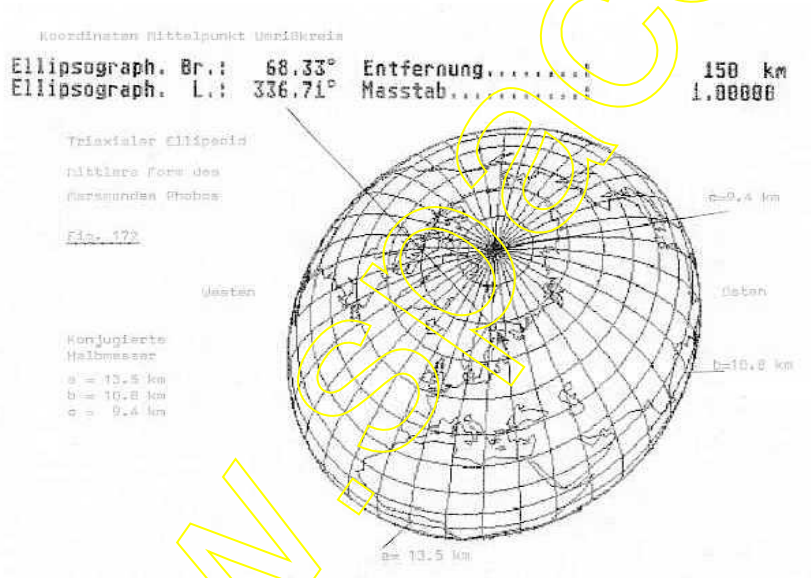
24.4 **Triaxialer Ellipsoid**

T.C. Duxbury's Darstellungsmethode eines mittleren triaxialen Ellipsoids (Duxbury-Hüllkörper); ist dem mittleren Oberflächenniveau der Marsmonde angeglichen.
Zu Gradnetzen unsymmetrischer Körper (Asteroiden, Monde) vgl. >The Shape of Gaspra<, Icarus, 107, 23-36 (1994).

Radius der bestangeglichenen (best-fit) Hauptachsen (km) der Ellipsoide: Marsmonde Phobos a,b,c=13.5x10.8x9.4 km, Deimos a,b,c=7.5x6.1x5.5 km;;
Jupitermond Amalthea a,b,c=135x83x75 km; Asteroiden: 243 Ida a,b,c=30.0x12.6x9.3 km; 951 Gaspra a,b,c=9.1x5.2x4.7 km.

Eine einfache explizite Formel zur Herleitung der Umrissebene triaxialer Ellipsoide ist selbst in Fachkreisen kaum bekannt. Zur Umrissdarstellung eines triaxialen Ellipsoids dienen meistens sehr umfangreiche Konstruktions- und Projektionsberechnungen. Der Entwurf des Verfassers beruht lediglich auf Kenntnis der planetograph. Breite (bmo) und Länge (zmo) des Mittelpunktes. Ist diese bekannt, ergeben sich die Längen (u) und Breiten (v) der Umrisspunkte durch die einfache explizite Formel: u=ACOS(-sin(v)*sin(bmo)/(cos(v)*cos(bmo)); 0<v<PI*2.
Statt die Länge (u) aus der Breite (v) zu berechnen, findet man umgekehrt die Breite aus der Länge durch: v=ATN(-(1/TAN(bmo))*COS(zmo-u)).
Die simple trigonometrische Formel berechnet die Breiten und Längen aller Orte im wahren Ortshorizont des Umrissmittelpunktes (bmo,zmo). Fig. 172.



```
//*************************************************************************
//                 Project Name:  Triaxial ellipsoids
//                       Author:  Horst Schumacher
//                  Description:  This program shows the best outlines
//                                of a three-axis ellipsoid
//
//*************************************************************************
REM PROGR in GFA-BASIC 16 BIT für Win 9x/ME/2000 (Trial-Version GFA-Basic für Windows http://www.gfasoft.gfa.net/de/produkte.htm)
OPENW #1,200,200,640,400,1
DEFFN r(x) = x - INT(x / (PI * 2)) * (PI * 2)
REM PROGR. VEKTORGRAFIK TRIAXIALER ELLIPSOID ----------
//SETCOLOR 0,0
xo = 320
yo = 200
hgx = (180 / PI) * 37.7952 //1/360 GRAD UMFANG = 1 cm = 37.7952 Pixel bei Windows 96 dpi
LINE xo - 5,yo,xo + 5,yo  //MITTELPUNKTMARKIERUNG UMRISSKREIS
LINE xo,yo - 5,xo,yo + 5
REM ----------
aee = 150  //EINTRAG ENTFERNUNG (KILOMETER)
r = (180 / PI) * (1 / aee) * 37.7952  //1 cm = 37.7952 Pixel = 1 GRAD in 1 RAD 57.29578 cm Abstand bei Windows 96 dpi
ska = 1  //EINTERAG MASSTAB
a = 13.5 //EINTRAG RADIUS A-ACHSE (KILOMETER)
b = 10.8  //EINTRAG RADIUS B-ACHSE
c = 9.4  //EINTRAG RADIUS C-ACHSE
xw = RAD(61)  //EINTRAG NEIGUNGSWINKEL  DREHUNG UM X-ACHSE
yw = RAD(30)  //EINTRAG POSITIONSWINKEL DREHUNG UM Y-ACHSE (ÄQUATOREBENE ELLIPSOID)
zw = RAD(341)  //EINTRAG LÄNGE DES ZENTRALMERIDIANS DREHUNG UM Z-ACHSE
REM ------------------------------
zw = FN r(PI / 2 - zw)
xw = PI / 2 - xw
ar = r * a
br = r * b
cr = r * c
ar1 = a
br1 = b
cr1 = c
REM ROTATIONSMATRIX -------
a11 = COS(yw) * COS(zw)
a12 = -COS(yw) * SIN(zw)
a13 = SIN(yw)
a21 = SIN(xw) * SIN(yw) * COS(zw) + COS(xw) * SIN(zw)
a22 = -SIN(xw) * SIN(yw) * SIN(zw) + COS(xw) * COS(zw)
a23 = -SIN(xw) * COS(yw)
a31 = -COS(xw) * SIN(yw) * COS(zw) + SIN(xw) * SIN(zw)
a32 = COS(xw) * SIN(yw) * SIN(zw) + SIN(xw) * COS(zw)
a33 = COS(xw) * COS(yw)
GOSUB enn  //ITERATION DER ELLIPSOGRAPH. KOORDINATEN DES UMRISSMITTELPUKTES
PRINT USING "Ellipsograph. Br.: ####.## Grad",DEG(bmo)
PRINT USING "Ellipsograph.  L.: ####.## Grad",DEG(zmo)
PRINT USING "Entfernung.....: ########## km",aee
PRINT USING "Masstab.....: ########.#####",ska
dr = 1
GOSUB umriss
RESTORE kont
FOR j% = 1 TO 1
  READ j2,o1,b,l //b,l=Breite,Länge
  b = RAD(b)
  l = RAD(360 - l) //MARSMONDE ELLIPSOGRAPH. LÄNGE NACH WESTEN 0...360 GRAD
  GOSUB en
  IF j% = j2 OR zo < 0.05 AND zo > -0.05 THEN
    j2 = 0
    PLOT xo + x * ska,yo + y * ska
  ENDIF
```

```
  IF zo > 0 THEN
    DRAW  TO xo + x * ska,yo + y * ska
  ENDIF
  FOR i% = 2 TO o1
    READ b,l
    b = RAD(b)
    l = RAD(360 - l)  //MARSMONDE ELLIPSOGRAPH. LÄNGE NACH WESTEN 0...360 GRAD
    GOSUB en
    IF j% = j2 OR zo < 0.05 AND zo > -0.05 THEN
      j2 = 0
      PLOT xo + x * ska,yo + y * ska
    ENDIF
    IF zo > 0
      DRAW  TO xo + x * ska,yo + y * ska
    ENDIF
  NEXT i%
NEXT j%
dr = 0
GOSUB netz
CLOSEW #1
KEYGET HALT%  //PRESS KEY ESCAPE
REM SUBROUTINE
PROCEDURE netz
  n = 1
  FOR l = 0 TO PI * 2 STEP RAD(10) //EINTRAG 1O,15,30,45,90 GRAD
    n = 1
    FOR b = PI / 2 TO -PI / 2 STEP -0.1
      GOSUB en
    NEXT b
  NEXT l
  n = 1
  FOR b = PI / 2 TO -PI / 2 STEP -RAD(10) //EINTRAG 10,15,45,90 GRAD
    n = 1
    FOR l = 0 TO PI * 2 + 0.1 STEP 0.1
      GOSUB en
    NEXT l
  NEXT b
RETURN
PROCEDURE en
  x = ar * COS(b) * COS(l)
  y = br * COS(b) * SIN(l)
  z = cr * SIN(b)
  x3 = a11 * x + a12 * y + a13 * z
  y3 = a21 * x + a22 * y + a23 * z
  z3 = a31 * x + a32 * y + a33 * z
  REM ZENTRALPERSPEKTIVE
  t = z3 / (z3 - hgx)
  x = x3 - t * x3
  y = y3 - t * y3
  zo = ASIN(SIN(b) * SIN(bmo) + COS(b) * COS(bmo) * COS(l - zmo))
  REM DRAWING
  IF dr = 0 THEN
    IF n = 1 THEN
      n = 0
      PLOT xo + x * ska,yo + y * ska
    ENDIF
    IF zo < 0.05 AND zo > -0.05 THEN
      PLOT xo + x * ska,yo + y * ska
    ENDIF
    IF zo > 0
      DRAW  TO xo + x * ska,yo + y * ska
    ELSE
      REM PLOT xo+x*ska,yo+y*ska
    ENDIF
```

```
  ENDIF
RETURN
PROCEDURE umriss
  nr = 0.005
  n = 1
  DO
    ADD b1,nr
    io = (-SIN(b1) * SIN(bmo)) / (COS(b1) * COS(bmo))
    IF ABS(io) > 1 THEN
      io = 0.99999999 * SGN(io)
      GOTO is
    ENDIF
    io = ACOS(io)
    l = FN r(io + zmo)
    b = b1
    GOSUB en
    IF n = 1 THEN
      n = 0
      PLOT xo + x * ska,yo + y * ska
    ENDIF
    DRAW  TO xo + x * ska,yo + y * ska
    EXIT IF ABS(b1) > PI * 2
    is:
  LOOP
RETURN
PROCEDURE enn //ITERATION DER SPHÄRISCHEN MITTELPUNKTKOORDINATEN
  zz = PI / 2
  REPEAT
    xx = 0 //2-D KOORDINATE MITTELPUNKT x=0,y=0
    yy = 0
    zz1 = zz //ITERATION DER Z-KOORDINATE
    x1 = (a11 * xx + a21 * yy + a31 * zz) / ar1 //ROTATIONSMATRIX INVERS
    y1 = (a12 * xx + a22 * yy + a32 * zz) / br1
    z1 = (a13 * xx + a23 * yy + a33 * zz) / cr1
    o = SQR(x1 ^ 2 + y1 ^ 2 + z1 ^ 2)
    bmo = ASIN(z1 / o)
    x2 = x1 / (o * COS(bmo))
    y2 = y1 / (o * COS(bmo))
    IF x2 == -1 THEN
      zmo = PI
    ELSE
      zmo = FN r(ATN(y2 / (1 + x2)) * 2)
    ENDIF
    x = ar1 * COS(bmo) * COS(zmo)
    y = br1 * COS(bmo) * SIN(zmo)
    z = cr1 * SIN(bmo)
    xx = a11 * x + a12 * y + a13 * z
    yy = a21 * x + a22 * y + a23 * z
    zz = a31 * x + a32 * y + a33 * z
    zp = ABS(zz1 - zz)
    //    PRINT zp
  UNTIL ABS(zz1 - zz) < 0.00001
RETURN
kont: //DATA WORLD COASTLINE UTILITIES s. S. 66 - ODER PHOBOS/DEIMOS KARTOGRAPHIE
DATA 1,15,31,32,16,39,12,43,10,44,12,51,5,48,-3,40,-6,39,-10,41,-15,41,-20,35,-24,36,-25,33,-29,33,-34,26,-34,19...
```

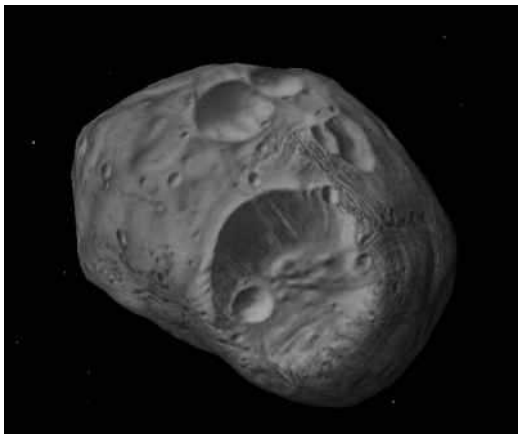
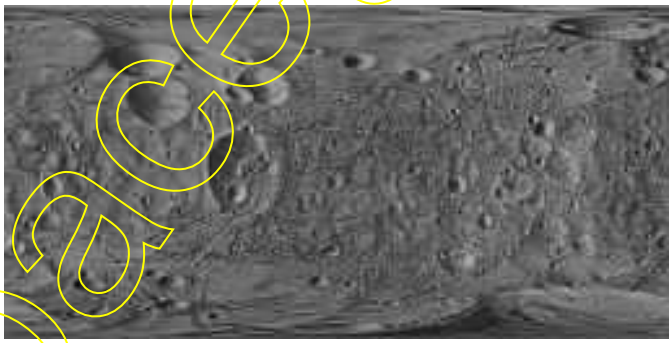


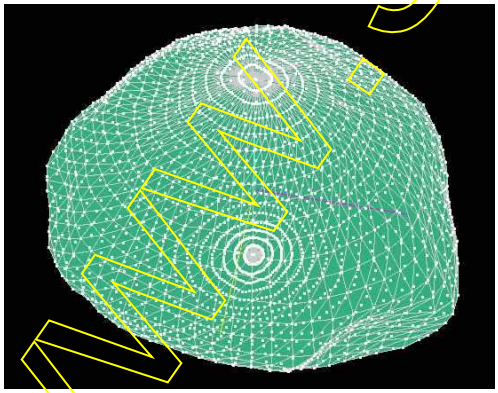
Abb. Non-Spherical World. OpenGL gerenderte morphographische Projektion des Marsmondes Phobos in Pseudo-3D.

Subsampled from the shape model of P. Thomas and colleagues at Cornell.
Morphographic coordinates of Phobos:

long, lat, radius (km)
0,-90,8.3726
0,-85,8.37995
0,-80,8.1935
0,-75,8.3111
0,-70,8.5178
0,-65,8.7549
0,-60,8.9935
0,-55,9.6682
0,-50,10.3925
0,-45,11.273
0,-40,12.0133
0,-35,12.42555
0,-30,12.7765
0,-25,12.90795
0,-20,12.966
0,-15,12.94865...

Phobos-Texture (cylindrical projection)

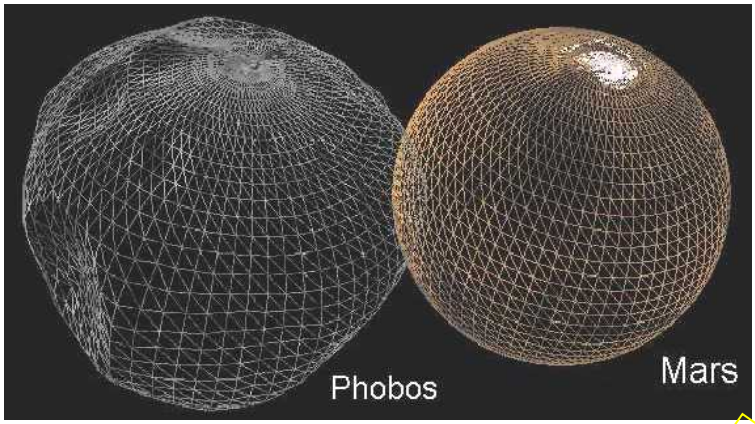




Wireframe shape model (morphographic coordinates).

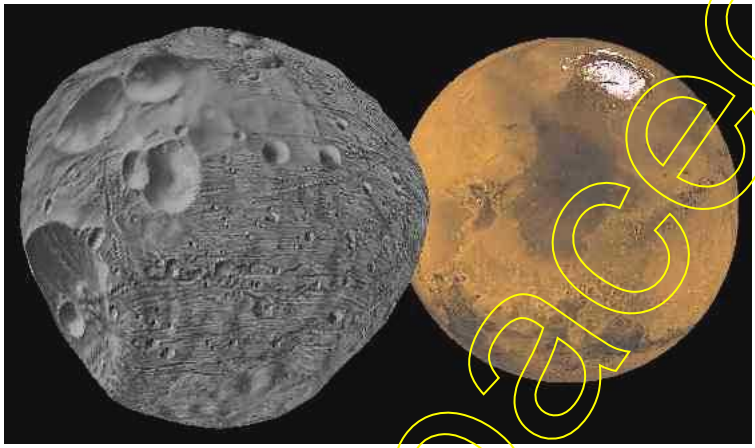
Tessellation und Morphographic mapping.

Wireframe



glPolygonMode(GL_FRONT_AND_BACK,GL_LINE)

Solid



glPolygonMode(GL_FRONT_AND_BACK,GL_FILL)

Download Borland C++ Builder 5 Sourcecode:http://www.spaceglobe.de/demo/SpaceglobePhobos.exe