Pulmonary Toolkit

https://github.com/tomdoel/pulmonarytoolkit

Tutorial 2

Exporting data

Version 1.6

Tom Doel

Overview

This tutorial is about how to save data from the Pulmonary Toolkit, including analysis results and graphs, segmentation meshes and airway centreline models.

Topic covered

- Generating analysis results and graphs
- The output folder
- Aside on coordinate systems
- Generating meshes of the airway lumen and lobar surfaces
- Generating results using the API

Requirements

You should follow the documents “Installing the Pulmonary Toolkit” and “Tutorial 1” to install the required software and make yourself familiar with the user interface.

1. First: update your git

Assuming you are using a git clone of the pulmonarytoolkit repository, you can update this using your git client (such as GitHub Desktop, GitKraken or SourceTree). Or you can do it from the command line using

```
$ git pull
```

2. Introduction - exporting data

The results generated by the Toolkit are stored in an internal format. If you want to export the data into a format you can use in your own programs, or there are a number of library functions to help you do this. Similarly, there are functions to help you draw graphs or save tables from numeric data.

A number of plugins are included, which perform some common exporting tasks. More generally, you can use the Toolkit's API and library functions to help you export.

3. Generating analysis results and graphs

We use the term **analysis** to describe algorithms which produce quantitative data (metrics and graphs). First, we'll look at the plugins already included with the Toolkit. Start Matlab and run the GUI using the command

```
>> ptk
```

Look at the plugins in the **Analysis** tab. Try some of the following plugins:

- **Lung analysis**: Measures volume, emphysema percentage (EP) and percentile density (PD), and airway radius and wall thickness for the trachea and main bronchi - **see Tutorial 4 for more information.**
- **Lobar analysis**: Performs the same analysis as for Lung analysis, but also provides measurements for each lobe and lobar bronchi. NB. This depends on successful lobe segmentation or corrected lobe segmentation.

- **Segmental analysis**: As above, but expends to the pulmonary segments (Experimental)
- **Axial metrics**, **Coronal metrics**, **Sagittal metrics** - divide the lung into thick slices along the specified axis, and then perform analysis for each slice, and produce graphs showing variations of density and emphysema along the axis. **See Tutorial 4 for more information.**

Analysis results produce output files in csv format, which you can load into Excel or any text editor. Some analysis plugins also produce graphs. All these are written into the **Output** folder, which we discuss next.

4. The output folder

The output folder is where files are written which are intended for your own use. For example, such files may include tables of data, graphs and geometric models derived from image segmentations. This is different from the cache folder, which is internally used by the Toolkit and should not be accessed directly.

Each dataset has its own output folder. This ensures there is no confusion between results generated from different datasets. The output folder is generally found at

```
~/TDPulmonaryToolkit/Output/<patient-name>/
```

where <patient-name> is the name of the patient (if a Dicom file) or the filename for that dataset. Results are grouped into subfodlers within the output folder, named according to the type of analysis which has been performed.

You can go directly to the output folder from the GUI by clicking the **Open output folder** button on the Analysis tab (NB this only works on Windows and Mac; for Linix it will display the path which you will need to browse to yourself).

Examine the output folder. You should find results files (text files and images) relating to the analysis plugins you ran earlier.

5. Airway and lobar meshes

The **Save Airway Mesh** plugin (under **Segment** / **Airways**) generates a smoothed surface mesh from the segmented airway lumen, and saves as an STL file. You will see this file in the output folder. You can use a 3D viewer (such as **Pleasant 3D** for Mac) to visualise this STL file.

The **Save Lobe Mesh** plugin (under **Segment** / **Lobes**) will save STL surface meshes for each lobe into the output folder.

6. Aside: coordinate systems

A common problem when comparing the outputs from different medical imaging software is inconsistency in the coordinate systems used. Since the images are discrete, voxels can be located by integer coordinates; however, this is not a good general system, since coordinates will change if the image is resized. When writing out meshes and centrelines, coordinates are usually given in mm, but there is often an inconsistency in the image origin.

The Dicom standard defines the origin as an arbitrary but fixed point inside the scanner. Therefore, all images from the same scanner will have the same Dicom origin. However, within each 2D image, coordinates are locally defined relative to an origin in the centre of the first voxel of the image (remember, medical images are usually 2D: see tutorial 1). So Dicom coordinates are generally computed by adding the local coordinates to an offset vector, which describes the relationship between the local origin and the scanner origin (this offset vector will change for each image slice, and will also depend on the pixel size). These Dicom coordinates are what you see in a medical imaging viewer such as OsiriX.

So far so good... except that image analysis software sometimes forgets about the Dicom offset when writing out files. For example, this might happen if the software does not write out a Dicom offset into a **.mhd** metaheader file. This means you end up with a coordinate system with an origin in the centre of the first voxel, instead of the correct scanner origin - meaning meshes generated with different software may not match up! It also means your origin is dependent on voxel size, and will change if you resample your image.

To avoid the dependency on voxel size, PTK uses a slightly different coordinate system, based on the corner of the image. Resampling an image always maintains this image corner, so the coordinates in mm never change.

However, PTK allows you to choose the coordinate system you want to use when exporting meshes, airway trees etc, so you can choose coordinates that match the system used by your software. There are currently 3 choices:

- **PTK coordinates** (corner of the image)
- **Dicom coordinates** (relative to the scanner origin
- **DicomUntranslated** (relative to the centre of the first voxel in the image)