはあまり十分とは言えず、結局専門的なプログラミング技術を必要とすることになり、問題は本質的にはいぜん解決されていないと思われた。

3-2. 研究の基盤要素とアプローチ

本研究実施者の一人(長嶋)は、これまでマルチメディア・インタラクティブ・アートの研究と実験を行う中で、視覚系の情報と聴覚系の情報とを有機的に結合するアプローチを続けてきた。ここでは、

- ビジュアル系、サウンド系のアーティストのコラボレーション
- Unixネットワークの活用とOpen-GLによるマルチメディア処理
- 複数のメディアの相互結合と相互変換によるマルチモーダル
- ヒューマンインターフェース手段としてのセンサフュージョン

等について、実験的検証を交えた研究を進めてきた。また一方で、インターネット・ホームページに関連した技術のセミナーの講演や書籍の執筆出版、企業の人材育成講習や、Javaプログラミングのセミナーを国内および海外で開催している。ここでは、既存の制作支援環境の問題点やコンテンツ制作のポイントも十分に把握しており、本研究に結び付くベースとなった。

3-3. 研究内容

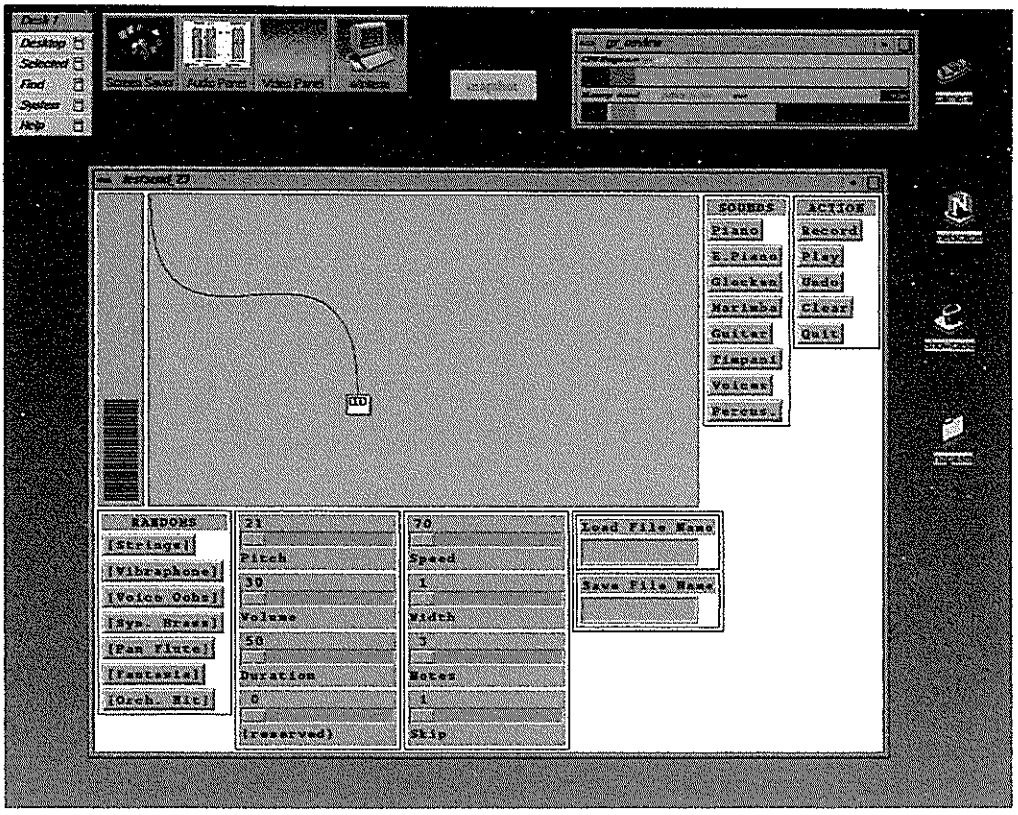
本研究では、効果的で芸術的なインターネット・ホームページを制作する際に、専門的なプログラミング技術を必要としないでコンテンツの内容に専念できるための、ホームページ・コンテンツ制作支援環境の実験と試作の開発を行った。具体的には、(1)視覚的なコンテンツと聴覚的なコンテンツを相互に有機的に結合したマルチモーダル環境の構築、(2)遠隔地のデザイナーが協力してコンテンツ制作を行えるコラボレーション支援環境、という2つのテーマについて、これらが融合して有効な制作環境をとなるような実験システムを検討した。また、もっとも重点を置く研究テーマとして、ヒューマンインターフェース部分のツール、具体的にはマルチメディア・コンテンツ制作支援に特化したセンサ方式について実験と開発研究を行った。以下、各研究項目とその成果について紹介する。

(1)プロトタイピングとオーサリング環境の開発実験

- 各種コンテンツの「素材」としての制作環境を整備した

・プロトタイピング環境をUnixのX-WindowおよびOpen-GL環境上に構築した
・多種の素材をマルチモーダルに結合させるマルチエージェント系
の実験開発を行った
・GUIのオーサリング環境をUnixのX-WindowおよびOpen-GL環境上に構築した

下図は試作したソフトウェアの画面の一例であり、画面内のマウスを動かすことでグラフィクスと連動してサウンドを配置し、一種のシーケンスデータとして簡単にプログラミングできるようになった。



このプログラムはUnix上でOpen-GLを用いてC言語で開発した。以下がそのソースコードである。

```
#include
#include
#include
#include
#include
#include
#include
#include

#define WORLD_XMIN          0.0
#define WORLD_XMAX          200.0
#define WORLD_YMIN          0.0
#define WORLD_YMAX          200.0
#define GREY            8
#define MOUSE_OBJ           1
#define LEN             700.0

/*** (^_^;) Bug Recovery of SGI Media Library --> Original MIDI Defines !! (^_^;) ***/
#define MES(x)              ((x[0]).mm.msgbuf)
#define midi_set_status(x,d)   (MES(x) = ((d > 0xbf) && (d < 0xe0)) ? 0x40000000 : 0x60000000, \
                    MES(x) &= 0xff0fffff, MES(x) |= ((d & 0xf0) << 16))
#define midi_set_channel(x,d)  (MES(x) &= 0xfff0ffff, MES(x) |= ((d & 0x0f) << 16))
#define midi_set_keyno(x,d)    (MES(x) &= 0xffff00ff, MES(x) |= ((d & 0x7f) << 8))
#define midi_set_velocity(x,d) (MES(x) &= 0xffffff00, MES(x) |= (d & 0x7f))

MIport *midi_port;
MIevent midi[100];
XtAppContext app_context;

int mx, my, xmin, xmax, ymin, ymax, button[3], color_no, action_mode, error_st;
float xrat, yrat, cx, cy;
int mode_color_table[7] = { 9, 10, 11, 12, 13, 14, 15 };

void init_window(), do_resize(), input(), draw_screen(), quit(), clear_screen();
void draw_mode(), paint_mode(), stamp_mode(), effect_mode(), erase_mode();
void midi_initialize(), midi_event_generate(), midi_transmit(), midi_tx_3byte();

main(int argc, char **argv)
{
```

```
    Arg args[20];
    int n;
    Widget glw, toplevel, form, frame, pushb[10];
    GLXconfig glxConfig [] = { { GLX_NORMAL, GLX_RGB, FALSE },
                { GLX_NORMAL, GLX_DOUBLE, TRUE }, { 0,0,0 }, };
    long background[] = { 0, 0, 0 };

    toplevel = XtAppInitialize(&app_context, "", (XrmOptionDescList)NULL , 0, (Cardinal*)&argc,
            (String*)argv, (String*)NULL, (ArgList)NULL, 0);
    n = 0;
    XtSetArg(args[n], XtNwidth, 900); n++;
    XtSetArg(args[n], XtNheight, 650); n++;
    form = XtCreateManagedWidget("", xmFormWidgetClass, toplevel, args, n);
    n = 0;
    XtSetArg(args[n], XmNbottomAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNrightAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNbottomOffset, 10); n++;
    XtSetArg(args[n], XmNrightOffset, 40); n++;
    pushb[0] = XtCreateManagedWidget("Quit", xmPushButtonWidgetClass, form, args, n);
    XtAddCallback(pushb[0], XmNactivateCallback, quit,0);
    n = 0;
    XtSetArg(args[n], XmNbottomAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNleftAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNbottomOffset, 10); n++;
    XtSetArg(args[n], XmNleftOffset, 40); n++;
    pushb[1] = XtCreateManagedWidget("Draw", xmPushButtonWidgetClass, form, args, n);
    XtAddCallback(pushb[1], XmNactivateCallback, draw_mode,0);
    n = 0;
    XtSetArg(args[n], XmNbottomAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNleftAttachment, XmATTACH_WIDGET); n++;
    XtSetArg(args[n], XmNbottomOffset, 10); n++;
    XtSetArg(args[n], XmNleftWidget, pushb[1]); n++;
    XtSetArg(args[n], XmNleftOffset, 30); n++;
    pushb[2] = XtCreateManagedWidget("Paint", xmPushButtonWidgetClass, form, args, n);
    XtAddCallback(pushb[2], XmNactivateCallback, paint_mode,0);
    n = 0;
    XtSetArg(args[n], XmNbottomAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNleftAttachment, XmATTACH_WIDGET); n++;
    XtSetArg(args[n], XmNbottomOffset, 10); n++;
    XtSetArg(args[n], XmNleftWidget, pushb[2]); n++;
    XtSetArg(args[n], XmNleftOffset, 30); n++;
    pushb[3] = XtCreateManagedWidget("Stamp", xmPushButtonWidgetClass, form, args, n);
    XtAddCallback(pushb[3], XmNactivateCallback, stamp_mode,0);
    n = 0;
    XtSetArg(args[n], XmNbottomAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNleftAttachment, XmATTACH_WIDGET); n++;
    XtSetArg(args[n], XmNbottomOffset, 10); n++;
    XtSetArg(args[n], XmNleftWidget, pushb[3]); n++;
    XtSetArg(args[n], XmNleftOffset, 30); n++;
    pushb[4] = XtCreateManagedWidget("Effect", xmPushButtonWidgetClass, form, args, n);
    XtAddCallback(pushb[4], XmNactivateCallback, effect_mode,0);
    n = 0;
    XtSetArg(args[n], XmNbottomAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNleftAttachment, XmATTACH_WIDGET); n++;
    XtSetArg(args[n], XmNbottomOffset, 10); n++;
    XtSetArg(args[n], XmNleftWidget, pushb[4]); n++;
    XtSetArg(args[n], XmNleftOffset, 30); n++;
    pushb[5] = XtCreateManagedWidget("Erase", xmPushButtonWidgetClass, form, args, n);
    XtAddCallback(pushb[5], XmNactivateCallback, erase_mode,0);
    n = 0;
    XtSetArg(args[n], XtNx, 30); n++;
    XtSetArg(args[n], XtNy, 30); n++;
    XtSetArg(args[n], XmNtopAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNleftAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNrightAttachment, XmATTACH_FORM); n++;
    XtSetArg(args[n], XmNbottomAttachment, XmATTACH_WIDGET); n++;
    XtSetArg(args[n], XmNtopOffset, 20); n++;
    XtSetArg(args[n], XmNleftOffset, 20); n++;
    XtSetArg(args[n], XmNrightOffset, 20); n++;
    XtSetArg(args[n], XmNbottomWidget, pushb[0]); n++;
    XtSetArg(args[n], XmNbottomOffset, 10); n++;
    frame = XtCreateManagedWidget("", xmFrameWidgetClass, form, args, n);
    n = 0;
    XtSetArg(args[n], GlxNglxConfig, glxConfig); n++;
    glw = XtCreateManagedWidget("", glxMDrawWidgetClass, frame, args, n);
    XtAddCallback(glw, GlxNginitCallback, init_window, 0);
    XtAddCallback(glw, GlxNinputCallback, input, 0);
    XtAddCallback(glw, GlxNresizeCallback, do_resize, 0);

    WidgetBackgroundToGlC3i(glw,background);
    XtRealizeWidget(toplevel);
    installColormap(toplevel,glw);
    XtAppMainLoop(app_context);
}

static void quit(Widget w, caddr_t client_data, caddr_t call_data)
{
    midi_tx_3byte( 160, 127, 127 );
    midi_tx_3byte( 160, 127, 0 );
    MIclose( midi_port );
    MIfreeport( midi_port );
    exit (0);
}

static void init_window(Widget w, caddr_t client_data, GlxDrawCallbackStruct *call_data)
{
    Matrix C_spline={{-0.5,1.5,-1.5,0.5},{1.0,-2.5,2.0,-0.5},{-0.5,0.0,0.5,0.0},{0.0,1.0,0.0,0.0},};
    ortho2( WORLD_XMIN, WORLD_XMAX, WORLD_YMIN, WORLD_YMAX );
    defbasis( 100, C_spline );
    curveprecision( 20 );
    curvebasis( 100);
    makeobj(MOUSE_OBJ);
        color(WHITE); rectf(-4.0,-6.0,4.0,6.0);
```

```
        color(BLACK); rect(-4.0,-6.0,4.0,6.0); rect(-2.75,0.0,-1.25,4.5);
        rect(-0.75,0.0, 0.75,4.5); rect( 1.25,0.0, 2.75,4.5);
        color(GREY); rectf(-3.0,-6.0,5.0,-7.0); rectf(4.0,-7.0,5.0,5.0);
    closeobj();
    button[0] = button[1] = button[2] = 0;
    do_resize(w,client_data,call_data);
    srandom(1);
    color_no = 6;
    action_mode = error_st = 0;
    midi_initialize();
    midi_tx_3byte( 159, 1, 127 );   /* to Draw */
    midi_tx_3byte( 159, 1, 0 );
}

static void do_resize(Widget w, caddr_t client_data, GlxDrawCallbackStruct *call_data)
{
    GLXwinset(XtDisplay(w), XtWindow(w));
    xmin = 0; xmax = (Screencoord)call_data->width - 1;
    ymin = 0; ymax = (Screencoord)call_data->height - 1;
    xrat = (WORLD_XMAX - WORLD_XMIN) / (float)(xmax - xmin);
    yrat = (WORLD_YMAX - WORLD_YMIN) / (float)(ymax - ymin);
    viewport( 0, xmax, 0, ymax );
    mx = xmax / 2; my = ymax / 2;
    draw_screen(w);
}

static void midi_initialize()
{
    MIconfig *c;
    c = MInewconfig();
    midi_port = MInewport();
    if( MIopen( midi_port, "rw", &c ) < 0 ) exit(-1);
}

static void midi_transmit( int status, int channel, int keyno, int velocity )
{
    midi_set_status( midi, ( status & 0xf0 ) );
    midi_set_channel( midi, ( channel & 0x0f ) );
    midi_set_keyno( midi, ( keyno & 0x7f ) );
    midi_set_velocity ( midi, ( velocity & 0x7f ) );
    if( MIsend( midi_port, midi, 1 ) < 0 ) exit(-1);
}

static void midi_tx_3byte( int status_byte, int keyno, int velocity )
{
    midi_transmit( ( status_byte & 0xf0 ), ( status_byte & 0x0f ), keyno, velocity );
}

static void draw_screen(Widget w)
{
    static Coord geom[4][3] = { { WORLD_XMIN, 6.0-LEN, 0.0 }, { 0.0, 6.0, 0.0 },
                { WORLD_XMIN, WORLD_YMAX, 0.0 }, { 0.0, WORLD_YMAX+LEN, 0.0 }, };

    GLXwinset(XtDisplay(w), XtWindow(w));
    color( mode_color_table[color_no] );
    clear();
    color(BLACK);
    geom[0][1] = cy + 6.0 - LEN;
    geom[1][0] = cx;
    geom[1][1] = cy + 6.0;
    geom[3][0] = cx;
    crvn(4,geom);
    pushmatrix();
        translate( cx, cy, 0.0 );
        callobj(MOUSE_OBJ);
        if ( button[0] == 1 ) rectf(-2.75,0.0,-1.25,4.5);
        if ( button[1] == 1 ) rectf(-0.75,0.0, 0.75,4.5);
        if ( button[2] == 1 ) rectf( 1.25,0.0, 2.75,4.5);
    popmatrix();
    swapbuffers();
}

static void clear_screen(Widget w)
{
    GLXwinset(XtDisplay(w), XtWindow(w));
    color( mode_color_table[color_no] );
    clear();
    swapbuffers();
    error_st = 0;
}

static void draw_mode(Widget w, caddr_t client_data, caddr_t call_data)
{
    int d, p=0;
    if( action_mode == 0 ) p = 20;
    else error_st == 0;
    d = 1 + p + random() % 16;
    midi_tx_3byte( 159, d, 127 );               /* to Draw */
    midi_tx_3byte( 159, d, 0 );
    action_mode = 0;
    color_no = (color_no+1) % 6;
    clear_screen(w);
}

static void paint_mode(Widget w, caddr_t client_data, caddr_t call_data)
{
    midi_tx_3byte( 159, 60, 127 );              /* to Paint */
    midi_tx_3byte( 159, 60, 0 );
    action_mode = 1;
    color_no = 6;
    clear_screen(w);
}
```

```
static void stamp_mode(Widget w, caddr_t client_data, caddr_t call_data)
{
    int d, p=0;
    if( action_mode == 2 ) p = 20;
    else error_st = 0;
    d = 71 + p + random() % 10;
    midi_tx_3byte( 159, d, 127 );          /* to Stamp */
    midi_tx_3byte( 159, d, 0 );
    action_mode = 2;
    color_no = (color_no+1) % 6;
    clear_screen(w);
}

static void effect_mode(Widget w, caddr_t client_data, caddr_t call_data)
{
    int d;
    d = random() % 3;
    midi_tx_3byte( 159, 121 + d, 127 );        /* Effect */
    midi_tx_3byte( 159, 121 + d, 0 );
    action_mode = 3;
    color_no = 6;
    clear_screen(w);
}

static void erase_mode(Widget w, caddr_t client_data, caddr_t call_data)
{
    int d;
    d = random() % 3;
    midi_tx_3byte( 159, 65 + d, 127 );         /* Erase */
    midi_tx_3byte( 159, 65 + d, 0 );
    action_mode = 3;
    color_no = 6;
    clear_screen(w);
}

static void input(Widget w, caddr_t client_data, GlxDrawCallbackStruct *call_data)
{
    int d, new_x, new_y, in_out = 0;

    mx = call_data->event->xbutton.x;
    my = call_data->event->xbutton.y;
    cx = (float)(mx - xmin) * xrat;
    cy = (float)(ymax - ymin - my) * yrat;
    new_x = 20 + (int)( cx * 0.4 );
    new_y = 20 + (int)( cy * 0.4 );
    switch(call_data->event->type){
        case ButtonPress:
            switch(call_data->event->xbutton.button){
                case Button1:
                    button[0] = 1;
                    if( action_mode == 1 ){
                        d = 41 + ( random() % 16 );
                        midi_tx_3byte( 159, d, 127 );
                        midi_tx_3byte( 159, d, 0 );
                        color_no = (color_no+1) % 6;
                    }
                    else if( action_mode == 0 ){
                        midi_tx_3byte( 157, new_x, new_y );
                        midi_tx_3byte( 157, new_x, 0 );
                    }
                    else if( action_mode == 2 ){
                        midi_tx_3byte( 158, new_x, new_y );
                        midi_tx_3byte( 158, new_x, 0 );
                        color_no = (color_no+1) % 6;
                    }
                    break;
                case Button2:  button[1] = 1;  break;
                case Button3:  button[2] = 1;  break;
            }
            draw_screen(w);
            break;
        case ButtonRelease:
            switch(call_data->event->xbutton.button){
                case Button1:  button[0] = 0;  break;
                case Button2:  button[1] = 0;  break;
                case Button3:  button[2] = 0;  break;
            }
            draw_screen(w);
            break;
        case MotionNotify:
            if (call_data->event->xmotion.state){
                if( button[0] == 1 ){
                    if( (cx<0.0)||(cx>200.0)||(cy<0.0)||(cy>200.0) ) in_out = 1;
                    if( (error_st == 0) && (in_out == 1) ){
                        midi_tx_3byte( 157, 120, 127 );
                        midi_tx_3byte( 157, 120, 0 );
                        error_st = 1;
                    }
                    else if( in_out == 0 ){
                        if( action_mode == 0 ){
                            midi_tx_3byte( 157, new_x, new_y );
                            midi_tx_3byte( 157, new_x, 0 );
                        }
                        else if( action_mode == 2 ){
                            midi_tx_3byte( 158, new_x, new_y );
                            midi_tx_3byte( 158, new_x, 0 );
                        }
                        if( error_st == 1 ) error_st = 0;
                    }
                }
                draw_screen(w);
            }
            break;
```

```
    }
}
```

関連して、この技術をメディアアートにおけるグラフィクス制御に応用する、というテーマでの研究を行い、情報処理学会音楽情報科学研究会において研究発表を行うことができた。添付資料(2)がである。

(2)インタラクティブ系とセンサ機構の開発実験

・マウス以外のコンテンツ制作支援ヒューマンインターフェースについて実験・検討を行った

・具体的なセンサとして、［加速度センサ］［ジャイロセンサ］［CCDカメラ］［ゲージセンサ］［静電センサ］等を検討し、実際にいくつかを製作・実験した

関連して、この技術をメディアアートのヒマンインターフェースに応用する、というテーマでの研究を行い、情報処理学会全国大会、および情報処理学会音楽情報科学研究会において研究発表を行うことができた。添付資料(1)(3)(4)がその論文である。

(3)Javaアプレット／HTMLへの自動変換機構の開発実験

・オーサリング環境の出力をJavaアプレットとして実現するための(半)自動ツールの開発実験を行った

・これと同時にHTMLファイル化してホームページに盛り込むための(半)自動ツールの開発実験を行った

この部分については、研究申請当時からインターネット環境の社会的向上がもっとも劇的であった。つまりそれだけ市場および産業界の要請が強い、という分野である。実験では、C言語で記述したマイコンボードのプログラムを与えて、これを逆解析しつつJavaソースコードに自動変換するツールを試作した。以下がそのソースプログラムである。これはインターネットにて公開している。

```
#include <stdio.h>
#include <stdlib.h>

FILE *fp;
int ni[3000];
unsigned char m[1000000], n[3000][256];

void byte_code_printf( long address, long length ){
    int k, l, p1, p2, p3, mm, m1, m2, m3, m4;
    long pp, p4, p5;
    pp = address;
    p1 = m[pp+1]+256*m[pp]; pp += 2;
    p2 = m[pp+1]+256*m[pp]; pp += 2;
    printf(" , max_stack = %d , max_locals = %d , ", p1, p2 );
    p4 = m[pp+3]+256*(m[pp+2]+256*(m[pp+1]+256*m[pp])); pp += 4;
    printf("code_length = %d\n", p4);
    k = 0;
    while( k < p4 ){
                printf("\n\t\t\t%06X\t", k );
                mm = m[pp++];
                switch(mm){
                        default :   printf("?? %02X ??\t\t(..... not defined .....)", mm);
                                    break;
                        case 16:    m1 = m[pp++];
                                    printf("bipush\t%02X\t(Push one-byte signed integer)", m1);
                                    k += 1; break;
                        case 17:    m1 = m[pp++]; m2 = m[pp++];
                                    printf("sipush\t%04X\t(Push two-byte signed integer)", 256*m1+m2);
                                    k += 2; break;
                        case 18:    m1 = m[pp++];
                                    printf("ldc1\t%02X\t(Push item from constant pool)", m1);
                                    k += 1; break;
                        case 19:    m1 = m[pp++]; m2 = m[pp++];
                                    printf("ldc2\t%04X\t(Push item from constant pool)", 256*m1+m2);
                                    k += 2; break;
                        case 20:    m1 = m[pp++]; m2 = m[pp++];
                                    printf("ldc2w\t%04X\t(Push long or double from constant pool)", 256*m1+m2);
                                    k += 2; break;
                        case 1:     printf("aconst_null\t\t(Push null object reference)");
                                    break;
                        case 2:     printf("iconst_m1\t\t(Push integer constant -1)");
                                    break;
                        case 3: case 4: case 5: case 6: case 7: case 8:
                                    printf("iconst_%d\t\t(Push integer constant)", mm-3);
                                    break;
                        case 9: case 10:
                                    printf("lconst_%d\t\t(Push long integer constant)", mm-9);
                                    break;
```

```
case 11: case 12: case 13:
        printf("fconst_%d\t\t(Push single float)", mm-11);
        break;
case 14: case 15:
        printf("dconst_%d\t\t(Push double float)", mm-14);
        break;
case 21:    m1 = m[pp++];
        printf("iload\t%02X\t(Load integer from local variable)", m1);
        k += 1; break;
case 26: case 27: case 28: case 29:
        printf("iload_%d\t\t(Load integer from local variable)", mm-26);
        break;
case 22:    m1 = m[pp++];
        printf("lload\t%02X\t(Load long integer from local variable)", m1);
        k += 1; break;
case 30: case 31: case 32: case 33:
        printf("lload_%d\t\t(Load long integer from local variable)", mm-30);
        break;
case 23:    m1 = m[pp++];
        printf("fload\t%02X\t(Load single float from local variable)", m1);
        k += 1; break;
case 34: case 35: case 36: case 37:
        printf("fload_%d\t\t(Load single float from local variable)", mm-34);
        break;
case 24:    m1 = m[pp++];
        printf("dload\t%02X\t(Load double float from local variable)", m1);
        k += 1; break;
case 38: case 39: case 40: case 41:
        printf("dload_%d\t\t(Load double float from local variable)", mm-38);
        break;
case 25:    m1 = m[pp++];
        printf("aload\t%02X\t(Load object reference from local variable)", m1);
        k += 1; break;
case 42: case 43: case 44: case 45:
        printf("aload_%d\t\t(Load object reference from local variable)", mm-42);
        break;
case 54:    m1 = m[pp++];
        printf("istore\t%02X\t(Store integer into local variable)", m1);
        k += 1; break;
case 59: case 60: case 61: case 62:
        printf("istore_%d\t\t(Store integer into local variable)", mm-59);
        break;
case 55:    m1 = m[pp++];
        printf("lstore\t%02X\t(Store long integer into local variable)", m1);
        k += 1; break;
case 63: case 64: case 65: case 66:
        printf("lstore_%d\t\t(Store long integer into local variable)", mm-63);
        break;
case 56:    m1 = m[pp++];
        printf("fstore\t%02X\t(Store float into local variable)", m1);
        k += 1; break;
case 67: case 68: case 69: case 70:
        printf("fstore_%d\t\t(Store float into local variable)", mm-67);
        break;
case 57:    m1 = m[pp++];
        printf("dstore\t%02X\t(Store double float into local variable)", m1);
        k += 1; break;
case 71: case 72: case 73: case 74:
        printf("dstore_%d\t\t(Store double float into local variable)", mm-71);
        break;
case 58:    m1 = m[pp++];
        printf("astore\t%02X\t(Store object reference into local variable)", m1);
        k += 1; break;
case 75: case 76: case 77: case 78:
        printf("astore_%d\t\t(Store object reference into local variable)", mm-75);
        break;
case 132:   m1 = m[pp++]; m2 = m[pp++];
        printf("iinc\t%02X %02X\t(Increment local variable by constant)", m1, m2);
        k += 2; break;
case 196:   m1 = m[pp++];
        printf("wide\t%02X\t(Wider index for accessing local variables in load, store and increment)", m1);
        k += 1; break;
case 188:   m1 = m[pp++];
        printf("newarray\t%02X\t(Allocate new array) : ", m1);
        k += 1;
        switch(m1){
                case 4:     printf("array type = T_BOOLEAN");  break;
                case 5:     printf("array type = T_CHAR");             break;
                case 6:     printf("array type = T_FLOAT");            break;
                case 7:     printf("array type = T_DOUBLE");   break;
                case 8:     printf("array type = T_BYTE");             break;
                case 9:     printf("array type = T_SHORT");            break;
                case 10:    printf("array type = T_INT");              break;
                case 11:    printf("array type = T_LONG");             break;
        }
        break;
case 189:   m1 = m[pp++]; m2 = m[pp++];
        printf("anewarray\t%04X\t(Allocate new array of references to objects)", 256*m1+m2);
        k += 2; break;
case 197:   m1 = m[pp++]; m2 = m[pp++]; m3 = m[pp++];
        printf("multianewarray\t%04X %02X\t(Allocate new multi-dimensional array)", 256*m1+m2, m3);
        k += 3; break;
case 190:   printf("arraylength\t\t(Get length of array)");
        break;
case 46:    printf("iaload\t\t(Load integer from array)");
        break;
case 47:    printf("laload\t\t(Load long integer from array)");
        break;
case 48:    printf("faload\t\t(Load single float from array)");
        break;
case 49:    printf("daload\t\t(Load double float from array)");
        break;
case 50:    printf("aaload\t\t(Load object reference from array)");
```

```
                break;
case 51:        printf("baload\t\t(Load signed byte from array)");
                break;
case 52:        printf("caload\t\t(Load character from array)");
                break;
case 53:        printf("saload\t\t(Load short from array)");
                break;
case 79:        printf("iastore\t\t(Store into integer array)");
                break;
case 80:        printf("lastore\t\t(Store into long integer array)");
                break;
case 81:        printf("fastore\t\t(Store into single float array)");
                break;
case 82:        printf("dastore\t\t(Store into double float array)");
                break;
case 83:        printf("aastore\t\t(Store into object reference array)");
                break;
case 84:        printf("bastore\t\t(Store into signed byte array)");
                break;
case 85:        printf("castore\t\t(Store into character array)");
                break;
case 86:        printf("sastore\t\t(Store into short array)");
                break;
case 0:         printf("nop\t\t(Do nothing)");
                break;
case 87:        printf("pop\t\t(Pop top stack word)");
                break;
case 88:        printf("pop2\t\t(Pop top two stack words)");
                break;
case 89:        printf("dup\t\t(Duplicate top stack word)");
                break;
case 92:        printf("dup2\t\t(Duplicate top two stack words)");
                break;
case 90:        printf("dup_x1\t\t(Duplicate top stack word and put two down)");
                break;
case 93:        printf("dup2_x1\t\t(Duplicate top two stack words and put two down)");
                break;
case 91:        printf("dup_x2\t\t(Duplicate top stack word and put three down)");
                break;
case 94:        printf("dup2_x2\t\t(Duplicate top two stack words and put three down)");
                break;
case 95:        printf("swap\t\t(Swap top two stack words)");
                break;
case 96:        printf("iadd\t\t(Integer add)");
                break;
case 97:        printf("ladd\t\t(Long integer add)");
                break;
case 98:        printf("fadd\t\t(Single floats add)");
                break;
case 99:        printf("dadd\t\t(Double floats add)");
                break;
case 100:       printf("isub\t\t(Integer subtract)");
                break;
case 101:       printf("lsub\t\t(Long integer subtract)");
                break;
case 102:       printf("fsub\t\t(Single float subtract)");
                break;
case 103:       printf("dsub\t\t(Double float subtract)");
                break;
case 104:       printf("imul\t\t(Integer multiply)");
                break;
case 105:       printf("lmul\t\t(Long integer multiply)");
                break;
case 106:       printf("fmul\t\t(Single float multiply)");
                break;
case 107:       printf("dmul\t\t(Double float multiply)");
                break;
case 108:       printf("idiv\t\t(Integer divide)");
                break;
case 109:       printf("ldiv\t\t(Long integer divide)");
                break;
case 110:       printf("fdiv\t\t(Single float divide)");
                break;
case 111:       printf("ddiv\t\t(Double float divide)");
                break;
case 112:       printf("irem\t\t(Integer remainder)");
                break;
case 113:       printf("lrem\t\t(Long integer remainder)");
                break;
case 114:       printf("frem\t\t(Single float remainder)");
                break;
case 115:       printf("drem\t\t(Double float remainder)");
                break;
case 116:       printf("ineg\t\t(Integer negate)");
                break;
case 117:       printf("lneg\t\t(Long integer negate)");
                break;
case 118:       printf("fneg\t\t(Single float negate)");
                break;
case 119:       printf("dneg\t\t(Double float negate)");
                break;
case 120:       printf("ishl\t\t(Integer shift left)");
                break;
case 122:       printf("ishr\t\t(Integer arithmetic shift right)");
                break;
case 124:       printf("iushr\t\t(Integer logical shift right)");
                break;
case 121:       printf("lshl\t\t(Long integer shift left)");
                break;
case 123:       printf("lshr\t\t(Long integer arithmetic shift right)");
                break;
case 125:       printf("lushr\t\t(Long integer logical shift right)");
                break;
```

```
case 126:   printf("iand\t\t(Integer boolean AND)");
            break;
case 127:   printf("land\t\t(Long integer boolean AND)");
            break;
case 128:   printf("ior\t\t(Integer boolean OR)");
            break;
case 129:   printf("lor\t\t(Long integer boolean OR)");
            break;
case 130:   printf("ixor\t\t(Integer boolean XOR)");
            break;
case 131:   printf("lxor\t\t(Long integer boolean XOR)");
            break;
case 133:   printf("i2l\t\t(Integer to long integer conversion)");
            break;
case 134:   printf("i2f\t\t(Integer to single float)");
            break;
case 135:   printf("i2d\t\t(Integer to double float)");
            break;
case 136:   printf("l2i\t\t(Long integer to integer)");
            break;
case 137:   printf("l2f\t\t(Long integer to single float)");
            break;
case 138:   printf("l2d\t\t(Long integer to double float)");
            break;
case 139:   printf("f2i\t\t(Single float to integer)");
            break;
case 140:   printf("f2l\t\t(Single float to long integer)");
            break;
case 141:   printf("f2d\t\t(Single float to double float)");
            break;
case 142:   printf("d2i\t\t(Double float to integer)");
            break;
case 143:   printf("d2l\t\t(Double float to long integer)");
            break;
case 144:   printf("d2f\t\t(Double float to single float)");
            break;
case 145:   printf("int2byte\t\t(Integer to signed byte)");
            break;
case 146:   printf("int2char\t\t(Integer to char)");
            break;
case 147:   printf("int2short\t\t(Integer to short)");
            break;
case 153:   m1 = m[pp++]; m2 = m[pp++];
            printf("ifeq\t%04X\t(Branch if equal to 0)", 256*m1+m2);
            k += 2; break;
case 198:   m1 = m[pp++]; m2 = m[pp++];
            printf("ifnull\t%04X\t(Branch if null)", 256*m1+m2);
            k += 2; break;
case 155:   m1 = m[pp++]; m2 = m[pp++];
            printf("iflt\t%04X\t(Branch if less than 0)", 256*m1+m2);
            k += 2; break;
case 158:   m1 = m[pp++]; m2 = m[pp++];
            printf("ifle\t%04X\t(Branch if less than or equal to 0)", 256*m1+m2);
            k += 2; break;
case 154:   m1 = m[pp++]; m2 = m[pp++];
            printf("ifne\t%04X\t(Branch if not equal to 0)", 256*m1+m2);
            k += 2; break;
case 199:   m1 = m[pp++]; m2 = m[pp++];
            printf("ifnonnull\t%04X\t(Branch if not null)", 256*m1+m2);
            k += 2; break;
case 157:   m1 = m[pp++]; m2 = m[pp++];
            printf("ifgt\t%04X\t(Branch if greater than 0)", 256*m1+m2);
            k += 2; break;
case 156:   m1 = m[pp++]; m2 = m[pp++];
            printf("ifge\t%04X\t(Branch if greater than or equal to 0)", 256*m1+m2);
            k += 2; break;
case 159:   m1 = m[pp++]; m2 = m[pp++];
            printf("if_icmpeq\t%04X\t(Branch if integers equal)", 256*m1+m2);
            k += 2; break;
case 160:   m1 = m[pp++]; m2 = m[pp++];
            printf("if_icmpne\t%04X\t(Branch if integers not equal)", 256*m1+m2);
            k += 2; break;
case 161:   m1 = m[pp++]; m2 = m[pp++];
            printf("if_icmplt\t%04X\t(Branch if integer less than)", 256*m1+m2);
            k += 2; break;
case 163:   m1 = m[pp++]; m2 = m[pp++];
            printf("if_icmpgt\t%04X\t(Branch if integer greater than)", 256*m1+m2);
            k += 2; break;
case 164:   m1 = m[pp++]; m2 = m[pp++];
            printf("if_icmple\t%04X\t(Branch if integer less than or equal to)", 256*m1+m2);
            k += 2; break;
case 162:   m1 = m[pp++]; m2 = m[pp++];
            printf("if_icmpge\t%04X\t(Branch if integer greater than or equal to)", 256*m1+m2);
            k += 2; break;
case 148:   printf("lcmp\t\t(Long integer compare)");
            break;
case 149:   printf("fcmpl\t\t(Single float compare (-1 on NaN))");
            break;
case 150:   printf("fcmpg\t\t(Single float compare (1 on NaN))");
            break;
case 151:   printf("dcmpl\t\t(Double float compare (-1 on NaN))");
            break;
case 152:   printf("dcmpg\t\t(Double float compare (1 on NaN))");
            break;
case 165:   m1 = m[pp++]; m2 = m[pp++];
            printf("if_acmpeq\t%04X\t(Branch if object references are equal)", 256*m1+m2);
            k += 2; break;
case 166:   m1 = m[pp++]; m2 = m[pp++];
            printf("if_acmpne\t%04X\t()Branch if object references not equal", 256*m1+m2);
            k += 2;
case 167:   m1 = m[pp++]; m2 = m[pp++];
            printf("goto\t%04X\t(Branch)", 256*m1+m2);
            k += 2; break;
```

```
                case 200:   m1 = m[pp++]; m2 = m[pp++];
                            m3 = m[pp++]; m4 = m[pp++];
                            printf("goto_w\t%04X %04X\t(Branch always (wide index))", 256*m1+m2, 256*m3+m4);
                            k += 2; break;
                case 168:   m1 = m[pp++]; m2 = m[pp++];
                            printf("jsr\t%04X\t(Jump subroutine)", 256*m1+m2);
                            k += 2; break;
                case 201:   m1 = m[pp++]; m2 = m[pp++];
                            m3 = m[pp++]; m4 = m[pp++];
                            printf("jsr_w\t%04X %04X\t(Jump subroutine (wide index))", 256*m1+m2, 256*m3+m4);
                            k += 2; break;
                case 169:   m1 = m[pp++];
                            printf("ret\t%02X\t(Return from subroutine)", m1);
                            k += 1; break;
                case 209:   m1 = m[pp++]; m2 = m[pp++];
                            printf("ret_w\t%04X\t(Return from subroutine (wide index))", 256*m1+m2);
                            k += 2; break;
                case 172:   printf("ireturn\t\t(Return integer from function)");
                            break;
                case 173:   printf("lreturn\t\t(Return long integer from function)");
                            break;
                case 174:   printf("freturn\t\t(Return single float from function)");
                            break;
                case 175:   printf("dreturn\t\t(Return double float from function)");
                            break;
                case 176:   printf("areturn\t\t(Return object reference from function)");
                            break;
                case 177:   printf("return\t\t(Return (void) from procedure)");
                            break;
                case 202:   printf("breakpoint\t\t(Stop and pass control to breakpoint handler)");
                            break;
                case 170:   printf("tableswitch\t\t(Access jump table by index and jump)");
                            printf("\n\t\t\t\t\t\t**********");
                            break;
                case 171:   printf("lookupswitch\t\t(Access jump table by key match and jump)");
                            printf("\n\t\t\t\t\t\t**********");
                            break;
                case 181:   m1 = m[pp++]; m2 = m[pp++];
                            printf("putfield\t%04X\t(Set field in object)", 256*m1+m2);
                            k += 2; break;
                case 180:   m1 = m[pp++]; m2 = m[pp++];
                            printf("getfield\t%04X\t(Fetch field from object)", 256*m1+m2);
                            k += 2; break;
                case 179:   m1 = m[pp++]; m2 = m[pp++];
                            printf("putstatic\t%04X\t(Set static field in class)", 256*m1+m2);
                            k += 2; break;
                case 178:   m1 = m[pp++]; m2 = m[pp++];
                            printf("getstatic\t%04X\t(Get static field from class)", 256*m1+m2);
                            k += 2; break;
                case 182:   m1 = m[pp++]; m2 = m[pp++];
                            printf("invokevirtual\t%04X\t(Invoke instance method)", 256*m1+m2);
                            k += 2; break;
                case 183:   m1 = m[pp++]; m2 = m[pp++];
                            printf("invokenonvirt\t%04X\t(Invoke instance method, dispatching based on compile-time type)", 256*m1+m2);
                            k += 2; break;
                case 184:   m1 = m[pp++]; m2 = m[pp++];
                            printf("invokestatic\t%04X\t(Invoke a class (static) method)", 256*m1+m2);
                            k += 2; break;
                case 185:   m1 = m[pp++]; m2 = m[pp++];
                            m3 = m[pp++]; m4 = m[pp++];
                            printf("invokeinterf\t%04X %02X %02X\t(Invoke interface method)", 256*m1+m2, m3, m4);
                            k += 2; break;
                case 191:   printf("athrow\t\t(Throw exception or error)");
                            break;
                case 187:   m1 = m[pp++]; m2 = m[pp++];
                            printf("new\t%04X\t(Create new object)", 256*m1+m2);
                            k += 2; break;
                case 192:   m1 = m[pp++]; m2 = m[pp++];
                            printf("checkcast\t%04X\t(Make sure object is of given type)", 256*m1+m2);
                            k += 2; break;
                case 193:   m1 = m[pp++]; m2 = m[pp++];
                            printf("instanceof\t%04X\t(Determine if an object is of given type)", 256*m1+m2);
                            k += 2; break;
                case 194:   printf("monitorenter\t\t(Enter monitored region of code)");
                            break;
                case 195:   printf("monitorexit\t\t(Exit monitored region of code)");
                            break;
            }
            k++;
    }
    p1 = m[pp+1]+256*m[pp]; pp += 2;
    printf("\n\n\t\texception_table_length = %d", p1 );
    if( p1 != 0 ){
            printf("\n");
            for(k=0;k<p1;k++){
                    p2 = m[pp+1]+256*m[pp]; pp += 2;
                    printf("\n\t\t\t%08X  start_pc = %d , ", pp, p2 );
                    p2 = m[pp+1]+256*m[pp]; pp += 2;
                    printf("end_pc = %d , ", p2 );
                    p2 = m[pp+1]+256*m[pp]; pp += 2;
                    printf("handler_pc = %d , ", p2 );
                    p2 = m[pp+1]+256*m[pp]; pp += 2;
                    if( p2 != 0 ) printf("catch_type = %s", n[ni[p2]] );
                    else printf("catch_type = all exceptions");
            }
    }
    p1 = m[pp+1]+256*m[pp]; pp += 2;
    printf("\n\n\t\tattributes_count = %d", p1 );
    if( p1 != 0 ){
            printf("\n");
            for(k=0;k<p1;k++){
                    p2 = m[pp+1]+256*m[pp]; pp += 2;
                    printf("\n\t\t\t%08X  type = %s , ", pp, n[p2] );
```

```
                        p4 = m[pp+3]+256*(m[pp+2]+256*(m[pp+1]+256*m[pp])); pp += 4;
                        printf("attribute_length = %d\n", p4);
                        for(l=0;l<p4;l++){
                                if((l%20)==0) printf("\n\t\t\t\t\t");
                                printf("%02X ", m[pp++] );
                        }
                }
        }
}

void access_flag_printf(int para){
        switch(para){
                case 0x0001:            printf("ACC_PUBLIC"); break;
                case 0x0002:            printf("ACC_PRIVATE");          break;
                case 0x0004:            printf("ACC_PROTECTED");        break;
                case 0x0008:            printf("ACC_STATIC"); break;
                case 0x0010:            printf("ACC_FINAL");    break;
                case 0x0020:            printf("ACC_SYNCHRONIZED");     break;
                case 0x0040:            printf("ACC_VOLATILE");         break;
                case 0x0080:            printf("ACC_TRANSIENT");        break;
                case 0x0100:            printf("ACC_NATIVE");           break;
                case 0x0200:            printf("ACC_INTERFACE");        break;
                case 0x0400:            printf("ACC_ABSTRACT");         break;
                default:        printf("( no flag )");          break;
        }
}

void main(int argc,char **argv){
        int d, i, j, k, ct, p0, p1, p2, p3, address=0;
        long pp=0, count=0, p4, p5;
        unsigned char ss, st[20], d0, d1, d2, d3;
        st[16]=0;
        if( argc != 2 ){ printf("\n... target file name missing (;_;) ...\n\n"); exit(1); }
        else if( (fp=fopen( argv[1], "rb" ))==NULL ){ printf("\n... target file [ %s ] is not found (;_;) ...\n\n", argv[1]); exit(1); }
        printf("\nTarget Java File Name = %s\n", argv[1]);
        while( (d = fgetc(fp)) >= 0 ){
                if( ( address % 16 ) == 0 ){ printf("\n\t%08X  :  ", address); ct = 0; }
                ss = d & 0xff; printf("%02X  ",ss); m[count++] = ss;
                if( ( address % 16 ) == 7 ) printf("-  ");
                if( ( ss > 0x1f ) && ( ss < 0x7f ) ) st[ct++] = ss;
                else st[ct++] = '.';
                if( ( address++ % 16 ) == 15 ) printf("  %s", st);
        }
        fclose(fp);
        printf("\n\nTotal File Length = %d bytes.", count);
        p0 = m[pp+3]+256*(m[pp+2]+256*(m[pp+1]+256*m[pp]));
        printf("\n\n\t%08X  Magic Code [CAFEBABE] = %08X", pp, p0 );
        pp += 4;
        if( p0 != 0xcafebabe ){ printf("\n\n... target file is not Java (;_;) ...\n\n"); exit(1); }
        else printf("  --  OK (^_^)");
        printf("\n\n\t%08X  Version : major version %d , minor version %d", pp, m[pp+3]+256*m[pp+2], m[pp+1]+256*m[pp] );
        pp += 4;
        p1 = m[pp+1]+256*m[pp]; printf("\n\n\t%08X  Constant Pool : total number = %d\n", pp, p1 ); pp += 2;
        for(i=1;i<p1;i++){
                printf("\n\t\t%08X   constant_pool[%d] = ", pp, i);
                d0 = m[pp++]; printf("<%X> ",d0);
                switch(d0){
                        case 7:         p2 = m[pp+1]+256*m[pp]; pp += 2; ni[i] = p2;
                                        printf("CONSTANT_Class , name_index = %d", p2);
                                        break;
                        case 9:         p2 = m[pp+1]+256*m[pp]; pp += 2;
                                        p3 = m[pp+1]+256*m[pp]; pp += 2;
                                        printf("CONSTANT_Fieldref , class_index = %d", p2);
                                        printf(" , name_and_type_index = %d", p3);
                                        break;
                        case 10:        p2 = m[pp+1]+256*m[pp]; pp += 2;
                                        p3 = m[pp+1]+256*m[pp]; pp += 2;
                                        printf("CONSTANT_Methodref , class_index = %d", p2);
                                        printf(" , name_and_type_index = %d", p3);
                                        break;
                        case 11:        p2 = m[pp+1]+256*m[pp]; pp += 2;
                                        p3 = m[pp+1]+256*m[pp]; pp += 2;
                                        printf("CONSTANT_InterfaceMethodref , class_index = %d", p2);
                                        printf(" , name_and_type_index = %d", p3);
                                        break;
                        case 8:         p2 = m[pp+1]+256*m[pp]; pp += 2; ni[i] = p2;
                                        printf("CONSTANT_String , name_index = %d", p2);
                                        break;
                        case 3:         p4 = m[pp+3]+256*(m[pp+2]+256*(m[pp+1]+256*m[pp])); pp += 4;
                                        printf("CONSTANT_Integer , value = %d", p4);
                                        break;
                        case 4:         p4 = m[pp+3]+256*(m[pp+2]+256*(m[pp+1]+256*m[pp])); pp += 4;
                                        printf("CONSTANT_Float , value = %d", p4);
                                        break;
                        case 5:         p4 = m[pp+3]+256*(m[pp+2]+256*(m[pp+1]+256*m[pp])); pp += 4;
                                        p5 = m[pp+3]+256*(m[pp+2]+256*(m[pp+1]+256*m[pp])); pp += 4;
                                        printf("CONSTANT_Float , high = %d , low = %d", p4, p5); i++;
                                        break;
                        case 6:         p4 = m[pp+3]+256*(m[pp+2]+256*(m[pp+1]+256*m[pp])); pp += 4;
                                        p5 = m[pp+3]+256*(m[pp+2]+256*(m[pp+1]+256*m[pp])); pp += 4;
                                        printf("CONSTANT_Double , high = %d , low = %d", p4, p5); i++;
                                        break;
                        case 12:        p2 = m[pp+1]+256*m[pp]; pp += 2;
                                        p3 = m[pp+1]+256*m[pp]; pp += 2;
                                        printf("CONSTANT_NameAndType , name_index = %d", p2); ni[i] = p2;
                                        printf(" , signature_index = %d", p3);
                                        break;
                        case 1: case 2:
                                        p2 = m[pp+1]+256*m[pp]; pp += 2;
                                        if(d0==1) printf("CONSTANT_Utf8 , ");
                                        else printf("CONSTANT_Unicode , ");
                                        k = 0;
                                        for(j=0;j<p2;j++){
```

```
                                        d1 = m[pp++];
                                        if((d1&0x80)==0) n[i][k++] = d1;
                                        else{
                                                d2 = m[pp++];
                                                if((d1&0xe0)==0xc0){
                                                        n[i][k++] = (64*(d1&0x1f)+(d2&0x3f))/256;
                                                        n[i][k++] = (64*(d1&0x1f)+(d2&0x3f))%256;
                                                }
                                                else{
                                                        d3 = m[pp++];
                                                        n[i][k++] = (256*16*(d1&0x0f)+64*(d2&0x3f)+(d3&0x3f))/256;
                                                        n[i][k++] = (256*16*(d1&0x0f)+64*(d2&0x3f)+(d3&0x3f))%256;
                                                }
                                        }
                                }
                                n[i][k] = 0; printf("Data = %s",n[i]);
                                break;
                        default:        printf("\nFormat Error !!!\n");
                                break;
                }
        }
        p1 = m[pp+1]+256*m[pp]; pp += 2; printf("\n\n\t%08X  Access Flag = %04X --- ", pp-2, p1 );
        access_flag_printf( p1 );
        p1 = m[pp+1]+256*m[pp]; pp += 2; printf("\n\n\t%08X  This Class = %s", pp-2, n[ni[p1]] );
        p1 = m[pp+1]+256*m[pp]; pp += 2; printf("\n\n\t%08X  Super Class = %s", pp-2, n[ni[p1]] );
        p1 = m[pp+1]+256*m[pp]; pp += 2; printf("\n\n\t%08X  Interface Count = %d", pp-2, p1 );
        if( p1 != 0 ){
                printf("\n");
                for(i=0;i<p1;i++){
                        p2 = m[pp+1]+256*m[pp]; pp += 2;
                        printf("\n\t\t%08X  interface[%d] : Name = %s", pp-2, i, n[ni[p2]] );
                }
        }
        p1 = m[pp+1]+256*m[pp]; pp += 2; printf("\n\n\t%08X  Fields Count = %d\n", pp-2, p1 );
        for(i=0;i<p1;i++){
                printf("\n\t\t%08X   field[%d] : ", pp, i);
                p2 = m[pp+1]+256*m[pp]; pp += 2; access_flag_printf( p2 );
                p2 = m[pp+1]+256*m[pp]; pp += 2; printf(" , Name = %s", n[p2] );
                p2 = m[pp+1]+256*m[pp]; pp += 2; printf(" , Signature = %s", n[p2] );
                p2 = m[pp+1]+256*m[pp]; pp += 2; printf(" , Attribute = %d", p2 );
                if( p2 != 0 ){
                        printf("\n");
                        for(j=0;j<p2;j++){
                                p3 = m[pp+1]+256*m[pp]; pp += 2;
                                printf("\n\t\t\ttype = %s , ", n[p3] );
                                p4 = m[pp+3]+256*(m[pp+2]+256*(m[pp+1]+256*m[pp])); pp += 4;
                                printf("length = %d\n", p4);
                                for(k=0;k<p4;k++){
                                        if((k%25)==0) printf("\n\t\t\t\t");
                                        printf("%02X ", m[pp++] );
                                }
                        }
                }
        }
        p1 = m[pp+1]+256*m[pp]; pp += 2; printf("\n\n\t%08X  Methods Count = %d", pp-2, p1 );
        for(i=0;i<p1;i++){
                printf("\n\n\t\t%08X   method[%d] : ", pp, i);
                p2 = m[pp+1]+256*m[pp]; pp += 2; access_flag_printf( p2 );
                p2 = m[pp+1]+256*m[pp]; pp += 2; printf(" , Name = %s", n[p2] );
                p2 = m[pp+1]+256*m[pp]; pp += 2; printf(" , Signature = %s", n[p2] );
                p2 = m[pp+1]+256*m[pp]; pp += 2; printf(" , Attribute = %d", p2 );
                if( p2 != 0 ){
                        printf("\n");
                        for(j=0;j<p2;j++){
                                p3 = m[pp+1]+256*m[pp]; pp += 2;
                                printf("\n\t\t\ttype = %s , ", n[p3] );
                                p4 = m[pp+3]+256*(m[pp+2]+256*(m[pp+1]+256*m[pp])); pp += 4;
                                printf("length = %d", p4);
                                p5 = pp; pp += p4;
                                if( strcmp( n[p3], "Code" ) == 0 ) byte_code_printf( p5, p4 );
                                else if( strcmp( n[p3], "Exceptions" ) == 0 ) byte_code_printf( p5, p4 );
                        }
                }
        }
        p1 = m[pp+1]+256*m[pp]; pp += 2; printf("\n\n\t%08X  Attributes Count = %d", pp-2, p1 );
        if( p1 != 0 ){
                for(i=0;i<p1;i++){
                        p2 = m[pp+1]+256*m[pp]; pp += 2;
                        printf("\n\n\t\t%08X  attribute[%d] : type = %s , ", pp-2, i, n[p2] );
                        p4 = m[pp+3]+256*(m[pp+2]+256*(m[pp+1]+256*m[pp])); pp += 4;
                        p5 = pp;
                        printf("length = %d\n", p4);
                        for(k=0;k<p4;k++){
                                if((k%25)==0) printf("\n\t\t\t\t");
                                printf("%02X ", m[pp++] );
                        }
                        if( ( p4 == 2 ) && ( strcmp( n[p2], "SourceFile" ) == 0 ) ){
                                p3 = m[p5+1]+256*m[p5];
                                printf("\n\n\t\t\tSource File Name = %s", n[p3] );
                        }
                }
        }
        printf("\n\n..... (^_^) Java analyze completely finished. (^_^)\n\n");
        exit(0);
}
```

このプログラムはインターネットで公開した後に、「OS/2研究会」および「Lisp研究会」から移植のためにソースを提供するよう依頼があるなど評判となったが、インターネット環境の成長により、同様の機能を

持つソフトウェアが多数登場してきたために、成果としてはここに吸収され埋没する結果となった。

(4)ネットワーク利用のコラボレーション機能の開発実験

・マルチプロセスをTCP/IPでリアルタイムに結合するネットワーク環境の開発実験を行った
・インターネットを経由して本研究のホームページ制作支援環境を
共同利用して共同制作するためのプロトコルとドライバについて実験を行った

3. 研究結果の報告と検討

以上のような研究結果についてまとめとして考察する。成果としては、目的そのもののシステム開発という点では予算的制約および時間的制約のために、完成システムとして稼動するような完成品ソフトウェアには至らなかった。しかし、個々のテーマに関する研究は着実に進展し、関連学会発表やシステムの実験、さらに試作ソフトウェアのソースプログラムをインターネットで公開することで、関連領域の研究者の注目を得て交流することもできた。これも一つの成果と考えられる。

今後、さらに機会を得てこのテーマでの研究を進めていきたいと考えている。

添付・参考文献

(1)論文「[サイバー楽器]のシステムデザインについて」、
『平成12年度前期全国大会講演論文集2』（情報処理学会）、2000年

(2)論文「メディアアートにおける画像系の制御について」、
『情報処理学会研究報告Vol.2000,No.76 (2000-MUS-36)』（情報処理学会）、2000年

(3)論文「[メディア・インスタレーションを用いたインタラクティブ・パフォーマンスについて」、
『平成13年度前期全国大会講演論文集2』（情報処理学会）、2001年

(4)論文「新・筋電センサ"MiniBioMuse-III"とその情報処理」、
『情報処理学会研究報告Vol.2001,No.82 (2001-MUS-41)』（情報処理学会）、2001年

以上

2G－6

「サイバー楽器」のシステムデザインについて

長嶋洋一

Art & Science Laboratory

1. はじめに

インタラクティブアートのヒューマンインターフェースとして、またメディアインスタレーションの一形態として、新しいタイプの「楽器」を創出する、という研究領域に関連して、個別の特定の対象を離れた一般的なシステムデザインの視点から考察した。大きく、(1)従来からのアコースティック楽器を拡張したタイプのサイバー楽器、(2)新しいセンサやデザインによる新コンセプトのタイプのサイバー楽器、という二つの潮流について、システム技術的・ヒューマン情報処理的・音楽美学的などの見地から検討し、新しいサイバー楽器の創出に向けた指針を提案する。

2. 過去のICMCの検討

この領域での先行研究および世界の関連研究のサーベイとして、ICMC1991からICMC1999までのProceedingsから、「センサ」「インターフェース」「新楽器」「演奏支援システム」などの視点での発表を調べた([1]-[26])。ここでは、楽器メーカ等から提供される汎用の装置/センサへの不満によるオリジナル化への強い意欲、ホストPCに頼らずマイコンにより小型軽量化を図る指針、柔軟な視点で多種のセンサと多用なパターン抽出を試みている研究の流れを確認することができた。論文集はICMAから全て提供されているので、興味のある方は原論文をぜひ参照されたい。

3. システム技術的な検討

一言で「サイバー楽器」と言っても、(1)従来からのアコースティック楽器を拡張する、(2)新しいセンサやデザインによる新コンセプト、という二つの潮流がある。前者においては、民族楽器や伝統楽器の音響的な特性や演奏技法を尊重しつつ音楽表現の可能性を拡張する、というニーズ指向の視点があり、後者においてはVR技術に対応した新しいセンサ技術やマイコン技術の進展を適用する、というシーズ指向が強く見られた。

エレクトロニクス技術、あるいは情報処理技術としてのシステムデザインとして「サイバー楽器」を見た場合には、単なるセンシング出力やパターン認識出力を音楽演奏情報に単純にマッピングしてきた状況への反省が重要である。これは、MIBURIやBioMuseがあまり経済的にヒットしない理由として、ハイテクのセンサ部分だけでは音楽やアートの道具として十分でない、アナログ的なセンサ出力を規範的な演奏情報に画一的に離散化することによる自由度の消失、等として専門家が過去に指摘してきたことと対応している。今後のアプローチとしては、柔軟で自由なマッピング、というのがキーワードになると思われる。

4. ヒューマン情報処理的な検討

ヒューマンインターフェースとしての「サイバー楽器」は、センサボックス部分のハードということでなく、最終的にサウンドやビジュアルのマルチメディア情報を対話的にリアルタイム生成するシステム全体のソフトウェアまでを含めて捉える傾向にある。ここで、前述のサーベイから得られた情報処理的な課題について以下に列記する。

センシング入力のフロントエンドにおいては、工学的なセンサとしての検出可能性に限定されない柔軟な視点が求められる。人間の身体はかなりの適応力と柔軟性を持っており、音楽家が「楽器」として習熟する能力は非常に高いものがあり、要求水準も高度である。また、あらゆるセンサ出力を単純に遠隔伝送・統合するのでなく、フロントエンド付近にて高性能CPUで信号処理・パ

System Design of "Cyber-Instruments"
Yoichi Nagashima (*nagasm@computer.org*)
Art & Science Laboratory

ターン認識処理を行う手法も、パフォーマンス性や携帯性能として重要である。

インタラクティブアートやメディアインスタレーションにおいて重要な「対話性」については、システム内に構築するモデルと現実世界からのセンサ入力情報、そして出力されるVR情報との対応(広義のマッピング問題)が共通のテーマとなっている観がある。新規なコンセプトのサイバー楽器であればこの仮想現実空間のモデリングが命となり、従来楽器を拡張したサイバー楽器の場合には、伝承された元の楽器の持つ音楽空間の特長を生かしたモデリングが重要になると考えられる。

5. 音楽美学的な検討

これまでテクノロジー主導であった「サイバー楽器」は、これからいよいよ、音楽やアートとしての本来的な視点がリードすることになる模様であり、ICMCでは既にその端緒を見て取れる。システムの工学的な部分はブラックボックスの陰に隠れ、アイデアとコンテンツの勝負の時代である。ここに求められるものとしては、アートの創造を妨げないヒューマンなオーサリング環境、そして創作と実現とが統合化された支援環境である。

エレクトーンを自動演奏した機構むき出しロボットの世紀はもう終わる。新しいサイバー楽器を研究開発しようとする者は、メカやハイテクに人間を縛ることなく、最低限の教養として音楽が聴こえ、音楽を理解し、アートする心が必要とされる時代となるようである。ICMCにおける美学的テーマの論文発表をぜひ参照されたい。

6. おわりに

インタラクティブアートのヒューマンインターフェースとは、関連領域を広く深く持っている研究対象であることが確認できた。机上の空論でなく、実際のシステム構築やアートへの応用として具体的な進展とともに研究を進めていきたい。

参考文献

[1] Eric Johnstone (McGill Univ.) : "A MIDI Foot Controller - The PodoBoard" , Proceedings of 1991 ICMC, pp.123-126.

[2] David Keane, et al. (Queen's Univ.) : "The MIDI Baton III", Proceedings of 1991 ICMC, pp.541-544.

[3] Graziano, et al. (CNR) : "Light baton : A System for Conducting Computer Music Performance" , Proceedings of 1992 ICMC, pp.73-76.

[4] Lippold Haken, et al. (Univ. of Illinois) : "The Continuum : A Continuous Music Keyboard" , Proceedings of 1992 ICMC, pp.81-84.

[5] Randy C. Marchany et al. (Virginia Tech Computing Center) : "A Programmable MIDI Instrument Controller Emulating a Hammer Dulcimer" , Proceedings of 1992 ICMC, pp.89-92.

[6] Yoichi Nagashima : "Real-time control system for 'psuedo granulation'" , Proceedings of 1992 ICMC, pp.404-405.

[7] Tsutomu Kanamori et al. (LIST) : "Gesture Sensor in Virtual Performer" , Proceedings of 1993 ICMC, pp.127-129.

[8] Brad Cariou (Univ. of Calgary) : "The aXi0 MIDI Controller" , Proceedings of 1994 ICMC, pp.163-166.

[9] Stuart Favilla (La Trobe Univ.) : "The LDR Controller" , Proceedings of 1994 ICMC, pp.177-180.

[10] Russel Pinkston et al. (Univ of Texas at Austin) : "A Touch Sensitive Dance Floor/MIDI Controller" , Proceedings of 1995 ICMC, pp.224-225.

[11] Haruhiro Katayose et al. (LIST) : "An Environment for Interactive Art" , Proceedings of 1996 ICMC, pp.173-176.

[12] Nicola Orio (CSC) : "A Gesture Interface Controlled by the Oral Cavity" , Proceedings of 1997 ICMC, pp.141-144.

[13] Hideyuki Sawada, et al. (Waseda Univ.) : "Aounds in Hands - A Sound Modifier Datagloves and Twiddle Interface" , Proceedings of 1997 ICMC, pp.309-312.

[14] Teresa Marrin et al. (MIT Media Lab.) : "The Digital Baton : a Versatile Performance Instrument" , Proceedings of 1997 ICMC, pp.313-316.

[15] Yoichi Nagashima (ASL) : "Biosensorfusion : New Interfaces for Interactive Multimedia Art" , Proceedings of 1998 ICMC, pp.129-132.

[16] Teresa Marrin et al. (MIT Media Lab.) : "The 'Conductor's Jacket' : A Device for recording expressive musical gestures" , Proceedings of 1998 ICMC, pp.215-219.

[17] Mark A Bromwich et al. (Univ. of Hudders field) : " 'Bodycoder' : A sensor suit and vocal performance mechanism for real-time performance" , Proceedings of 1998 ICMC, pp.292-295.

[18] Niall Griffith et al. (Univ. of Limerick) : "LITEFOOT - A Floor space for recording dance and controlling media" , Proceedings of 1998 ICMC, pp.475-481.

[19] Shigeyuki Hirai et al. (LIST) : "Software Sensors for Interactive Digital Art" , Proceedings of 1998 ICMC, pp.514-517.

[20] Kai-Yuh Hsiao et al. (MIT Media Lab.) : "A New Continuous Multimodal Musical Controller Using Wireless Magnetic Tags" , Proceedings of 1999 ICMC, pp.24-27.

[21] Yoichi Nagashima (ASL) : " It's SHO time --- An Interactive Environment for SHO(Sheng) Performance" , Proceedings of 1999 ICMC, pp.32-35.

[22] Dan Trueman et al. (Prinston Univ.) : "BoSSA : The Deconstructed Violin Reconstructed" , Proceedings of 1999 ICMC, pp.232-239.

[23] Wayne Siegel et al. (DIEM) : "Composing for the Digital Dance Interface" , Proceedings of 1999 ICMC, pp.276-277.

[24] Joseph Paradiso et al. (MIT Media Lab.) : "Interactive Music for Instrumented Dancing Shoes" , Proceedings of 1999 ICMC, pp.453-456.

[25] http://nagasm.org/ASL/

[26] http://nagasm.org/hightech/

[27] http://www.ICMC2000.org/

音楽情報科学 36－6
（2000．8．5）

メディアアートにおける画像系の制御について

長嶋洋一*　中村文隆**
*静岡文化芸術大学/ASL　**神戸山手女子短期大学

インタラクティブなマルチメディア・アートやメディア・インスタレーションを実現するために、静止画や動画などの画像系を[センサ・MAX]系と連携させライブ制御する手法と問題点・課題について検討した。

A Study of Graphical Control in Media-Art

Y.Nagashima*/#　F.Nakamura**
*Shizuoka University of Art and Culture / Art & Science Laboratory
**Kobe Yamate College
nagasm@computer.org

This paper is intended as an investigation of some methods of controlling graphics part in computer music, media installations and interactive multimedia art. We have been producing many sensors, interfaces and interactive systems for computer music. Now in this study the main stress falls on controlling graphics in real-time and interactively with human performers. We will report some methods and discuss the problems.

1. はじめに

コンピュータ音楽を中心としたメディアアートに関するテーマの研究活動とともに、その具体的な応用を実験的に検証する意味で、実際にいろいろなインタラクティブ・マルチメディア作品を創作して公演・発表する活動を行っている(1-11)。実験的なシステムをリアルタイムパフォーマンスに応用することで、作品の中から新たな研究テーマや課題が出てくることも多い。最近ではグラフィックアーティストやダンサーとのコラボレーションによって、インタラクティブ・マルチメディア・アートとしての方向に展開している。本稿では特に、リアルタイムオーサリング環境"MAX"と結び付けて、グラフィクスをライブ制御する手法について手法について紹介するとともに、その問題点と課題について検討する。

2. 国内の従来研究

過去の情報処理学会音楽情報科学研究会研究報告(1993年4月以降)の中で「マルチメディアアートにおいてグラフィクス系をリアルタイム制御する」というような視点から従来研究を調べてみると、以下のようなアプローチが行われてきた。後藤ら(12)は、RMCPシステムのクライアントとしてSGI上にOpen-GLでリアルタイムCGシステムを実装した。長嶋(13)は、AMIGAコンピュータによるCG系とMAXのサウンド系とをMIDIによりインターフェースした。松田(14)らは、NeXTコンピュータにライブビデオ画像を取り込み、ISPW上のFTSでの転送を試みた。片寄(15)らは、簡易なアニメーションとしてMAXのPICTオブジェクトを利用した。高城(16)は、SGI Indigo2を用いたグラフィクス系をFTSによりISPWと結合させた。平野(17)らは、SGI O2とNTマシン上のPure Dataを組み合わせたシステムを構築した。また平野(18)らは、オーサリングソフト"Director"の外部拡張オブジェクトとして、Serial、HyperMIDI、MIDIXTRA等を利用してMIDI系とグラフィクスを連携させた。

大きく分類すれば、(1) Open-GLのリアルタイムCGをSGI、NT(Pd)、Linux等の環境にMIDI対応として実装する手法、そして、(2) Directorの外部拡張によりMIDIシステムとリンクさせる(ただし最近のシリアル非対応のMac (USB)では、DirectorのMIDI I/Fに問題あり)、という二つの手法が現状でのbetter solutionであると思われる。これは海外でも基本的には同じである。

3. 作品"Asian Edge"での事例

筆者の初めてのマルチメディア・パフォーマンス作品"Asian Edge" (3)では、performanceをセンシングするオリジナルセンサとともに、MIDI系とリンクしてリアルタイムにグラフィクス系を制御するためのオリジナルマシンを、作曲の一部として製作した。

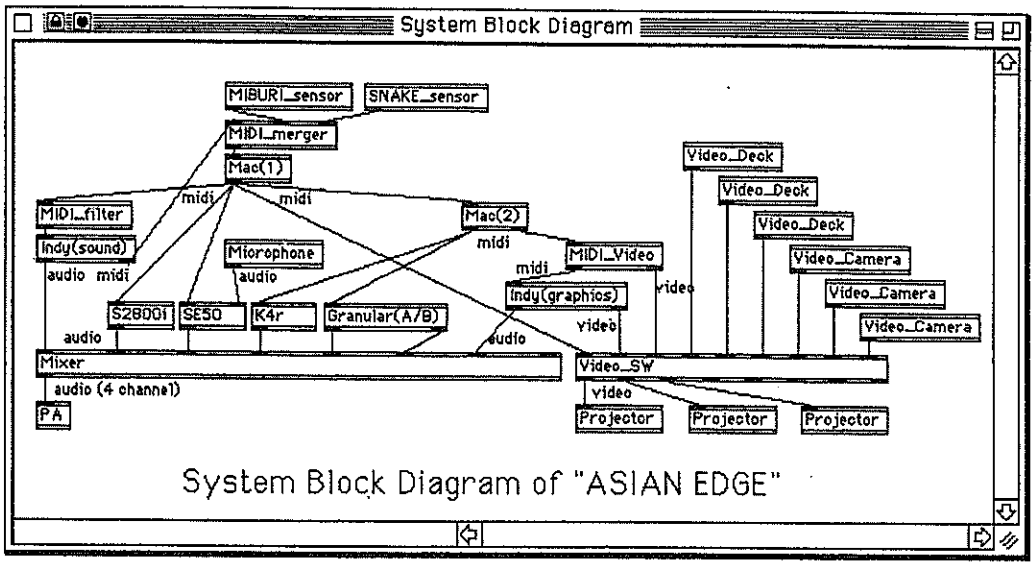


図 1. 作品 "Asian Edge"のシステムブロック図

図1のシステム図の中では"MIDI_Video"と名付けているこのマシンは、8入力8出力のビデオ信号マトリクススイッチLSIを、MIDI入力によってライブコントロールする(19)。フレームメモリ等に画像情報を蓄積しないのでアナログ信号切り替えのノイズ等が出るものの、ライブパフォーマンスをCCDカメラで取り込んだ画像などを瞬間的に切り替える時にはほとんどノイズが気にならず、むしろ高速レスポンスの重要性を確認することができた。

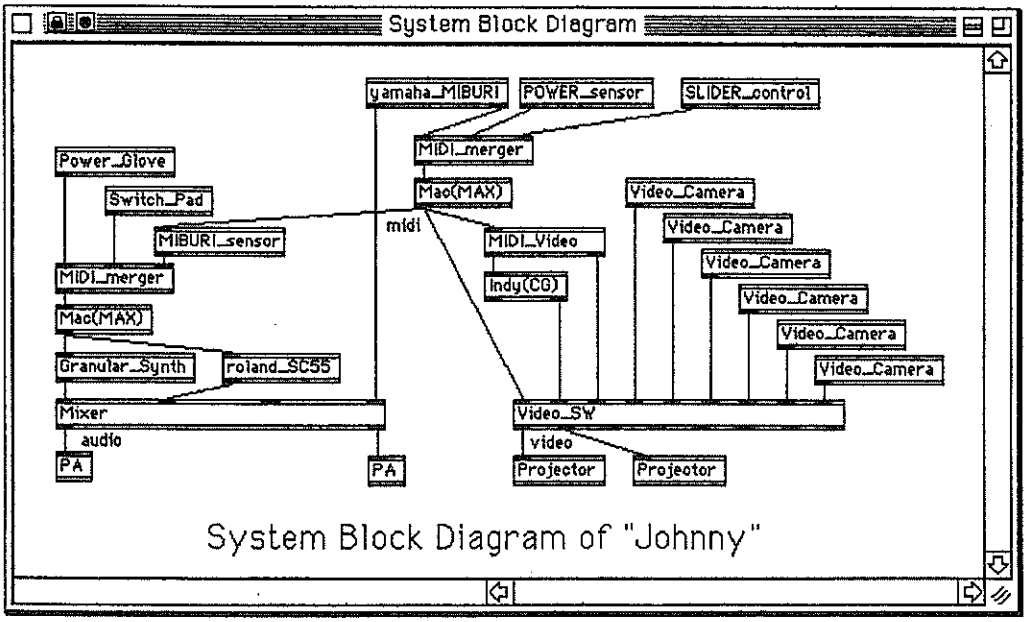


図 2. 作品 "Johnny"のシステムブロック図

4. 作品"Johnny"での事例

"Asian Edge"に続く作品"Johnny"でも、図2のように基本的には同じようなシステム構成を採用した。この作品では、ステージ上の3人のperformerからのライブ情報をマージしたセンサフュージョン情報をMAXに取り込み、サウンド系のMAXとグラフィクス系に分岐して送っている。技術的にポイントとなったのは、リアルタ

イム3D-CG生成のSGI上のソフトウェアに対して、サウンド系まで含めた多量のMIDIをそのまま送ってはいけない、という部分である。Uinx系としては唯一、MIDIの入出力がまともに機能するSGIであるが、出力はともかくMIDI入力に対して、IRIXは多量のトラフィックには耐えられず、コアダンプする間もなくカーネルからフリーズする場合があり、専用のプログラマブルMIDIフィルタを製作して、CGに必要な情報だけをフィルタリングする必要があった。

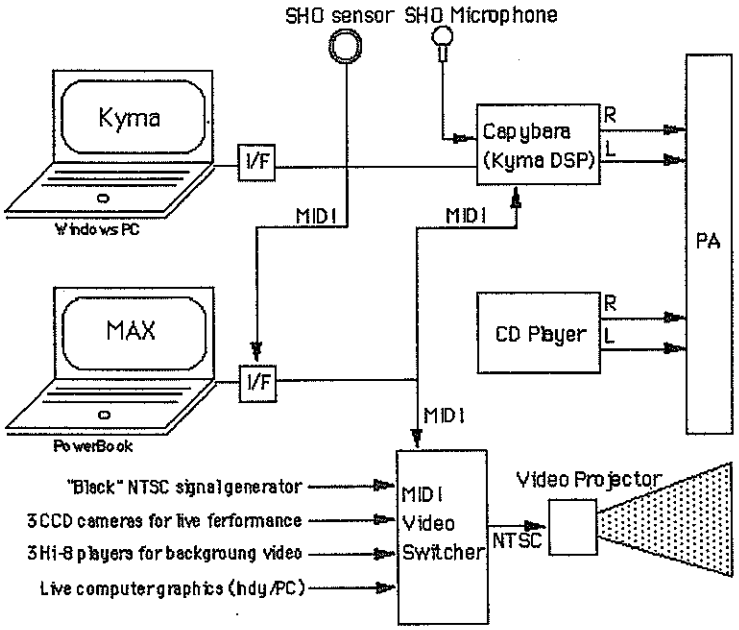


図 3. 作品 "Visional Legend"のシステムブロック図

5. 作品"Visional Legend"での事例

図3は、1998国際コンピュータ音楽フェスティバルで初演した作品"Visional Legend"のシステム図である。この作品のメインテーマは、雅楽の伝統楽器である「笙」の表現可能性を拡張するための笙ブレスセンサの活用である(19)。ここでも同じMIDIビデオスイッチャを利用しているが、グラフィクスのソースとして別にパソコンを用意して、あらかじめ制作した静止画映像をスライドショー表示させたものも映像ソースの一つとして使用した。また、小型ビデオエフェクタ装置(Videonics社)のパネル操作をMIDI経由でMAXライブ制御するシステムも制作したが最終的には公演には使わなかった。この作品では、笙のライブ音響に対してKymaによりリアルタイム音響信号処理を行う、という部分にもっとも重点を置いたが、Kymaの信号処理アルゴリズムのパラメータを全てMAXからMIDI経由で送ることで、サウンド系とグラフィクス系はMAX上で対等にプログラミングできた。これは、筆者がMAXを作曲のベースとする最大の理由である。図4はこの作品の笙のサウンドを処理するKymaのパッチ、図5は、グラフィクスを含む作品全体を制御するMAXパッチである。

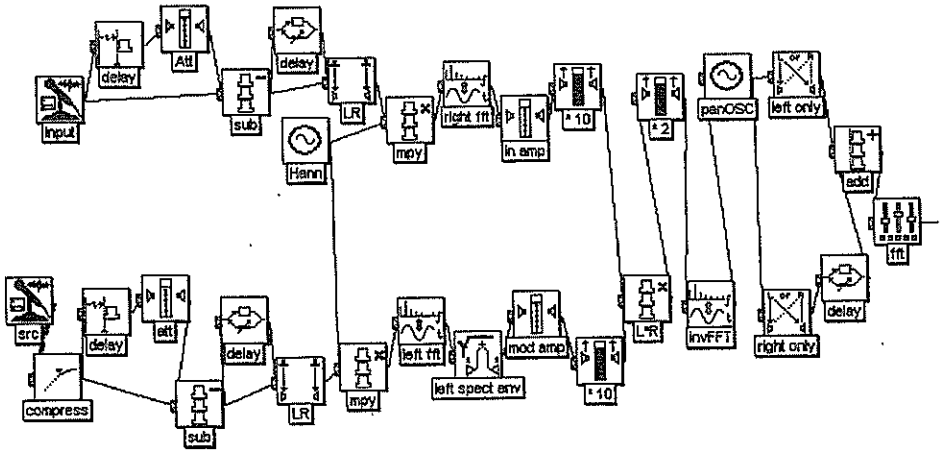


図 4. 作品 "Visional Legend"のためのKymaパッチ

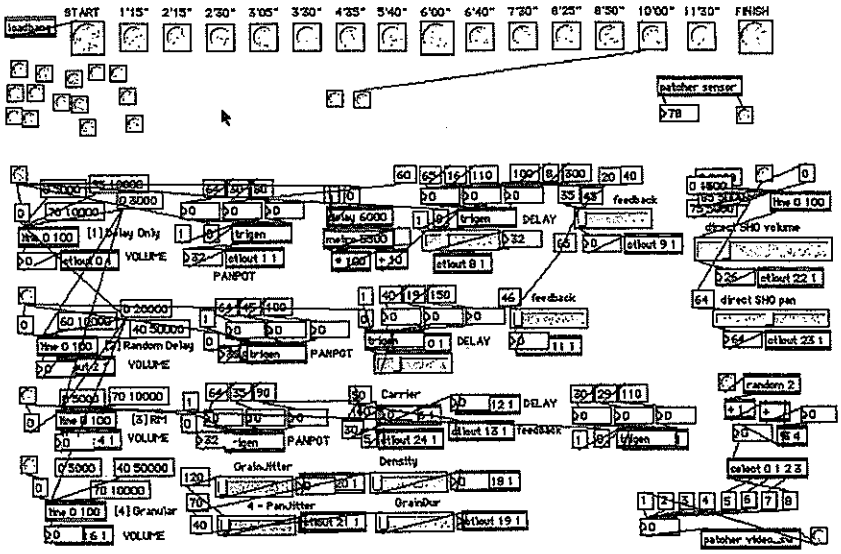


図 5. 作品 "Visional Legend"のためのMAXパッチ

6. 作品"森海"での事例

図6は、2000年5月28日に静岡文化芸術大学公開デー[20]において発表した、体験型インタラクティブ・メディアインスタレーション作品「森海(しんかい)」のシステムブロック図である。この企画は、デザイン学部技術造形学科の教員と一部選抜学生とのコラボレーションにより、わずか3週間でセンサからCGまで制作するという初めての企画だったため、サウンド系とグラフィクス系の仕掛けの部分にはあまり冒険ができない状況で、同じMIDIビデオスイッチャを使用した。ただしディスプレイ系として、学内の8台の大型プラズマディスプレイを並べるという壮観な展示空間を実現できた点が、ビデオ系統のスイッチマトリクスという機能を直接的に表現する格好の機会となった。学生がPhotoshopで制作したCG静止画は、Directorによって各種のモードでページ推移するスライドショーとして出力したものをビデオに記録してエンドレス再生し、3系統のライブCCDカメラ画像とともにMIDIビデオスイッチャの入力として利用した。背景音響パートとセンサ対応イベント音響も、全てMAXによって生成した。

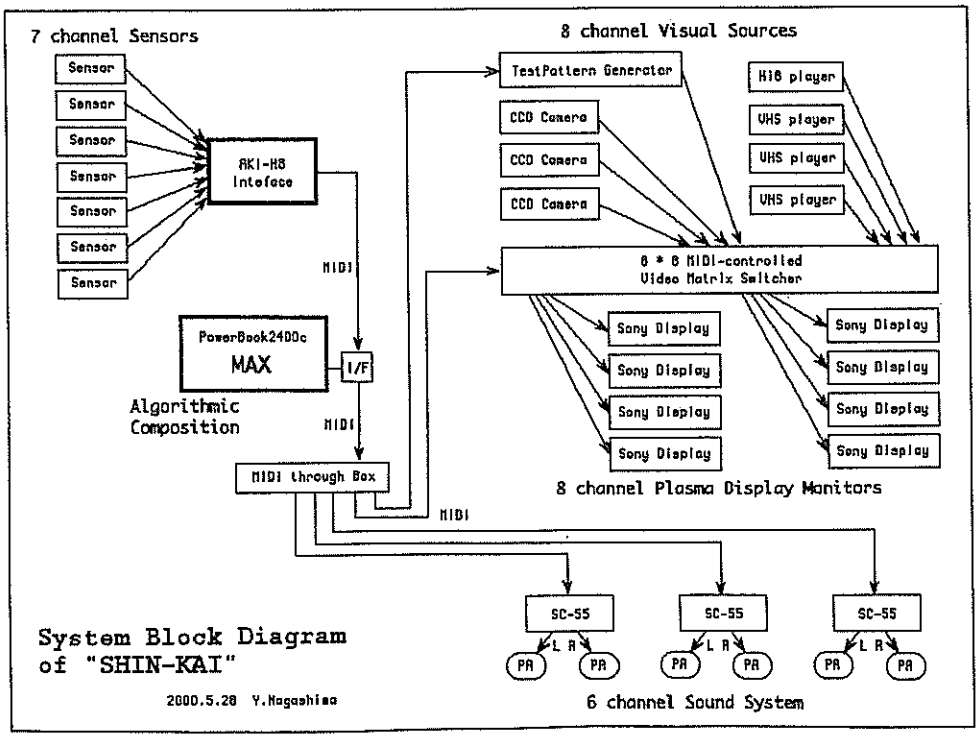


図 6. インスタレーション作品 "森海"のシステムブロック図

7. 画像系の制御に求められる性能

MAXをベースとしセンサ系とMIDIでインターフェースする、というインタラクティブアートやメディアインスタレーションの場合、画像系のリアルタイム制御に求められる機能・性能・注意点として、以下のようなものがある。

(1) レスポンス性能[21]

音源のレスポンス(MIDI情報を受け取ってから実際のサウンドが生成されるまでの遅延)と同等であるためには、いくら悪くても100msec、できれば30-50msec以内で画像を更新したい。これはかなり厳しい要求であるが、特にパフォーマンスの要素としては妥協できない性能であり、筆者の手法の選択においては重要である。

(2) 連続動作性

テープというメディアを使用することで、連続的に展示する場合に「巻き戻し」という必要性が生じる(無人化が困難)ことは重大な問題となる。また、2週間とか1年間という長期間の展示の場合には、テープをCD-RやDVDやHDという連続アクセスメディア化しても、基本的に「メカの劣化と寿命」という問題点がある。演奏時間として10分とか15分と限定されている場合には、Hi-8テープ等で問題なく実現できる。

(3) OSのハングアップ

UnixやMS-DOSなどに比べて、ブラックボックスの陰でこっそり不十分なメモリ管理をしているWindows系のOS(95,98,NT,...)では、連続無人運転のインスタレーションを構築することがあまりに危険である、という議論はとても多い。ネットワークサーバOSと同じ議論である。突然に爆弾マークを出すMacOSも同様である。

(4) 画質と機器

コンピュータのRGB出力そのものをRGB対応の大型ディスプレイやプロジェクタに接続する、というものを最上画質の[A]ランクとすれば、[B]パソコン出力をスキャンコンバータでSビデオに変換、[C]パソコン出力をスキャンコンバータでNTSCビデオに変換、という各種の画質が存在する。グラフィクスをスイッチングした場合には、もっとも画質の悪い機器に応じたレベルに全体が低下することになる。経験上は、XGAだSVGAだVGAだ、という画素数の性能はあまり気にならないことが多い。

(5) 静止画の画像切り替え

多数の静止画を動的に切り替えて表示する「スライドショー」はもっとも単純なアニメーションであるが、この画面切り替えにDirectorのような多数の機能を持ったものはあまり見かけない。フェイドイン/アウトの機能もブラックアウト期間が気になる。あるいは、切り替えのタイミングとしてタイマーとマウスクリックだけでなく、せめてシリアル入力のトリガがあれば、と筆者は思うのだが見かけない。何か情報があればお知らせいただければ幸いである。

8. 考えられる画像制御手法と検討

前節のような検討とともに、筆者がメディアアートのためのリアルタイム画像制御手法としてリストアップしたのは以下のようなものである。この中には、既に実現したもの、現在実験中のもの、実験するつもりで準備しているもの、おそらく自分ではやらずに誰かからの情報を待っているもの、等が混在しているが、実現手法を問題とするのでなく、作品をいかに実現するか、というゴールを忘れずに、いろいろとチャレンジしていきたいと考えている。

(1) MAX内でQuickTimeムービーのオブジェクトを使う。マシンの性能とQTムービーの画像サイズ等に依存する。本来の制御系であるMAX部分がQTムービー処理に足を取られるようでちょっと気持ちが悪い気がする。

(2) MAX内でPICTオブジェクトによりリアルタイムグラフィクスを行う。PICTサイズが大きくなると、いちいちHDからのロード時間が遅延として気になったが、MAXバッチからPICTまで全てRAMディスク上に置くと、この問題はかなり改善された。次作ではこの手法を試してみるつもりである。

(3) MAXのDrawオブジェクトによる描画。Open-GLのCGソフトを書いてしまうとあまりに物足りないが、実験的な試作においては有効かもしれない。

(4) ハードによるパターンジェネレータとAKI-H8等によるMIDI制御化。秋葉原のビデオ信号生成キット等をカードマイコンでMIDI化する手法で、一部(ブラック信号)で実際に使用中。

(5) 市販ビデオエフェクタ機器をAKI-H8等でMIDI化。すでに実現しているが、パネルスイッチの部分をカードマイコンによりMIDI制御しただけでは、実際のライブ操作としてはあまりに遅くて使えないことが判明。インスタ向き。

(6) MIDI制御ビデオスイッチャをAKI-H8等でオリジナル製作。既に多数の作品で使用中。次のバージョンとしては、入出力のより多系列化が課題となっている。LSIは比較的安価。

(7) MIDIでDirectorを制御する。ADBやモデムポートのあったMacでは比較的容易にXtrasが使えるようだが、USB/IEEE1394ベースのG3/G4ではなかなか問題があるようである。現在のところはまだ使えていない。

(8) RS232CでDirectorを制御する。Directerの内部的なXtrasのSerialPortを使うために、別にAKI-H8等で「MIDIとRS232Cの相互変換マシン」を製作した。しかしUSBのMacでは、「USBとシリアルとの相互変換アダプタ」との相性が不調で、これも現在まで使えていない。

(9) SGIでOpen-GLでMIDI対応のCGソフトを書く。これは実績あり。Drwa系のCGにはとても有効。問題はムービーや実画像などのグラフィクスの部分で、まだ実験を尽くしていない。

(10) Pd/GEM、BeOS、DirectXなど。現在調査中。その道の専門家によると「使える」らしい。(?)
(11) 静止画ビューワ/スライドショー/QTプレーヤ等をスクリプトによりシリアルから制御。そういうものが見つからなければ、自作するしかないのかもしれない。
(12) ノンリニア・ビデオ編集機器をリアルタイム制御する。実際にシリアル等の制御を受けるものがあり、資金力があれば実現可能か。

グラフィクスのライブ制御の全体を考える上で重要なのは、グラフィクスに対して「再生するのか」「生成するのか」「効果を加えるのか」という、サウンド系と似たような指針の選択であると思われる(図7)。

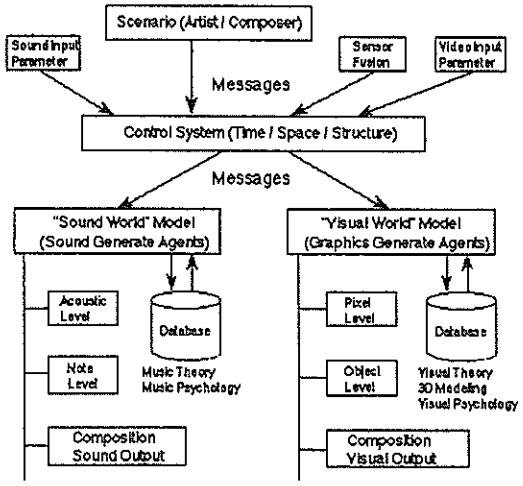


図 7. サウンドとグラフィクスの密接な関係

10. おわりに

研究と実験と創作と発表とは自動車で言えば四輪であると思われる。もっとも近い筆者の次なる挑戦は、2000年9月17日(日)静岡大学浜松キャンパスにおける、情報処理学会・電子情報通信学会等合同シンポジウムでの「アートとエンターティンメント」セッションでの招待講演における、インタラクティブアート作品の新作発表公演である。具体的なターゲットに臆せず挑戦し、多くのご批評、ご意見、ご議論をいただきたいと思う。

参考文献

[1] Y.Nagashima : Multimedia Interactive Art : System Design and Artistic Concept of Real-Time Performance with Computer Graphics and Computer Music, Proceedings of Sixth International Conference on Human-Computer nteraction (ELSEVIER) 、1995年
[2] Y.Nagashima et al. : 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、1995年
[3] Y.Nagashima : 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 (95-MUS-16)』(情報処理学会)、1995年
[4] Y.Nagashima : インタラクティブ・マルチメディア作品 "Asian Edge" について、京都芸術短期大学紀要［瓜生］第19号1996年、1997年
[5] Y.Nagashima : センサを利用したメディア・アートとインスタレーションの創作、京都芸術短期大学紀要［瓜生］第20号1997年、1998年
[6] Y.Nagashima : 生体センサによる音楽表現の拡大と演奏表現の支援について、情報処理学会研究報告 Vol.98,No.74 (98-MUS-26)、1998年
[7] Y.Nagashima : Real-Time Interactive Performance with Computer Graphics and Computer Music、Proceedings of the 7th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design, and Evaluation of Man-Machine Systems、1998年
[8] Y.Nagashima : BioSensorFusion:New Interfaces for Interactive Multimedia Art、Proceedings of 1998 International Computer Music Conference、1998年
[9] Y.Nagashima : インタラクティブ・アートにおけるアルゴリズム作曲と即興について、神戸山手女子短期大学紀要・第41号1998年、1999年
[10] Y.Nagashima et al. : "It's SHO time" -- An Interactive Environment for SHO(Sheng) Performance、Proceedings of 1999 International Computer Music Conference、1999年
[11] Y.Nagashima : クラシック音楽とコンピュータ音楽、神戸山手女子短期大学紀要・第42号1999年、2000年
[12] 後藤 et al. : MIDI制御のための分散協調システム --遠隔地間の合奏を目指して--、情報処理学会研究報告 Vol.93,No.109 (93-MUS-4)、1993年
[13] 長嶋 :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.97,No.71 (94-MUS-7)、1994年
[14] 松田 et al. : NeXT・ISPWとObjective-C・MAXを利用した新たなコンピュータ音楽環境へのチャレンジ、情報処理学会研究報告 Vol.95,No.19 (95-MUS-9)、1995年
[15] 片寄 et al. : MAXを利用したVoice Shooting Game、情報処理学会研究報告 Vol.95,No.74 (95-MUS-11)、1995年
[16] 高城 : NeXT with ISPWとSGIワークステーションによるインタラクティブ環境の実現、情報処理学会研究報告 Vol.96,No.102 (96-MUS-17)、1996年
[17] 平野 et al. : デジタルロストワールド計画(2) デジタルスタジオの構築と"PureData"への実装について、情報処理学会研究報告 Vol.98,No.47 (98-MUS-25)、1998年
[18] 平野 et al. : without MAX - Directorによるインタラクティブアート、情報処理学会研究報告 Vol.98,No.74 (98-MUS-26)、1998年
[19] http://nagasm.org/hightech/02-06/index.html
[20] http://nagasm.org/hightech/SUAC/index.html
[21] 長嶋 : MIDI音源の発音遅延と音源アルゴリズムに関する検討、情報処理学会研究報告 Vol.99,No.68 (99-MUS-31)、1999年

4M－6

メディア・インスタレーションを用いたインタラクティブ・パフォーマンスについて

長嶋洋一

静岡文化芸術大学(SUAC)

1. はじめに

「コンピュータ音楽」は図Aのように、狭義には音楽情報処理システムとしてある程度閉じた世界を構成しているが、コンピュータシステムと人間・外界とのインターフェースとしてセンサ技術を取り入れた広義のコンピュータ音楽の世界では、所謂メディア・アート、インタラクティブ・アート、メディア・インスタレーション等もその一つの形態として認知されるようになった[1][2]。

2000年4月に開学した静岡文化芸術大学(SUAC)デザイン学部技術造形学科では、多くの領域でのデザインの一つとして、この広義のコンピュータ音楽をテーマとした作品制作活動を既に開始してきた[3]。本稿では、この広義のコンピュータ音楽の特性について考察するとともに、SUACにおける具体的な作品制作・発表の過程を分析して、新しい表現と芸術創造の可能性について検討した。

2. インスタレーション

2000年5月28日に開催された「SUAC一般公開デー」において、教員と有志学生によつて共同制作した作品"森海"は、センサを活用した典型的なメディア・インスタレーションであった[3]。ここでは大型プラズマディスプレイを8台並べたスペース内で、来場者の移動を赤外線ビームセンサでセンシングして、Maxにより画像とサウンドとをインタラクティブに駆動した。このような作品は、一種のギャラリーである展示空間に次々と来場者が訪れるような場で有効であるが、あまり混雑していると誰の動きにシステムが反応しているかが判らなくなる、という課題も発見できた。サウンドの部分を聞かせたい、という意図が強い場合にはこの課題はかなり重大な問題となりうる。

3. パフォーマンス

2000年9月17日「インタラクティブ・メディアアート」というタイトルのシンポジウムの場で公演した作品"Wandering Highlander"は、古典的な意味でのパフォーマンスを伴ったインタラクティブ・メディアアート作品であった[3]。

ここでは、両手首・両肘・両肩の関節の曲げセンサを装着したPerformerのダンスにより、固定的再生でなくリアルタイムに音楽と映像が生成・駆動された。Maxによりセンサ情報を音楽・グラフィクスのシステムとリアルタイムに結び付ける、という意味では前述のインスタレーションと同じであるが、ステージ様のスペースで「開始」と「終了」という概念を自然に提供することで、間違いなく、これはパフォーマンス(演奏)であった。

4. Team風虎の作品"Windmill"

2000年12月16-17日の情報処理学会音楽情報科学研究会・インターカレッジコンピュータ音楽コンサートにおいて、SUACからは初めて、学生だけで制作した作品を対外発表した[3]。形態としては、「造形作品を用いたパフォーマンス」という作品のライブ公演である。これは図Aの広義のコンピュータ音楽の形態において、ある意味ではインスタレーション、ある意味では自然の状況を受け(光に反応)、そして「演奏時間」を持つパフォーマンス作品、でもあるという欲張ったものである。

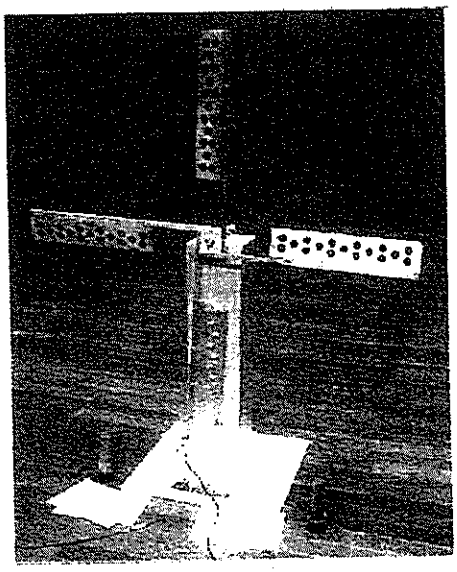


図1. オブジェ「虎造」

この作品"Windmill"は図1のように、ステージ上に大小4個のオリジナル造形作品(光を受けて回る「風車」)があり、ここにライブ制御の照明や、サーチライトや光の遮蔽物を持ったperformersが加わったパフォーマンスを行うものである。風車は変化する光とともに動きが変化し、この変化をセ

Study of Interactive Performance using Media Installation
Yoichi Nagashima (*nagasm@computer.org*)
Shizuoka University of Art and Culture

ンシングした情報がMax上のアルゴリズム作曲系を駆動して、背景音響系とともに全体のサウンドを構成した。

5. 自己評価と考察

この作品は、ステージ上に設置されたオブジェだけを見ていると、そのままインスタレーション作品としてロビー展示をしても通用する、と見えたかもしれない。しかし共同でこの作品を制作し、演出やリハーサルを重ねてPerformanceに至ったプロジェクトメンバーのレポート等を検討すると、間違いなくこの作品はインタラクティブ・パフォーマンス作品であったことが判る[4]。

インスタレーション作品は来場者にその鑑賞の形態・方法・姿勢・時間などを自由に委ねるのに対して、パフォーマンス(公演)の形態をとる作品は、客席で静かに鑑賞すること、そして上演時間という制約を聴衆に強制することになる。その一方で、インスタレーションでしばしば起きる「意図に気付かずに通過される」という問題点に対して、「演出」という要素によって積極的にメッセージや関係性をアピールすることができる。この両者はある部分では二律背反であり、メディアアート作家の永遠の課題でもあった[5]。

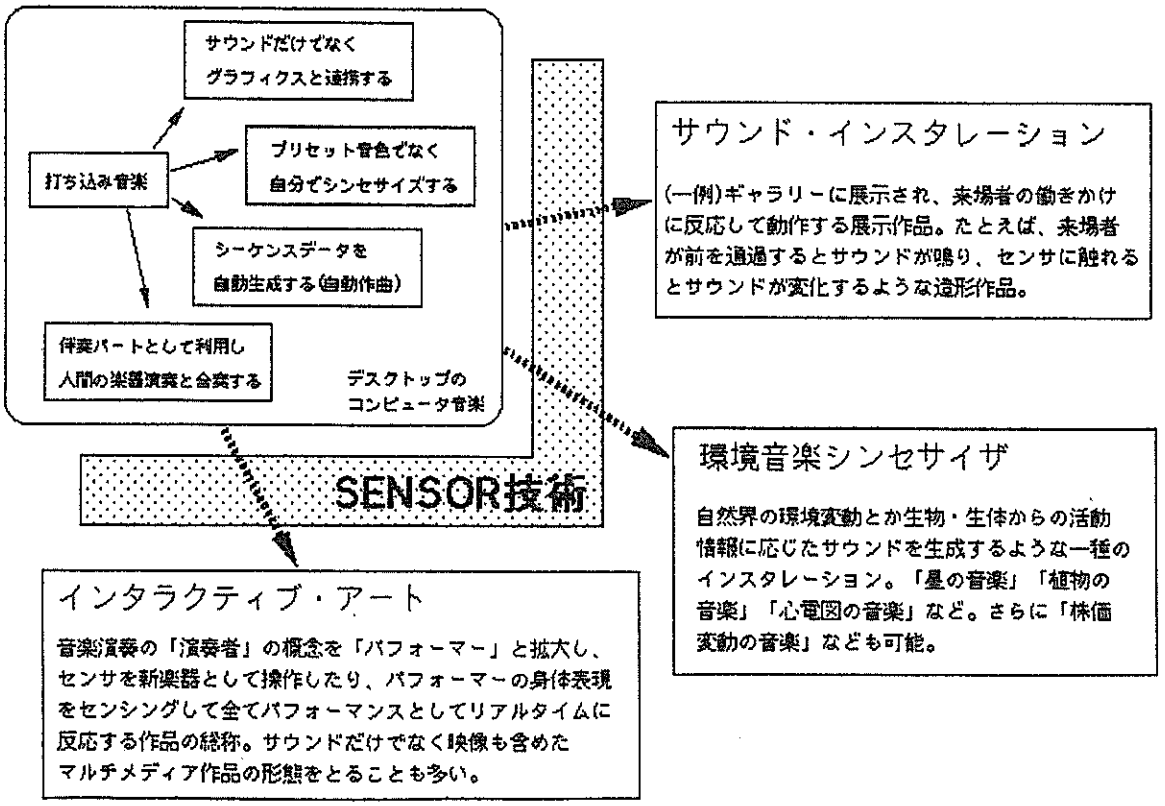
今回の作品 "Windmill" においては、「楽器」の延長としてのセンサという形態でなく、明らかにオブジェ、という造形作品としての要素を強く打ち出すとともに、背景音楽やPerformerに対応するサウンドの部分では音楽作品であった。そして人間とセンサとの間に「光」という自然環境要因が介在するとともに、複数のPerformerが振り付けまで考慮してパフォーマンスすることで、そこに独特な表現空間を実現できた。筆者のこれまでの作品ではセンサは「広義の楽器」の範疇を超えるものではなかったが、期せずして到達したこの新しい可能性は新鮮なものであった。もちろん、今回の公演を見て「自分で光を操って風車を回したり鳴らしてみたい」という感想を持った聴衆もおり、これはこの作品がインスタレーションへと進化する可能性も示している[5]。

6. おわりに

今回の作品によって、メディアアートの新しい可能性をあらためて知ることができた。今後も机上の空論でなく、実際の作品制作やシステム構築の具体的な進展とともに研究を進めていきたい。

参考文献

[1] 長嶋洋一「コンピュータサウンドの世界」、CQ出版、1999年
[2] 長嶋・橋本・平賀・平田編「コンピュータと音楽の世界」、共立出版、1998年
[3] 長嶋洋一「静岡文化芸術大学スタジオレポート」、情報処理学会研究報告 Vol.2000,No.118 (2000-MUS-38)、情報処理学会、2000年
[4] http://nagasm.org/SUAC/
[5] http://nagasm.org/ASL/



図A. コンピュータミュージックの展開

音楽情報科学 41－1
（2001．8．4）

新・筋電センサ"MiniBioMuse-III"とその情報処理

長嶋洋一

SUAC(静岡文化芸術大学)

新たに開発した、片腕8チャンネル・両腕同時計測型の筋電センサ"MiniBioMuse-III"について報告する。生体センシングで問題となるノイズ対策、多チャンネル同時計測手法、応用等についても考察した。

A new electromyogram sensor : "MiniBioMuse-III"

Yoichi Nagashima

Shizuoka University of Art and Culture

nagasm@computer.org

This paper reports the development of new electromyogram sensor called "MiniBioMuse-III". This sensor detects 16 channels (both arms) of EMG and converts the information to MIDI signal. I discuss the technical points of noise reduction, multi channel (time sharing) A/D conversion and microprocessing.

1. はじめに

コンピュータ音楽を中心としたメディア・アート、システムと人間とのインターフェース等に関するテーマの研究活動とともに、その具体的な応用を実験的に検証する意味で、実際にいろいろなインタラクティブ・マルチメディア作品を創作して公演・発表する活動を行っている(1-9)。オリジナルセンサを用いた実験的なシステムをリアルタイムパフォーマンスに応用することで、開発の過程や作品・公演の中から新たな研究テーマや課題が出てくることも多い。本稿ではその一つのサブテーマであるオリジナルセンサの研究開発に関して、新しく開発した筋電センサ"MiniBioMuse-III"について報告する。

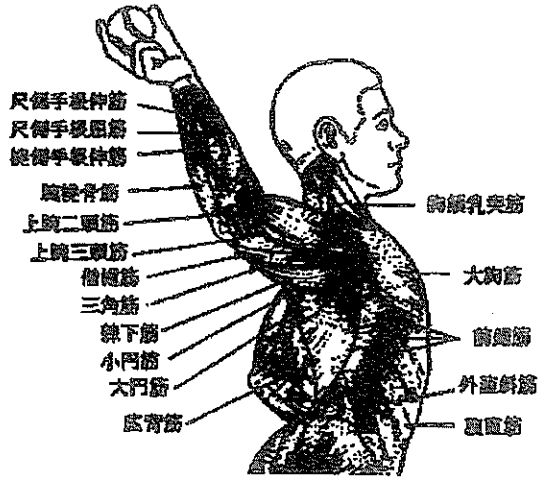


図1　上半身の筋肉のいろいろ

2. 腕の筋電情報のセンシング

人間の随意運動のセンシング対象としては、歴史的に「手」「腕」(図1、図2)が重視されており、文献においても「腕の筋電情報」を取り上げたものは多い(10-23)。

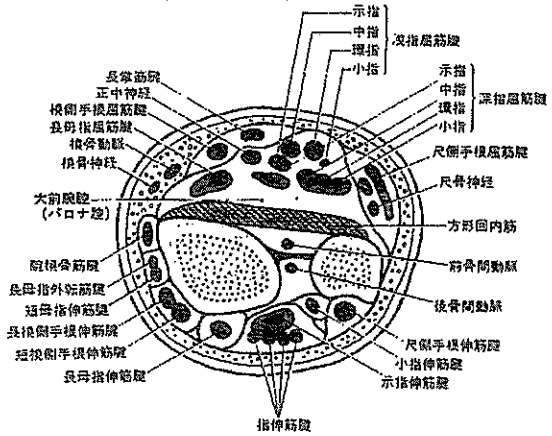


図2　前腕遠位の横断面

筋電センサを活用したコンピュータ音楽やインターフェースに関するテーマでは、IRCAM / CCRMAのAtau TanakaによるBioMuseを利用したヒューマンインターフェースの研究と音楽パフォーマンス活動(sensorband)が国際的によく知られている(24-26)。ここでは、両腕の異なる2個所ずつに筋電センサを取り付け、伝統的な楽器と変わらない修練と習熟により、身体表現としての演奏情報をリアルタイム音響合成パラメータに適用した演奏などを行っている(図3)。

図3　BioMuseを用いた演奏風景(Atau Tanaka)

このBioMuseは市販の製品であるが、センサの銀-塩化銀電極を導電ジェルによって取り付ける手間、その電極の寿命と交換の手間、システムとしての大きさと重さ、そして何より高価である(約3万ドル)ことなど、活用しているAtau Tanaka氏自身がいくつもの課題を指摘するものだった(図4)。もともと音楽用途というよりも、身体障害者のための意志伝達手段や、脳波・眼球筋肉運動などの検出にも利用できる汎用生体センサであるため、医用機器としての信頼性やコスト要求からして当然であるとも言える。

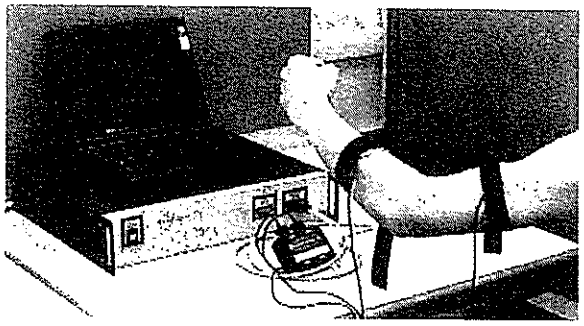


図4　BioMuse

3. "MiniBioMuse-I"と"MiniBioMuse-II"

筆者は研究協力者としてアナログ電子計測の専門家の照岡正樹氏らと交流し[27]、各種の高精度センサ、生体センサ等を研究開発してきた[28-29]。そこでテーマとして「小型軽量(可搬)」「バッテリ駆動」「リアルタイムに筋電情報をMIDI化」する「シンプルで安価」な筋電センサを目標として掲げ、敢えて"MiniBioMuse"と名付けて実験・開発を進めてきた。以下、その概要と検討事項について簡単に紹介する。

3-1. 筋電センサ開発の課題

通常の物理量センサに比べて、生体センサには「個人差」「高感度」「ノイズ抑止」「使用感」などの課題が加わる。「個人差」とは、同じ生理指標でも個人ごとのばらつきが大きく、筋電で言えば、非力な(体育会系でない)人の中にはまったく筋電パルスが検出できない人もいる、という状況のことである。「高感度」については当然のことで、人間は電気鰻ではないので、電気信号として得られる情報は全て微弱なものなので、高倍率増幅は必至である。「ノイズ抑止」は技術的にはもっとも重要なもので、生体から発生する他の信号、周囲環境から混入するノイズ信号とともに、ハム(商用交流電源の高調波ノイズ成分)の除去が切実な課題となる。「使用感」とは、ベッドに固定されているわけではなく音楽演奏という身体表現に利用することを目的としているので、自然な動作を制限するような形態でセンサを取り付けることができない、という実装上の課題である。

3-2. "MiniBioMuse-I"

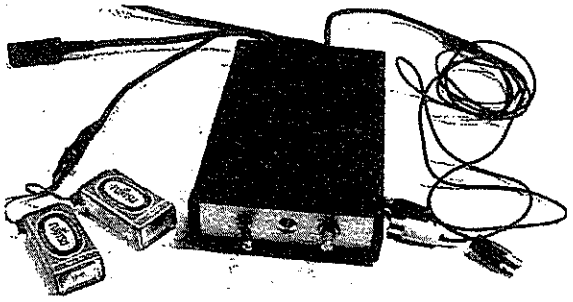


図5　MiniBioMuse-I

図5は、筆者が初めて開発した初代の筋電センサ"MiniBioMuse-I"である。回路としてはOPアンプによる差動増幅回路を採用し、両腕のセンサ電極(パソコンのメモリ増設時に静電気破壊を避けるために利用するリストバンドを改造)だけでなく足首にハムをキャンセルするための第3の電極を取り付けた。OPアンプのために006P電池を2つ使用するなど課題も多かったが、VHS

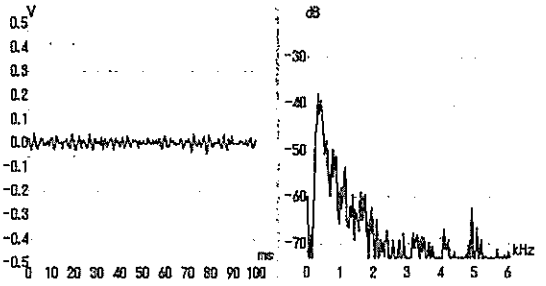


図6　MiniBioMuse-Iの平静時出力の例

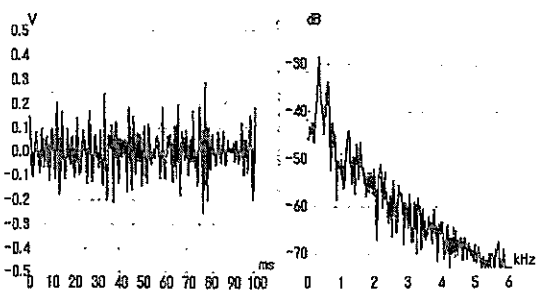


図7　MiniBioMuse-Iの緊張時出力の例

ビデオテープ程のサイズながら筋電ノイズそのものをアナログ出力しつつ同時にA/D変換してMIDI出力する、という機能には、Atau Tanaka氏も良好な評価を与えた(後継機に期待表明)。図6は筋肉を弛緩させた時の、図7は筋肉を緊張させた時の"MiniBioMuse-I"のアナログ出力信号(左側がオシロ、右側がスペアナ)である。ハムを中心とした残留ノイズが見て取れる。

3-3. "MiniBioMuse-II"

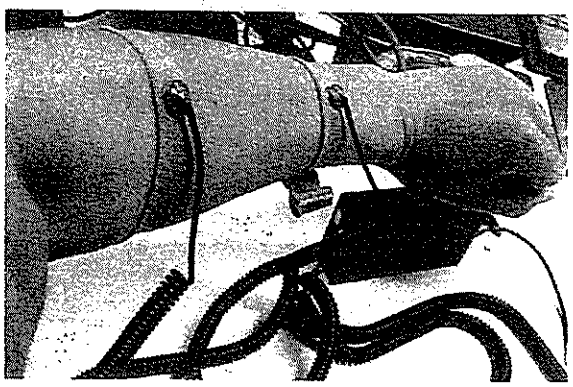


図8　MiniBioMuse-II

図8は、"MiniBioMuse-I"から新たな改良により開発した"MiniBioMuse-II"である。電子回路的には、ノイズの点で限界のあるOPアンプによるフロントエンド回路から、図9のような、高感度デュアルFETを用いたディスクリート・トランジスタ回路へと発展した。これは、特性の揃った2つのFETを金属ケースで熱結合した特殊なFETである2SK146により、単一電源で良好な高倍率差動増幅回路を実現したものであり、小型ケースに2チャンネル*2電極とコモン電極の全ての回路を格納した。アナログ電圧出力はケーブルで延長したサブボックスでMIDI化するように分離した結果、照明などノイズ環境の劣悪なステージでのライブパフォーマンス(京都と神戸で公演)にも使用できる、という実績を得た。

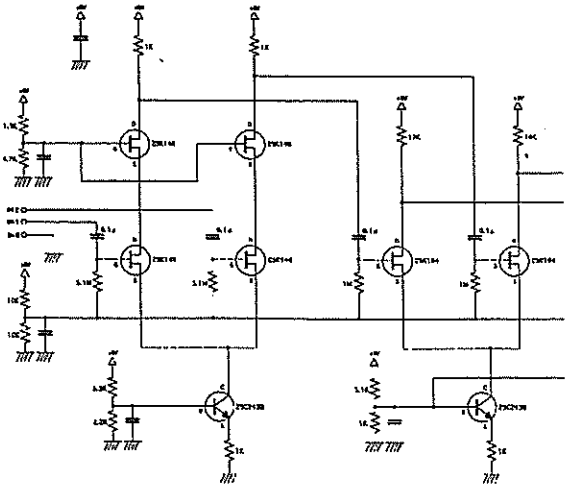


図9　MiniBioMuse-IIのフロントエンド回路(一部)

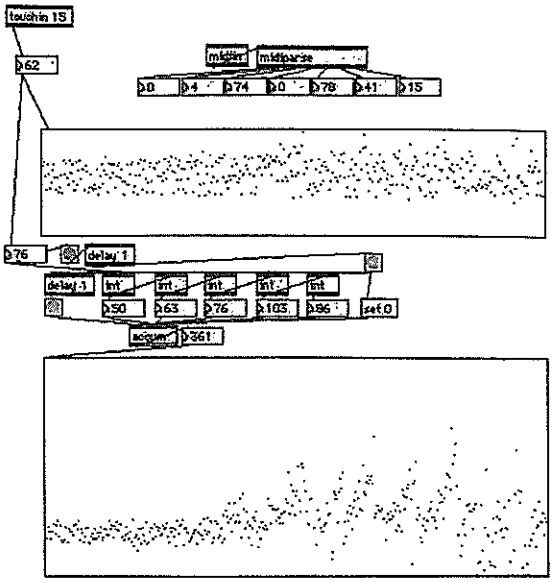


図10　MiniBioMuse-IIの出力の一例

図10は、"MiniBioMuse-II"からのMIDI情報を実際の音楽において利用するためのMaxパッチによる処理の一例である。上段のグラフはセンサからの直接の情報をそのままプロットしたものであり、下段ではこれを移動平均する処理により、右側で筋肉が緊張状態になった波動が明確に得られているのが分かる。

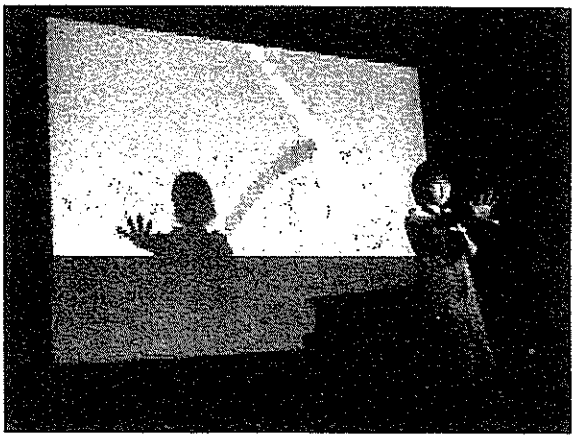


図11　MiniBioMuse-IIを用いた演奏風景

図11は、筆者の作品"Bio-Cosmic Storm"の京都での公演(1998年)の風景である。ステージ上のピアニストは両腕に"MiniBioMuse-II"のセンサを取り付け、そこから直接出力される筋電ノイズを音源としてSuperColliderでリアルタイム音響信号処理するシステムをコントロールするMaxのためのMIDI情報も同時に演奏出力した。ピアニストはステージ上のピアノの鍵盤に触れてはいけない、という指示のもと、鍵盤上空5cmでピアノ曲をシャドー演奏したりピアノそのものを押したりして、その筋電情報によりリアルタイムCGとともに楽音を生成した。

4. "MiniBioMuse-III"の開発

前作"MiniBioMuse-II"からその後の進展は1年以上、停滞した。その理由は、図9にある改良された筋電センサ・フロントエンド回路によっても完全にはハムが取れないこと、そしてかなりの小型化によって作品公演まで実現したことで、それ以上の大きな性能向上の具体的な目標が見えなかったことによる。それが2000年から2001年にかけて急速に進展し、最終的に新しい"MiniBioMuse-III"として一気に結実した。以下、その過程と概要について紹介する。

4-1. 独立成分分析による多チャンネル計測

BioMuseのように腕に単一ないし2つの電極を用いるのでなく、腕をぐるりと巻くように多数の電極を用いて同時計測し、複数の筋肉の作用をパターン認識の一手法である独立成分分析により検出する、という藤原の研究(30)と交流・情報交換する機会を得た。Atau Tanakaは電極の位置について独自のトレーニングにより(非対称な)適正設置場所を獲得したが、この研究によれば、ベルト状の複数電極により情報検出とパターン認識の汎用性が得られる可能性があった。

4-2. ソフトウェアDSPによるノッチフィルタ

ステージなど劣悪なノイズ環境で筋電センサを使用する上でもっとも問題となるハムノイズの除去手法として、古くから知られたノッチフィルタをA/Dセンシングのマイクロプロセッサのソフトウェアで実現する、という可能性を実験する段階が到来した。具体的には、文献(31)にある図12のようなアルゴリズムを、AKI-H8のCPUプログラムとして実装する実験を行った。

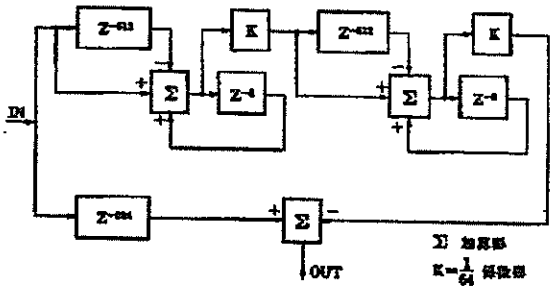


図12　ノッチフィルタのアルゴリズム例

後述する"MiniBioMuse-III"の開発の過程で図12のアルゴリズムをAKI-H8に実装した結果、図13のような特性で50Hz/60Hzのハム成分をカットする筈であったフィルタ出力は、実際には目に見える(耳に聞こえる)違いを見出せないという結果であった。その理由は、(1) AKI-H8のアナログ入力A/Dを8ビット精度モードとして動作させた、(2) AKI-H8の内蔵RAMの容量からくる制限により、ディジタルフィルタのビット幅として16ビット精度の確保が困難なため8ビット幅とした、という点にある。これにより、ビットシフト演算によって計算途中のデータが丸められて実質的には消滅してしまった。そこで折衷案として、8ビット幅の処理でも有効となる図12のアルゴリズムの一部をAKI-H8のソフトウェアとして実装して、何もない状態よりは有意に有効なソフトウェア・フィルタリング(一種の積分型フィルタ)を実現することができた。

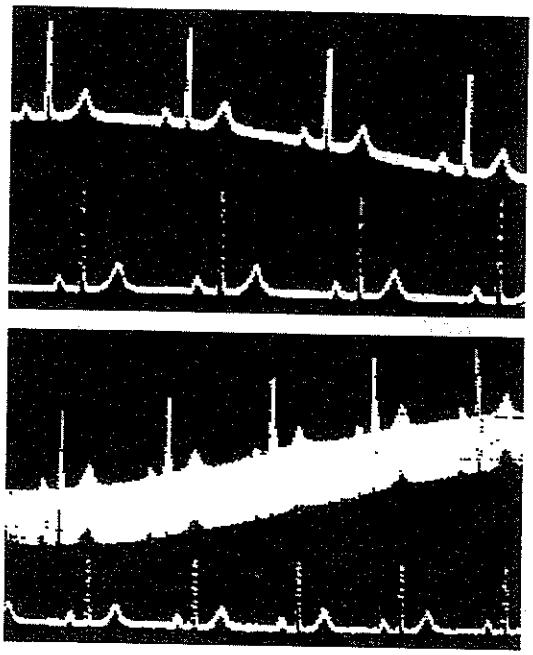


図13　ノッチフィルタ(上段:入力 / 下段:出力)

4-3. フロントエンド回路の改良

温度結合されたデュアルFET : 2SK146による図9の回路でもう限界と思われたフロントエンド回路についても、多チャンネル高密度実装を視野に入れた簡易型を狙いながらさらに性能を向上させる努力を進めた。具体的には、図14のように単一電源回路から3Vリチウム電池による2電源回路に変更して、オフセット特性とダイナミックレンジを向上させた。さらにフロントエンド回路にOPアンプによるローパスフィルタまで実装することとした。これにより、増幅回路としての性能を向上させつつ同時にA/D・CPU回路に至る信号ラインのローノイズ化を実現することができた。

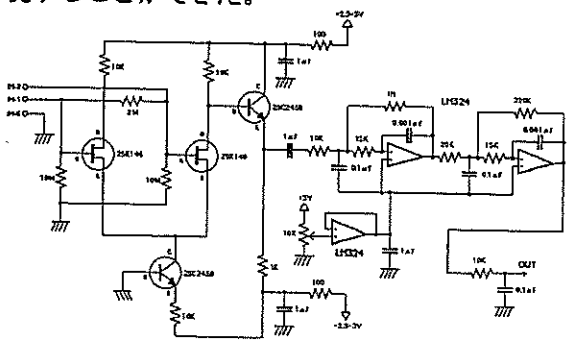


図14　MiniBioMuse-IIIのフロントエンド回路

4-4. 電極バンドの改良

前作"MiniBioMuse-II"までは、センシング電極として、静電気帯電防止用バンドを流用した簡易電極であったが、この部分を大幅に改良したオリジナル電極を制作することにした。

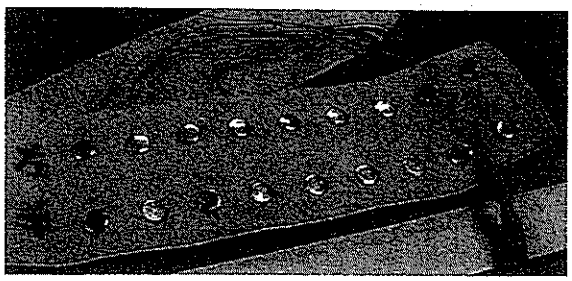


図15　MiniBioMuse-IIの電極部分

図15はその電極バンド部分である。ベースとなっているのは介護用の伸縮ベルト(マジックテープで固定)で、ここに洋裁用の金属ボタンをカシメて、その上に純銀円板をハンダ付けした。前作まではこのボタンを単に嵌め合わせていたが、接触抵抗と接触不良があるために電線を直接にハンダ付けすることにした。2電極のペアが合計9列並んでいて、中央の1列がノイズ抑止のための差動回路の基準電位(アース)となる。

4-5. 高密度空中配線による実装

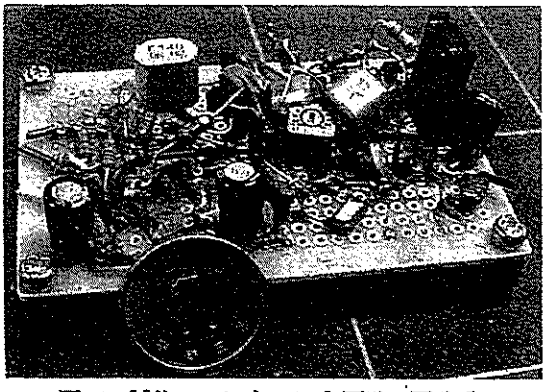


図16　試作フロントエンド回路(1回路分)

図16は、図14の回路を1回路分だけ試作した基板である。これを片腕の8チャンネル分で図17のように製作するには相当の困難があった。

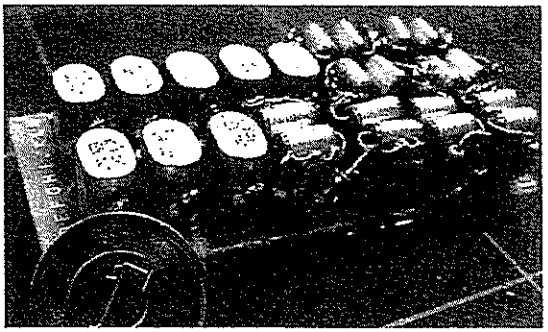


図17　フロントエンド回路(8回路分)

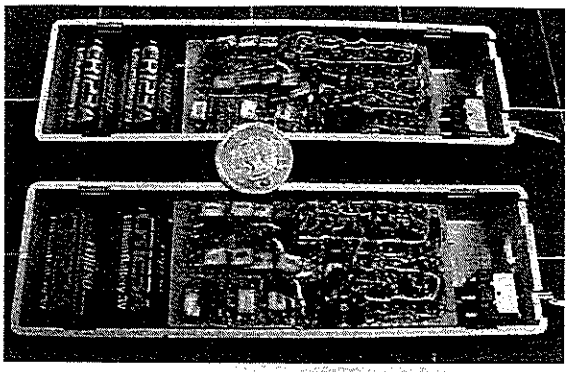


図18　フロントエンド部分(16ch)

図18は、両腕16チャンネル分が完成して、それぞれ2個のリチウム電池とともにケースに入れた様子である。図19は、この2個のフロントエンドボックスと接続する、AKI-H8の内蔵されたA/D - MIDIメイン・インターフェースであり、こちらは単3乾電池4本の+6V電源とした。

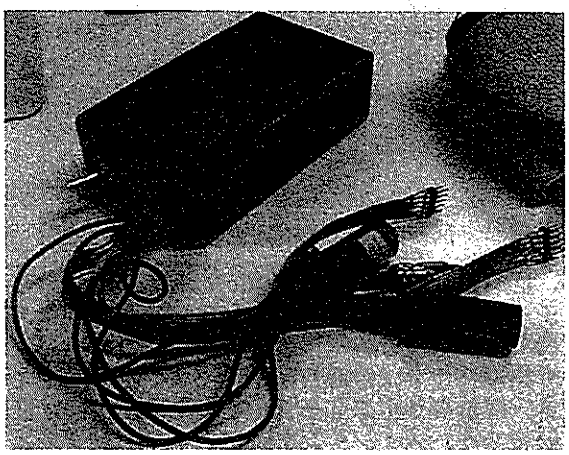


図19　メイン・インターフェース部分

4-6. マイコンAKI-H8部分

図20のマイコン部分では、制御情報のMIDI入力と8ビット2chのD/A出力ポートを加えた。

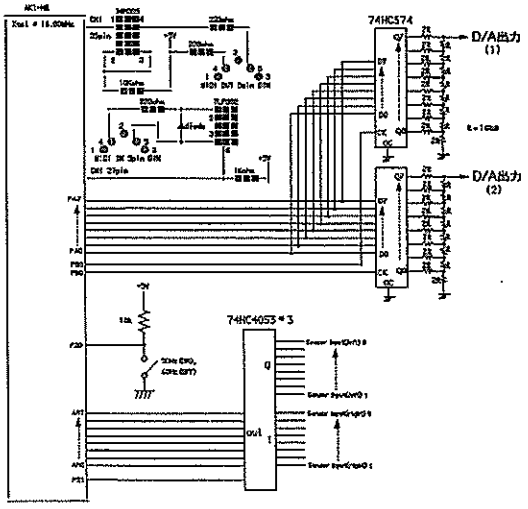


図20　コントローラAKI-H8回路

5. "MiniBioMuse-III"の動作とその検討

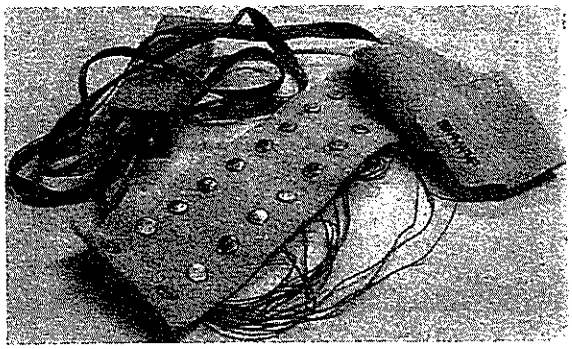


図21　MiniBioMuse-IIIの片腕部分

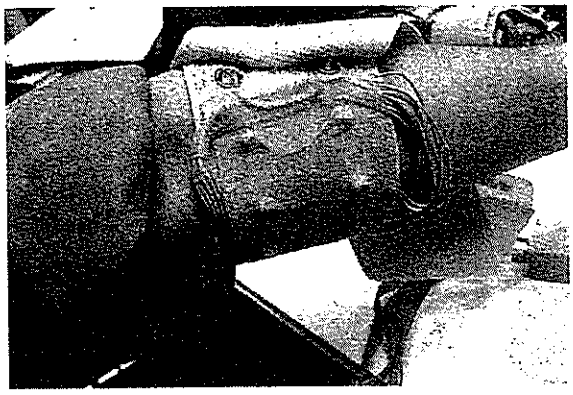


図22　MiniBioMuse-IIIの装着風景

図21は、完成した"MiniBioMuse-III"の片腕8チャンネル部分の電極からセンシングボックスまで、図22はこの電極を装着した様子である。介護用ベルトは伸縮自在で無理なく密着固定でき、軽快なリボンケーブルとともに使用感は良好である。メインインターフェースボックスとの接続は図19に見えるフラットケーブルコネクタにより行う(向きが異なり混同しない)。

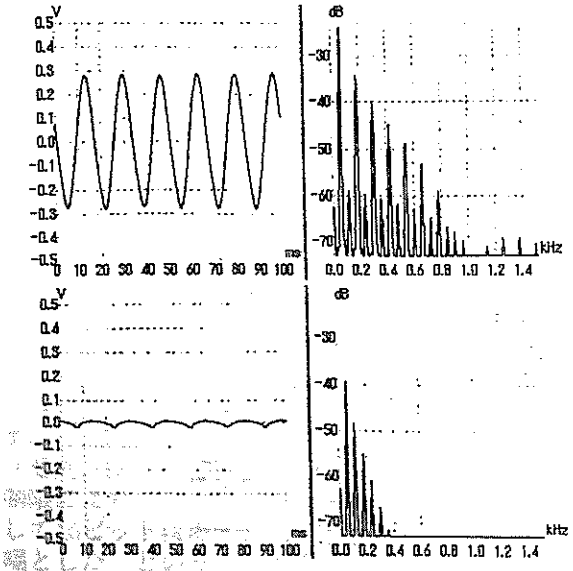


図23　MiniBioMuse-IIIの平静時出力の例

5-1. アナログセンシング出力信号

図23は、"MiniBioMuse-III"の両腕16チャンネルのフロントエンド回路からの出力を同一条件で計測した平静時信号のうち、もっともハムノイズの大きなチャンネル(上段)と小さなチャンネル(下段)の様子である(全てのデータは文献(32)を参照のこと)。これは位置関係の異なるアース電極を共通とし、また腕の筋肉の配置の様子によりばらつきが出るのは、ある意味で当然のこととして吸収しなければならない。

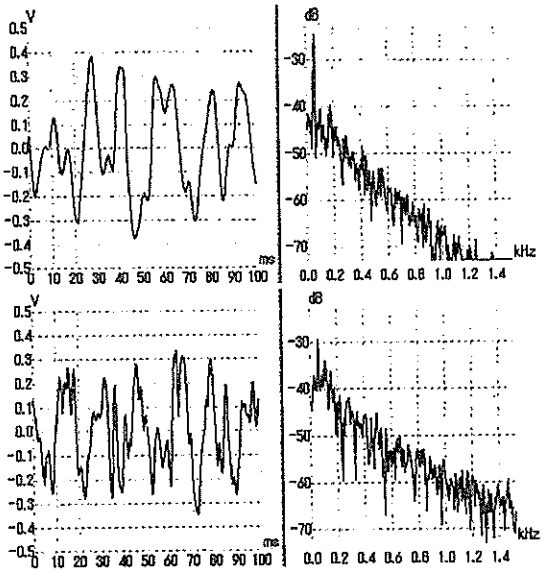


図24　MiniBioMuse-IIIの緊張時出力の例

図24は緊張時信号の一例であり、ハム成分が乗っているものの、明らかに筋電信号がある。この図23/24の信号は、図6/7の初代よりも劣るように見えるかもしれないが、電源電圧が1/3であるにもかかわらず、後段でのノイズフィルタリング前にここまでの信号が安定に得られるというのは画期的進展なのである。

5-2. ディジタルフィルタリング出力信号

フロントエンド回路部分のOPアンプによるフィルタ以外に"MiniBioMuse-III"がソフトウェア的に行っているのは、図12のディジタルフィルタのアルゴリズムの一部を利用した積分フィルタリングと、MIDI出力のためにダウンサンプリングを重ねて時間平均(平滑化)を行っている部分である。ここではまず、前者について実際の計測データを用いて紹介する。

図25は、上段が"MiniBioMuse-III"の16チャンネルのうちのあるチャンネルのフロントエンド平静時信号である。ハム成分のシンプルな正弦波であり、これをそのまま音源素材として使用することは困難である。ところがこの図の下段では、ほとんど信号が消えている。これはAKI-H8によるソフトウェア・フィルタリングの出力を増設したD/Aポートから出力したそのままの

信号であり、アンチエイリアシング回路の前段ということでディジタル的に変化しているものの、特性としてハムに対応した周期の信号がキャンセルされていることがわかる。

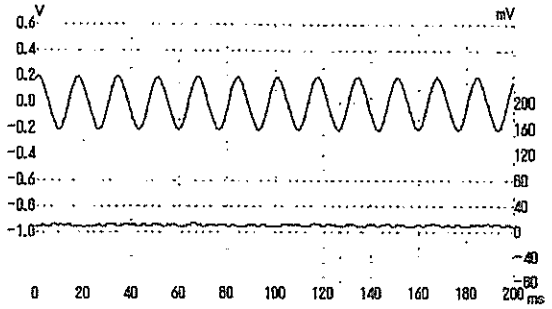


図25　MiniBioMuse-IIIの平静時出力の例(1)

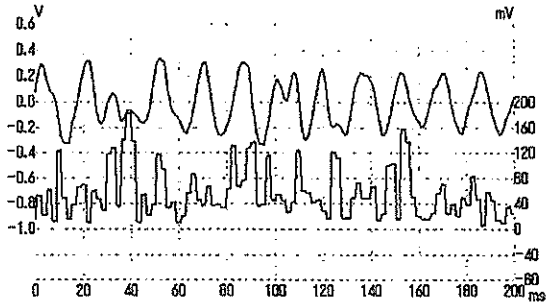


図26　MiniBioMuse-IIIの緊張時出力の例(1)

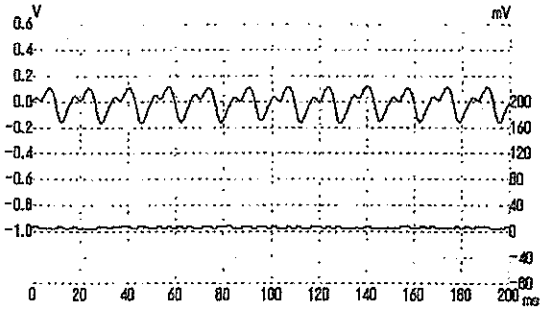


図27　MiniBioMuse-IIIの平静時出力の例(2)

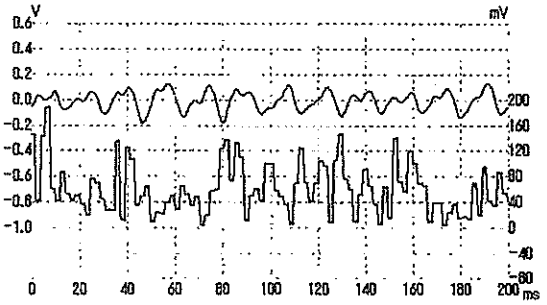


図28　MiniBioMuse-IIIの緊張時出力の例(2)

図26は図25のチャンネルの緊張時の出力信号であり、ハム成分をキャンセルしつつ筋電成分を抽出できている。図27と図28は同様に別のチャンネルでの平静時と緊張時の様子である。

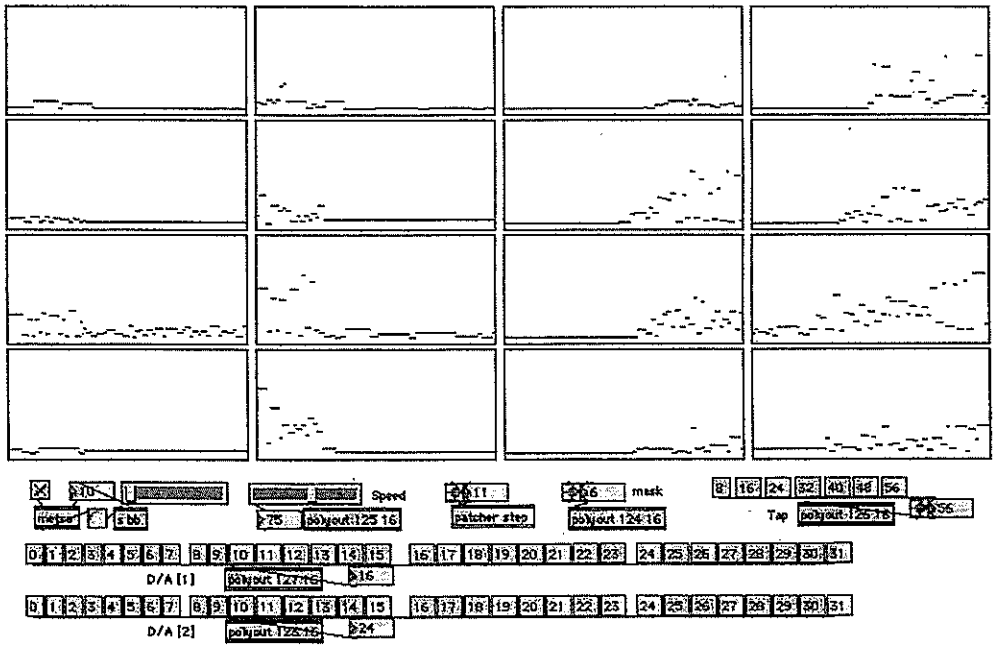


図29　MiniBioMuse-IIIのMIDI出力情報の例(1)

図29は、"MiniBioMuse-III"のMIDI出力情報をMaxでリアルタイム表示した様子の一例である。画面は左上から下に「左手1ch」-「4ch」、いちばん右側が「右手5ch」-「8ch」である。この図では、時間的に前半は左手を緊張させ、後半で右手を緊張させている(各グラフは左にスクロールする)。図30は両腕を交互に緊張・弛緩させた例であり、時間的同期性とそれぞれの腕の筋肉の動作の違いが見て取れる。ここからMaxのアルゴリズムとして、独立成分分析やニューラルネットワークのような手法でパターン認識の情報処理を実現することも容易であろう。

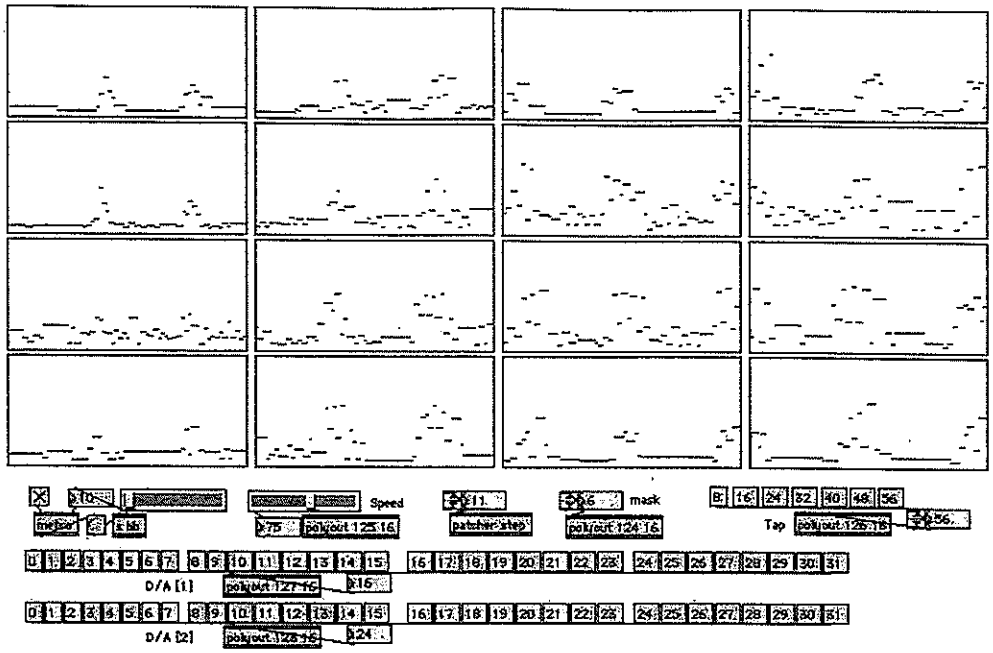


図30　MiniBioMuse-IIIのMIDI出力情報の例(2)

6. おわりに

新しい筋電センサ"MiniBioMuse-III"のメイキングについて報告した。これまでの研究の進め方を踏襲するとすれば、次にはこれを作品公演・パフォーマンスに活用してみて、さらに次のステップのための課題を集める段階である。その目標として、2001年9月に予定しているフランス・ドイツ演奏旅行における新作公演を考えており、その内容や結果は、またいずれ次の機会に報告したい。

参考文献

[1] Y.Nagashima : Multimedia Interactive Art : System Design and Artistic Concept of Real-Time Performance with Computer Graphics and Computer Music, Proceedings of Sixth International Conference on Human-Computer nteraction (ELSEVIER), 1995年

[2] Y.Nagashima et al. : 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, 1995年

[3] 長嶋洋一：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 (95-MUS-16)」（情報処理学会）、1995年

[4] 長嶋洋一：センサを利用したメディア・アートとインスタレーションの創作、京都芸術短期大学紀要［瓜生］第20号1997年、1998年

[5] 長嶋洋一：生体センサによる音楽表現の拡大と演奏表現の支援について、情報処理学会研究報告 Vol.98,No.74 (98-MUS-26)、1998年

[6] Y.Nagashima : Real-Time Interactive Performance with Computer Graphics and Computer Music, Proceedings of the 7th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design, and Evaluation of Man-Machine Systems, 1998年

[7] Y.Nagashima : BioSensorFusion:New Interfaces for Interactive Multimedia Art, Proceedings of 1998 International Computer Music Conference, 1998年

[8] Y.Nagashima et al. : "It's SHO time" -- An Interactive Environment for SHO(Sheng) Performance, Proceedings of 1999 International Computer Music Conference, 1999年

[9] 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.97,No.71 (94-MUS-7)、1994年

[10] i.Steve Ditlea, "The PC goes ready-to-wear", IEEE SPECTRUM October 2000, pp.35-39

[11] 斎藤正男、「生体工学」、電子情報通信学会、1985年、pp.33-57

[12] 星宮望、「生体情報計測」、森北出版、1997年、pp.40-66

[13] 細田耐一監修、「生体時系列データ解析の新展開」、北海道大学図書刊行会、1996年、pp.273-297

[14] 上羽康夫、「手　その機能と解剖」、金芳堂、1970年、pp.169-190

[15] シュフラー・S.シュミット、「からだの構造と機能」、西村書店、1998年、pp.113-117

[16] 岩瀬善彦・森本武利、「やさしい生理学」、南江堂、1969年、pp.244-259

[17] 生体情報の可視化技術編集委員会、「生体情報の可視化技術」、コロナ社、1997年、pp.185-210

[18] 日本生理人類学会計測研究部会、「人間科学計測ハンドブック」、技報堂出版、1996年、pp.252-262

[19] 池田譲一他訳、「生体工学」、コロナ社、1974年、pp.46-62

[20] 戸川達男、「生体計測とセンサ」、コロナ社、1986年、pp.260-269

[21] 松村道一、「ニューロサイエンス入門」、サイエンス社、1995年、pp.126-130

[22] 木村雄治、「体を測る」、コロナ社、1995年、pp.61-63

[23] 川上雅之・岩崎英人編著、「ヒューマンサイエンス」、不昧堂出版、1998年、pp.35-49

[24] Atau Tanaka : Musical Technical Issues in Using Interactive Instrument Technology with Application to the BioMuse, Proceedings of 1993 International Computer Music Conference, 1993年

[25] William Putnam : The Use of The Electromyogram for the Control of Musical Performance, Doctoral Thesis of Stanford University, 1993

[26] http://www.sensorband.com/index.html

[27] http://nagasm.org/VPP/index.html

[28] 長嶋洋一：「身体情報と生理情報」、長嶋・橋本・平賀・平田編「コンピュータと音楽の世界」、共立出版、1997年、pp.342-356

[29] 長嶋洋一：「コンピュータサウンドの世界」、CQ出版、1999年、pp.148-166

[30] 藤原義久・前川聡、「独立成分分析による筋電データからの各指運動の分離」、信学技報MBE99-7、電子情報通信学会、1999年、pp.41-46

[31] 中村尚五、「ビギナーズディジタルフィルタ」、東京電機大出版局、1989年、pp.154-173

[32] http://www.suac.ac.jp/ASL/14-05/index.html

[33] http://nagasm.org/ASL/index.html