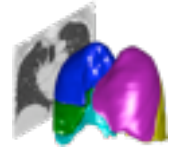


# Pulmonary Toolkit



<https://github.com/tomdoel/pulmonarytoolkit>

## Tutorial 1

## Loading and visualising data

Version 1.2

Tom Doel

### Overview

This tutorial is an introduction to the graphical user interface (GUI) of the Pulmonary Toolkit, including how to load and visualise data.

### Topic covered

- Starting and closing the GUI
- Loading and visualising data
- Aside on medical imaging data and how it is stored on disk
- What are Plugins, and how to run them
- 3D visualisation and saving images
- Examples: airway segmentation, trachea detection and lung segmentation

### Requirements

You should follow the document “Installing the Pulmonary Toolkit” to install the required software and download the PTK before following this tutorial.

# 1. Introduction

## What is the Pulmonary Toolkit?

It's three things:


1. A suite of algorithms for medical image analysis;
2. A software framework which allows you to perform sophisticated (lung) image analysis by connecting these algorithms together in a robust and efficient way;
3. A graphical user interface (GUI), which provides a simple way of running these algorithms and visualising the results.

This tutorial will focus on using the graphical interface. In future tutorials, we will see how to operate the Toolkit without the user interface.

## 2. Starting the PTK

To start the GUI:

1. Start Matlab.
2. Change the current Matlab directory to the main folder on your hard disk containing the Pulmonary Toolkit (this is the folder you specified when checking out the Toolkit in your Subversion client). This is the folder containing the file **ptk.m**.

*Note 1: the current Matlab folder is shown below the Matlab ribbon. You can choose a different folder using the yellow/green “Browser for folder” icon  to the left of this.*

*Note 2: you can add this folder to the Matlab path if you wish to avoid having to change the folder in future.*

3. In the command window, type

```
>>ptk
```

This will bring up the splash screen.



The very first time you run the Toolkit, it needs to perform initialisation, including compiling several C++ files. This will happen automatically, but it could take a few minutes.

### Compilation errors

If errors are reported on the command window, there may be a problem with the settings for your C++ compiler. Check you have a compiler installed. You may need to run Matlab's **mex -setup** command to find and select the right compiler. See Matlab's **mex** documentation for more information.

## 3. Loading data

Once the main window appears, you will want to load some lung images.

1. Click the **Import Data** button in the toolbar panel at the bottom left of the screen.
2. Choose a 3D lung dataset to load from your hard disk. PTK supports **Dicom** and **metaheader/raw** files (more on imaging data later). Dicom data normally comprise multiple files called **IM1**, **IM2**, etc. Choose any one of these files and PTK will load the whole set. For metaheader/raw files, choose the header file (extension **.mhd**).

The first time you load a dataset, PTK will run some algorithms to locate the lung within the image you have loaded. This may take a few minutes.

## 4. The patient and series list

The sidebar at the left of the screen shows a list of all the patients that have been imported into PTK. Beneath this is a list of series for the currently selected patient. If you have linked datasets (e.g. for multimodal or high dose-low dose analysis), there is an additional list showing the series linked to the currently selected series - this is covered later.

Click on a series to load. Click on a patient to select that patient and load the last series you viewed for that patient. If you have not yet viewed a patient, PTK will select the best dataset (a CT dataset if available, and the one with the largest number of slices).

## 5. The Patient Browser

The **Patient Browser** button brings up a separate dialog showing you every series for every patient. On the left is a list of patients - click on a patient to scroll the right panel to that patient. Click on a series to load that series. Currently, there is just one dataset. Later, when we are working with more than one dataset, we will use the Patient Browser to switch between them. For now you can close the Patient Browser.

## 6. Aside: medical imaging data

It's important to remember that 3D clinical datasets are not acquired as a single 3D block, but as a large number of 2D slices. When we view this data in 3D we are splicing the slices together, assuming this is a reasonable thing to do. Usually this is fine, but occasionally in some datasets you may see odd-looking effects such as gaps, repeated sections of the lung or discontinuities. These are artefacts of the way the patient has been scanned, and you should be aware that this may affect analysis. The PTK will warn you if it detects unusual image features.

### Dicom files

Medical imaging data is normally stored in Dicom files. For practical purposes, you can think of Dicom as an image format, although strictly speaking, it's actually a set of standards which clinical hardware and software is supposed to follow<sup>1</sup>. Dicom files are stored in complicated hierarchies, so it's worth understanding what these mean:

- Images are grouped by **patient**, usually in folders PA0, PA1, PA2 etc.
- Each **patient** has one or more **study**, where a study is a group of imaging carried out in one session. Studies are usually stored in folders ST0, ST1 etc., under the patient.
- Each **study** consists of a number of **series**. Series are usually stored in folders SE0, SE1 etc. under the study.
- Each **Series** is a set of 2D image slices. These images are stored in files **IM0**, **IM1**, etc. (or similar) in the series folder. Sometimes the files have the **.dcm** extension.
- Some series form a 3D dataset, and others do not

So, the image file **PA0/ST1/SE3/IM5** is image 5 in series 3 of study 1 of patient 0.

Each Dicom image file contains an extensive metaheader, which provides information about the imaging parameters used to acquire the data. You can view the Dicom metadata using a Dicom viewer such as OsiriX, or using Matlab's **dicominfo** command.

---

<sup>1</sup> I say 'supposed', because even the biggest health multinationals get parts of the Dicom standard wrong in their images.

### mhd files

PTK also supports loading from mhd files. These comprise a text header file (.mhd) and a raw data file (.raw). These files have already been assembled into a 3D dataset, and are commonly used in academic circles (e.g. the ITK and VTK projects). However, they usually lack the range of metadata found in Dicom header information, and so using Dicom files is preferred if possible. Note that the filename of the raw file is stored in the .mhd file, so if you rename the .raw file you also need to rename it in the .mhd file.

## 7. Visualising the loaded data

Now your dataset should have loaded. You will see a 2D slice from the 3D dataset. What you are seeing in CT is a greyscale image representing the radiodensity of tissue. This is calibrated and (to a good approximation, in general) is proportional to tissue density.

### Window/level

The first thing to do is set the **window** and **level** in order to visualise the features you are interested in. There's a quick way to do this - select the "Lung" button inside the Window/Level presets pane at the right of the screen. Try out other presets.

#### What are window and level?

These are like contrast and brightness. If you modify the contrast and brightness of a photo, you can often make details appear that you couldn't see before. The same is true with CT data. The range of greyscale values that CT can generate is greater than your eye can distinguish, so you need to select a range of values you are interested in. The width of this range is the window, and the centrepoint is the level.

Now try using the window/level tool. Note the W/L button underneath the image is selected (if it isn't, select it now).

Click inside the lung image and drag the mouse around. Note how the values of the window and level sliders below the image change. Select different Window/Level presets from the presets panel and see how the numbers change again.

You can manually edit the window and level values if you want specific values.

### Moving through image slices

There are 4 ways to move through the image slices in the 3D image.

- Scroll the mousewheel while over the image
- up/down cursors

## Tutorial 1: Loading and visualising data

- Scrollbar to the right of the image
- Select the **Cine** tool (n key) (in the toolbar under the image), and drag up/down inside the image

## Image orientation

The three image orientations are **Coronal**, **Sagittal** and **Axial**.

Use the buttons at the bottom-left of the screen, or the keys **C**, **S**, **A** to change orientation

## Density values

As you move the mouse cursor over the image, the value of the image under the cursor is displayed underneath the image (after the coordinate values). For CT, two values are shown; the raw image value (I:xxx) and the calibrated Hounsfield value (HU:xxx).

X:86 Y:323 Z:190 I:1051 HU:27

### Hounsfield units

Air is -1000 HU. Water is 0 HU.

To a good approximation, tissue density can be computed in g/l by adding 1000 to the HU value.

## Zoom and pan

**SHIFT+mouse drag:** pan the image.

**CTRL+mouse drag:** zooms the image

Or use the Zoom and Pan buttons (these work with standard Matlab zoom/pan tools; you can right-click to reset the view). The P and Z keys select these tools.



## 8. Switching between datasets

Use the **Import Data** button to load a second dataset.

Now open the Patient Browser again, by clicking the button at the top-left of the screen. You now have two datasets loaded.

The Patient Browser shows you all datasets you have in the database. When you load or import new datasets, they will appear in the list of datasets at the right of the Patient Browser. To switch between datasets, click on the dataset you wish to load on the right.

Datasets are grouped by patient. On the left of the Patient Browser is a list of all patients you have available in the database. Click on a patient to scroll the list of datasets to the selected patient.

As an alternative to the Patient Browser, the list of patients and series at the left of the PTK image window can be used to quickly change between datasets. Try this now.

## 9. Closing PTK

Use the normal window close button to close PTK.

Now, start PTK again by typing

```
>>ptk
```

You should find your last dataset loads immediately, and your window/level and other settings are preserved.

## 10. Introducing plugins

We'll now introduce the more advanced features of the PTK. The panel at the right hand side of the screen shows the **Plugins** that are currently available (the panel is scrollable; if some buttons are off-screen you can use the scrollbar or the mouse wheel to scroll the panel).

Each plugin represents a particular algorithm you can run. There are two types of plugins:

1. The plugins in the toolbar at the bottom of the screen generally represent how the data is visualised on screen (such as the window/level presets we used above), but don't actually affect the data itself. These also include plugins which load new data such as the **Import Data** buttons we discussed earlier.
2. The plugins in the tabbed panels to the right generally perform specific image analysis tasks which generate an output, such as an airway or lung segmentation. These plugins are generally grouped into themes related to their application, e.g. **Airways** or **Lobes**. Some plugins represent intermediate stages in a more complicated process (for example, the lobe segmentation includes many intermediate stages).

### Example: airway segmentation

The way plugins work is best illustrated with an example. Click the **Airways** button (under the **Segment** tab). This launches the airway segmentation algorithm, which may take a few minutes to run. When this completes, the airway lumen will be highlighted in blue. There may also be some regions highlighted in red. These are areas which were initially segmented by the algorithm, but then determined to be false and have been removed from the final result. These are shown for illustration only, to help the user assess the performance of the algorithm.

Now scroll through the image. The airways are illustrated throughout the image, as a transparent blue overlay. Click the **Show Overlay / Hide Overlay** button in the **SEGMENTATION DISPLAY** section in the Segment tab to toggle the overlay on and off (or press the **O** key). You can adjust the transparency of the airways overlay using the **Opacity** slider.

This illustrates how the algorithmic results are visualised. When you run a plugin, a **visual representation** of the output is displayed overlaid on the image.

Note that what you see is only a visual representation of the result. The airway segmentation algorithm actually produces a datastructure describing the connectedness of branches of the airway tree, as well as centreline and radius information. What you see onscreen is just the lumen.

We will see in later tutorials how to access and work with the underlying datastructure of the output from this algorithm.

Normally, the colour zero indicates no segmentation and is transparent, but you can make a zero segmentation value display as black by using the **Show zero as black** button (or press the **t** key).

### Trachea detection

Let's run the trachea detection plugin. This isn't normally directly visible (it is run as part of the airway segmentation but is hidden to the gui user). To show such hidden plugins, click the **Developer Tools on** button in the toolbar at the bottom right of the screen. Now click on the **Plugins** tab which will appear. Click the **Trachea** button in the **Airways** section. This will show the result of the trachea plugin, which shows the trachea lumen (in green) and the topmost point in the trachea (highlighted in red).

The result is displayed almost instantly. The reason is that the Airways plugin has already run the Trachea plugin, because it requires the trachea starting point as part of its input. The result of the trachea detection algorithm was automatically cached, so that when you run it again the result appears instantly.

Now click **Airways** again. You will see the airway results appear almost instantly. Again, this is because the result has been cached. Now quit the PTK and then restart. Try loading the Airways and Trachea again. Again, they will appear almost instantly, because the results are cached on disk and so are available between sessions.

This illustrates one of the most powerful features of the PTK. You can divide algorithms into modular sections (represented by Plugins), and each stage will be cached to improve

execution time. Furthermore, this allows you to easily check the results at each intermediate stage to ensure they appear as you expect.

Click the **Developer tools off** button to hide the plugins unless you prefer to keep them available.

### Lung segmentation

Finally, go back to the **Segment** tab and click on the **Lungs** button to launch the lung segmentation algorithm. This will find the left and right lung regions and label them with colours 1 (blue) and 2 (green).

### 3D visualisation

Click the **3D** button after the **Left and Right Lungs** algorithm completes. This will open a new window and display the lungs as a Matlab 3D image. The **3D** button displays the overlay image currently visible as a 3D image, and can be used with any suitable plugin result.

However, use the **3D airways** plugin for better visualisation of small structures such as airways and vessels.

## 11. Exporting images from the GUI

If you want to save the currently displayed images, you can use the **Export Image** button (to save the underlying medical data), the **Export Overlay** button (to save the transparent overlay image), or the **Capture** button (which creates a screen capture of the current visualisation window, including any transparent overlays).

These buttons just output the images you see on screen. We will cover the saving of the actual outputs of the plugin algorithms, we will cover this in future tutorials.