情報処理学会論文誌　Vol. 52　No. 12　3096–3105 (Dec. 2011)

フィジカル・インタラクションを使ったメディア造形基礎教育におけるプログラミング学習の実践

有賀妙子[†1]　森　公一[†1]

コンピュータやセンサ，I/O デバイスなどを用いた表現は，鑑賞者と作品とのインタラクティブな関係性を実現する．メディア・テクノロジがもたらした表現の特性に注目し，人の行為に応じて動的にグラフィックイメージが生成されるインタラクティブインスタレーション作品を制作する内容の，メディア造形基礎教育の構築を試みてきた．そして，そのための教材としてツールキットを開発した．インタラクティブなメディア表現を実現するには，コンピュータプログラムの学習が必要である．模倣する→再生産する→創作するというフェーズを通してプログラミングの理解を進める目的で，その出発点となるサンプルライブラリを用意した．本論ではそれらを紹介するとともに，作品におけるサンプルプログラムの流用，学生のプログラミング理解の認識，プログラミング学習に対する意欲について考察する．

Learning Computer Programs in a Basic Course for Interactive Media Contents

Taeko Ariga[†1] and Koichi Mori[†1]

The expressive way of using media technologies such as computers, sensors, and I/O modules has produced new interactive relations between humans and artworks. We developed a training course and original toolkits that enable students to create an interactive installation and learn how to interrelate graphic images generated by a program with human actions. To create an interactive media artwork needs learning of computer programs. We also developed program libraries to help students understanding programming and use them as a starting point of a “imitate-reproduct-create” learning process. This paper describes the teaching materials and considers appropriation of sample programs to students’ programs, students’ recognition of understanding to program, and motivation for learning to program.

1. はじめに

コンピュータ，センサ，I/O モジュールのような情報技術，電子技術を使った表現手法が，1990 年代初頭からアートの世界に現れ，メディアアートと呼ばれるジャンルを切り開いた．多様な表現実験が行われているが，とりわけ鑑賞者の身体的行為とそれに呼応する映像や音声のフィードバックを表現の重要な要素とする傾向，すなわちフィジカル・インタラクションを前提とする表現が数多く試みられ，メディア・テクノロジに特有の感覚領域が実験されてきた．筆者らはメディア・テクノロジが提供するインタラクションの表現特性に注目し，メディア造形教育とでも呼ぶべき基礎的な教育内容とそのためのツールキットを開発し，実践してきた[1)]．これをセンサリビジョン (Sensory Vision) プロジェクトと呼んでいる．本論ではそのうち，コンピュータプログラムの学習に着目し，初学者のプログラミング理解の認識変化や学習意欲について考察する．

2. 造形基礎教育プログラムの目的

本教育プログラムでは，音センサ，赤外線センサなどを用い，インタラクティブなメディア造形作品 (インスタレーション) の制作を行う．これはセンサからの入力に応答して動的に生成されるグラフィックと人の行為がインタラクションをとることで鑑賞する作品である．

画家は絵画制作の前段階において行うデッサンあるいは習作を通じて，光や色彩，形態や質感などの感性的次元を獲得する．これと同様にデジタルメディアを使った造形表現の領域で，感性的次元の気づき，制作力獲得のための基礎的トレーニングとして本教育プログラムを構築した．この教育プログラムを通して学生に期待するのは，入力とそれに応じた出力といった単なるメカニズムにとどまらず，固定的な意味関係を超えた新しい行為と感覚のインタラクティブな関係を探究することである．加えて，メディア・テクノロジの扱いに関する知識やスキルの向上も目指し，表現的側面と技術的側面の双方の力の育成を目的とする．

3. 本教育プログラムにおけるプログラミング学習の意味

メディア技術を表現の手段とした造形基礎教育である本教育プログラムにおいて，プログラミングの学習によって得られる利点は，次のような点である．

†1 同志社女子大学学芸学部情報メディア学科
Department of Information and Media, Doshisha Women’s College of Liberal Arts

(a) 数理的な手法による制作力をつける．
　数理的手法に基づく表現方法を見つけ出し，実現する経験をする．
(b) 違いを生み出す力を実感する．
　既存のアプリケーションが提供する表現方法を使用するだけでなく，違いを生み出す達成感，自分がやったという主体感を持つ．
(c) 制作・表現の幅を広げる．
　プログラムの知識を必要とする制作ツールを今後の制作に選択できる，またそれを使用する際の理解を容易にする．

非コンピュータ科学の文脈の中でプログラムを学ぶ利点は，いままで多く論じられている．その先駆者で LOGO 言語の開発者 Papert は，「タートルに行動や「思考」するのを教えることが，自分自身の行動や思考を熟視することにつながる」とタートルグラフィックスを使ったプログラミング学習の利点を述べている[2]．本教育プログラムでは，プログラムの工夫がグラフィック要素の動きとなって見えることで，驚きや楽しみを体験しながら，新たな表現を求めて数理的思考を深めるところに特徴がある．そしてプログラムによる表現を試しながら作り出し (上記 (a))，作品を完成させることで主体感の獲得 (b) を経て，プログラムを今後の制作に適用 (c) していけるようになることが，プログラミング学習の目指すところである．

本教育プログラムが，プログラミング学習の上記の利点を発揮するには，フィジカル・インタラクションとそれに応答するグラフィックスがもたらす驚きや楽しみを学生が継続して持てる必要がある．一方で，教育プログラムの実施において，プログラミングの理解にかけられる時間は限られており，14 回の授業のうち 4 回で次のようなプログラム要素の説明をした後は，自学となる．

・変数
・メソッドの実行
・算術演算と代入
・制御 (if else 文と for 文)
・配列
・位置の計算 (等速・加速運動)
・クラスの活用
・音声ファイルの再生
・画像ファイルの描画
・センサからの入力値の比例変換
・粒子システムの利用

そこで，次の 3 つのフェーズを経て自学ができるよう，5.3 節で述べるサンプルライブラリを用意し，模倣することからプログラム制作を始め，それを修正し工夫を積み重ねながら，作品を完成するよう，指導した．

(i)　模倣する · · · 既存のプログラムを真似て，使う段階．テストをしながら，プログラム内の記述を調べ，説明されたプログラム要素の使われ方や，その意味を知る．
(ii)　再生産する · · · 既存のプログラムを改造して実験する段階．学生はグラフィックスの動きが数理的メカニズムを使ってどう制御されるかを理解できる．その理解が次のフェーズの独自の動きの創作につながる．
(iii)　創作する · · · これまでのプログラミングの理解をもとに独自の動きを生み出す数理的手続きを考え出し，それをインタラクティブアニメーションの制作に適用できる．

3 つの段階的な過程を通して，驚きや楽しみが継続でき，学習意欲につながると期待する．

4. 関 連 研 究

リベラルアーツ教育の一環として，芸術とコンピュータ科学の領域を横断した内容の教育が多く試みられている．たとえば，Ursyn ら[3),4)] や Ribner ら[5)] の教育プログラムは，コンピュータ科学専攻と芸術専攻の学生が同じクラスでマルチメディアコンテンツを制作する内容で，双方の理解とスキルの向上を目的にしている．我々の教育内容は，専門が異なる学生たちの学び合いが主眼ではなく，フィジカル・インタラクションを持つメディア作品が与える人の行為と感覚の関係について考察を深めることを目指す．

また，センサ技術や I/O モジュールを使った教育プログラム (たとえば文献 6)，7)) も開発されているが，多くは工学系教育の文脈で実施され，ロボティックスに関する学習に焦点がある．Kim ら[8)] は，芸術とコンピュータ科学の双方の基礎的要素を学ぶためのコースとしてインタラクティブな「触れる」展示を行うプロジェクトを提案した．芸術と IT 技術の視点をベースにインタラクティブなアート作品を制作するという点で，我々の教育内容と共通点がある．Kim のコースでは，多くの機械的，電子的工作を求めるのに対し，我々は基礎的な学習プログラム内でも作品としてのかたちが作りやすい，仮想的なメタファであるモーショングラフィックスを応答部分として使った．

センサリビジョンプロジェクトでは，教材としてオリジナル I/O モジュールを含むハードウェアツールキットを開発した．電子工学の知識やハンダ工作が不要な I/O モジュールやセンサとして，PicoCricket Kit[9)] や pri/pro[10)] がある．MINDSTORM がロボットを作るキットとして特化されているのに対し，光や音声や動きなどを使った芸術的な制作物のためにデザインされたキットで，子供向けのワークショップなどで使用されている．PicoCricket Kit はキットとしての完成度は高いが，子供を主たる対象にしていることもあり，本格的な作品制作に応用するには発展性に欠けるところがある．

PicoCricket とプログラミング環境 Scratch の開発者 Resnick は，imagine → create → play → share → reflect のプロセスを Kindergarten approach と呼び，構成主義的な学びを通して創造的な思考力を養成する方法を提案している[11]．アメリカ National Science Foundation は，Computational thinking (CT) に関する白書[12] の中で，CT スキルの向上のために Use-Modify-Create の 3 つのフェーズを通して学習を進めるフレームワークを提案した．これは 3 章で述べたプログラミング学習の 3 つのフェーズの考えと共通する．

5. 造形基礎教育プログラムの内容

5.1 テーブル・インタラクション

行為と感覚がインタラクティブに関係し合う場として，白い木製のボックス (W500 × D500 × H900 mm，**図 1**) を用意した．天井から吊るしたプロジェクタの下にボックスを設置し，天板に映像を投影する．箱の内部にはコンピュータや I/O モジュールなどを置くことができ，天板は箱本体から取り外しが可能で，穴開けや接着などの加工が容易にできるようにした．天板にセンサを埋めこむことで，テーブル面がインタラクティブなインタフェースとして機能する．

テーブルは触る，たたく，なでる，息を吹くなどさまざまな行為を誘発し，鑑賞者のそのような行為に応じて，生成されたグラフィックイメージが動き，変化し，それが多様な感覚経験を生み出す．

5.2 行為をセンシングする装置

鑑賞者の身体的行為をデータ化し，コンピュータに取り込むにはセンサなどの入力装置と，入力装置からのアナログ信号をデジタル信号に変換する I/O モジュール (AD コンバータ) が必要である．そのために本教育プログラムではオリジナルのハードウェアツールキットと必要なソフトウェアを開発した．

ハードウェアツールキットにはセンサとして音センサ，光センサ，距離センサを用意し，初学者向けの使いやすさを考慮したオリジナルの I/O モジュールを開発した (Sensory Vision (SV) モジュールと呼ぶ，**図 2** の右から 3 つめ)．4 つのアナログ入力ポート，4 つのデジタル (PWM) 出力ポートを備えた AD コンバータで，擬似 RS-232 シリアル信号で通信する．Arduino や Gainer[13] など類似機能を持つ I/O モジュールは存在するが，ブレッドボードが必要である，あるいは接続が不安定になりやすいといった，本教育プログラムにとっては実装上の欠点がある．SV モジュールはセンサからのアナログ値 (0–5 V) を 0–255 の 8 bit 信号に変換する．各種センサと I/O モジュールの接続は，センサ側にミニピンプラグ，I/O モジュール側にミニピンジャックのコネクタを採用することで，ハンダなどの電子工作が不要で，初学者でも簡単に実験を行うことができるよう工夫した．

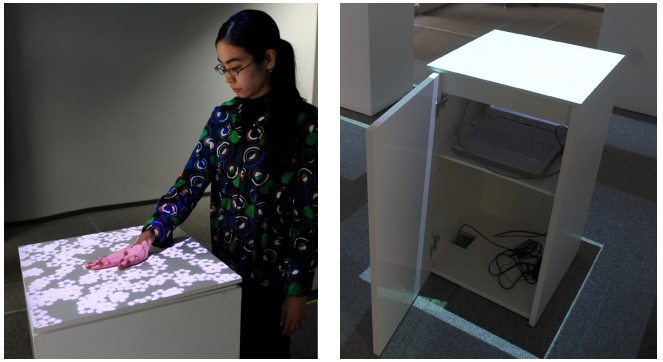

図 1　インタラクションの場
Fig. 1　A field for interaction.

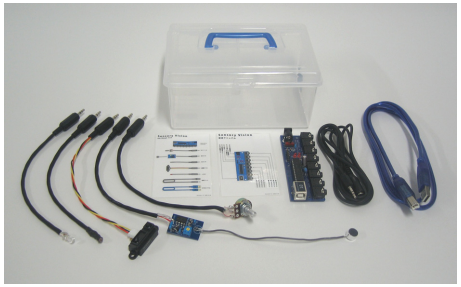

図 2　ハードウェアツールキット
Fig. 2　The hardware toolkit.

5.3 行為に応答する画像イメージの生成

鑑賞者の行為に基づき生成し変化する動的なグラフィックイメージを生成するため，プログラミング環境として Processing を採用した[14]．ゲームやアニメーションのようなインタラクティブメディア用として，主に年少者を対象に Scratch や Etoys などのビジュアルプログラミング環境が開発されている[15]．Processing はグラフィック描画やアニメーションを容易に記述できる機能を持ち，プロトタイプ的作品にとどまらず，実用的な作品にも利用できるという長所がある．しかし，テキストでコーディングするという伝統的なプログラミ

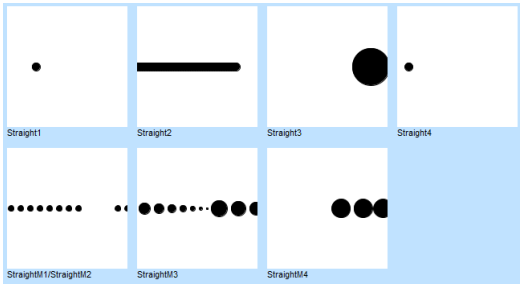

図 3　1 方向 (直線) の運動プログラム
Fig. 3　Programs for the one direction basic movement of line.

ング言語の範疇にあり，ビジュアルプログラミング環境と比較すると学ばなくてもプログラムできるという簡便さはない．その点をふまえたうえで，本教育プログラムではプロジェクト後の実際の作品制作への発展を考え，拡張性，応用性の高い Processing を選択した．3 章で述べたように，模倣，再生産，創作と進む自学によるプログラミングの理解のために 2 種のサンプルライブラリを用意した．「基本運動のライブラリ」と，「センサリインタラクションのライブラリ」である．前者は，グラフィックオブジェクトの基本的な運動のサンプルプログラム集で，プログラムにより動的グラフィックスを生成する基本を学生に示す．グラフィックオブジェクト (たとえば円) の運動を，1 方向の運動，往復運動，回転，振動，拡張の 5 つの基本パターンに分類し，それぞれのパターン内で軌道や速度などを変えた 70 種の運動のプログラムを用意した．

たとえば，1 方向の運動パターンには，直線，波，円，螺旋の軌跡で運動するプログラム，さらに加えて等速/加速運動，大きさやオブジェクトの数を変えて運動するプログラムが含まれる．**図 3** は 1 方向の運動のうち，軌道が直線となるプログラムの実行結果で，左上から順に，等速運動，軌跡を残す等速運動，大きさが変化しながら等速運動，加速運動，同じ大きさの円列の移動，大きさが異なる円列の移動，大きさが変化しながら移動する円列である．このような単純な動きを生成するプログラムによって，大きさが変化することで奥行きが感じられたり，加速することで力を感じたりすることを観察し，オブジェクトの動きが人の感覚にもたらす効果を確認する．

センサリインタラクションと名づけたライブラリは，動的にグラフィックを制御するための方法を示すためのサンプル集で，20 種のプログラムを含み，次の 4 種に分けられる．

① センサからのデータを別の範囲の値にマップして，大きさ，速度，角度，長さなどの変化に使う．たとえば音センサからの 180–220 の値を，100–0 の値に変換し，オブジェクトの大きさの値とする場合，音センサのノイズを無視し，音の大きさとオブジェクトの大きさを反比例させる方法をこのサンプルを通して知る．

② センサからの値をスイッチとして使う．閾値を設定し，それによって運動を開始する，あるいは停めるといった制御の方法を学ぶサンプルである．

③ 時間経過を計測することでタイマ機能を実現する．たとえば，動きを開始後，一定時間経過した後に停める方法を知るサンプルである．

④ 物理的力場における粒子の位置を計算する粒子システムを使い，粒子の動きと入力値とを連動させる．

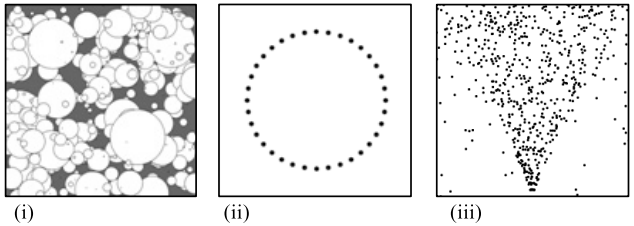

図 4　インタラクションを持つプログラムの例
Fig. 4　Programs with the sensory interaction.

図 4 は，センサからの入力に応答するサンプルプログラムの実行例で，(i) 入力値の変化量により描く円の大きさが変化する，(ii) 入力値が閾値を超えると円が弾けるように広がる，(iii) 噴水のように粒子が吹き出し，入力値により高さが変化する．

ライブラリは，オブジェクトの運動が人の視覚にどのような刺激を与えるかを考察するのを手助けする．学生はライブラリ中のサンプルをセンサを変えて実験を行い，サンプルが示す運動を観察することで，人の行為とグラフィックの動きが生み出す感覚の関係を発見できる．このような実験と観察の後，学生は制作内容を企画するが，その実現のため，企画内容に合わせてまずは真似るプログラムを選び，それを修正，拡張し，加えて必要に応じて順次新たな動きを追加していくよう，学生に指示している．この際，サンプルプログラムは模倣する→再生産する→創作するというプログラミング学習の 3 つのフェーズの出発点となる．

5.4　授業プロセスと作品例

センサリビジョンプロジェクトの授業では，1 つのインタラクティブインスタレーションを，2 人がチームとなって，次のようなプロセスで制作，発表する．

(1)　プロジェクトの目的の理解
(2)　ハードウェアツールキットと基礎エレクトロニクスの理解
(3)　コンピュータプログラミング (Processing) の理解
(4)　ハードウェア・ソフトウェアツールキットを使った実験
(5)　行為と感覚の記述演習
(6)　インタラクションの構想設計
(7)　実装
(8)　作品プレゼンテーション
(9)　作品展示

2008 年と 2009 年の 2 年間，約 30 人のクラス (2 年次生) に対して，5.2 節で説明したツールキットを使い，上記のプロセスの (1) から (8) を半期 14 回の授業の中で実施し，その後学内ギャラリで展示した．2008 年の 16 のチームのうち，13 チームが音センサを使い，テーブルを叩く，弾く，こする，引っかく，あるいは息を吹きかけるといった行為を誘発するコンテンツを制作した．また，4 つのチームが赤外線センサを使い，手を置くあるいはかざすといった行為に基づいたコンテンツを制作した．2009 年は 15 チームのうち，12 チームが赤外線センサを，8 チームが音センサを，4 チームが光センサを使った．2008 年は音センサを使った作品が多かったが，これらの作品では鑑賞者がバンバンと強くテーブルを叩く行為をする傾向があり，行為と視覚のフィードバックによる繊細な感覚の体験を妨げることになりがちだった．これを受けて 2009 年の開始時に荒い行為は適切ではないことを指摘し，その結果柔らかい，ゆっくりとした行為を想定した赤外線センサの使用が 2009 年に増えた．**表 1** に，チームが作品に採用した行為とインタラクションの例を，センサの種類とともに示す．

図 5 は表 1 の a の展示時の写真で，テーブルを叩くことで，半透明の円が現れ，テーブルを真っ赤に染めていく．鑑賞者は絵具の入った見えない風船が破れて，赤い染みがテーブルに飛び散るような感覚を持つ．**図 6** は表 1 の f で水槽の中の金魚とのインタラクションの感覚をイメージしている．**図 7** は表 1 の g で，手を押し付けることで中央の円に弾性を感じたり，手の圧力に堪えきれず弾け飛んだりする感覚を体験する．

6. プログラミング学習から見た授業実践の評価と考察

6.1 作品のプログラム部分の評価

インタラクティブなメディア作品としての魅力と，それを実現するプログラムの記述の適切さや独自性とは一致しない．メディアアート作品として鑑賞者に与える感覚経験や新し

表 1　採用された行為とインタラクション
Table 1　Actions and interactions in projects.

	作品名	鑑賞者の行為 (使用したセンサ)	インタラクション
a	散	叩く，弾く (音センサ)	ランダムに円が現われ，重なる．絵具が飛散るように全体が真っ赤になっていく
b	The steam	吹く (音センサ)	ふたりの鑑賞者が息を吹きかけると，蒸気が揺らめく．
c	戯れ	叩く，こする，上下に動かす (赤外線センサ)	鑑賞者の手に，赤い円がじゃれつくように動く．
d	delete	引っかく，叩く (音センサ)	ぎざぎざの白い線が黒い画面に現われる．カリカリと皮をむく様に，面を白くする．
e	Perfume	手をかざす，押さえる (赤外線センサ)	小円が外周へ向けて弾ける，集まる，あふれ出す．しみだす匂いを手で塞ぐ感覚．
f	KINGYO	手を近づける，手を打つ (音センサ，赤外線センサ)	朱と黒の円が集まる，回転する，散る．水槽の金魚を覗いて遊ぶ感覚．
g	弾	手を押し付ける (赤外線センサ)	中央の赤い円の不透明度と大きさが増加，その後多数の小さい円が飛散る．粘着性の液体が染み出し，面を埋め尽くす感覚．

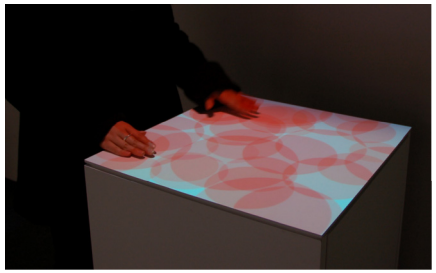

図 5　作品例：散
Fig. 5　Student's work (Scatter).

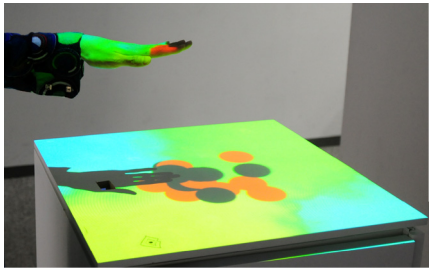

図 6　作品例：KINGYO
Fig. 6　Student's work (Gold fish).

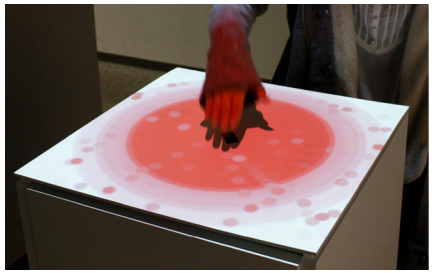

図 7　作品例：弾
Fig. 7　Student's work (Pop).

サンプルプログラム Bubble (図 4 の(i))

```
void draw() {
    pa = sv.input[0]; // 以前の入力ポートの値を保存
    sv.getInputAll();   // 新しい値を入力

    henka = abs(sv.input[0]-pa); //変化量

    x = int(random(width));//円の中央ランダム
    y = int(random(height));


    ellipse(x, y, henka, henka);
    if(sv.input[1] <= level){   //入力 1 が level 以下なら
        background(102); // 画面クリア
    }
}
```

学生のプログラム：散

```
void draw() {
    pa = sv.input[0]; // 以前の入力ポートの値を保存
    sv.getInputAll();   // 新しい値を入力

    henka = abs(sv.input[0]+pa); //変化量
    if(henka > 150){
        cnt=0;
        x = int(random(width));//円の中央ランダム
        y = int(random(height));
        fill(0,henka,300,50);
        ellipse(x, y, henka/2, henka/2);
    }
    if(cnt > 100){
        fill(0,0,255,20); //半透明の四角形を描く
        rect(0,0,width,height);
    }else{
        cnt ++;
    }
}
```

図 8　再生産フェーズの学生プログラムにおける流用
Fig. 8　Appropriation in a students' program in the reproductive phase.

い発見の評価とは切り離して，ソースコードを把握できている 24 の作品を，サンプルプログラムの流用に着目して分析した．そして，プログラミング学習のフェーズと対応した次の 3 段階で評価した．(i) 模倣 (3 作品)，既存プログラムの模倣にとどまる．サンプルのクラスを直接使用している，あるいは単に描画関連メソッド呼び出しの追加や引数の変更を加えたもの．(ii) 再生産 (9 作品)，既存プログラムのメカニズムを流用し，修正したもの．サンプルコードの制御構造に手を加えて流用しているが独自な動きを含まない．(iii) 創作 (12 作品)，独自のメカニズムによる独自の動きを実現した．創作フェーズの 12 作品のうち，ソースコードにプログラムの流用が見られるのが 6，流用がないのが 6 であった．全作品の 75%がサンプルプログラムを明確に流用するかたちで制作されている．図 7 の作品は模倣フェーズ，図 5 は再生産フェーズ，図 6 は創作フェーズの例である．

図 8 と**図 9** は学生作品のプログラムソースの一部を流用元のサンプルと比較したものである (網掛け部分が変更点)．図 8 は図 5 の作品のプログラムで，新たな図形の描画，色や位置の変更だけでなく，入力値による制御を加えて「絵具を散らす」表現をプログラムで実現しているところが単なる模倣から進んだ部分である．図 9 は図 6 のプログラムで，サンプルライブラリの振動する円のクラスに中央へ移動するメソッドを追加している．このわずかな追加が運動に独自性を与え，サンプルを流用しつつ，創作フェーズといえる発展が見られる例である．

プログラミング初学者が中心であるクラスで，プログラムの必要要素を理解し，制御プログラムを作成するにあたり，サンプルプログラムライブラリは，「模倣」し，「変更して再生産」し，「作り出す」出発点として重要な役割を果たしている．

サンプルプログラム VibrateBall (ランダムに振動する円)

```
class VibrateBall{
    float x, y ; // 位置(座標)
    float originX, originY;
    float sx, sy; // 移動速度
    int wide; //動く範囲幅
//    中略
    void display() {
        sx = random(min, max);    //変化量はランダム
        sy = random(min, max);
        x = x+sx;
        y = y+sy;
        if(x>originX+wide){//右
            x = originX+wide;
            sx = -sx;
        }
        //    中略
        ellipse(x, y, d, d);
    }
}
```

学生のプログラム：KINGYO

```
class VibrateBall{
    float x, y ; // 位置(座標)
    float originX, originY;
    float sx, sy; // 移動速度
    int wide; //動く範囲幅
//    中略
    void display() {
        sx = random(min, max);    //変化量はランダム
        sy = random(min, max);
        x = x+sx;
        y = y+sy;
        if(x>originX+wide){//右
            x = originX+wide;
            sx = -sx;
        }
        //    中略
        ellipse(x, y, d, d);
    }
    void Shuugou(int kinjichi,int step){
        if(abs(x-width/2)>kinjichi) x= x + (width/2-x)/step;
        else x = width/2;
        if(abs(y-height/2)>kinjichi) y=y+ (height/2-y)/step;
        else y = height/2;
    }
}
```

図 9　模倣フェーズの学生プログラムにおける流用
Fig. 9　Appropriation in a students' program in the creative phase.

6.2　プログラミングの理解に関するアンケート調査

プログラムによってビジュアル表現を生み出す制作プロセスを経て，学生がプログラミングの理解に対してどのような認識を持ったかを調べるために，プロジェクト開始前にプロジェクトへの期待 (自由記述) と自分の技術的知識を判定するアンケートを，また終了後にプロジェクトの振返りと技術的知識を判定するアンケートを実施した．振返りに関する設問は，プロジェクトは「おおいにおもしろかったか/まったくおもしろくなかったか」と，新しいことを「おおいに学んだか/まったく学べなかったか」の 2 つで，それぞれ 10 段階で判断し，加えて自由記述でその理由の記入を求めた．技術的知識を自己判定する設問はプロジェクト前後とも同じ内容で，次の項目について「よく知っている 10」から「まったく知らない 1」の 10 段階のいずれかの選択を求めた．

- 電気の流れや抵抗について
- センサから値を得ることについて
- A/D (アナログ/デジタル) 変換について
- プログラムを書くことについて
- グラフィックスをプログラムで生成することについて

表 2　プログラム理解認識のプロジェクト前後での変化
Table 2　Change in understanding of programming before and after the project.

変化	2008 年(回答数：28 人，平均 1.5)			2009 年(回答数：21 人，平均 2.0)		
増加	22 人 (79%)	3 以上増加	6(21%)	17 人 (81%)	3 以上増加	9(43%)
		2 の増加	7(25%)		2 の増加	4(19%)
		1 の増加	9(32%)		1 の増加	4(19%)
変化なし	2 人 (7%)			2 人 (9.5%)		
減少	4 人 (14%，−1)			2 人 (9.5%，−1：1 人，−2：1 人)		

2 つのクラスの理解認識の変化量データに等分散を仮定し(f(27,21)=0.81, p=0.31)，スチューデントの t 検定を行った結果，平均値に有意差は認められない(t(47)=0.99, p=0.32).

- インタラクティブなメディアアートについて

学生自身の主観的自己評価を問うこれらのアンケートの項目のうち，「プログラムを書くこと」項目は 3 章で述べたプログラミング学習の利点の (a) 数理的手法による制作力の獲得を反映し，「新しいことを学んだか」項目は (b) 達成感・主体感の獲得と関係する項目と考えている．プログラミングの知識を問うテストによって，知識量の変化を推測することも考えられるが，それは行っていない．本プロジェクトのプログラミング学習は，プログラミング知識の獲得よりも，制作表現の幅を広げるためにプログラムを道具として使う意欲と自信を持たせるところに主目的があり，その点に関する学生の主観的認識を，主にこの 2 つの項目の回答から考察する．

プログラムを書くことについての回答の，プロジェクト前後での変化を**表 2** に示す．2 年間を合わせると 49 人中 39 人 (80%) がプログラムについての理解認識の変化がプラスであったと回答し，26 人 (53%) が 2 段階以上増加したと回答した．クラスには Java プログラミングの半期科目を履修済みの学生が全体で 13 人いた．この 13 人中プログラム理解の認識が 2 以上増加したのは 5 人 (32%)，1 の増加は 6 人 (46%) であった．一方未履修者 36 人では 2 以上増加したのは 21 人 (58%)，1 の増加は 8 人 (22%) であった．初学者の理解認識の増加が，既習者よりも大きいが，どちらも約 80%の学生がプログラミング理解の認識が向上した．

次に，プログラミング理解の認識と学びの実感の関係を確認するため，回答のあった 49 人の理解認識の変化量を縦軸に，新しいことを学んだかどうかの回答を横軸にプロットした散布図 (バブルチャート) を**図 10** に示す．相関係数は 2008 年 0.18，2009 年 0.15 であり，ともに有意な相関はない．図 10 を 4 つの領域に分け，散布図上に A，B，C，D の名を示した．「学んだ感」が平均の 7 (2008 年の平均値は 7.3，2009 年は 7.1) 以上かそれ未満か，また理解認識の変化がプラスかゼロ以下かの 4 領域である．各領域内にプロットされた人

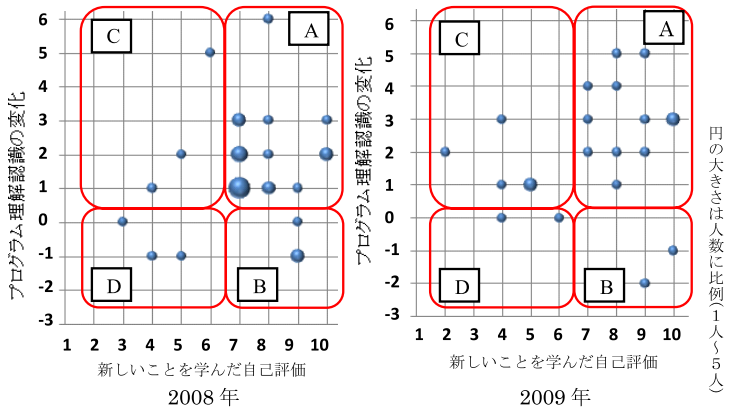

図 10　プログラミング理解認識の変化と学びの実感との関係
Fig. 10　Relationship between change in understanding of programming and sense of learning in a project.

表 3　領域 (図 10) ごとの人数とアンケート回答の平均値
Table 3　Number of data and average results of questionnaires for each area in Fig. 10.

	2008 年(回答数：28 人)				2009 年(回答数：21 人)			
図 10 の領域	A	B	C	D	A	B	C	D
人数	19 (67%)	3 (11%)	3 (11%)	3 (11%)	12 (57%)	2 (10%)	5 (24%)	2 (10%)
面白かった感平均	6.2	8	7	5.7	6.3	6.5	7.0	6.0
プログラミング理解認識の変化平均	2	-0.6	2.7	-0.7	3.1	-1.5	1.6	0

数を，アンケートの「面白かったか」項目の回答平均，プログラミング理解認識変化の平均ともに，**表 3** に示した．学生の自己評価として，理解認識の変化がプラスで，高い学びの実感を示すことを期待するが，両年あわせると 63%の学生がそれに相当する A 領域に属する回答をしている．「おもしろかったか」項目の回答平均はどの領域も同じ程度で，プログラミング学習の困難さがプロジェクト全体への興味を減じてはいない．

6.3　自由記述コメントの分析

プログラミング理解の認識や学んだ感についてどのような認識の学生が，どのようなコメントを書いたかを考察するため，プロジェクト終了後アンケートの自由記述に対し，コーディングとカテゴリ化を行った．グランデッドセオリのオープンコーディングの手法を援用し，8 つのカテゴリを抽出し，肯定的と否定的の 2 つに分類した (**表 4**)．カテゴリ「制作姿

表 4　自由記述コメントのオープンコード化によるカテゴリ
Table 4　Categories for students' comment by open coding.

肯定的カテゴリー		否定的カテゴリー	
⬢	プログラムを(少しは)理解した(a)	✖	主体感なし
■	数理的手法による表現(a)	●	プログラムは困難
●	主体感達成感あり(b)	⬟	制作姿勢への自省
▲	将来へ向けての意志(c)		
★	プログラムによる表現の可能性(c)		

表内のマークは表 5，図 11 内のマークと対応．
(a) (b) (c)は，3 章のプログラミング学習の利点と対応

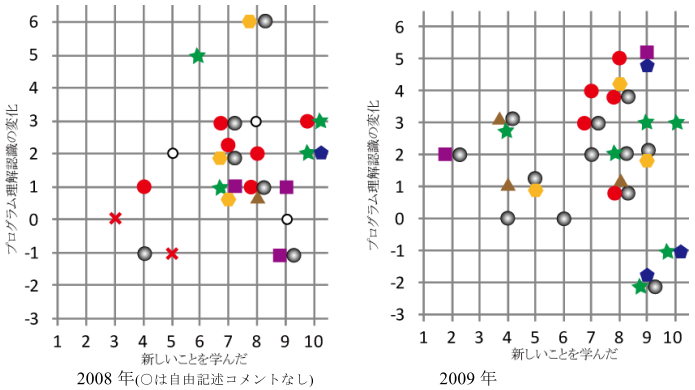

図 11　自由記述コメントのカテゴリと図 10 上のデータとの関係
Fig. 11　Relationship between categories of comments and data in Fig. 10.

勢への自省」は，「プログラムはむずかしい！」と言い放って，理解の困難さの原因をプログラミング側におくのではなく，「他の課題と重なり集中できなかった」など自分側におくコメントである．肯定的カテゴリが，3 章で述べたプログラミング学習の利点 (a)，(b)，(c) のいずれと関係するかを表 4 のカテゴリ名の横に合わせて示した．そして，表 4 でカテゴリにつけたマークを図 10 の点の上に重ねて**図 11** とした．実際のコメントの一部を**表 5** に示す．表 5 のマークは表 4 と対応しており，カテゴリに属するコメントの具体例が分かる．図 10 上のどの学生のコメントかが分かるように，図 10 上の領域名と座標も示した．

肯定的カテゴリは，図 10 の A，B，C いずれの領域にも現れる．肯定的カテゴリのうち，利点 (a) に対応するカテゴリのコメントは，1 例を除き，A ならびに C 領域に現れる．また，「主体感達成感あり」カテゴリも A，C 領域だけに現れる．プログラミングへの理解が進ん

表 5　プロジェクトで学んだことに関する自由記述コメント
Table 5　Comments about what you have learned.

図 11 での位置*1	マーク*2	プラスの振り返りコメント	マーク*2	「プログラミングが大変」だったことに関するコメント
A 2008 年 [7,3]	●	思った動きが実現した時はとてもうれしい．	●	持っている能力が少なすぎて，できることが少なかったのが残念．
A 2008 年 [8,6]	⬢	以前はわからなかったプログラムを理解できるようになった．	●	プログラムが難しく，思うとおりに動かなかったときは苦しかった．
A 2008 年 [7,1]	●	プログラムを使って自分の感覚を形にしていくのがおもしろかった．	●	プログラムなかなかうまくいかなかったので，ちょっとしんどかった．
A 2009 年 [10,3]	★	自分は(助けは要るが)にもできるという可能性．	●	プログラムが難解で，サポートしてもらわないとわからなかった．
A 2008 年 [8,1]	▲	もっとプログラミングを理解してもう一度作ってみたい．	●	やりたいこととプログラミング能力が一致せずにとても大変だった．
B 2008 年 [9,−1]	■	プログラムを変更していくうちにおもしろい動きを発見できた	●	プログラムの難しさを学んだ．かなりつらかった．
B 2009 年 [9,−2]	★	センサ，I/O モジュールを使い，プログラムを動かすことは広い分野への可能性が感じられて新鮮．	⬟	多忙な状況の中，忙しくちゃんとプログラムについて考えられなかった．
C 2009 年 [5,1]	⬢	プログラムを前よりは少し理解できるようになった．	●	プログラムが思った通りに進まないことが多かった．
C 2009 年 [4,1]	▲	思った通りにならなくても，それに近づけるようにする向上心を学べた	●	プログラミングが難しすぎた
C 2009 年 [4,3]	★ ▲	プログラムによっていろいろな結果が得られるところがおもしろい．思い通りにならなくても，理想に近づけようとするくじけない心を学んだ．	●	プログラミングはとても難しい．
D 2008 年 [3, 0]			✖	プログラミングが複雑で，自分で作り上げた気がしない．
D 2008 年 [4,−1]			●	プログラミングの知識のなさを学んだ．

*1：A,B,C,D は図 10 の領域を，[M, N]は散布図上の座標で，M は学んだ感を，N はプログラミング理解度の変化を表す．
*2：表 4，図 11 のマークと対応している．

だという認識と数理表現の面白さの発見が，主体感を高める関係にある．「将来へ向けての意志」や「表現の可能性」カテゴリは，A，B，C 領域に偏りなく見られる．これらのコメントにはプログラミング理解認識がマイナスの変化であっても，「もう 1 度やってみよう」「プログラムを動かすことは広い分野への可能性が感じられて新鮮」と，プログラム理解の認識に関係なく，プログラムの知識を次の活動に活用しようとする意欲が表現されている．自由記述コメントのカテゴリ分析から，9 割の学生が A，B，C 領域にあたる認識を示しており，これらの学生はプログラミング学習の利点 (a)，(b)，(c) のいずれかを得ていると考える．

否定的カテゴリの「プログラムは困難」のコメントは，4 領域いずれにも現れるが，A，B，C 領域の学生は「プログラミングは難しく」ても，学ぶ意味を肯定的にとらえているのに対し，D 領域には肯定的コメントはない．また，否定的カテゴリの「主体感なし」は D のみに見られる．1 割に相当する D 領域の学生は表 3 に示したようにプロジェクト全体の面白

さの認識は他の領域の学生と違いはないが，プログラミング学習の利点を得られていない．

6.4　インタラクティブなグラフィックス制作とプログラミング学習意欲

3 章でも述べたように，本教育プログラムの中で行うプログラミング学習には次の特徴があり，これらがプログラミング学習の利点獲得へつながると期待した．

(1)　プログラムの結果が，インタラクティブなモーショングラフィックスとして直接見えることで，驚きや楽しみを体験する．真似る，変える，新たに作り出すという学びのフェーズを経る中で驚きが繰り返される．

(2)　他のチームからの刺激を受けながら，行為とグラフィックス表現との新たな関係を探る過程で，数理的な制御を工夫し，思考を深める．

アンケート回答の「数理的手法による表現」「主体感達成感あり」カテゴリのコメントから，この点を見ると，たとえば「プログラムを変更していくうちにおもしろい動きを発見できた」(2008 年 [9, -1]) や「(コンパイルが) 成功したときに画面上にぱっと広がる結果が想像どおりであったり，意外な仕上りであったり，つねに驚きとある程度の緊張感が面白かった」(2009 年 [7, 4]) の記述が表しているように，プログラムを実行した結果がインタラクティブなモーショングラフィックスとして現れることが，動きを実現するプログラムの探求をおもしろくしている様子が分かる．模倣から再生産，創作へと進める間驚きが繰り返されることが重要で，それが学習の意欲を高めたと考える．その面白みを学習当初から実感し，驚きを継続するために，模倣の出発点であるサンプルライブラリが重要な役割を担っている．

6.5　プログラミング理解の認識と作品のプログラム部分の評価

図 11 にプロットされているどの学生が，6.1 節で評価したどのフェーズのプログラムを書いたかを，**図 12** に示した．アンケート回答とソースコード提出がともに整っている学生のみをプロットしている．A，B，C 領域では大きな偏りはなく，プログラミング理解に関する回答とプログラム部分の評価との間にはっきりとした傾向や関係は見出せなかった．一方，データ数は少ないが，D 領域にプロットされた回答の学生のプログラムは皆創作フェーズであった．創作フェーズのプログラムを書くには，プログラム要素を理解したうえに，新しい動きを実現する数理手続きを生み出す必要があるが，D 領域にプロットされた学生の回答とそのプログラムを見ると，高度な動きを目指しすぎて，プログラミングに苦労し，構想を変更して完成させたものの，困難さが勝って主体感を得られない様子が分かる．たとえば，2009 年の [6, 0] の学生は粒子システムを使い，オリジナルな動きを含む創作フェーズのプログラムを制作したが，ライブラリを使いこなせず，構想どおりにはならなかったため達成感が得られなかった．粒子システムを使ったサンプルは，数理的表現の美しさや発展性

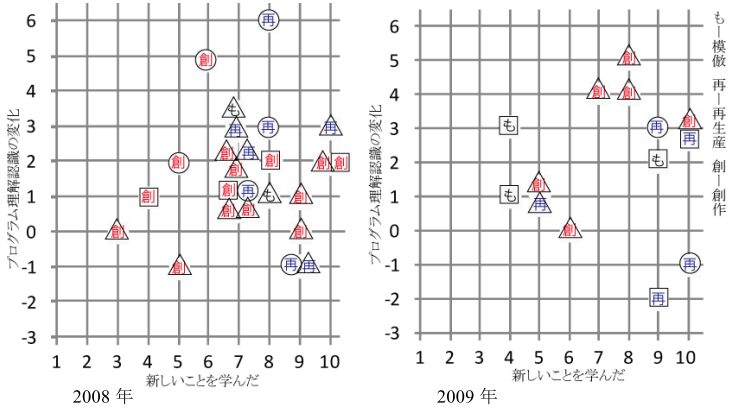

図 12　プログラムの評価と図 11 上のデータとの関係
Fig. 12　Relationship between evaluation of programs and data in Fig. 11.

を示すよい例であるが，シンプルな運動と比較すると高度である．そのようなプログラムを模倣の出発点とし，独自の動きにチャレンジしたことが，結果として反対の効果を招いた．途中で挫折せず，プログラミングの学習意欲を持続するためには，出発点となるサンプルプログラムのシンプルさが重要であることを示唆している．

メディアアート作品としての評価は，鑑賞者の主観で成立するので，プログラミング学習の認識や意欲との関係を論じられないが，参考までに，授業終了時の教員による作品の評価を図 12 にプロットの形状で示した．鑑賞者に与える体験の豊かさ，アイディアの斬新さを教員 2 人が主観的に 3 段階で評価したものである (○は優，□は良，△は可の 3 段階)．創作フェーズのプログラムは作品としての評価は C が多い．オリジナルな動きを導入して行為と感覚の体験を実現しようとしたが，実装が中途半端な形にとどまった結果，作品としての評価が低くなっている．模倣フェーズのプログラムは，作品としての評価が B または C で，模倣のままだと行為と感覚の新しい体験を生み出すまでに至らないという結果が表れている．

7. ま と め

フィジカル・インタラクションにより，鑑賞者の行為に応じて生成し変化する動的なグラフィックコンテンツの制作を通して，行為と感覚の複雑かつ繊細で多様性に満ちた関係性を探究する造形基礎教育プログラムを構築し，実践した．

授業内ではプログラミングの学習にとれる時間が限られるため，サンプルライブラリを使

い，「模倣する→再生産する→創作する」の段階的フェーズを通して自学により理解を進めた．サンプルライブラリは，シンプルなグラフィックの運動によるインタラクションが人の感覚にどうとらえられるかの発想の基本として，また，ソースコードを流用することでプログラムの土台として，有効に機能した．数理的表現の工夫がインタラクティブなモーショングラフィックスとしてただちに反映されることが，驚きを生み，それが 3 つのフェーズの学習過程を通して繰り返されることが，学習の意欲を高めた．結果として 8 割の学生にプログラミングの理解の認識の向上をもたらした．9 割の学生はプログラミングはむずかしいと考えながらも，プログラミング学習に期待される利点を反映する認識を示しており，モーショングラフィックスを生成するプログラム制作が学習意欲を向上させたと考える．しかし，1 割程度の学生はプログラミングに否定的な徒労感を覚えることになった．サンプルプログラムのソースの分かりやすさ，種類などの改良を行うことを中心に，教育内容の改善を図っていきたい．

謝辞　制作キットの開発，プロジェクトの実施における，真下武久氏，作花愛梨氏の協力に感謝する．なお，本研究は同志社女子大学研究助成金の支援を受けた．

参 考 文 献

1) Ariga, T. and Mori, K.: Sensory Vision-Development of a Course for Physical Interaction and Graphics, *Computers & Graphics*, Vol.34, No.6, pp.800–810 (2010).
2) Papert, S.: *MINDSTORMS Children, Computers, and Powerful Ideas*, 2nd Edition, Basic Books, New York (1993). 奥村貴世子（訳）：マインドストーム—子供，コンピューター，そして強力なアイディア，未来社 (1995)，引用部分は邦訳 p.36.
3) Ursyn, A., Mills, L., Hobgood, B., et al.: Combining Art Skills with Programming in Teaching Computer Art Graphics, *ACM SIGGRAPH Computer Graphics*, Vol.31, No.3, pp.60–61 (1997).
4) Ursyn, A. and Scott, T.: Web with art and computer science, *Proc. SIGGRAPH 2007 Educators Program*, ACM (2007).
5) Ribner, N. and Metaxas, T.: The Art and Science of Multimedia: An Interdisciplinary Approach to Teaching Multimedia at a Liberal Arts College, *Proc. SIGGRAPH'98 Educators Program*, pp.46–51, ACM (1998).
6) Wolz, U.: Teaching design and project management with logo RCX robots, *Proc. SIGCSE'01*, pp.95–99, ACM (2001).
7) Lawhead, P., Duncan, M., Constance, B., et al.: A road map for teaching introductory programming using LEGO© mindstorms robots, *Proc. ITiCSE-WGR'02*, pp.191–201, ACM (2002).
8) Kim, J.K., Coluntino, D., Martin, F., et al.: Artbotics: community-based collaborative art and technology education, *Proc. SIGGRAPH 2007 Educators Program*, ACM (2007).
9) PicoCricket, available from ⟨http://www.picocricket.com/⟩ (accessed 2011-09-19).
10) Kuwakubo, R.: DIY hardware: Reinventing hardware for the digital do-it-yourself revolution, *Proc. SIGGRAPH ASIA 2009 Art Gallery & Emerging Technologies*, pp.66–67, ACM (2009).
11) Resnick, M.: All I really need to know (about creative thinking) I learned (by studying how children learn) in kindergarten, *Proc. 6th ACM SIGCHI Conference on Creativity & Cognition* (*C&C'07*), ACM (2007).
12) The ITEST Small Group on Computational Thinking, White Paper Working Group: Computational Thinking for Youth, available from ⟨http://itestlrc.edc.org/resources/computational-thinking-youth-white-paper⟩ (accessed 2011-09-19).
13) Kobayashi, S., Endo, T., Harada K., et al.: A Reconfigurable I/O Module and Software Libraries for Education, *Proc. NIME'06*, IRCAM (2006).
14) Processing, available from ⟨http://processing.org/⟩ (accessed 2011-09-19).
15) Resnick, M., Ohshima, Y., Flanagan, M., et al.: Growing Up Programming: Democratizing the Creation of Dynamic, Interactive Media, *Proc. CHI'09* (2009).

(平成 23 年 3 月 8 日受付)
(平成 23 年 9 月 12 日採録)

有賀　妙子（正会員）
1980 年東京工業大学大学院理工学研究科修士課程修了．現在，同志社女子大学情報メディア学科教授．著書に，『すべての人のための Java プログラミング』（共立出版），『マルチメディア表現』（実教出版）等．情報デザインコンテンツ制作の教材開発とその評価に関する研究に従事．ACM，日本デザイン学会各会員．

森　　公一
1984 年大阪教育大学大学院教育学研究科修士課程修了．現在，同志社女子大学情報メディア学科教授．メディアアート作品の制作，理論研究に従事．現在は脳の血行動態等の生体情報を用いたバイオフィードバックによる表現の可能性を探求．日本映像学会会員．