岡本安晴　2001.2.1；2003.2

関数の立体図

２変数の関数の値が立体図あるいは等高線図として表されると視覚的に理解しやすくなる。ここでは、立体図の簡単な描き方について説明する。

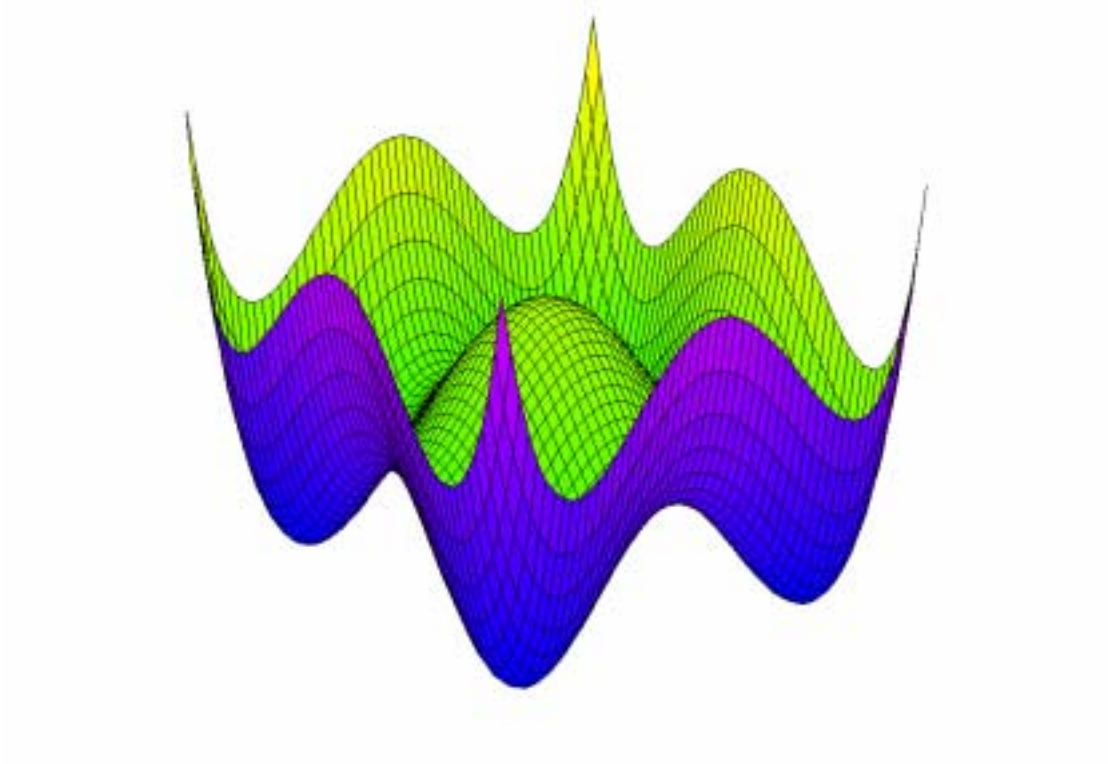

図１　曲面の立体図

図１は関数 $z = f(x,y) = (x/2)^4 + (y/2)^4 - 2((x/2)^2 + (y/2)^2)$ の立体図である。この図は、関数の値を独立変数の変域の格子点で求め、隣り合う格子点で囲まれる小矩形上の関数の局面を四角のタイルとして奥の方から 1 枚ずつ描いたものである。関数の局面の位置する空間の描画画面上での位置は透視図により以下のように求めている。

透視図の画面を図２のように設定する。ここでの説明は森（1974、第 4.2 節）(1) に従っているが、直方体と透視図の関係を統一する、角度はすべて正の値で表わす、などの変更を行っている。

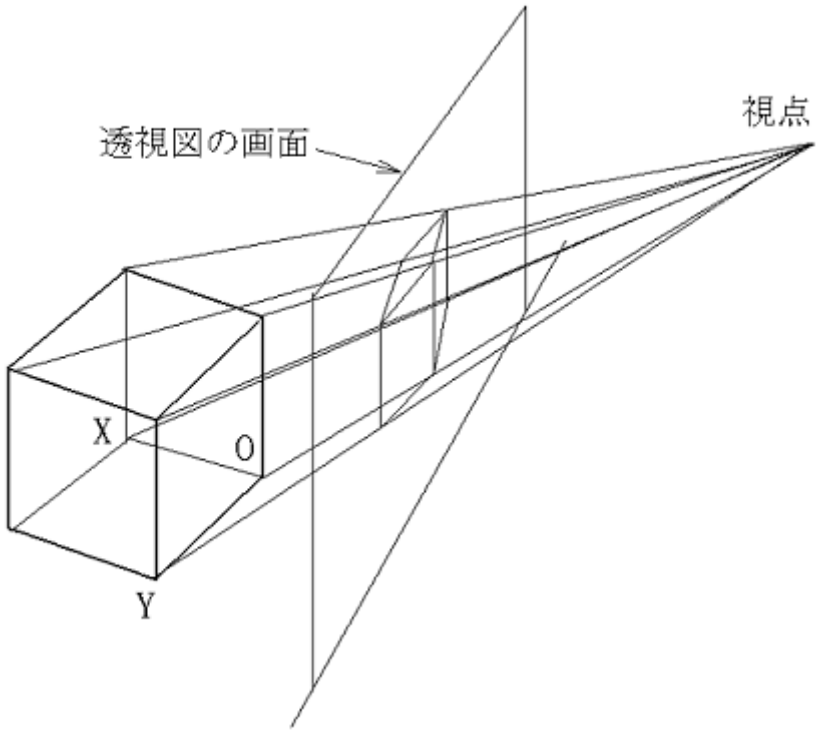

図２　立体図を透視図で描く

X 軸と Y 軸を直方体の辺 OX と OY にとり、関数の曲面が直方体内に納まるように設定して透視図の画面に立体図を描く。２変数 x と y の変域が $[a,b]$ と $[c,d]$ のとき、一次式による変数変換によって、それぞれの変域が $[0,X_w]$ と $[0,Y_w]$ であるようにできる。また、値域が $[u,v]$ のきは、一次式による変換によって値域が $[0,Z_h]$ であるようにできる。これらの変換によって、変域が長方形である関数の曲面は、大きさが $X_w \times Y_w \times Z_h$ の直方体に納めることができる（図３）。

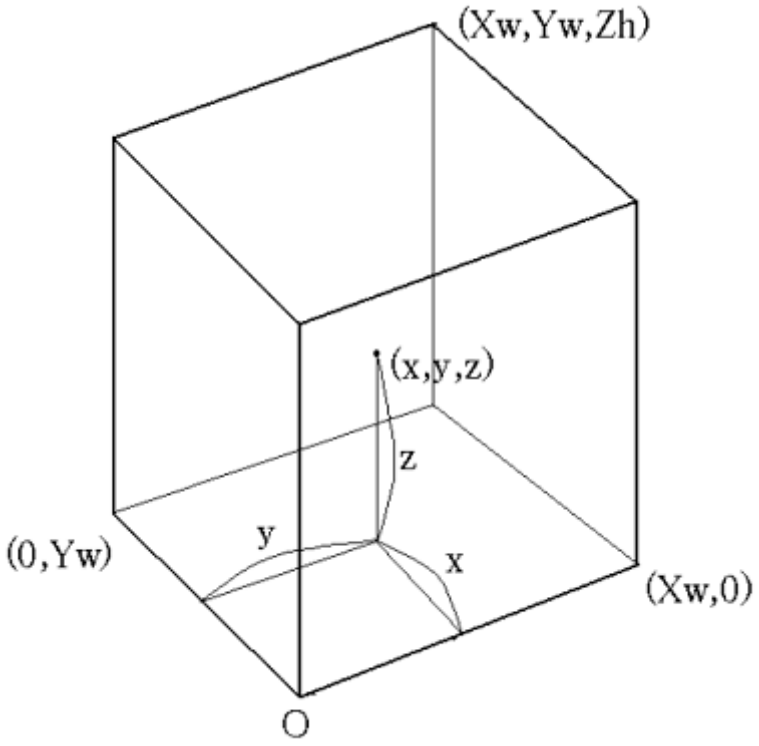

図３ 関数を納める直方体

直方体の内部の点(x, y, z)が、透視図の画面にどのように投影されるか調べる。
直方体を透視画面に対して図４のように傾ける。

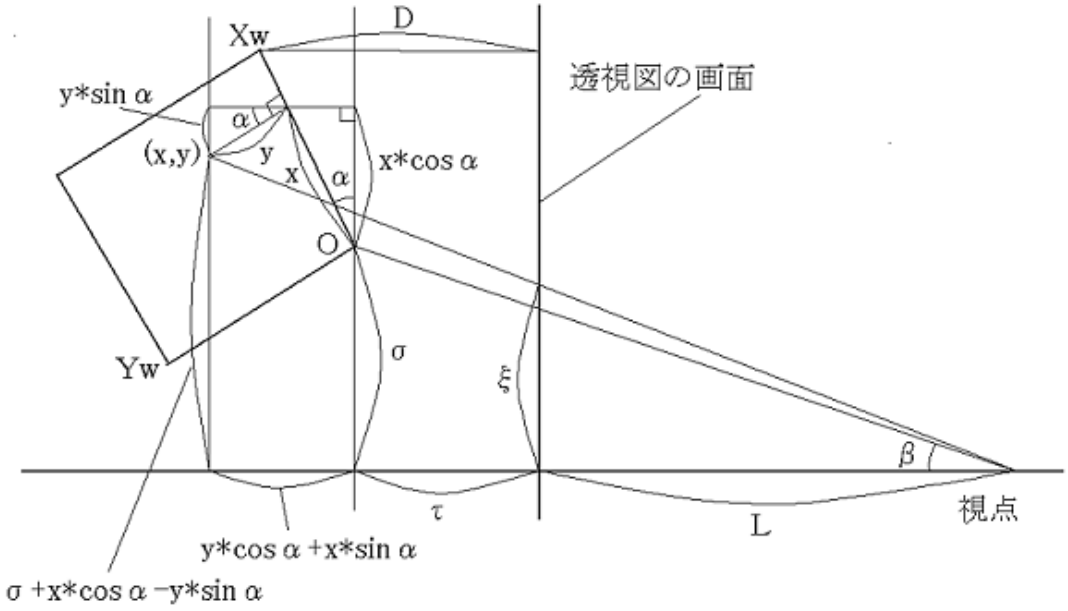

図４ 透視図の画面と点(x, y)の位置

図４は、真上（Z 軸方向）から見た図として、ＸＹ平面上の２次元で直方体と透視画面の関係を表わすものである。直方体の一辺 OX と透視画面との角度がαであるように直方体を置く。点(x, y, z)は XY 平面状では点(x, y)として表わされている。

点(x, y)の透視図の画面上における水平方向の位置は、視点と点(x, y, z)を結ぶ線が透視図の画面と交わる点として、視点から画面へ下ろした垂線の足からの距離ξで表わされている。ξは、以下のように求めることができる。

視点から透視図の画面へ下ろした垂線と点 O との距離をσ、垂線の長さを L、点 O と透視図の画面との距離をτ、点$(X_w,0)$と透視図の画面との距離を D とおく（図４）。直方体は透視図の画面に対してαだけ半時計周りの方向に回転しているものとし、視点から透視図の画面に下ろした垂線と視点と点 O を結ぶ直線のなす角をβとおく。このとき、図４において、視点、点(x, y)、および視点から透視図の画面へ下ろした垂線へ点(x, y)から下ろした垂線の足の３点で作られる直角三角形に注目して次式を得る。

$$\frac{\xi}{\sigma + x\cos\alpha - y\sin\alpha} = \frac{L}{y\cos\alpha + x\sin\alpha + \tau + L}$$

また、

$$\tau = D - X_w \sin\alpha \qquad (1)$$
$$\sigma = (\tau + L)\tan\beta$$

が成り立つ。

したがって、

$$\xi = \frac{\sigma + x\cos\alpha - y\sin\alpha}{y\cos\alpha + x\sin\alpha + \tau + L} \cdot L$$
$$= \frac{x\cos\alpha - y\sin\alpha + (D - X_w \sin\alpha + L)\tan\beta}{y\cos\alpha + x\sin\alpha + D - X_w \sin\alpha + L} \cdot L \qquad (2)$$

となる。

次に、透視図の画面上の垂直方向について考える。

図５は、直方体を真横、透視図の画面と平行な方向から見たものである。視点と点(x, y, z)を結ぶ直線が透視図の画面と交わる点の高さを視点の高さから下方向への距離ηで表わす。ηの符号は、視線が下方向を向いているので負とする。

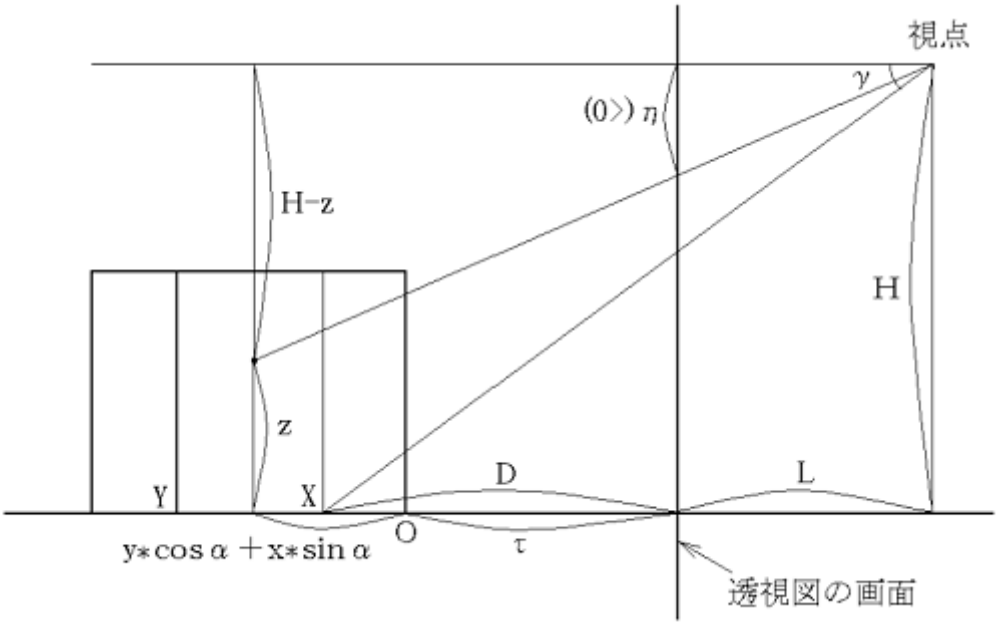

図５ 透視図の画面上での高さ

図５において、直方体の点$(X_w,0,0)$を視線が見下ろす角度をγとし、視点の直方体の置かれている平面からの高さを H とする。このとき、点(x,y,z)と視点、および視点から水平に引いた直線と点(x,y,z)を通る垂直線との交点の３点で作られる直角三角形（真横から見た様子を表わす図５におけるもの）に注目して次式を得る。

$$\frac{-\eta}{H-z}=\frac{L}{y\cos\alpha+x\sin\alpha+\tau+L} \qquad (3)$$

また、視点、視点から水平面に下ろした垂線の足、および視点から角度γで見下ろす直線と水平面との交点の３点で作られる直角三角形に注目して次式を得る。

$$\tan\gamma=\frac{H}{D+L} \qquad (4)$$

式（１）、（３）、（４）より次式が導かれる。

$$\eta=\frac{z-(D+L)\tan\gamma}{y\cos\alpha+x\sin\alpha+D-X_w\sin\alpha+L}\cdot L \qquad (5)$$

曲面上の点(x,y,z)の透視図の画面上の位置は、式（２）および（５）で与えられるにξおよびηによって点(ξ,η)と表わされる。

立体図描画用ユニット UMyPerspec1

関数の曲面の立体図を図１のように描くためのユニットが UMyPerspec03 である。このユニットのフォームは、図６のように用意されている。

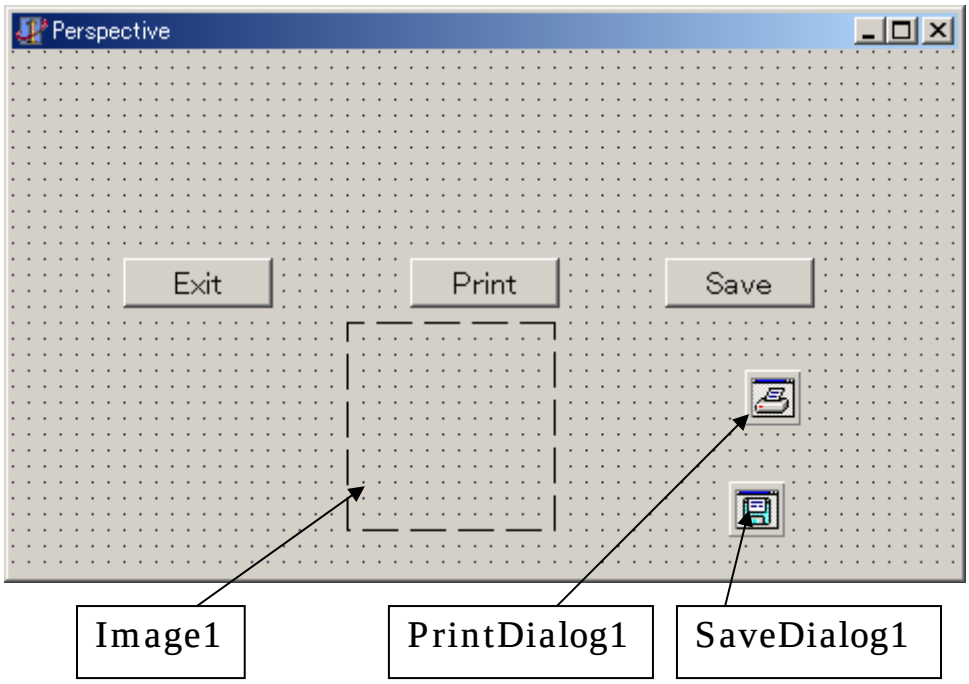

図６ ユニット UMyPerspec1 のフォーム

UMyPerspec03 において宣言されている手続き DrawPerspec を、関数と変域を指定して呼出すと（リスト１、２）、描画用のフォームが生成、表示される。

リスト１　関数の宣言

```
function CheckF( x, y : Extended ) : Extended; //  立体図を描く曲面の関数
begin
    CheckF:=sqr(sqr(0.5*x))+sqr(sqr(0.5*y))-2*(sqr(0.5*x)+sqr(0.5*y));
end;
```

リスト２　手続き DrawPerspec の呼び出し

```
DrawPerspec( CheckF, -3.0, 3.0, -3.0, 3.0 );
```

手続き DrawPerspec の始めの部分はリスト３のようになっている。

リスト３ 手続き DrawPerspec の宣言の始めの部分

```
procedure DrawPerspec( CheckF : TFunc;  //  立体図を描く関数
                       pxmin, pxmax,    //  ｘの変域（最小値と最大値）
                       pymin, pymax     //  ｙの変域（最小値と最大値）
                              : Extended );
var  i, j : Longint;
begin
    //   変数ｘとｙの変域の設定
    MinX:=pxmin; MaxX:=pxmax;
    MinY:=pymin; MaxY:=pymax;

    FPerspec:=TFPerspec.Create(Application); //  フォームの生成

    with FPerspec do
      begin
        //   フォームの準備
        Visible:=True;
        UpDate;
        WindowState:=wsMaximized;
        UpDate;

        with ExitButton do
          begin  Top:=0; Left:=0;  end;
        with SButton do
          begin  Top:=0; Left:=ExitButton.Width;  end;
        with PButton do
          begin  Top:=0; Left:=ExitButton.Width+SButton.Width; end;
        UpDate;
```

この描画用のフォームは最大化されて、その左上に「Exit」、「Save」、「Print」の３つのボタンが表示される（図７）。

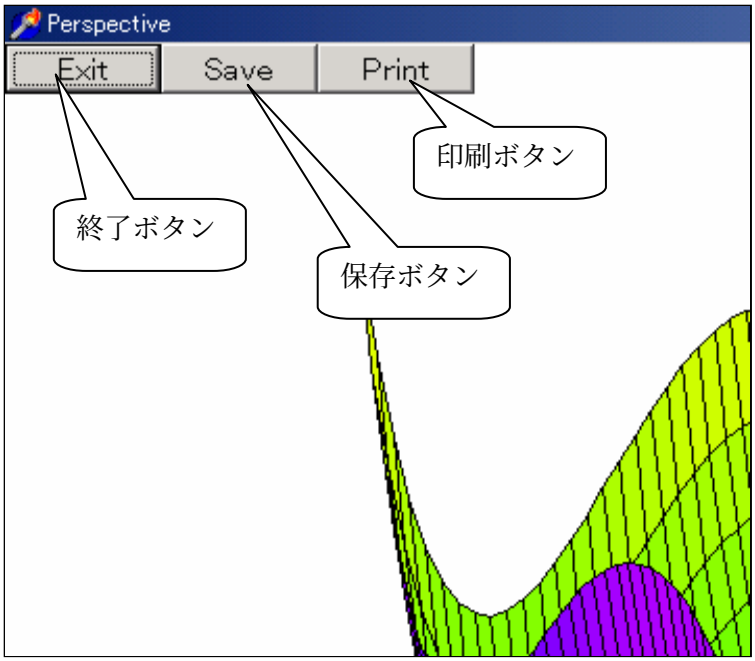

図７　描画終了時の画面

Exit ボタンをクリックするとプログラムの実行が終了する。

Save ボタンをクリックすると、描画図形が*.bmp ファイルとして保存される。描画図形は、Image コンポーネントの Canvas に描かれてから、Picture プロパティの SaveToFile メソッドによって保存される。保存先のファイル名の設定は、SaveDialog1 コンポーネントによって行っている。保存したファイルは、お絵描きソフトで読み込んだり、ワープロの文書に貼り付けたりして利用できる。

Print ボタンをクリックすると、立体図がプリンタに出力される。立体図のプリンタへの出力は、Save ボタンのクリックで保存した*.bmp ファイルをワープロの文書に貼り付けて出力することもできるが、この Print ボタンのクリックによる方が、プリンタの Canvas の解像度で印刷されるのできれいな出力が得られる。

プリンターへの出力は、Printer オブジェクトの BeginDoc メソッドの実行後、Printer の Canvas への描画によって行う。描画終了後、EndDoc を実行する。Printer オブジェクトを利用するときは、Printers ユニットを uses 節に宣言しておく。

ユニット UMyPerspec03 の使用例であるユニットファイル UPerpec03.pas（リスト３）のフォームは、図８のように用意されている。

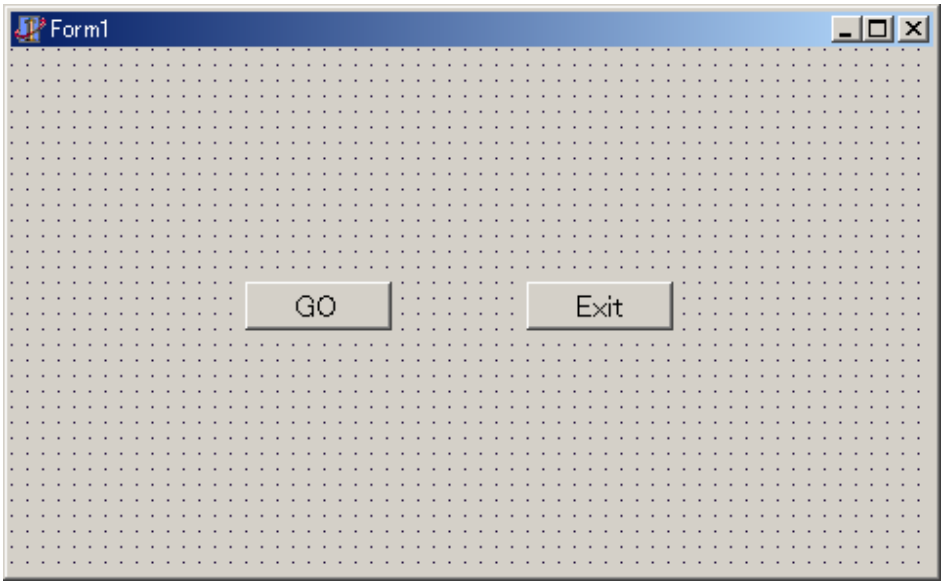

図８　リスト２のユニット UPerspec03 のフォーム

GO ボタンのクリックで手続き GOButtonClick が実行され、手続き DrawPerspec によって関数 CheckF の立体図（図１）が描かれる。

リスト３　ユニット UPerspec03.pas

```
unit UPerspec03;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls;

type
  TForm1 = class(TForm)
    GOButton: TButton;
    ExitButton: TButton;
    procedure ExitButtonClick(Sender: TObject);
    procedure GOButtonClick(Sender: TObject);
  private
    { Private 宣言 }
  public
    { Public 宣言 }
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}
```

```
uses
  UMyPerspec03;       //  立体図を描くためのユニット

procedure TForm1.ExitButtonClick(Sender: TObject);
begin
            Close;
end;

function CheckF( x, y : Extended ) : Extended; //  立体図を描く曲面の関数
  begin
        CheckF:=sqr(sqr(0.5*x))+sqr(sqr(0.5*y))-2*(sqr(0.5*x)+sqr(0.5*y));
  end;

procedure TForm1.GOButtonClick(Sender: TObject);
begin
         ExitButton.SetFocus;  UpDate;

         DrawPerspec( CheckF, -3.0, 3.0, -3.0, 3.0 );
end;

end.
```

参 考 文 献

（1）森 正武　「曲線と曲面」 Pp.150、 教育出版株式会社、1974.

（2）岡本安晴 「Delphi プログラミング入門」 Pp.207、CQ 出版株式会社、1997.