==========

"Topology-Constrained Surface Reconstruction From Cross-sections"

Publication: Siggraph 2015
Author: Ming Zou (mingzou.cn@gmail), Michelle Holloway, Nathan Carr, Tao Ju
Version: 1.0.0
Last update: Sep 20, 2015

I. Introduction==========

CycleGrouping implements the algorithm proposed in paper "Topology-Constrained Surface Reconstruction From Cross-sections" (Siggraph 2015). Given a set of parallel or non-parallel 3D contours (a .contour file, see bellow) together with a desired genus, the algorithm generates a watertight mesh (Result.obj), which interpolates the input contours and has the desired genus. If a 3D volume is available, the algorithm can also take the volume (a .mrc file, see bellow) into consideration and guide the reconstruction towards the shape saved in the 3D volume.

The output mesh computed by the algorithm is not smoothed. To complete the pipeline, a third-party fairing/smoothing method is adopted in the distributed executable and source code. (http://www.cs.wustl.edu/~taoju/lliu/paper/ctr2suf/program.html)

Users can choose to use this suggested method to get Result_afterSmooth.obj or apply any other kind of fairing/smoothing method to further polish the results. Note that, in order to interpolate the contours, the smoothing method should keep the contours' location unchanged. The contour edge information of the output mesh is saved in ContourEdges.txt for downstream smoothing methods (see bellow).

(The smoothing method could take up to 1~2 minutes depending on the number of iterations. Do not close the terminal while the smoothing is still running.)

II. What's in the package==========

- exe_data/CycleGrouping_parallel.exe
  The runnable executable with the topology exploration part of the algorithm paralyzed.

- exe_data/CycleGrouping_serial.exe
  The runnable executable with the topology exploration part of the algorithm running in serial.

- exe_data/Data/*
  Examples.

- exe_data/*.sh
  Scripts and suggested parameters to run the examples in ./Data.

- src/*
  The source code, which was compiled on Windows 8, with Visual Studio 2012 under 32bit platform.

(parallelization can be switched on/off by setting the _DO_PARALLEL_ to be 1/0 in Utility.h in the source code.)

III. How to run==========

[Usage]

```
CycleGrouping.exe <in_contour_file> <in_vol_file> <in_vol_bbox_file> <out_dir> <tet_vol_limit>
                  <random_walk_beta> <n_fair_out_loop> <n_fair_loop> <n_fair_in_loop> <genus>
```

ARGUMENT	VALUE	DESCRIPTION	EXAMPLE
1) in_contour_file	string	The input contours, a .contour file.	knot.contour
2) in_vol_file	string	The optional input volume, a .mrc file; if not applicable, set an invalid file path, for example, "NULL".	knot.mrc
3) in_vol_bbox_file	string	The optional bounding box of the input volume; if not applicable, set an invalid file path, for example, "NULL".	knot.bbox
4) out_dir	string	The directory where output files are saved.	./Result/

5) tet_vol_limit	float	Parameter used in TETGEN for tetrahedralization.	10
6) random_walk_beta	float	Parameter used in RandomWalk for applying volume.	50
7) n_fair_out_loop	int	Parameter used in JuFair smoothing algorithm, # of JuFair operations before refinement.	200
8) n_fair_loop	int	Parameter used in JuFair smoothing algorithm, # of loops for (refinement + fairing) operation.	10
9) n_fair_in_loop	int	Parameter used in JuFair smoothing algorithm, # of JuFair operations inside the loop.	50
10) genus	int	the desired genus, a non-negative integer.	1

Notes:
1. .contour is the same input format used in Ctr2Suf.
   (http://www.cs.wustl.edu/~taoju/lliu/paper/ctr2suf/program.html)
   The input files in ./Data can be also used in Ctr2Suf.

2. .mrc uses the MRC file format and can be visualized with UCSF Chimera.
   (http://www.cgl.ucsf.edu/chimera/)

   Specifically, the volume data of ./Data/ChickHeart uses a different volume format, which is developed by Michelle Holloway. This volume data can be visualized in the VolumeViewer.
   (http://volumeviewer.cse.wustl.edu/VolumeViewer/Home.html)

3. .bbox defines the bounding box (along xyz axes) of the volume data, in the same coordinate system of .contour. If a volume is given but no bounding box is specified, the algorithm uses the bounding box of the contours as the bounding box of the the volume, which could introduce inaccuracy and generate undesired results while applying the volume to the reconstruction.

4. tet_vol_limit is the volume constrain used to control tetrahedralization in TETGEN.
   This value is the desired volume of a single tetrahedron.
   For more information, please refer to the “-a Imposes Volume Constrains” section in the readme of TETGEN: http://wias-berlin.de/software/tetgen/files/tetgen-manual.pdf

   All input contours (and volume) will be scaled and centered in a 100*100*100 cube before further tetrahedralization. Users can use this number as a reference while setting this value.

5. random_walk_beta is the beta value used in Random Walk to define the weights between nodes.
   This value is only used when a volume is given. The higher the value, the more biased the result towards the volume data.

   How to set this value highly depends on the quality of the volume data. For our examples, numbers between 10~200 works reasonably.

   For more information, please refer to equation 1 in “Random Walks for Image Segmentation”.
   http://cns-web.bu.edu/~lgrady/grady2006random.pdf

III. File Format

1. .contour file (contour)
Same as the .contour input format used in Ctr2Suf. Please refer to the following webpage for more information: http://www.cs.wustl.edu/~taoju/lliu/paper/ctr2suf/program.html

2. .mrc file (volume)
The MRC file format: https://en.wikipedia.org/wiki/MRC_(file_format).
Suggested visualization tool: UCSF Chimera: http://www.cgl.ucsf.edu/chimera/

3. .bbox file (volume)
The bounding box (along xyz axes) of the volume data, in the same coordinate system of contours defined in the .contour file. There are six lines in the .bbox file:

LINE NUMBER	DESCRIPTION
1	min x
2	min y
3	min z
4	max x
5	max y
6	max z

4. ContourEdges.txt (output)
The contour edge information of the output Result.obj, using the vertex indices of Result.obj. This edge information can be used in downstream smoothing methods to reserve the locations of the contours.

FILE FORMAT	DESCRIPTION
# of edges	total number of the contour edges
e_0[0] e_0[1]	the two vertices of an edge, one edge per-line
e_1[0] e_1[1]	2nd edge
…	
e_n[0] e_n[1]	last edge

==========