PSXTraffic Manual

version 4.7.0

Dec 8, 2023

by Nico Kaan

1 Introduction

With PSXTraffic you can fly with real live traffic in P3D or MSFS. The traffic data is provided by a separate pay ware source called **RealTraffic**. RealTraffic streams position updates of all aircraft around you every two seconds. PSXTraffic buffers the traffic 30 seconds for vectorizing. Hence, plus some internet lag the traffic you see is approximately only 45 seconds behind real time only!

The traffic data originates from ADS-B transponder data of aircraft. When airborne this gives very good results. On the ground it depends on airport policy. At some airports pilots must have their transponder on while on other airports they must switch it off. At those airports there will be not much taxiing traffic to see. But on the other airports you may follow the aircraft after landing taxiing to the gate or from the gate to the take-off runway.

The live aircraft injected (spawn) in your simulator are representations of real aircraft that are there at that time. They are **controlled by live ATC** and cannot be controlled by artificial ATC programs for Flight Simulators, such as ProATC/Pilot2ATC. The aircraft (models/liveries) are spawned in your Sim as real as possible, depending on how many different AI aircraft you have installed.

PSXTraffic determines the position for each live aircraft **hundred times per second** by calculating latitude, longitude, height, bank, and pitch. At each new frame PSXTraffic pushes the latest calculated position in your Simulator via SimConnect. Hence it depends on your fps how many of these positions will show up in your Sim.

Live airborne traffic will be shown within **a lateral range from 2 to 100 nm** around the user aircraft. Live ground traffic and live parked aircraft is shown at your origin or destination airport. Live ground traffic can be shown at airports you overfly too (optional).

PSXTraffic **knows the exact parking positions at each airpor**t in your Flight Simulator, it places departing live traffic at gates at the exact position with the right heading. Even so, a live aircraft ending its flight at a gate will be placed exactly on spot.

When you start a session on the ground PSXTraffic will generate static parked aircraft. It will use the options defined in a so-called **airport file**. Airport files included and produced offline by PSXTraffic + RealTraffic. They will be updated regularly to reflect changes in airlines and the summer/wintertime . Airport files contain information about the parking positions, what airline parks where with what model (type) at what half-hours of the day, etc.

PSXTraffic will try to match the options with the current UTC half-hour as much as possible. It will change the static parked aircraft to the situation of the next half-hour by removing and adding aircraft.

When you fly away from the departure airport all parked aircraft (static and live) will be removed. This happens when you are more than 30nm away without a destination set up, or much sooner if you do have a destination set.

When your destination airport comes into sight, say within 20 nm, PSXTraffic places static parked aircraft at the parking positions of the destination airport. It will make the best actual fit, based on the information in the airport file or based on a snapshot generated by the Recorder tool (optional, see 9.5). If you have a professional RT license, the Recorder tool records all live parked aircraft at your destination while you are flying to it!

PSXTraffic controls landing gear, throttles, spoilers, flaps, reverse thrust, wheels turning in alignment with the speed and all lights of live aircraft.

The load on your computer of PSXTraffic is very low; it consumes less than half percent of your cpu cycles. Injecting AI aircraft in your Simulator has of course some negative impact on your fps. PSXTraffic provides for built in functions to limit that drop. However, 40+ fps can be achieved, even with middle class PC's.

If you are new to PSXTraffic follow the instructions in the Quick Start Guide for P3D or the Quick Start Guide for MSFS. For questions & answers and discussions use my Support Forum at AVSIM or search the FAQ.

1.1 System Requirements

- **Windows 10** or later, Windows **11** is the **preferred** OS.

- **Microsoft Visual C++ Redistributable for Visual Studio 2022**
  You need Microsoft Visual C++ Redistributable for Visual Studio 2022. The standalone installers for x86 (**`VC_redist.x86.exe`**) and x64 (**`VC_redist.x64.exe`**) can be downloaded here at Microsoft.

1.2 Disclaimer

The data coming from the RealTraffic is not always correct. This may lead to the following shortcomings:

- Sometimes aircraft do not land at the runway but in the fields …
- Some airlines/aircraft-types always seem to land besides the runway as-if their ADS-B equipment is not sending the right info.
- Aircraft may disappear from the RealTraffic stream for more than 30 seconds which will lead to disappearance from your Flight Simulator while popping up again (a little) later.
- The ADS-B signal may become blocked by buildings at airports which leads to a temporary stop of a taxiing aircraft.

Note that if you do not like the movements of AI aircraft at the ground you can uncheck GND traffic in RealTraffic.

Some add-on airports have runways that are not at the exact (real live) location or miss new taxiways or runways.

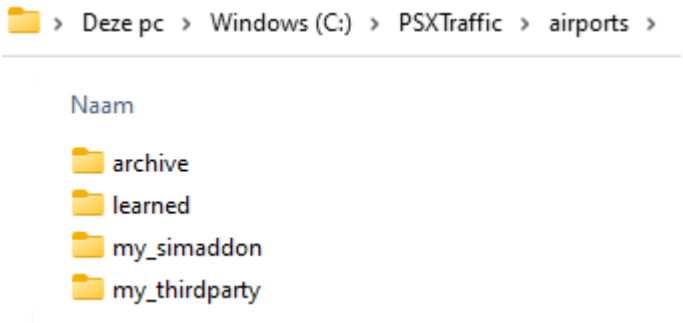
2 Installation

2.1 Download and install PSXTraffic

Download the PSXTraffic software for your Simulator from my site.
Unzip and run the installer with **administrator** rights.

> Install at your Simulator computer.
>
> The default install locations are **C:\PSXTraffic, C:\PSXTraffic_P3D** or **C:\PSXTraffic_P3Dv4**
>
> You may change that to another drive, but don't install it in the Community folder or in the Program Files folders, and preferably do not change the installation folder name!

After installation, you will have a folder containing the files and a few sub-folders.



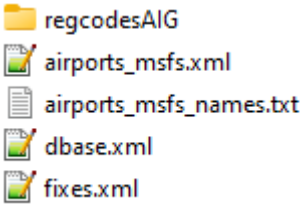
The **airport\archive** with its subfolders contains offline learned airport files. These will be updated with each new version of PSXTraffic.

The **airports\learned** with its subfolders contains data that is learned on the fly at your end, and remains available with each update. It is not overwritten or deleted.

The **airports\my_thirdparty** holds the result of the scan for your airports (see chapter 6). A <icao>.txt file contains the link to the file in the archive. A <icao>.xml is an empty airport file, there is no file available in the archive. PSXTraffic will have placed this file in the TO_BE_LEARNED_OFFLINE folder too (see 6.3). This empty airport file is used until a file in the archive becomes available.

The **airports\my_simaddon** (MSFS) or **airports\my_ORBX** (P3D) holds the result of the scan for your simaddon files or ORBX files (see chapter 6). A <icao>.txt file contains the link to the file in the archive. A <icao>.xml is an empty airport file, there is no file available in the archive. PSXTraffic will have placed this file in the TO_BE_LEARNED_OFFLINE folder too (see 6.3). This empty airport file is used (till a file in the archive becomes available).

The **input** folder contains data needed for PSXTraffic



The **output** folder has two subfolders **info** and **errors** with info about the liveries and airports scans made by PSXTraffic.

PSXT.exe and **PSXTraffic.exe** are for MSFS, **PSXT_P3D.exe** and **PSXTraffic_P3D.exe** are for P3Dv5 or P3Dv6 (both are 64 bit programs).

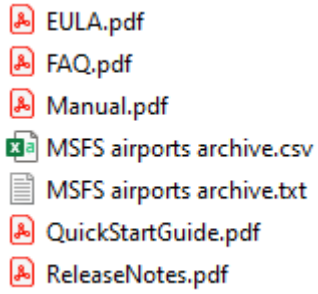
PSXTraffic_P3Dv4.exe is for P3Dv4 (is a 32 bit program); for this program there not yet a PSXT version.

Don't move/copy the executable outside the installation folder.

On your desktop you will find shortcuts to the executables of PSXT and PSXTraffic (MSFS),

or, for P3D, PSXT_P3D and PSXTraffic_P3D.exe.

Recorder.exe is a program that is being used by PSXT or PSXT_P3D (see 9.5). Don't start it yourself.



The **docs** sub-folder contains several documents. The PSXTraffic_EULA is the End-User License Agreement. This is a legally enforceable contract between you and the Author. By accepting the installation, you agree to be bound by the terms of this EULA.

2.1.1 Update PSXTraffic to a new version

PSXTraffic will tell you in the title bar (top line) of the GUI that a new version is available. Download the setup file, unzip, and run the installer. It will first uninstall your previous version and then install the new one *(so don't uninstall the previous version yourself!)*.

2.1.2 Complete removal of PSXTraffic from your PC

Removing PSXTraffic from your PC can be done running the uninstaller from the Programs list or the apps list, just like any other Windows program.

To completely remove all files, delete the installation folder and delete the folders

C:\Users\<user_name>\AppData\Roaming\PSXTraffic or
C:\Users\<user_name>\AppData\Roaming\PSXTraffic_P3D

2.2 Download and install RealTraffic.

You must install RealTraffic at the same computer as PSXTraffic.

2.3 Using more computers running PSXTraffic at the same time (for enthusiasts only …)

If you want to run a complex set up with different computers each running the Flight Simulator generating part of the Simulator screen (with WideView for instance) you do the following.

Install PSXTraffic at all PCs. At one (the main) PC install RealTraffic too, don't install RealTraffic at the other PC's, and make sure there is **no** C:\Users\<yourname>\AppData\Roaming\InsideSystems\RealTraffic.jar still present.

Make sure each simulator has the same liveries and add-on airports.

Important: In this setup you must switch **Random off** in all PSXTraffic instances otherwise each PSXTraffic may (and will) react (randomly) different in choosing a livery or choosing a static parked aircraft location.

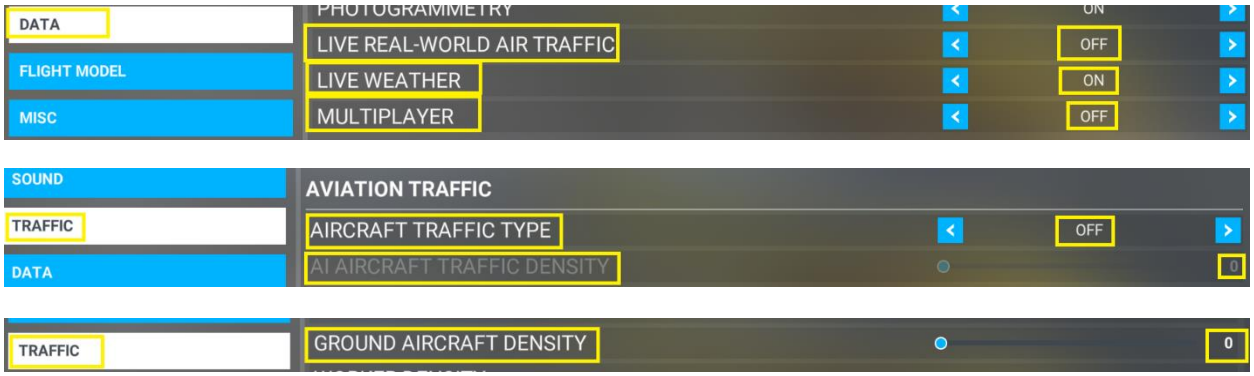
3 Operations

The order in which you start the following three programs does not matter.

3.1 Start and configure Flight Simulator

- Set the day/time to system time.
- Fly in 1 x simulation rate (no acceleration)
- Use Live Weather, you need the actual QNH to match with the QNH of RealTraffic!
- P3D: Make sure the Airline Traffic density and General aviation traffic density is set at **0%!**
- P3D: Select an aircraft as vehicle (Pedestrians, Avatars, are not allowed)

- MSFS:



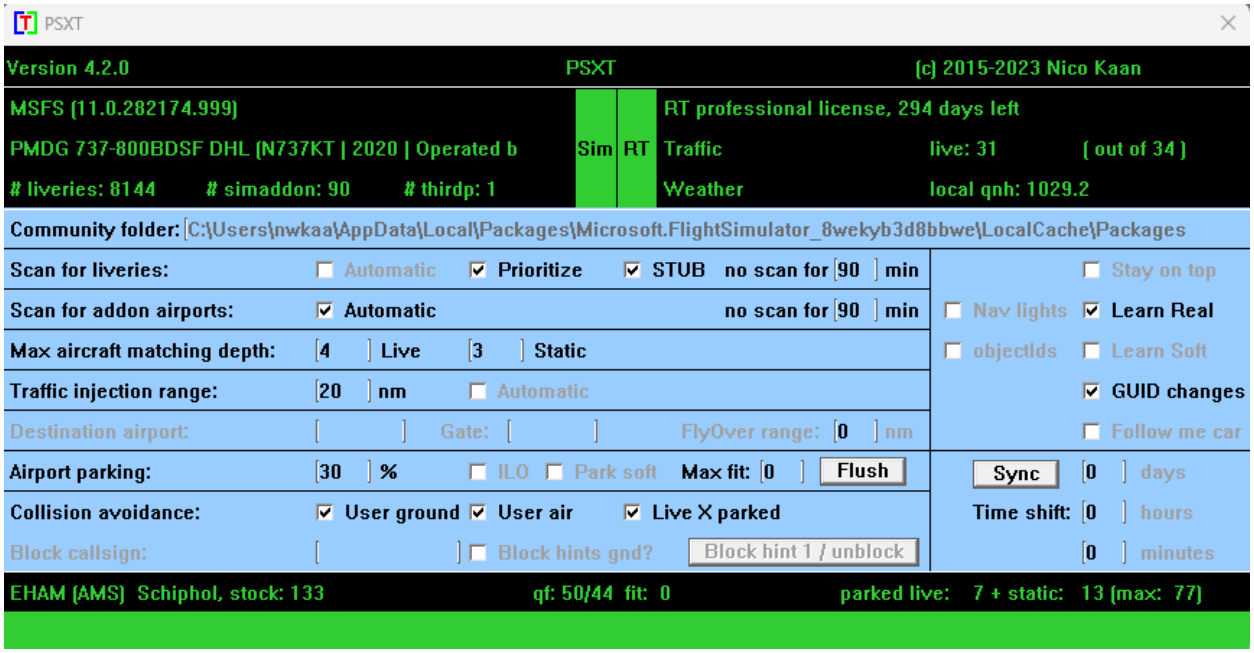
3.2 PSXTraffic or PSXTraffic_P3D only: Start and configure RealTraffic.

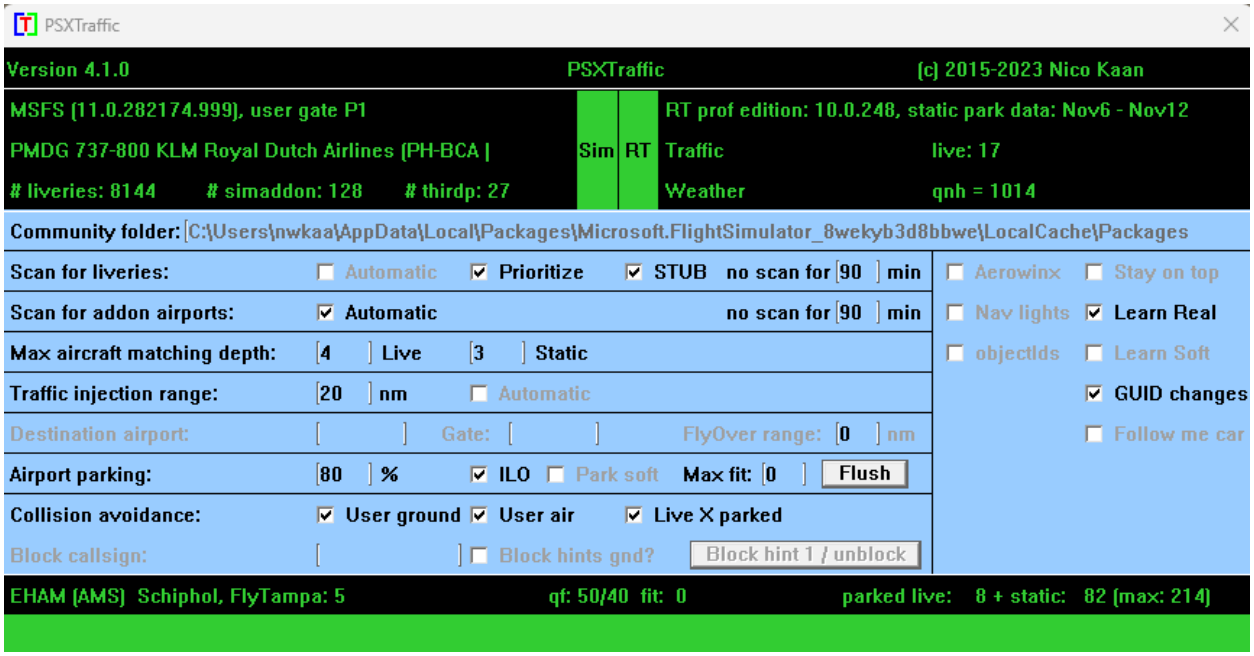
Select PSXTraffic for FSX/P3D/MSFS for Simulator in use.

All the necessary parameters will be set automatically:
Broadcast traffic to 3rd party apps, Disable PSX Traffic injection, Show GND TFC and Show AIR TFC should all be checked. Auto select GND/AIR TFC should be unchecked, localhost as Simulator Host IP.

3.3 Start PSXT or PSXTraffic

Start the program via the shortcut on your desktop.





These programs differ only in the way they get the RealTraffic data: directly from the RT servers (PSXT), or by a locally installed RealTraffic App (PSXTraffic).

The PSXT versions are more efficient and more stable. Use PSXTraffic only if you really want to see the RT Radar or you want destination learning (RT Prof license only).

The programs are interchangeable, they use the same parameters file, the same airport files, liveries, and so on.

At each start-up PSX/PSXTraffic will check if there is a new version available. If so, the green version message in the main window will turn yellow, and the new version number will be indicated. For patches and minor updates there is no obligation to upgrade, for a major version update there is. In the latter case you will be notified by the error message "error 99: This version has expired".

Important!

PSXTraffic will search for your Simulator Install Path first. If not found you have to do something:

MSFS:

Type a full path to your Community folder in the "Community Folder" text box of the GUI and press Enter. You may give it up/to or with Community included, like:

C:\MSFS\Community
or
C:\MSFS

P3D:
Type a full path to the top-folder of your Simulator in the "Install Path" text box of the GUI and press Enter, like:

D:\Program Files\Lockheed Martin\Prepar3D v5

You can change between P3D versions by adjusting this line.
Or if this method does not work, set hidden parameter (see 3.3.4) P3D_VERSION at 5 or 6

The black box below the header is divided in two sections enclosing two vertical status bars. The left section is about Flight Simulator; the right is about RealTraffic.

3.3.1 Sim vertical status bar ("Sim")

Black: not yet initialised, Red: not connected, Yellow: connected but Sim is not running, Green: all ok!

3.3.2 Flight Simulator info block (both for PSXT as PSXTraffic)

This block shows the name and version of your Flight Simulator and the title of the user aircraft. The title is in yellow if the wingspan is not known. If the user aircraft is parked at a gate, the id of the gate (in the airport file) is shown.

Also shown is the number of ai aircraft (liveries) available for matching and the number of available simaddon (MSFS) or ORBX (P3D), and third party airport files. If the colour of these numbers is yellow, it means that not all airport are off-line learned (filled with parking options).

3.3.3 RT vertical status bar ("RT")

Black: not yet initialised, Red: connection lost, Yellow: connected but no traffic data receiving, Green: all ok! If no data is being received over the last minute the bar lights up in yellow.

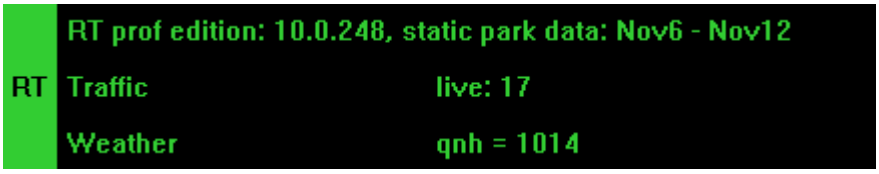
3.3.4 PSXT RealTraffic info block

This block shows the version of RealTraffic and the license you have.

The word "Traffic" shows if traffic messages are being received. If so the number of live aircraft (*flying, taxing, not the live parked*) in your Simulator is shown. The out of number is the number of aircraft received for your lateral range (but not necessarily spawned because of not wanted models (Gliders, ground vehicles, light aircraft if you do not have light aircraft, , or..).

The word "Weather" shows if METAR messages are being received If so the local qnh is shown or, in flight and park airport set, the QNH at the park airport.

3.3.5 PSXTraffic RealTraffic info block



This block shows the version of RealTraffic and the license you have.

Also shown is the week with RT ground traffic data, that is used for static parked aircraft options (offline learned airport files).

The word "Traffic" shows if the stream with traffic messages is connected. If so the number of live aircraft (*flying, taxing, not the live parked*) in your Simulator is shown.
The word "Weather" shows if the stream with METAR messages is connected. If so the local qnh is shown.

3.4 Parameters

The light blue area in the middle contains the check- and text boxes for parameters that can be set.

In the

C:\Users\<user_name>\AppData\Roaming\PSXTraffic folder
C:\Users\<user_name>\AppData\Roaming\PSXTraffic_P3D folder

a file **parameters.xml** will be created and updated.

You can always go back to all default settings by removing the file parameters.xml.

Parameters that cannot be set in the GUI are called "hidden" parameters, you must give them a value in the file parameters.xml.

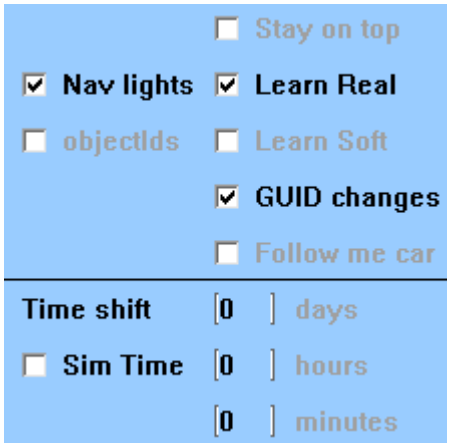
Important!

How to enter a value in a textbox in the UI?

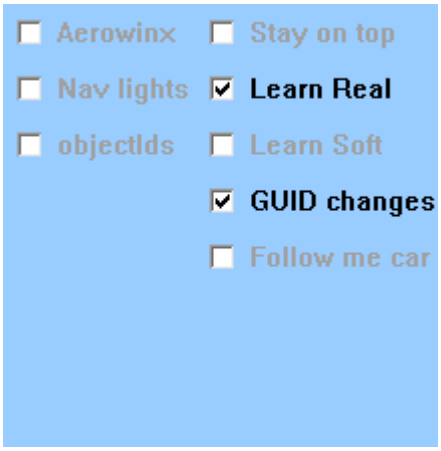
Put the cursor in the textbox (indicated by the light square brackets) key in or edit the value, and when you are ready **press the Enter** key to make the new value active! In the green line at the bottom feedback will be given about your new value.

4 Meta parameters

PSXT:



PSXTraffic:



4.1 Aerowinx PSX

Default: **false**

When checked, PSXTraffic will listen to the traffic data send by RealTraffic that is under control of Aerowinx PSX. This way PSX users that have MSFS or P3D as scenery generator will see live aircraft in that scenery too (and not only at the Aerowinx TCAS).

Note that in this mode PSXTraffic assumes a standard licensed RT. Learn Destination is not possible.

4.2 Port number

Default: **10747**

With the hidden (see 3.3.4) parameter PORT_NUMBER you may set a different port number for the communication with the RealTraffic app (and change the one in RealTraffic accordingly!).

4.3 Stay on top.

Default: **off**.

If on the PSXTraffic window will always stay on top of your screen

4.4 Random (PSXTraffic versions only, not in PSXT)

Default: **true**

If the hidden parameter RANDOM is set to false, there will be no randomness in livery matching and no randomness in the gates chosen for static aircraft.

You better leave this value at true. It is needed for users running PSXTraffic's at more than one computer at the same time (section 2.3) **only**.

4.5 GUID changes

Default: **true**

If the set to false, live aircraft in the sim will not be updated (a so called "livery change") to reflect new origin/destination parameters, new callsign info (resulting in a better livery match), re-initialisation of the lights. These (positive) changes have as side-effect that the GUID parameter of the

object changes and if you were watching at that aircraft in spot mode (for a video stream for instance) you would no longer see the aircraft (although it still is in the sim at the same place).

4.6 Nav Lights while parked.

Default: **true**

Navigation lights of static and live **parked** aircraft will be on. This parameter is not airport specific.

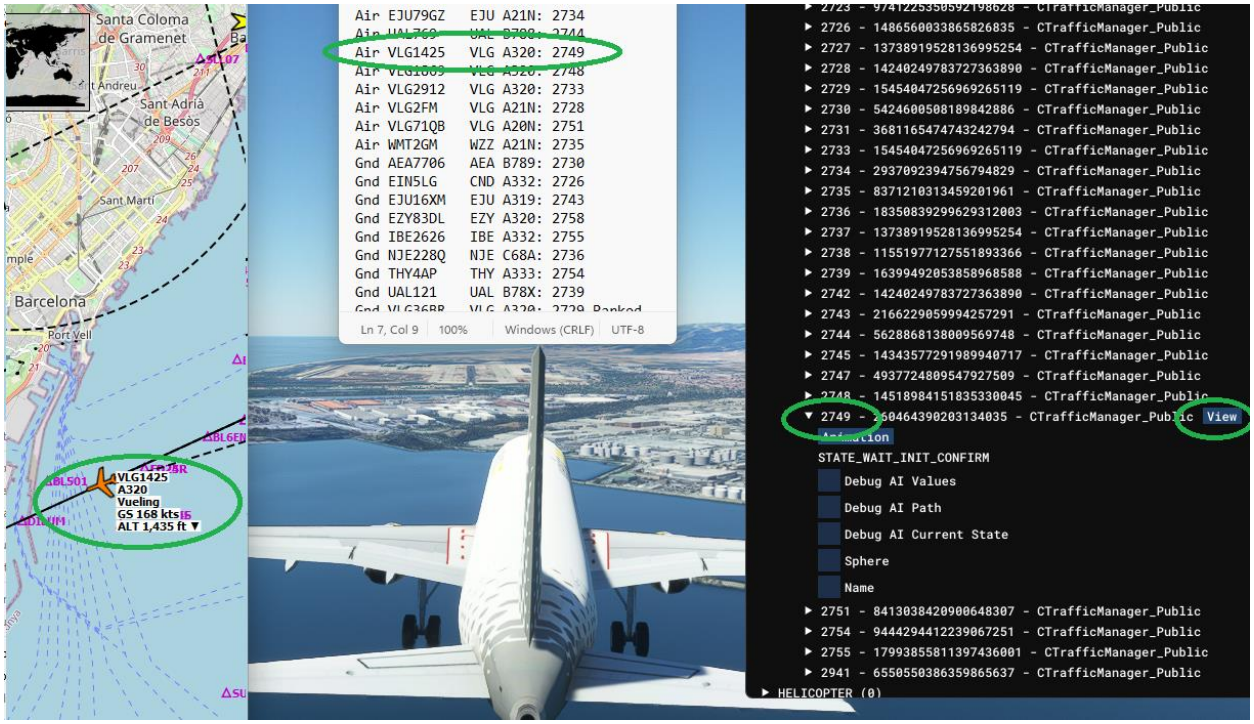
4.7 objectIds [MSFS only]

Default: **off**

Especially for Spotters!

If checked PSXTraffic will – every minute – generate a list of the live aircraft with their objectIds in SimConnect. Listed are Airborne/Gnd, Callsign, Airline, Type and whether or not Parked at a gate, and what gate.

With the objectId it is easier to spot an aircraft. Look in Little Nav Map for the callsign, pick up the objectId from the list and select the SimObject in Debug mode of MSFS: Debug -> Simobjects-> Containers -> Airplane -> Containers and click View.



The list is generated in the installation Folder of PSXTraffic, where you can also find the Log.txt, and is named **Callsigns2ObjectIds.txt**

4.8 Learn Real

Default: **true**

PSXTraffic comes with lots of airports in the archive that are already learned offline with a set of ground traffic data for one week. Normally this will do.

However, for some airports this is not sufficient. With this Learn option, PSXTraffic will learn the airport on-the-fly, on top of what is already in the archive.

The learned options will be saved in an airport file <ICAO>.xml in the folder
`airports\learned\stock`, or

airports\learned\simaddon, or
airports\learned\thirdparty\<publisher>
depending on the publisher, and is not overwritten by each PSXTraffic version update.

When an airport file is loaded, PSXTraffic reads the file from the archive plus the learned options from the file in the learned hierarchy.

If a real live aircraft positions at a gate or an aircraft park at a gate, the **airline, opr** (optional, see 7.1) and **type** combination is added as an option to the airport file with the attribute **day**=**"n"** (n = the code of the day (utc), 1 ... 7, 1 = Monday), plus the current half hour (utc time) is recorded with the attribute **hh="m".** The hour is rounded, the range is 0 ... 47.

The entry in the airport file may look like:

```
<aircraft>
  <opt airl="KLM" type="E190" day="3" hh="17,18" reg="PH-EXA" cs="KLM1002" />
</aircraft>
```

The airline and type codes must be according to ICAO rules.

This value is stored in your Learned airport file. The value in learned file overrules (but not changes) the value of Park Soft in the parameters.xml file (the UI). This way you can set different values for different airports.

Also the parameters Learn Soft (4.9), ILO (10.2), Park Soft (10.3) and Max fit (10.4) are stored in the learned airport file.

4.9 Learn soft

Default: **true**

[read first Learn Real above]

This value is stored in your Learned airport file. The value in learned file overrules (but not changes) the value of Park Soft in the parameters.xml file (the UI). This way you can set different values for different airports.

If the standard operating procedures (SOP) at an airport forces pilots to switch off their transponder after landing, or to switch it on when they take-off, there will be no traffic visible taxiing from/to gates and PSXTraffic will not learn and thus not update the airport file.

Hence, PSXTraffic will also use the aircraft that land and/or take off at the airport to update the airport file by adding the same data as in real learning, but with the attribute **soft="true"**, to a randomly chosen gate out of the gates that fit best. The coding of the hours is the same as in real learning.

Although the gate assignments will not be according to reality, you will at least see parked aircraft of airlines and types that fly to/from the airport at the right hour.

Light aircraft and business jets, or PVT airlines, will be placed at non cargo ramp positions (hence not at gates)

PSXTraffic also uses the info from live aircraft parked at gates in the airport file -if available- to improve the soft matching.
Note that real learning will always take place, soft learning as long as the real part of the Quality factor is < 60 or the airport has < 20 parking positions

4.10 Custom made airport files

There are three ways to add custom information (manual additions):

a) You may add park options to the parking positions in the learned file.

b) You may add a new **parking position** to the learned file that is not existing in the file in the archive.

Specify it in the same way as in the airport files, plus attribute **manual="true"**

c) You may create a **complete** airport file as **alternative** for one in the archive.

Specify the parameters **airport_name** and **publisher** and add parameter **handmade="true"**. **Switch on Learn Real and/or Learn Soft**.
No need to add the attribute manual to parking positions, as in b).

PSXTraffic will read the airport altitude from the archived version. The number of positions and the Qf will be calculated and added automatically later.

Place the file in the corresponding folder in the learned hierarchy. For example, suppose you want to define the airport file for Aerosoft EDDF, then place it in airports\learned\thirdparty\Aerosoft.

PSXTraffic will then take your file (only) if Learn Real and/or Learn Soft is true, and add learned options. In the airport line in the UI (h) will appear after the name of the publisher.

Note that in this case you do not benefit from the offline learned data in the corresponding archive file

4.11 Follow me [MSFS only]

Default: **off** (disabled if no folder "**car-Mitsubishi-Pajero**" has been found in Community)

It is not always easy to know if you have "taken over (blocked)" a real live aircraft, where it exactly is and when it is its (your!) turn to enter the runway. Therefore, this nice gimmick:

If follow me is checked and a **block** (see 11.6 and 11.5) is wanted and the user aircraft is on the ground, a follow me car will appear at the position of the real live aircraft, instead of not showing the blocked aircraft at all. There is no collision avoidance with the user aircraft, so you may run over it.

PSXTraffic will show the o'clock position and distance to the Follow Me car in the information line. At take-off the follow me car will disappear if its groundspeed is more than 45 kts.

Important: Make sure you have downloaded and installed the freeware Driveable Mitsubishi Pajero IO 2022 version **2.0**.

Be aware, go to the bottom of the site for 2.0!

Unzip and place the **car-Mitsubishi-Pajero** folder in your Community folder. That's all. There is no need to scan for Liveries.

4.12 Time Shift (PSXT and PSXT_P3D only)

With these boxes you may enter a time shift up to 4 weeks ago.
Ranges are: days: 0 .. 28, hours: 0 .. 23 and minutes: 0 .. 59

PSXT will read Traffic and Weather data with that time shift.

With the **Sim Time** checkbox the time in your Simulator determines the time in RT, but again, no more than 4 weeks ago is possible.

5 Scanning liveries

PSXTraffic must know what liveries are available in your Simulator for matching. It will scan for your installed ai aircraft (models/liveries). It searches for titles, airline codes, aircraft types and registry codes (tail numbers). By default, it tries to find your liveries automatically, however, see next section.

5.1 Automatic

Default: **On**

If you want to specify the paths to the folders with your liveries yourself, you should **uncheck** the **Automatic** check box.

Specify max 20 full paths to folders with AI aircraft. The parameters in **parameters.xml** file (see 3.3.4) are **`lfolder0 … lfolder19`**.

MSFS Examples:

```
<parameter name="lfolder1" value="D:\aig-aitraffic-oci-beta" />
<parameter name="lfolder2" value="D:\fsltl-traffic-base" />
<parameter name="lfolder3" value="D:\justflight-aircraft-traffic-fleet" />
```

P3D Examples:

```
<parameter name="lfolder1" value="D:\AI\FSPXAI" />
<parameter name="lfolder2" value="D:\AI\FAIB Boeing" />
<parameter name="lfolder3" value="D:\AI\FAIB Airbus" />
```

5.2 Prioritize

Default: **Off**

If **prioritize** is checked in a non-automatic scan for liveries, a livery for an airline + type combination in a higher folder has priority over a livery for the same airline + type combination in a lower folder. Hence liveries in the AI package indicated by lfolder0 have priority over liveries in the AI package indicated by lfolder1*. (0 is defined to be higher than 1 and so on).*
The one in the lower folder will be neglected and it will not be added to the "working set" of liveries (the file liveries\AI_liveries.xml) from which PSXTraffic performs the matching with a real live aircraft.

Information about the liveries that were skipped can be found in the file
`info\LiveriesSkippedOnPriority.txt.`

5.3 STUB

Default: **true**

If this parameter is true, liveries with a title ending with "_STUB" (as in the FSLTL package) will be **excluded** from the scan results.

5.4 no scan for XX minutes

Default: **1**, range 0 .. 90

If the value is 0 PSXTraffic will **always** scan (at the start of the program) for your liveries, otherwise it will scan if you have not run PSXTraffic for more than that number of minutes, or if you have changed one of the scan parameters.

More detailed info, for those interested, about the Liveries scan can be found in chapter 13

6 Scanning add-on airports

PSXTraffic must know what add-on airports you have installed. The task of the scan is to either generate empty **airport files** with information about the parking positions or provide short cuts to the already existing filled files in the **`airports\archive`** folders.

Information about your third-party airport files are stored in the subfolder **`my_thirdparty.`**
For MSFS, information about the airport files from Microsoft and Asobo that come with MSFS are stored in **`my_simaddon.`**
For P3D, information about the airport files from ORBX are stored `in` **`my_ORBX`**.

For MSFS, PSXTraffic uses the information in the BGL path, and or the Manufacturer and Creator information in the manifest file, for determining the **Publisher**. If that info is missing or just a few initials, PSXTraffic cannot determine the Publisher. This may happen for freeware airports, from fltsim.to for instance.

For P3D, PSXTraffic van use the BGL path only to determine the publisher. It is therefore recommended to group your add-on airports under subfolders with the name of the Publisher.

You can see the BGL path, Manufacturer and Creator info found in the generated -empty- airport file. Read more in FAQ #38.

MSFS only: Airports bought on the Marketplace (Store) are encrypted, PSXTraffic cannot read them, so don't buy there! See PSXTraffic FAQ #37 for a workaround.

6.1 Automatic (or manual)

Default: **On**

If you want to specify the paths to the folders with add-on airports yourself, you should **uncheck** the **Automatic** check box.

Specify max 20 full paths to folders with addon airports. The parameters in **parameters.xml** (3.3.4). are **`afolder0 … afolder19.`**

```
<parameter name="afolder0" value="D:\Addons\Airports" />
```

A folder search includes all sub-folders!

Information about the search is stored in the file **`output\info\AirportsScanLog.txt.`**

6.2 no scan for XX minutes

Default: **1**, range 0 .. 90

If the value is 0 PSXTraffic will **always** scan (at the start of the program) for your add-on airports, otherwise it will scan if you have not run PSXTraffic for more than that number of minutes, or if you have changed one of the scan parameters.

6.3 What to do with empty airports or airports that need an update?

Not all addon airport may already have been **off-line** learned (filled) by PSXTraffic and RealTraffic, or newer versions have arrived. If you have these, you will see a yellow message in the UI.

Please zip the folder "**TO_BE_LEARNED_OFFLINE**" (*you can find that in the PSXTraffic installation folder*) and mail that please to leveld757@gmail.com

7 Aircraft parameters

By default, PSXTraffic spawns business jets, large, and heavy types of live aircraft. Ground vehicles, Gliders, Airships and Balloons are not spawn.
If you have these liveries installed it will also spawn light aircraft (GA), high-performance aircraft (F16, F35 and so on) and helicopters. You can switch these categories off via hidden parameters in the parameters.xml file (see 3.3.4).
Note that your helicopter liveries should have Category=airplane in their aircraft.cfg file, and not Helicopter.

7.1 Live matching depth

Default 4, range 1 … 4.

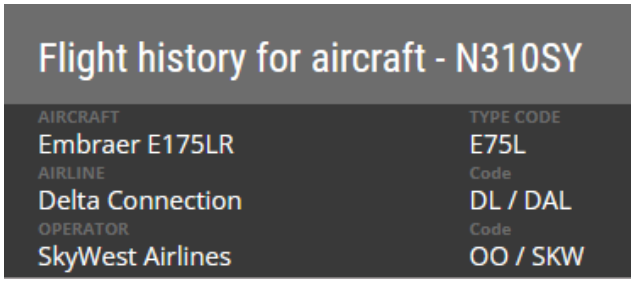
This parameter determines how many matching steps PSXTraffic will take at max in a search for a livery for a real live aircraft. The lower the better match (but you may see less aircraft).

Next to that PSXTraffic has info about what airline and type belong to a registration code in the file **`input\dbase.xml.`** It tries first a data base lookup based on registration code. In that file is also info about airlines that operated by other airlines.

With the information available a livery for a live aircraft is searched in a best effort matching process in up to four steps:

Step	Match	Indicated in the Log with
1	Match of an aircraft by registration code	=1=
2	Match of an aircraft by airline + operator* codes and type	=2=
3	Match an aircraft by airline + operator and similar type of wanted type	=3=
4	Match of a generic livery by airline ZZZ and type. [a generic livery: icao_airline (MSFS) or atc_parking_codes (P3D)=ZZZ] *This will result in a so called white or house paint.*	=4=

*operator: often operator is the same as airline, but there are operators that operate for an airline.



See this example from FlightRadar24, the **operator** is **SKW**, operating for **airline DAL.**

There are two special cases:

=c= means "matched" from cache, an aircraft that re-appeared, after a hick-up in RT
=u= means "updated" same livery but changes in origin and or destination and/or callsign

For Light aircraft, business jets and helicopters there are two extra steps, if live matching depth = 4

5	Match an aircraft by type (so whatever airline)	=5=
6	Match an aircraft by a similar type of wanted type (whatever airline)	=6=

7.2 Static matching depth

Default 3, range 1 …3.

This parameter determines how many matching steps PSXTraffic will take at max in a search for a livery for a static parked aircraft. For the steps see the similar steps defined in the previous section. For Light aircraft, business jets and helicopters there are two extra steps, as in the previous section.

8 Traffic injection parameters

PSXTraffic always adds a simple flight plan with departure and destination to the live aircraft in your Simulator.

In P3D you can then tick Flight Plan in the Information options for Vehicle labels. You will ten see **from -> to** displayed above each live aircraft.

External programs such as **Little Navmap** and **EFB** will benefit and show this information too.

This simple flight plan is just added information, it has no meaning for automated flights. PSXTraffic still controls each aircraft.

The following parameters all have effect on the traffic that will show up in your Simulator.

8.1 Lateral range

Default: **40 nm**, range 2 … 100 nm.

All airborne live aircraft within the lateral range of the user aircraft will be injected.
All live ground traffic at you origin or destination airport and within the lateral range will be injected.

If you are using FSUIPC make sure that the settings for "Limit TCAS range" on the misc tab are equal to the value in PSXTraffic. Don't use the FSUIPC Traffic limiter or other traffic limiters, that will disturb/ruin PSXTraffic's bookkeeping.

8.2 Automatic

Default: **on**.

The checkbox **automatic** comes in handy for pilots making a flight in cockpit view from departure to destination. If selected, PSXTraffic sets the lateral range parameter according to the stage of your flight. Hence, only the aircraft that may be of interest to you as a pilot flying will be injected in your simulator and will be visible at your TCAS, resulting in a as small as possible drop on your fps.

This mode is also recommended when flying to/from "heavy" airports such as EGLL and KLAX.

Note it is either auto range or setting the range manually (8.1).

9 User destination airport

The following parameters all have to do with the aircraft that you are flying, the so-called user aircraft.

9.1 User destination airport

Default **empty**.

Enter here the ICAO code of the airport you are (going to fly)/ flying to.
PSXTraffic will park static aircraft and put live aircraft at the airport when the airport is in sight, otherwise it will start (*too late …*) when you have landed.

9.2 Gate

Default **empty**.

Enter the name of the gate, as defined in the airport file if you want the gate to be excluded from a static or live aircraft. This will ensure that your gate is empty when you arrive.
The gate name you entered will be checked for validity and info is given whether your aircraft will fit at that gate.

The gate names in the airport file are not encrypted, look for the **id** field as in <parkpos id="**A91**"

9.3 FlyOver range

Default: **0**, range 0 .. 20 nm

If greater than 0, live ground traffic at airports you **"fly over"** is shown too. PSXTraffic takes the nearest airport within a radius (range) from your aircraft.

Static parked aircraft are not placed, to make this function fast and smooth. The UI will only show the airport code, airport name and publisher.

9.4 User flight plans folder

Default: **empty**.

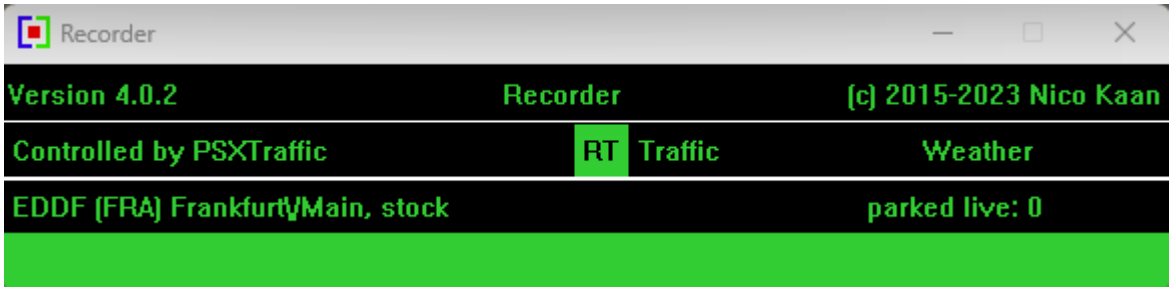
The hidden parameter FLIGHTPLANS_FOLDER defines the path to the folder with **.pln** files for the user aircraft. Enter either a subfolder of PSXTraffic or a full path.

PSXTraffic may detect your Destination airport from a **.pln** file exported by a Flight Planner (such as PFPX and ProATC) in your flight plans folder. It will take the latest created **.pln** file with a <DepartureID> equal to your actual origin airport. The destination airport will be taken from the <DestinationID> entry in the Flight Plan. The airport codes must be according to the ICAO format.

MSFS: If PSXTraffic sees "pmdg-aircraft" in the flightplans folder path, it will look for the destination in a .rte flight plan generated by **SimBrief**.
Note that PMDG expects flightplans in ..\Community\pmdg-aircraft-73X\Config\Flightplans

9.5 **Destination Learning** [PSXT or PSXT_P3D and RT Professional only]

Always on, but available only if you have a RealTraffic professional license.



PSXTraffic will make use of a separate **Recorder** program that opens a **second** traffic data stream from the RealTraffic servers, to learn and collect the live aircraft that park at your destination airport, while you are flying to it.

After you have filled in a destination (don't forget to type Enter), and if the destination is > 30 nm away, PSXTraffic will start Recorder minimized, you don't have to do anything.

When the destination airport is in sight, PSXTraffic will command the Recorder program to generate a snapshot file with the positions of actual live parked aircraft, and the positions and times of aircraft that have parked but has left, and exit.

The snapshot file is immediately used by PSXTraffic to set up the destination airport with the actually parked aircraft, and to add all the options to your learned airport file.

This way you'll get the most actual situation, the right airlines and types at the right gate, more than you would get with the standard license. And you will have a better learned airport file.

Optional: together with the ILO option (10.2) you will have the possibility to work with sufficient live (parked) aircraft only at arrival.

The actions of Recorder are logged in **RecorderLog.txt**, in the same folder as Log.txt.

10 Airport parameters

10.1 % parked Live + Static

Default: 30

Value may be changed in the GUI, to be stored locally.

This parameter determines the wanted and maximum number of parking positions at the airport that may have a live or static parked aircraft. This value is the percentage of the number of parking positions at the airport. If the maximum has been reached and a new live want to park, the oldest static aircraft will disappear.

If you set the value at 0, only live aircraft may and will park (without a max)!

A live aircraft that has parked but has no longer an Real Traffic data connection will remain parked for maximum 3 hours (heavies) or 90 minutes (the rest), or it may of course start its return flight.

PSXTraffic will automatically search for the best allocation of static parked aircraft, based on the parking options with day, hour, and real info found in the airport file. In the Log.txt you'll see a message whether there were sufficient parking options for the hour of the day or not.

PSXTraffic will first search for options for the actual half hour of the day. If there are not sufficient options found it will search for the actual half hour -1, if still not the wanted number has been found it will try actual half hour -2 half hours, etc, until -47 (and may cross a day boundary).

Every 20 minutes necessary changes in parking positions will be re-calculated and implemented within a minute (unless ILO is true, see below).

If you fly away from an airport without a destination set the static parked aircraft will stay till you are >= 40 nm away from the airport ("because you maybe want to return.

If you fly away from an airport **with** a destination set, the static parked aircraft will be removed if you are >= 5 nm from the airport.

If you fly to an airport, with destination set, static aircraft will be parked if the airport is within your lateral range or at 20 nm distance.

10.1.1 Custom default parking percentage

Alternatively, one may set a hidden parameter, directly in the parameters.xml file, see 3.3.4,
CUSTOM_PARKING_PERCENTAGE value range -1 … 100.
If >=0, this value will **overrule** -but not overwrite- the locally stored value, and or the default value of 30.

10.2 ILO (Initial load only)

Default: **false**

If on, PSXTraffic will generate static aircraft at the first load of the airport, or after a percentage change by the user. No adjustments every twenty minutes.

If on, and you have a professional RT license, and there are live parked aircraft, no static aircraft will be parked at first load. The live parked from the snap shot from **Recorder** (9.5) will be taken as first load instead.

Live parked aircraft will stay parked, until another live aircraft needs the gate or it starts a new flight; the time constraints (see 10.1) do not apply.

This value is stored in your Learned airport file. The value in learned file overrules (but not changes) the value of ILO in the parameters.xml file (the UI). This way you can set different values for different airports.

10.3 Park soft

Default: **true**

If checked, soft options from the airport file will be used for static parking if there are not sufficient real options.
This value is stored in your Learned airport file. The value in learned file overrules (but not changes) the value of Park Soft in the parameters.xml file (the UI). This way you can set different values for different airports.

10.4 Max fit

Default: 47, range 0 .. 47

If 0, only real, and soft (depending on Park Soft in 10.3), options in the airport file for the current day of the week and current half hour will be used for parking of static aircraft.

For values > 0, say **n**, it will search back for **n** half hours (max 47 == 1 day), see also 10.1.

This value is stored in your Learned airport file. The value in learned file overrules (but not changes) the value of Now in the parameters.xml file (the UI). This way you can set different values for different airports.

10.5 Flush button

If pressed, PSXTraffic will remove all static parked aircraft and set ILO (10.2) to true. This will leave you with live (parked) aircraft.

11 Collision avoidance

At busy airports it is sometimes difficult to find a place in the departure queue. At London Heathrow for instance, there may be 13 aircraft waiting for take-off in two parallel queues, using different runway entries. Sometimes there is even an occasional aircraft landing at that runway….

You can activate near collision avoidance functions, or you can block one of these aircraft (not showing up) and take its place.

The following features may make your live as a pilot easier…

11.1 Collision avoidance user aircraft with other aircraft (live + static) on the ground

Default: **true**

PSXTraffic detects a near collision if two aircraft are at a distance from each other less than the sum of their wingspans divided by two.

If checked collisions between the user aircraft and live aircraft on the ground or static parked aircraft will be prevented by removing the other aircraft until it is safe.

11.2 Collision avoidance of the user aircraft with live aircraft in the air

Default: **true**

PSXTraffic detects a near collision if two aircraft are at a distance from each other less than 4 miles, within an altitude difference of 1’000 feet and a “width” less than 700 ft.

PSXTraffic removes/hides the live aircraft temporarily until it is safe to show it again.

As an extension PSXTraffic also removes a live aircraft when during a landing, starting from 500 ft above ground, it detects a live aircraft right in front (entering or already on the runway), within 1 nm.

11.3 Collison avoidance live aircraft x static parked aircraft

Default: **true**

PSXTraffic detects a near collision of two aircraft are at a distance from each other less than the sum of their wingspans divided by two. If checked collisions between live and static parked aircraft will be prevented by removing the static parked aircraft.

11.4 Callsign text box (for blocking)

Default **empty**.

Enter the callsign of a real live aircraft. As a result, the real live aircraft will not be spawned in your Simulator, or if it already was in your Sim, it will no longer be visible in your Simulator.

You may as well enter an already spawned aircraft or one that will be spawned shortly.

On the ground you may also replace the blocked aircraft by a follow me car (see 4.11) .

11.5 Blocking hints?

Default: **off (ground aircraft only)**

If checked, and there is no blocked aircraft (see next section), PSXTraffic will show nearby aircraft for blocking.

If user aircraft is on the ground, PSXTraffic searches, within 300 meters in all directions, for live non-parked aircraft with groundspeeds < 45kts.

Shown are the (max) three best, it favours the one in front of you.

11.6 Block/Unblock push button

If pushed and Callsign is empty, PSXTraffic will take the hint#1 (see information line of the GUI) as callsign for blocking automatically, no need for entering it in the callsign text box yourself. If the callsign is not empty, it will unblock.

For VR users: the block/unblock trigger can also be performed by toggling a light switch On-Off-On or Off-On-Off. Make sure you wait a second before toggling back, but do not wait longer than 3 seconds. If you toggle too fast or wait too long the trigger will not be recognised.

The following five switches may be used for this trigger mechanism, to be defined by you with the parameter `LIGHT_SWITCH_CODE`

Values are: 2 = Beacon, 4 = Landing, **8 = Taxi (default),** 128 = Wing and 256 = Logo
You can test for your user aircraft if one of these switches is recognised by PSXTraffic. If so you will get a message in the Log.txt, like this: "*Taxi light switch (code 8) triggers!*".

Note that the PMDG B738 supports all light switches and the Fenix A320 supports none. The Captain Sim B763's taxi light switch works.

12 Session information

During a run of PSXTraffic a minimum of essential information is provided by several means.

12.1 Log file

During each PSXTraffic session a file **Log.txt** will be written in the PSXTraffic installation folder, whilst saving up to 2 previous version (**Log0.txt** and **Log00.txt**).
Have a look into this file every now and then, and if you have problems add it to your post in the Forum.

12.2 Recorder Log file (RT Professional only)

During each Recorder session a file Recoder**Log.txt** will be written in the PSXTraffic installation folder, whilst saving up to 2 previous version (**RecoderLog0.txt** and **RecoderLog00.txt**).
Read about the Recorder in section 9.5

12.3 Parked airport line

The black box below the parameters section provides information about parking of aircraft at your departure and/or destination based on the information in the offline learned airport file.

The ICAO and IATA code (if available) of your origin or destination airport. The airport name and the publisher from the airport addon.

(h)
If the file is handmade (see 4.10 section c) (h) will appear after the name of the publisher.

:n gives the number of real and soft updates found during this session

qf:<real[/soft]>

The quality factor from the airport file (0 .. 100) as well for real as for soft. If real >= 60 there are no soft options so there is not shown a Qf soft.

fit: <n>

How the wanted percentage of static aircraft fit with the parking options in the airport file, in half hours. If "**now**", all static aircraft fit with the current day and half hour. If 1 they fit within current half hour -1 half hour, and so on till - 47. (See 10.1).

A ***** following the fit text means that **soft** options has been used (too).

Shown is the number of live aircraft parked, the number of static aircraft parked and between parenthesis the max allowed total number. *Note that the live parked are not included in the live count in the RT section of the GUI*

12.4 Messages

At the bottom line of the GUI several messages appear.
Information messages show up in yellow font against a green background**.**
Error messages show up in yellow font against a red background. Error messages are severe. After reading, you must exit PSXTraffic with the Windows X button.

13 In depth information about Liveries scan

13.1 How are liveries in your Sim found?

Liveries are found by scanning **aircraft.cfg** and **sim.cfg** files.

Special case **IVAO MTL** (P3D/MSFS) **/ ivao_x-csl** (MSFS):
The folder with the MTL must be named IVAO_MTL (capitals) or ivao_x-csl (lower case) otherwise PSXTraffic will scan the files as if they were normal liveries!

13.2 Registration codes (tail numbers)

There are two possible ways to specify registration codes for liveries, **in-line** and **offline**. The in-line definitions are processed first.

13.2.1 In-line specification of registration code

The registration code is taken from the **atc_id=** line. Registration codes may be 12 characters max. One registration code (only) is recognized and recommended to make it unique.
If you want more registration codes to map to the same livery, you must the offline method of section 13.2.2.

If a registration code is used by more than one livery, the first one found will be kept.

The errors found in processing the inline regcodes can be found in the file **output\errors\InlineRegcodes.txt.**

13.2.2 Offline specification of registration code(-s)

In folder **input\my_regcodes**, one may put one or more text files. The name is not relevant but it must be unique, with extension **.txt**.

In the file one should specify the (exact) title of the livery followed by a semicolon, followed by a list of registration codes separated by commas. A livery title may not appear in more lines.

Liveries that have just one registration code in an offline file, will be treated as being **special**, they will only show up as live aircraft, not as static parked aircraft.

If one adds **;#** to a line, **all** registration codes will be treated as special, for example:
AIGAIM_easyJet Europe Airbus A320-200SL - Europcar;OE-IVC,OE-IVT;#

This method has the advantage that you do not have to edit the [fltsim.x] sections, which is important when using AIG. Another even more important advantage is that one can share these files. Files for AIG are already available in **input\regcodesAIG.** All PSXTraffic users automatically benefit**.**
Note that the data in the regcodesAIG folder is processed first before PSXTraffic processes any definitions supplied by the user.

An offline specified registration code **overrules** one defined in-line (*the inline regcodes are processed first*), and a newer offline definition overrules an earlier offline definition for the same registration code!

In the Log.txt file a summary is given of the results, and the details of replacements are summarized in **output\info\OfflineRegcodes.txt.**

If you specify a livery title per line only, PSXTraffic will treat it as a **don't use** livery.

13.3 Files about errors found in the scan

With these files it will be easier for you to track/pinpoint, and manually repair errors in your AI aircraft package(-s). You should do that if you want to make the best out of your AI aircraft, otherwise you can also choose to go on without paying attention to it.

- **output\errors\Fixes.txt**
  Some obviously wrong Aircraft type codes will be fixed automatically.
  Some obviously wrong airline codes (found in atc_parking_codes=) are fixed as well.
  The fixes are applied internally, your **aircraft.cfg** file(-s) remains untouched.
- **output\errors\InvalidAirlineCodes.txt**
  This file contains invalid ICAO airline codes.
- **output\errors\InvalidAircraftCodes.txt**
  This file contains invalid ICAO aircraft type codes.
- **output\errors\InlineRegcodes.txt**
  It shows output\errors in the [fltsim.x] sections about non-unique registration codes.
- **output\errors\InvalidAircraftCfgFiles.txt**
  Contains invalid aircraft.cfg files. Problems may be:
  - [fltsim.x] numbering not consecutive.
  - missing atc_model keys, icao_designator etc.
- **output\errors\IncompleteFltsimSections.txt**
  The file contains the incomplete [fltsim.x] sections; a section is incomplete:
  - if the title key is not unique
  - if the texture key is invalid
  - if the model key is invalid
  - if the sim key is invalid
  - if there is more than one title key
  - if there is more than one atc_id key
  - if there is more than one atc_parking_codes key

  If incomplete, the livery is skipped.

13.4 Files with information about the scan

These files give info about non-erroneous issues from the search.

- **output\LiveriesScanned.xml**
  It shows the results of the liveries scan. These are the liveries that will be used in matching.
  Warning: do not make any changes in this file, it is read by PSXTraffic when a scan is not needed because of 5.4.

- **output\info\AircraftCfgWithNoFltsimSection.txt**
  This file gives all the aircraft.cfg file(-s) that are "empty", having no [fltsim.x] section.
- **output\info\AirlineTypeConversions.txt**
  It shows the conversions of the airlines + types found in your liveries (from the aircraft.cfg files) to the values found in the PSXTraffic database. The corrected values are stored internally, your aircraft.cfg files remain untouched.
- **output\info\LiveriesSkippedOnPriority.txt**
  See 5.2
- **output\info\OfflineRegcodes.txt**
  It shows the results of processing the offline registration code files (13.2.2).