ＯｐｅｎＧＬによるＣＡＶＥとファントムへの３次元表示

井門　俊治　（いど　しゅんじ）
埼玉工業大学　工学部　情報システム学科

利用環境: Windows， LINUX， IRIX， AVS， AVS-MPE， OpenGL，VRML，Java3D

１．目的

CAVE の表示するためには、従来は OpenGL によるプログラミングが行われていた。これに対して、２０００年末より、AVS において CAVE 対応が可能となった。また、ほかにも多くの CAVE への表示のツールが開発されてきた。特に３次元 CG のコンテンツも VR4MAX やほかの変換ツールにより、CAVE の中に表示できるようになった。

一方、Java3D による３次元表示も Windows、LINUX、 IRIX、などの環境において可能となってきた。このことから、VRML、AVS (MGF)、Java3D とあわせて、OpenGL による３次元グラフィックスプログラミングも、３次元グラフィックスプログラミングの教育の観点から、再度採用している。さらに、ファントム（触感型の VR 装置）においては、OpenGL およびそれと相似性のある OpenHL を用いた触感プログラミングが可能であり、OpenGL プログラムの流通性はよいことがわかっている。

本稿では、CAVE およびファントムへの表示の観点から、３次元グラフィックスプログラミングの開発とポータビリティについて述べる。

２．方法

（1）３次元グラフィックスプログラミング

OpenGL、VRML、AVS(mgf)、Java といったグラフィックスプログラムを使って、同じ３次元モデルを作成する。OpenGL、MGF で作成した３次元モデルも CAVE に出力し、３次元モデルの共有を調べる。OpenGL プログラムは、OpenHL を用いてファントム用にも移植し、プログラムのポータビリティを検証する。一般的に、それぞれのプログラム（またはスクリプト）の表示の可能性は下記のようになる。

表１　３次元グラフィックスプログラミングの表示

	OpenGL	AVS(MGF)	VRML	Java3D
Windows	○	○	○	○
LINUX	○	○	○	○
IRIX	○	○	○	○
CAVE	○	○		
PHANTOM	○			

（2）OpenGL プログラムの CAVE 用への書き換え

OpenGL で作成した３次元グラフィックプログラムを CAVE に出力する。OpenGL のグラフィックプログラムは、ヘッダーファイルを足し、メイン関数の内容を変えて出力する。下記の表２、表３で示したように、int main のところを書き換え、include 文を付け加えることにより CAVE に表示することが可能になる。

Display 表示の初期化が CAVE 表示の初期化に置き換わり、CAVE 用のプログラムに書き換えることが出来る

表２　１画面（Windows、LINUX、IRIX）用の OpenGL

```
#include<stdio.h>
#include<windows.h>
#include<GL/gl.h>
#include<GL/glu.h>
#include<GL/glut.h>
・
main(int argc, char** argv)
{
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGBA| GLUT_DEPTH);
glutInitWindowPosition(0,0);
glutInitWindowSize(800,900);
glutCreateWindow(argv[0]);
glOrtho(-15, 15, -15, 15, -15, 15);
myinit();
glutDisplayFunc(display);
glutMainLoop();
}
```

表３　CAVE 上の OpenGL

```
#include <stdio.h>
#include <GL/glu.h>
#include<GL/gl.h>
#include <GL/glut.h>
#include <cave_ogl.h>
#include <unistd.h>
・
int main(int argc, char **argv)
{
CAVESetOption(CAVE_PROJ_USEMO
DELVIEW,0);
CAVEConfigure(&argc,argv, NULL);
CAVEInit();
CAVEInitApplication(init, 0);
CAVEDisplay(display, 0);
while(!CAVEgetbutton(CAVE_ESCKEY))
{
sginap(10);
}
CAVEExit();
}
```

（２）OpenGL プログラムの PHANTOM 用への書き換え

PHANTOM は、センサブルテクノロジー社が開発した、指先と仮想オブジェクトとの相互作用が 1 点で行

われるポイント型で位置入力と力出力が可能な触感デバイスである（図１、図２）。PHANTOM を介して仮想オブジェクトに触れると形状や質感など動きに応じた触覚情報がリアルタイムにユーザ側の指先に反映される。PHANTOM を介して仮想オブジェクトに触れると形状や質感など動きに応じた触覚情報がリアルタイムにユーザ側の指先に反映される。入力は位置(x,y,z)の３自由度とポインタの回転(yaw, roll, pitch)の３自由度の計６自由度。出力は位置の３自由度とポインタの回転も含めた６自由度である。

図１ PHANTOM Omni

図２ PHANTOM Desktop

PHANTOM 用のアプリケーションを開発するためには、OpenHarptics を用いる。これは、複雑な計算を扱う PHANTOM の制御プログラミングを低レベルで扱える。全てのタイプの PHANTOM に対応しているので、異なるタイプであっても同じプログラムのまま修正することなく実行することが可能である。

比較的高レベルの処理を扱う HL(Haptic Library)API と、より低レベルな処理が可能な HD(Haptic Device)API の２つから構成されている。HLAPI では、OpenGL をサポートしているので、点、線、ポリゴン、NURBS 曲面等の基本図形を直接反力モデルとして使用でき、移動、回転、拡大・縮小、ライティング、シェーディング、テクスチャマッピング等を利用することができ、従来の OpenGL プログラムを活用できる。

このため、CAVE と PHANTOM で同一のプログラムを共有できる。両者でソケット通信を行うことにより、ネットワーク触感のメリットとして通常 PC から得られる情報よりも多くの情報を体感により得ることができる。

３．結果

①（ブドウ）

球１４個、円柱２本使用して作成したブドウの VRML を参考（図３）に、OpenGL グラフィックスプログラム（図４）を作成した。また、この OpenGL プログラムを参考に Java（図５）、MGF（図６）、を作成した。

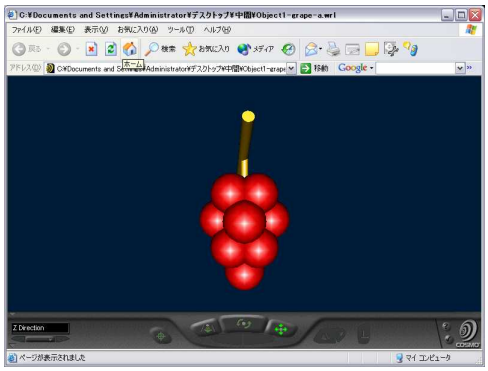


図３　ブドウ（VRML）

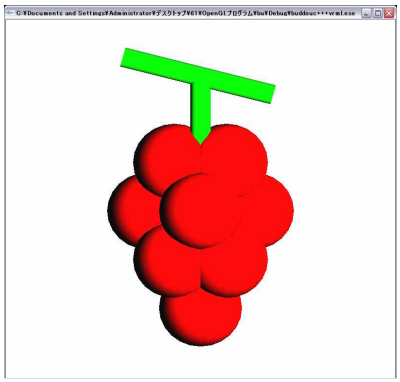


図４　ブドウ（OpenGL）

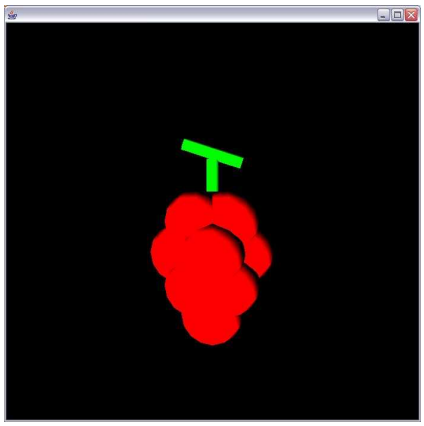


図５　ブドウ（Java）

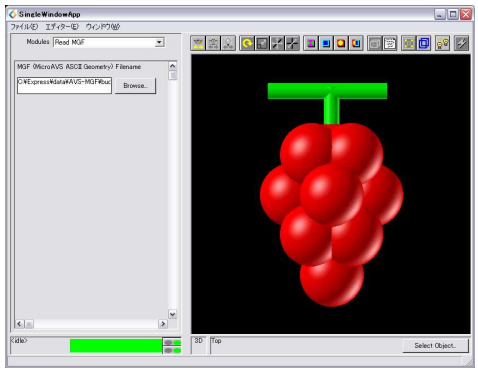


図６　ブドウ（MGF）

OpenGL（図５）、MGF（図７）をそれぞれ、CAVE に表示し比較をした（図８、図９）。

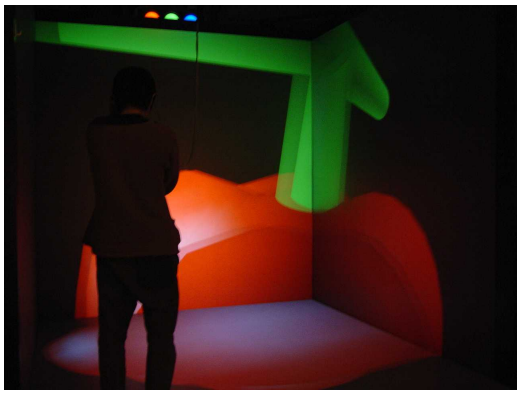


図７　ブドウ（MGF-CAVE）

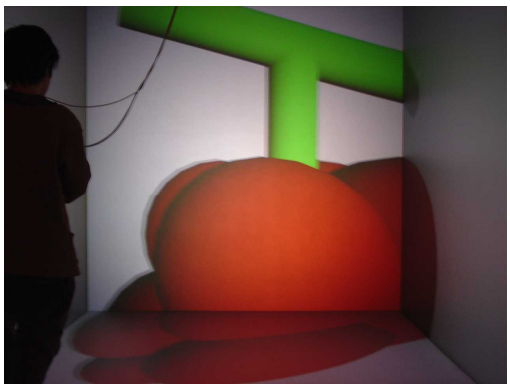


図８　ブドウ（OpenGL）

②ダイヤモンド

ポリゴンにより作成したダイヤモンドの図形を同様に、VRML（図９）、OpenGL（図１０）、Java3D（図１１）、MGF（図１２）で作成し、表示した。先と同様に CAVE にも表示した（図１３、図１４）。

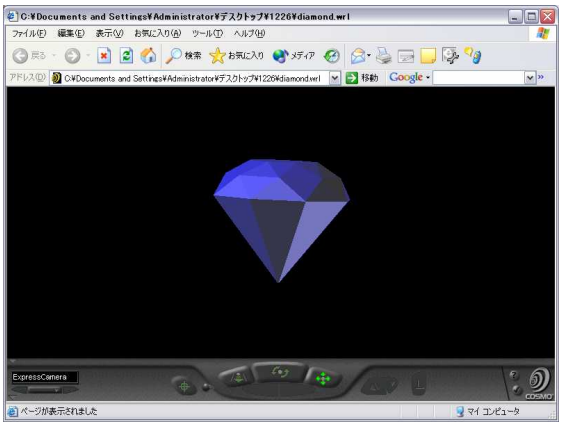


図９　ダイヤモンド（VRML）

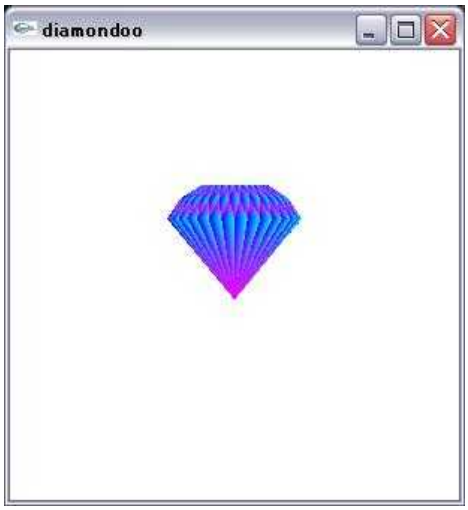


図１０　ダイヤモンド（OpenGL）

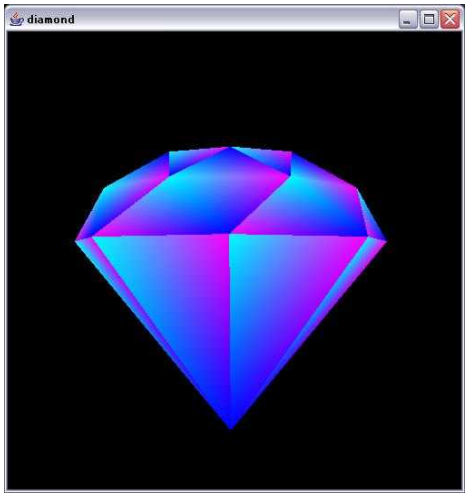


図１１　ダイヤモンド（Java）

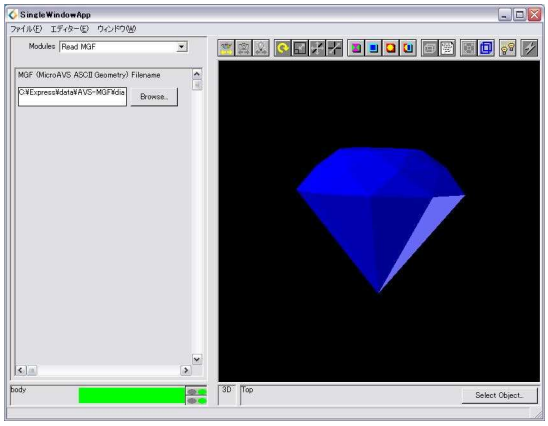


図１２　ダイヤモンド（MGF）

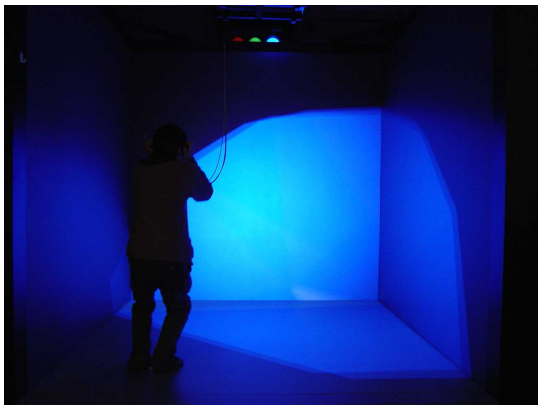


図１３　ダイヤモンド（MGF-CAVE）

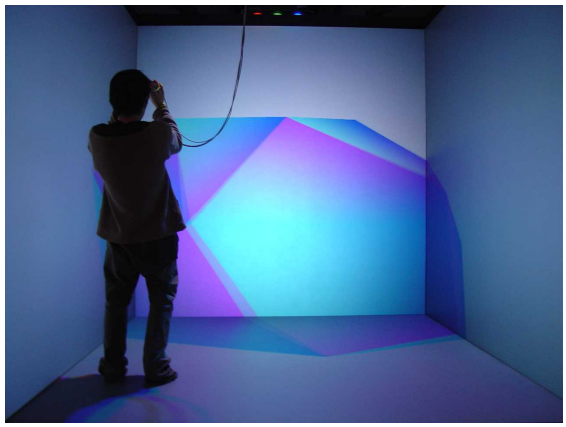


図１４　ダイヤモンド（OpenGL）

③PHANTOM Omni への OpenGL の表示

OpenGL、OpenHL を用いた触感体験用の C プログラムを開発した。実際に３次元仮想オブジェクトに触感している様子を図１５に示す。図１６は PHANTOM におけるポリゴン表示であり図１６の画面にも表示されている。さらに複雑化することで多様な形状物を作成し、触感体験を与えることができる。

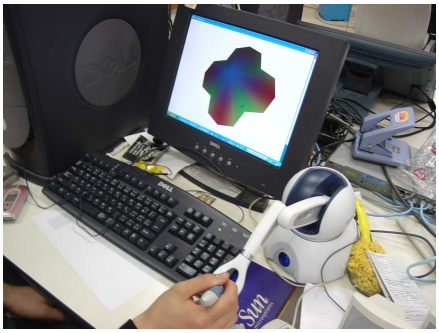


図１５　PHANTOM にて触感を得ている例

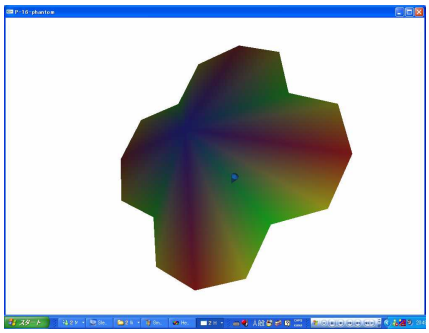


図１６　ポリゴンによる多面体

図１７、図１８は、OpenGL および Java3D にて、分子模型（C3H8）を表示した様子を示している。OpenGL プログラムを、OpenHL のルーチンを追加して、PHANTOM での触感を可能にした様子を図１９に示して

いる。OpenHL のキーファンクションを与え使用者が PHANTOM の操作によりオブジェクトの移動をできるようにしている。

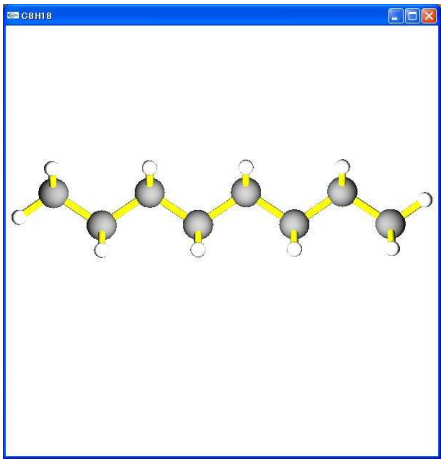


図１７　C_3H_8 分子模型（OpenGL）

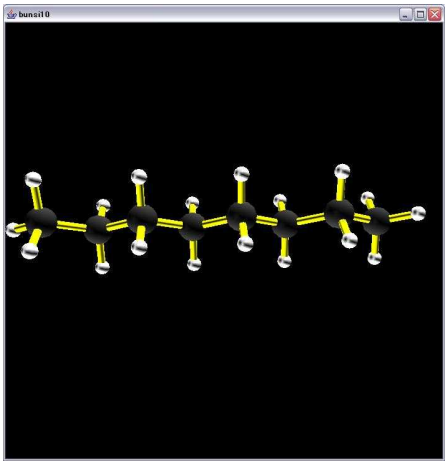


図１８　C_3H_8 分子模型　（Java3D）

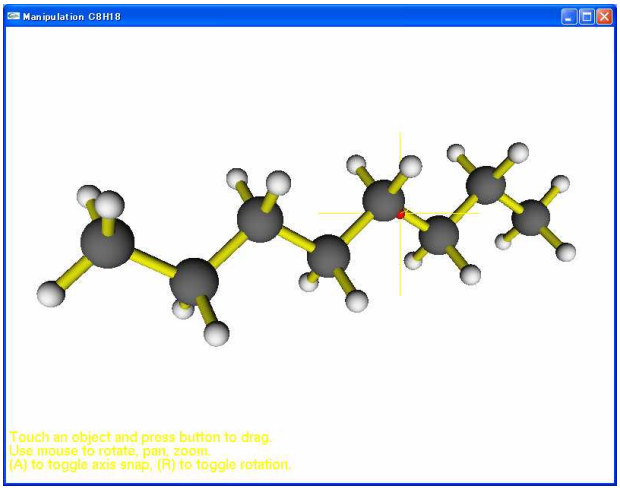
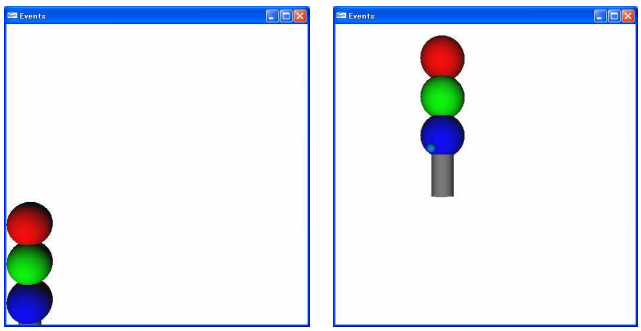


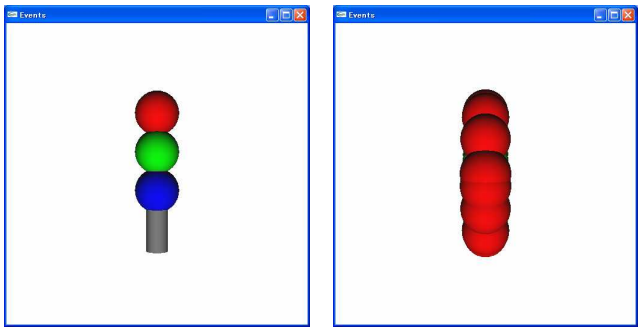
図１９　OpenHL による PHANTOM のファンクション

アニメーションに OpenHL によるキーファンクションを与え、運動のインタラクティブ性を実現したものが図２０と図２１である。図２０では X 軸方向へ移動する団子のオブジェクトに触れると Y 軸方向へ移動するものである。図２１は静止状態の団子に触れると X 軸方向へ回転するものである。両図とも(a)は触感前、(b)は触感後となっている。



(a)　　(b)

図２０　Y 軸方向へ直線運動



(a)　　(b)

図２１　X 軸方向へ回転運動

OpenHL は OpenGL との対応性があるため、OpenGL グラフィックプログラムを基にした PHANTOM のコンテンツの開発は容易であるといえる。触感で得られる情報は多く、教育面での利点は高いといえる。

４．結論

VRML，Java、mgf、OpenGL 共にオブジェクトの指定、座標の指定は、プログラムで指定できることから共通性が大きい。ポリゴンで画像の作成は、VRML、mgf、OpenGL、Java はポリゴンの頂点が同じなので、座標の共有が出来る。また CAVE へのデータ、プログラムの移植も（慣れは必要であるが）容易である。

OpenGL によるプログラムに対して、OpenHL ルーチンを追加することにより、PHANTOM への移植もプログラム共有を行いつつ行うことができた。OpenHL の OpenGL サポートにより容易な開発を行うことができた。触感により得られる情報により、理解が深まることが利点である。

参考文献

（１）　三浦憲二郎「OpenGL グラフィックス入門」朝倉書店（１９９６年）

（２）　田中成典・小林孝史「VRML の達人」森北出版株式会社　１９９９年

（３）　中野祥孝「OpenGL による３次元グラフィックスの研究」
（埼玉工業大学電子工学科２００４年度卒業論文）

（４）　中島賢氏「グラフィック環境の開発に関する研究」
（埼玉工業大学電子工学科２００４年度卒業論文）

（４）　大澤厚雄「３次元グラフィックスによる可視化の研究」
（埼玉工業大学情報工学科２００５年度卒業論文）

（５）　恩田祐樹「３次元グラフィックスプログラミングの研究」
（埼玉工業大学情報工学科２００５年度卒業論文）

（７）　冨田亮　「３次元可視化の研究」
（埼玉工業大学情報工学科２００６年度卒業論文）

（８）　一山友希「ＯｐｅｎＧＬによる PHANTOM コンテンツの開発」
（埼玉工業大学情報工学科２００６年度卒業論文）