Tutorial 44: Creation of solids and kinematics from 3D curves and transformation matrices

2018-07-24: Tim C. Lueth, MIMED - Technische Universität München, Germany (URL: http://www.mimed.de) - Last Change: 2018-07-24

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

Motivation for this tutorial: (Originally SolidGeometry 4.2 required)

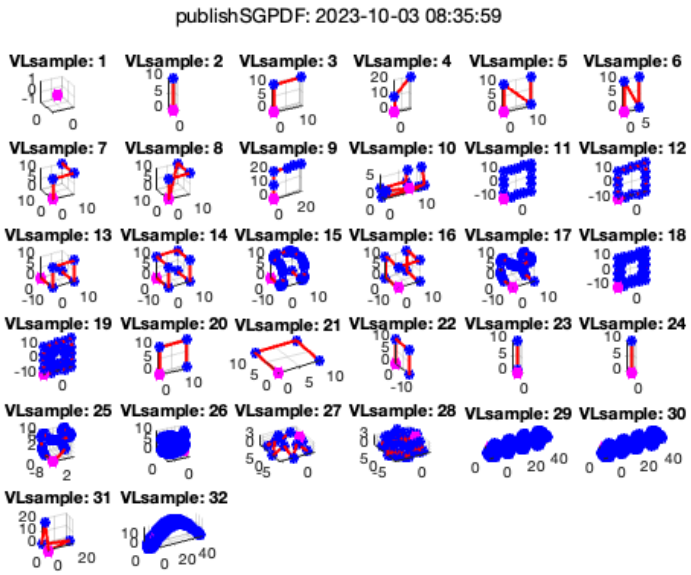
The creation of solids from space curves and a cross-section polygon is not trivial. The affected persons in the SGLib are VLsample - for generating example curves VLedgeNormal - Non trivial function for creating normal orthogonal vectors The creation of solids from space curves and a cross-section polygon is not trivial. The affected functions in the SGLib are VLsample - for generating example curves VLedgeNormal - Non-trivial function for generating normal orthogonal vectors SGcontourtube - The first function with rotating matrices (error in special cases) SGcontourtube2 - The new function with VLedgeNormal (previously error-free) SGofCPLCVVLR - Now based on SGcontourtube2 SGof2T - Now based on SGcontourtube2 SGTofDenavitHartenberg - Based on SGof2T SGTofDHset - Based on SGTofDenavitHartenberg

List of function introduced in this tutorial

****

Using VLsample to create example funktions

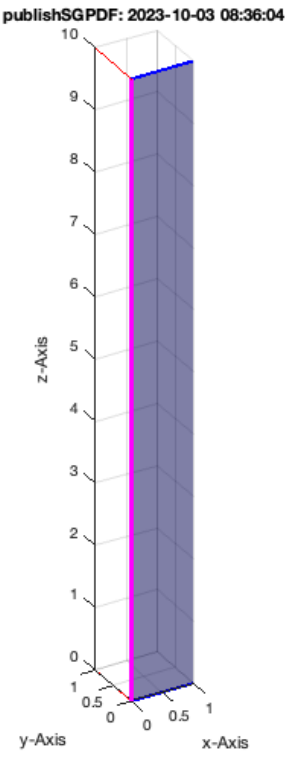
```
VLsample;
```



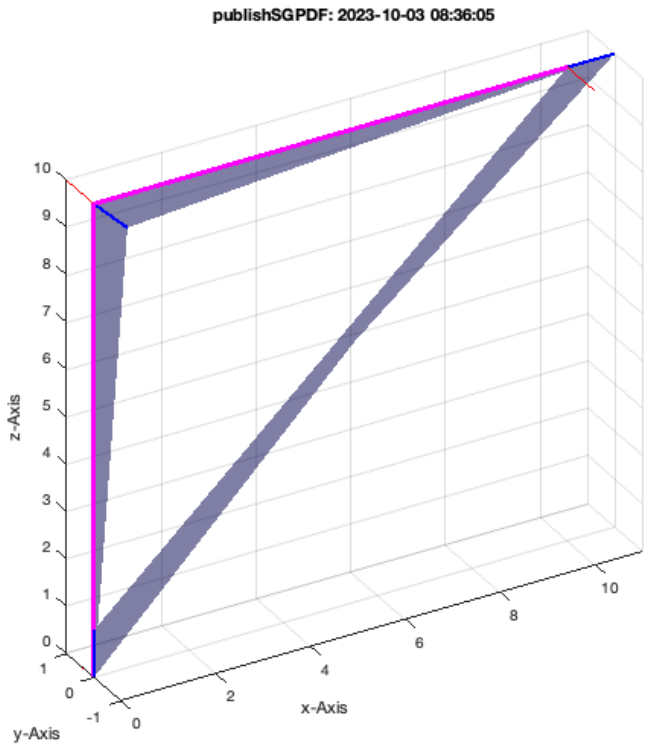
Creating edge normal function for an open spatial curve

If angles are larger than 90 degree (pi/2)

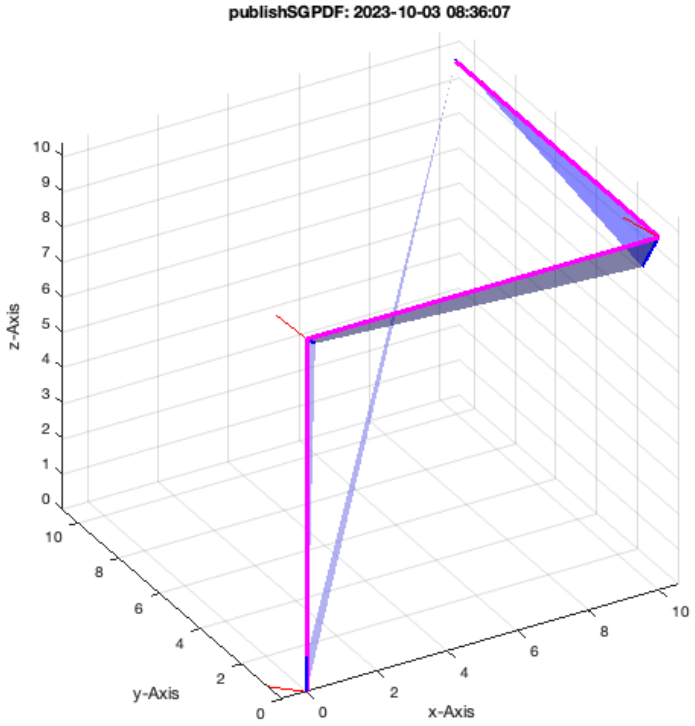
```
VLedgeNormal(VLsample(2));
```



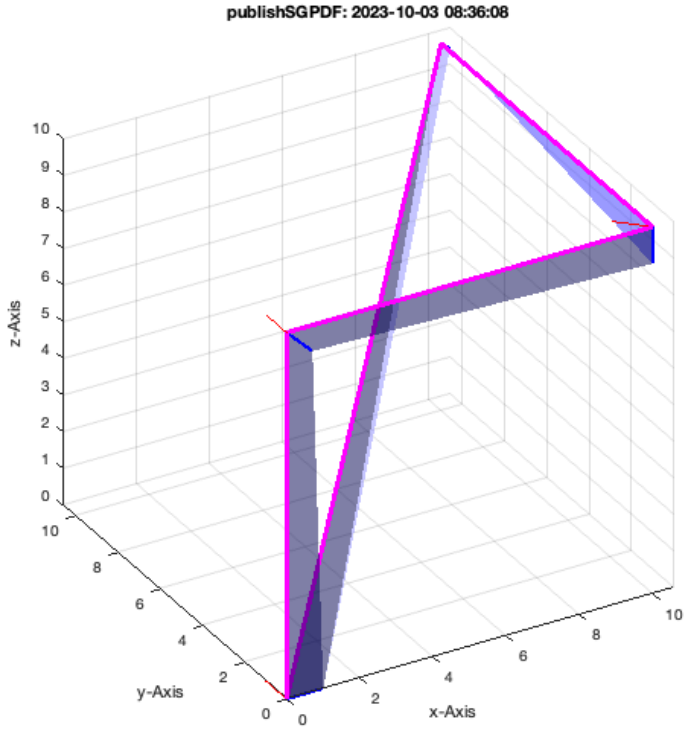

```
VLedgeNormal(VLsample(3));
```



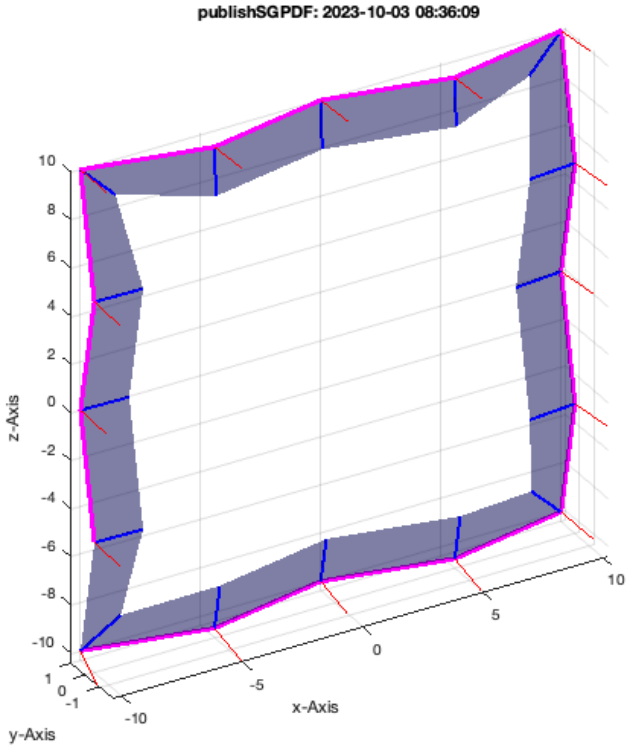

```
VLedgeNormal(VLsample(7));
```



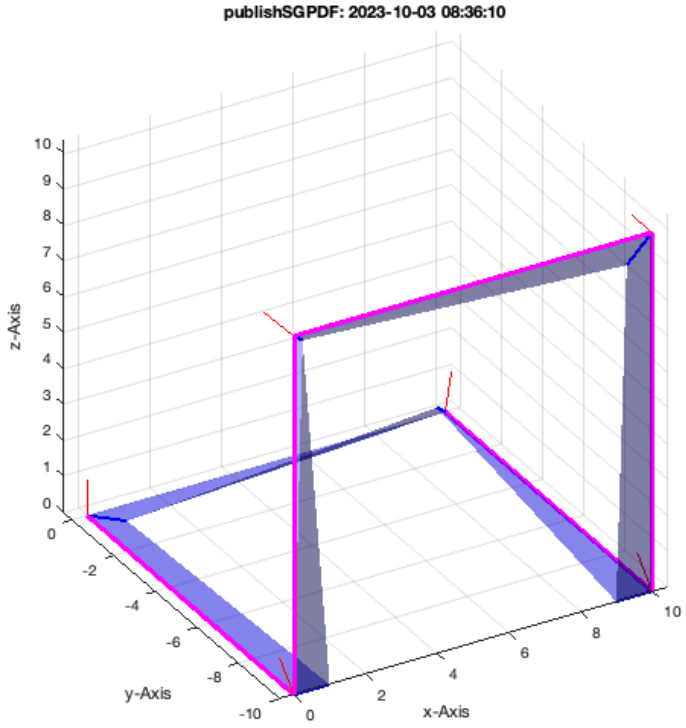

```
VLedgeNormal(VLsample(8));
```



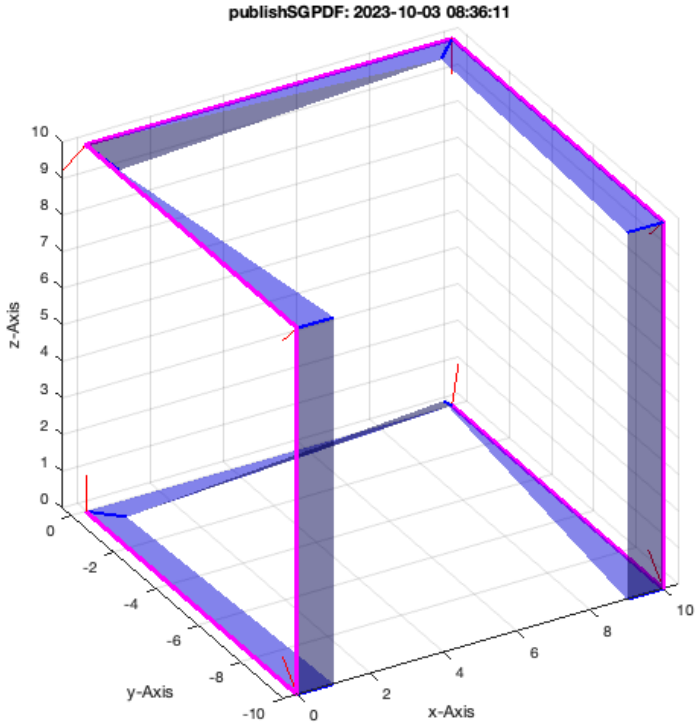

```
VLedgeNormal(VLsample(12));
```



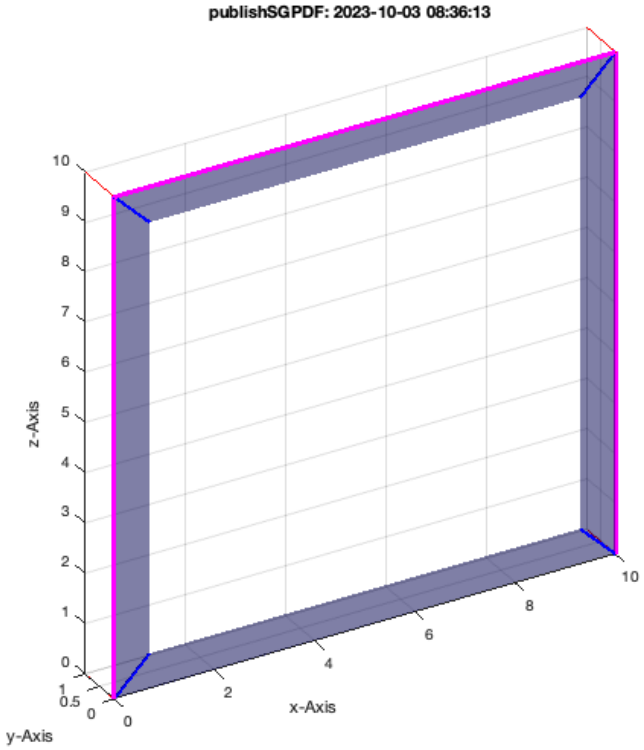

```
VLedgeNormal(VLsample(13));
```



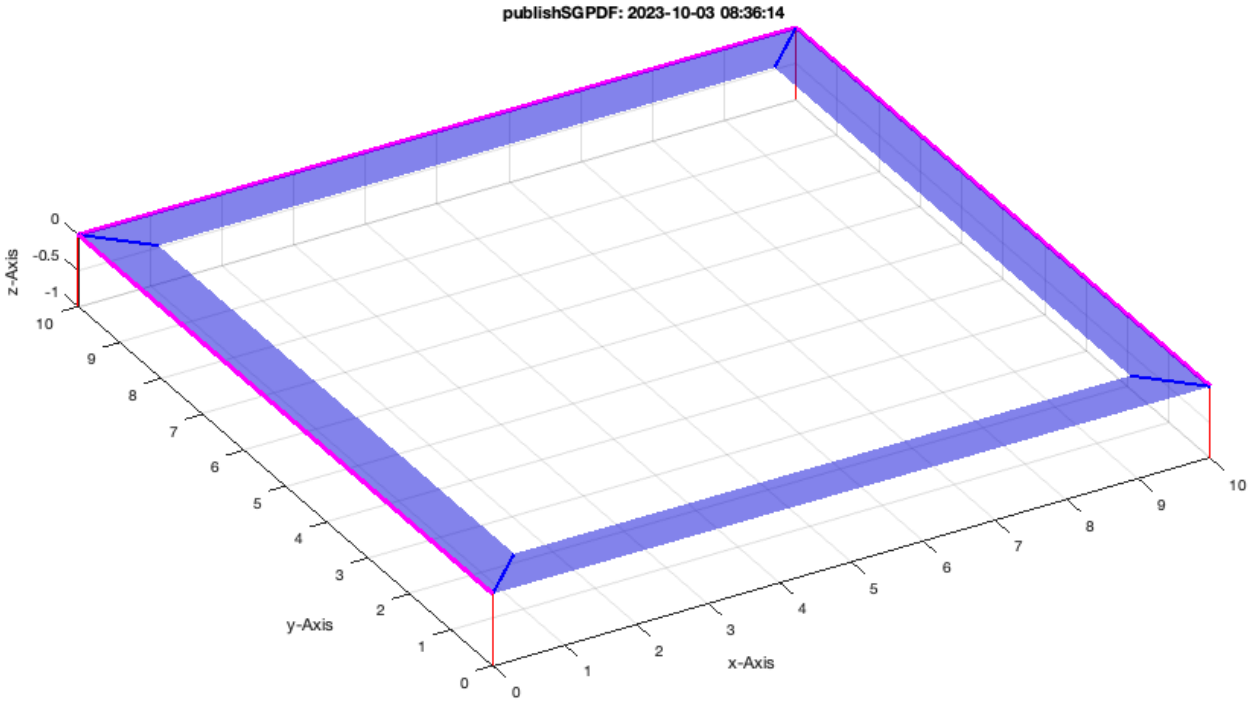

```
VLedgeNormal(VLsample(14));
```



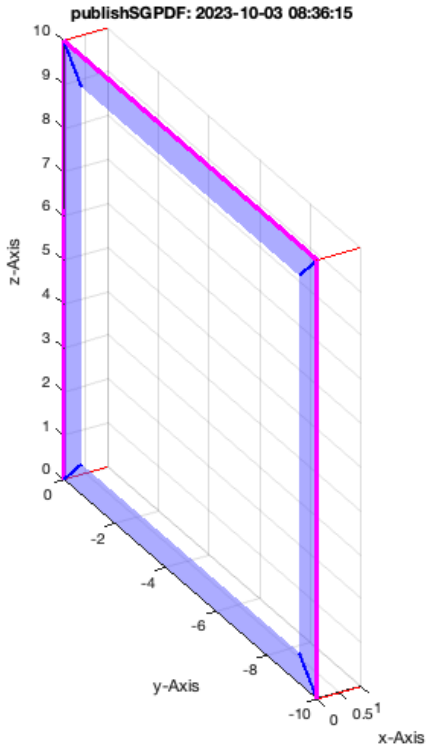

```
VLedgeNormal(VLsample(20));
```




```
VLedgeNormal(VLsample(21));
```



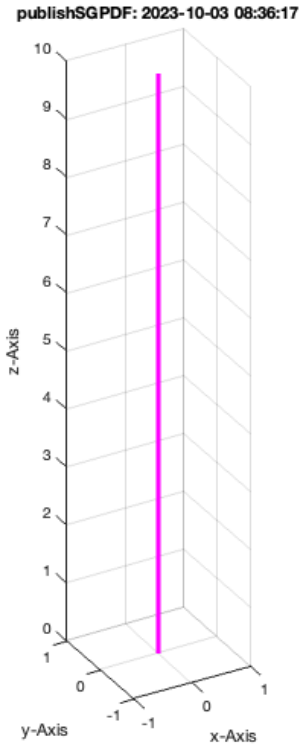

```
VLedgeNormal(VLsample(22));
```



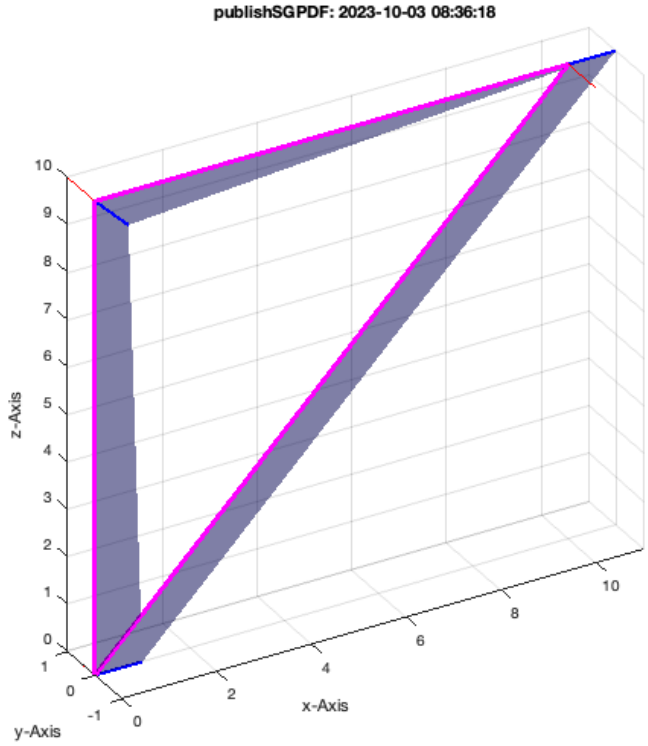
Creating normal function for a closed spatial curve

If angles are larger than 90 degree (pi/2)

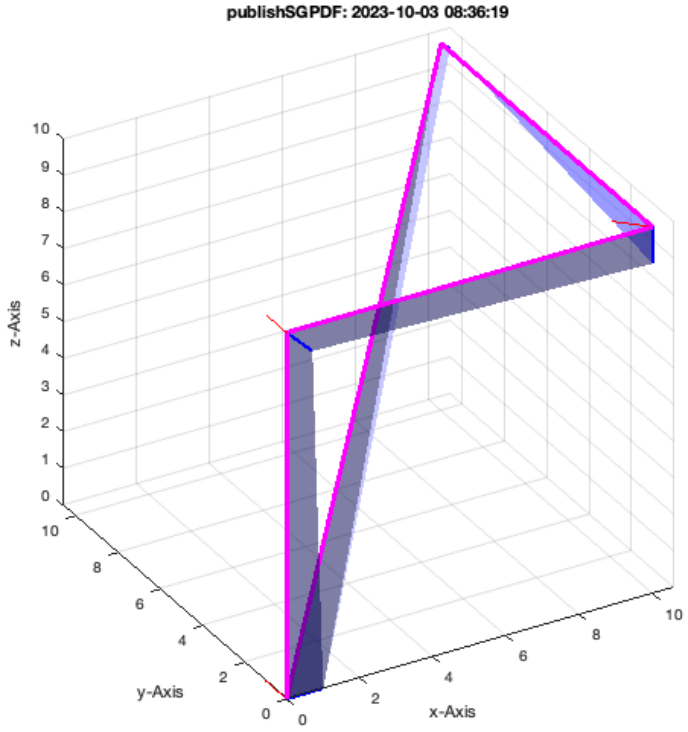
```
VLedgeNormal(CVLofVL(VLsample(2)));
```



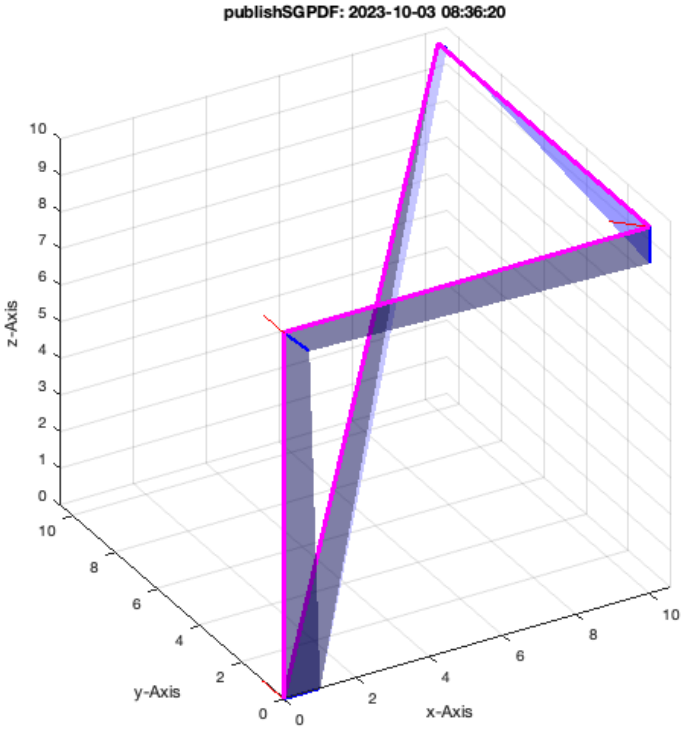

```
VLedgeNormal(CVLofVL(VLsample(3)));
```



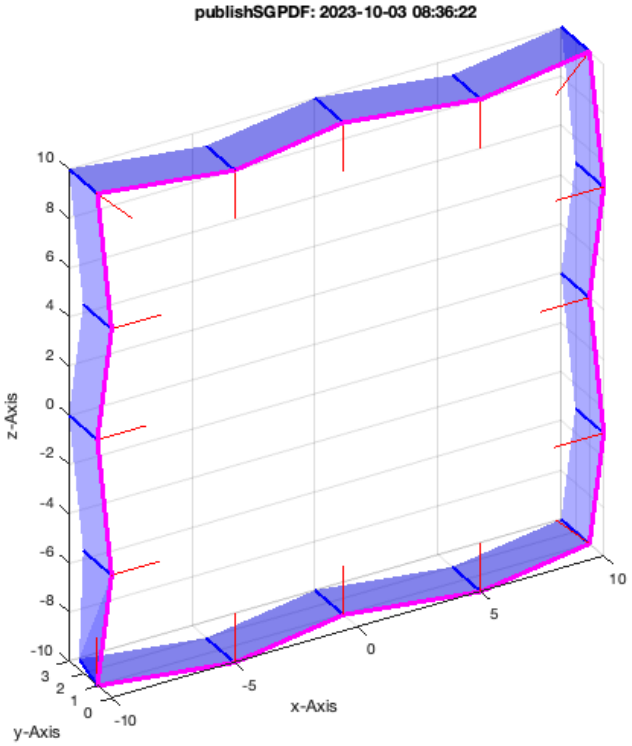

```
VLedgeNormal(CVLofVL(VLsample(7)));
```



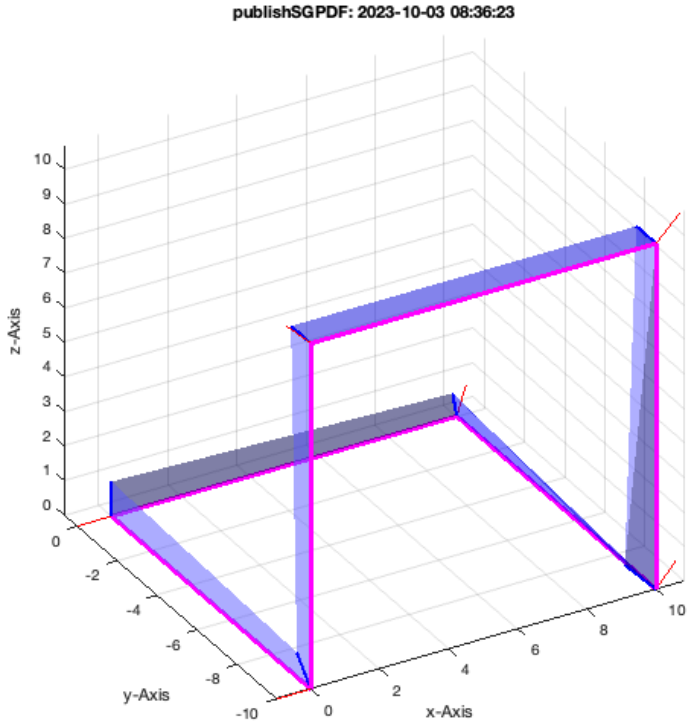

```
VLedgeNormal(CVLofVL(VLsample(8)));
```



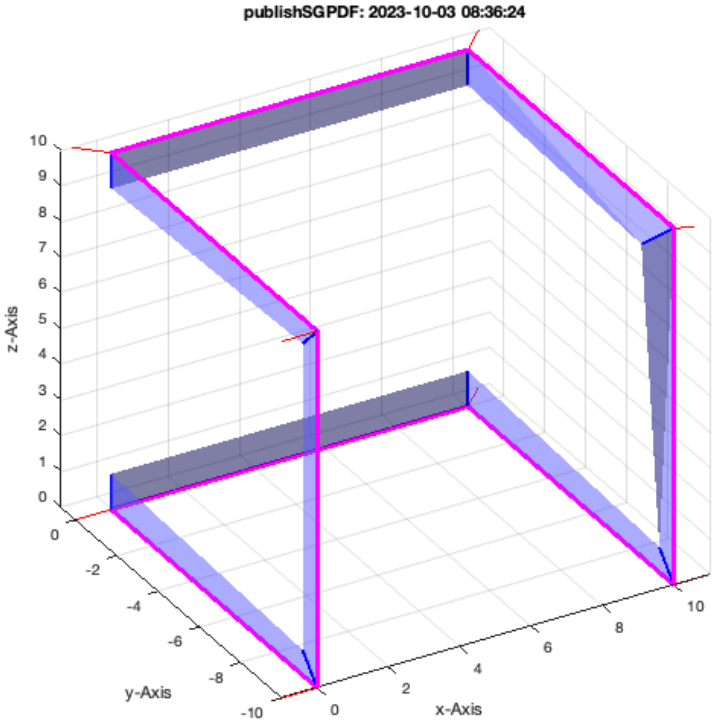

```
VLedgeNormal(CVLofVL(VLsample(12)));
```



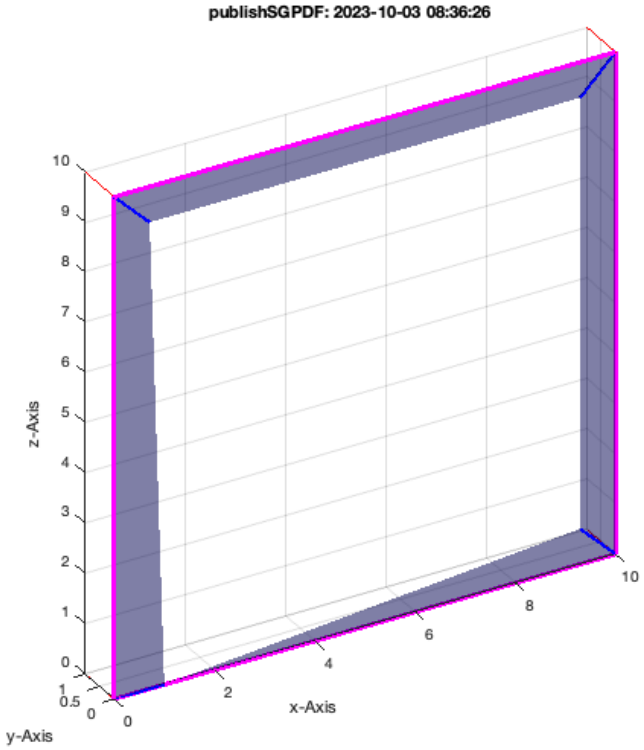

```
VLedgeNormal(CVLofVL(VLsample(13)));
```



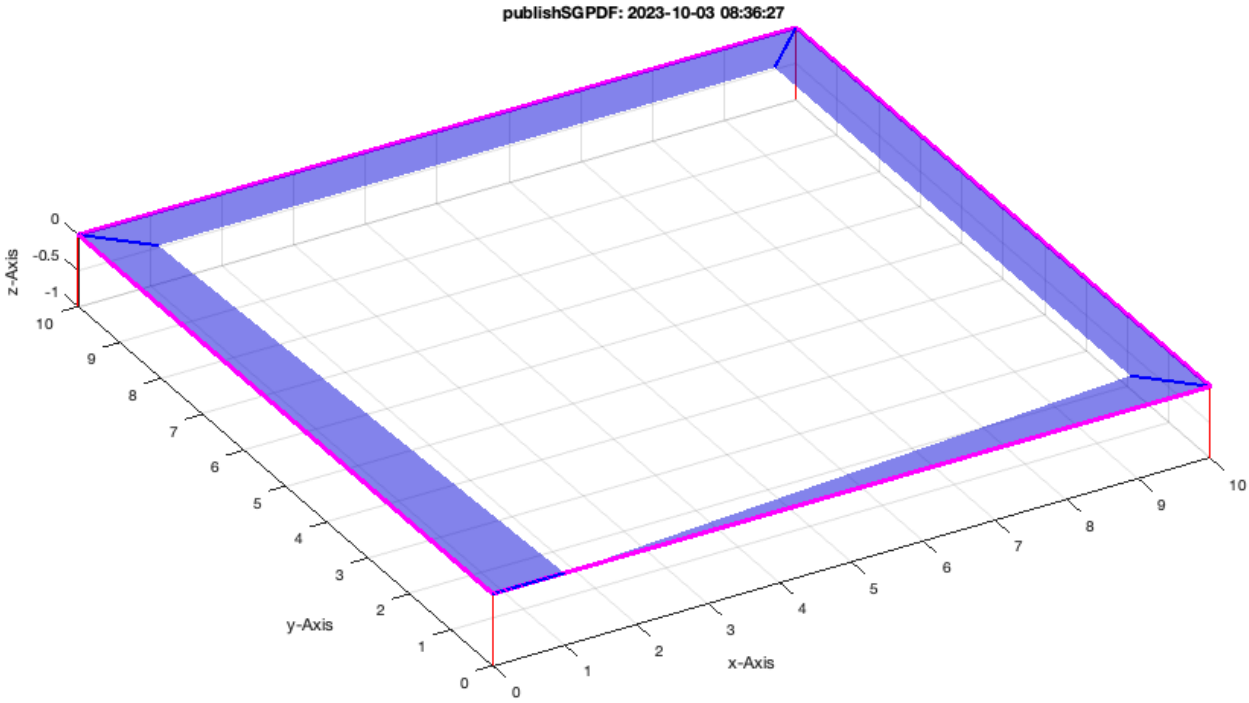

```
VLedgeNormal(CVLofVL(VLsample(14)));
```



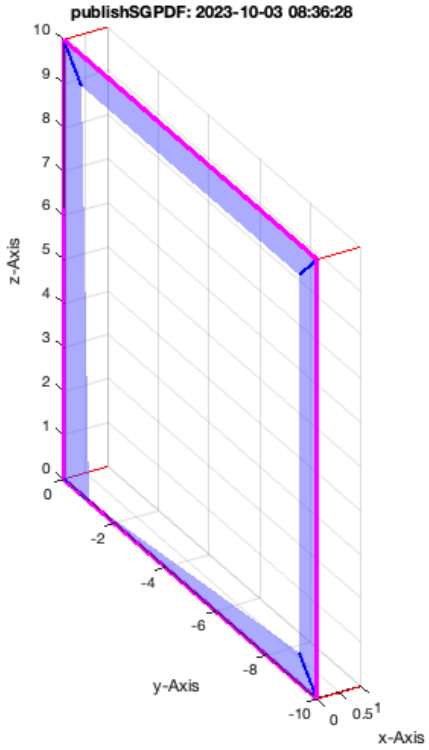

```
VLedgeNormal(CVLofVL(VLsample(20)));
```




```
VLedgeNormal(CVLofVL(VLsample(21)));
```



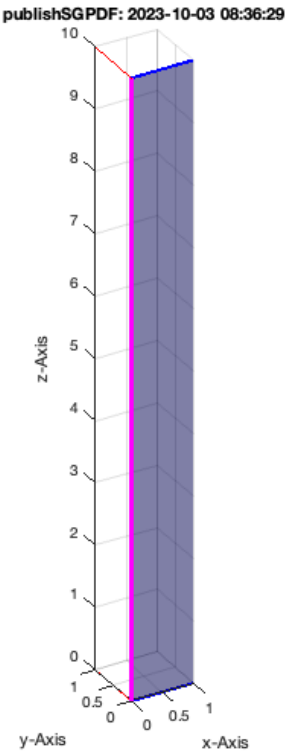

```
VLedgeNormal(CVLofVL(VLsample(22)));
```



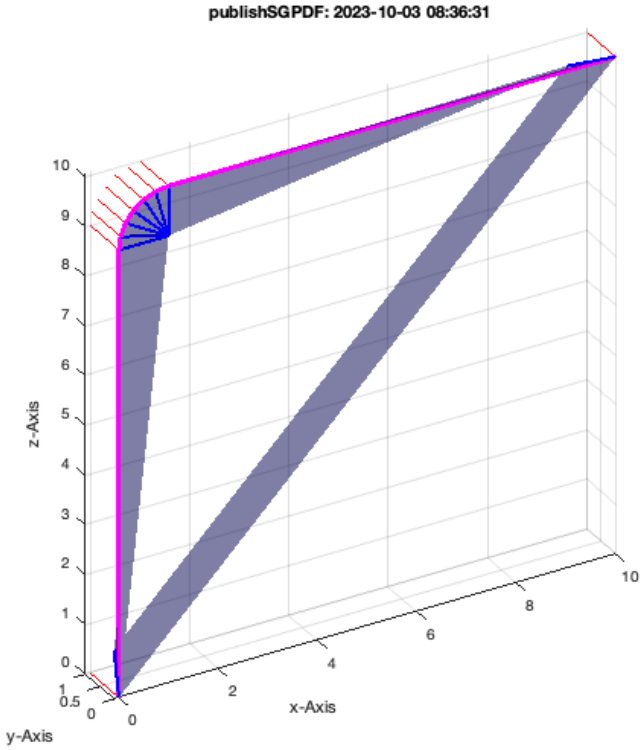
Creating normal function for open spatial radial curve

If angles are larger than 90 degree (pi/2)

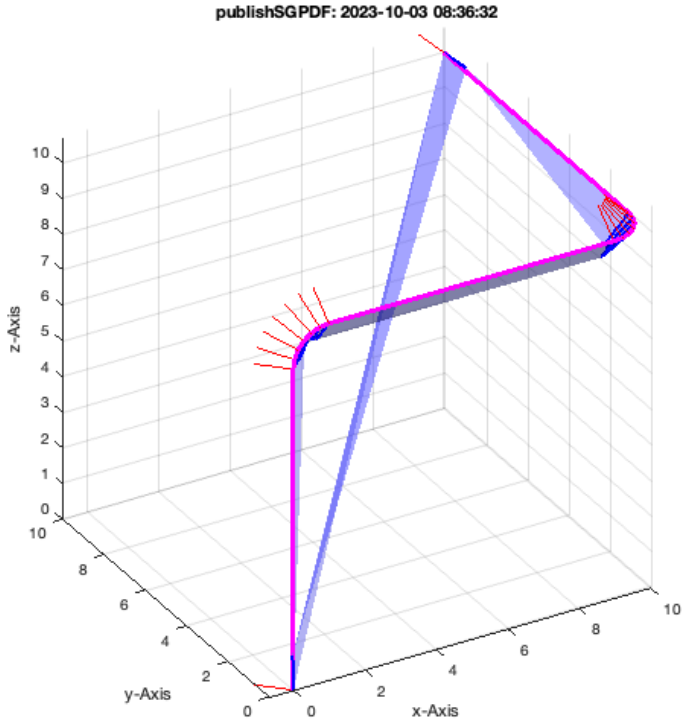
```
VLedgeNormal(VLradialEdges(VLsample(2)));
```



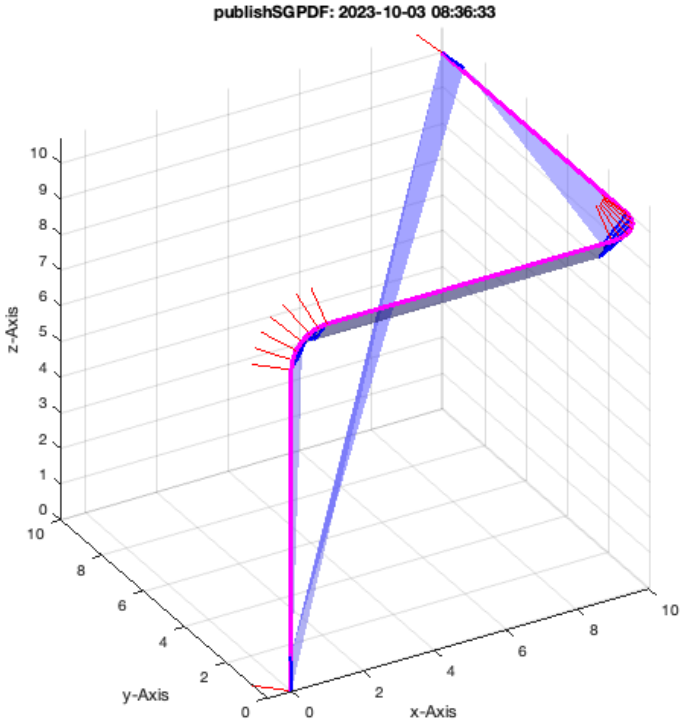

```
VLedgeNormal(VLradialEdges(VLsample(3)));
```



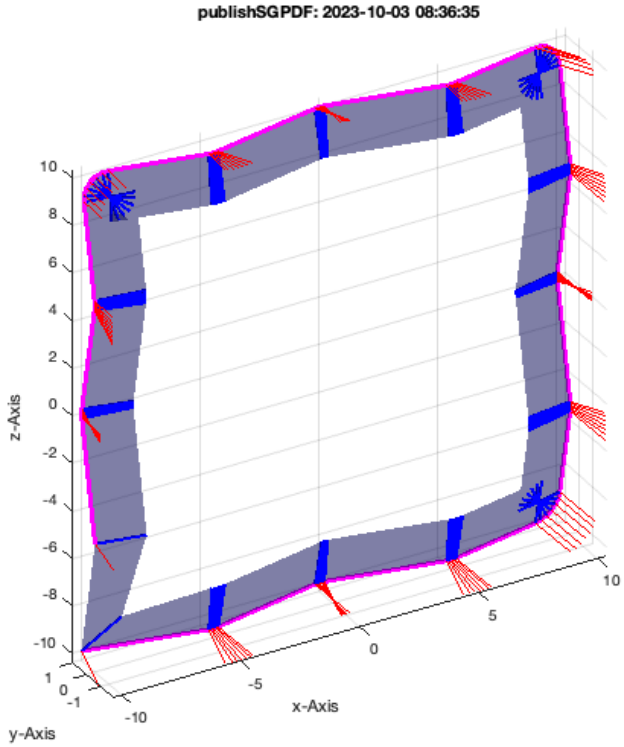

```
VLedgeNormal(VLradialEdges(VLsample(7)));
```



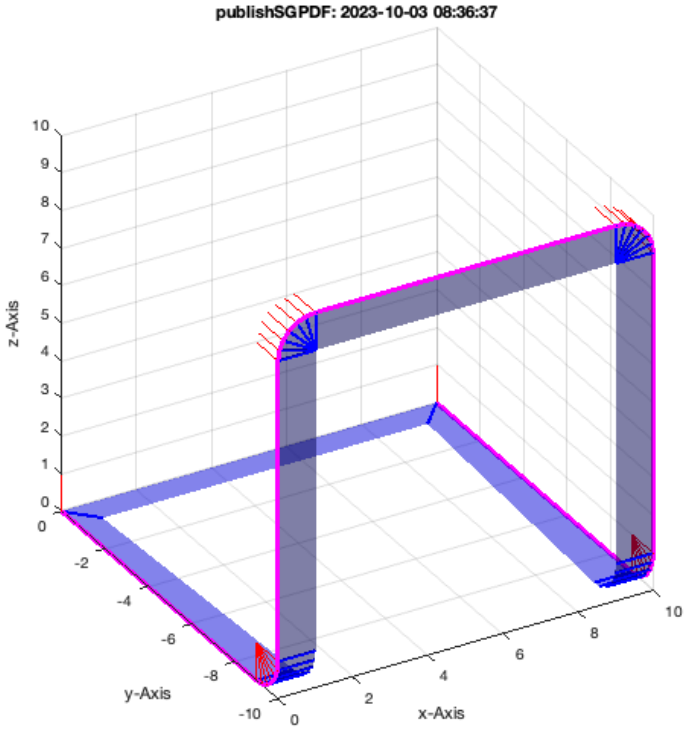

```
VLedgeNormal(VLradialEdges(VLsample(8)));
```



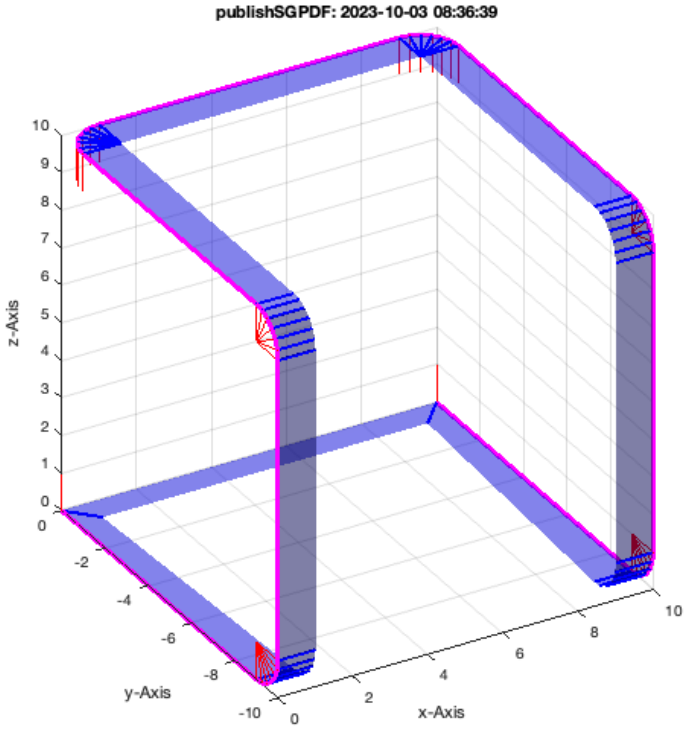

```
VLedgeNormal(VLradialEdges(VLsample(12)));
```



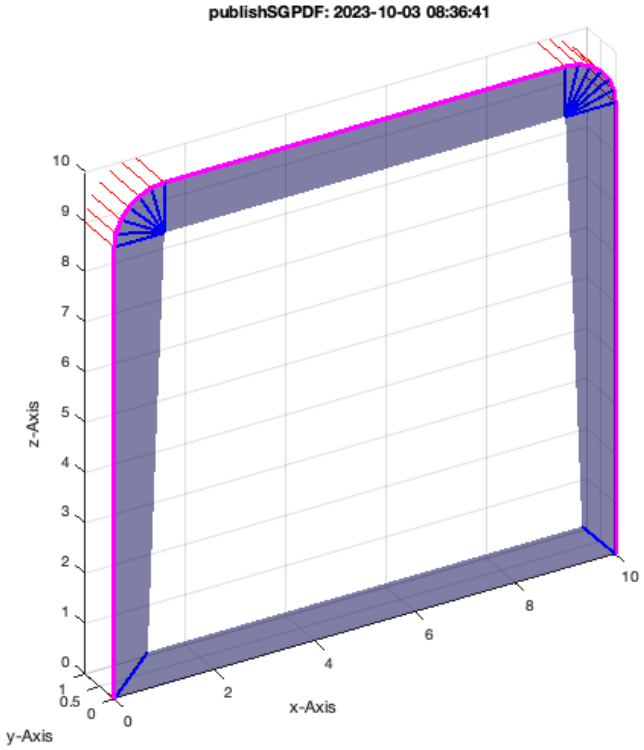

```
VLedgeNormal(VLradialEdges(VLsample(13)));
```



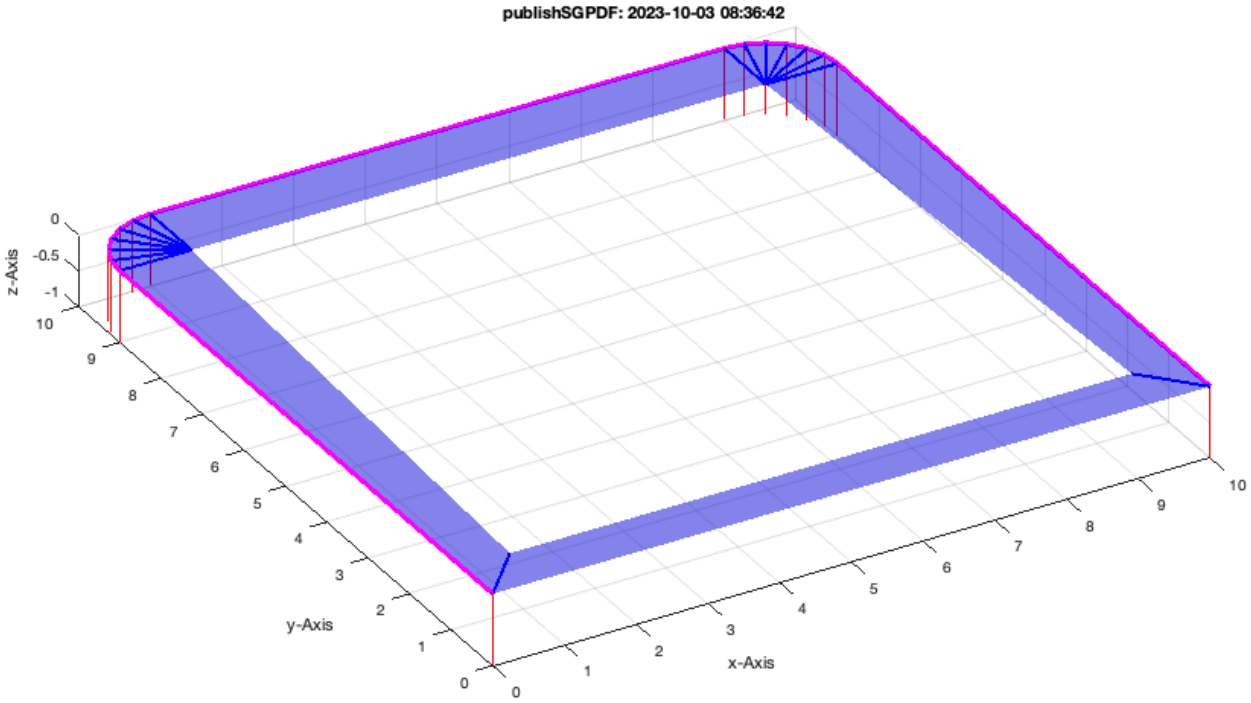

```
VLedgeNormal(VLradialEdges(VLsample(14)));
```



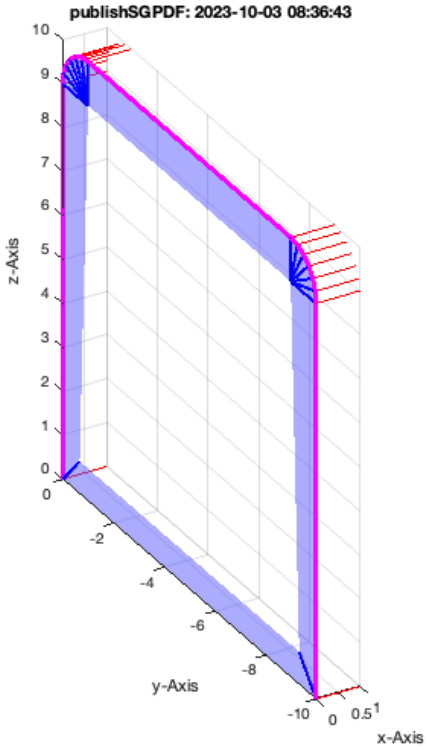

```
VLedgeNormal(VLradialEdges(VLsample(20)));
```




```
VLedgeNormal(VLradialEdges(VLsample(21)));
```



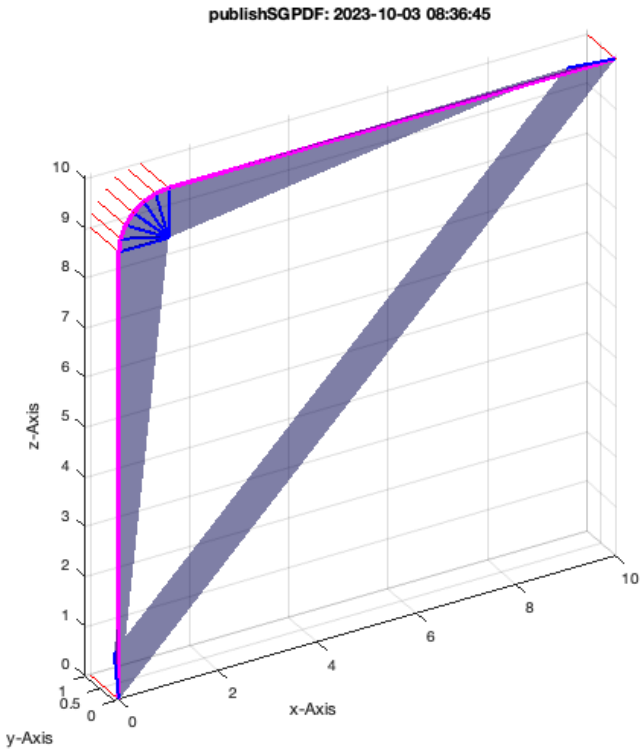

```
VLedgeNormal(VLradialEdges(VLsample(22)));
```



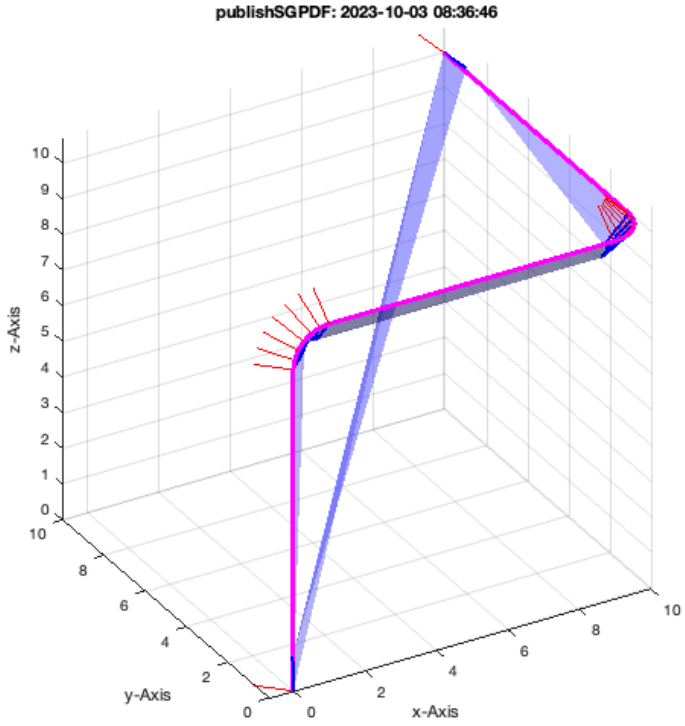
Creating normal function for closed spatial radial curve

If angles are larger than 90 degree (pi/2)

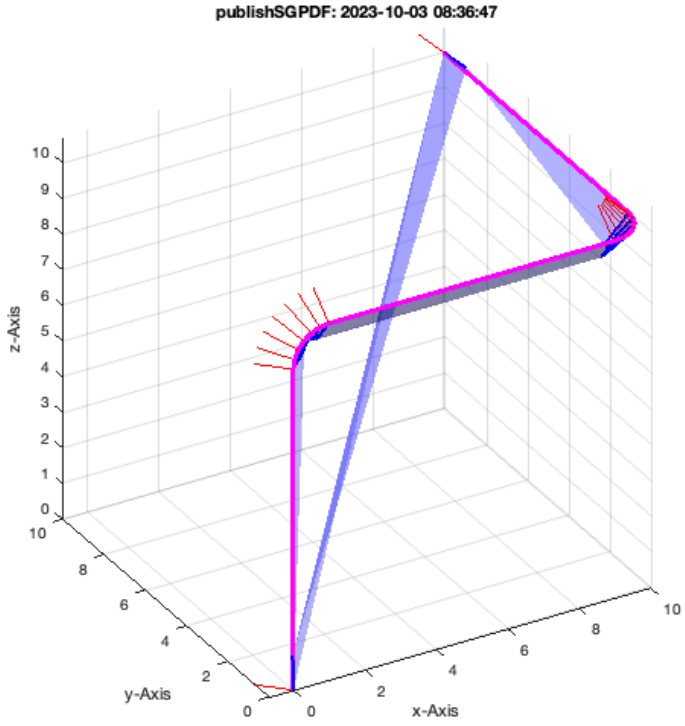
```
VLedgeNormal(VLradialEdges(CVLofVL(VLsample(3))));
```



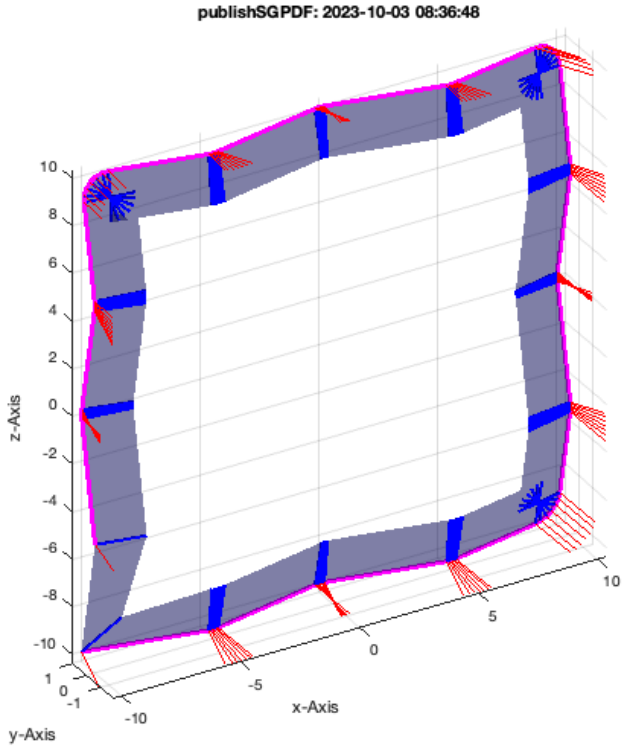

```
VLedgeNormal(VLradialEdges(CVLofVL(VLsample(7))));
```



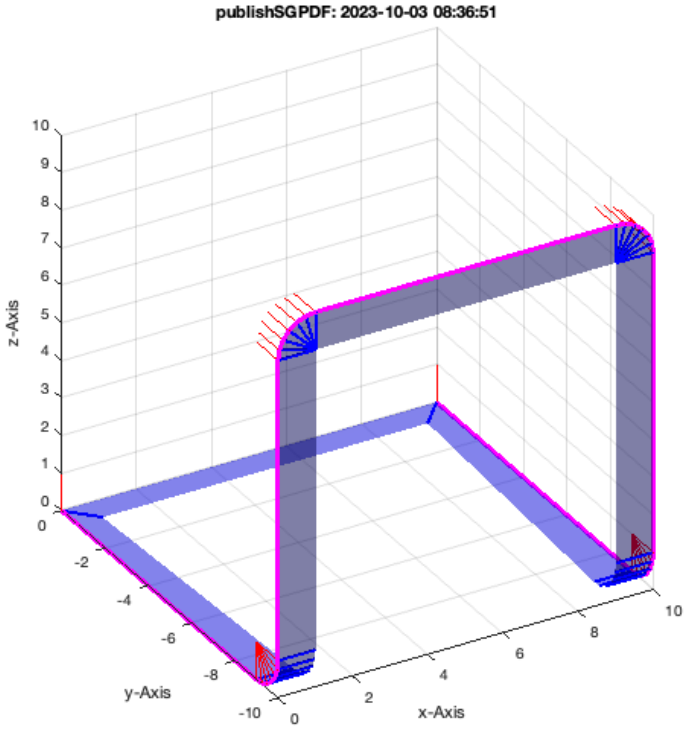

```
VLedgeNormal(VLradialEdges(CVLofVL(VLsample(8))));
```



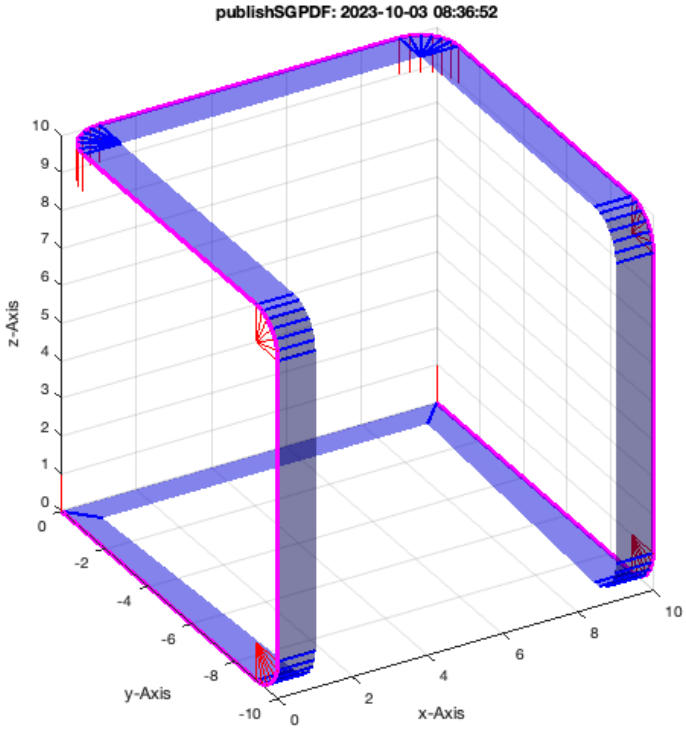

```
VLedgeNormal(VLradialEdges(CVLofVL(VLsample(12))));
```



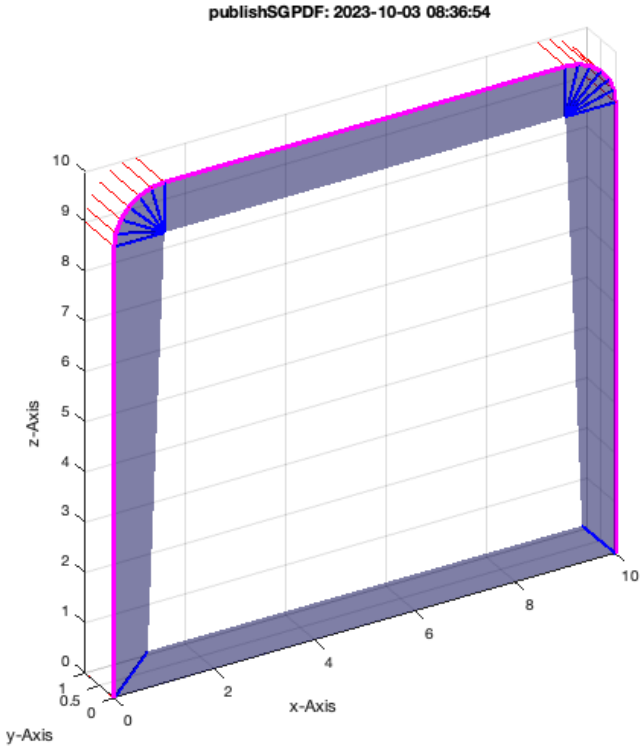

```
VLedgeNormal(VLradialEdges(CVLofVL(VLsample(13))));
```



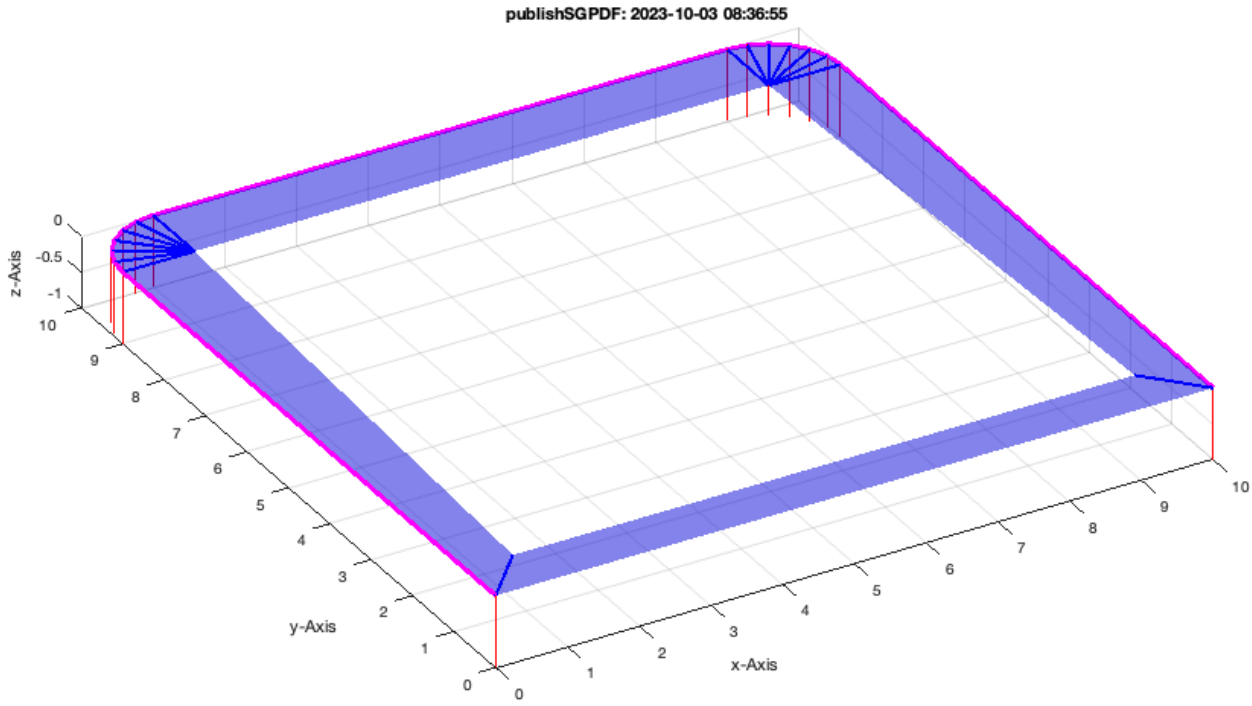

```
VLedgeNormal(VLradialEdges(CVLofVL(VLsample(14))));
```



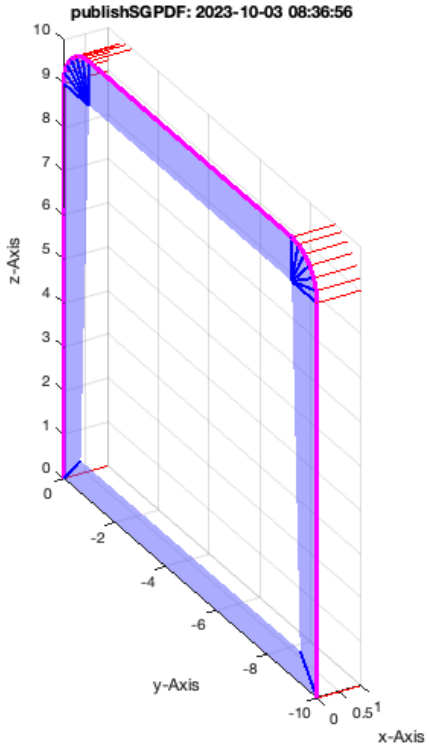

```
VLedgeNormal(VLradialEdges(CVLofVL(VLsample(20))));
```




```
VLedgeNormal(VLradialEdges(CVLofVL(VLsample(21))));
```



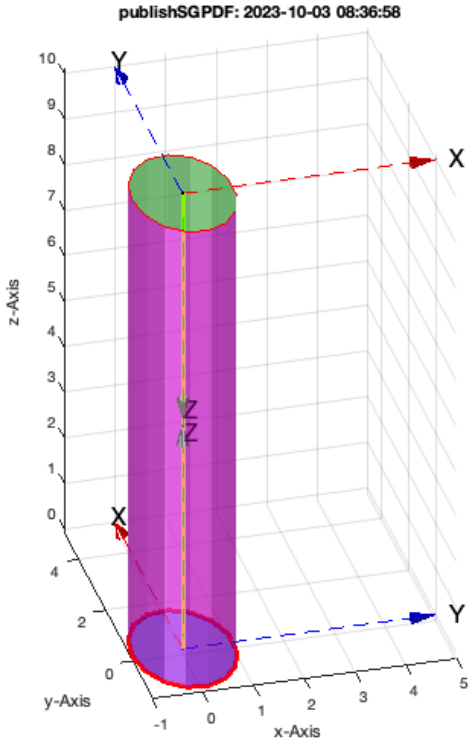

```
VLedgeNormal(VLradialEdges(CVLofVL(VLsample(22))));
```



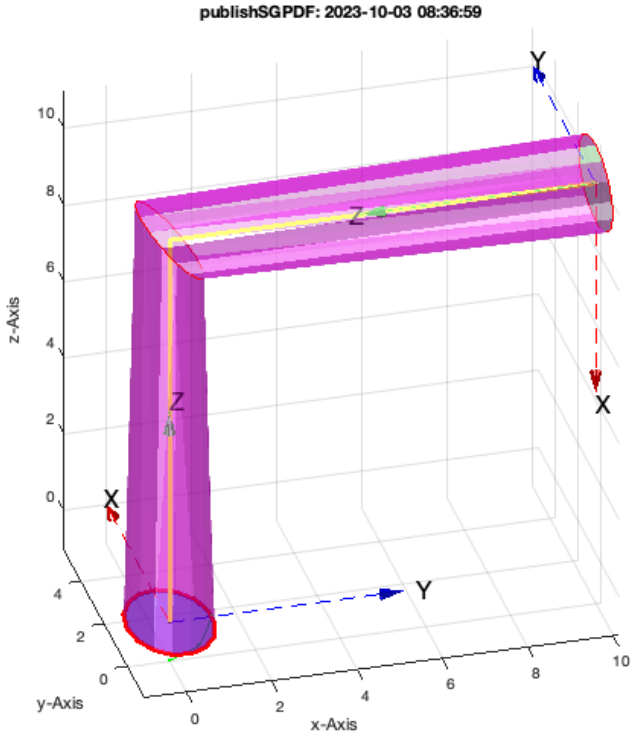
Creating Solid Geometries open

If angles are larger than 90 degree (pi/2)

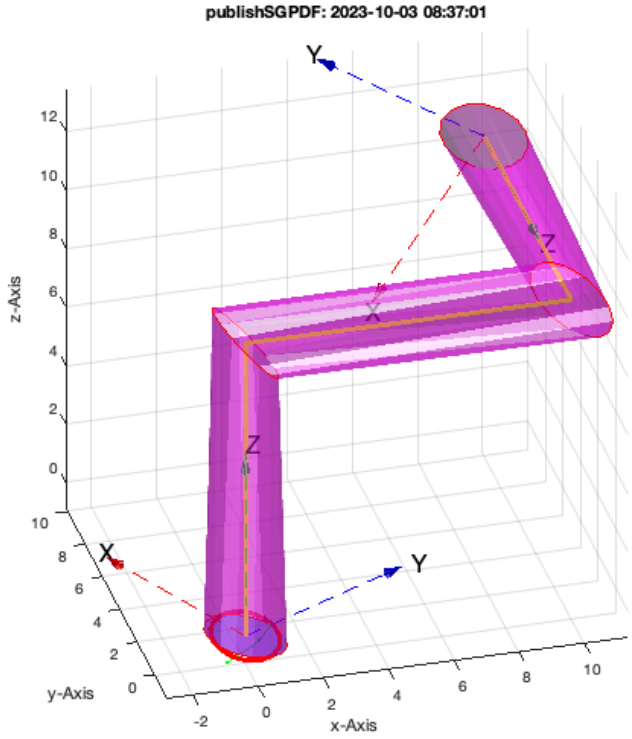
```
SGcontourtube2(PLcircle(1,'','',1.5),VLsample(2));
```



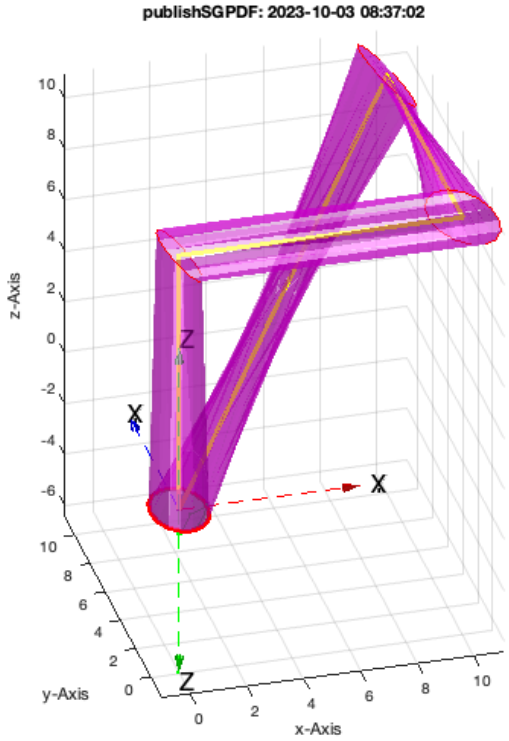

```
SGcontourtube2(PLcircle(1,'','',1.5),VLsample(3));
```



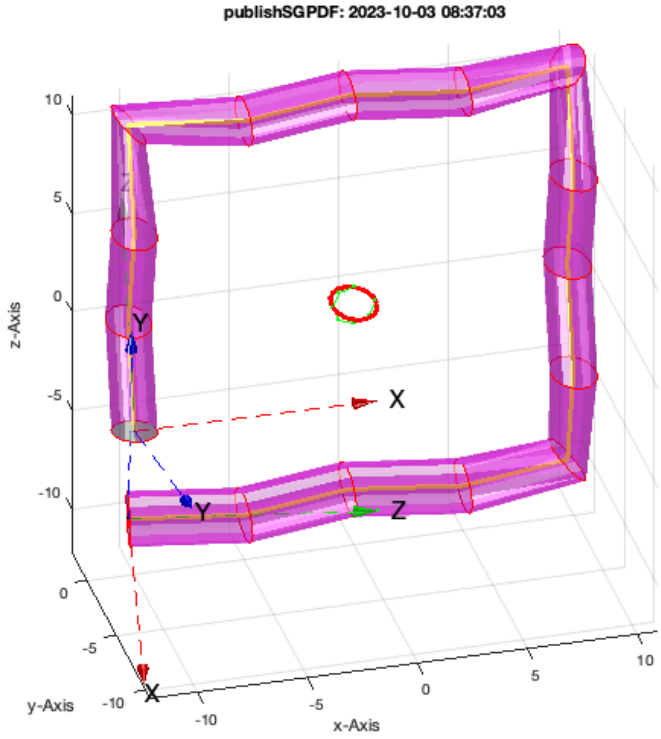

```
SGcontourtube2(PLcircle(1,'','',1.5),VLsample(7));
```



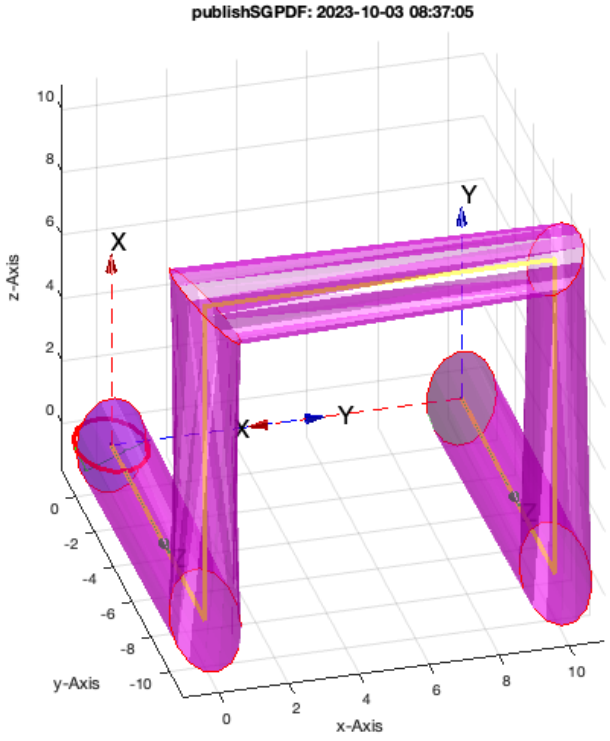

```
SGcontourtube2(PLcircle(1,'','',1.5),VLsample(8));
```



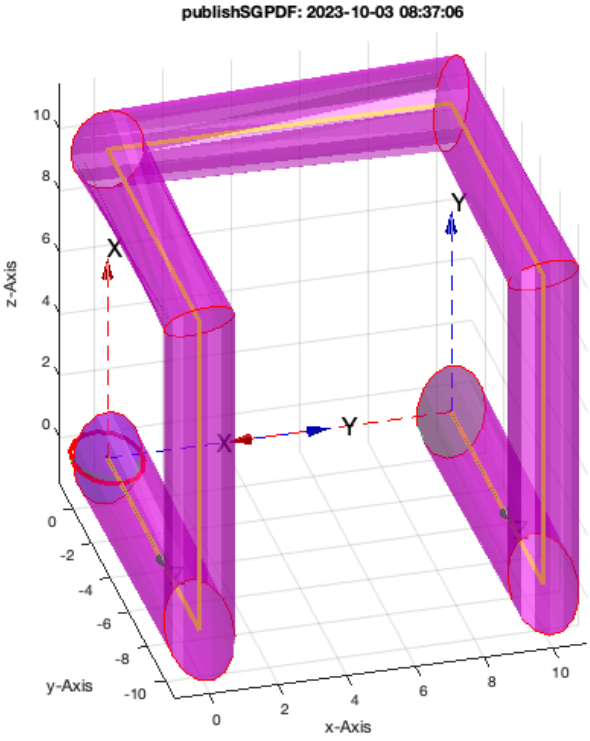

```
SGcontourtube2(PLcircle(1,'','',1.5),VLsample(12));
```



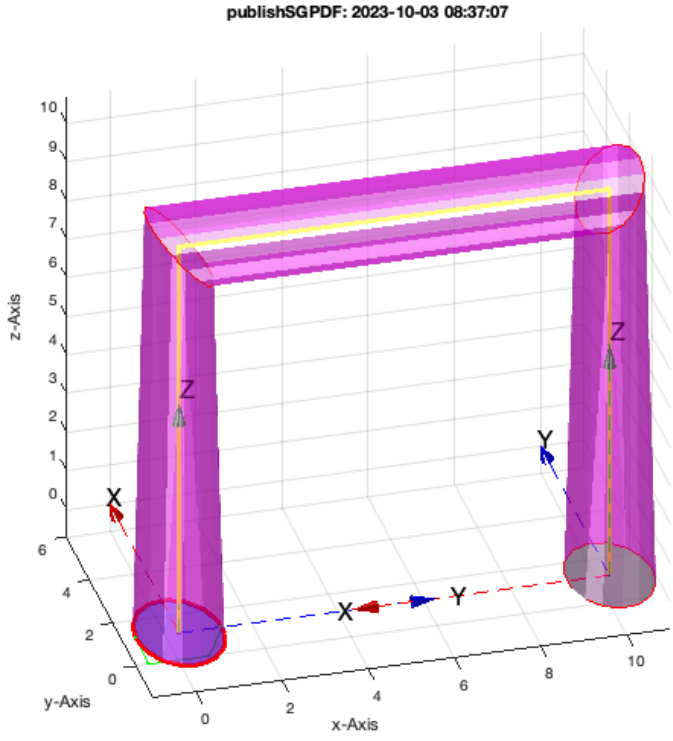

```
SGcontourtube2(PLcircle(1,'','',1.5),VLsample(13));
```



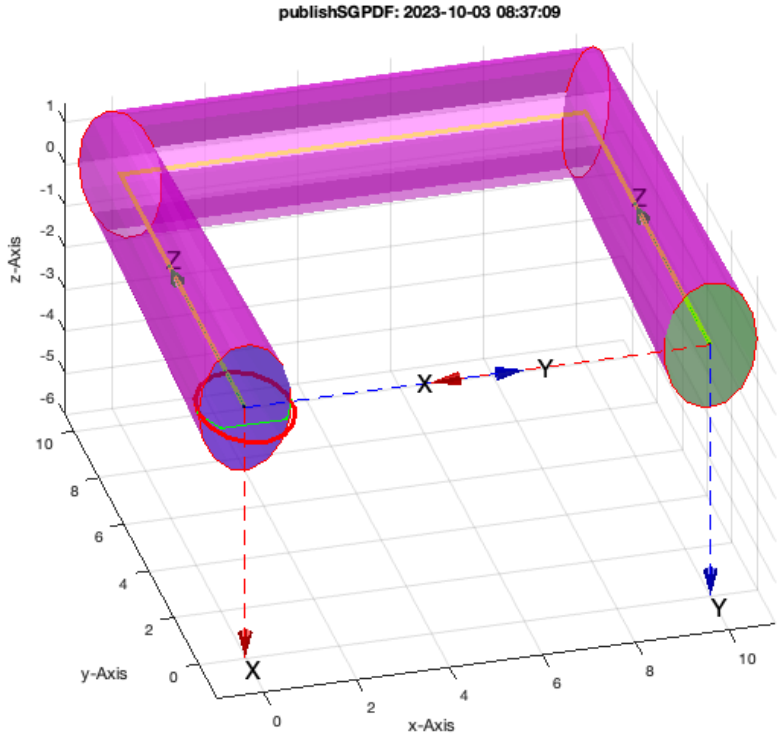

```
SGcontourtube2(PLcircle(1,'','',1.5),VLsample(14));
```



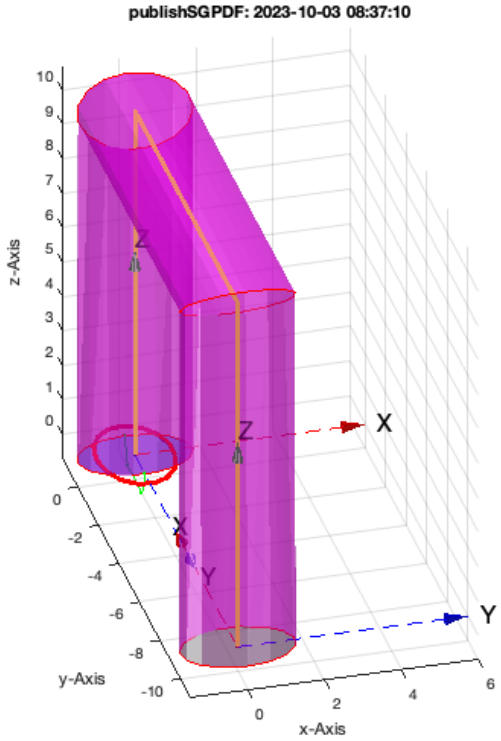

```
SGcontourtube2(PLcircle(1,'','',1.5),VLsample(20));
```




```
SGcontourtube2(PLcircle(1,'','',1.5),VLsample(21));
```



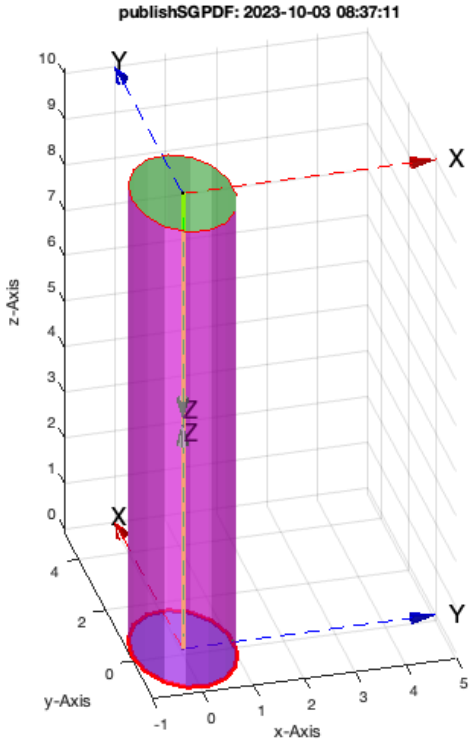

```
SGcontourtube2(PLcircle(1,'','',1.5),VLsample(22));
```



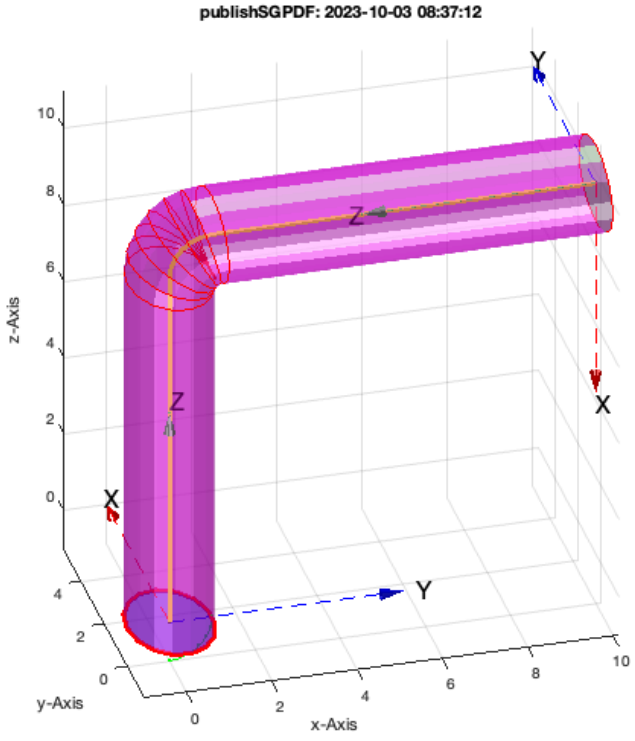
Creating Solid Geometries open

If angles are larger than 90 degree (pi/2)

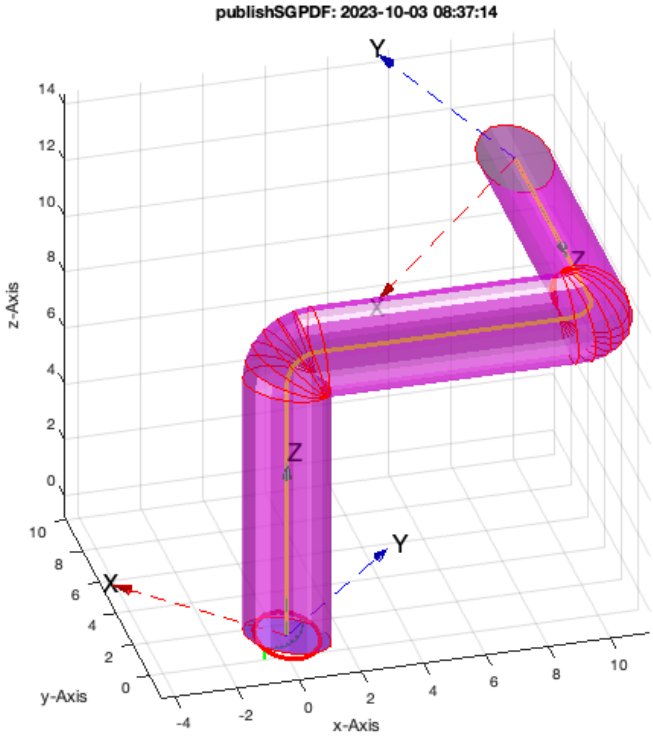
```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(VLsample(2)));
```



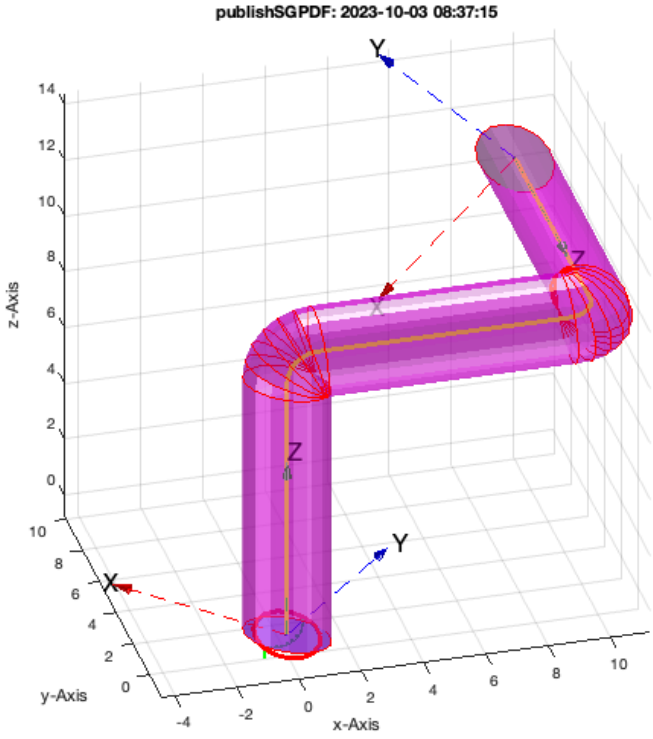

```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(VLsample(3)));
```



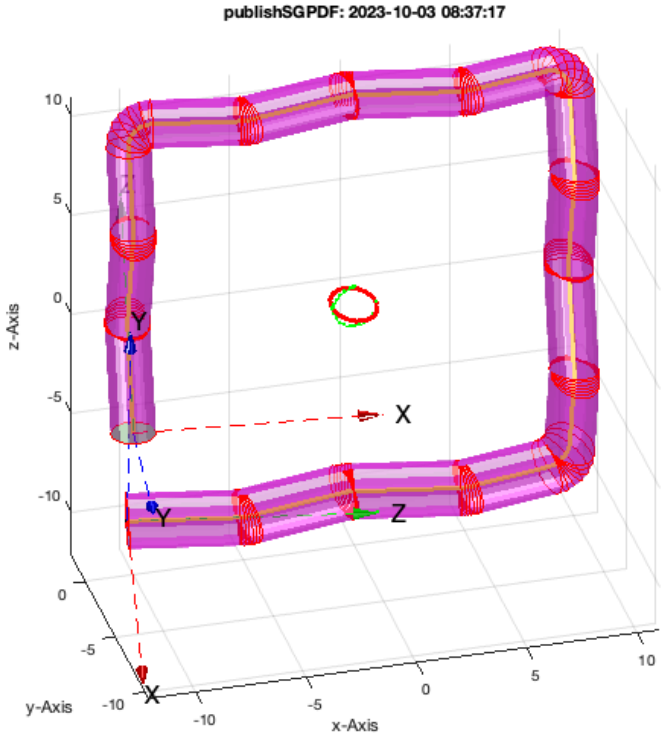

```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(VLsample(7)));
```



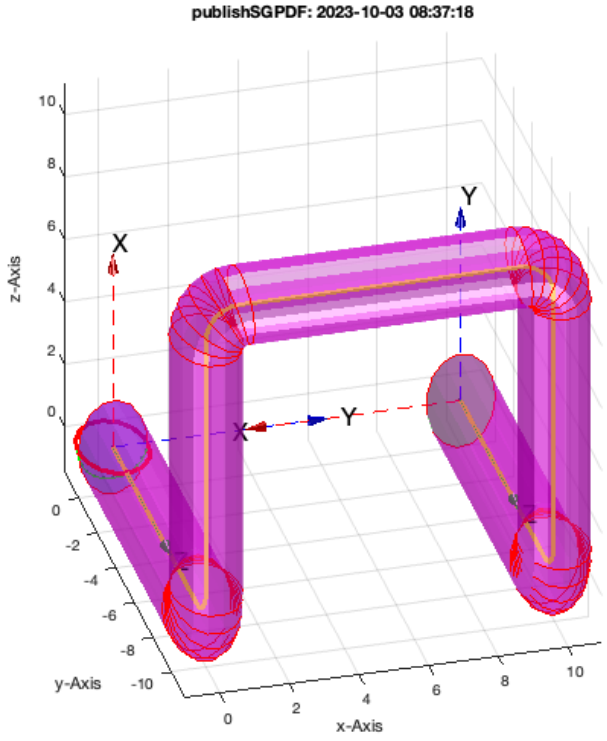

```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(VLsample(8)));
```



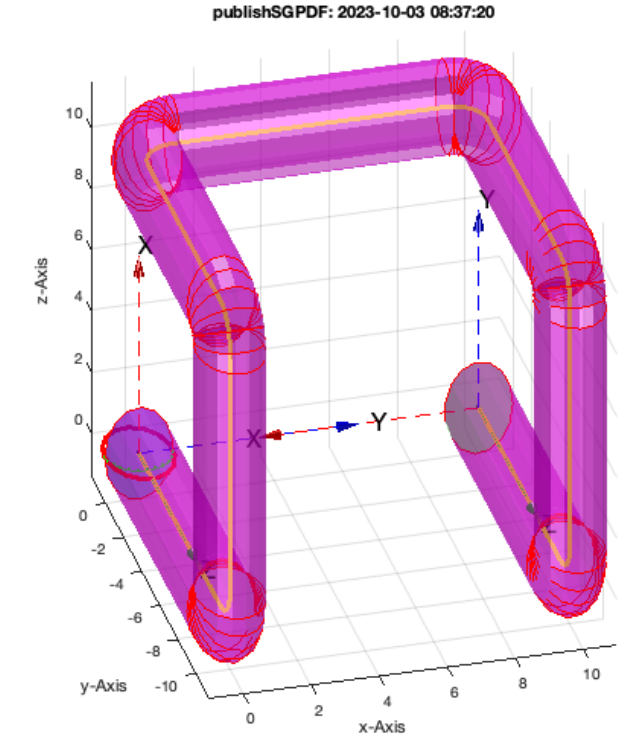

```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(VLsample(12)));
```



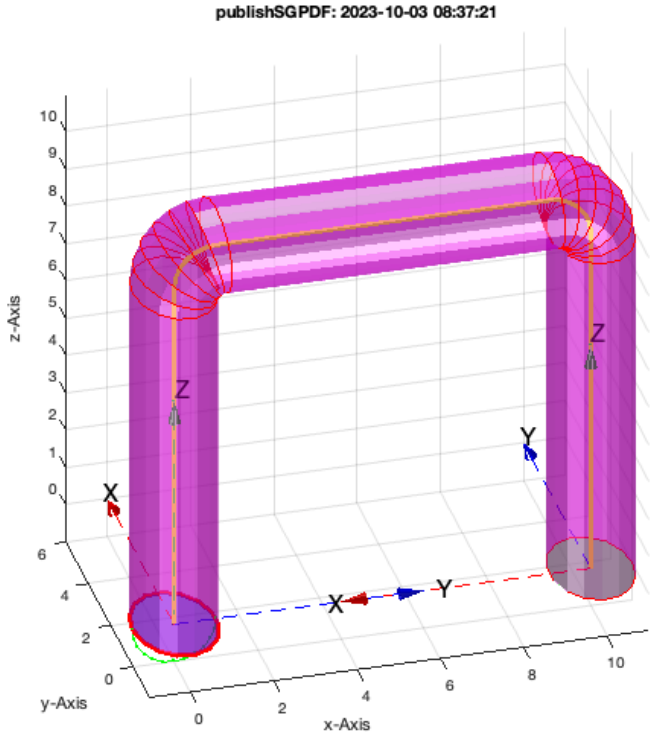

```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(VLsample(13)));
```



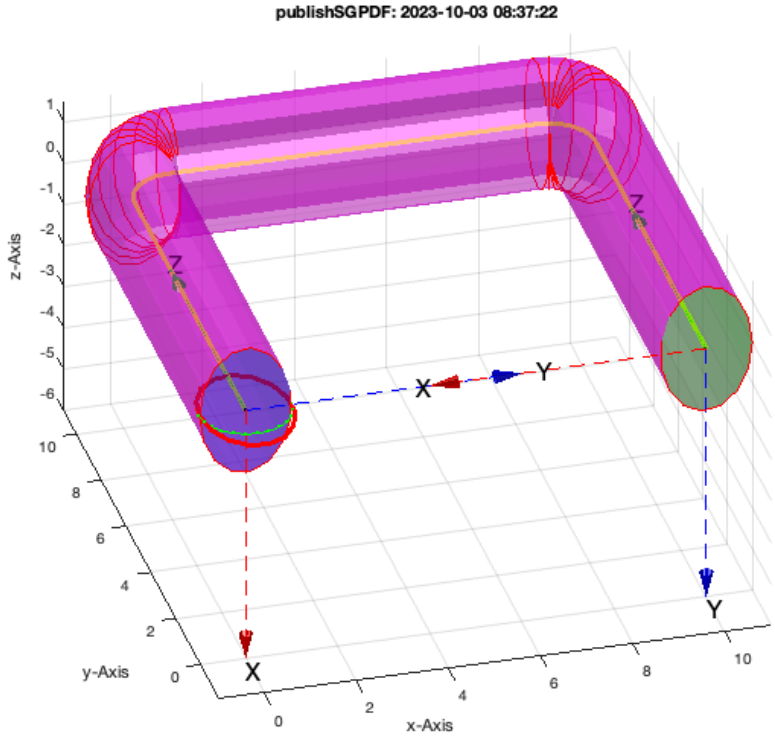

```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(VLsample(14)));
```



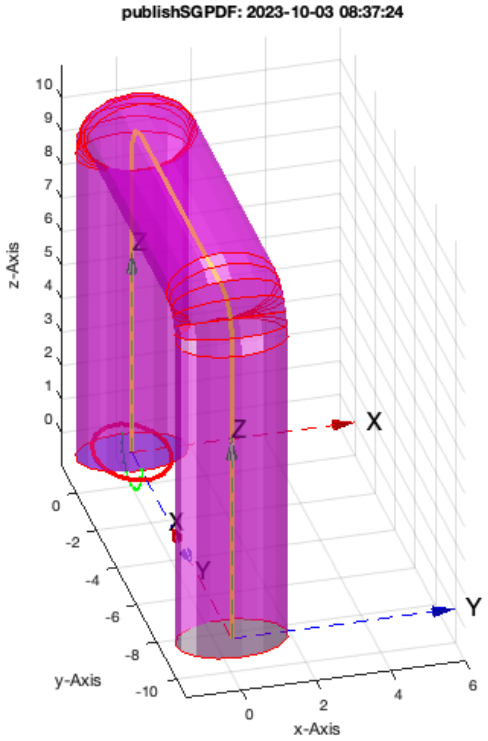

```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(VLsample(20)));
```




```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(VLsample(21)));
```



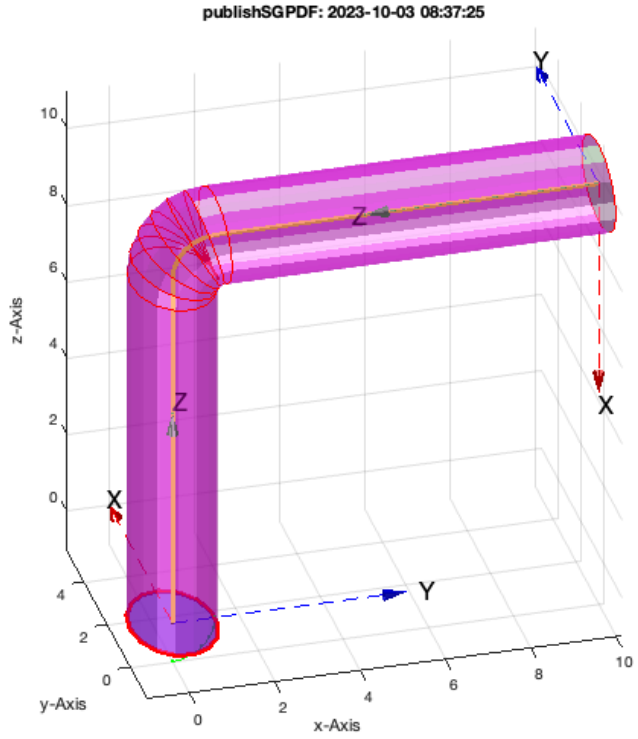

```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(VLsample(22)));
```



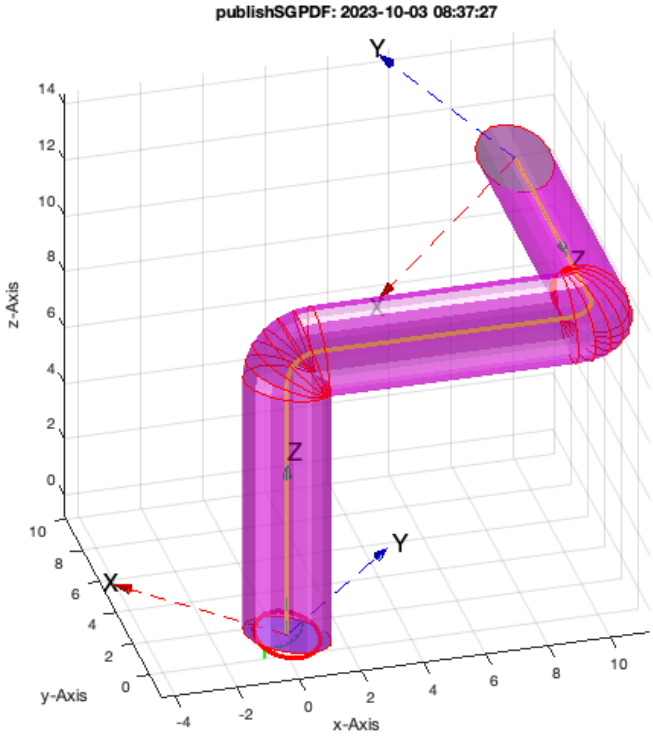
Creating Solid Geometries open

If angles are larger than 90 degree (pi/2)

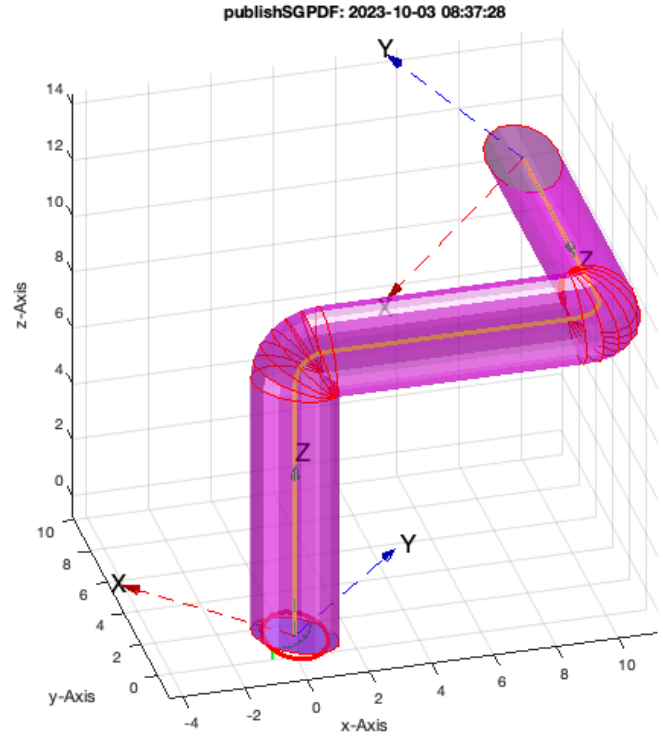
```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(CVLofVL(VLsample(3))));
```



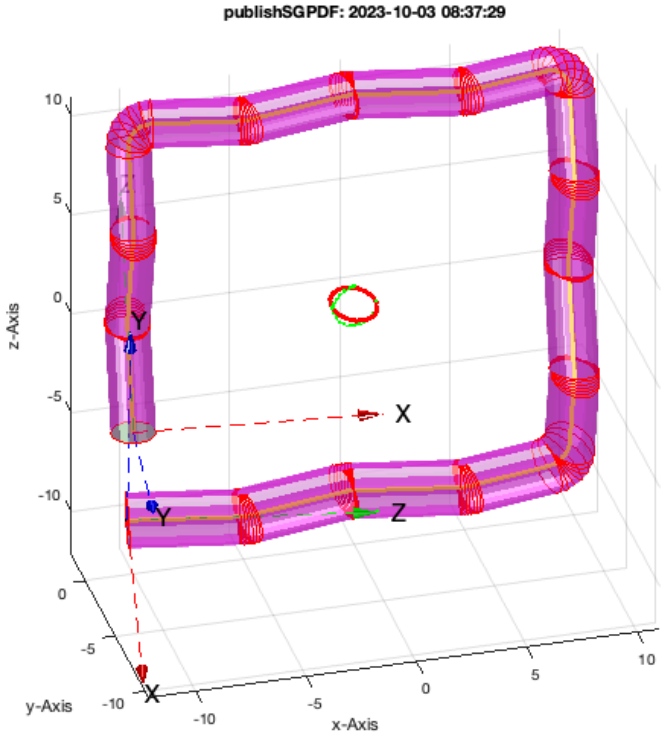

```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(CVLofVL(VLsample(7))));
```



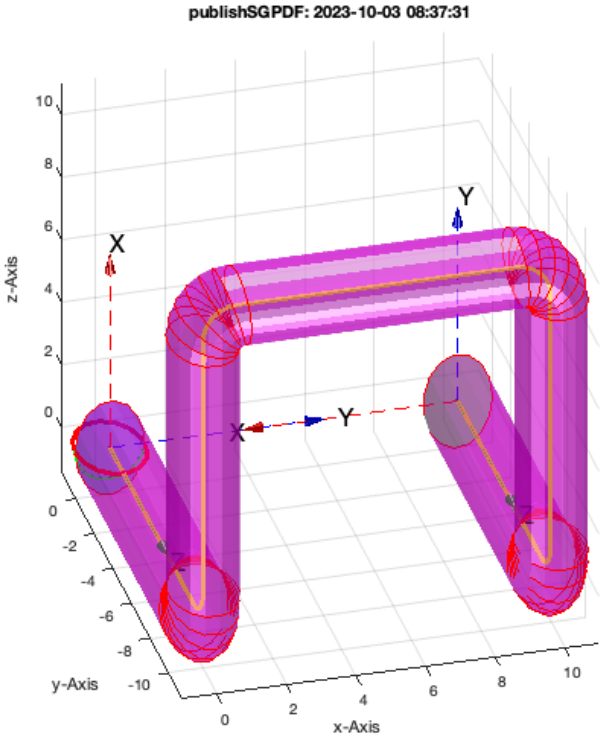

```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(CVLofVL(VLsample(8))));
```



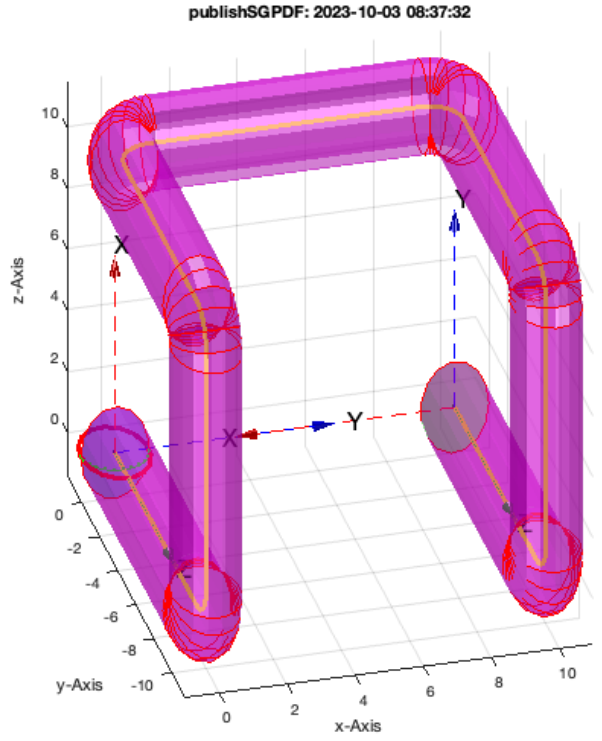

```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(CVLofVL(VLsample(12))));
```



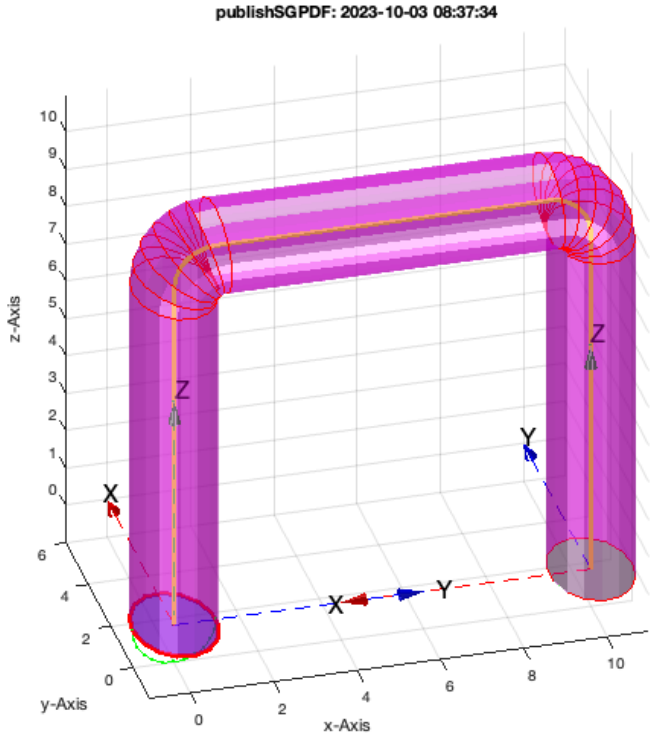

```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(CVLofVL(VLsample(13))));
```



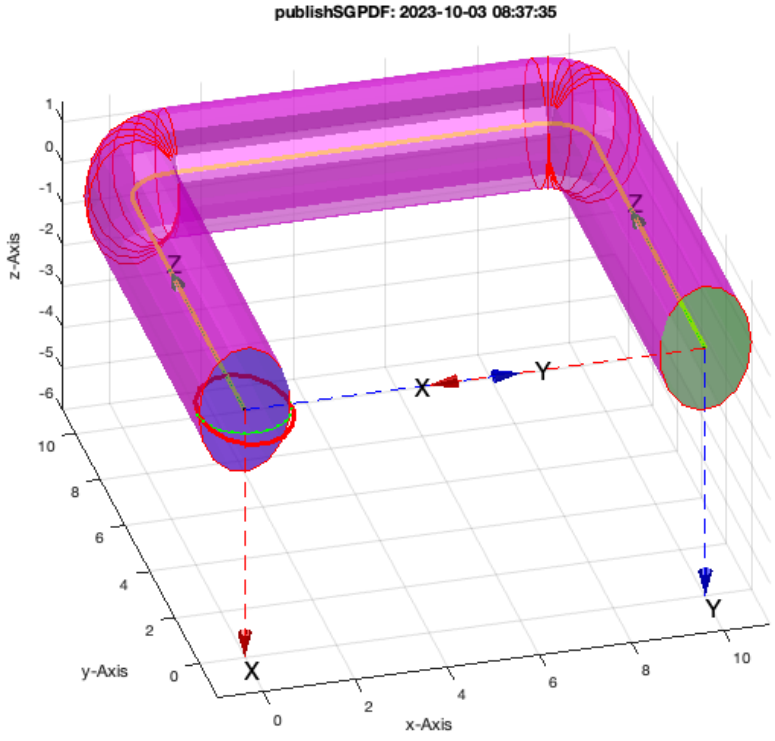

```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(CVLofVL(VLsample(14))));
```



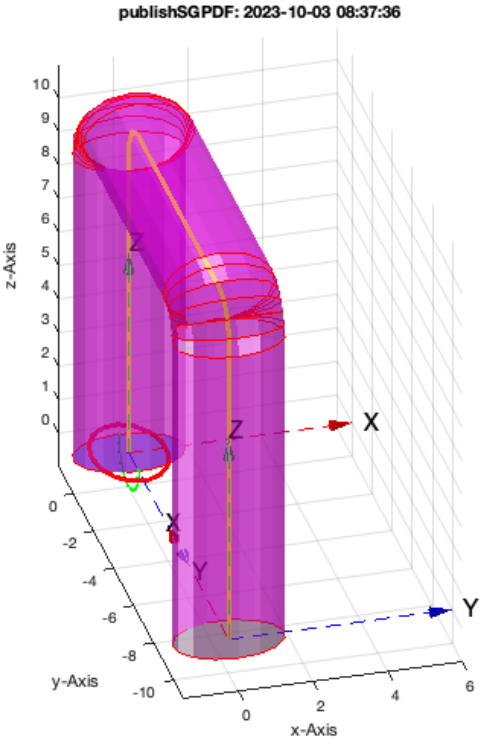

```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(CVLofVL(VLsample(20))));
```



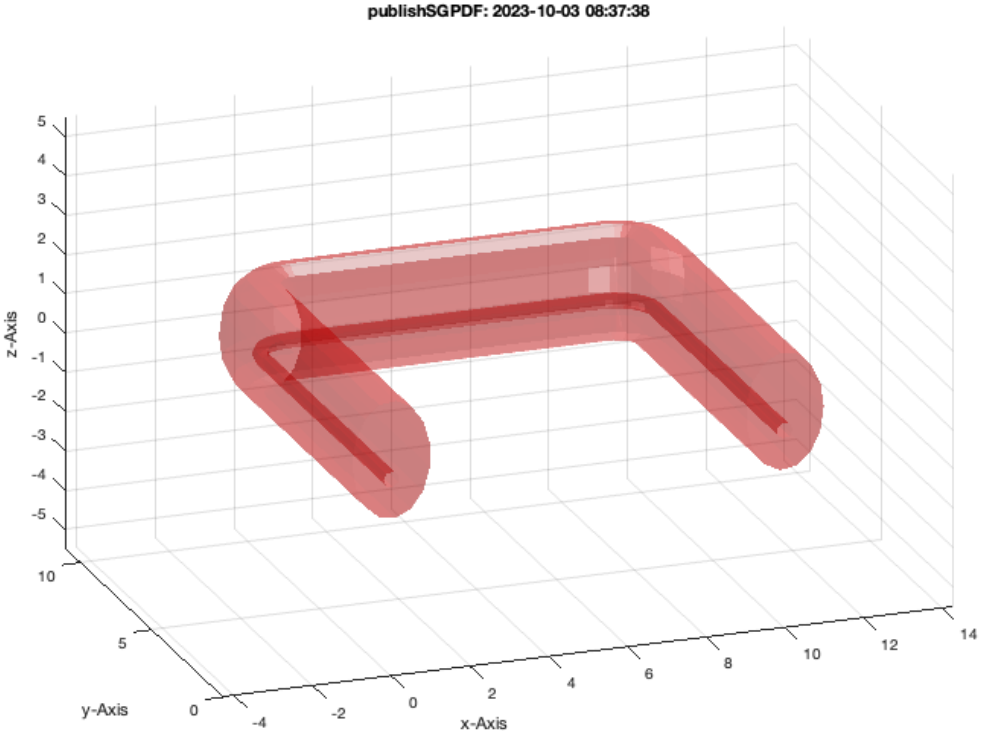

```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(CVLofVL(VLsample(21))));
```




```
SGcontourtube2(PLcircle(1,'','',1.5),VLradialEdges(CVLofVL(VLsample(22))));
```




```
SGcontourtube2([PLcircle(1,'','',1.5);NaN NaN;PLcircle(0.2)+[0 0.5]],VLradialEdges(VLsample(21))); SG=ans;
SGfigure(SG);VLFLplotlight(1,0.3); view(-20,20);
```



1. Conversion between triangle surface model and tetrahedon volumen model

Final Remarks

```
close all
```

```
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 08:37:39!
Executed 03-Oct-2023 08:37:41 by 'timlueth' on a MACI64 using Mac OSX 13.6 | R2023a Update 5 | SG-Lib 5.4
======================================= Used Matlab products: =======================================
database_toolbox
distrib_computing_toolbox
fixed_point_toolbox
image_toolbox
map_toolbox
matlab
pde_toolbox
simmechanics
simscape
simulink
=================================================================================================
```

Published with MATLAB® R2023a