

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

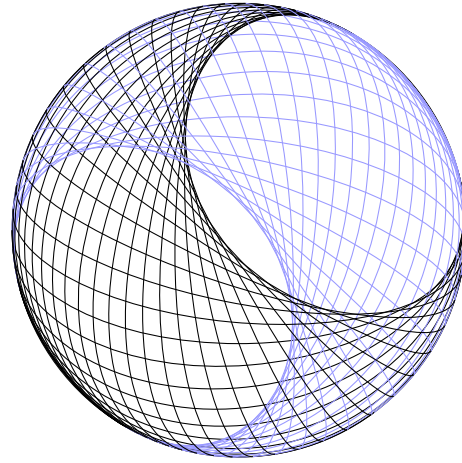
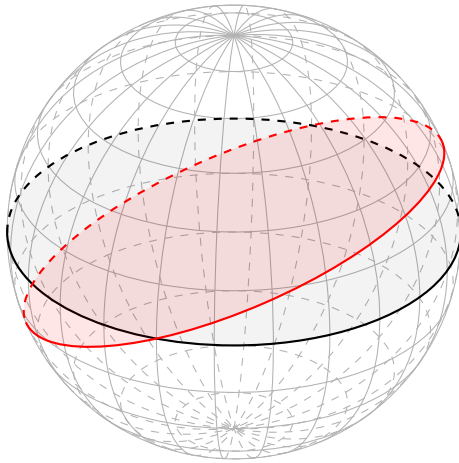
Matthias Wolff<sup>[0000-0002-3895-7313]</sup>

BTU Cottbus-Senftenberg

August 7, 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.



```

1 \documentclass{standalone}
2 \usepackage{tikz-3dplot-circleofsphere}
3 \begin{document}
4   \centering
5   \def\r{3}
6   \tdplotsetmaincoords{60}{125}
7   \begin{tikzpicture}[tdplot_main_coords]
8     \draw[tdplot_screen_coords,thin,black!30] (0,0,0) circle (\r);
9     \foreach \a in {-75,-60,...,75}
10      {\tdplotCsDrawLatCircle[thin,black!30]{\r}{\a}}
11     \foreach \a in {0,15,...,165}
12      {\tdplotCsDrawLonCircle[thin,black!30]{\r}{\a}}
13     \tdplotCsDrawLatCircle%
14       [thick,tdplotCsFill/.style={opacity=0.05}]{\r}{0}
15     \tdplotCsDrawGreatCircle%
16       [red,thick,tdplotCsFill/.style={opacity=0.1}]{\r}{105}{-23.5}
17   \end{tikzpicture}
18 \end{document}

```

```

1 \documentclass{standalone}
2 \usepackage{tikz-3dplot-circleofsphere}
3 \begin{document}
4   \centering
5   \def\r{3}
6   \tdplotsetmaincoords{60}{125}
7   \begin{tikzpicture}[tdplot_main_coords]
8     \def\epsilon{80};
9     \draw[tdplot_screen_coords,very thin] (0,0,0) circle (\r);
10    \foreach \a in {0,5,...,175} {
11      \tdplotCsDrawGreatCircle%
12        [thin,tdplotCsBack/.style={thin,blue!40}]{%
13        {\r}{\a}{90*\sin(\a)*\sin(\epsilon)}
14    }
15  \end{tikzpicture}
16 \end{document}

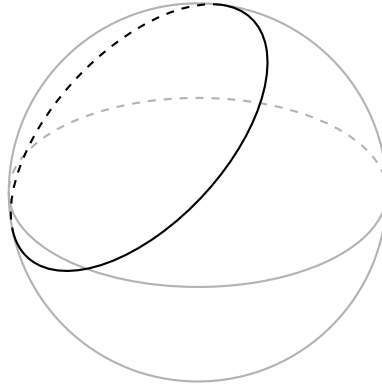
```

# Contents

<b>1</b>	<b>Just Looking for the Minimalist Code?</b>	<b>3</b>
<b>2</b>	<b>The <code>tikz-3dplot-circleofsphere</code> Package</b>	<b>4</b>
2.1	Installation . . . . .	4
2.2	Drawing Commands . . . . .	4
	<code>\tdplotCsDrawCircle[style]{r}{alpha}{beta}{epsilon}</code> . . . . .	4
	<code>\tdplotCsDrawGreatCircle[style]{r}{alpha}{beta}</code> . . . . .	5
	<code>\tdplotCsDrawLatCircle[style]{r}{epsilon}</code> . . . . .	5
	<code>\tdplotCsDrawLonCircle[style]{r}{alpha}</code> . . . . .	6
	<code>\tdplotCsDrawPoint[style]{r}{alpha}{beta}</code> . . . . .	7
2.3	Auxiliary Commands . . . . .	7
	<code>\tdplotCsFrontsidePoint</code> . . . . .	7
	<code>\tdplotCsBacksidePoint</code> . . . . .	7
	<code>\tdplotCsComputeTransformRotScreen</code> . . . . .	7
2.4	Known Issues . . . . .	8
<b>3</b>	<b>Implementation Details</b>	<b>8</b>
3.1	The Maths . . . . .	8
	Circles on a Sphere . . . . .	8
	Coordinate Transforms with <code>tikz-3dplot</code> . . . . .	8
	Drawing Circles of a Sphere . . . . .	11
3.2	The Package Source Code . . . . .	12
3.3	An Auxiliary Matlab Script . . . . .	16
	<b>References</b>	<b>18</b>

# 1 Just Looking for the Minimalist Code?

There you go!



```

1 \documentclass{standalone}
2 \usepackage{tikz,tikz-3dplot}
3 %% >> MINIMALIST CIRCLE OF SHPERE DRAWING CODE -----
4 \newcommand\scircle[4]{%
5   \tdplotsetrotatedcoords{#2}{#3}{0} % Rotate coordinate system
6   \let\alpha\tdplotalpha % alpha (rotated coord. system)
7   \let\beta\tdplotbeta % beta (rotated coord. system)
8   \let\phi\tdplotmainphi % phi (main coord. system)
9   \let\theta\tdplotmaintheta % theta (main coord. system)
10  \pgfmathsetmacro\azx{\cos(\alpha)*\cos(\beta)*\sin(\phi)*\sin(\theta) - \sin(\beta)*\cos(\theta) - \cos(\beta)*\cos(\phi)*\sin(\alpha)*\sin(\theta)}
11  \pgfmathsetmacro\azy{-\cos(\alpha)*\cos(\beta)*\sin(\phi)*\sin(\theta) - \sin(\alpha)*\sin(\phi)*\sin(\theta)}
12  \pgfmathsetmacro\azz{\cos(\beta)*\cos(\theta) + \cos(\alpha)*\sin(\beta)*\sin(\phi)*\sin(\theta) - \cos(\phi)*\sin(\alpha)*\sin(\beta)*\sin(\theta)}
13  \pgfmathsetmacro\re{#1*\cos(#4)} % Radius of circle
14  \pgfmathsetmacro\ze{#1*\sin(#4)} % z-coordinate of drawing plane
15  \pgfmathsetmacro\coX{\ze*\cos(#2)*\sin(#3)} % x-coordinate offset for ze
16  \pgfmathsetmacro\coY{\ze*\sin(#2)*\sin(#3)} % y-coordinate offset for ze
17  \pgfmathsetmacro\coZ{\ze*\cos(#3)} % z-coordinate offset for ze
18  \coordinate (coffs) at (\coX,\coY,\coZ); % Offset as coordinate value
19  \tdplotsetrotatedcoordsorigin{(coffs)} % Offset coordinate system
20  \begin{scope}[tdplot_rotated_coords] % Drawing scope >>
21    \pgfmathsetmacro\tanEps{tan(#4)} % Tangent of elevation angle
22    \pgfmathsetmacro\bOneside{((\tanEps)^2)>=((\azx)^2+(\azy)^2)/(\azz)^2)} % Circle entirely on one side?
23    \ifthenelse{\bOneside=1}{% % Circle on one side of sphere >>
24      \pgfmathsetmacro\bFrontside{(\azx*\re+\azz*\ze)>=0} % Circle entirely on front side?
25      \ifthenelse{\bFrontside=1}{%
26        \draw (0,0) circle (\re); % Draw on front side
27        \draw[dashed] (0,0) circle (\re); % Draw on back side
28      }{% << Circle on both sides >>
29        \pgfmathsetmacro\u{\azy} % Substitution u=...
30        \pgfmathsetmacro\v{\sqrt{(\azx)^2 + (\azy)^2 - (\azz)^2*(\tanEps)^2}} % Substitution v=...
31        \pgfmathsetmacro\w{\azx - \azz*\tanEps} % Substitution w=...
32        \pgfmathsetmacro\phiBf{2*atan2(\u-\v,\w)} % Back->front crossing angle
33        \pgfmathsetmacro\phiFb{2*atan2(\u+\v,\w)} % Front->back crossing angle
34        \pgfmathsetmacro\bUnwrapA{\phiFb-\phiBf}>360} % Unwrap front->back angle #1?
35        \pgfmathsetmacro\bUnwrapB{\phiBf>\phiFb} % Unwrap front->back angle #2?
36        \ifthenelse{\bUnwrapA=1}{\pgfmathsetmacro\phiBf{\phiBf+360}}{} % Unwrap front->back angle #1
37        \ifthenelse{\bUnwrapB=1}{\pgfmathsetmacro\phiBf{\phiBf-360}}{} % Unwrap front->back angle #2
38        \draw[dashed] (\phiFb:\re) arc (\phiFb:\phiBf+360:\re); % Draw back side arc
39        \draw (\phiBf:\re) arc (\phiBf:\phiFb:\re); % Draw back side arc
40      } % <<
41    \end{scope} % << (Drawing scope)
42 }
43 %% << -----
44 \begin{document}
45 \tdplotsetmaincoords{60}{125} % Set main coordinatate system
46 \begin{tikzpicture}[thick,tdplot_main_coords] % TikZ picture >>
47   \begin{scope}[black!30] % Draw in gray >>
48     \draw[tdplot_screen_coords] (0,0,0) circle (2.5); % Sphere outline
49     \scircle{2.5}{0}{0}{0} % Equator
50   \end{scope} % <<
51   \scircle{2.5}{-40}{40}{30} % Draw another sphere circle
52 \end{tikzpicture} % <<
53 \end{document}

```

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

## 2 The tikz-3dplot-circleofsphere Package

### 2.1 Installation

Download `tikz-3dplot-circleofsphere.sty` from [3] file into your project folder and include the package with `\usepackage{tikz-3dplot-circleofsphere}`.

### 2.2 Drawing Commands

```
\tdplotCsDrawCircle[style]{r}{alpha}{beta}{epsilon}
```

Draws a circle of a sphere.

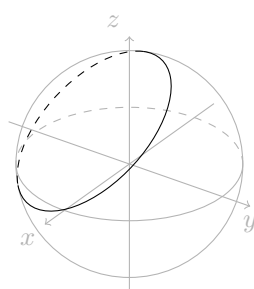
#### Parameters

<code>style</code>	TikZ style
	<ul style="list-style-type: none"> <li>• use <code>\tdplotCsFront/.style={...}</code> to style the front side arc</li> <li>• use <code>\tdplotCsBack/.style={...}</code> to style the back side arc</li> <li>• use <code>\tdplotCsFill/.style={...}</code> to style the circle filling</li> <li>• use <code>\tdplotCsDrawAux</code> to draw some auxiliary information</li> </ul>
<code>r</code>	Radius of sphere
<code>alpha</code>	Azimuthal angle of drawing plane. Passed as <code>alpha</code> to <code>\tdplotsetrotatedcoords{alpha}{beta}{gamma}</code>
<code>beta</code>	Polar angle of drawing plane. Passed as <code>beta</code> to <code>\tdplotsetrotatedcoords{alpha}{beta}{gamma}</code>
<code>epsilon</code>	Elevation angle of circle above the drawing plane. Permissible values are $-90 < \epsilon < 90$ . Use 0 for drawing a great circle.

#### Output

—none—

#### Example



```

1 \def\r{1.5}
2 \tdplotsetmaincoords{60}{125}
3 \begin{tikzpicture}[tdplot_main_coords]
4   \begin{scope}[thin,black!30]
5     \draw[>-] (-1.3*\r,0,0) -- (1.3*\r,0,0) node[anchor=north east] {$x$};
6     \draw[>-] (0,-1.3*\r,0) -- (0,1.3*\r,0) node[anchor=north] {$y$};
7     \draw[>-] (0,0,-1.3*\r) -- (0,0,1.3*\r) node[anchor=south east] {$z$};
8     \draw[tdplot_screen_coords] (0,0,0) circle (\r);
9     \tdplotCsDrawLatCircle{\r}{0}
10  \end{scope}
11  \tdplotCsDrawCircle{\r}{-40}{40}{30}
12 \end{tikzpicture}

```

```
\tdplotCsDrawGreatCircle[style]{r}{alpha}{beta}
```

Draws a great circle.

Equivalent to `\tdplotCsDrawCircle[style]{r}{alpha}{beta}{0}`.

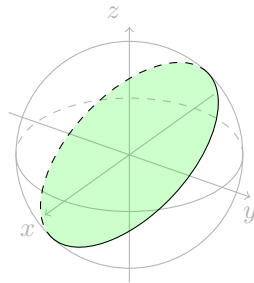
### Parameters

<code>style</code>	TikZ style <ul style="list-style-type: none"> <li>• use <code>\tdplotCsFront/.style={...}</code> to style the front side arc</li> <li>• use <code>\tdplotCsBack/.style={...}</code> to style the back side arc</li> <li>• use <code>\tdplotCsFill/.style={...}</code> to style the circle filling</li> <li>• use <code>\tdplotCsDrawAux</code> to draw some auxiliary information</li> </ul>
<code>r</code>	Radius of sphere
<code>alpha</code>	Azimuthal angle of drawing plane. Passed as <code>alpha</code> to <code>\tdplotsetrotatedcoords{alpha}{beta}{gamma}</code>
<code>beta</code>	Polar angle of drawing plane. Passed as <code>beta</code> to <code>\tdplotsetrotatedcoords{alpha}{beta}{gamma}</code>

### Output

—none—

### Example



```

1 \def\r{1.5}
2 \tdplotsetmaincoords{60}{125}
3 \begin{tikzpicture}[tdplot_main_coords]
4   \begin{scope}[thin,black!30]
5     \draw[>-] (-1.3*\r,0,0) -- (1.3*\r,0,0) node[anchor=north east] {$x$};
6     \draw[>-] (0,-1.3*\r,0) -- (0,1.3*\r,0) node[anchor=north] {$y$};
7     \draw[>-] (0,0,-1.3*\r) -- (0,0,1.3*\r) node[anchor=south east] {$z$};
8     \draw[tdplot_screen_coords] (0,0,0) circle (\r);
9     \tdplotCsDrawLatCircle{\r}{0}
10    \end{scope}
11    \tdplotCsDrawGreatCircle[tdplotCsFill/.style={green,opacity=0.2}]{\r}{-40}{40}
12 \end{tikzpicture}

```

```
\tdplotCsDrawLatCircle[style]{r}{epsilon}
```

Draws a circle of latitude.

Equivalent to `\tdplotCsDrawCircle[style]{r}{0}{0}{epsilon}`.

### Parameters

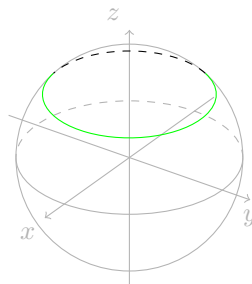
<code>style</code>	TikZ style <ul style="list-style-type: none"> <li>• use <code>\tdplotCsFront/.style={...}</code> to style the front side arc</li> <li>• use <code>\tdplotCsBack/.style={...}</code> to style the back side arc</li> <li>• use <code>\tdplotCsFill/.style={...}</code> to style the circle filling</li> <li>• use <code>\tdplotCsDrawAux</code> to draw some auxiliary information</li> </ul>
<code>r</code>	Radius of sphere

**epsilon** Elevation angle of circle above the drawing plane. Permissible values are  $-90 < \text{epsilon} < 90$ . Use 0 for drawing the sphere equator.

## Output

—none—

## Example



```

1 \def\r{1.5}
2 \tdplotsetmaincoords{60}{125}
3 \begin{tikzpicture}[tdplot_main_coords]
4   \begin{scope}[thin,black!30]
5     \draw[>-] (-1.3*\r,0,0) -- (1.3*\r,0,0) node[anchor=north east] {$x$};
6     \draw[>-] (0,-1.3*\r,0) -- (0,1.3*\r,0) node[anchor=north] {$y$};
7     \draw[>-] (0,0,-1.3*\r) -- (0,0,1.3*\r) node[anchor=south east] {$z$};
8     \draw[tdplot_screen_coords] (0,0,0) circle (\r);
9     \tdplotCsDrawLatCircle{\r}{0}
10  \end{scope}
11  \tdplotCsDrawLatCircle[tdplotCsFront/.style={green}]{\r}{40}
12 \end{tikzpicture}

```

`\tdplotCsDrawLonCircle[style]{r}{alpha}`

Draws a circle of longitude.

Equivalent to `\tdplotCsDrawCircle[style]{r}{alpha}{90}{0}`.

## Parameters

**style** TikZ style

- use `tdplotCsFront/.style={...}` to style the front side arc
- use `tdplotCsBack/.style={...}` to style the back side arc
- use `tdplotCsFill/.style={...}` to style the circle filling
- use `tdplotCsDrawAux` to draw some auxiliary information

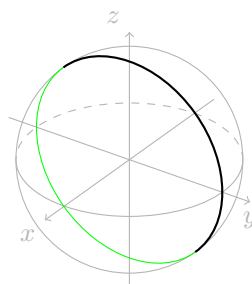
**r** Radius of sphere

**alpha** Azimuthal angle of drawing plane.  
Passed as `alpha` to `\tdplotsetrotatedcoords{alpha}{beta}{gamma}`

## Output

—none—

## Example



```

1 \def\r{1.5}
2 \tdplotsetmaincoords{60}{125}
3 \begin{tikzpicture}[tdplot_main_coords]
4   \begin{scope}[thin,black!30]
5     \draw[>-] (-1.3*\r,0,0) -- (1.3*\r,0,0) node[anchor=north east] {$x$};
6     \draw[>-] (0,-1.3*\r,0) -- (0,1.3*\r,0) node[anchor=north] {$y$};
7     \draw[>-] (0,0,-1.3*\r) -- (0,0,1.3*\r) node[anchor=south east] {$z$};
8     \draw[tdplot_screen_coords] (0,0,0) circle (\r);
9     \tdplotCsDrawLatCircle{\r}{0}
10  \end{scope}
11  \tdplotCsDrawLonCircle[thick,tdplotCsBack/.style={thin,solid,green}]{\r}{0}
12 \end{tikzpicture}

```

```
\tdplotCsDrawPoint[style]{r}{alpha}{beta}
```

Draws a point on a sphere.

### Parameters

<b>style</b>	TikZ style <ul style="list-style-type: none"> <li>• use <code>tdplotPtFront/.style={...}</code> to style a front side point</li> <li>• use <code>tdplotPtBack/.style={...}</code> to style a back side point</li> </ul>
<b>r</b>	Radius of sphere
<b>alpha</b>	Azimuthal angle of drawing plane. Passed as <code>alpha</code> to <code>\tdplotsetrotatedcoords{alpha}{beta}{gamma}</code>
<b>beta</b>	Polar angle of drawing plane. Passed as <code>beta</code> to <code>\tdplotsetrotatedcoords{alpha}{beta}{gamma}</code>

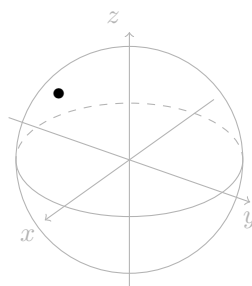
### Output

—none—

### Remarks

- Redefine `\tdplotCsFrontsidePoint` to customize drawing of a front side point.
- Redefine `\tdplotCsBacksidePoint` to customize drawing of a back side point.

### Example



```

1 \def\r{1.5}
2 \tdplotsetmaincoords{60}{125}
3 \begin{tikzpicture}[tdplot_main_coords]
4   \begin{scope}[thin,black!30]
5     \draw[>-] (-1.3*\r,0,0) -- (1.3*\r,0,0) node[anchor=north east] {$x$};
6     \draw[>-] (0,-1.3*\r,0) -- (0,1.3*\r,0) node[anchor=north] {$y$};
7     \draw[>-] (0,0,-1.3*\r) -- (0,0,1.3*\r) node[anchor=south east] {$z$};
8     \draw[tdplot_screen_coords] (0,0,0) circle (\r);
9     \tdplotCsDrawLatCircle{\r}{0}
10  \end{scope}
11  \tdplotCsDrawPoint{\r}{-40}{40}
12 \end{tikzpicture}

```

## 2.3 Auxiliary Commands

```
\tdplotCsFrontsidePoint
```

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

```
\tdplotCsBacksidePoint
```

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

```
\tdplotCsComputeTransformRotScreen
```

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 Known Issues

- The `tdplotCsFill` and `tdplotCsDrawAux` styles are only effective when specified directly with the drawing command.

# 3 Implementation Details

## 3.1 The Maths

### Circles on a Sphere

We consider circles on a sphere of radius  $r$  as illustrated in Fig. 1. For drawing a great circle, i.e. a circle whose center coincides with the center of the sphere (blue in Fig. 1), we rotate the coordinate system by two Euler angles, an azimuthal angle  $0^\circ \leq \alpha < 360^\circ$  and a polar angle  $0^\circ \leq \beta < 360^\circ$ , and draw on the new  $x_r y_r$  plane. For drawing small circles (red in Fig. 1), we additionally elevate the drawing plane by an angle  $-90^\circ < \epsilon < 90^\circ$ ,  $\epsilon \neq 0$ , and draw on the rotated and elevated  $x_{ro} y_{ro}$  plane. The tricky part of drawing circles of spheres is to determine which part is on the back side of the sphere and to draw it a different style, e.g., dashed.

### Coordinate Transforms with `tikz-3dplot`

We use the `circle` and `arc` path construction operations of TikZ for drawing. As TikZ will only draw circles and arcs on the  $xy$ -plane, we need to rotate and possibly offset the coordinate system as described above using the `tikz-3dplot` [1] package.

First, `tikz-3dplot` provides a *main coordinate system* which is basically defining the view point on a 3D coordinate system. Denote by  $P = (x y z)^\top$  a point in the 3D coordinate system. `tikz-3dplot` transforms that point in to screen coordinates  $P' = (x' y' z')^\top$  by

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



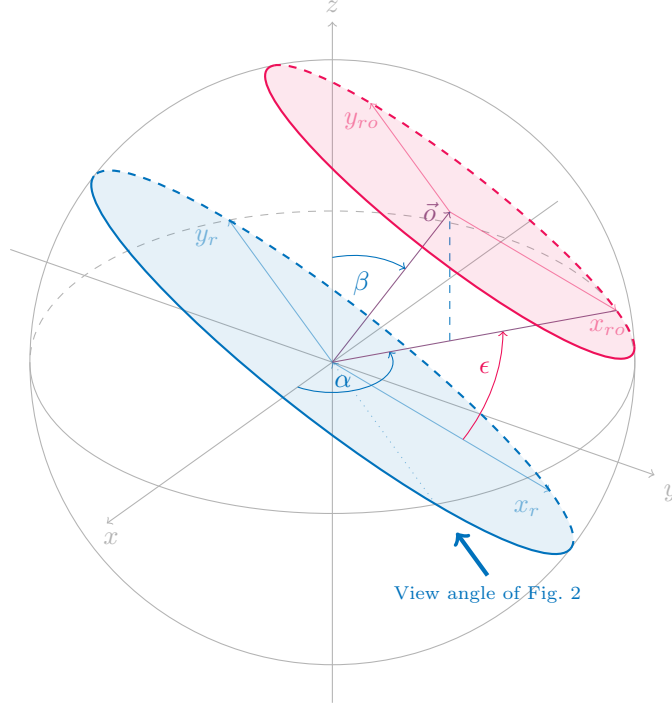


Figure 1: A great circle (blue) and a small circle (red).

with the rotation matrix<sup>1</sup>

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

We set the main coordinate system by `\tdplotsetmaincoords{\langle\phi\rangle}\{\langle\theta\rangle\}`.

Second, for drawing circles and arcs outside the  $xy$ -plane, we need to rotate the coordinate system further. To this end, we use `tikz-3dplot`'s *rotated coordinate system*<sup>2</sup>

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

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

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

<sup>1</sup>Eqn. (2.1) in [1] seems to be incorrect. I used a version with changes marked in red: Since  $(R^{z'}(\phi) R^x(\theta))^{\top} = R^x(\theta)^{\top} R^{z'}(\phi)^{\top}$ , rotations are performed in reverse order and direction.

<sup>2</sup>Eqn. (2.4) in [1] seems to be incorrect. I used a version with changes marked in red: Rotations are performed in reverse order.

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

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

The rotated coordinate system is set by `\tdplotsetrotatedcoords{⟨α⟩}{⟨β⟩}{0}`.

With the coordinate transforms described so far, we can only draw great circles. For drawing small circles, we additionally need to offset the origin of the rotated coordinate system. To this end we define an elevation angle  $\epsilon$  which defines the height

$$z_e = r \sin \epsilon \tag{6}$$

of the offset drawing plane over the rotated one as illustrated in Fig. 2 ( $r$  is still the radius of the sphere). As the circle to be drawn must lie on the sphere, its radius

$$r_e = r \cos \epsilon \tag{7}$$

decreases with increasing elevation.

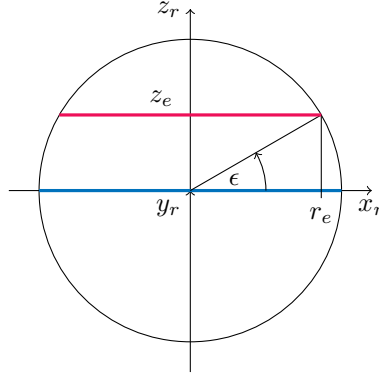


Figure 2: Illustration of  $z$ -coordinate and radius of an elevated circle on a sphere

The offset vector (see Fig. 1) is given by

$$\vec{o} = D(\alpha, \beta, 0) \begin{pmatrix} 0 \\ 0 \\ z_e \end{pmatrix} = \begin{pmatrix} z_e \cos \alpha \sin \beta \\ z_e \sin \alpha \sin \beta \\ z_e \cos \beta \end{pmatrix} = \begin{pmatrix} r \sin \epsilon \cos \alpha \sin \beta \\ r \sin \epsilon \sin \alpha \sin \beta \\ r \sin \epsilon \cos \beta \end{pmatrix} \tag{8}$$

and applied through `\tdplotsetrotatedcoordsorigin{⟨o⟩}` command of `tikz-3dplot`.

## Drawing Circles of a Sphere

The parametric representation of a circle of a sphere in the rotated and offset coordinate system is

$$\begin{pmatrix} x(\varphi) \\ y(\varphi) \\ z(\varphi) \end{pmatrix} = \begin{pmatrix} r_e \cos \varphi \\ r_e \sin \varphi \\ 0 \end{pmatrix}. \quad (9)$$

After applying the coordinate transforms described above, we could just draw this circle by

$$\backslash\text{draw (0,0) circle } \langle r_e \rangle;$$

However, as we want to visualize the parts of the circle lying on the front and back sides of the sphere, we consider the parametric representation in the rotated but not *not(!)* offset coordinate system

$$\begin{pmatrix} x(\varphi) \\ y(\varphi) \\ z(\varphi) \end{pmatrix} = \begin{pmatrix} r_e \cos \varphi \\ r_e \sin \varphi \\ z_e \end{pmatrix}. \quad (10)$$

The respective screen coordinates are

$$\begin{aligned} \begin{pmatrix} x'(\varphi) \\ y'(\varphi) \\ z'(\varphi) \end{pmatrix} &= A \begin{pmatrix} x(\varphi) \\ y(\varphi) \\ z(\varphi) \end{pmatrix} = \begin{pmatrix} a_{xx} & a_{xy} & a_{xz} \\ a_{yx} & a_{yy} & a_{yz} \\ a_{zx} & a_{zy} & a_{zz} \end{pmatrix} \begin{pmatrix} r \cos \epsilon \cos \varphi \\ r \cos \epsilon \sin \varphi \\ r \sin \epsilon \end{pmatrix} \\ &= \begin{pmatrix} a_{xx} \cdot r \cos \epsilon \cos \varphi + a_{xy} \cdot r \cos \epsilon \sin \varphi + a_{xz} \cdot r \sin \epsilon \\ a_{yx} \cdot r \cos \epsilon \cos \varphi + a_{yy} \cdot r \cos \epsilon \sin \varphi + a_{yz} \cdot r \sin \epsilon \\ a_{zx} \cdot r \cos \epsilon \cos \varphi + a_{zy} \cdot r \cos \epsilon \sin \varphi + a_{zz} \cdot r \sin \epsilon \end{pmatrix}. \end{aligned} \quad (11)$$

By examining the  $z'(\varphi)$  coordinate we can determine which parts of the circle are

$$\begin{aligned} &\text{on the front side (or "above" the drawing paper)} \quad z'(\varphi) > 0 \quad \text{and} \\ &\text{on the back side (or "below" the drawing paper)} \quad z'(\varphi) < 0 \end{aligned} \quad (12)$$

of the sphere. We denote by  $\varphi_0$  the crossing angles between the front and back sides. In order to determine them we solve

$$0 \stackrel{!}{=} z'(\varphi_0) = a_{zx} \cdot r \cos \epsilon \cos \varphi_0 + a_{zy} \cdot r \cos \epsilon \sin \varphi_0 + a_{zz} \cdot r \sin \epsilon. \quad (13)$$

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

$$\tan\left(\frac{\varphi_0}{2}\right) = \frac{a_{zy} \cos \epsilon \pm \sqrt{a_{zx}^2 \cos^2 \epsilon + a_{zy}^2 \cos^2 \epsilon - a_{zz}^2 \sin^2 \epsilon}}{a_{zx} \cos \epsilon - a_{zz} \sin \epsilon} \quad (14)$$

$$= \frac{a_{zy} \pm \sqrt{a_{zx}^2 + a_{zy}^2 - a_{zz}^2 \tan^2 \epsilon}}{a_{zx} - a_{zz} \tan \epsilon}, \quad (15)$$

where

$$a_{zz}^2 \sin^2 \epsilon \geq (a_{zx}^2 + a_{zy}^2) \cos^2 \epsilon \quad \rightsquigarrow \quad \tan^2 \epsilon \geq \frac{a_{zx}^2 + a_{zy}^2}{a_{zz}^2} \quad (16)$$

must hold. With the substitutions

$$u = a_{zy}, \quad (17)$$

$$v = \sqrt{a_{zx}^2 + a_{zy}^2 - a_{zz}^2 \tan^2 \epsilon} \quad \text{and} \quad (18)$$

$$w = a_{zx} - a_{zz} \tan \epsilon \quad (19)$$

we get

$$\tan\left(\frac{\varphi_0}{2}\right) = \frac{u \pm v}{w} \quad \rightsquigarrow \quad \varphi_0 = \begin{cases} 2 \arctan 2(u + v, w) \\ 2 \arctan 2(u - v, w) \end{cases} \quad (20)$$

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

$$\arctan2(x, y) = \begin{cases} \arctan\left(\frac{x}{y}\right) & y > 0 \\ \arctan\left(\frac{x}{y}\right) + \pi & y < 0, x \geq 0 \\ \arctan\left(\frac{x}{y}\right) - \pi & y < 0, x < 0 \\ \frac{\pi}{2} & y = 0, x > 0 \\ -\frac{\pi}{2} & y = 0, x < 0 \\ 0 & y = 0, x = 0 \end{cases} \quad (21)$$

Iff condition (16) holds, Eqn. (13) has exactly two solutions,<sup>3</sup>

$\varphi_{0,\text{bf}}$  : angle of back to front side crossing and  
 $\varphi_{0,\text{fb}}$  : angle of front to back side crossing,

As PGF's `atan2` function takes values in  $[-360^\circ, +360^\circ]$ , we unwrap  $\varphi_{0,\text{bf}}$  and  $\varphi_{0,\text{fb}}$  as follows:

$$\text{if } \varphi_{0,\text{fb}} - \varphi_{0,\text{bf}} > 360^\circ \text{ then } \varphi_{0,\text{bf}} \leftarrow \varphi_{0,\text{bf}} + 360^\circ, \quad (22)$$

$$\text{if } \varphi_{0,\text{bf}} > \varphi_{0,\text{fb}} \text{ then } \varphi_{0,\text{bf}} \leftarrow \varphi_{0,\text{bf}} - 360^\circ. \quad (23)$$

Now we can draw the arc on the sphere's back side in the rotated *and* offset coordinate system by

`\draw (<\varphi_{0,\text{fb}}>:<r_e>) arc (<\varphi_{0,\text{fb}}>:<\varphi_{0,\text{bf}} + 360^\circ>:<r_e>);`

and the arc on the sphere's front side by

`\draw (<\varphi_{0,\text{bf}}>:<r_e>) arc (<\varphi_{0,\text{bf}}>:<\varphi_{0,\text{fb}}>:<r_e>);`

Otherwise, iff condition (16) does not hold, Eqn. (13) has no solutions, which means that the circle lies entirely either on the front side or on the back side of the sphere. In order to determine which is the case, we test if an arbitrary point on the circle lies above or below the drawing paper (cf. Eq. (12)). We choose  $\varphi_0 = 0$  and evaluate the right side of Eq. (13)

$$a_{zx} \cdot r \cos \epsilon + a_{zz} \cdot r \sin \epsilon \begin{cases} \geq 0 : \text{ front side circle,} \\ < 0 : \text{ back side circle.} \end{cases} \quad (24)$$

Depending on the result, we draw the circle

`\draw (0,0) circle (<r_e>);`

in the front side or back side style.

## 3.2 The Package Source Code

```
1 %% == LaTeX PACKAGE tikz-3dplot-circleofsphere =====
2 %%   Drawing circles of a sphere with tikz-3dplot
3 %%
4 %% Matthias Wolff, BTU Cottbus-Sentenberg
5 %% July 27, 2018
6 %%
7 %% References:
8 %% [1] J. Hein. The tikz-3dplot package. 2012. Online, retrieved July 20, 2018.
9 %%      http://mirror.ctan.org/graphics/pgf/contrib/tikz-3dplot/tikz-3dplot_documentation.pdf
10 %% [2] T. Tantau. TikZ & PGF - Manual for Version 3.0.1a. 2015. Online, retrieved July 22, 2018.
11 %%      http://mirror.ctan.org/graphics/pgf/base/doc/pgfmanual.pdf
12 %% [3] Drawing Great Circles
13 %%      https://tex.stackexchange.com/questions/168521/spherical-triangles-and-great-circles
14
15 %% == REQUIRED PACKAGES =====
16
```

<sup>3</sup>which coincide iff the left and right sides of condition (16) are equal

```

17 \RequirePackage{xifthen}
18 \RequirePackage{tikz}
19 \RequirePackage{tikz-3dplot}
20
21 %% == TikZ STYLES =====
22
23 \tikzset{
24   tdplotCsFront/.style={solid},
25   tdplotCsBack/.style={dashed},
26   tdplotCsFill/.style={opacity=0},
27   tdplotPtFront/.style={},
28   tdplotPtBack/.style={},
29   tdplotCsDrawAux/.style={}
30 }
31
32 %% == COMMANDS =====
33
34 \newcommand{\tdplotCsComputeTransformRotScreen}{%
35   % Computes the elements of the full rotation matrix
36   %
37   %   A = [\axx \axy \axz]
38   %       [\ayx \ayy \ayz]
39   %       [\azz \azy \azz].
40   %
41   % Output:
42   %   \axx - Element A(1,1) of rotation matrix
43   %   \axy - Element A(1,2) of rotation matrix
44   %   ...
45   %   \azz - Element A(3,3) of rotation matrix
46   %
47   \let\atdplotalpha
48   \let\atdplotbeta
49   \let\atdplotmainphi
50   \let\atdplotmaintheta
51   % Row 1: [\axx \axy \axz]
52   \pgfmathsetmacro\axx{cos(\a)*cos(\b)*cos(\p) + cos(\b)*sin(\a)*sin(\p)}
53   \pgfmathsetmacro\axy{cos(\a)*sin(\p) - cos(\p)*sin(\a)}
54   \pgfmathsetmacro\axz{cos(\a)*cos(\p)*sin(\b) + sin(\a)*sin(\b)*sin(\p)}
55   % Row 2: [\ayx \ayy \ayz]
56   \pgfmathsetmacro\ayx{cos(\b)*cos(\p)*sin(\a)*cos(\t) - cos(\a)*cos(\b)*cos(\t)*sin(\p) - sin(\b)*sin(\t)}
57   \pgfmathsetmacro\ayy{cos(\a)*cos(\p)*cos(\t) + sin(\a)*cos(\t)*sin(\p)}
58   \pgfmathsetmacro\ayz{cos(\b)*sin(\t) - cos(\a)*sin(\b)*cos(\t)*sin(\p) + cos(\p)*sin(\a)*sin(\b)*cos(\t)}
59   % Row 3: [\azz \azy \azz]
60   \pgfmathsetmacro\azz{cos(\a)*cos(\b)*sin(\p)*sin(\t) - sin(\b)*cos(\t) - cos(\b)*cos(\p)*sin(\a)*sin(\t)}
61   \pgfmathsetmacro\azy{-cos(\a)*cos(\p)*sin(\t) - sin(\a)*sin(\p)*sin(\t)}
62   \pgfmathsetmacro\azz{cos(\b)*cos(\t) + cos(\a)*sin(\b)*sin(\p)*sin(\t) - cos(\p)*sin(\a)*sin(\b)*sin(\t)}
63 }
64
65 % -----
66
67 \newcommand{\tdplotCsDrawCircle}[5][{}]{%
68   % Draws a circle of a sphere.
69   %
70   % Input:
71   %   #1 - TikZ style
72   %       - use tdplotCsFront/.style={...} to style the front side arc
73   %       - use tdplotCsBack/.style={...} to style the back side arc
74   %       - use tdplotCsFill/.style={...} to style the circle filling
75   %       - use tdplotCsDrawAux to draw some auxiliary information
76   %   #2 - Radius of sphere
77   %   #3 - Azimuthal angle of drawing plane 1)
78   %   #4 - Polar angle of drawing plane 2)
79   %   #5 - Elevation angle of circle above the drawing plane. Permissible
80   %       values are -90 < #5 < 90. Use 0 for drawing a great circle.
81   %
82   % Output:
83   %   none
84   %
85   % Footnotes:
86   %   1) passed as alpha to \tdplotsetrotatedcoords{alpha}{beta}{gamma}
87   %   2) passed as beta to \tdplotsetrotatedcoords{alpha}{beta}{gamma}
88   \begin{scope}[#1]

```

% Macro scope >>

```

89 % Do some computation
90 \pgfmathsetmacro\r {#2}
91 \pgfmathsetmacro\alp{#3}
92 \pgfmathsetmacro\bet{#4}
93 \pgfmathsetmacro\eps{#5}
94 \pgfmathsetmacro\re {\r*cos(\eps)}
95 \pgfmathsetmacro\ze {\r*sin(\eps)}
96 \pgfmathsetmacro\coX{\ze*cos(\alp)*sin(\bet)}
97 \pgfmathsetmacro\coY{\ze*sin(\alp)*sin(\bet)}
98 \pgfmathsetmacro\coZ{\ze*cos(\bet)}
99 \coordinate (coffs) at (\coX,\coY,\coZ);
100 % Rotate and offset coordinate system
101 \tdplotsetrotatedcoords{\alp}{\bet}{0}
102 \tdplotsetrotatedcoordsorigin{(coffs)}
103 % Draw
104 \begin{scope}[tdplot_rotated_coords]
105 \tdplotCsComputeTransformRotScreen
106 \pgfmathsetmacro\tanEps{tan(\eps)}
107 \pgfmathsetmacro\bOneside{((\tanEps)^2)>=((\azx)^2+(\azy)^2)/(\azz)^2)}
108 \ifthenelse{\isin{tdplotCsFill}{#1}}{
109 \fill[tdplotCsFill] (0,0) circle (\re);
110 }{
111 \ifthenelse{\bOneside=1}{
112 \pgfmathsetmacro\bFrontside{(\azx*\re+\azz*\ze)>=0}
113 \ifthenelse{\bFrontside=1}
114 {\draw[tdplotCsFront] (0,0) circle (\re);}
115 {\draw[tdplotCsBack] (0,0) circle (\re);}
116 }{
117 \pgfmathsetmacro\u{\azy}
118 \pgfmathsetmacro\v{\sqrt{(\azx)^2+(\azy)^2-(\azz)^2*(\tanEps)^2}}
119 \pgfmathsetmacro\w{\azx-\azz*\tanEps}
120 \pgfmathsetmacro\phiBf{2*atan2(\u-\v,\w)}
121 \pgfmathsetmacro\phiFb{2*atan2(\u+\v,\w)}
122 \pgfmathsetmacro\bUnwrapA{(\phiFb-\phiBf)>360}
123 \pgfmathsetmacro\bUnwrapB{\phiBf>\phiFb}
124 \ifthenelse{\bUnwrapA=1}{\pgfmathsetmacro\phiBf{\phiBf+360}}{
125 \ifthenelse{\bUnwrapB=1}{\pgfmathsetmacro\phiBf{\phiBf-360}}{
126 \draw[tdplotCsBack] (\phiFb:\re) arc (\phiFb:\phiBf+360:\re);
127 \draw[tdplotCsFront] (\phiBf:\re) arc (\phiBf:\phiFb:\re);
128 }
129 }
130 % Auxiliary drawing (for debugging and illustration)
131 \ifthenelse{\isin{tdplotCsDrawAux}{#1}}{
132 \draw[red!40,->] (-\re,0,0) -- (\re,0,0) node[anchor=north] {$x_d$};
133 \draw[red!40,->] (0,-\re,0) -- (0,\re,0) node[anchor=north] {$y_d$};
134 \draw[red!40,->] (0,0,0) -- (0,0,\re) node[anchor=north] {$z_d$};
135 \ifthenelse{\bOneside=0}{
136 \node[red] at (\phiBf:\re) {$\circ$};
137 \node[red] at (\phiFb:\re) {$\times$};
138 }{
139 \coordinate (coffs) at (-\coX,-\coY,-\coZ);
140 \tdplotsetrotatedcoordsorigin{(coffs)}
141 \begin{scope}[tdplot_rotated_coords]
142 \node[tdplot_screen_coords,red,anchor=north west] at (0.7*\r,-0.9*\r)
143 {\parbox{200pt}{\footnotesize
144 $\theta=\tdplotmaintheta^\circ$, $\phi=\tdplotmainphi^\circ$\\
145 $\alpha=\alp^\circ$, $\beta=\bet^\circ$,
146 $\epsilon=\!\!\!\epsilon^\circ$\\
147 $a_{zx}=\azx$, $a_{zy}=\azy$, $a_{zz}=\azz$\\
148 $r_e=\!\!\!r$, $z_e=\!\!\!ze$\\
149 $\texttt{\textbackslash bOneside}$\\
150 \ifthenelse{\bOneside=1}{
151 $\texttt{\textbackslash bFrontside}$\\
152 }{
153 $\texttt{\textbackslash bUnwrapA}$\\
154 $\texttt{\textbackslash bUnwrapB}$\\
155 $\texttt{\textbackslash phiBf}$\\
156 $\texttt{\textbackslash phiFb}$\\
157 }
158 }
159 \end{scope}
160 \end{scope}

```

```

161 \end{scope} % << (Macro scope)
162 }
163
164 % -----
165
166 \newcommand{\tdplotCsDrawGreatCircle}[4][]{%
167 % Draws a great circle.
168 %
169 % Input:
170 % #1 - TikZ style
171 % - use tdplotCsFront/.style={...} to style the front side arc
172 % - use tdplotCsBack/.style={...} to style the back side arc
173 % - use tdplotCsFill/.style={...} to style the circle filling
174 % - use tdplotCsDrawAux to draw some auxiliary information
175 % #2 - Radius of sphere
176 % #3 - Azimuthal angle of drawing plane 1)
177 % #4 - Polar angle of drawing plane 2)
178 %
179 % Output:
180 % none
181 %
182 % Footnotes:
183 % 1) passed as alpha to \tdplotsetrotatedcoords{alpha}{beta}{gamma}
184 % 2) passed as beta to \tdplotsetrotatedcoords{alpha}{beta}{gamma}
185 \tdplotCsDrawCircle[#1]{#2}{#3}{#4}{0}
186 }
187
188 % -----
189
190 \newcommand{\tdplotCsDrawLatCircle}[3][]{%
191 % Draws a circle of latitude.
192 %
193 % Input:
194 % #1 - TikZ style
195 % - use tdplotCsFront/.style={...} to style the front side arc
196 % - use tdplotCsBack/.style={...} to style the back side arc
197 % - use tdplotCsFill/.style={...} to style the circle filling
198 % - use tdplotCsDrawAux to draw some auxiliary information
199 % #2 - Radius of sphere
200 % #3 - Elevation angle of circle above the drawing plane. Permissible
201 % values are -90 < #5 < 90. Use 0 for drawing a great circle.
202 %
203 % Output:
204 % none
205 \tdplotCsDrawCircle[#1]{#2}{0}{0}{#3}
206 }
207
208 % -----
209
210 \newcommand{\tdplotCsDrawLonCircle}[3][]{%
211 % Draws a circle of longitude.
212 %
213 % Input:
214 % #1 - TikZ style
215 % - use tdplotCsFront/.style={...} to style the front side arc
216 % - use tdplotCsBack/.style={...} to style the back side arc
217 % - use tdplotCsFill/.style={...} to style the circle filling
218 % - use tdplotCsDrawAux to draw some auxiliary information
219 % #2 - Radius of sphere
220 % #3 - Azimuthal angle of drawing plane 1)
221 %
222 % Output:
223 % none
224 %
225 % Footnotes:
226 % 1) passed as alpha to \tdplotsetrotatedcoords{alpha}{beta}{gamma}
227 \tdplotCsDrawCircle[#1]{#2}{#3}{90}{0}
228 }
229
230 % -----
231
232 \newcommand{\tdplotCsFrontsidePoint}{%

```

```

233 % Invoked by \tdplotCsDrawPoint to draw a point on the front side of a sphere.
234 % Redefine to customize.
235 \textbullet%
236 }
237
238 % -----
239
240 \newcommand{\tdtlotCsBacksidePoint}{%
241 % Invoked by \tdplotCsDrawPoint to draw a point on the back side of a sphere.
242 % Redefine to customize.
243 $\circ$%
244 }
245
246 % -----
247
248 \newcommand{\tdplotCsDrawPoint}[4][]{%
249 % Draws a point on a sphere.
250 %
251 % Input:
252 % #1 - TikZ style
253 %      - use \tdplotPtFront/.style={...} to style a front side point
254 %      - use \tdplotPtBack/.style={...} to style a back side point
255 % #2 - Radius of sphere
256 % #3 - Azimuthal angle of drawing plane 1)
257 % #4 - Polar angle of drawing plane 2)
258 %
259 % Output:
260 % none
261 %
262 % Remarks:
263 % - Redefine \tdplotCsFrontsidePoint to customize drawing of a front side
264 %   point.
265 % - Redefine \tdplotCsBacksidePoint to customize drawing of a back side
266 %   point.
267 %
268 % Footnotes:
269 % 1) passed as alpha to \tdplotsetrotatedcoords{alpha}{beta}{gamma}
270 % 2) passed as beta to \tdplotsetrotatedcoords{alpha}{beta}{gamma}
271 \begin{scope}[#1]
272   \pgfmathsetmacro{\r}{#2}
273   \pgfmathsetmacro{\alp}{#3}
274   \pgfmathsetmacro{\bet}{#4}
275   \tdplotsetrotatedcoords{\alp}{\bet}{0}
276   \begin{scope}[tdplot_rotated_coords]
277     \tdplotCsComputeTransformRotScreen
278     \pgfmathsetmacro{\bVisible}{\azz>0}
279     \ifthenelse{\bVisible=1}{%
280       \node[tdplotPtFront] at (0,0,\r) {\tdplotCsFrontsidePoint};
281     }{%
282       \node[tdplotPtBack] at (0,0,\r) {\tdtlotCsBacksidePoint};
283     }
284   \end{scope}
285 \end{scope}
286 }
287
288 %% == EOF =====

```

```

% Macro scope >>
% Parse radius
% Parse alpha angle
% Parse beta angle
% Set rotated coord. system
% Draw in rotated coord. system >>
% Get \azz
% Test if point is on visible side
% Point on front side >>
% Draw it
% << Point on back side >>
% Draw it
% <<
% <<
% <<

```

### 3.3 An Auxiliary Matlab Script

```

1 %% == LaTeX PACKAGE tikz-3dplot-circleofsphere =====
2 %   Drawing circles of a sphere with tikz-3dplot
3 %
4 %   Matthias Wolff, BTU Cottbus-Sentenberg
5 %   July 26, 2018
6 %
7 %   References:
8 %   [1] J. Hein. The tikz-3dplot package. 2012. Online, retrieved July 20, 2018.
9 %       https://mirror.hmc.edu/ctan/graphics/pgf/contrib/tikz-3dplot/tikz-3dplot\_documentation.pdf
10 %
11
12 %% Rotation matrices =====
13 syms a b p t

```



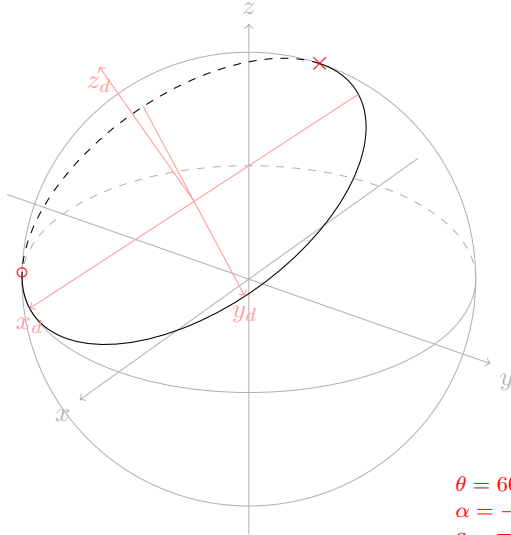
```

14
15 % R rotation matrix -----
16 Rz = [ cos(p) -sin(p) 0
17         sin(p) cos(p) 0
18         0 0 1 ];
19
20 Rx = [ 1 0 0
21         0 cos(t) -sin(t)
22         0 sin(t) cos(t) ];
23
24 % - [1] eq. (2.1) line 2
25 % R = Rz*Rx; disp(R);
26
27 % - [1] eq. (2.1) line 3
28 % R = [ cos(p) sin(p) 0
29         -cos(t)*sin(p) cos(t)*cos(p) -sin(t)
30         sin(t)*sin(p) -sin(t)*cos(p) cos(t) ];
31
32 % - [1] eq. (2.1) line 3, corrected
33 R = (Rz*Rx).';
34
35 % -- D rotation matrix -----
36 Dz = [ cos(a) -sin(a) 0
37         sin(a) cos(a) 0
38         0 0 1 ];
39
40 Dy = [ cos(b) 0 sin(b)
41         0 1 0
42         -sin(b) 0 cos(b) ];
43
44 Dx = [ 1 0 0
45         0 cos(b) -sin(b)
46         0 sin(b) cos(b) ];
47
48 D = Dz*Dy; disp(D);
49
50 % -- Full rotation matrix -----
51 A = R*D; disp(A);
52 axx = A(1,1); axy = A(1,2); axz = A(1,3);
53 ayx = A(2,1); ayy = A(2,2); ayz = A(2,3);
54 azx = A(3,1); azy = A(3,2); azz = A(3,3);
55
56 %% == Transform a vector (world -> screen) =====
57 syms x y z
58 p = [ x
59       y
60       z ];
61 q=A*p;
62 disp(q);
63
64 %% == View angle =====
65 syms p0 r eps axz azy azz
66 assume(p0,'real');
67 assume(r,'real');
68 assume(eps,'real');
69 assume(axz,'real');
70 assume(azy,'real');
71 assume(azz,'real');
72 eqn = axz*r*cos(eps)*cos(p0) + azy*r*cos(eps)*sin(p0) + azz*r*sin(eps) == 0
73 solve(eqn,p0,'Real',true)
74
75 %% == EOF =====

```

## References

- [1] Jeff Hein. The `tikz-3dplot` package. [http://mirror.ctan.org/graphics/pgf/contrib/tikz-3dplot/tikz-3dplot\\_documentation.pdf](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.
- [3] Matthias Wolff. The `tikz-3dplot-circleofsphere` package: Drawing circles of a sphere with `tikz-3dplot`. <https://github.com/matthias-wolff/tikz-3dplot-circleofsphere>, 2018. Retrieved: July 27, 2018.



$\theta = 60.0^\circ, \phi = 125.0^\circ$   
 $\alpha = -40.0^\circ, \beta = 30^\circ, \epsilon = 30^\circ$   
 $a_{zx} = -0.05588, a_{zy} = 0.8365, a_{zz} = 0.54507$   
 $r_e = 2.59808, z_e = 1.5$   
 $\backslash bOneside=0, \backslash bUnwrapA=0, \backslash bUnwrapB=1$   
 $o: \backslash phiBf = -18.22858^\circ, x: \backslash phiFb = 205.86197^\circ$

```

1 \documentclass{standalone}
2 \usepackage[dvipsnames]{xcolor}
3 \usepackage{tikz-3dplot-circleofsphere}
4
5 \begin{document}
6
7 \def\elev{ 30} \pgfmathsetmacro{\tdpTheta}{90-\elev}
8 \def\azim{ 35} \pgfmathsetmacro{\tdpPhi}{90+\azim}
9 \def\R{3}
10 \tdplotsetmaincoords{\tdpTheta}{\tdpPhi}
11 \begin{tikzpicture}[scale=1,tdplot_main_coords]
12 \begin{scope}[black!30,name=auxiliary]
13 \draw[tdplot_screen_coords] (0,0,0) circle (\R);
14 \draw[>-] (-1.3*\R,0,0) -- (1.3*\R,0,0) node[anchor=north east]{$x$};
15 \draw[>-] (0,-1.3*\R,0) -- (0,1.3*\R,0) node[anchor=north west]{$y$};
16 \draw[>-] (0,0,-1.3*\R) -- (0,0,1.3*\R) node[anchor=south]{$z$};
17 \tdplotCsDrawCircle{\R}{0}{0};
18 \end{scope}
19 \begin{scope}
20 % \tdplotCsDrawLatCircle[tdplotCsDrawAux]{\R}{-30}
21 % --
22 \tdplotCsDrawCircle[tdplotCsDrawAux]{\R}{-40}{30}{30}
23 % --
24 % \foreach \a in {0,15,...,345}
25 % { \tdplotCsDrawCircle[very thin,gray]{\R}{\a}{90}{0} }
26 % \foreach \a in {-75,-60,...,75}
27 % { \tdplotCsDrawCircle[very thin,gray]{\R}{0}{0}{\a} }
28 % -- Pathologic cases -->
29 % \tdplotCsDrawCircle{\R}{35}{60}{0}
30 % <--
31 \end{scope}
32 \end{tikzpicture}
33
34 \end{document}

```