3DSpineMS Spine Feature Extractor

Table of Contents

Getting Started 1
Walkthrough 1
Notes 4
Frequently asked questions (F.A.Q.) 4

Extracts dendritic spines features.

Getting Started

3DSpineMS SFE is built in MATLAB®. It requires **MATLAB 7.6 (R2008a) or superior**.

Dependencies

3DSpineMS SFE also uses the following software dependencies, all of them are bundled as libraries in the package.

- Geom3D, License: BSD.
- Smooth Triangulated Mesh, License: BSD.
- Toolbox Fast Marching, License: BSD.
- Conversion of conics parameters, License: BSD.
- Write cell array to text file, License: BSD.
- Ellipse Fit (Direct method), License: BSD.
- SplitFV, License: BSD.
- Unify Mesh Normals, License: BSD.
- Mesh Voxelisation, License: BSD.

Installation

1. Decompress/Copy SpineFeatureExtractor folder to any place. For example: `C:/matlab/mylibraries/SpineFeatureExtractor`
2. Add that folder and its subfolders to Matlab search path. (See `addpath`)

```
addpath(genpath('C:/matlab/mylibraries/SpineFeatureExtractor'))
```

Walkthrough

3DSpineMS SFE provides `newsfe` class which exposes up to five public functions for using the tool in different ways:

- **processSpines** : Processes dendrites TIF and VRML files to obtain spines .MAT files.
- **repairSpines** : Repairs spines fragmentation and their neck.
- **computeLevelCurves** : Computes spines level curves.
- **extractFeatures** : Computes and extracts spines features.
- **runAll** : Runs all previous functions.

Creating a sfe instance

To start using the tool, creating a new `sfe` is needed. The `newsfe` constructor expects one parameter:

- **OutputPath** : The absolute path to the folder where output files will be saved. For example: `C:/3DSpineMS/sfe/data`

Usage:

```
sfe = newsfe('C:/3DSpineMS/sfe/data');
```

Processing spines (`processSpines`)

The first step that can be performed with this tool, is to process the spines, which means to convert TIF and VRML dendrite source files to spine matlab matrices (.MAT files).

This function needs the following input parameters:

- **root_TIF** : Path to the dendrite TIF files, can be a compressed .ZIP file or a folder. For example: `C:/3DSpineMS/sfe/sources/root_TIFs.zip`
- **root_VRML** : Path to the dendrite VRML files, can be a compressed .ZIP file or a folder. For example: `C:/3DSpineMS/sfe/sources/root_VRMLs.zip`

Usage:

```
root_TIF = 'C:/3DSpineMS/sfe/sources/root_TIFs.zip';
root_VRML = 'C:/3DSpineMS/sfe/sources/root_VRMLs.zip';
sfe.processSpines(root_TIF, root_VRML);
```

Resulting .MAT files will be saved into a folder called `SPINE_MAT` located inside output folder configured when `sfe` instance was created.

Note: If .ZIP files are choosed as inputs, their contents will be decompressed into the folders called `DENDRITE_TIF` and `DENDRITE_VRML` inside output folder configured when `sfe` instance was created.

Repairing spines (`repairSpines`)

This step is for repairing spines fragmentation and also their neck. A spine is fragmented when the VRML 3D model contains more than one part. Reparation process is to try to merge all parts into one.

There are three fragmentation levels:

- Correct.
- Partially fragmented.

- Fragmented.

This function needs the following input parameters:

- **root_MAT** : Path to the spines .MAT files, can be a .ZIP file or a folder. For example: `C:/3DSpineMS/sfe/data/SPINE_MAT`, (output of processSpines which is commonly used before repairSpines).
- **root_ipoints** : Path to the dendrite insertion points VRML files, can be a .ZIP file or a folder. For example: `C:/3DSpineMS/sfe/sources/root_insertion_points.zip`.

Usage:

```
root_MAT = 'C:/3DSpineMS/sfe/data/SPINE_MAT';
root_ipoints = 'C:/3DSpineMS/sfe/sources/root_insertion_points.zip';
sfe.repairSpines(root_MAT, root_ipoints);
```

Reparation process will generate three files in the output folder containing the paths of the spines depending their fragmentation level. Filenames are as follows:

- `CORRECT_SPINES_PATHS.txt` for correct ones.
- `PARTIALLY_FRAGMENTED_SPINES_PATHS.txt` for partially fragmented ones.
- `FRAGMENTED_SPINES_PATHS.txt` for fragmented ones.

Repaired spines will be saved in the output folder into a folder called `SPINE_REPAIRED`.

This process also performs spine neck reparation which is to rebuild the neck of those spines which do not appear to be attached to the dendrite. Spines with repaired neck will be saved in the output folder into a folder called `SPINE_NECK_REPAIRED`.

Computing level curves (`computeLevelCurves`)

This step is for computing spine level curves. The spine level curves help to define the morphology of the spine. This curves are useful for extracting spine features and also for being able to rebuild the spine and simulate new ones.

Those spines which are considered bad (double curve defect) can be removed automatically or manually. If manual removing is choosed, the user will be asked for the correctness of the spine which is currently rendered.

- If the spine is correct, the user must press **Enter**.
- If the spine is bad, the user must press **D** (delete) and then **Enter**. This will remove the spine from the folder to avoid a future feature extraction.

This function needs the following input parameters:

- **root_neck_repaired** : Path to the folder with repaired spines used to calculate their level curves. For example: `C:/3DSpineMS/sfe/data/SPINE_NECK_REPAIRED`, (output of repairSpines which is commonly used before computeLevelCurves).
- **num_curves** : Number of computed level curves. For example: 8.

- **remove_auto** : If is TRUE, those spines with double curve defect will be removed automatically, otherwise user will be asked for removing spines which could present double curve defect. For example: true.
- **threshold** : Used to decide when double curve defect exists. The smaller the threshold value, the more the number of double curve defects detected.

Usage:

```
root_neck_repaired = 'C:/3DSpineMS/sfe/data/SPINE_NECK_REPAIRED';
num_curves = 8;
remove_auto = true;
threshold = 2;
sfe.computeLevelCurves(root_neck_repaired, num_curves, remove_auto,
 threshold);
```

After computing level curves, .MAT files inside `root_neck_repaired` folder will be updated including a cell array with the curves data.

Extracting features (`extractFeatures`)

features extraction step.

Running all steps (`runAll`)

running all steps.

Notes

notes.

Frequently asked questions (F.A.Q.)

faq.

Published with MATLAB® R2016b