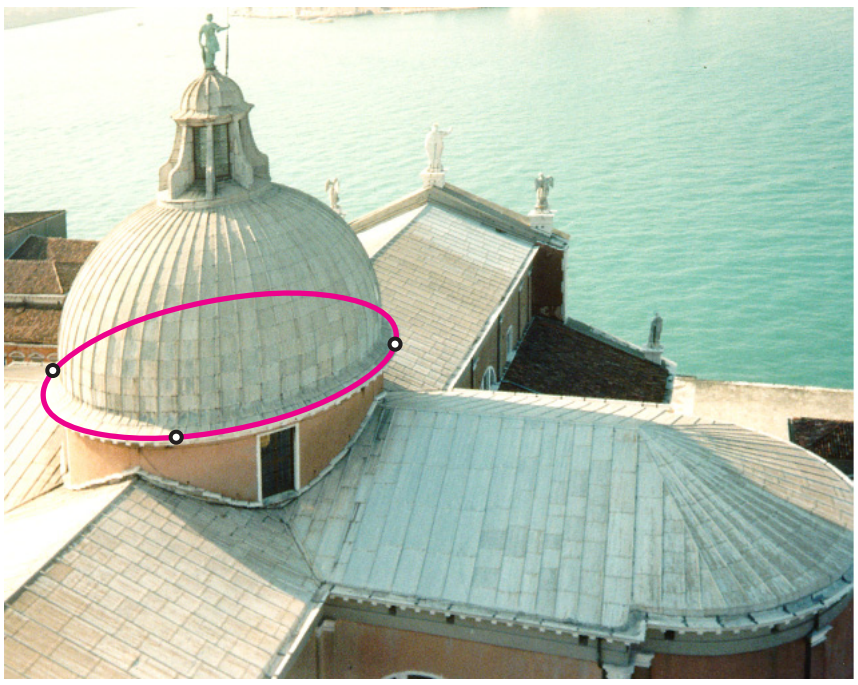
Gernot Hoffmann

Circle in 3D



Contents

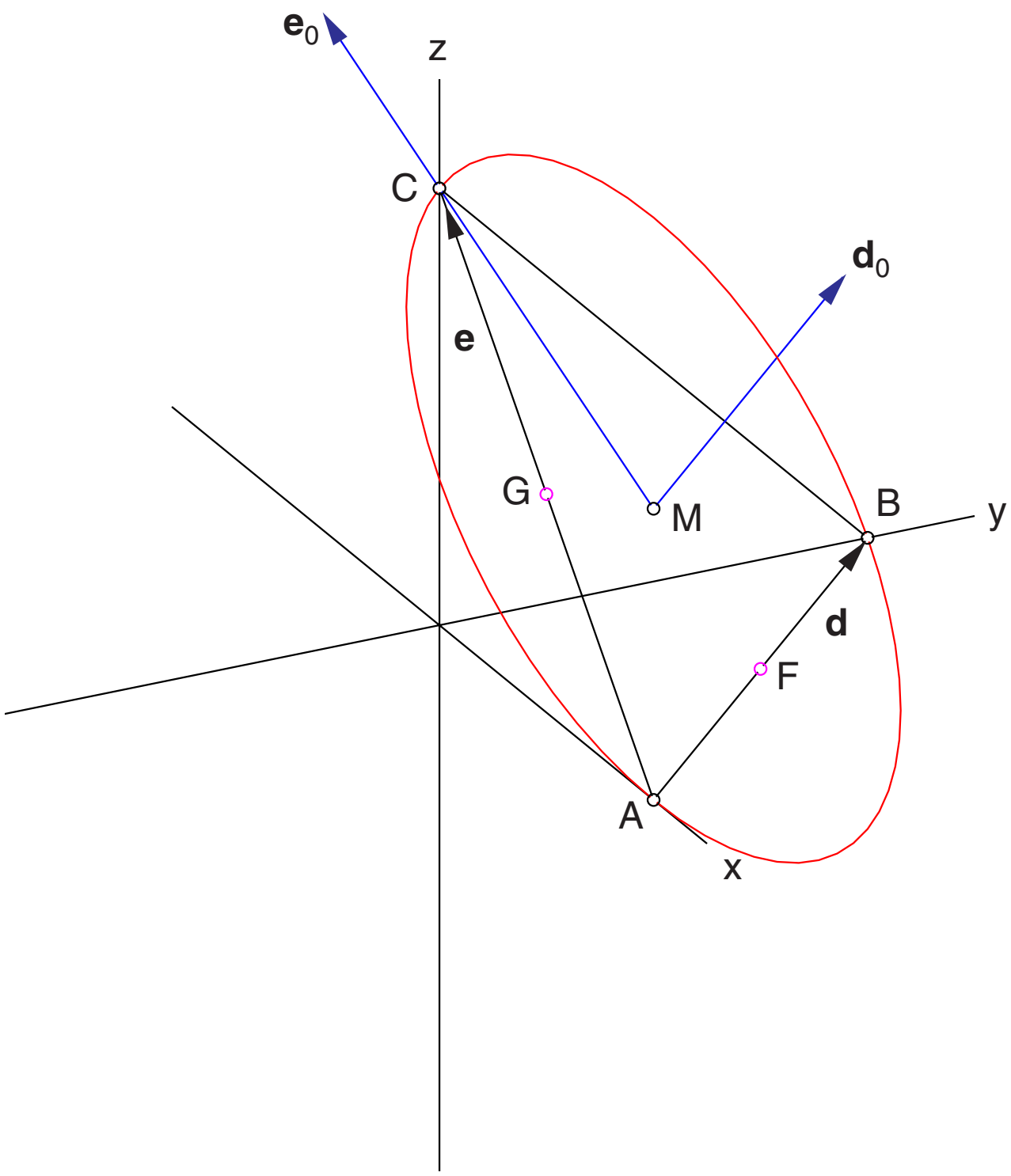
1.1 Circle in 3D / Introduction

Draw a circle in 3D through three points A, B, C.

1. Find center M as intersection of three planes:
   Through midpoint F, orthogonal to edge AB
   Through midpoint G, orthogonal to edge AC
   Triangle plane ABC
   This delivers three linear equations for M.

2. Calculate first base vector $\mathbf{d}_0$ by normalizing **d**.
3. Calculate second base vector $\mathbf{e}_0$ by Gramm-Schmidt, using **d** and **e**.

4. Draw circle in 3D and project points by camera onto view plane.



2.1 Circle in 3D / Mathematics

Calculations are done in this order:

Corner vectors
$\mathbf{a}, \mathbf{b}, \mathbf{c}$

Edges
$\mathbf{d} = \mathbf{b} - \mathbf{a}$
$\mathbf{e} = \mathbf{c} - \mathbf{a}$

Midpoints on edges
$\mathbf{f} = \mathbf{a} + 0.5\mathbf{d}$
$\mathbf{g} = \mathbf{a} + 0.5\mathbf{e}$

Triangle plane normal
$\mathbf{n} = \mathbf{d} \times \mathbf{e}$

Center M is the intersection of three planes
$\mathbf{d}^T\mathbf{x} = \mathbf{d}^T\mathbf{f}$
$\mathbf{e}^T\mathbf{x} = \mathbf{e}^T\mathbf{g}$
$\mathbf{n}^T\mathbf{x} = \mathbf{n}^T\mathbf{a}$

Solve the three linear equations
$\mathbf{A}\mathbf{x} = \mathbf{y}$
$\mathbf{m} = \mathbf{x}$

Radius vector
$\mathbf{r} = \mathbf{a} - \mathbf{m}$

Radius
$R = |\mathbf{r}|$

Orthogonal base vectors (Gramm-Schmidt)
$$\mathbf{h} = \mathbf{e} - \frac{\mathbf{d}^T\mathbf{e}}{\mathbf{d}^T\mathbf{d}}\mathbf{d}$$

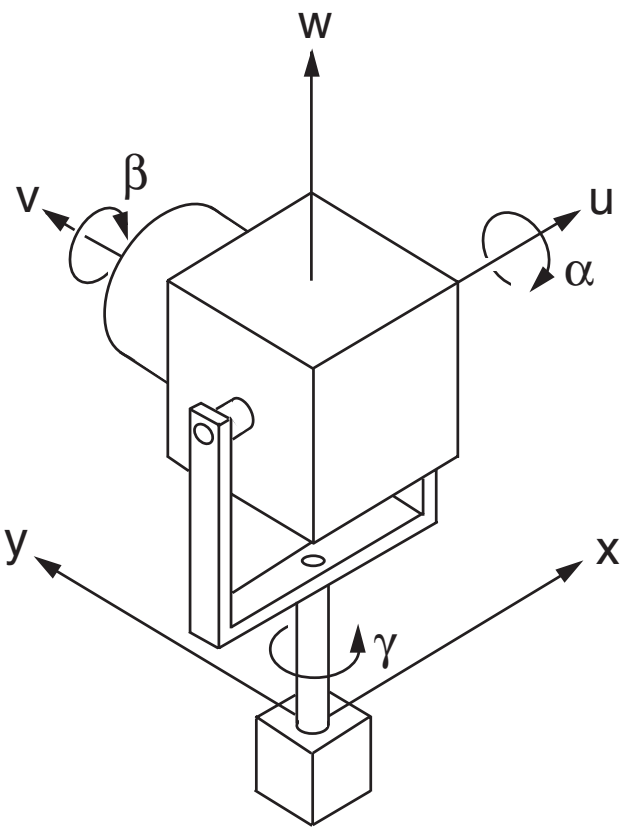
Normalization
$$\mathbf{d}_0 = \frac{\mathbf{d}}{|\mathbf{d}|}$$
$$\mathbf{e}_0 = \frac{\mathbf{h}}{|\mathbf{h}|}$$

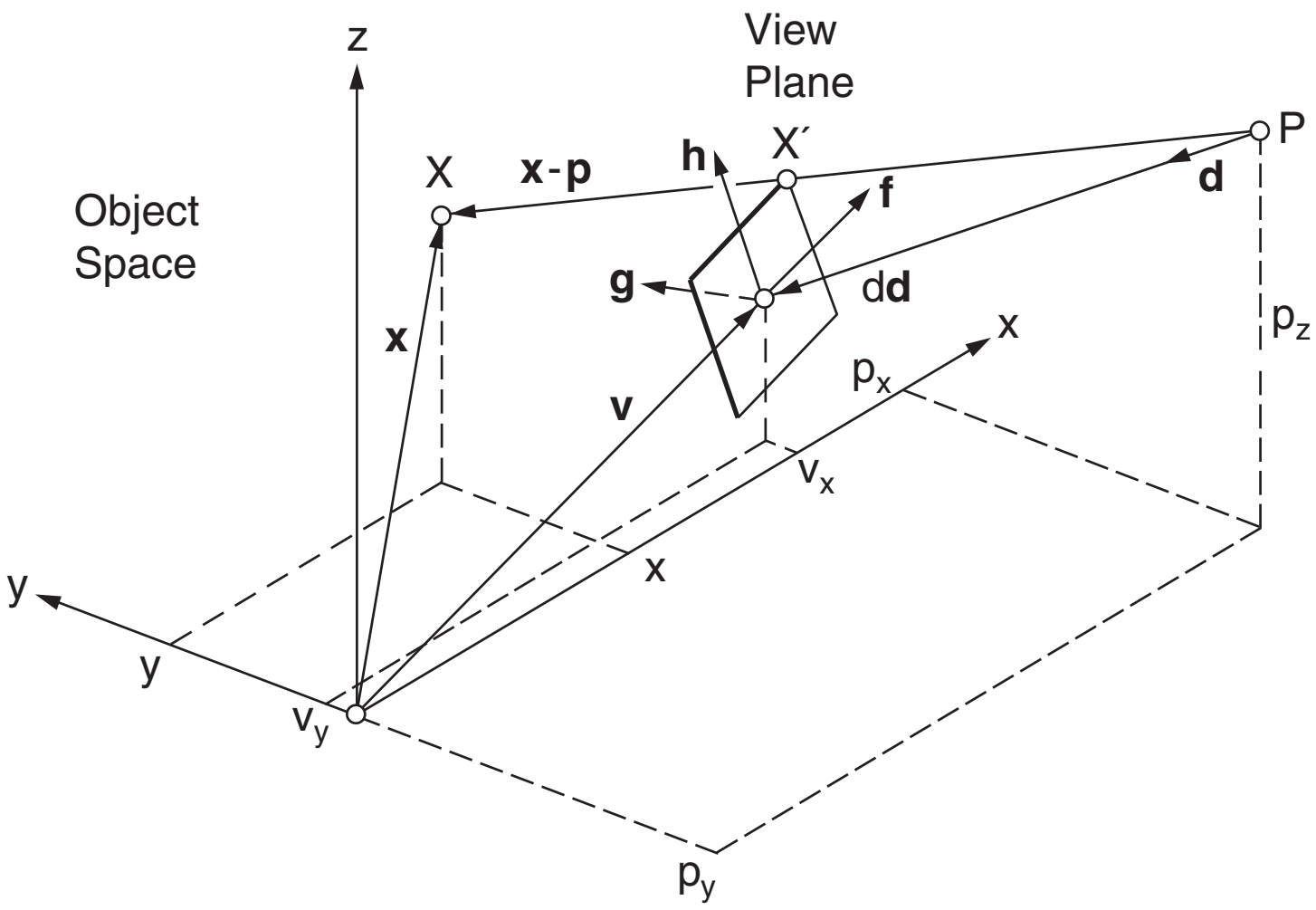
Circle points in 3D by parameter α
$\mathbf{x} = \mathbf{m} + R\cos(\alpha)\mathbf{d}_0 + R\sin(\alpha)\mathbf{e}_0$

Map $\mathbf{x}$ by camera to view plane

3.1 Circle in 3D / Projections

The necessary projections are defined in [1] and [2]. P is the center of projection, V the view point. The view plane is for the actual applications orthogonal to the view line.
d, **f**, **g** and **h** are not the same vectors as in the previous chapters.





4.1 Circle in 3D / PostScript Code

```
%!PS-Adobe-3.0   EPSF-3.0
%%BoundingBox:   0 0 808 808
%%Creator:       Gernot Hoffmann
%%Title:         Circle-3D
%%CreationDate:  Sept.05 2005

% Draw a circle through 3 points in 3D

/mm {2.834646 mul} def

/sx  100 mm def     % Length -sx..+sx
/sy  100 mm def
/x0  140 mm def     % Center
/y0  140 mm def

% Camera projection
/Map (Par) def       % Par Parallel projection
             % Per Perspective projection
/Bet  0 def          % Roll angle, default 0

/Px  10 def          % Center of projection
/Py  -5 def
/Pz   5 def

/Vx   0 def          % View point
/Vy   0 def
/Vz   0 def

/MatCam % Camera rotation matrix
{/eps 1E-4 def
/dx Px Vx sub def
/dy Py Vy sub neg def
/dz Pz Vz sub neg def
/dr dx dup mul dy dup mul add sqrt def
  dx abs eps gt dy abs eps gt or
{/Gam dx dy atan def }
{/Gam 0 def          } ifelse
  dz abs eps gt dr abs eps gt or
{/Alf dz dr atan def }
{/Alf 0 def          } ifelse
/ca Alf cos def /sa Alf sin def
/cb Bet cos def /sb Bet sin def
/cg Gam cos def /sg Gam sin def
/c11 cb cg mul sa sb mul sg mul sub def
/c12 cb sg mul sa sb mul cg mul add def
/c13 ca sb mul neg def
/c21 ca sg mul neg def
/c22 ca cg mul def
/c23 sa def
/c31 sb cg mul sa cb mul sg mul add def
/c32 sb sg mul sa cb mul cg mul sub def
/c33 ca cb mul def
/DD  dx dup mul dy dup mul add dz dup mul add sqrt def
} def

/Transf
{% Map=Par Parallel
 % Map=Per Perspective
 % xf yf zf on the stack expected
   /zc exch Pz sub def
   /yc exch Py sub def
   /xc exch Px sub def
   /uc c11 xc mul c12 yc mul add c13 zc mul add def
   /wc c31 xc mul c32 yc mul add c33 zc mul add def
   Map (Par) eq
   { /fd uc def /hd wc def } if
    Map (Per) eq
   {/vc c21 xc mul c22 yc mul add c23 zc mul add def
    /fd uc vc div DD mul def /hd wc vc div DD mul def } if
} def
```

4.2 Circle in 3D / PostScript code

```
/Axes
{  0 setgray
   0.3 mm sx div setlinewidth
   newpath
   -1  0  0 Transf fd hd moveto  1 0 0  Transf fd hd lineto
    0 -1  0 Transf fd hd moveto  0 1 0  Transf fd hd lineto
    0  0 -1 Transf fd hd moveto  0 0 1  Transf fd hd lineto
   stroke
  /fh 18 sx div def
  /Helvetica findfont fh scalefont setfont
%   1.1 0   0   Transf fd hd moveto (x) show
%   0   1.1 0   Transf fd hd moveto (y) show
%   0   0   1.1 Transf fd hd moveto (z) show
 } def

/AsubB
% C = A-B;  Matrix [ax ay az]-Matrix [bx by bz]—>Matrix [cx cy cz]
{/cbz exch def /cby exch def /cbx exch def
 /caz exch def /cay exch def /cax exch def
 /ccx cax cbx sub def /ccy cay cby sub def /ccz caz cbz sub def
  ccz ccy ccx
} def

/AmidB
% C=A+0.5*B;  Matrix [ax ay az]+0.5*Matrix [bx by bz]—>Matrix [cx cy cz]
{/cbz exch def /cby exch def /cbx exch def
 /caz exch def /cay exch def /cax exch def
 /ccx cax cbx 0.5 mul add def
 /ccy cay cby 0.5 mul add def
 /ccz caz cbz 0.5 mul add def
  ccz ccy ccx
} def

/AcroB
% C=AxB  Matrix [ax ay az] x Matrix [bx by bz]—>Matrix [cx cy cz]
{/cbz exch def /cby exch def /cbx exch def
 /caz exch def /cay exch def /cax exch def
 /ccx cay cbz mul caz cby mul sub def
 /ccy caz cbx mul cax cbz mul sub def
 /ccz cax cby mul cay cbx mul sub def
  ccz ccy ccx
} def

/AdotB
% c=A*B  Matrix [ax ay az]*Matrix [bx by bz]—>c
{/cbz exch def /cby exch def /cbx exch def
 /caz exch def /cay exch def /cax exch def
 /ccd cax cbx mul cay cby mul add caz cbz mul add def
  ccd
} bind def

/NormA
% A=A/Length(A)
% Matrix [ax ay az]—>Matrix [ax ay az]
{/caz exch def /cay exch def /cax exch def
 /ccd cax dup mul cay dup mul add caz dup mul add sqrt def
  caz ccd div cay ccd div cax ccd div
} def

/LengA
% c=Leng(A) Matrix [ax ay az]—>c
{/caz exch def /cay exch def /cax exch def
 /ccd cax dup mul cay dup mul add caz dup mul add sqrt def
  ccd
} def

/Dot
{/ccz exch def /ccy exch def /ccx exch def
  0.3 mm sx div setlinewidth
  ccx ccy ccz Transf
  currentcolor /Blu exch def /Grn exch def /Red exch def
  1 1 1 setrgbcolor
  newpath
  fd hd 0.7 mm sx div 0 360 arc fill
  Red Grn Blu setrgbcolor
  fd hd 1 mm sx div 0 360 arc stroke
} def
```

4.3 Circle in 3D / PostScript code

```
/Line
{/c1z exch def /c1y exch def /c1x exch def
 /c2z exch def /c2y exch def /c2x exch def
  0.3 mm sx div setlinewidth
  c1x c1y c1z Transf fd hd moveto
  c2x c2y c2z Transf fd hd lineto stroke
} def

/Circle
{ 1 0 0 setrgbcolor
 newpath
/Ang 0 def
/Nan 60 def
/Dan 360 Nan div def
/Can Ang cos Rad mul def
/San Ang sin Rad mul def
 Mx Dx0 Can mul add Ex0 San mul add
 My Dy0 Can mul add Ey0 San mul add
 Mz Dz0 Can mul add Ez0 San mul add
 Transf fd hd moveto
 1 1 Nan
 { pop
  /Ang Ang Dan add def
  /Can Ang cos Rad mul def
  /San Ang sin Rad mul def
   Mx Dx0 Can mul add Ex0 San mul add
   My Dy0 Can mul add Ey0 San mul add
   Mz Dz0 Can mul add Ez0 San mul add
   Transf fd hd lineto
 } for
  stroke
} def

% Linear equation solver
/N  3 def
/n1 N 1 sub def

/ANN N N mul array def
/XN  N array def
/YN  N array def

/i 0 def /j 0 def /k 0 def
/dA 0 def /max 0 def /s 0 def /h 0 def /q 0 def /aik 0 def /bik 0 def /bkk 0 def
/pa 0 def /ca 0 def

/m
% Memory map; i,k on stack; top=N*i+k
{ exch
  N mul add
} bind def

/HoGaussP
{% Solve  ANN*XN=YN for XN and det(ANN)
 % Overwrite all arrays
/pa N array def
/ca N array def
/n1 N 1 sub def
/dA 1 def
0 1 n1 1 sub
{/k exch def
 /max 0 def
  pa k 0 put
  k 1 n1
  {/i exch def
   /s 0 def
    k 1 n1
    {/j exch def
     /s s ANN i j m get abs add def
    } for %j
     /bik ANN i k m get abs def
     /q bik s div def
      q max gt {/max q def pa k i put } if
  } for %i
```

4.4 Circle in 3D / PostScript code

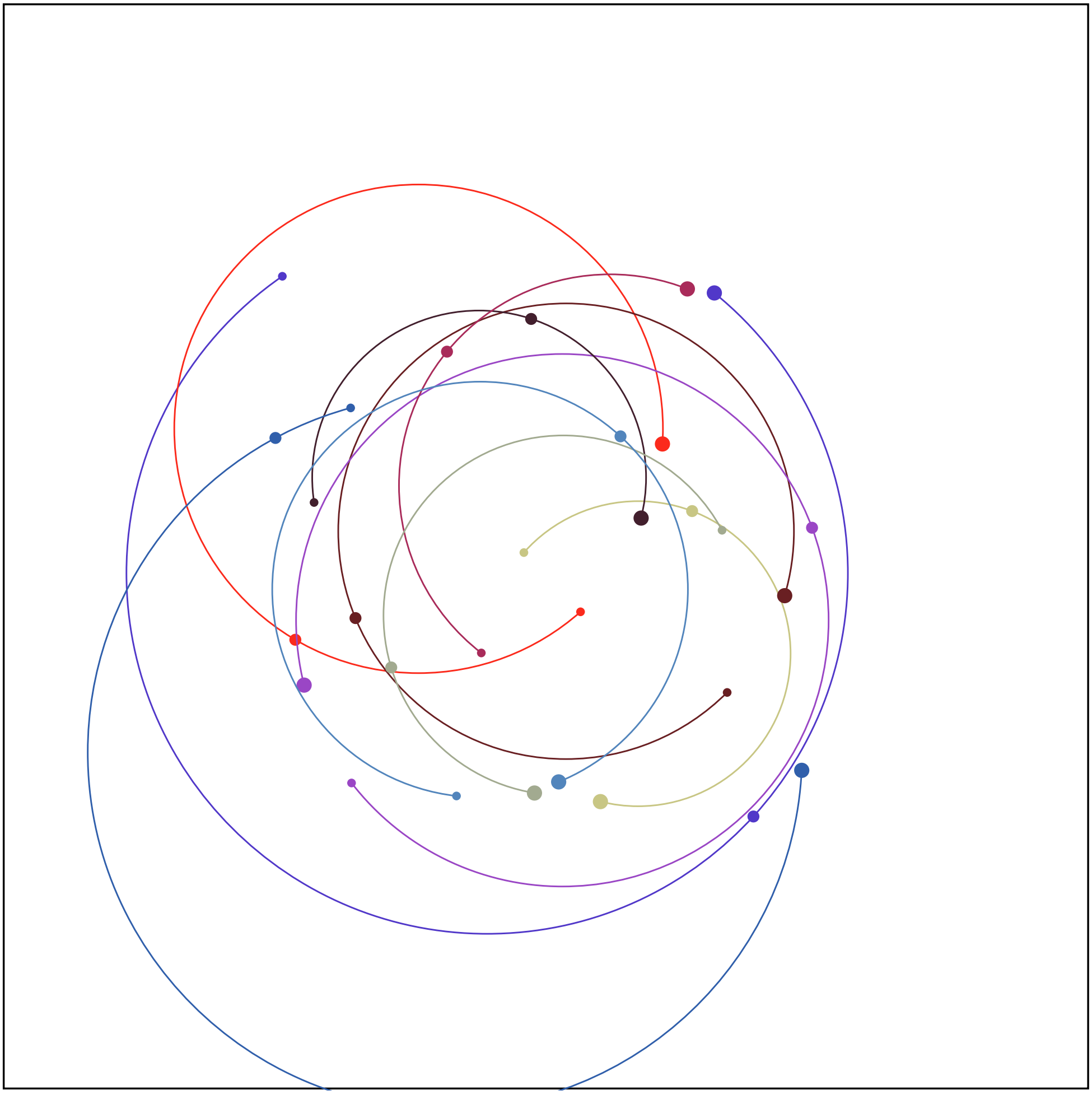
```
pa k get k ne
{/dA dA neg def
  0 1 n1
  {/j exch def
   /h ANN k j m get def
    ANN k j m ANN pa k get j m get put
    ANN pa k get j m h put
  } for %j
} if
/dA dA ANN k k m get mul def
k 1 add 1 n1
{/i exch def
 /aik ANN i k m get def
 /akk ANN k k m get def
 /bkk akk abs def
  bkk 1 lt
  {/bik aik abs def
  } if
  ANN i k m aik akk div put
  k 1 add 1 n1
  {/j exch def
     ANN i j m ANN i j m get ANN i k m get ANN k j m get mul sub put
   } for %j
  } for %i
} for %k
/dA dA ANN n1 n1 m get mul
0 1 n1 1 sub
{/k exch def
  pa k get k ne
 {/h YN k get def
   YN k YN pa k get get put
   YN pa k get h put
 } if
} for %k
0 1 n1
{/i exch def
  ca i YN i get put
  0 1 i 1 sub
  {/j exch def
    ca i ca i get ANN i j m get ca j get mul sub put
  } for %j
} for %i
n1 -1 0
{/i exch def
 /s ca i get def
  i 1 add 1 n1
  {/k exch def
   /s s ANN i k m get YN k get mul sub def
  } for %k
 /bik ANN i i m get abs def
  YN i s ANN i i m get div put
} for %i
0 1 n1
{/i exch def
  XN i YN i get put
} for %i
} def % HoGaussP

% Begin

false setstrokeadjust

x0 y0 translate
sx sy scale

% Set camera
MatCam
```

4.5 Circle in 3D / PostScript code

```
% Corners
/Ax +0.8 def /Ay +0.0 def /Az +0.0 def
/Bx +0.0 def /By +0.8 def /Bz +0.0 def
/Cx +0.0 def /Cy +0.0 def /Cz +0.8 def

% Edges
Bx By Bz Ax Ay Az AsubB /Dx exch def /Dy exch def /Dz exch def
Cx Cy Cz Ax Ay Az AsubB /Ex exch def /Ey exch def /Ez exch def

% Normal
Dx Dy Dz Ex Ey Ez AcroB /Nx exch def /Ny exch def /Nz exch def

% Midpoints of edges
Ax Ay Az Dx Dy Dz AmidB /Fx exch def /Fy exch def /Fz exch def
Ax Ay Az Ex Ey Ez AmidB /Gx exch def /Gy exch def /Gz exch def

% Center M of sphere/circle
YN 0 Dx Dy Dz Fx Fy Fz AdotB put
YN 1 Ex Ey Ez Gx Gy Gz AdotB put
YN 2 Nx Ny Nz Ax Ay Az AdotB put
ANN 0 Dx put ANN 1 Dy put ANN 2 Dz put
ANN 3 Ex put ANN 4 Ey put ANN 5 Ez put
ANN 6 Nx put ANN 7 Ny put ANN 8 Nz put
HoGaussP
/Mx XN 0 get def /My XN 1 get def /Mz XN 2 get def

% Radius
Ax Ay Az Mx My Mz AsubB /Rx exch def /Ry exch def /Rz exch def
/Rad Rx Ry Rz LengA def

% Gramm-Schmidt for orthogonal vector base in 3D
/DE Dx Dy Dz Ex Ey Ez AdotB def
/DD Dx Dy Dz Dx Dy Dz AdotB def
/KK DE DD div def
/Hx Ex Dx KK mul sub def
/Hy Ey Dy KK mul sub def
/Hz Ez Dz KK mul sub def

% Normalized base vectors
Dx Dy Dz NormA /Dx0 exch def /Dy0 exch def /Dz0 exch def
Hx Hy Hz NormA /Ex0 exch def /Ey0 exch def /Ez0 exch def

Axes

% Edges
0 0 0 setrgbcolor
Ax Ay Az Bx By Bz Line
Bx By Bz Cx Cy Cz Line
Cx Cy Cz Ax Ay Az Line

% Vector base in 3D
0 0 1 setrgbcolor
Mx My Mz Mx Dx0 add My Dy0 add Mz Dz0 add Line
Mx My Mz Mx Ex0 add My Ey0 add Mz Ez0 add Line

% Draw circle in 3D by parameter ‘angle’ using vector base
Circle

% Corners
0 0 0 setrgbcolor
Ax Ay Az Dot
Bx By Bz Dot
Cx Cy Cz Dot

% Midpoints
1 0 1 setrgbcolor
Fx Fy Fz Dot
Gx Gy Gz Dot

% Center of sphere/circle
0 0 0 setrgbcolor
Mx My Mz Dot

showpage
```

5.1 Circle in 2D / Task

Given are three points A,B,C. Draw the circle segment through the points just in this order: from A to C via B. The example shows the points A,B,C with increasing dot size.



6.1 Circle in 2D / Mathematics

The 2D concept is similar to the 3D concept. Instead of three planes we have two lines. Point A can be connected to C by one of two circle segments. Valid is the segment which contains B. The angle between the A-radius **p** and the B-radius **q** is β. The angle between the A-radius **p** and the C-radius **r** is γ. For $\gamma > \beta$ the segment has to be drawn counterclockwise.

Corner vectors

$$\mathbf{a}, \mathbf{b}, \mathbf{c}$$

Edges

$$\mathbf{d} = \mathbf{b} - \mathbf{a}$$
$$\mathbf{e} = \mathbf{c} - \mathbf{a}$$

Midpoints on edges

$$\mathbf{f} = \mathbf{a} + 0.5\mathbf{d}$$
$$\mathbf{g} = \mathbf{a} + 0.5\mathbf{e}$$

Center M is the intersection of two lines

$$\mathbf{d}^T\mathbf{x} = \mathbf{d}^T\mathbf{f}$$
$$\mathbf{e}^T\mathbf{x} = \mathbf{e}^T\mathbf{g}$$

Solve the two linear equations by *Cramer*

$$\mathbf{A}\,\mathbf{x} = \mathbf{y}$$
$$\mathbf{m} = \mathbf{x}$$

Radii

$$\mathbf{p} = \mathbf{a} - \mathbf{m}$$
$$\mathbf{q} = \mathbf{b} - \mathbf{m}$$
$$\mathbf{r} = \mathbf{c} - \mathbf{m}$$
$$R = |\mathbf{p}|$$

Angles

$$\alpha_a = \text{atan2}(p_y, p_x)$$
$$\alpha_b = \text{atan2}(q_y, q_x)$$
$$\alpha_c = \text{atan2}(r_y, r_x)$$
$$\beta = \text{atan2}((\mathbf{p} \times \mathbf{q})_z, \mathbf{p}^T\mathbf{q})$$
$$\gamma = \text{atan2}((\mathbf{p} \times \mathbf{r})_z, \mathbf{p}^T\mathbf{r})$$

Draw at **m** with R from α_a to α_c
If $\gamma > \beta$ then draw ccw else draw cw

PostScript atan2
/angle num den atan def

PostScript circle
m_x m_y R α_a α_c arc stroke % ccw
m_x m_y R α_a α_c arcn stroke % cw

7.1 Circle in 2D / PostScript code

```
%!PS-Adobe-3.0  EPSF-3.0
%%BoundingBox:  0 0 568 568
%%Creator:      Gernot Hoffmann
%%Title:        Circle2D
%%CreationDate: March 02 / 2007

% Circle through three points

/mm {2.834646 mul} def

/DotF
{1 dict
 begin
 /R exch def
 /y exch def
 /x exch def
  x y R 0 360 arc fill
 end
} def

/Srand
% Initialization and seed for random generator
% Call:     n Srand
% Requires: none
{/rand exch def       % Input seed
 /rb 2 15 exp def     % 15 bit
 /ra  181 def         % Optimized for period 8192
 /rand rand 2 mul 1 add def % Optimize seed
} def

/Xrand % random float number range 0..+1
% Call:     Xrand
% Requires: Srand called once
{/rand ra rand mul dup rb div cvi rb mul sub round cvi def
  rand rb div % on stack
} def

/Circle2D
{ 1 dict
 begin
% Corners
/cy exch def /cx exch def
/by exch def /bx exch def
/ay exch def /ax exch def
% Edges
/dx bx ax sub def
/dy by ay sub def
/ex cx ax sub def
/ey cy ay sub def
% Centers of Edges
/fx ax dx 0.5 mul add def
/fy ay dy 0.5 mul add def
/gx ax ex 0.5 mul add def
/gy ay ey 0.5 mul add def
% Build linear equations
/r1 dx fx mul dy fy mul add def
/r2 ex gx mul ey gy mul add def
% Solve by Cramer
/D0 dx ey mul dy ex mul sub def
/Dx r1 ey mul dy r2 mul sub def
/Dy dx r2 mul r1 ex mul sub def
% Center of Circle
D0 abs 1E-6 gt
{
/mx Dx D0 div def
/my Dy D0 div def
% Radii
/px ax mx sub def
/py ay my sub def
/qx bx mx sub def
/qy by my sub def
/rx cx mx sub def
/ry cy my sub def
% Radius
/R px dup mul py dup mul add sqrt def
```

7.2 Circle in 2D / PostScript code

```
% Angles
/aa py px atan def
/ab qy qx atan def
/ac ry rx atan def
% Check direction
/bet px qy mul py qx mul sub px qx mul py qy mul add atan def
/gam px ry mul py rx mul sub px rx mul py ry mul add atan def
% Draw circle segment
 gam bet gt  {mx my R aa ac arc  stroke}
             {mx my R aa ac arcn stroke} ifelse
} if
end
} def

 1 1 566 566 rectstroke

/sc 100 mm def
/tr  50 mm def

 tr tr translate
 sc sc scale
 0.3 mm sc div setlinewidth

 1 Srand % use any seed

10
{/ax Xrand def /ay Xrand def
 /bx Xrand def /by Xrand def
 /cx Xrand def /cy Xrand def
  Xrand Xrand Xrand setrgbcolor
  ax ay 0.008 DotF
  bx by 0.011 DotF
  cx cy 0.014 DotF
  ax ay bx by cx cy Circle2D
} repeat

showpage
```

8.1 References

[1] G.Hoffmann
Euler angles and projections
http://www.fho-emden.de/~hoffmann/euler26112001.pdf

[2] G.Hoffmann
Planar projections
http://www.fho-emden.de/~hoffmann/project18032004.pdf

This doc:
http://www.fho-emden.de/~hoffmann/circle3d06092005.pdf

Gernot Hoffmann
March 03 / 2007
Website
Load Browser / Click here