The `tikz-3dplot-circleofsphere` Package: Drawing circles of a sphere with `tikz-3dplot`

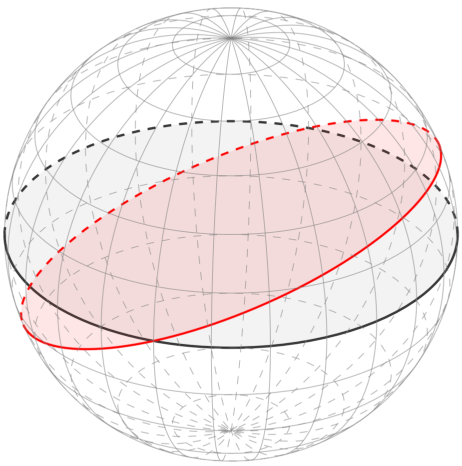
Matthias Wolff [0000–0002–3895–7313]

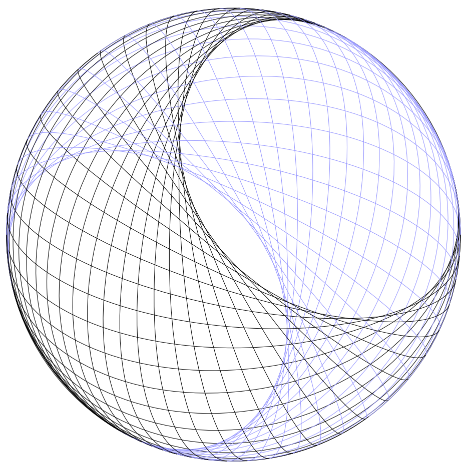
BTU Cottbus-Senftenberg

July 27, 2018

Abstract

A *circle of a sphere* is a circle drawn on a spherical surface like, for instance, circles of latitude or longitude. Circles in arbitrary 3D positions can be drawn with TikZ [2] very easily using a transformed coordinate system provided by the `tikz-3dplot` package [1] (that is because TikZ can only draw circles on the xy-plane). However, automatically distinguishing the parts of the circle lying on the front and back sides of the sphere, e.g. by drawing a solid arc on the front side and a dashed one on the back side, is a somewhat tricky feat. The `tikz-3dplot-circleofsphere` package will perform that feat for you.





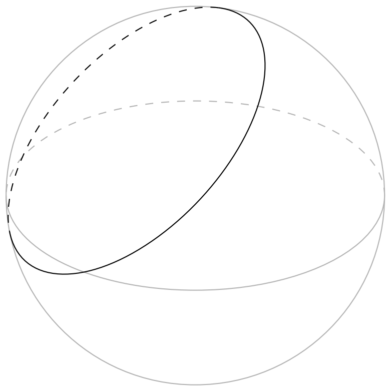
```
\documentclass{standalone}
\usepackage{tikz-3dplot-circleofsphere}
\begin{document}
  \centering
  \def\R{3}
  \tdplotsetmaincoords{60}{125}
  \begin{tikzpicture}[tdplot_main_coords]
    \draw[tdplot_screen_coords,very thin,gray] (0,0,0) circle (\R);
    \tdplotCsDrawLatCircle%
      [thick,tdplotCsFill/.style={opacity=0.05}]{\R}{0}
    \tdplotCsDrawGreatCircle%
      [red,thick,tdplotCsFill/.style={opacity=0.1}]{\R}{105}{-23.5}
    \foreach \a in {-75,-60,...,75}
      {\tdplotCsDrawLatCircle[very thin,gray]{\R}{\a}}
    \foreach \a in {0,15,...,165}
      {\tdplotCsDrawLonCircle[very thin,gray]{\R}{\a}}
  \end{tikzpicture}
\end{document}
```

```
\documentclass{standalone}
\usepackage{tikz-3dplot-circleofsphere}
\begin{document}
  \centering
  \def\R{3}
  \tdplotsetmaincoords{60}{125}
  \begin{tikzpicture}[tdplot_main_coords]
    \def\e{80};
    \draw[tdplot_screen_coords,very thin] (0,0,0) circle (\R);
    \foreach \a in {0,5,...,175} {
      \tdplotCsDrawGreatCircle%
        [very thin, tdplotCsBack/.style={very thin,blue!40}]%
        {\R}{\a}{90*sin(\a)*sin(\e)}
    }
  \end{tikzpicture}
\end{document}
```

Contents

1 Just Looking for the Minimalist Code?

There you go!



```
\documentclass{standalone}
\usepackage{tikz,tikz-3dplot}
%% >> MINIMALIST CIRCLE OF SHPERE DRAWING CODE ___________________________________________________________________
\newcommand\scircle[4]{%
  \tdplotsetrotatedcoords{#2}{#3}{0}                                                % Rotate coordinate system
  \let\a\tdplotalpha                                                                % alpha (rotated coord. system)
  \let\b\tdplotbeta                                                                 % beta (rotated coord. system)
  \let\p\tdplotmainphi                                                              % phi (main coord. system)
  \let\t\tdplotmaintheta                                                            % theta (main coord. system)
  \pgfmathsetmacro\azx{cos(\a)*cos(\b)*sin(\p)*sin(\t) - sin(\b)*cos(\t) - cos(\b)*cos(\p)*sin(\a)*sin(\t)}
  \pgfmathsetmacro\azy{-cos(\a)*cos(\p)*sin(\t) - sin(\a)*sin(\p)*sin(\t)}
  \pgfmathsetmacro\azz{cos(\b)*cos(\t) + cos(\a)*sin(\b)*sin(\p)*sin(\t) - cos(\p)*sin(\a)*sin(\b)*sin(\t)}
  \pgfmathsetmacro\Re {#1*cos(#4)}                                                  % Radius of circle
  \pgfmathsetmacro\ze {#1*sin(#4)}                                                  % z-coordinate of drawing plane
  \pgfmathsetmacro\coX{\ze*cos(#2)*sin(#3)}                                         % x-coordinate offset for ze
  \pgfmathsetmacro\coY{\ze*sin(#2)*sin(#3)}                                         % y-coordinate offset for ze
  \pgfmathsetmacro\coZ{\ze*cos(#3)}                                                 % z-coordinate offset for ze
  \coordinate (coffs) at (\coX,\coY,\coZ);                                          % Offset as coordinate value
  \tdplotsetrotatedcoordsorigin{(coffs)}                                            % Offset coordinate system
  \begin{scope}[tdplot_rotated_coords]                                              % Drawing scope >>
    \pgfmathsetmacro\tanEps{tan(#4)}                                                %   Tangent of elevation angle
    \pgfmathsetmacro\bOneside{((\tanEps)^2)>=(((\azx)^2+(\azy)^2)/(\azz)^2)}        %   Circle entirely on one side?
    \ifthenelse{\bOneside=1}{%                                                      %   Circle on one side of sphere >>
      \pgfmathsetmacro\bFrontside{(\azx*\Re+\azz*\ze)>=0}                           %     Circle entirely on front side?
       \ifthenelse{\bFrontside=1}                                                   %     |
        {\draw (0,0) circle (\Re);}                                                 %     Draw on front side
        {\draw[dashed] (0,0) circle (\Re);}                                         %     Draw on back side
    }{%                                                                             %   << Circle on both sides >>
      \pgfmathsetmacro\u{\azy}                                                      %     Substitution u=...
      \pgfmathsetmacro\v{sqrt( (\azx)^2 + (\azy)^2 - (\azz)^2*(\tanEps)^2 )}        %     Substitution v=...
      \pgfmathsetmacro\w{\azx - \azz*\tanEps}                                       %     Substitution w=...
      \pgfmathsetmacro\aPhiBf{2*atan2(\u-\v,\w)}                                    %     Back->front crossing angle
      \pgfmathsetmacro\aPhiFb{2*atan2(\u+\v,\w)}                                    %     Front->back crossing angle
      \pgfmathsetmacro\bUnwrapA{(\aPhiFb-\aPhiBf)>360}                              %     Unwrap front->back angle #1?
      \pgfmathsetmacro\bUnwrapB{\aPhiBf>\aPhiFb}                                    %     Unwrap front->back angle #2?
      \ifthenelse{\bUnwrapA=1}{\pgfmathsetmacro\aPhiBf{\aPhiBf+360}}{}              %     Unwrap front->back angle #1
      \ifthenelse{\bUnwrapB=1}{\pgfmathsetmacro\aPhiBf{\aPhiBf-360}}{}              %     Unwrap front->back angle #2
      \draw[dashed] (\aPhiFb:\Re) arc (\aPhiFb:{\aPhiBf+360}:\Re);                  %     Draw back side arc
      \draw (\aPhiBf:\Re) arc (\aPhiBf:\aPhiFb:\Re);                                %     Draw back side arc
    }                                                                               %   <<
  \end{scope}                                                                       % << (Drawing scope)
}
%% << _____________________________________________________________________________________________________________
\begin{document}
  \tdplotsetmaincoords{60}{125}                                                     % Set main coordintate system
  \begin{tikzpicture}[tdplot_main_coords]                                           % TikZ picture >>
    \begin{scope}[black!30]                                                         %   Draw in gray >>
      \draw[tdplot_screen_coords] (0,0,0) circle (2.5);                             %     Sphere outline
      \scircle{2.5}{0}{0}{0}                                                        %     Equator
    \end{scope}                                                                     %   <<
    \scircle{2.5}{-40}{40}{30}                                                      %   Draw another sphere circle
  \end{tikzpicture}                                                                 % <<
\end{document}
```

Want some more convenience or interested in what we did? Read on...

2 The `tikz-3dplot-circleofsphere` Package

2.1 Installation

Just copy the `tikz-3dplot-circleofsphere.sty` file into your project folder and include the package with `\usepackage{tikz-3dplot-circleofsphere}`. That's all.

2.2 Drawing Commands

`\tdplotgcDrawGreatCircle[style]{alpha}{beta}{R}`

Draws a great circle on a sphere. The circle is drawn on the xy-plane of a coordinate system rotated by `\tdplotsetrotatedcoords`. See Example **??** in Section 2.4 for an illustration.

Parameters

`style` TikZ style (optional). Use

- `tdplotGcFront/.style={...}` to style the front side semicircle,
- `tdplotGcBack/.style={...}` to style the back side semicircle,
- `tdplotGcFill/.style={...}` to style the fill of the circle.

The default fill style is `opcacity=0`. To make the interior of the circle visible, you must specifiy an opacity value > 0, e.g. `tdplotGcFill/.style={opacity=0.1}`.

`alpha` α angle of rotated coordinate system, to be passed to `\tdplotsetrotatedcoords`

`beta` β angle of rotated coordinate system, to be passed to `\tdplotsetrotatedcoords`

`R` Radius of sphere

Output

none

Remarks

Use `\tdplotgcDrawGreatCircleExtras[style]{alpha}{beta}{R}` to draw some extra information for the great circle. See Example **??** in Section 2.4 for an illustration.

`\tdplotgcDrawPoint[style]{alpha}{beta}{R}`

Draws a point on a sphere. The point is drawn at position $(0, 0, \mathtt{R})$ of a coordinate system rotated by `\tdplotsetrotatedcoords`. See Example **??** in Section 2.4 for an illustration.

Parameters

`style` TikZ style (optional). Use

- `tdplotPtFront/.style={...}` to style a front side point and
- `tdplotPtBack/.style={...}` to style a back side point.

`alpha` α angle of rotated coordinate system, to be passed to `\tdplotsetrotatedcoords`

`beta` β angle of rotated coordinate system, to be passed to `\tdplotsetrotatedcoords`

`R` Radius of sphere

Output

none

Remarks

Use `\tdplotgcDrawPointExtras[style]{alpha}{beta}{R}` to draw some extra information for the great circle.

2.3 Auxiliary Commands

`\tdtdplotgcFrontsidePoint`

Invoked by `\tdplotgcDrawPoint` to draw a point on the front side of a sphere. Redefine to customize.

Parameters

none

Output

none

`\tdtdplotgcBacksidePoint`

Invoked by `\tdplotgcDrawPoint` to draw a point on the back side of a sphere. Redefine to customize.

Parameters

none

Output

none

`\tdplotgcComputeTransformRotScreen`

Computes the elements of the full rotation matrix

$$A = \begin{pmatrix} a_{xx} & a_{xy} & a_{xz} \\ a_{yx} & a_{yy} & a_{yz} \\ a_{zx} & a_{zy} & a_{zz} \end{pmatrix}.$$

See Section 3.1 for details.

Parameters

none

Output

`\axx`	Element a_{xx} of full rotation matrix
`\axy`	Element a_{xy} of full rotation matrix

...

\azz   Element a_{zz} of full rotation matrix

Remarks

The command uses some internal variables of `tikz-3dplot`, namely `\tdplotalpha`, `\tdplotbeta`, `\tdplotmainphi`, and `\tdplotmaintheta`.

2.4 Examples

Examples ?? and ?? (see below) demonstrate the usage of the `tikz-3dplot-circleofsphere` package.

3 Implementation Details

3.1 The Maths

Front and Back Side Arcs of Great Circles

Denote by $P = (x\,y\,z)^\top$ a point in a 3D coordinate system. `tikz-3dplot` [1] transforms that point in to screen coordinates $P' = (x'\,y'\,z')^\top$ by

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = R^d(\phi,\theta) \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{1}$$

with the rotation matrix[1]

$$\begin{aligned} R^d(\phi,\theta) &= \left(R^{z'}(\phi)\, R^x(\theta)\right)^\top \qquad (2) \\ &= \left(\begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{pmatrix}\right)^\top \\ &= \begin{pmatrix} \cos\phi & \sin\phi & 0 \\ -\cos\theta\sin\phi & \cos\theta\cos\phi & +\sin\theta \\ \sin\theta\sin\phi & -\sin\theta\cos\phi & \cos\theta \end{pmatrix}. \end{aligned}$$

As TikZ can draw arcs and circles on the xy plane only, we need to rotate the coordinate frame again for drawing great circles in arbitrary position. To this end, we use `tikz-3dplot`'s rotated coordinate system[2]

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = R^d(\phi,\theta)\, D(\alpha,\beta,\gamma) \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{3}$$

with the rotation matrix (cf. [1, p. 7])

$$\begin{aligned} D(\alpha,\beta,0) &= R^z(\alpha) R^y(\beta) \qquad (4) \\ &= \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} \\ &= \begin{pmatrix} \cos\alpha\cos\beta & -\sin\alpha & \cos\alpha\sin\beta \\ \sin\alpha\cos\beta & \cos\alpha & \sin\alpha\sin\beta \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} \end{aligned}$$

[1] Note that equation (2.1) in [1] is wrong! Corrections are marked in red.
[2] Note that equation (2.4) in [1] is wrong! Corrections are marked in red.

where we deliberately omitted the last rotation $R^z(\gamma)$ by choosing $\gamma = 0$. Thus, the full rotation matrix for drawing a great circle is

$$
\begin{aligned}
A = \begin{pmatrix} a_{xx} & a_{xy} & a_{xz} \\ a_{yx} & a_{yy} & a_{yz} \\ a_{zx} & a_{zy} & a_{zz} \end{pmatrix} &= R^d(\phi, \theta)\, D(\alpha, \beta, 0) \qquad (5)\\
&= \begin{pmatrix} \cos\phi & \sin\phi & 0 \\ -\cos\theta\sin\phi & \cos\theta\cos\phi & \sin\theta \\ \sin\theta\sin\phi & -\sin\theta\cos\phi & \cos\theta \end{pmatrix} \begin{pmatrix} \cos\alpha\cos\beta & -\sin\alpha & \cos\alpha\sin\beta \\ \sin\alpha\cos\beta & \cos\alpha & \sin\alpha\sin\beta \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} \\
&= \left(\begin{matrix} \cos\alpha\cos\beta\cos\phi + \cos\beta\sin\alpha\sin\phi \\ \cos\beta\cos\phi\sin\alpha\cos\theta - \cos\alpha\cos\beta\cos\theta\sin\phi - \sin\beta\sin\theta \\ \cos\alpha\cos\beta\sin\phi\sin\theta - \sin\beta\cos\theta - \cos\beta\cos\phi\sin\alpha\sin\theta \end{matrix} \right. \\
&\qquad \begin{matrix} \cos\alpha\sin\phi - \cos\phi\sin\alpha \\ \cos\alpha\cos\phi\cos\theta + \sin\alpha\cos\theta\sin\phi \\ -\cos\alpha\cos\phi\sin\theta - \sin\alpha\sin\phi\sin\theta \end{matrix} \\
&\qquad \left. \begin{matrix} \cos\alpha\cos\phi\sin\beta + \sin\alpha\sin\beta\sin\phi \\ \cos\beta\sin\theta - \cos\alpha\sin\beta\cos\theta\sin\phi + \cos\phi\sin\alpha\sin\beta\cos\theta \\ \cos\beta\cos\theta + \cos\alpha\sin\beta\sin\phi\sin\theta - \cos\phi\sin\alpha\sin\beta\sin\theta \end{matrix} \right)
\end{aligned}
$$

Independently of A, the boundary of the sphere with radius r is a circle with the parametric representation

$$
\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} r\cos\varphi \\ r\sin\varphi \\ 0 \end{pmatrix} \qquad (6)
$$

in screen coordindates.

The screen coordinates of a rotated great circle with the parametric representation

$$
\begin{pmatrix} x(\varphi) \\ y(\varphi) \\ z(\varphi) \end{pmatrix} = \begin{pmatrix} r\cos\varphi \\ r\sin\varphi \\ 0 \end{pmatrix} \qquad (7)
$$

are

$$
\begin{aligned}
\begin{pmatrix} x'(\varphi) \\ y'(\varphi) \\ z'(\varphi) \end{pmatrix} = A \begin{pmatrix} x(\varphi) \\ y(\varphi) \\ z(\varphi) \end{pmatrix} = A \begin{pmatrix} r\cos\varphi \\ r\sin\varphi \\ 0 \end{pmatrix} &= \begin{pmatrix} a_{xx} & a_{xy} & a_{xz} \\ a_{yx} & a_{yy} & a_{yz} \\ a_{zx} & a_{zy} & a_{zz} \end{pmatrix} \begin{pmatrix} r\cos\varphi \\ r\sin\varphi \\ 0 \end{pmatrix} \qquad (8)\\
&= \begin{pmatrix} a_{xx}\cdot r\cos\varphi + a_{xy}\cdot r\sin\varphi \\ a_{yx}\cdot r\cos\varphi + a_{yy}\cdot r\sin\varphi \\ a_{zx}\cdot r\cos\varphi + a_{zy}\cdot r\sin\varphi \end{pmatrix}
\end{aligned}
$$

The $z'(\varphi)$ coordinates are not plotted. However, they are useful for determining which semicircle of the rotated great circle is

$$
\begin{aligned}
\text{visible} \quad & z'(\varphi) \geq 0 \quad \text{and which part is} \qquad (9)\\
\text{invisible} \quad & z'(\varphi) < 0.
\end{aligned}
$$

We denote by φ_0 the two boundary angles between the visible and invisible semicircles. Assuming $r \neq 0$ and $\cos\varphi_0 \neq 0$ we compute them by

$$
\begin{aligned}
0 \overset{!}{=} z'(\varphi_0) &= a_{zx}\cdot r\cos\varphi_0 + a_{zy}\cdot r\sin\varphi_0 \qquad (10)\\
&= a_{zx} + a_{zy}\tan\varphi_0
\end{aligned}
$$

from which we derive

$$
\tan\varphi_0 = -\frac{a_{zx}}{a_{zy}}, \qquad (11)
$$

where a_{zx} and a_{zy} are taken from Eqn. (5):

$$a_{zx} = \cos\alpha\cos\phi\sin\beta + \sin\alpha\sin\beta\sin\phi \tag{12}$$
$$a_{zy} = \cos\beta\sin\theta - \cos\alpha\sin\beta\cos\theta\sin\phi + \cos\phi\sin\alpha\sin\beta\cos\theta. \tag{13}$$

The angle φ_0 is then

$$\varphi_0 = \operatorname{arctan2}(-a_{zx}, a_{zy}). \tag{14}$$

Here we used the $\operatorname{arctan2}(x, y)$ function which is defined as

$$\operatorname{arctan2}(x,y) = \begin{cases} \arctan\left(\frac{x}{y}\right) & y > 0 \\ \arctan\left(\frac{x}{y}\right) + \pi & y < 0, x > 0 \\ \pi & y < 0, x = 0 \\ \arctan\left(\frac{x}{y}\right) - \pi & y < 0, x < 0 \\ \frac{\pi}{2} & y = 0, x > 0 \\ -\frac{\pi}{2} & y = 0, x < 0 \end{cases} \tag{15}$$

Front and Back Side Arcs of General Sphere Circles

A great circle in the rotated xy drawing plane is, in parametric form,

$$\begin{pmatrix} x(\varphi) \\ y(\varphi) \\ z(\varphi) \end{pmatrix} = \begin{pmatrix} r\cos\varphi \\ r\sin\varphi \\ 0 \end{pmatrix}. \tag{16}$$

For an arbitrary circle at an elevation ϵ from the rotated drawing we would have get a radius $r_e = r\cos\epsilon$ and a z-coordinate $z_e = r\sin\epsilon$. The parametric form is then

$$\begin{pmatrix} x(\varphi) \\ y(\varphi) \\ z(\varphi) \end{pmatrix} = \begin{pmatrix} r_e\cos\varphi \\ r_e\sin\varphi \\ z_e \end{pmatrix} = \begin{pmatrix} r\cos\epsilon\cos\varphi \\ r\cos\epsilon\sin\varphi \\ r\sin\epsilon \end{pmatrix}. \tag{17}$$

Fig. 1 shows an illustration.

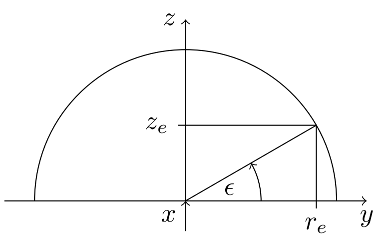


Figure 1: Illustration of z-coordinate and radius of an elevated circle on a sphere

The respective screen coordinates are

$$\begin{aligned} \begin{pmatrix} x'(\varphi) \\ y'(\varphi) \\ z'(\varphi) \end{pmatrix} &= A\begin{pmatrix} x(\varphi) \\ y(\varphi) \\ z(\varphi) \end{pmatrix} = \begin{pmatrix} a_{xx} & a_{xy} & a_{xz} \\ a_{yx} & a_{yy} & a_{yz} \\ a_{zx} & a_{zy} & a_{zz} \end{pmatrix}\begin{pmatrix} r\cos\epsilon\cos\varphi \\ r\cos\epsilon\sin\varphi \\ r\sin\epsilon \end{pmatrix} \\ &= \begin{pmatrix} a_{xx}\cdot r\cos\epsilon\cos\varphi + a_{xy}\cdot r\cos\epsilon\sin\varphi + a_{xz}\cdot r\sin\epsilon \\ a_{yx}\cdot r\cos\epsilon\cos\varphi + a_{yy}\cdot r\cos\epsilon\sin\varphi + a_{yz}\cdot r\sin\epsilon \\ a_{zx}\cdot r\cos\epsilon\cos\varphi + a_{zy}\cdot r\cos\epsilon\sin\varphi + a_{zz}\cdot r\sin\epsilon \end{pmatrix}. \end{aligned} \tag{18}$$

Again, we determine the angle φ_0 where $z'(\varphi_0) = 0$ by solving

$$0 \stackrel{!}{=} z'(\varphi_0) = a_{zx}\cdot r\cos\epsilon\cos\varphi_0 + a_{zy}\cdot r\cos\epsilon\sin\varphi_0 + a_{zz}\cdot r\sin\epsilon. \tag{19}$$

I frankly admit that I was too lazy to puzzle this out myself ;-) Matlab says

$$\tan\left(\frac{\varphi_0}{2}\right) = \frac{a_{zy}\cos\epsilon \pm \sqrt{a_{zx}^2\cos^2\epsilon + a_{zy}^2\cos^2\epsilon - a_{zz}^2\sin^2\epsilon}}{a_{zx}\cos\epsilon - a_{zz}\sin\epsilon} \tag{20}$$

$$= \frac{a_{zy} \pm \sqrt{a_{zx}^2 + a_{zy}^2 - a_{zz}^2\tan^2\epsilon}}{a_{zx} - a_{zz}\tan\epsilon}, \tag{21}$$

where

$$a_{zz}^2\sin^2\epsilon \geq (a_{zx}^2 + a_{zy}^2)\cos^2\epsilon \quad \rightsquigarrow \quad \tan^2\epsilon \geq \frac{a_{zx}^2 + a_{zy}^2}{a_{zz}^2} \tag{22}$$

must hold. With the substitutions

$$u = a_{zy}, \tag{23}$$

$$v = \sqrt{a_{zx}^2 + a_{zy}^2 - a_{zz}^2\tan^2\epsilon} \quad \text{and} \tag{24}$$

$$w = a_{zx} - a_{zz}\tan\epsilon \tag{25}$$

we get

$$\tan\left(\frac{\varphi_0}{2}\right) = \frac{u \pm v}{w} \quad \rightsquigarrow \quad \varphi_0 = \begin{cases} 2\arctan 2(u+v, w) \\ 2\arctan 2(u-v, w) \end{cases} \tag{26}$$

3.2 The Package Source Code

```
%% == LaTeX PACKAGE tikz-3dplot-circleofsphere ================================
%%    Drawing circles of a sphere with tikz-3dplot
%%
%% Matthias Wolff, BTU Cottbus-Sentenberg
%% July 26, 2018
%%
%% References:
%% [1] J. Hein. The tikz-3dplot package. 2012. Online, retrieved July 20, 2018.
%%     https://mirror.hmc.edu/ctan/graphics/pgf/contrib/tikz-3dplot/tikz-3dplot_documentation.pdf
%% [2] T. Tantau. TikZ & PGF - Manual for Version 3.0.1a. 2015. Online, retrieved July 22, 2018.
%%     https://mirror.reismil.ch/CTAN/graphics/pgf/base/doc/pgfmanual.pdf
%% [3] Drawing Great Circles
%%     https://tex.stackexchange.com/questions/168521/spherical-triangles-and-great-circles

%% == REQUIRED PACKAGES ======================================================

\RequirePackage{xifthen}
\RequirePackage{tikz}
\RequirePackage{tikz-3dplot}

%% == TikZ STYLES ============================================================

\tikzset{
  tdplotCsFront/.style={solid},
  tdplotCsBack/.style={dashed},
  tdplotCsFill/.style={opacity=0},
  tdplotPtFront/.style={},
  tdplotPtBack/.style={},
  tdplotCsDrawAux/.style={}
}

%% == COMMANDS ===============================================================

\newcommand{\tdplotCsComputeTransformRotScreen}{%
  % Computes the elements of the full rotation matrix
  %
  %   A = [\axx \axy \axz]
  %       [\ayx \ayy \ayz]
  %       [\azx \azy \azz].
```

```
  %
  % Ouput:
  %   \axx - Element A(1,1) of rotation matrix
  %   \axy - Element A(1,2) of rotation matrix
  %   ...
  %   \azz - Element A(3,3) of rotation matrix
  %
  \let\a\tdplotalpha
  \let\b\tdplotbeta
  \let\p\tdplotmainphi
  \let\t\tdplotmaintheta
  % Row 1: [\axx \axy \axz]
  \pgfmathsetmacro\axx{cos(\a)*cos(\b)*cos(\p) + cos(\b)*sin(\a)*sin(\p)}
  \pgfmathsetmacro\axy{cos(\a)*sin(\p) - cos(\p)*sin(\a)}
  \pgfmathsetmacro\axz{cos(\a)*cos(\p)*sin(\b) + sin(\a)*sin(\b)*sin(\p)}
  % Row 2: [\ayx \ayy \ayz]
  \pgfmathsetmacro\ayx{cos(\b)*cos(\p)*sin(\a)*cos(\t) - cos(\a)*cos(\b)*cos(\t)*sin(\p) - sin(\b)*sin(\t)}
  \pgfmathsetmacro\ayy{cos(\a)*cos(\p)*cos(\t) + sin(\a)*cos(\t)*sin(\p)}
  \pgfmathsetmacro\ayz{cos(\b)*sin(\t) - cos(\a)*sin(\b)*cos(\t)*sin(\p) + cos(\p)*sin(\a)*sin(\b)*cos(\t)}
  % Row 3: [\azx \azy \azz]
  \pgfmathsetmacro\azx{cos(\a)*cos(\b)*sin(\p)*sin(\t) - sin(\b)*cos(\t) - cos(\b)*cos(\p)*sin(\a)*sin(\t)}
  \pgfmathsetmacro\azy{-cos(\a)*cos(\p)*sin(\t) - sin(\a)*sin(\p)*sin(\t)}
  \pgfmathsetmacro\azz{cos(\b)*cos(\t) + cos(\a)*sin(\b)*sin(\p)*sin(\t) - cos(\p)*sin(\a)*sin(\b)*sin(\t)}
}

\newcommand{\tdplotCsDrawCircleOfSphere}[5][]{%
  % Draws a circle of a sphere.
  %
  % Input:
  %   #1 - TikZ style
  %        - use tdplotCsFront/.style={blub} to style the visible semicircle
  %        - use tdplotCsBack/.style={blah} to style the invisible semicircle
  %        - use tdplotCsFill/.style={foo} to style the fill of the circle
  %        - use tdplotCsDrawAux to draw some auxiliary information
  %   #2 - Radius of sphere
  %   #3 - Azimutal angle of drawing plane 1)
  %   #4 - Polar angle of drawing plane 2)
  %   #5 - Elevation angle of circle above the drawing plane. Permissible
  %        values are -90 < #5 < 90. Use 0 for drawing a great circle.
  %
  % Ouput:
  %   none
  %
  % Footnotes:
  %   1) passed as alpha to \tdplotsetrotatedcoords{alpha}{beta}{gamma}
  %   2) passed as beta to \tdplotsetrotatedcoords{alpha}{beta}{gamma}
  \begin{scope}[#1]                                                    % Macro scope >>
    % Do some computation                                              #   ----------------------------------
    \pgfmathsetmacro\R   {#2}                                          %   Parse radius
    \pgfmathsetmacro\aAlp{#3}                                          %   Parse azimuthal angle (alpha)
    \pgfmathsetmacro\aBet{#4}                                          %   Parse polar angle (beta)
    \pgfmathsetmacro\aEps{#5}                                          %   Parse elevation angle (epsilon)
    \pgfmathsetmacro\Re  {\R*cos(\aEps)}                               %   Radius of circle
    \pgfmathsetmacro\ze  {\R*sin(\aEps)}                               %   z-coordinate of drawing plane
    \pgfmathsetmacro\coX {\ze*cos(\aAlp)*sin(\aBet)}                   %   x-coordinate offset for ze
    \pgfmathsetmacro\coY {\ze*sin(\aAlp)*sin(\aBet)}                   %   y-coordinate offset for ze
    \pgfmathsetmacro\coZ {\ze*cos(\aBet)}                              %   z-coordinate offset for ze
    \coordinate (coffs) at (\coX,\coY,\coZ);                           %   Offset as coordinate value
    % Rotate and offset coordinate system                              %   ----------------------------------
    \tdplotsetrotatedcoords{\aAlp}{\aBet}{0}                           %   Rotate coordinate system
    \tdplotsetrotatedcoordsorigin{(coffs)}                             %   Offset coordinate system
    % Draw                                                             %   ----------------------------------
    \begin{scope}[tdplot_rotated_coords]                               %   Drawing scope >>
      \tdplotCsComputeTransformRotScreen                               %     Compute full rotation matrix
      \pgfmathsetmacro\tanEps{tan(\aEps)}                              %     Tangent of elevation angle
      \pgfmathsetmacro\bOneside{((\tanEps)^2)>=(((\azx)^2+(\azy)^2)/(\azz)^2)} %     Circle entirely on one side?
      \fill[tdplotCsFill] (0,0) circle (\Re);                          %     Draw fill of circle
      \ifthenelse{\bOneside=1}{                                        %     Circle on one side of sphere >>
        \pgfmathsetmacro\bFrontside{(\azx*\Re+\azz*\ze)>=0}            %       Circle entirely on front side?
         \ifthenelse{\bFrontside=1}                                    %       |
          {\draw[tdplotCsFront] (0,0) circle (\Re);}                   %       Draw on front side
          {\draw[tdplotCsBack]  (0,0) circle (\Re);}                   %       Draw on back side
```

```
      }{                                                                    %     << Circle on both sides >>
        \pgfmathsetmacro\u{\azy}                                            %       Substitution u=...
        \pgfmathsetmacro\v{sqrt( (\azx)^2 + (\azy)^2 - (\azz)^2*(\tanEps)^2 )} %    Substitution v=...
        \pgfmathsetmacro\w{\azx - \azz*\tanEps}                             %       Substitution w=...
        \pgfmathsetmacro\aPhiBf{2*atan2(\u-\v,\w)}                          %       Back->front crossing angle
        \pgfmathsetmacro\aPhiFb{2*atan2(\u+\v,\w)}                          %       Front->back crossing angle
        \pgfmathsetmacro\bUnwrapA{(\aPhiFb-\aPhiBf)>360}                    %       Unwrap front->back angle #1?
        \pgfmathsetmacro\bUnwrapB{\aPhiBf>\aPhiFb}                          %       Unwrap front->back angle #2?
        \ifthenelse{\bUnwrapA=1}{\pgfmathsetmacro\aPhiBf{\aPhiBf+360}}{}   %       Unwrap front->back angle #1
        \ifthenelse{\bUnwrapB=1}{\pgfmathsetmacro\aPhiBf{\aPhiBf-360}}{}   %       Unwrap front->back angle #2
        \draw[tdplotCsBack]  (\aPhiFb:\Re) arc (\aPhiFb:{\aPhiBf+360}:\Re); %      Draw back side arc
        \draw[tdplotCsFront] (\aPhiBf:\Re) arc (\aPhiBf:\aPhiFb:\Re);       %       Draw back side arc
      }                                                                     %     <<
      % Auxliliary drawing (for debugging and illustration)                 %     - - - - - - - - - - - - - - - - -
      \ifthenelse{\isin{tdplotCsDrawAux}{#1}}{                              %     Auxiliary drawing activated >>
        \draw[red!40,->] (-\Re,0,0) -- (\Re,0,0) node[anchor=north] {$x_d$}; %     x-axis of drawing corrd. system
        \draw[red!40,->] (0,-\Re,0) -- (0,\Re,0) node[anchor=north] {$y_d$}; %     y-axis of drawing corrd. system
        \draw[red!40,->] (0,0,0)    -- (0,0,\Re) node[anchor=north] {$z_d$}; %     z-axis of drawing corrd. system
        \ifthenelse{\bOneside=0}{                                           %       Circ.on both sides of sphere >>
          \node[red] at (\aPhiBf:\Re) {$\circ$};                            %         Indicate back-front crossing
          \node[red] at (\aPhiFb:\Re) {$\times$};                           %         Indicate front-back crossing
        }{}                                                                 %       <<
        \coordinate (coffs) at (-\coX,-\coY,-\coZ);                         %       HACK: Forcibly reset ...
        \tdplotsetrotatedcoordsorigin{(coffs)}                              %       ... coordinate system
        \begin{scope}[tdplot_rotated_coords]                                %       Aux. display scope >>
          \node[tdplot_screen_coords,red,anchor=north west] at (0.7*\R,-0.9*\R) % Make a litte display ...
            {\parbox{200pt}{\footnotesize                                   %         ... >>
              $\theta=\tdplotmaintheta^\circ, \phi=\tdplotmainphi^\circ$\\  %           Main coord. sys. parameters
              $\alpha=\aAlp^\circ, \beta=\aBet^\circ,                       %           Rot. coord. sys. parameters
               \epsilon\!=\!\aEps^\circ\!$\\                                %           Drawing plane elev. angle
              $a_{zx}=\azx, a_{zy}=\azy, a_{zz}=\azz$\\                     %           Elems. of full rot. matrix
              $r_e\!=\!\Re, z_e\!=\!\ze$\\                                  %           Radius and z-elevation
              $\texttt{\textbackslash bOneside}\!=\!\bOneside$,             %           One-side circle flag
              \ifthenelse{\bOneside=1}{                                     %           One-side circle >>
                $\texttt{\textbackslash bFrontside}\!=\!\bFrontside$\\      %             Front-side flag
              }{                                                            %           << Two-side circle >>
                $\texttt{\textbackslash bUnwrapA}\!=\!\bUnwrapA$,           %             Angle unwrap flag #1
                $\texttt{\textbackslash bUnwrapB}\!=\!\bUnwrapB$\\          %             Angle unwrap flag #2
                $\circ\!: \!\texttt{\textbackslash aPhiBf}\!=\!\aPhiBf^\circ\!, % Back-front crossing angle
                 \times\!:\!\texttt{\textbackslash aPhiFb}\!=\!\aPhiFb^\circ$\\ % Front-back crossing angle
              }                                                             %           <<
            }};                                                             %         <<
        \end{scope}                                                         %       << (Aux. display scope)
      }{}                                                                   %     << (Auxiliary drawing activated)
    \end{scope}                                                             %   << (Drawing scope)
  \end{scope}                                                               % << (Macro scope)
}

\newcommand{\tdplotCsDrawGreatCircle}[4][]{%
  % TODO: ...
  \tdplotCsDrawCircleOfSphere[#1]{#2}{#3}{#4}{0}
}

\newcommand{\tdplotCsDrawLatCircle}[3][]{%
  % TODO: ...
  \tdplotCsDrawCircleOfSphere[#1]{#2}{0}{0}{#3}
}

\newcommand{\tdplotCsDrawLonCircle}[3][]{%
  % TODO: ...
  \tdplotCsDrawCircleOfSphere[#1]{#2}{{#3+90}}{90}{0}
}

%% == EOF ====================================================================
```

3.3 An Auxiliary Matlab Script

```
%% == LaTeX PACKAGE tikz-3dplot-circleofsphere ================================
%     Drawing circles of a sphere with tikz-3dplot
%
%  Matthias Wolff, BTU Cottbus-Sentenberg
%  July 26, 2018
```

```
%
%  References:
%  [1] J. Hein. The tikz-3dplot package. 2012. Online, retrieved July 20, 2018.
%      https://mirror.hmc.edu/ctan/graphics/pgf/contrib/tikz-3dplot/tikz-3dplot_documentation.pdf
%

%% Rotation matrices ========================================================
syms a b p t

% R rotation matrix ---------------------------------------------------------
Rz = [  cos(p) -sin(p)   0
        sin(p)  cos(p)   0
        0       0        1      ];

Rx = [  1       0        0
        0       cos(t)  -sin(t)
        0       sin(t)   cos(t) ];

% - [1] eq. (2.1) line 2
% R = Rz*Rx; disp(R);

% - [1] eq. (2.1) line 3
% R = [  cos(p)         sin(p)         0
%       -cos(t)*sin(p)  cos(t)*cos(p) -sin(t)
%        sin(t)*sin(p) -sin(t)*cos(p)  cos(t) ];

% - [1] eq. (2.1) line 3, corrected
R = (Rz*Rx).';

% -- D rotation matrix ------------------------------------------------------
Dz = [  cos(a) -sin(a)  0
        sin(a)  cos(a)  0
        0       0       1      ];

Dy = [  cos(b)  0       sin(b)
        0       1       0
       -sin(b)  0       cos(b) ];

Dx = [  1       0       0
        0       cos(b) -sin(b)
        0       sin(b)  cos(b) ];

D = Dz*Dy; disp(D);

% -- Full rotation matrix ---------------------------------------------------
A = R*D; disp(A);
axx = A(1,1); axy = A(1,2); axz = A(1,3);
ayx = A(2,1); ayy = A(2,2); ayz = A(2,3);
azx = A(3,1); azy = A(3,2); azz = A(3,3);

%% == Transform a vector (world -> screen) ==================================
syms x y z
p = [ x
      y
      z ];
q=A*p;
disp(q);

%% == View angle ============================================================
syms p0 r eps azx azy azz
assume(p0,'real');
assume(r,'real');
assume(eps,'real');
assume(azx,'real');
assume(azy,'real');
assume(azz,'real');
eqn = azx*r*cos(eps)*cos(p0) + azy*r*cos(eps)*sin(p0) + azz*r*sin(eps) == 0
solve(eqn,p0,'Real',true)

% syms p0 u v w
% assume(p0,'real');
% assume(u,'real');
```

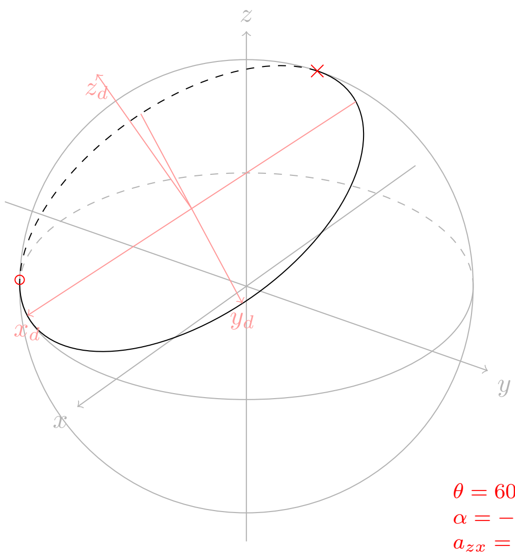
```
% assume(v,'real');
% assume(w,'real');
% eqn = u*cos(p0) + v*sin(p0) + w == 0;
% solve(eqn,p0,'Real',true)

%% == EOF =====================================================================
```

References

[1] Jeff Hein. The tikz-3dplot package. http://mirror.ctan.org/graphics/pgf/contrib/tikz-3dplot/tikz-3dplot_documentation.pdf, 2012. Retrieved: July 27, 2018.

[2] Till Tantau. Tikz & pgf - manual for version 3.0.1a. http://mirror.ctan.org/graphics/pgf/base/doc/pgfmanual.pdf, 2015. Retrieved: July 27, 2018.



$\theta = 60.0°, \phi = 125.0°$
$\alpha = -40.0°, \beta = 30°, \epsilon = 30°$
$a_{zx} = -0.05588, a_{zy} = 0.8365, a_{zz} = 0.54507$
$r_e = 2.59808, z_e = 1.5$
$\texttt{\textbackslash bOneside} = 0,\ \texttt{\textbackslash bUnwrapA} = 0,\ \texttt{\textbackslash bUnwrapB} = 1$
$\circ : \texttt{\textbackslash aPhiBf} = -18.22858°, \times : \texttt{\textbackslash aPhiFb} = 205.86197°$

```latex
\documentclass{standalone}
\usepackage[dvipsnames]{xcolor}
\usepackage{tikz-3dplot-circleofsphere}

\begin{document}

  \def\elev{  30} \pgfmathsetmacro{\tdpTheta}{90-\elev}
  \def\azim{  35} \pgfmathsetmacro{\tdpPhi}{90+\azim}
  \def\R{3}
  \tdplotsetmaincoords{\tdpTheta}{\tdpPhi}
  \begin{tikzpicture}[scale=1,tdplot_main_coords]
    \begin{scope}[black!30,name=auxiliary]
      \draw[tdplot_screen_coords] (0,0,0) circle (\R);
      \draw[->] (-1.3*\R,0,0) -- (1.3*\R,0,0) node[anchor=north east]{$x$};
      \draw[->] (0,-1.3*\R,0) -- (0,1.3*\R,0) node[anchor=north west]{$y$};
      \draw[->] (0,0,-1.3*\R) -- (0,0,1.3*\R) node[anchor=south]{$z$};
      \tdplotCsDrawCircleOfSphere{\R}{0}{0}{0};
    \end{scope}
    \begin{scope}
%      \tdplotCsDrawLatCircle[tdplotCsDrawAux]{\R}{-30}
      % --
      \tdplotCsDrawCircleOfSphere[tdplotCsDrawAux]{\R}{-40}{30}{30}
      % --
%       \foreach \a in {0,15,...,345}
%         { \tdplotCsDrawCircleOfSphere[very thin,gray]{\R}{\a}{90}{0} }
%       \foreach \a in {-75,-60,...,75}
%         { \tdplotCsDrawCircleOfSphere[very thin,gray]{\R}{0}{0}{\a} }
      % -- Pathologic cases -->
%       \tdplotCsDrawCircleOfSphere{\R}{35}{60}{0}
      % <--
    \end{scope}
  \end{tikzpicture}

\end{document}
```