Tutorial 53: SKOL - Soft Kill Option for Large Displacement by Yilun Sun

2020-07-28: Tim C. Lueth, Professor at Technische Universität München, Germany (URL: http://www.SG-Lib.org) - Last Change: 2020-08-07

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: Creation of Kinematic Chains and Robot Structures
- Tutorial 36: Creating a Patient-Individual Arm-Skin Protector-Shell
- Tutorial 37: Dimensioning of STL Files and Surface Data
- Tutorial 38: Some more solid geometry modelling function
- Tutorial 39: HEBO Modules robot design
- Tutorial 40: JACO Robot Simulation and Control
- Tutorial 41: Inserting Blades, Cuts and Joints into Solid Geometries
- Tutorial 42: Performing FEM Stress and Displacement Analysis and Structural Optimization of Solids
- Tutorial 43: Performing FEM Structural Optimization (CAO) and Topological Optimization (SKO) of Solids

Motivation for this tutorial: (Originally SolidGeometry 4.9 required)

```
dbprintf('This tutorial has the task to explain the functions of Yilun Sun (MIMED PhD student since 2017) and to test the functions with every new vers
dbprintf('The SKOL functions allow the design of solid-state joint mechanisms by technical laymen without knowledge of mechanism theory');
% function VLFL_EXP53

dbprintf('This tutorial automatically designs a forceps with solid joints and the special feature of a flexible gripping surface.');
dbprintf('First, we design an outer contour that already contains the surface to be moved later. ');
dbprintf('On the other hand it is the maximum area that might be occupied by the mechanism later.');
dbprintf('Make sure that the moving face can be distinguished from a longer face line by an angle greater than 45 degrees');

% CPLF

% Automatic design of adaptive compliant forceps
```

```
VLFL_EXP53: This tutorial has the task to explain the functions of Yilun Sun (MIMED PhD student since 2017) and to test the functions with every new ve
VLFL_EXP53: The SKOL functions allow the design of solid-state joint mechanisms by technical laymen without knowledge of mechanism theory
VLFL_EXP53: This tutorial automatically designs a forceps with solid joints and the special feature of a flexible gripping surface.
VLFL_EXP53: First, we design an outer contour that already contains the surface to be moved later.
VLFL_EXP53: On the other hand it is the maximum area that might be occupied by the mechanism later.
VLFL_EXP53: Make sure that the moving face can be distinguished from a longer face line by an angle greater than 45 degrees
```

Define the maximum outline contour

```
  [~,CPLoutline] = SGofCPLcommand('b 100 30,move 0 -15,enter,b 30 10 -30,move 65 -5,enter,b 6 4,move 20,-,-,enter,c 20 15,move -35 -20,-,enter,b 40 10,
%   CPLoutline=CPLradialEdges(CPLoutline,5);

%   [~,CPLoutline] = SGofCPLcommand('b 100 30,move 0 -15,enter,b 30 10 -30,move 65 -5,enter,b 6 4,move 20,-,-,enter,b 40 10,move -35.5,-,enter,b 100 36
```

Define the fixation contours

```
  bb=BBofCPL(CPLoutline);
  CPLfix1=PLsquare(1)+[bb(2)+2 bb(4)+2]; % Base element WILL DEFINITELY BE UNMOVED PART OF STRUCTURE!
%   CPLfix1=PLsquare(1)+[-15 -1]; % Base element WILL DEFINITELY BE UNMOVED PART OF STRUCTURE!

%   [~,CPLfix1] = SGofCPLcommand('b 1,move 40 40');  % Base element WILL DEFINITELY BE UNMOVED PART OF STRUCTURE!
  [~,CPLfix2] = SGofCPLcommand('b 20 1');          % FACE TO REALTIVELTY MOVE THE

  condfixation = {CPLfix1,[1 1];CPLfix2,[1 1]};    %   [x y ] => > 0=false   1 = true(fixed)
%   condfixation = {CPLfix1,[1 1]} %;CPLfix2,[0 1]};
```

Define the load contours

```
[~,CPLload] = SGofCPLcommand('b 10,move -35 -16');
```

```
[~,CPLload] = SGofCPLcommand('b 5 20,move -35 -20');   % MUST BE OVERLAPP OUTSIDE AND INSIDE OF CPLOUTLINE
  condload={CPLload, [-0.000 +0.001]};                    % Force in [x y] in Newton
%   condload={CPLload, [0 -0.001]};                   % Force in [x y] in Newton
```

DEFINE FLEXIBLE CONTOURS

```
  CPL1spring = CPLload;                                % LOAD MUST BE FLEXIBLE [X=0 Y=1/2 CPLload
  pout=[50 -10]; CPL2spring=PLsquare(1)+pout;          % Movable Part
  [~,CPL3spring] = SGofCPLcommand('d 10 38 -5');       % OTHER FLEXIBLE ELEMENTS
  condspring = {CPLload,abs(condload{1,2}/2);CPL2spring,[0 0.0005];CPL3spring,[0.000 0.00075]};
  condspring = {CPLload,abs(condload{1,2}/2);CPL2spring,[0 0.0005]}; %;CPL3spring,[0.000 0.00075]};
```

DEFINE LOAD DIRECTION AND SIGN x- x+ y- y+

```
  poutxy = 2;     % 1==x 2 y=y
  poutdir = 1;    % +1 / -1
```

DEFINE AREA OF NO TOPOLOGY OPTIMIZATION

```
  [~,CPLnokill] = SGofCPLcommand('c 26 18,move -35 -20,enter,b 30 10 -30,move 65 -5.5,+');
%   [~,CPLnokill] = SGofCPLcommand('c 26 20,move -35 -20');
%
  CPLnokill=CPLintersect(CPLnokill,CPLoutline);
```

No kill by creation of walls of all the inside contours (!) AND IF A CLEAR SURFACE IS WANTED

Additonal Parameter

```
VolFrac = 0.3;    % 30 percent of CPL outline
MaxIter = 50;     % Number of iterations

h=0.4;        % height
E0=1;         % E moudlus
nu=0.3;       % Poissin
```

JUSRT PLOT

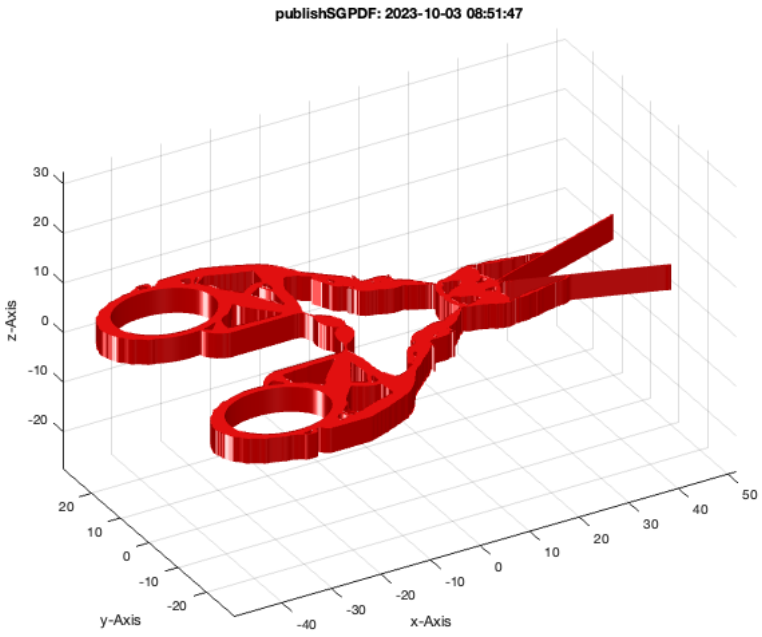
```
SGfigure; CPSplot(CPLoutline,'b',0.2,'k',1);  CPSplot(condload{:,1},'r',1);   CPSplot(condfixation(:,1),'m',1);  % return
CPSplot(condspring(:,1),'g',1);  CPSplot(CPLnokill,'y',0.2,'y');


copyfig(gcf,100);


warn=warning('off','MATLAB:nearlySingularMatrix' );  % UNCLEAR USAGE
     warning('off','MATLAB:polyshape:repairedBySimplify');
    [CPL_TOP,z,fem] = CPLTopCM_SUN(CPLoutline,h,E0,nu,condfixation,condload,condspring,pout,poutxy,poutdir,CPLnokill,VolFrac,MaxIter);
warning(warn);

SG2 = SGofCPLz(CPL_TOP,5);T2.VL = SG2.VL;T2.FL = SG2.FL;
      SGfigure(T2);
```

```
sf =
   100
It: 1    Objective: -0.020      Change: 0.700
It: 2    Objective: -0.024      Change: 0.050
It: 3    Objective: -0.006      Change: 0.050
It: 4    Objective: -0.001      Change: 0.050
It: 5    Objective: -0.000      Change: 0.050
It: 6    Objective: 0.000       Change: 0.050
It: 7    Objective: 0.000       Change: 0.050
It: 8    Objective: 0.000       Change: 0.050
It: 9    Objective: 0.001       Change: 0.050
It: 10   Objective: 0.002       Change: 0.050
It: 11   Objective: 0.005       Change: 0.050
It: 12   Objective: 0.008       Change: 0.050
It: 13   Objective: 0.013       Change: 0.050
It: 14   Objective: 0.018       Change: 0.050
It: 15   Objective: 0.020       Change: 0.050
It: 16   Objective: 0.022       Change: 0.050
It: 17   Objective: 0.024       Change: 0.050
It: 18   Objective: 0.025       Change: 0.050
It: 19   Objective: 0.027       Change: 0.050
It: 20   Objective: 0.028       Change: 0.050
It: 21   Objective: 0.029       Change: 0.050
It: 22   Objective: 0.030       Change: 0.050
It: 23   Objective: 0.031       Change: 0.050
It: 24   Objective: 0.031       Change: 0.050
It: 25   Objective: 0.032       Change: 0.050
It: 26   Objective: 0.033       Change: 0.050
It: 27   Objective: 0.033       Change: 0.050
It: 28   Objective: 0.033       Change: 0.050
It: 29   Objective: 0.034       Change: 0.050
It: 30   Objective: 0.034       Change: 0.050
It: 31   Objective: 0.034       Change: 0.050
It: 32   Objective: 0.035       Change: 0.050
It: 33   Objective: 0.035       Change: 0.050
It: 34   Objective: 0.035       Change: 0.050
It: 35   Objective: 0.036       Change: 0.050
It: 36   Objective: 0.036       Change: 0.050
It: 37   Objective: 0.036       Change: 0.050
It: 38   Objective: 0.036       Change: 0.050
It: 39   Objective: 0.037       Change: 0.050
It: 40   Objective: 0.037       Change: 0.050
It: 41   Objective: 0.037       Change: 0.050
It: 42   Objective: 0.037       Change: 0.050
It: 43   Objective: 0.037       Change: 0.050
It: 44   Objective: 0.038       Change: 0.050
It: 45   Objective: 0.038       Change: 0.050
It: 46   Objective: 0.038       Change: 0.050
It: 47   Objective: 0.038       Change: 0.050
It: 48   Objective: 0.038       Change: 0.050
It: 49   Objective: 0.038       Change: 0.050
It: 50   Objective: 0.038       Change: 0.050
```



Final Remarks

close all VLFLlicense

Published with MATLAB® R2023a