Tutorial 16: Create Tube-Style Solids by Succeeding Polygons

2015-10-04: Tim C. Lueth, MIMED - Technische Universität München, Germany (URL: http://www.mimed.de) - Last Change: 2017-07-07

Contents

Complete List of all Tutorials with Publishable MATLAB Files of this Solid-Geoemtries Toolbox

The following topics are covered an explained in the specific tutorials:

- Tutorial 01: First Steps Using the VLFL-Toolbox for Solid Object Design
- Tutorial 02: Using the VLFL-Toolbox for STL-File Export and Import
- Tutorial 03: Closed 2D Contours and Boolean Operations in 2D
- Tutorial 04: 2½D Design Using Boolean Operators on Closed Polygon Lists (CPL)
- Tutorial 05: Creation, Relative Positioning and Merging of Solid Geometries (SG)
- Tutorial 06: Relative Positioning and Alignment of Solid Geometries (SG)
- Tutorial 07: Rotation of Closed Polygon Lists for Solid Geometry Design
- Tutorial 08: Slicing, Closing, Cutting and Separation of Solid Geometries
- Tutorial 09: Boolean Operations with Solid Geometries
- Tutorial 10: Packaging of Sets of Solid Geometries (SG)
- Tutorial 11: Attaching Coordinates Frames to Create Kinematik Models
- Tutorial 12: Define Robot Kinematics and Detect Collisions
- Tutorial 13: Mounting Faces and Conversion of Blocks into Leightweight-structures
- Tutorial 14: Manipulation Functions for Closed Polygons and Laser Cutting (SVG)
- Tutorial 15: Create a Solid by 2 Closed Polygons
- Tutorial 16: Create Tube-Style Solids by Succeeding Polygons
- Tutorial 17: Filling and Bending of Polygons and Solids
- Tutorial 18: Analyzing and modifying STL files from CSG modeler (Catia)
- Tutorial 19: Creating drawing templates and dimensioning from polygon lines
- Tutorial 20: Programmatically Interface to SimMechanics Multi-Body Toolbox
- Tutorial 21: Programmatically Convert Joints into Drives (SimMechanics)
- Tutorial 22: Adding Simulink Signals to Record Frame Movements
- Tutorial 23: Automatic Creation of a Missing Link and 3D Print of a Complete Model
- Tutorial 24: Automatic Creation of a Joint Limitations
- Tutorial 25: Automatic Creation of Video Titels, Endtitels and Textpages
- Tutorial 26: Create Mechanisms using Universal Planar Links
- Tutorial 27: Fourbar-Linkage: 2 Pose Syntheses and Linkage Export for 3D Printing
- Tutorial 28: Fourbar-Linkage: 3 Pose Syntheses and Linkage Export for 3D Printing
- Tutorial 29: Create a multi body simulation using several mass points
- Tutorial 30: Creating graphical drawings using point, lines, surfaces, frames etc.
- Tutorial 31: Importing 3D Medical DICOM Image Data and converting into 3D Solids
- Tutorial 32: Exchanging Data with a FileMaker Database
- Tutorial 33: Using a Round-Robin realtime multi-tasking system
- Tutorial 34: 2D Projection Images and Camera Coordinate System Reconstruction
- Tutorial 35: Collection of Ideas for Tutorials
- Tutorial 36: Creating a Patient-Individual Arm-Skin Protector-Shell

Motivation for this tutorial: (Originally SolidGeometry 2.7 required)

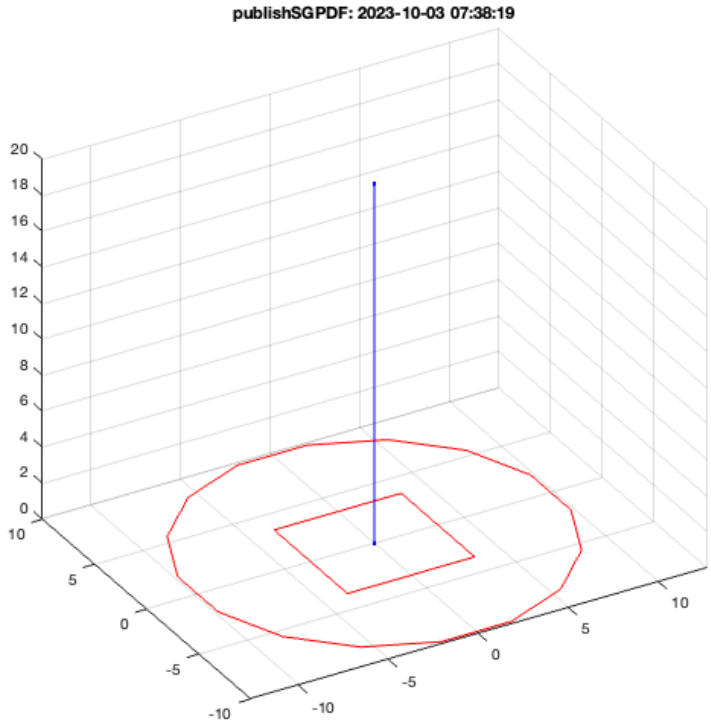
2. Tube generation by repeating identical CPL along a 3D path

For robotics design,very often we have a wish to extrude a CPL not only in an orthognal z direction to the xy-plane, but in an any desired direction even along a path in 3D space. Intutively we expect a result, but this is not easy to achieve automatically. Anyway, for those tasks we have two functions:

- SGcontourtube - repeats a CPL along a path in 3D
- FLofCVL

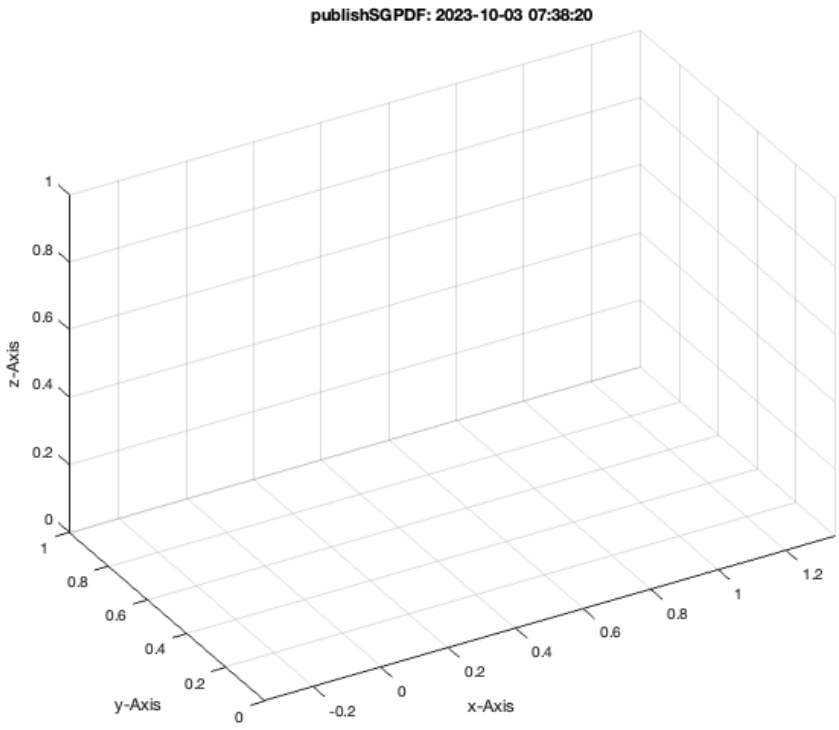
Let us start with a planar contour in the x/y-plane, and an orthogonal path

```
SGfigure; axis on ; grid on;
CPL=CPLsample(8);
CPLplot(CPL,'r-');

VL=[0 0 0; 0 0 20];
VLplot(VL,'b.-'); view(-30,30);
```



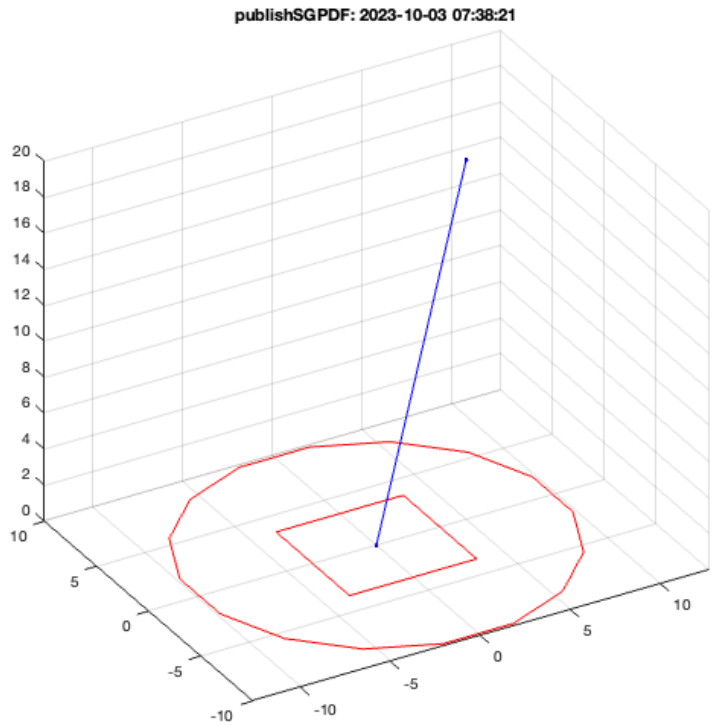
The final result looks as expected

```
SG=SGcontourtube(CPL,VL); SGfigure(SG); VLFLplotlight(1,1);view (-30,30);
```



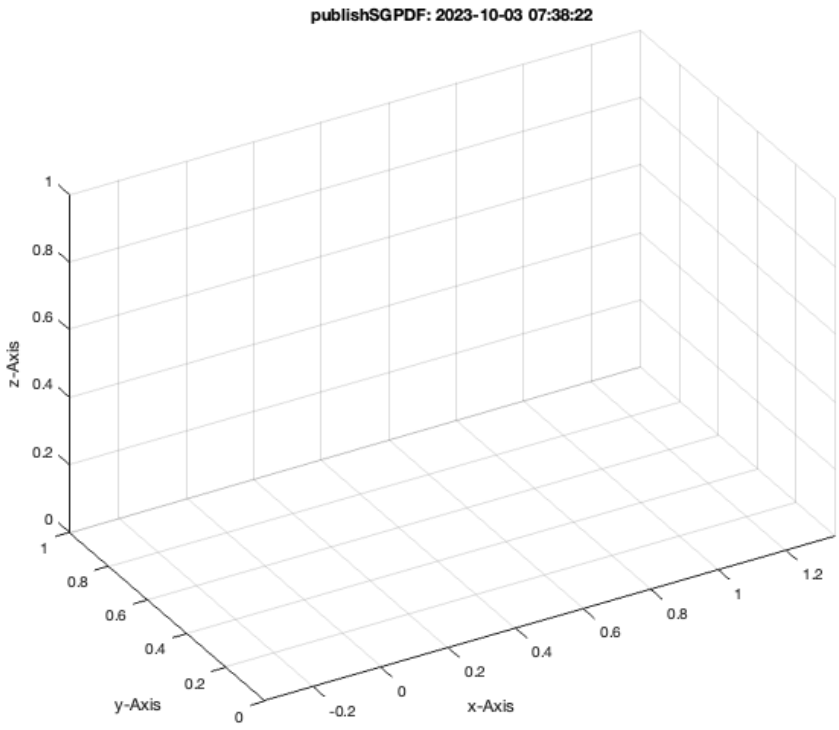
Now we change the path a little

```
SGfigure;  axis on ; grid on;
CPLplot(CPL,'r-');
VL=[0 0 0; 5 0 20];
VLplot(VL,'b.-'); view(-30,30);
```



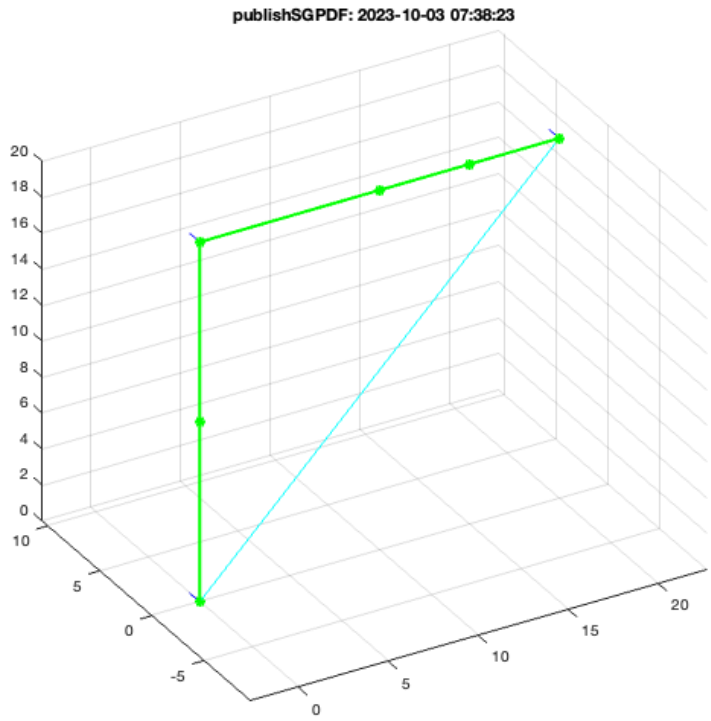
The final result may not look as expected

```
SG=SGcontourtube(CPL,VL);
SGfigure; axis on ; grid on;
SGplot(SG,'m');
VLFLplotlight(1,1);view (-30,30);
```



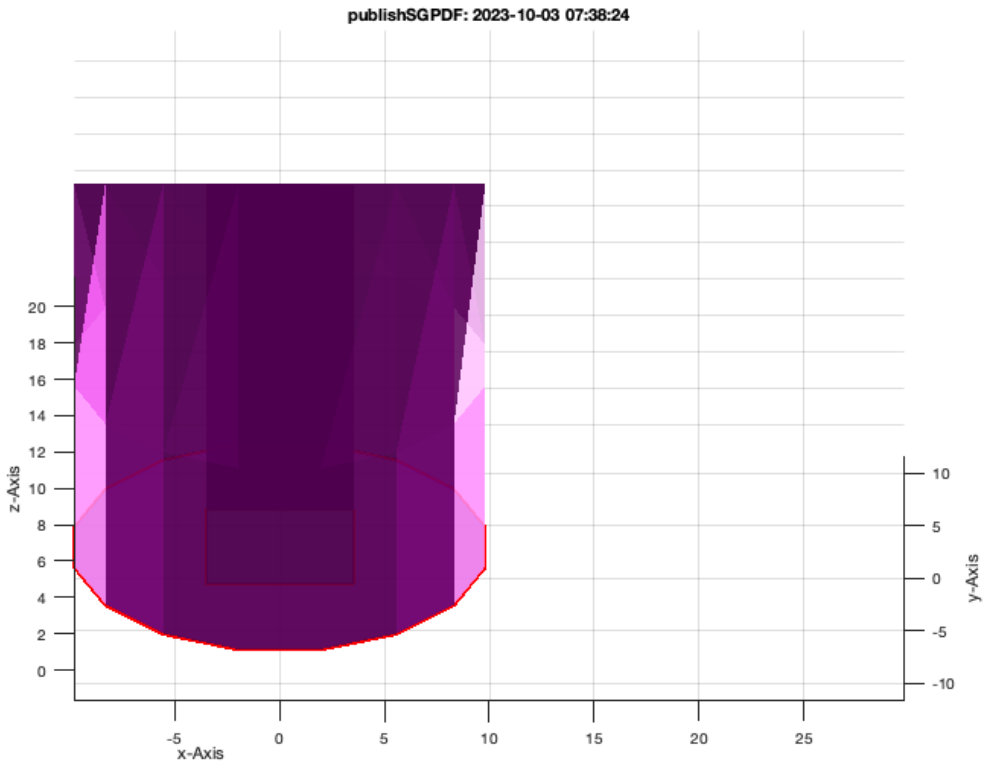
Now we change the path, still a planar one and anylyze the angle situation The cynan colored path explain that the vertex list has to be closed to analyze the first an last angle. Furthermore we see that more than one point has no defined orthogonal vector since it sits on a straight line

```
SGfigure; CPLplot(CPL,'r-');  axis on ; grid on;
VL=[0 0 0; 0 0 10; 0 0 20; 10 0 20; 15 0 20; 20 0 20];
VLangle(VL);
```



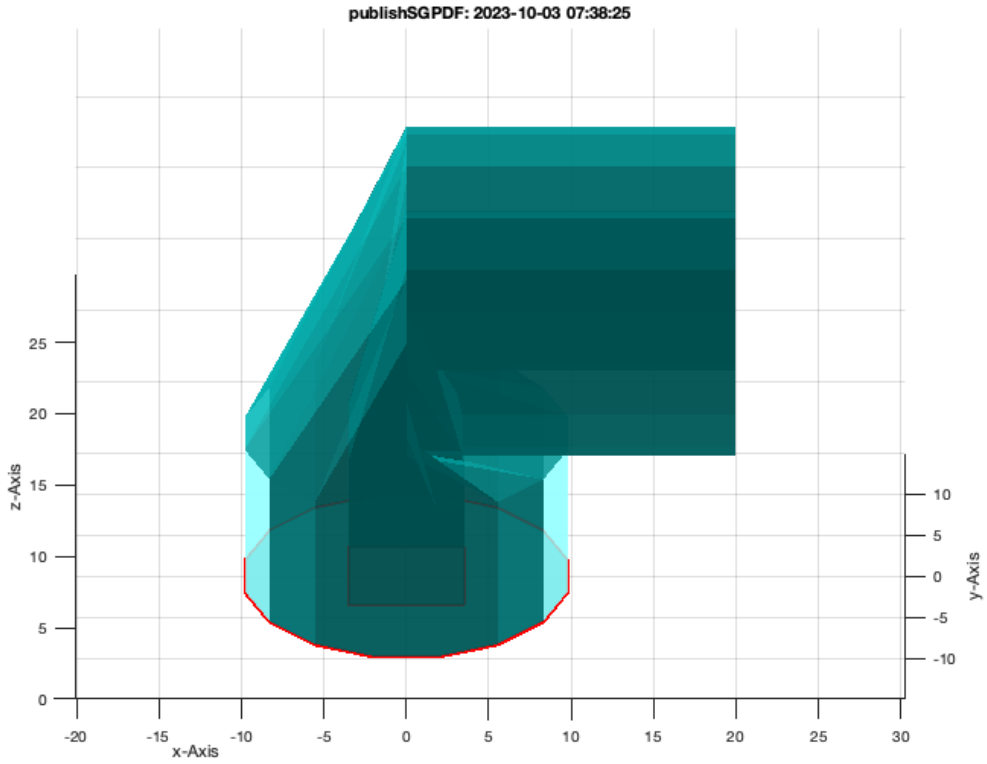
The result is not may be we expected

```
SGfigure; CPLplot(CPL,'r-',2);  axis on ; grid on;
SG=SGcontourtube(CPL,VL); SGplot(SG,'m'); VLFLplotlight(1,0.8);view (0,30);
```

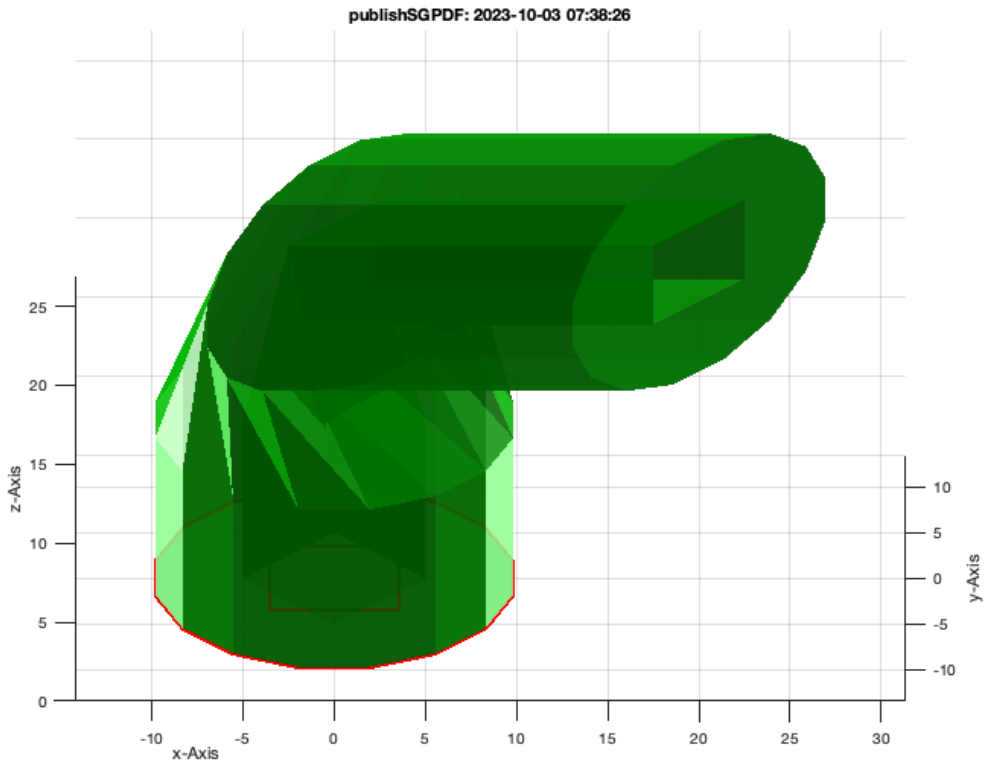


The result can be adjusted by defining the ex-vector at the start point

```
SGfigure; CPLplot(CPL,'r-',2);  axis on ; grid on;
SG=SGcontourtube(CPL,VL,[0 1 0]); SGplot(SG,'c'); VLFLplotlight(1,0.8);view (0,30);
```



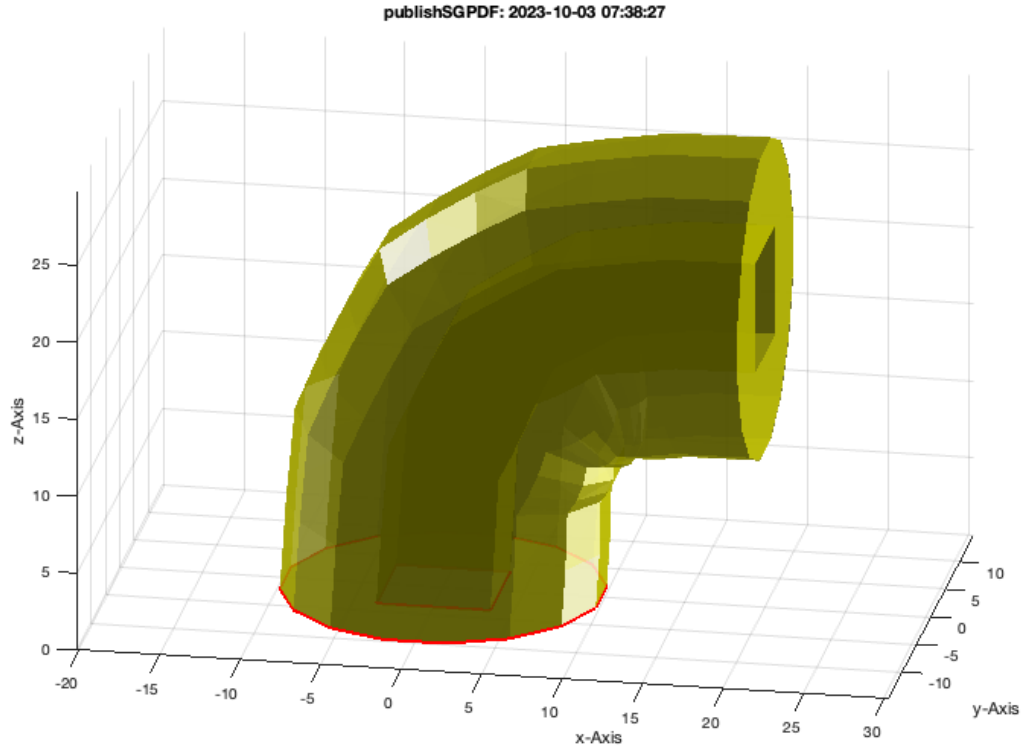

```
SGfigure; CPLplot(CPL,'r-',2);  axis on ; grid on;
SG=SGcontourtube(CPL,VL,[1 1 0]); SGplot(SG,'g'); VLFLplotlight(1,0.8);view (0,30);
```



One lazy approach is delivered by SGofCPLCVLR without a given radius

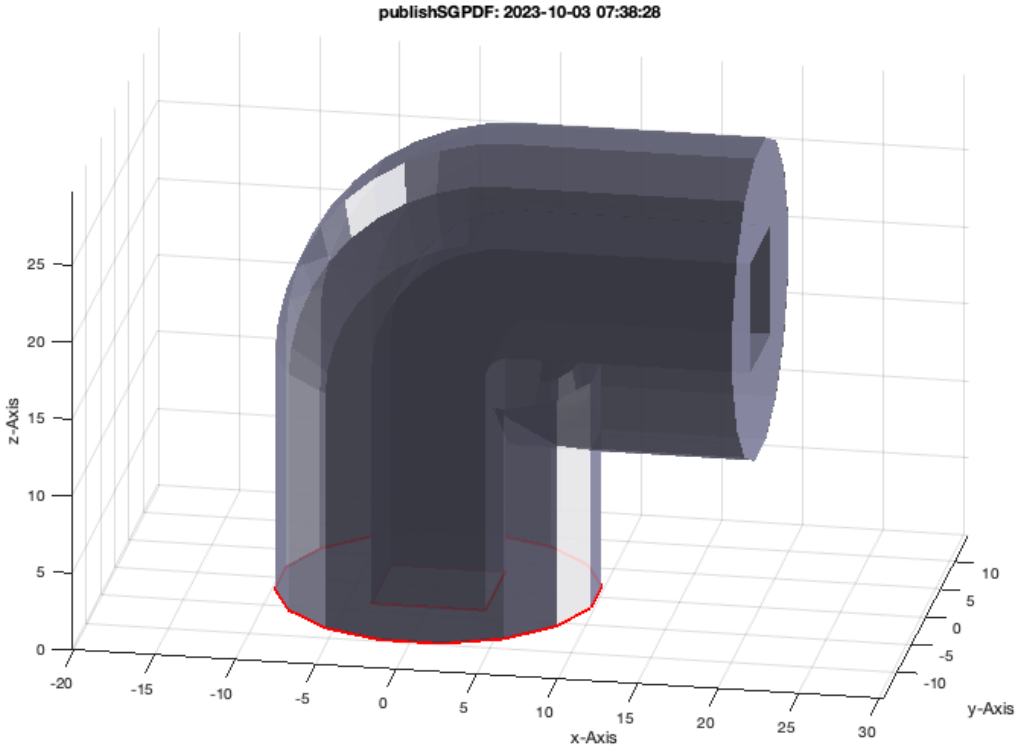
```
SGfigure; CPLplot(CPL,'r-',2);  axis on ; grid on;
SG=SGofCPLCVLR(CPL,VL); SGplot(SG,'y'); VLFLplotlight(1,0.8);view (10,20);
```

```
Warning: 1st Euler angle does not fit to vertex list path direction
Warning: Last Euler angle does not fit to vertex list path direction
```



One lazy approach is delivered by SGofCPLCVLR including a radius 5

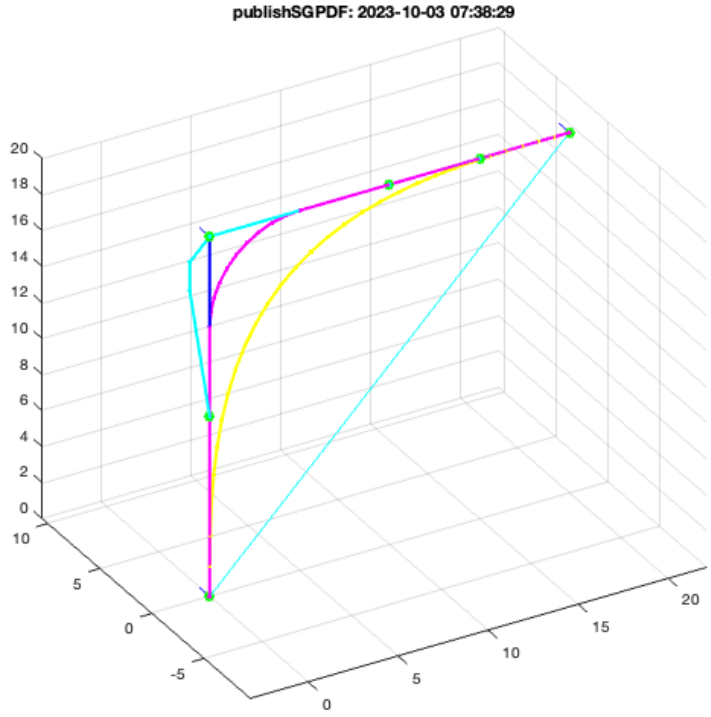
```
SGfigure; CPLplot(CPL,'r-',2);  axis on ; grid on;
SG=SGofCPLCVLR(CPL,VL,5); SGplot(SG,'w'); VLFLplotlight(1,0.8);view (10,20);
```



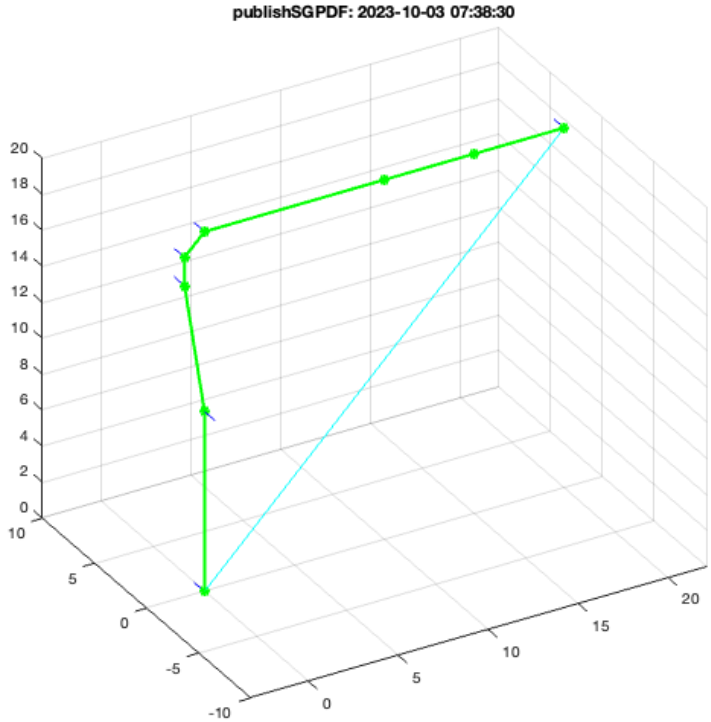
3. Tube generation using a 3D path with Bezier-curves or radial edges

Create a 2D closed polygon line to by copied in 3D space

```
SGfigure; CPLplot(CPL,'r-');  axis on ; grid on;
VL=[0 0 0; 0 0 10; 0 0 20; 10 0 20; 15 0 20; 20 0 20];
VLangle(VL);

VLB=VLBezierC(VL,30);
VLR=VLRadiusC(VL,pi/4,2);
VLr=VLradialEdges(VL,5);
VLplot(VL,'b.-',2); view (-30,30);
VLplot(VLB,'y.-',2); view (-30,30);
VLplot(VLR,'c.-',2); view (-30,30);
VLplot(VLr,'m.-',2); view (-30,30);
```

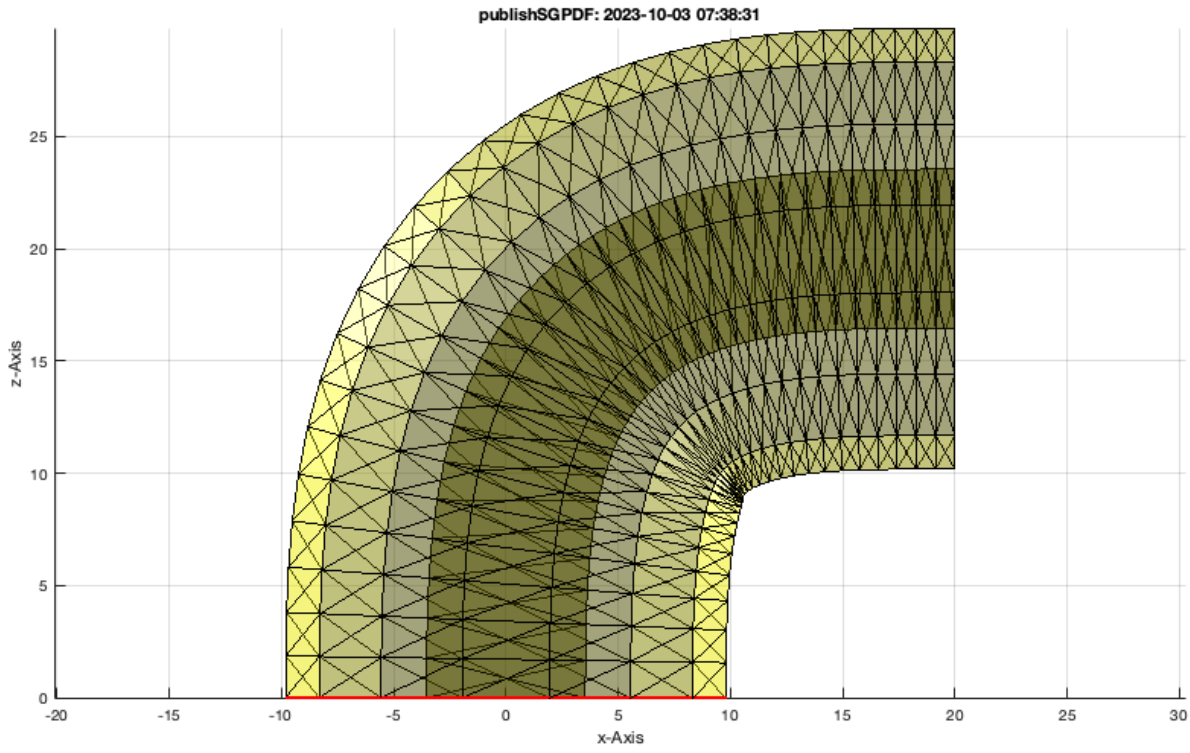



```
VLangle(VLR);
```



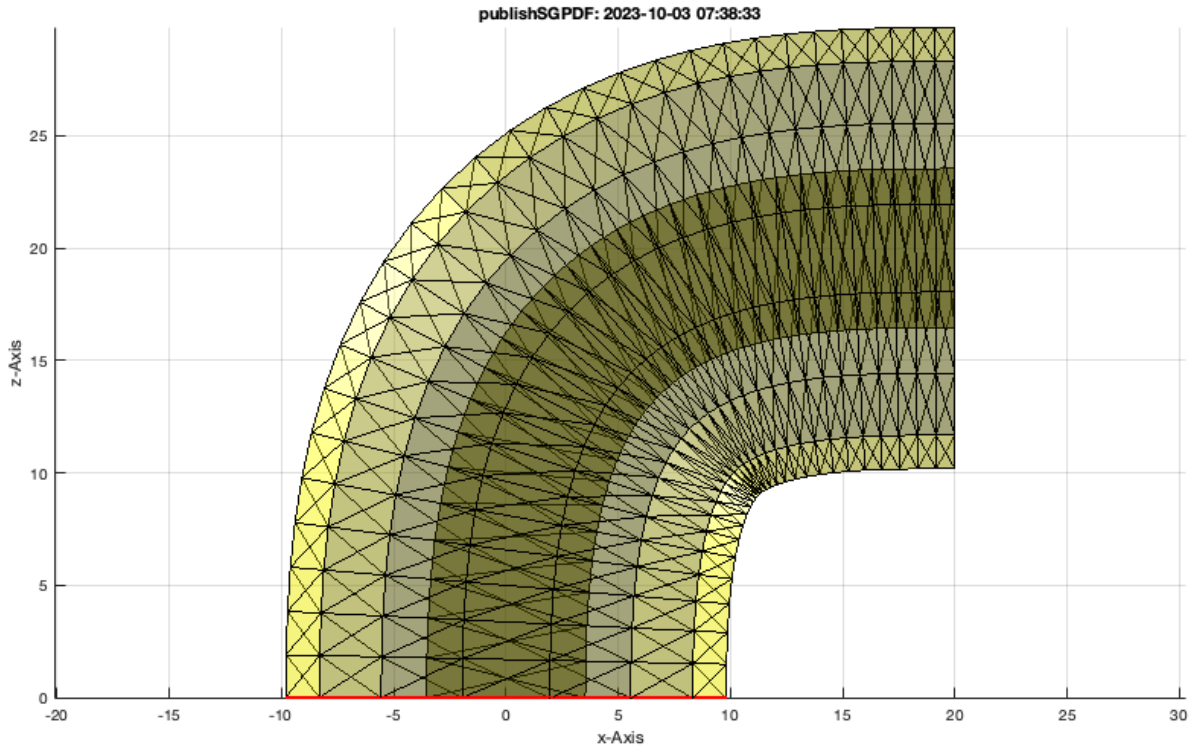
The Bezier-curve tube

```
SGfigure; CPLplot(CPL,'r-',2);  axis on ; grid on;
SG=SGcontourtube(CPL,VLB); SGplot(SG,'y'); VLFLplotlight(0,0.3);view (0,0);
```



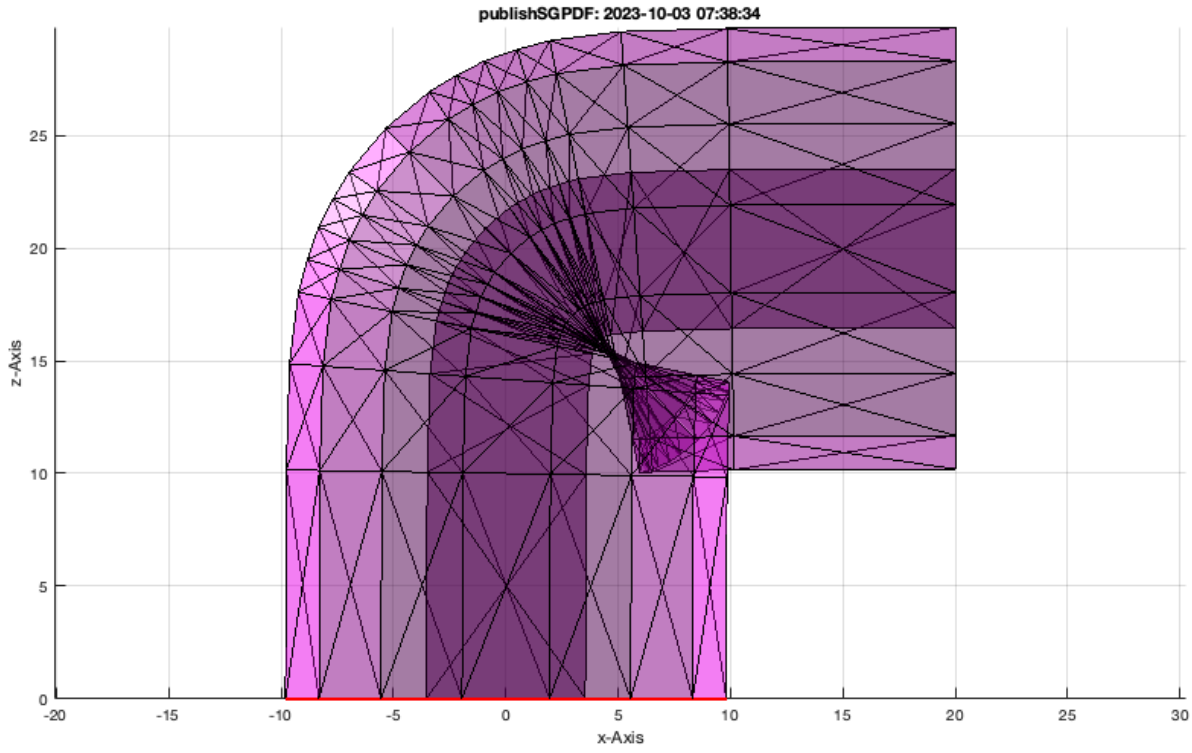
The Bezier-curve tube

```
SGfigure; CPLplot(CPL,'r-',2);  axis on ; grid on;
SG=SGofCPLCVLR(CPL,VLB); SGplot(SG,'y'); VLFLplotlight(0,0.3);view (0,0);
```



The Radial-curve tube

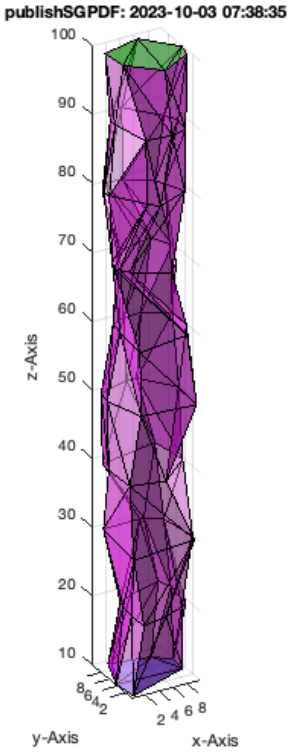
```
SGfigure; CPLplot(CPL,'r-',2);  axis on ; grid on;
SG=SGofCPLCVLR(CPL,VLr); SGplot(SG,'m'); VLFLplotlight(0,0.3);view (0,0);
```



4. Creating solids by closed polygons in different height: z-coordinate

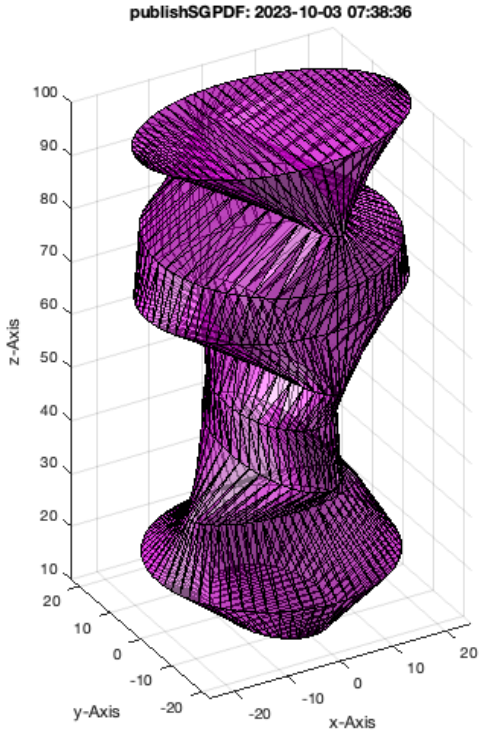
The connection of contours in different z-values works currently only with ONE contour per z-value

```
SGfigure; view(-30,30); axis on; grid on;
VL=[]; for i=1:10; VL=[VL;VLaddz(PLconvexhull(10*rand(20,2)),i*10)]; end;
[FLB,FLW,FLT]=FLofCVL(VL);
VLFLplot(VL,FLB,'b'); VLFLplot(VL,FLW,'m'); VLFLplot(VL,FLT,'g'); VLFLplotlight(0,0.5)
```



Same us but this time with ellipoids

```
SGfigure; view(-30,30); axis on; grid on;
VL=[]; for i=1:10; VL=[VL;VLaddz(PLcircle(5+20*rand,[],[],5+20*rand),i*10)]; end;
[FLB,FLW,FLT]=FLofCVL(VL); FL=[FLB;FLW;FLT];
VLFLplot(VL,FL,'m'); VLFLplotlight(0,0.5)
```



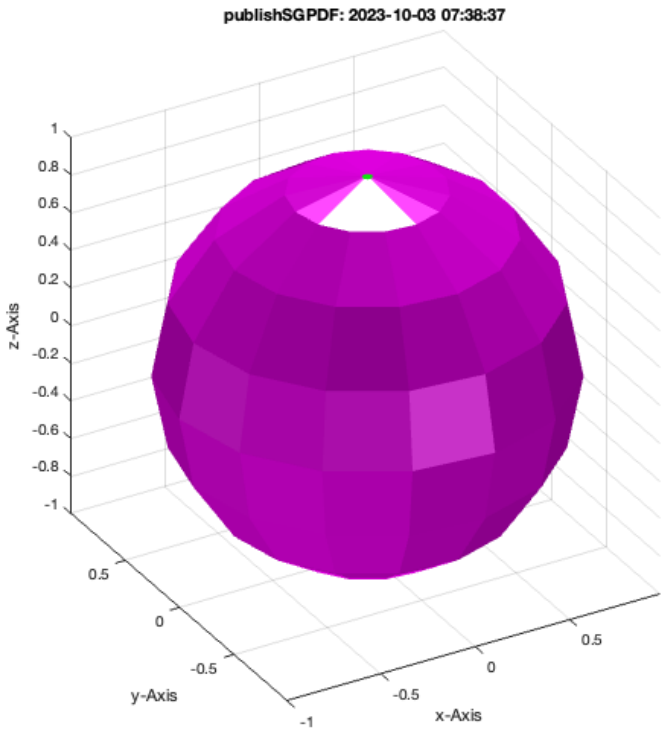
5. Creating a sphere with minmal number of points

For the creation of spherical joints, we need sphered shaped geometries. Those spheres consist of circular point lists in different z-height. The number of points of each polygon, the number of polygons an the z-resolution depend on the size of the sphere.

A sphere with just 1mm radius and a resolution of 50µm (default) has only hundreds of facets.

```
SGsphere(1)
```

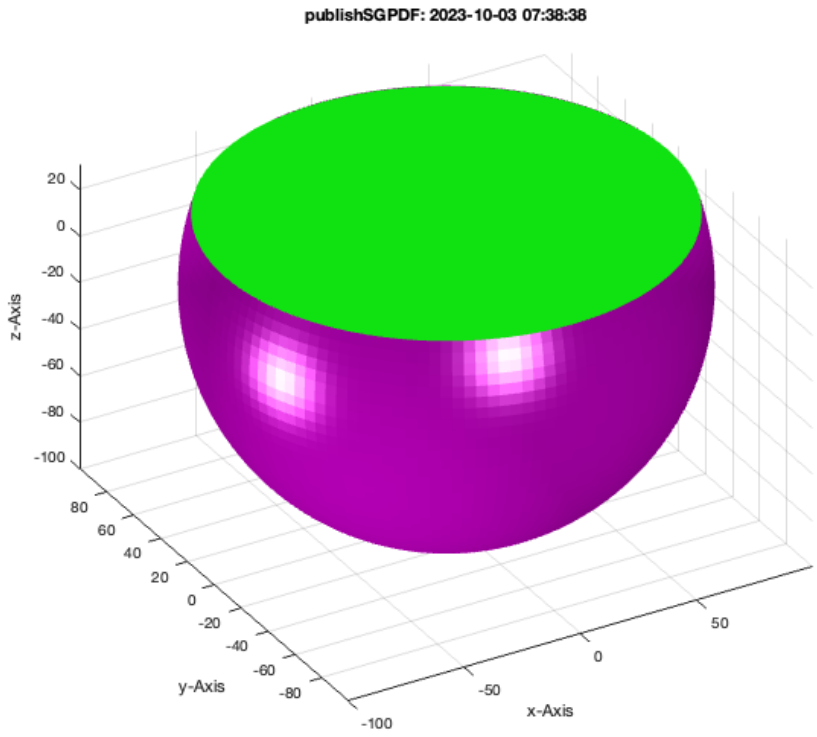
```
ans =
  struct with fields:

    VL: [135×3 double]
    FL: [266×3 double]
```



Asphere with 100mm radius and a resolution of 50µm (default) has then thausands of facets.

```
SGsphere(100,[],pi/10)
```

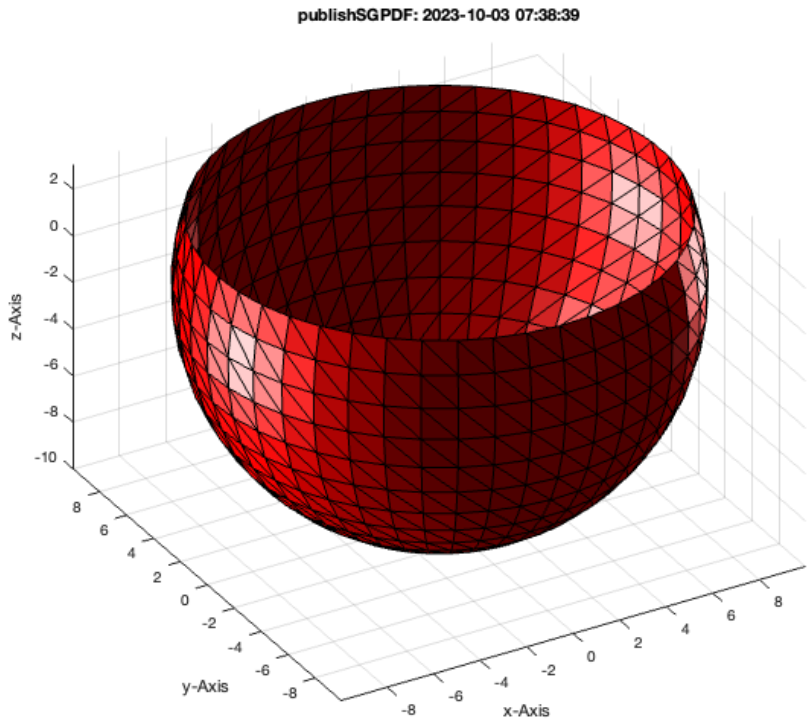
```
ans =
  struct with fields:

    VL: [6063×3 double]
    FL: [12122×3 double]
```



6. Creating a spherical joint

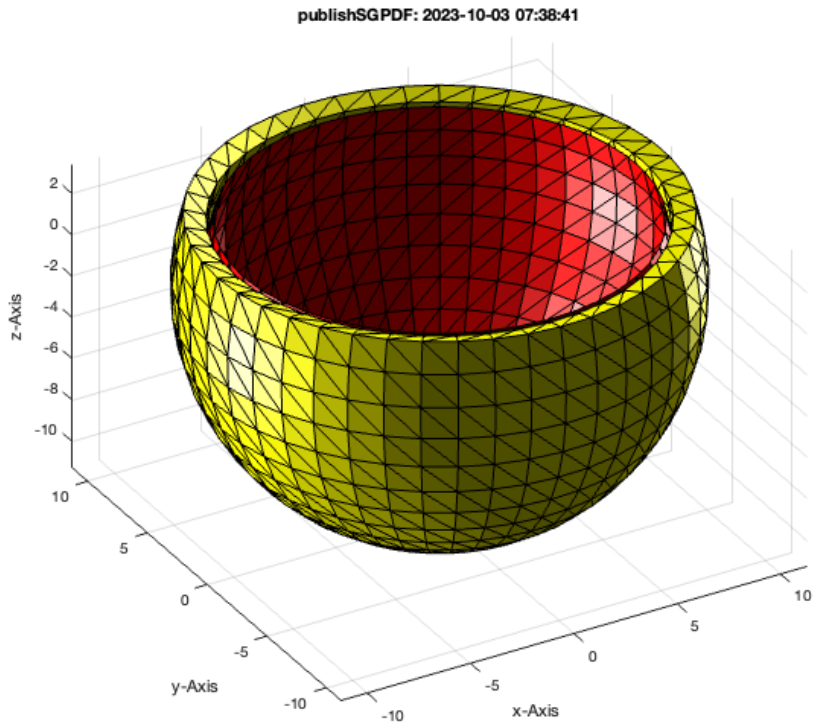
First we have to create a sphere and separate the spherical surface

```
SGfigure; view(-30,30);
[~,~,SG]=MLofSG(SGsphere(10,[],pi/10));
VLFLplot(SG.VL,SG.FL(SG.ML(:,1)==1,:));
```



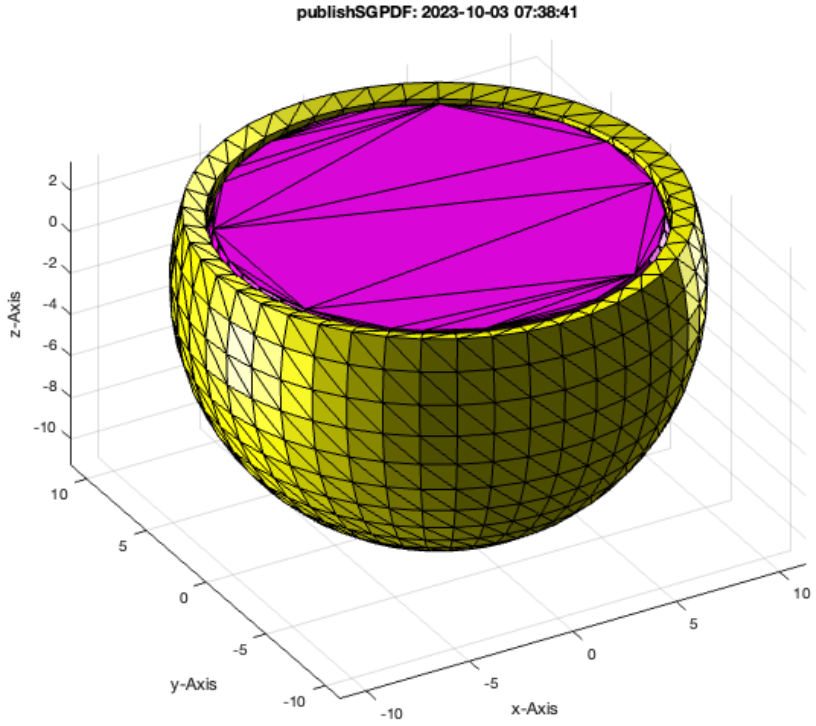
Now create the surface for the joint from the spherical surfacewith tickness 1 as bearing

```
SGofSurface(SG.VL,SG.FL(SG.ML(:,1)==1,:),1);
```



Now fill in the sphere ball as joint

```
SGplot(SG,'m');
```



Final remarks on toolbox version and execution date

```
VLFLlicense
```

```
This VLFL-Lib, Rel.  (2023-Oct-03), is for limited non commercial educational use only!
Licensee: Tim Lueth (Development Version)!
Please contact Tim Lueth, Professor at TU Munich, Germany!
WARNING: This VLFL-Lib (Rel. ) license will exceed at 06-Jul-2078 07:38:42!
Executed 03-Oct-2023 07:38:44 by 'timlueth' on a MACI64 using Mac OSX 13.6 | R2023a Update 5 | SG-Lib 5.4
======================================== Used Matlab products: ========================================
distrib_computing_toolbox
map_toolbox
matlab
================================================================================================================
```

- *Tim Lueth, tested and compiled on OSX 10.7.5 with Matlab 2014b on 2015-10-12*
- _____, executed and published on 64 Bit PC using Windows with Matlab 2015a on 2015-xx-xx_

Published with MATLAB® R2023a