2006－MUS－68 (5)
2006／12／15

DIPSにおける新たなプログラミングインタフェース "DIPS-PIC"

濱野 峻行
国立音楽大学
hamano@kcm-sd.ac.jp

松田 周
Digital Art Creation
名古屋大学大学院情報科学研究科
国立音楽大学
shu@dacreation.com

概要: DIPS はMax/MSP のエクスターナルオブジェクトとして実装されているリアルタイム映像処理・生成環境であるが、操作性の向上が課題となっていた。「DIPS-PIC(Programming Interface for Creators)」はDIPS の機能に創作者の視点に立った直感的なプログラミングインタフェースを追加し、開発を容易にするものである。本稿ではその実例を紹介し、今後の可能性について考察する。

New Programming interface "DIPS-PIC" in DIPS

Takyauki Hamano
Kunitachi College of Music
hamano@kcm-sd.ac.jp

Shu Matsuda
Digital Art Creation
Graduate School of Information Science Nagoya University
Kunitachi College of Music
shu@dacreation.com

Abstract: DIPS is a realtime image processing and generating environment implemented as external objects. It had a problem in usability. "DIPS-PIC(Programming Interface for Creators)" adds new intuitively programming interface that will ease the programming and support the creator. In this paper, I would like to introduce some examples and consider about the future.

1. はじめに

DIPS(Digital Image Processing with Sound)は1997年より現在まで開発が続けられてきている、映像処理や生成をリアルタイムで実行できる環境である。音と映像のインタラクションに特化しているのがこの環境の特徴である。DIPS はSGI 社によって開発されたフリーの 3 次元グラフィクスライブラリ「OpenGL」をベースにしているため、音響処理の情報を非常に高速に映像へ反映させることができる。またDIPSには100種類以上の命令があり、ビデオカメラ入力や映像解析、ムービーファイルの再生など、各媒体との連携機能が充実していることも大きな特徴である。最新バージョンのDIPS3からMax/MSPのエクスターナルオブジェクトとして実装されている。

DIPS のオブジェクトはOpenGL のパラメータや命令を詳細に設定したり実行したりすることができる。その反面オブジェクトやパラメータの量が増えることとなり、プログラミングを専門としない一般のユーザには使いづらいものであった。

そこで創作者が直感的かつ容易に制作できるようにプログラミングインタフェースを追加するのが、「DIPS-PIC」である。DIPS-PIC を用いることで、パラメータを詳細に設定できるという今までの柔軟性を損なうことなく簡潔なプログラミングが可能となり、創作者のアイディアをダイレクトに映像表現へ反映させることができるようになった。その使用例と課題点を、今後の可能性と併せて述べる。

2. DIPS-PIC について

2.1 DIPS-PIC の特徴

「DIPS-PIC」は DIPS Programming Interface for Creators の略であり(以下 PIC)、先に述べたように

DIPS の操作性を向上させるものである。その特徴を以下に 4 つ挙げる。

(1) 独自のプログラミングインタフェース

まず一つ目の特徴は、新しいプログラミングインタフェースによる操作性の向上である。PIC は約 30 のオブジェクトから成り立っており、各オブジェクトに対して映像処理・生成の機能一つが割り当てられている。それぞれの PIC オブジェクトは独自の「設定ウィンドウ」を持っており、パラメータが一覧として表示される。各パラメータは設定ウィンドウ上のスライドバーやナンバーボックスで指定・変更できる。設定ウィンドウが各オブジェクトに予め用意されているので、ユーザの負担が大きく軽減された。

(2) OpenGL についての知識が不要

また PIC は、OpenGL について精通していない人でも手軽に使うことができる特徴を持つ。例えば DIPS では各オブジェクトが OpenGL の機能一つに対応しているので、単純な図形を表示させるだけでも多くの初期化命令を実行しなければならない。それは一般のユーザにとっては大きな障壁となっている。そこで PIC では、一つのオブジェクトに複数の DIPS の機能を包括させた。ユーザは PIC の一つのオブジェクトに対して一つの映像処理・生成機能が割り当てられていると捉えて使うことができるので、OpenGL の技術を習得していなくても利用することができる。

(3) プログラミングの柔軟性

次に、プログラミングの柔軟性を DIPS から引き継いだことが特徴に挙げられる。PIC では各オブジェクトにアーギュメントで初期値を設定したり、インレットから値を入力したりすることができる。従って値の設定をオブジェクトの外部から詳細に指定できるという今までの DIPS の利点を損なわずに、より使いやすい環境にすることができた。上級者にとっても満足に映像プログラミングができる環境となった。

(4) パラメータの自動保存

最後の特徴は、全ての PIC オブジェクトのパラメータは終了時に保存されるということである。PIC はパッチを閉じる際に全ての PIC オブジェクトのパラメータをリスト化し、「<パッチ名>.dips」という名前のファイルを作って自動的に保存する。そして再びパッチを開く際に「<パッチ名>.dips」ファイルを読み込み、自動的に各オブジェクトに保存されている値を設定する。旧来は手動でアーギュメントとして値を設定したり loadmess オブジェクトを用いて値が設定されるようにしたりしなければならなかったが、PIC では全てのパラメータが自動的に保存されるため労力の軽減と生産性の向上につなげることができた。

2.2 DIPS-PIC の使用例

PIC の基本的な使用例を具体的に示していきたい。

まず PIC を利用するための準備が必要である。PIC の実体はサブパッチであるため、Max/MSP の File Preferences より PIC のオブジェクト群の入っているディレクトリのパスを指定することで利用可能となる。

では Max/MSP の上で PIC を使って画面上に物体を表示させてみたい。PIC の中で一番最初に必要なのは「dpic.root」オブジェクトである。このオブジェクトは DIPS の描画ウィンドウを生成し、描画ウィンドウの位置やサイズ、描画のフレームレートなどを制御するためのものである。このオブジェクトをダブルクリックすると設定ウィンドウが表示され、ウィンドウ位置やサイズの変更ができる(図 1)。

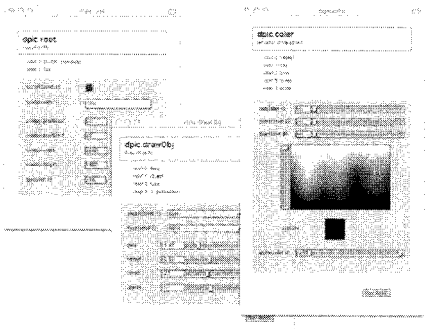

図 1　DIPS-PIC の設定ウィンドウの例

設定ウィンドウには「show detail」ボタンがあり、クリックするとオブジェクトの処理の詳細な内容が表示される。これにより上級者は全ての処理内

容を把握することができる。

「dpic.root」オブジェクトの次に必要なのは「dpic.clear」オブジェクトである。このオブジェクトは、画面を任意の色で塗りつぶすためのものである。「dpic.root」オブジェクト同様、オブジェクトをダブルクリックすると設定ウィンドウが表示され、そこから色を選択することができる。色はカラーピッカーを使って選ぶことができるので、容易に変更可能となっている。

次に画面に物体を描画する。ここで使うのは「dpic.drawObj」オブジェクトである。設定ウィンドウのプルダウンメニューから 9 種類の物体の中から描画するものを選択でき、輪郭のみ表示するか塗りつぶして表示するかを選択することができる。描画する物体を選択すると、選んだ物体に対する設定項目が下方に表示される。以上の 3 つのオブジェクトを縦方向で順番に並べて接続すると、各オブジェクトが有効となって物体が描画される。

最後に自動保存の有効化のために closebang オブジェクトを作り、dpic.root のインレットに接続する。これによって終了時に closebang から bang メッセージが dpic.root へ送られ、自動的に DIPS-PIC オブジェクトの全てのパラメータを保存する。以上の手順によって描画した結果が、下の図 2 の左である。

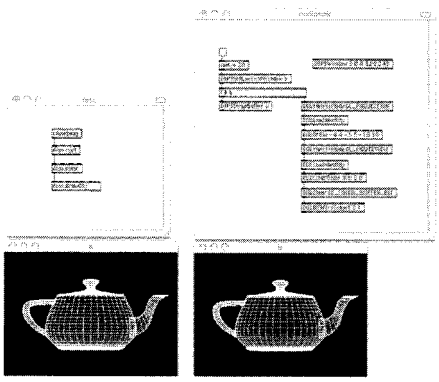

図 2　DIPS-PIC 使用(左)と不使用(右)による物体描画パッチの比較

このように 4 つのオブジェクトのみで物体を DIPS の画面に表示することができ、ユーザは簡単に映像制作に取りかかることができる。上の図 2 の左は PIC を使用して物体を描画したパッチの例、右は使用せずに本来の DIPS の機能のみで描画したパッチの例である。PIC によってパッチを単純にすることが可能となった。

2.3 その他の基本的な DIPS-PIC オブジェクト

PIC のオブジェクト群の中で根幹となっているものは約 20 個である。先の具体例に出てきたオブジェクトの他に、主要なものを以下に列挙する。

(1) 描画関連
dpic.point: 点の描画
dpic.line: 線の描画
dpic.circle: 円や多角形の描画
dpic.rect: 長方形の描画

(2) 色・混合関連
dpic.color: これから描画する物体の色の指定
dpic.blendOn(Off): 既に描画したものとこれから描画する物体との色の混合の設定
dpic.lightOn(Off): 照光処理の設定、現実の物体のように光が当たる様子を擬似的に再現する

(3) 空間設定関連
dpic.setMatrix: 座標空間の設定
dpic.move: 物体の移動
dpic.rotate: 物体の回転
dpic.scale: 物体の拡大・縮小

(4) テクスチャ関連・その他
dpic.textureOn(Off): テクスチャの有効化と無効化、テクスチャの指定
dpic.loadimage: 画像ファイルを読み込んでテクスチャに割り当てる
dpic.copytotex: 描画した画面をテクスチャに貼り付ける
dpic.video: コンピュータに接続されたビデオカメラから映像を取り込む

3. エフェクトオブジェクト

エフェクトとは制作した画像に対して任意に変化を与えるものであるが、PIC にはエフェクトのオブジェクトが多く含まれている。各エフェクトオブジェクトは、パッチ内で一番最後に実行されるオブジェクトの下に配置・接続するだけで、すぐに使うことができる。エフェクトは描画する毎

に一回適用され、映像に対してリアルタイムに効果を得ることができる。またエフェクトをいくつも重ね合わせることも可能である。エフェクトオブジェクトは、OpenGL エフェクトと Core Image エフェクトとに大別される。それぞれについて説明していきたい。

3.1 OpenGL エフェクト

OpenGL エフェクトは OpenGL の基本的な機能のみを利用して作られたエフェクトであり、約 10 種類のオブジェクトがある。

OpenGL エフェクトには、Motion blur、Recursivev blur、Radial blur の 3 つのブラーエフェクトが含まれている。これは後に述べる Core Image とは異なり、動画に対して効果的なエフェクトである。またディゾルブや波状変形、タイル状逆転処理などの独自のエフェクトも含まれている。

PIC の基本オブジェクトと同様に、オブジェクト内の設定ウィンドウからエフェクトのパラメータを指定することができる。

3.2 Core Image エフェクト

「Core Image」とは、OpenGL の画像に対してフィルタやトランジション、エフェクトなどを適用することのできるものである。これは Mac OS X Tiger 以降、標準で組み込まれている機能である。Core Image は「Image Unit」と呼ばれるプラグイン群を呼び出すことで各エフェクト等を実行する。Mac OS X Tiger に含まれているプラグインは約 110 種類ある。Core Image を使うことで、複雑な効果を容易に得られることができる。Core Image エフェクトは OpenGL エフェクトと同じように使うことができる。

このように PIC では非常に多くのエフェクトを簡単に扱えるようになったので、映像表現の幅を大きく広げることができるようになった。

4. 課題点

研究によって現出した問題を以下に挙げる。

4.1 自動保存機能

PIC のパラメータは dpic.root オブジェクトが終了時に bang メッセージを得ることによって保存される。そのため、メインパッチ上でパッチが閉じられることを検知するために closebang オブジェクトが必要となる。closebang オブジェクトを接続することを忘れると、再びパッチを開いたときに PIC オブジェクトの全ての値は初期値に設定されてしまう。今後、dpic.root オブジェクトのエクスターナルオブジェクト化を含めて、解決方法を検討していきたい。

4.2 同一パッチのオープン

Max/MSP では一つのパッチファイルを複数開くことができる。このため誤って PIC のパッチを複数開いた場合、DIPS の描画画面が生成できなかったり、パラメータが誤って保存されてしまう危険性がある。現在は dpic.root オブジェクト内で JavaScript の処理を行い、同一パッチを複数開いても処理を行わないように対処しているが完全に安全な方法とは言えない。今後は他の方法によりパッチにロックをかけるよう模索したい。

5. まとめと今後の可能性

以上本稿では、DIPS の新しいプログラミングインタフェース「DIPS-PIC」について述べた。「DIPS-PIC」によって映像生成・処理における技術的なハードルを低くすることができたといえる。またリアルタイムに複雑なエフェクトを実行することができるので、音楽との連携をとった高度な映像がより作りやすくなったといえる。これは音と映像のインタラクティブな作品を制作する者にとっては、大きなメリットだと思われる。

今後も様々な映像制作環境の研究を重ねていき、創作者にとってより有益な環境を提供できるよう開発を進めて参りたい。

参考文献

[1]Miyama,C, Rai,T, Matsuda,S, Ando,D, "Introduction of DIPS Programming Technique", in Proceddings of the International Computer Music Conference 2003.

[2]OpenGL Architecture Review Board, Maison Woo, Jackie Neider, Tom Davis: The Official Guide to Learning OpenGL

[3]Apple Computer, Inc.: Apple - Mac OS X - Core Image - http://www.apple.com/macosx/features/coreimage/