|  |
| --- |
| PyAero 1.1 |
| User Manual |
| PyAero is an open-source airfoil contour analysis and CFD meshing tool |

2015

**PyAero User Manual**

Version 1.1

Andreas Ennemoser

January 2015

**Table of contents**

[1. Introduction 5](#_Toc409172953)

[1.1. What is PyAero 5](#_Toc409172954)

[1.2. Features 5](#_Toc409172955)

[1.3. System requirements 5](#_Toc409172956)

[1.4. Installation 5](#_Toc409172957)

[1.5. Code repository 5](#_Toc409172958)

[2. User interface 6](#_Toc409172959)

[2.1. Main screen 7](#_Toc409172960)

[2.2. Menus 7](#_Toc409172961)

[2.2.1. File menu 8](#_Toc409172962)

[2.3. Toolbar 9](#_Toc409172963)

[3. Contour analysis 9](#_Toc409172964)

[4. Mesh generation 9](#_Toc409172965)

**List of figures**

[Figure 1‑1: PyAero main window - code excerpt 6](#_Toc409177861)

[Figure 2‑1: Graphical user interface of PyAero 7](#_Toc409177862)

[Figure 2‑2: PyAero menu structure 8](#_Toc409177863)

[Figure 2‑3: Typical format of an airfoil geometry description 8](#_Toc409177864)

[Figure 2‑4: Print preview dialog 9](#_Toc409177865)

# Introduction

## What is PyAero

PyAero is an open-source airfoil contour analysis and CFD meshing tool. The main intention of writing the software was to make an easy to use tool for 2D airfoil meshing and subsequent CFD analysis.

PyAero, at least at the moment, does not do the CFD calculation itself. In a later stage it might be possible that PyAero will be interfaced with existing open source CFD methods.

As an initial step towards aerodynamic calculations, a panel method from **AeroPython** (©2014 Lorena A. Barba, Olivier Mesnard) has been implemented. The code can be downloaded from:

<http://nbviewer.ipython.org/github/barbagroup/AeroPython/blob/master/lessons/11_Lesson11_vortexSourcePanelMethod.ipynb>

There are minor changes done to the code which are dedicated purely to interface AeroPython with PyAero.

In the course of the development it turned out, that airfoil contours are often described through a limited number of points (approx. 60 points). When meshing such contours, if not interpolated by splines, the resulting mesh would end up with artefacts, which would deteriorate the quality of the analysis results.

Therefore, in addition to the mesh generation, some additional features have been implemented. These features are intended to be able to analyze and improve the airfoil contour. The improvement process is supported by point insertion and spline interpolation techniques. First and second derivatives of the contour allow for control of the contour as well as curvature smoothness.

## Features

The main features of PyAero are:

* Load and display airfoil contour files. Different file formats as used by other standard software for airfoil analysis (e.g. XFOIL) are supported
* Analyze and improve the airfoil contour with respect to contour and curvature smoothness. This is purely a geometric improvement of the given airfoil coordinates
* Generate CFD meshes for airfoil analysis (C-type, O-type) including specific functions for boundary layer adjustment as well as trailing edge treatment
* Basic aerodynamic assessment using AeroPython

## System requirements

## Installation

The installation of the software under Windows is as easy as copying the executable to an arbitrary folder. Double clicking PyAero.exe starts the application.

## Code repository

The source code and Windows binary are located on Github.

🡪 <https://github.com/chiefenne/PyAero>

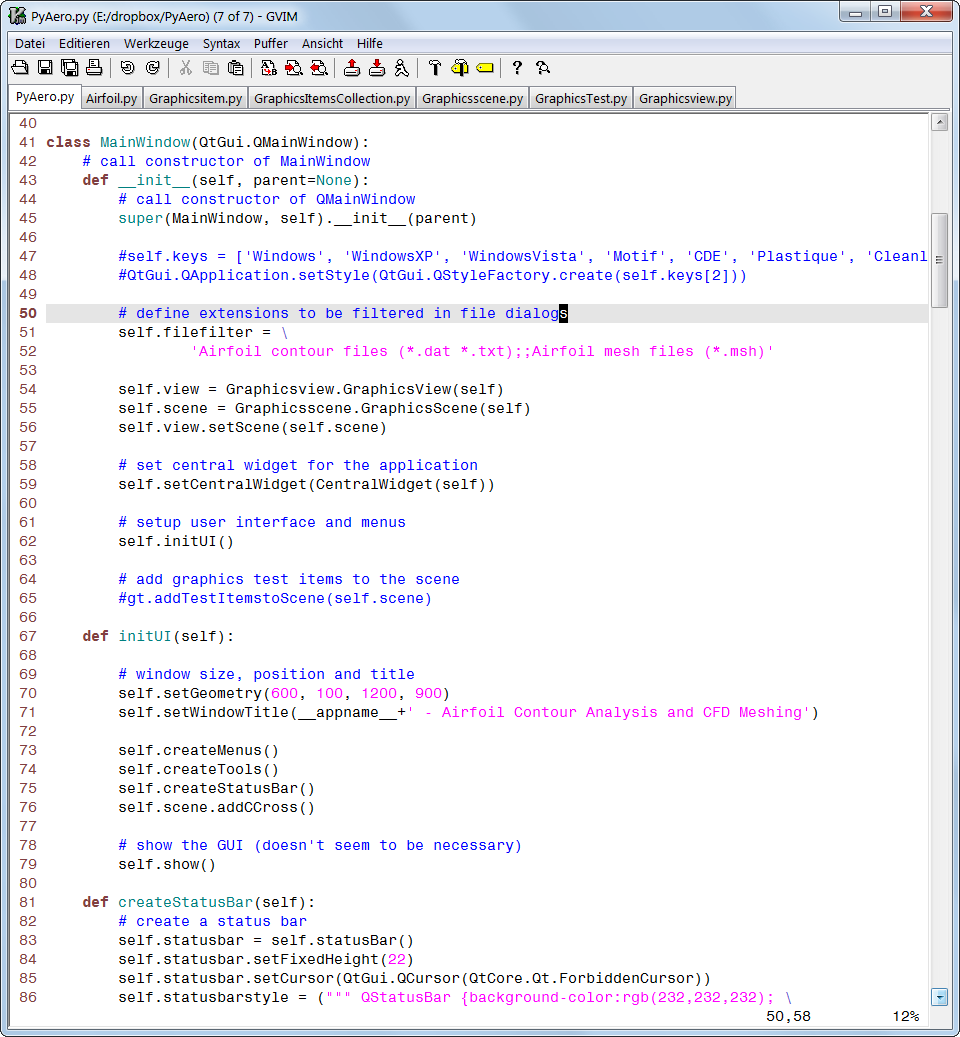
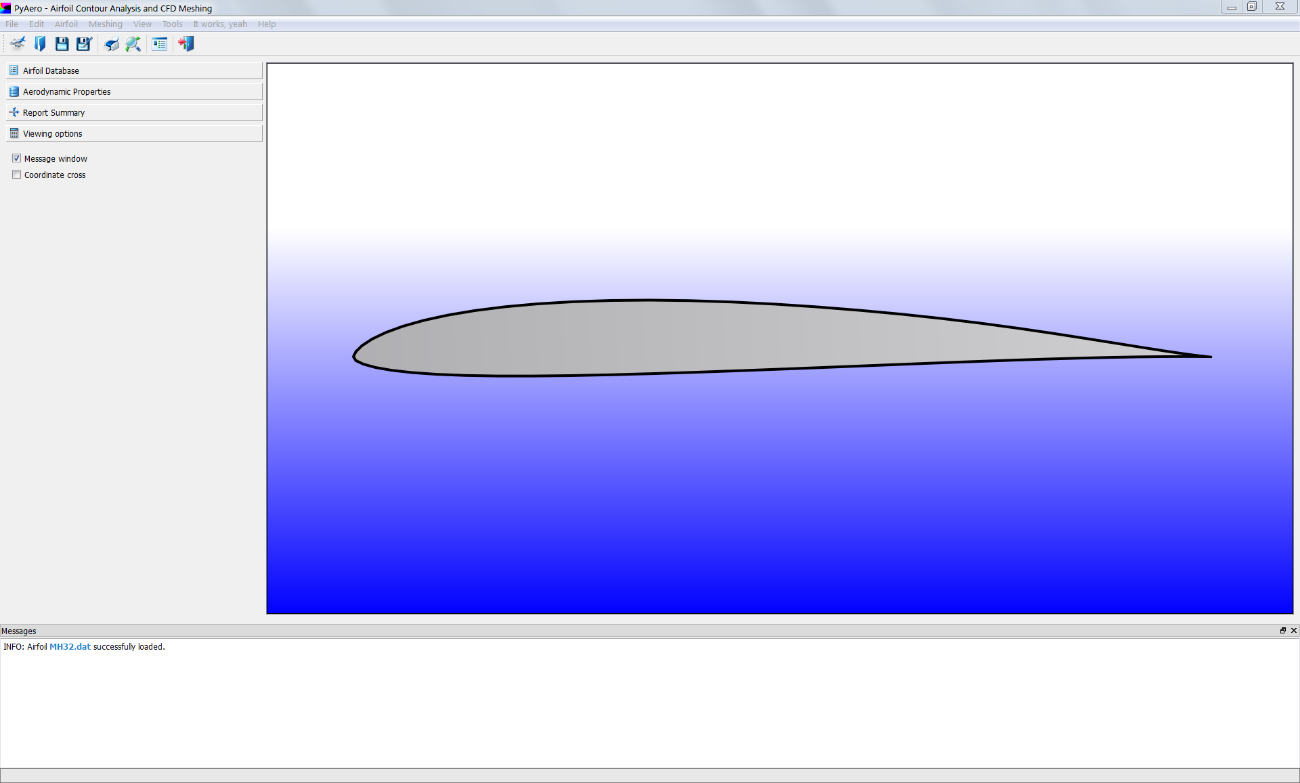


Figure 1‑1: PyAero main window - code excerpt

# User interface

PyAero comes with a graphical user interface (GUI) written in PYTHON and PyQt4 (<http://www.riverbankcomputing.com/software/pyqt/intro>). The reason for using PyQt4 is that it works with Python 2.7.x., whereas PyQt5 needs Python 3.x. May be at a later point in time a migration to the more modern packages will be undertaken.

## Main screen



Graphics View

Message window

Toolbox Functions

Toolbar

Menu bar

Figure 2‑1: Graphical user interface of PyAero

The graphics view has initially a gradient background fill. This can be toggled to constant color by using either the “CTRL+b” shortcut or clicking the respective submenu in the View menu.

## Menus

Quite some of the menus in PyAero are pretty much the same as in typical desktop software. For standard menus as File or Print the documentation will be kept short. See Figure 2‑1 for the location of the menubar in the GUI and Figure 2‑2 for an overview of the menu structure.

The menus in the menubar and the tools in the toolbar (see Toolbar) are coded in a dynamic way. That is, all menus and toolbar items (and their respective handlers/callbacks) are read from XML files (PMenu.xml, PToolbar.xml).The graphical user interface is automatically populated using the entries of those files. With this structure in place, menus and toolbar can easily be extended and customized. When adding new menus and thus functionality, it is required to provide respective handlers (referring to PyQt nomenclature so-called “slots”) to take care of the newly introduced methods.

Figure 2‑2: PyAero menu structure

### File menu

The File menu is, as pointed out before, a typical desktop software menu. It consists of a set of dialogs for file manipulation, printing and alike.

#### Open

Open is used to load airfoil contour files or meshes. At the moment files in a format like also used in XFOIL (<http://web.mit.edu/drela/Public/web/xfoil/>) can be loaded. See Figure 2‑3 for the layout/format for such a file. Only the first few lines of the file are printed here. Lines starting with a hash sign (*“*#*”*) at the beginning are interpreted as comment lines (those are not used inside PyAero).

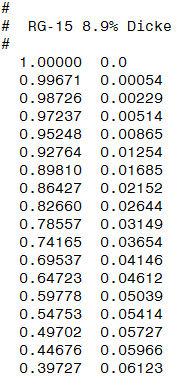


Figure 2‑3: Typical format of an airfoil geometry description file

#### Save

Save is used to save airfoil contour files or meshes. The file is saved using the name from the loaded airfoil.

#### Save as

Save as is used to save airfoil contour files or meshes. The file is saved using a new name that can be specified in the dialog which pops up.

#### Print

The Print menu allows for printing the content of the graphics view. Only graphics items as airfoil, coordinate cross, etc. will be plotted. The background gradient of the main graphics view will not be plotted. In case, that the background gradient shall be on the image, use the screenshot option.

#### Print Preview

The Print Preview menu allows previewing the printout prior to printing. Actual printing then can be started directly from the preview dialog, see Figure 2‑3.

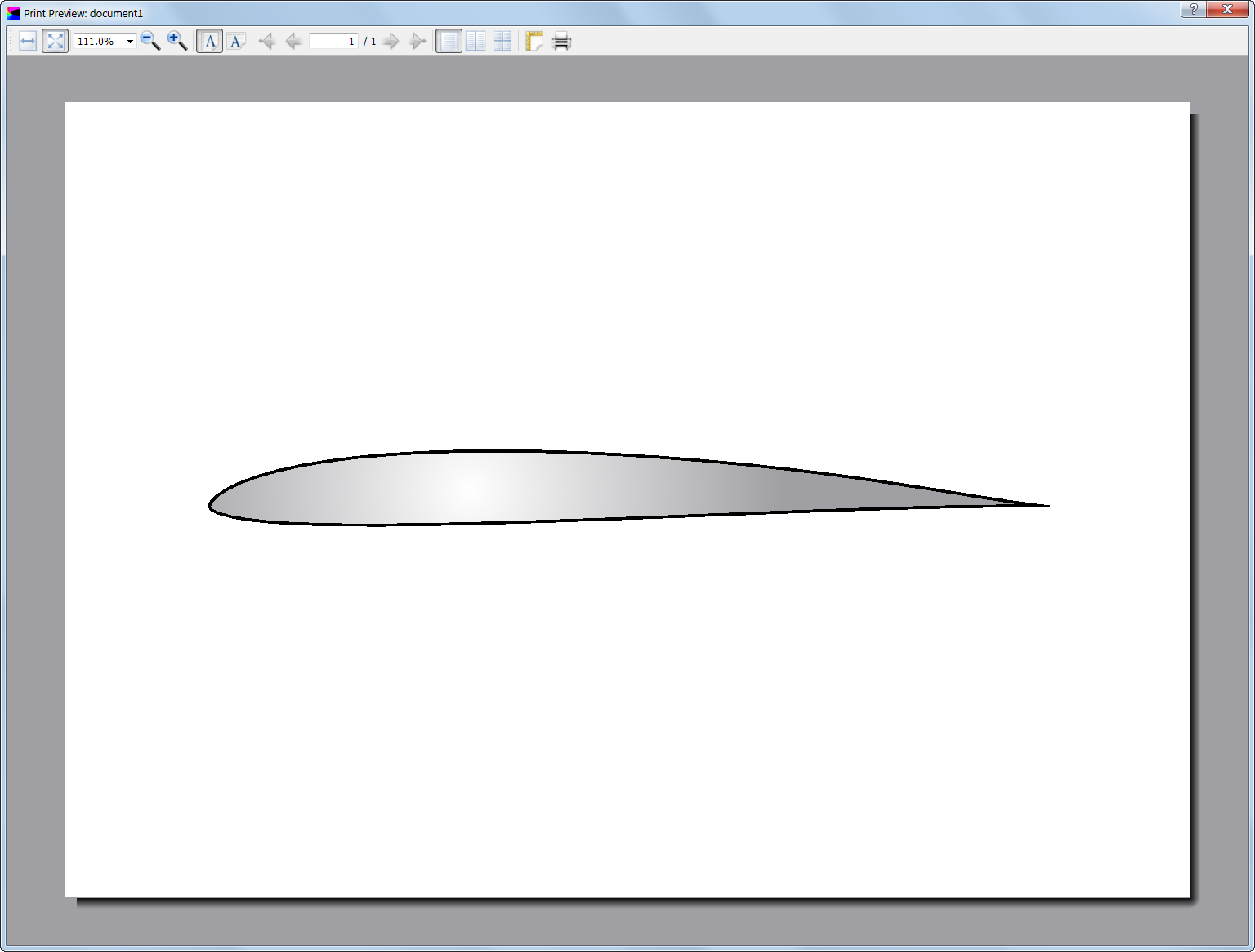


Figure 2‑4: Print preview dialog

#### View

The View menu carries submenus which allow controlling the visibility of items in the graphics view (see Figure 2‑1). The menus there should be self-explanatory by their name. If available, along the submenus there are their shortcuts listed. For example, “CTRL+f” is used to refit the currently active airfoil into the view. This is typically used after zooming and panning around in the scene.

#### Exit

Exit does what is says and closes all windows and exits PyAero.

## Toolbar

The toolbar is located below the menubar (see Figure 2‑1) and allows access to several menus. The toolbar does not carry all menus and submenus of the menubar. Instead the most often used menus are available here for convenient access. As described in Menus, the toolbar can be configured via an XML file (*PToolbar.xml*). Each tool consists of a tooltip, an icon and a handler which is a so-called PyQt “slot” that is is triggered by clicking on the respective icon.

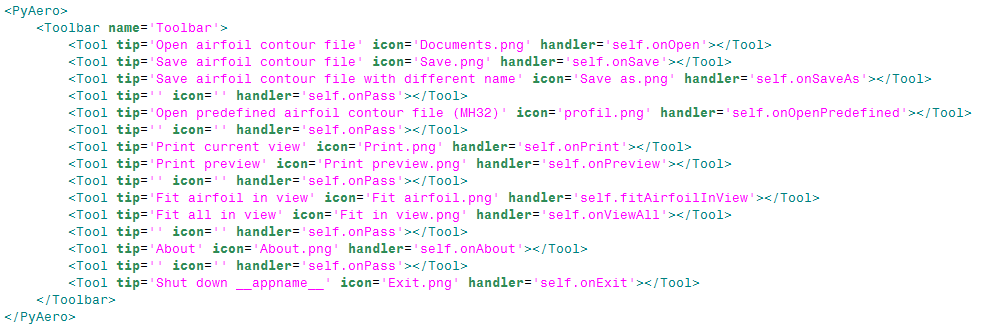


Figure 2‑5: Layout of PToolbar.xml

When hovering over the icons from the toolbar a tooltip describing the function of the button pops up.



Figure 2‑6: Toolbar icons for convenient access to menus and functions

## Graphics View

# Contour analysis

# Mesh generation