音楽情報科学 38－5
（2000．12．17）

DIPSプログラミングの実際

国立音楽大学大学院音楽研究科
橋田 光代　　松田 周*

国立音楽大学音楽デザイン学科
美山 千香士　　安藤大地

概要

"DIPS"(Digital Image Processing with Sound)は，松田周によって jMax に実装されたリアルタイム画像処理オブジェクト群である．jMax がオブジェクト指向 GUI プログラミング環境で元来実現する MIDI 信号処理，音響信号処理に加え，OpenGL をも含めたさまざまな画像信号処理を可能にする．さらに，これらの信号処理機能間の相互作用を容易に実現する．本稿では DIPS プログラミングの実際を各項目に分け紹介する，また，DIPS エクスターナル・オブジェクトの開発状況についても報告する．

DIPS programming tutorial

Mitsuyo Hashida, Chikashi Miyama, Daichi Ando, Shu Matsuda*
Sonology Department, Kunitachi College of Music

Abstract

"DIPS"(Digital Image Processing with Sound), developed by Shu Matsuda, is a set of jMax external objects for the real-time image processing. The DIPS makes it possible to handle image processing including OpenGL programming in the object-oriented GUI programming environment of jMax, along with its original MIDI and audio signal processing. Thus it realizes the interaction among these three signal processing more easily. Here we would like to introduce the practical DIPS programming and later discuss DIPS external objects under development at our studio.

1. はじめに

SIGMUS36 において，jMax 上に実装されたリアルタイム映像処理オブジェクト群"DIPS"と DIPS の機能を拡張する DIPS エクスターナルオブジェクトについての報告を行った．今回はそれを踏まえて，DIPS の核となる OpenGL のプログラミングを，実際に Max インターフェイス上で行うための方法を解説する．また，現在開発中の DIPS エクスターナルオブジェクトについても報告を行う．

本稿では以下のカテゴリに分けて紹介する．

1. プログラミングの実際
   - A. 全体の流れ
   - B. 初期設定
   - C. 3D モデルの作成
   - D. 視界変換モデリング変換
   - E. 照光処理
   - F. テクスチャ・マッピング
   - G. 二次元画像のラスタ
2. DIPS エクスターナルオブジェクトの開発状況

---

* 現在，デンハーグ王立音楽院(オランダ)
* Koninklijk Conservatorium

2. プログラミングの実際

A. 全体の流れ

DIPS 上でのプログラミングは大きく次のような流れとなっている. (figure 1.)

OpenGL 関数を呼び出す DIPS オブジェクトの名称は, 次に示すように, OpenGL 関数であることを示す「gl」を大文字にし, さらに頭に大文字の「D」を付加している. 後述する各種パッケージオブジェクトは DGL の代わりに DPX などがつく.

DIPS では, 各関数オブジェクトはインレットからの"bang"信号を受けるとただ一度実行される. そのため, 連続したオブジェクト描画(アニメーション)を行うには, Max の"metro"オブジェクトを利用し, 連続して"bang"信号が流れるようにする. ただし, 各種ステータス管理に必要な初期設定オブジェクトは本動作の最初に一度だけ実行できれば良いので, 別途にまとめて"loadbang"を送れば良い.

メインの処理においては, 描画表示を行うウィンドウの指定"DGLXMakeCurrent", オブジェクト描画に関する初期化"DGLLoadIdentity", 描画画面の消去カラーの設定と消去"DGLClearColor", "DGLClear"を行った上で, 3D モデルやビデオ映像処理の描画を行う. 処理の最後にある"DGLXSwapBuffars"は, 描画が高速かつ滑らかに処理されるようにバッファを交換するためのオブジェクトである.

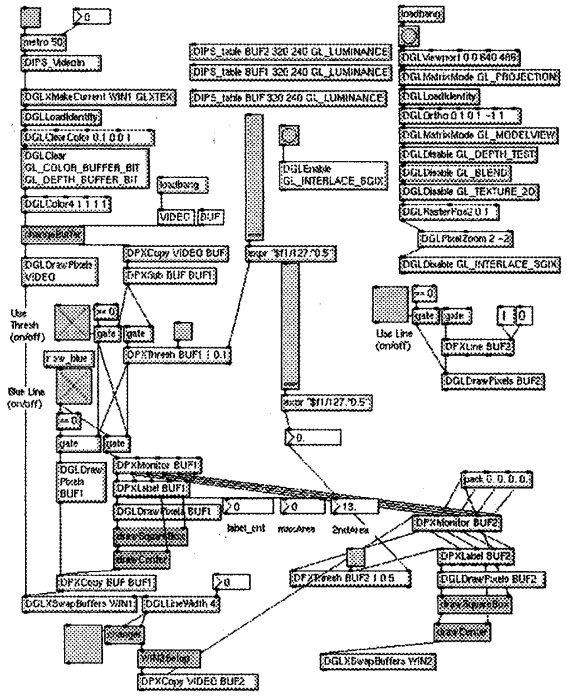

figure 1.

B. 初期設定

描画される空間世界の定義を全処理の最初に行う. この図では"DGLViewport"から"DGLMatrixMode[GL_MODELVIEW]"までが該当する. (figure 2.)

glEnable()と glDisable()は各種描画処理に必要なステータスの管理のパラメータ(GL_LIGHTING, GL_DEPTH_TEST, GL_TEXTURE_2D など)を指定する. これらは各 jMax パッチのメイン処理に依存するので, 必ずしも必要であるとは限らない.

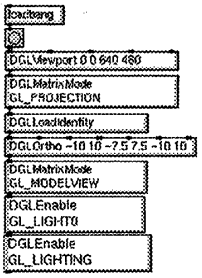

figure 2.

C. 3D モデルの作成

OpenGL で単純な幾何学的描画プリミティヴの生成を行うには, 下記のように glBegin(), glEnd()を用いて, その間に描画で必要な頂点を示す glVertex*()を用意する.

```
glBegin(GL_POLYGON);
        glVertex*(x1, y2, ...);
        glVertex*(x2, y2, ...);
                :
        glVertex*(xn, yn, ...);
glEnd();
```

同じ処理を DIPS で行うと次のようになる. (figure 3.)

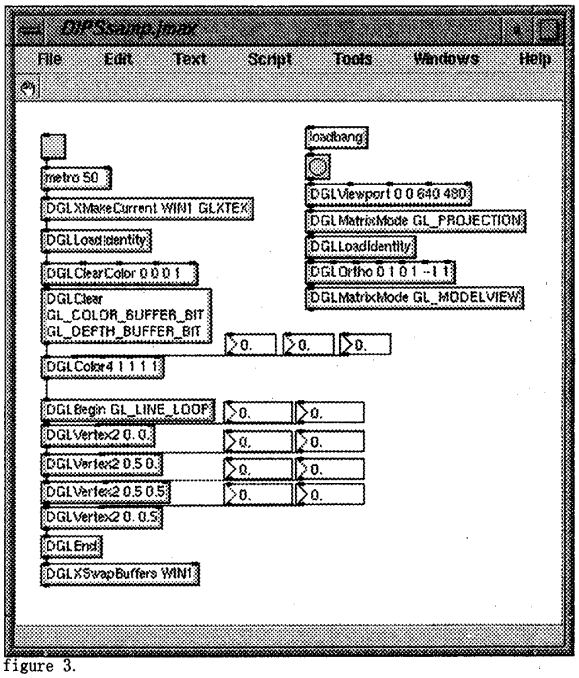

figure 3.

"DGLBegin"で指定している属性は, 以降に指示された頂点をどのように描画させるかを表している. 図ではポリゴンを表示させるようにしているが, 他にも点(GL_POINTS)や線分(GL_LINES, GL_LINE_STRIP, GL_LINE_LOOP)として描画することも可能である.

"DGLBegin"以降に並ぶ"DGLVertex2"オブジェクトは描画ウィンドウ中の頂点座標を示す. オブジェクト名の最後につく数字 (図では"2") はオブジェクト描画に用いる頂点座標の数を示し, 最低2つ(x, y)から最高4つ(x, y, z, w)の値を用いることができる. この数字の指定に従って, 続くアーギュメントの指定数も変わる. オブジェクト上方にあらわれるインレットは, このアーギュメントの値を変更させることができる.

D. 視界変換とモデリング変換

a. 視界変換

視界変換(カメラワーク)を行う場合は, モデルのレンダリングの前に, カメラの位置, カメラの方向, カメラの上方向

を，gluLookAt()によって指定する．右図はその一連のプロセスを示したものである．(figure 4.) 図ではカメラを，座標(5.0, 2.0, -13.0)の位置に設置し，座標(0.0, 0.4, 0.0)にレンズを向け，y 軸の正の方向を上方向とする設定が施されている．

b. モデリング変換

モデルを移動や回転，拡大・縮小させるには，それぞれ glTranslate()，glRotate()，glScale()の関数を用いる．これを DIPS 上で行うと図のようになる．(figure 5.) 図ではモデルを(2.0, 1.0, -20.0)移動し，(18, 36, 18)度回転させ，(2.0, 1.0, 1.0)の拡大・縮小を施している．

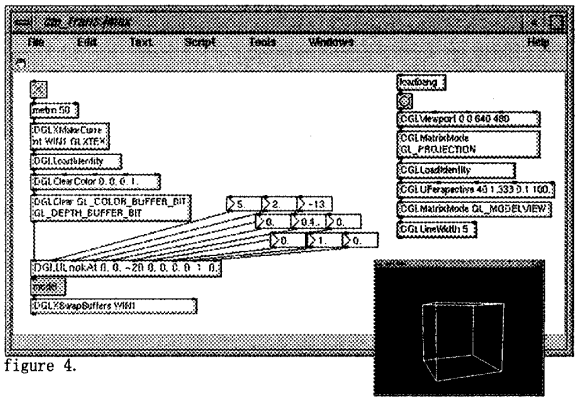
figure 4.

E. 照光処理

OpenGL で照光処理の効果を得るには，ライト使用の有効化，ライトの有効化と，ライトとモデルの照光設定，そしてモデルのレンダリングというプロセスを経る．

```
glEnable(GL_LIGHTING);
glEnable(GL_LIGHT0);
glMaterial*((GL_FRONT,GL_SPECULAR,ms_array);
glMaterial*(GL_FRONT,GL_SHININESS,50.0);
glLight*(GL_LIGHT0,GL_DIFFUSE,ld_array);
glLight*(GL_LIGHT0,GL_POSITION,lp_array);
        :
    モデルのレンダリング
        :
```

DIPS 上でこれを実装すると右図のようになる．(figure 6.) 図中右部分は初期化を行っており，左部分では"metro"からの"bang"信号により照光処理を伴ったレンダリングが繰り返されている．"DGLLight"では，値を適応させるライトの選択と光の種類の選択をアーギュメントとして設定する．5 つあるインレットのうち右の 4 つにはフロートボックスが接続されているが，これらは，それぞれ拡散光の RGBA に対応しており，一番左のインレットに"bang"信号が送られたとき，設定された値を用いて OpenGL 関数を呼び出す．これにより，リアルタイムに拡散光の値を変化させることが可能となっている．

"DGLMaterial"はモデルの材質の設定を行うオブジェクトで，モデルの適応面の設定と材質特性の選択が"DGLLight"同様に行える．

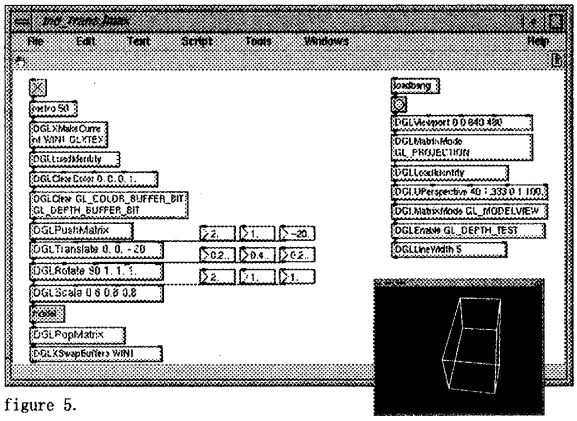
figure 5.

F. テクスチャ・マッピング

OpenGL 上でテクスチャをモデルに適応する場合は，概ね，テクスチャの定義，テクスチャの有効化，モデルのレンダリング，テクスチャの無効化の過程を経る．

```
glTexture2D(GL_TEXTURE_2D, 0, GL_RGBA,imgWidth,
      imgHeight, 0, GL_RGBA, GL_UNSIGNED_BYTE, img);
glEnable(GL_TEXTURE_2D);
        :
    モデルのレンダリング
        :
glDisable(GL_TEXTURE_2D);
```

DIPS 上でテクスチャを定義するには"DIPS_table"という特別なオブジェクトを用いる．これは，画像データのメモリテーブルであり，DIPS 上でテクスチャを用いる場合は，あらかじめ動画ファイルをこのテーブル上に読み込んでおく必要がある．扱える動画ファイルは SGI 社の Digital Media Library のサポートする形式であり，アーギュメントには，取り込む動画の絶対パス，解像度，フレーム数を指定する．その上で，"DIPS_makeCurrentTable"というオブジェクトをレンダリングの前に用いると，テクスチャとしてモデルに適応させる動画のフレーム位置を 1 フレーム単位で自由に選択することができるようになる．

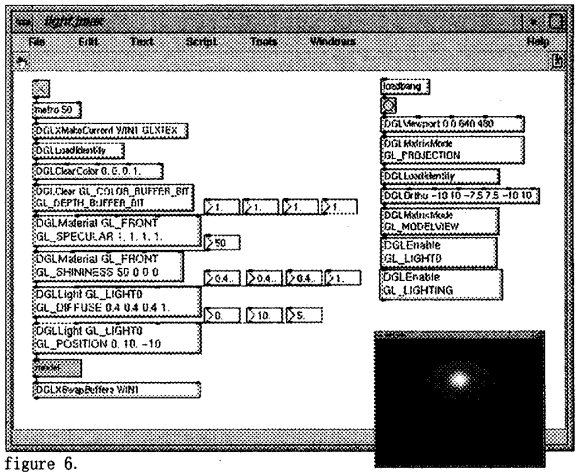
figure 6.

右図(figure 7.)のパッチでは"DIPS_table"が用いられているが，ここでは bubble.mv という動画ファイルを 256×256 の解像度で 20 フレーム読み込み，それを TEX という名前で定義している．また，実際にモデルをレンダリングする前に"DIPS_makeCurrentTable"があるが，これは TEX テーブルの 15 フレーム目の画像をモデルに適応させる設定をしている．

G. 二次元画像のラスタ

OpenGL では，三次元画像の他に，二次元の画像も扱うことができる．

メモリ上にある画像は glDrawPixels()によってフレームバッファ上にラスタされるが，その際に，ラスタ位置を glRasterPos*() によって，ピクセルの拡大・縮小を glPixelZoom*()によって決定し，実際のレンダリングを行う．(figure 8.)

前述した"DIPS_table"で取り込んだ動画は，テクスチャに適応するだけでなく，そのままフレームバッファにラスタすることもできる．図の例では，動画ファイル bubble.mv が読み込まれ，TEX という名前で定義されている．図中左では，"DGLPixelZoom"と"DGLRasterPos"によってラスタ位置とピクセルの拡大・縮小率を設定している．

後述するビデオ系エフェクトをフレームバッファにラスタする時にも，これらのオブジェクトは重要な役割を果たす．

3. DIPS エクスターナルオブジェクトの開発状況

DIPS はエクスターナルオブジェクト開発のためのライブラリを提供しており，国立音楽大学音楽デザイン学科では，これらの環境を用いて様々な DIPS エクスターナルオブジェクトの開発が進められている．

jMax は様々なオブジェクトを種類分けしてパッケージと呼ばれるライブラリを作っている．例えば，ISPW に存在したシグナルやコントロール系のオブジェクト群は"ispw"パッケージにまとめられており，数学的な計算のオブジェクトなどは"math"パッケージに集められている．一つ一つのパッケージがバイナリのライブラリとなっていて，jMax はそれらのパッケージを起動時に取り込んでユーザに提供している．全体の再コンパイルや複雑な設定ファイルの書き換えを行わなくてもパッケージの追加は可能で，DIPS 自体も 1 つのパッケージとして jMax 上に存在している．

現在開発が行われているのは，パーティクルや外部 3D モデリングソフトからのデータを扱う"D3D"パッケージ，ビデオエフェクトやビデオ画像解析などを行う"DPX"パッケージの 2 つである．

美山によって開発が行われている"D3D"パッケージは，3D モデルやパーティクルを効率的に扱うオブジェクト群である．

"D3DOBJTable"は OBJ 形式のファイルに記述された 3D モデルをメモリ上に取り込むオブジェクトである．一度メモリに取り込まれたモデルは"D3DOBJRender"を用いることにより実際にレンダリングされる．また，その過程において"D3DOBJHandler"により特定範囲のモデルの頂点座標を自由に変え，モデルを変形させることができる．

さらに，"D3DOBJParticle"により"D3DOBJTable"に確保されている 3D モデルを一つ一つの粒子としたパーティクル処理を行うことも可能となっている．

"DPX"パッケージはビデオエフェクトやビデオ画像解析などを行うオブジェクト群である．ビデオエフェクトは安藤が，ビデオ画像解析は橋田が開発を行っている．

ビデオエフェクトは，"DPXIir1"，"DPXFir3"，"DPXWaves"，"DPXRipple"等の画像フィルタで構成されている．これらのフィルタは"DIPS_table"の値を直接変化させるものであり，組み合わせて使うことも可能になっている．

画像解析は，ビットマップ画像の輝度の差分計算を行う"DPXSub"を使用することにより行う．このオブジェクトの処理結果を"DIPS_table"に格納し，"DPXHistX"，

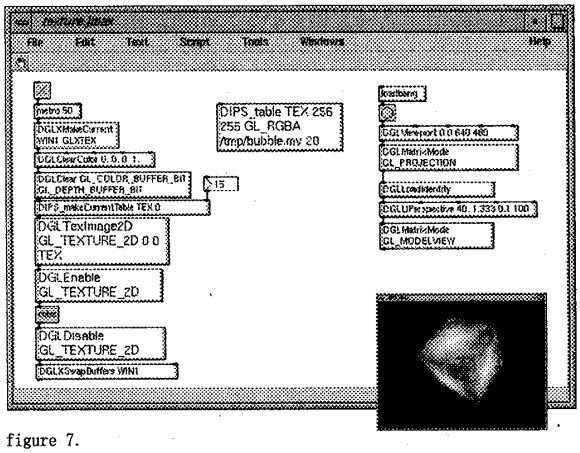

figure 7.

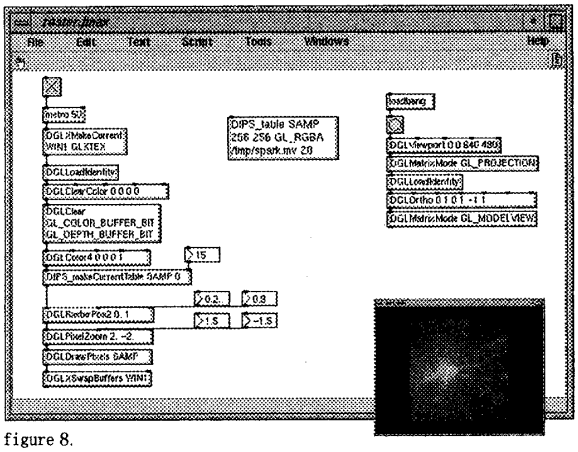

figure 8.

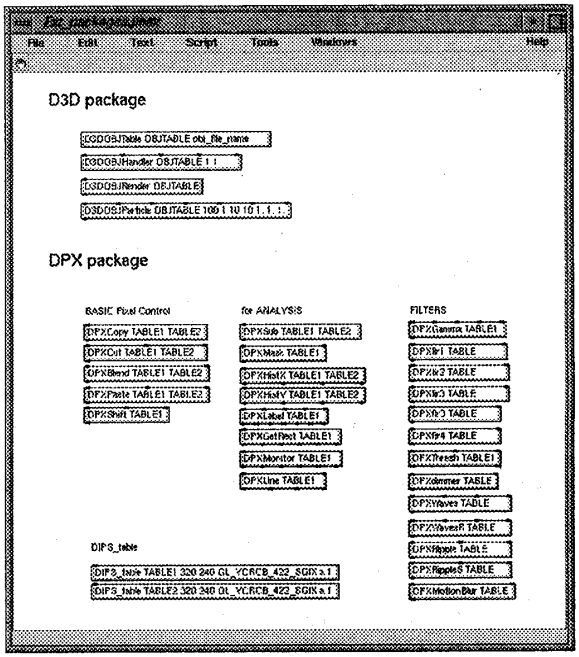

figure 9.

"DPXLine"等のオブジェクトの処理を行って必要な値を取り出し，様々な形で利用することが可能になっている．

画像解析は，橋田が以前から行っていた，ダンスパフォーマンスから音楽を導き出す研究に基づいている．(figure 9.)

4. まとめ

本稿では，DIPS 上での実際のプログラミングについての解説と，DIPS エクスターナルオブジェクトの開発状況についての報告を行った．

Max インターフェイス上でのプログラミングは，テキストでコードを記述することに慣れている作品制作者にとっては，やや習得しにくく煩雑に思えるかもしれないが，コンパイルなどの手間をかけずに試行錯誤を繰り返せる，という利点がある．

また，今回は触れなかったが Max インターフェイスを採用しているということで，音響処理との連携も容易である．これらの利点を活かして DIPS を様々な形で利用した作品も増えてきている．

今後，さらにユーザからの意見を元に開発を進めていきたい．

参考文献

[1] Mason Woo, Jackie Neider, Tom Davis「OpenGL プログラミング ガイド 第二版」, アジソン・ウェスレイ・パブリッシャーズ・ジャパン株式会社 (1997)

[2] 松田 周 「コンピューターを用いたリアルタイム音声・映像信号処理システムのための作品創作-システム構築と作品創作について-」, 国立音楽大学大学院音楽研究科修士論文 (1999)

[3] 松田 周 「DIPS: Max のためのリアルタイム映像処理オブジェクト群」, 情報処理学会研究報告, 2000-MUS-36 (2000)

[4] 橋田 光代, 美山 千香士, 安藤 大地 「DIPS エクスターナルオブジェクトの開発と作品制作への応用」, 情報処理学会研究報告, 2000-MUS-36 (2000)

[5] Shu Matsuda, Takayuki Rai「DIPS: Real-Time Digital Image Processing Objects for Max Environment」, International Computer Music Conference 2000 (2000), Berlin

[6] François Déchelle, Norbert Schnell, Riccardo Borghrsi, Nicola Orio 「The jMax Environment: An Overview of New Features」, International Computer Music Conference 2000 (2000), Berlin