

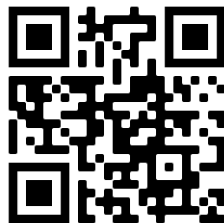
PSXT Manual



version 16.7.2

Oct 18, 2026

by Nico Kaan



This manual can be discussed and questioned online
in your own language with my ChatGPT PSXT Tutor
[ChatGPT PSXT Tutor](#)

Copyright © 2015–2026, Nico W. Kaan, The Netherlands. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means—electronic, mechanical, photocopying, recording, or otherwise—without the prior written permission of Nico W. Kaan.

1	Introduction	4
1.1	System Requirements	5
1.2	PSXT software	5
1.3	Third-party software used	5
1.4	Disclaimer	5
2	Installation	6
2.1	Download and install PSXT	6
2.1.1	Update PSXT to a new version	6
2.1.2	Complete removal of PSXT from your PC	7
2.2	Install the RealTraffic Application [optionally]	7
2.3	Using multiple computers running MSFS+PSXT at the same time [RTApp only]	7
2.4	Install AI aircraft packages	7
3	Operations	8
3.1	Start and configure Flight Simulator	8
3.2	Start PSXT	8
3.3	Compact / Expand button	8
3.4	Info blocks	8
3.4.1	Sim vertical status bar	8
3.4.2	Flight Simulator info block	9
3.4.3	RT vertical status bar	9
3.4.4	RealTraffic info block	9
3.5	Parameters (the light blue area)	9
3.6	Park airport information	9
3.7	Messages line	9
4	Configuration	10
4.1	MSFS selector [MSFS only]	10
4.2	QNH unit	10
4.3	RealTraffic source selector	10
4.4	Stay on top	10
5	Data sources	11
5.1	Airports	11
5.1.1	Skip scan	11
5.1.2	TBLO button	11
5.2	Liveries	11
5.2.1	Skip scan	12
5.2.2	Automatic checkbox + drop-down list (MSFS only)	12
5.2.3	MSFS: Manual mode	12
5.2.4	P3D: Manual mode (by default)	12
5.2.5	Priority	13
6	Collision avoidance	14
6.1	Collision avoidance user aircraft with other aircraft (live + static) on the ground	14
6.2	Collision avoidance of the user aircraft with live aircraft in the air	14
6.3	Type of traffic selector	14
6.4	Aircraft push button	14
6.5	Block push button	15
6.6	Unblock push button	15
7	Traffic injection	16
7.1	Radius	16
7.2	Auto radius	16
7.3	Max Live	16
7.4	Taxiing	17
7.5	Max matching depth	17
7.6	With extra TCAS coverage above this FL	18
7.7	Include four extra categories of aircraft	18
8	Parking	19
8.1	Update interval (min)	19

8.2	Park button.....	19
8.3	Destination	19
8.4	Reserved stand	19
8.5	(Within) range nm	19
8.6	Below height (above ground)	19
8.7	RT parked only	20
8.8	Max fit.....	20
8.9	Max park	20
8.10	Auto park	20
8.11	Target parking percentage	20
8.12	Week	21
8.13	Soft	21
8.14	Flush button.....	21
9	Park airports.....	22
10	Airport files.....	23
10.1	Custom made airport files.....	24
10.2	Editing park_airport.xml	24
11	Miscellaneous parameters	26
11.1	Random.....	26
11.2	Flight messages.....	26
11.3	Aerowinx PSX (RTApp only)	26
11.4	GUID changes	26
11.5	Object ID list (MSFS only).....	26
11.6	Nav Lights	26
11.7	Port number (RTApp only)	26
12	Time Shift [RT Direct only]	28
13	Park airport information	29
13.1	First line	29
13.2	Second line	29
13.3	Third line	29

1 Introduction

This manual applies to two versions of PSXT, one for MSFS2020/MSFS2024, and one for P3D.

In this manual the term PSXT is used generically and may refer to either of the two.

With PSXT, you can fly with real-time traffic. The traffic data is provided by a separate source called RealTraffic. The data can be read from the RealTraffic App or directly from RT Servers.

RealTraffic streams aircraft position updates every three to 5 seconds, or occasionally every ten seconds. PSXT buffers the traffic for 30 seconds for vectorisation. Including internet delay, the aircraft you see are typically only about 45 seconds behind real time.

The traffic data originates from ADS-B transponder signals. This works very well while aircraft are airborne. On the ground, it depends on the airport's policy: at some airports, pilots are required to keep transponders on, while at others they must switch them off. Consequently, you may see little taxiing traffic at some airports, whereas at others, you can follow aircraft taxiing to and from stands.

The live aircraft injected (spawned) into your simulator represent actual aircraft present at that moment. They are controlled by live ATC. The aircraft (models and liveries) are spawned as realistically as possible, depending on your installed AI models and parameter settings.

PSXT determines the position of each live aircraft 100 times per second by calculating latitude, longitude, altitude, bank, and pitch. At every frame, PSXT sends the most recent calculated position to your simulator via SimConnect. Therefore, the number of positions visible depends on your simulator's frame rate.

Live airborne traffic is shown within a lateral range of 2 to 100 NM from the user aircraft. Live ground traffic and parked aircraft are shown at airports where you land, remain, or optionally fly over.

PSXT knows the parking positions at each airport in your simulator. It places departing live aircraft at stands with the correct heading. Likewise, aircraft completing their flight are positioned precisely at their stand. In MSFS, PSXT also knows the exact locations and elevations of all runways and uses this information when aircraft land and take off.

While on the ground, PSXT also generates static parked aircraft. It first uses data from RealTraffic and then supplements this using options defined in so-called **airport files**. These files are created offline by PSXT and RealTraffic and are updated weekly to reflect changes in airlines, aircraft models, seasonal schedules, etc.

Airport files include details about parking positions, airline usage, aircraft types, and corresponding time slots (in half-hour increments). PSXT tries to match current UTC time as closely as possible to these options. It will dynamically update parked aircraft for the next half-hour by adding and removing aircraft.

When you depart from an airport, all aircraft at that location (static and live) will be removed. As your destination comes into range, PSXT places static parked aircraft at the appropriate parking positions, making the best possible match based on the current airport file.

PSXT controls gear, throttle, flaps, wheel rotation (aligned with speed), navigation lights, taxi lights, landing lights, cabin lights, logo lights, strobes, and beacon lights.

The performance impact of PSXT is minimal — it uses less than 0.5% of your CPU. Injecting AI aircraft into your simulator naturally affects FPS, but even mid-range PCs can achieve 40+ FPS.

If you're new to PSXT, follow the Quick Start Guides.

For questions and discussion, use the Support Forum at AVSIM and/or consult the FAQ.

1.1 System Requirements

- **Windows 10** or later, Windows **11** is the **preferred** OS.
- **Microsoft Visual C++ Redistributable for Visual Studio 2022**
Install Microsoft Visual C++ Redistributable for Visual Studio 2022, version 17, architecture x64.
It can be [downloaded here](#) at Microsoft (scroll down to **Download Visual C++ Redistributables**). Run **VC_redist.x64.exe**

1.2 PSXT software

PSXT is written in C++23 with Microsoft Visual Studio 2026.

All PSXT-related files are coded in UTF-8.

1.3 Third-party software used

PSXT uses the following libraries via **vcpkg**:

libzippp, libzip, pugixml, nlohmann-json, ms-gsl, libsodium, curl and winreg.

1.4 Disclaimer

The data provided by RealTraffic is not always accurate or complete. This may result in the following issues:

- Some airports have no ground coverage.
- Aircraft may vanish from the stream for over 30 seconds, causing them to disappear from your simulator before reappearing later.
- Taxiing aircraft may temporarily disappear if the ADS-B signal is blocked by buildings.

Note also that some add-on airports may not exactly match real-world layouts and could lack updated taxiways or runways.

2 Installation

2.1 Download and install PSXT

Download the PSXT software for your simulator from my site and unzip it.

Install at your computer running MSFS/P3D.

The default install locations are C:\PSXT or C:\PSXT_P3D

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!

Run the installer.

Administrator rights are only needed if you install into *Program Files* (so don't do that).

For a custom location such as C:\PSXT, **administrator rights are not required**.

After installation you will have a folder containing the files and a few subfolders containing:

The executable PSXT.exe. On your desktop you will find a shortcut to the executable.

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

The docs sub-folder contains several documents. The PSXT_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.

The **input** folder contains data needed by PSXT.

AIG-regcodes info, the database and airports data files are stream-encrypted files.

These files are updated automatically whenever a new version is uploaded to the PSXT web server.

Please note that these files are for PSXT use only and should not be opened manually.

When you run PSXT the following files will be created:

The output folder, with two subfolders \info and \errors with information about the livery and airport scans made by PSXT.

The **airports\learned** with its subfolders contains data that is learned on the fly on your system, 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. An <icao>.txt file contains the link to the file in the archive at the PSXT website. An <icao>.xml is an empty airport file, there is no file available in the archive. PSXT will have placed this file in the TBLO (to be learned offline) folder too (see 4.1.2). 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. A .txt file contains the link to the file in the archive at the PSXT website. An <icao>.xml is an empty airport file, there is no file available in the archive. PSXT will have placed this file in the TBLO (to be learned offline) folder too (see 4.1.2). This empty airport file is used until a file in the archive becomes available.

2.1.1 Update PSXT to a new version

When PSXT starts, it automatically checks for a new version. If one is available, it will download the installer (install_new_version.exe) and place it—along with a README.txt and the release notes—in the PSXT\new_version folder.

PSXT will then close and open the PSXT\new_version folder in File Explorer. Simply run the installer (.exe). Administrator rights are normally not required (see README.txt for details).

The installer will first uninstall your current version and then install the new one, so there's no need to uninstall manually. Your language preference and license acceptance are preserved from the initial installation, making the update process quick and seamless.

2.1.2 Complete removal of PSXT from your PC

Removing PSXT from your PC can be done by running the uninstaller from the Programs or Apps list, just like any other Windows application. To completely remove all files, delete the installation folder and the folder

C:\Users\<user_name>\AppData\Roaming\PSXT (or whichever location applies in your setup).

2.2 Install the RealTraffic Application [optionally]

PSXT works in RTApp mode (see 4.4) with a RealTraffic app that is installed on the same PC or on another machine in your LAN. No configuration in PSXT is required. In the RT App, point at the IP address where PSXT is running and you're good to go.

For information about purchasing a Real Traffic license and obtaining an API token, see the Quick Start Guide.

2.3 Using multiple computers running MSFS+PSXT at the same time [RTApp only]

If you want to run a setup with different computers, each running MSFS and generating part of the simulator screen, do the following:

Install PSXT on all PCs. On one PC, set the hidden parameter PSX_SERVER_HERE=True, and on all the others set PSX_SERVER_HERE=False. Install the RealTraffic app on one PC; it does not matter which one.

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

Important: In this setup you must switch **Random off** in all PSXT instances otherwise each PSXT instance may react differently when choosing a livery or choosing a static parked aircraft location.

2.4 Install AI aircraft packages

Recommended packages are AIG, FSLTL, JustFlight and IVAO_MTL (see the Quick Start Guide).

Important if you are using AIG in MSFS:

The lights of AI aircraft are also controlled by MSFS, which may conflict with PSXT and is sometimes completely wrong. You may see taxiing aircraft with strobes on, for instance. To fix this, go to

\Community\AIG-aigtraffic-modelbehavior\ModelBehaviorDefs\AIG\Customization.xml
and set AIG_TC to False in AIG_TC_PSXT_Check_Parameters

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 **1x simulation rate** (no time compression)
Rationale: Running with time compression gives incorrect UTC times for new learned entries to the airport file, because the time in RT is no longer synchronous with your Sim time.
- 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 are set at **0%!**
- P3D: Select an aircraft as the vehicle (pedestrians and avatars are not allowed)
- MSFS20/24: See the [Quick Start Guide](#)

3.2 Start PSXT

Start the program using the shortcut on your desktop.

During each PSXT session a file **Log.txt** will be written in the PSXT installation folder.

PSXT for MSFS will automatically switch to the Sim you are running (2020 or 2024).

To select the MSFS version manually, use the selector in the top left corner of the blue area in the UI.

Select the RT source.

PSXT will search for your simulator install path. If it is not found, a folder picker will appear.

MSFS 2020 / MSFS 2024: Select the Community folder in the folder picker that appears (2024: not Community2024). It must be the Community folder in the default installation of MSFS, the folder whose parent is Packages!

P3D: Select your P3D install path up to Prepar3D v5 or Prepare3D v6

You can also force the version with the hidden parameter (see 3.5) P3D_VERSION at 5 or 6

The header gives version information, program name, copyright statement and an Expand/Compact button.

The black box below the header is divided into two sections each containing a vertical status bar. The left section relates to Flight Simulator, while the right section displays the status of RealTraffic.

Note that you can also use the scroll wheel of your mouse to select a value in the drop down boxes!

3.3 Compact / Expand button

Default: Compact

This toggle button changes the main window between full and compact view. The full view is only needed to set parameters. When flying or spotting compact view is sufficient.

3.4 Info blocks

3.4.1 Sim vertical status bar

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

3.4.2 Flight Simulator info block

This block displays the name and version of your Flight Simulator. If the user aircraft is parked at a stand, the stand ID (as defined in the airport file) is shown.

Additionally, the block shows the number of AI aircraft (liveries) available for matching, as well as the number of available sim add-on (MSFS) or ORBX (P3D) and third-party airport files. If any of these numbers are displayed in yellow, it indicates that not all airports have been offline learned (i.e. filled with parking options).

3.4.3 RT vertical status bar

Black: not yet initialised, Red: connection lost, Yellow: connected but no traffic data received, Green: all OK! If no data has been received during the last minute the bar lights up in yellow.

3.4.4 RealTraffic info block

This block shows the version of RealTraffic and your license.

However, if RT_QUALITY (9.8) is checked, every 5 minutes the average and standard deviation of the time between two data points in the RT traffic data stream is given.

The word "QNH" shows if METAR messages are being received; if so, the local QNH is shown. If there is no QNH data for that area, the word QNH is in yellow.

The word "live aircraft" shows if traffic messages are being received. If so the number of live aircraft (flying or taxiing) in your Simulator is shown. If there are no aircraft for that area, the word is in yellow.

Note that parked live aircraft are not included in the live count, that number is given in the status line in the airport area at the bottom (11.1.3)

3.5 Parameters (the light blue area)

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

In the **C:\Users\<user_name>\AppData\Roaming\PSXT** folder

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

You can always revert to the default settings by deleting the parameters.xml file.

Parameters that cannot be set via the user interface are called "hidden parameters"; you must assign values to these directly in the parameters.xml file.

The parameters are described in Chapters 4 through 9.

In compact view, this block is not shown.

3.6 Park airport information

This block gives information about the *actual parking situation* at an airport. For details, go to Chapter 11.

If there is no airport in sight and you are in **compact** view, this block is **not shown**.

3.7 Messages line

On the bottom line of the GUI several messages may appear.

Information messages appear in white font against a green background.

Error messages appear in white font against a red background. Error messages are severe. After reading, you must exit PSXT with the Windows X button.

4 Configuration

4.1 MSFS selector [MSFS only]

This will be set automatically to the sim you are running (MSFS 2020 or MSFS 2024). But you may switch over to the other sim manually with this drop box.

4.2 QNH unit

Switch here between hPa (default) and inches, for the qnh value in the UI.

4.3 RealTraffic source selector

Default: **RT Direct**

A choice between **RT Direct**, using the RT Servers directly, or **RT App**, the RealTraffic Application.

4.4 Stay on top

Default: **off**.

If on the PSXT window will always stay on top of your screen.

5 Data sources

5.1 Airports

PSXT must know what (add-on) airports you have installed. The purpose of the scan is to either generate empty **airport files** (10) containing parking position data, or to create **shortcuts** to existing files in the online archive.

PSXT will by default perform a scan when the program is started, unless the **skip scan** checkbox is ticked).

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

For P3D, information about the airport files from ORBX are stored in **my_simaddon**.

For MSFS, PSXT uses the BGL file path as well as the **Manufacturer** and **Creator** fields from the *manifest.json* file, for determining the **Publisher**. If that information is missing or just a few initials, PSXT cannot determine the Publisher, a situation that commonly occurs with freeware airports from sources such as flightsim.to.

For P3D, PSXT can only rely on the BGL file path to determine the publisher. Therefore, it is recommended to group your add-on airports into subfolders named after the Publisher. You can view the BGL path, Manufacturer and Creator info found in the generated (empty) airport file. See FAQ #38 for more information.

A folder search includes all subfolders! Information about the search is stored in the file `output\info\AirportsScanLog.txt`.

MSFS

PSXT always scans your Community, Community2024 (MSFS2024 only) and Official folders for airports (symbolic links included).

P3D

Define a maximum of 10 full paths to folders with add-on airports.

The parameters in `parameters.xml` (see 3.5) are named `afolder0` ... `afolder9`.

Example:

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

5.1.1 Skip scan

Default: **off**

If checked, the scan for airports at the start of the program is skipped.

5.1.2 TBLO button

Not all add-on airports may already be in the archive on my web server, or newer versions may have arrived. Read more on airport files in section 8 first.

If so, the button will turn yellow

Please do the following: **Push the button and mail 'tblo.zip'** to leveld757@gmail.com

You can find `tblo.zip` in the `tblo` folder of PSXT. However, pressing the `tblo` button will open File Explorer at that folder automatically!

5.2 Liveries

PSXT must know which liveries are available in your Simulator in order to perform accurate matching. By default it will perform a scan when the program is started, unless the **skip scan** checkbox is ticked).

It will scan for your installed AI aircraft (models/liveries), looking for titles, airline codes, aircraft models and registration codes (tail numbers).

PSXT attempts to detect your liveries automatically, however, see the sections below more details.

For those interested, more in-depth information about the livery scan can be found in the PSXT Manual Appendix.

5.2.1 Skip scan

Default: **off**

If checked, the scan for liveries at the start of the program is skipped.

5.2.2 Automatic checkbox + drop-down list (MSFS only)

Default: **On** with selector at **AIG, FSLTL, JF, MTL**

In Automatic mode PSXT will first search for the AI aircraft packages AIG (aig-aitraffic-oci), then FSLTL (fsltl-traffic-base), then FSTraffic (justflight-aircraft-traffic-fleet), and finally the IVAO_MTL aircraft package in your Community folder.

You may change the order and/or limit the number of AI packages with the selector. Just pick your preference from the drop-down list.

After you have selected a new value, you must manually restart PSXT to make the selection active.

After that, it will search for a folder **user-aitraffic** in Community. Here you can place your aircraft from MAIW or FSX or other sources that are not available in the first three packages.

The order (or hierarchy) is important when you are using Prioritize; airline + model liveries in a higher package have priority over the same (airline + model) liveries in a lower package.

P3D: It will search for the addons.cfg and simobjects.cfg files for packages defining AI aircraft.

5.2.3 MSFS: Manual mode

If you want to specify the paths to the folders with your liveries yourself, you should uncheck the Automatic checkbox. Then define a maximum of 10 full paths to folders with AI aircraft. The parameters in the parameters.xml file (see 3.5) are lfolder0 ... lfolder9.

The order (or hierarchy) is important when you are using Prioritize; airline + model liveries in a higher package have priority over the same (airline + model) liveries in a lower package.

Examples:

```
<parameter name="lfolder0" value="D:\aig-aitraffic-oci" />
<parameter name="lfolder1" value="D:\fsltl-traffic-base" />
<parameter name="lfolder2" value="D:\justflight-aircraft-traffic-fleet" />
```

5.2.4 P3D: Manual mode (by default)

Define a maximum of 10 full paths to folders with your AI aircraft. The parameters in the parameters.xml file (see 3.5) are lfolder0 ... lfolder9.

The order (or hierarchy) is important when you are using Prioritize; airline + model liveries in a higher package have priority over the same (airline + model) liveries in a lower package.

Examples:

```
<parameter name="lfolder0" value="D:\AI\FSPXAI" />
<parameter name="lfolder1" value="D:\AI\FAIB Boeing" />
<parameter name="lfolder2" value="D:\AI\FAIB Airbus" />
```

5.2.5 Priority

Default: **On**

If **priority** is checked, 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 Ifolder0 have priority over liveries in the AI package indicated by Ifolder1. (*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 from which PSXT performs the matching with a real live aircraft.

6 Collision avoidance

You can activate near-collision avoidance functions, or you can block one of these aircraft and take its place (mirroring).

At busy airports, it can sometimes be difficult to find a place in the departure queue. At London Heathrow, for example, there may be as many as 13 aircraft waiting for take-off in two parallel queues, each using different runway entry points. Occasionally, an aircraft may even land on that same runway...

The following features may make your life as a pilot easier...

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

Default: **true**

PSXT detects a near collision if two aircraft are closer to each other 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.

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

Default: **true**

PSXT detects near collisions in the air only below 7000 ft. Above that you have to act on TCAS warnings yourself.

Live aircraft are **never removed if they are more than 0.5 nm behind the user.**

Removal is only considered for aircraft **ahead of the user**, and only during vectors or on final. The goal is not ATC conflict resolution, but to prevent unrealistic close merges and nuisance TCAS alerts while preserving live traffic realism.

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

6.3 Type of traffic selector

Default: **All traffic**

Select all traffic, airborne only or ground traffic only.

6.4 Aircraft push button

This button fills the drop-down box below with the current traffic.

For each aircraft, the callsign, registration code, and hex ID are shown. Select an aircraft.

MSFS only:

In MSFS, each aircraft has an object ID in SimConnect. Spotters need the object ID of a specific aircraft in the sim in order to follow it in Spot view.

This is how it works:

Look up the callsign in Little Navmap. Using that callsign, select the aircraft in the drop-down list and note its object ID.

In MSFS Dev Mode, go to:

Debug → SimObjects → Containers → Airplane → Containers, and click View.

Object IDs in MSFS 2020 are shown in decimal format, while in MSFS 2024 they are shown in hexadecimal format.

6.5 Block push button

This removes the aircraft from the sim and prevents it from reappearing.

6.6 Unblock push button

By pressing this button, you unblock the aircraft, and it will reappear in the simulator after approximately 30 seconds.

7 Traffic injection

The following parameters all control how much traffic and what type of traffic is injected in your Sim.

By default PSXT spawns **heavy** aircraft, **large** aircraft and **business jets**.
*Ground vehicles, gliders, airships, and balloons are **not** spawned!*

In P3D and MSFS2020 (*unfortunately not in MSFS2024*), PSXT adds a basic flight plan — with origin and destination — to each live aircraft in Simconnect (= your simulator).

In P3D, you can enable Flight Plan under the Information Options for vehicle labels to display the route (from → to) above each live aircraft.

External programmes such as Little Navmap and EFB also benefit from this, as they can display the flight plan information.

Note that this simple flight plan is purely informational — it does not affect aircraft behaviour or trigger automated flights. PSXT remains fully in control of each aircraft.

7.1 Radius

Default: **40 nm**, range 2 ... 100 nm (in steps of 1).

All airborne live aircraft within a lateral ("horizontal") radius of the user aircraft will be injected.
 All live ground traffic at airports where you land or park within the lateral range will also be injected.

If you are using FSUIPC, ensure that the Limit TCAS range setting on the Misc tab matches the value set in PSXT. Do not use the FSUIPC traffic limiter or any other traffic limiters, as they will interfere with PSXT's internal traffic management.

7.2 Auto radius

Default: on

This checkbox is useful for pilots flying from departure to destination in cockpit view. When enabled, PSXT dynamically adjusts the Radius parameter according to the phase of your flight. This ensures that only aircraft relevant to you as a pilot are injected into the simulator and shown on your TCAS, resulting in a smaller impact on FPS.

The function works as follows. It is based on the user aircraft's altitude:

On ground: RANGE_GROUND
 Below FLIGHT_LEVEL_1: RANGE_1
 Below FLIGHT_LEVEL_2: RANGE_2
 Below FLIGHT_LEVEL_3: RANGE_3
 Above FLIGHT_LEVEL_3: RANGE_ABOVE

Requirements: FLIGHT_LEVEL_3 > FLIGHT_LEVEL_2 > FLIGHT_LEVEL_1 and
 RANGE_ABOVE >= RANGE_3 >= RANGE_2 >= RANGE_1 >= RANGE_GROUND.

The default values (hidden parameters):
 RANGE_GROUND=10 nm, RANGE_1 = 20 nm, RANGE_2 = 40 nm, RANGE_3 = 50 nm and
 RANGE_ABOVE=60 nm, FLIGHT_LEVEL_1=100, FLIGHT_LEVEL_2=200, FLIGHT_LEVEL_3=300,

Changes are smoothed: PSXT uses linear interpolation between flight levels.

7.3 Max Live

Default: 500, range 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 75, 100, 500

Leave it at 500 if you are not using it.

It is best to set Radius first and then, if needed, Max Live.

With this parameter the maximum total number of **live** aircraft (**parked and not parked**) can be limited. It may accidentally overshoot, but PSXT will correct that later.

This value is stored in your learned airport file. The value in the learned file overrides the value of Max Live in the UI while the airport is loaded. This way you can set different values for different airports. When you leave the airport, the default global setting is used again

7.4 Taxiing

There are sometimes gaps in the traffic data received from RealTraffic (RT). PSXT attempts to bridge these gaps by temporarily controlling the aircraft itself.

A special case is a taxiing aircraft. If no new data is received, PSXT will hold the aircraft at its last known position. If another live aircraft is about to collide with it, PSXT will remove the holding aircraft to prevent a conflict.

An aircraft can remain in this holding state for a maximum of 3 minutes (default) *).

If new data becomes available for a holding aircraft, PSXT will allow it to continue provided the new reported position is less than 250 meters (default) *) from the held position. Otherwise, the aircraft will be removed.

*) With the two drop-down lists you can customize this algorithm.

7.5 Max matching depth

Default **4**, range 1 ... 7.

This parameter determines the maximum number of matching steps PSXT will perform when searching for a livery for an aircraft (both live and static). The lower the value, the better the match quality — but you may see fewer aircraft. **1** is the strictest (lowest) setting.

Liveries are matched using a best-effort process of up to eight steps:

Step	Match	Indicated in the Log with
1	Match of an aircraft by registration code. If the match is unique—that is, the registration code corresponds to exactly one livery—you will see a 0 in the Log. Otherwise, a 1 will be shown.	=0/1=
2	PVT aircraft only: match of a generic livery with airline ZZZ and model [a generic livery: icao_airline (MSFS) or atc_parking_codes (P3D)= ZZZ] <i>This will result in a so-called white or house paint</i> or a match of an aircraft by airline and/or operator* codes and model	=4= =2=
3	Match an aircraft by airline and/or operator* and a similar model of the wanted model	=3=
4	Match of a generic livery with airline ZZZ and model. <i>This will result in a so-called white or house paint</i>	=4=
5	Match an aircraft by model (whatever airline)	=5=
6	Match an aircraft by a similar model of the wanted model (whatever airline)	=6=
7	Match an aircraft by a model of the same category (whatever airline)	=7=
8	Match an aircraft by first a B738 or second an A320 livery ... Note that this step is meant for TCAS coverage use only, and will probably never happen, unless you have no bizjets.	=8=

For light aircraft, business jets and helicopters PSXT will also try steps 5, 6 and 7, if matching is set at 4.

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

Flight history for aircraft - N310SY	
AIRCRAFT	TYPE CODE
Embraer E175LR	E75L
AIRLINE	Code
Delta Connection	DL / DAL
OPERATOR	Code
SkyWest Airlines	OO / SKW

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

Special case: =c= means "matched" from cache, an aircraft that reappeared after a hiccup in RT

7.6 With extra TCAS coverage above this FL

When the user aircraft is above this FL, aircraft matching is forced to step 8, for aircraft that are flying above that flight level too.

This ensures that all nearby aircraft are shown on the TCAS (and in Little Navmap), regardless of aircraft type or airline.

Visual matching is irrelevant at these distances.

7.7 Include four extra categories of aircraft

Default: **On** **)

If a checkbox is ticked, PSXT spawn also **twin-engine propellor** (light) aircraft, **light** aircraft (GA), **helicopters** and **high-performance** fighter jets (F16, F35, and so on).

Note that your helicopter liveries should have Category=airplane in their aircraft.cfg file, and not Helicopter.

If you do not have models from one of these categories, the checkbox becomes disabled.

If you switch a value from on to off, any already available aircraft of that type will be removed from the Sim, and new ones will no longer appear.

8 Parking

If an aircraft is on the ground within 5 nm from a known airport, an airport file for that location — containing information about which airline parks where, when, and with which aircraft type — is downloaded from my archive hosted on my website.

An airport file is also loaded when flying, if the conditions set for range and height above ground are met, and the airport is one of your **park airports** (Chapter 9). This allows it to support parking and traffic at airports you fly over.

PSXT uses the airport file for parking live and static aircraft.

Note that the downloaded file is available for inspection at airports\park_airport.xml.

When you fly away from an airport, the static parked aircraft will remain visible until you are out of range — or earlier, if another airport comes within range and is closer.

8.1 Update interval (min)

Default: 20 minutes

Every 20 minutes (default), necessary changes to parking positions are calculated. The words "Airport parking" are shown in yellow if scheduled parking takes place.

8.2 Park button

Pressing the **Park** button immediately starts the calculation of the best parking positions at that time of day. You can use this if you want recent parameter changes to take effect right away.

This function operates in addition to the regular 20 minute schedule. However, pressing the button will reset the schedule, meaning a new 20 minute interval begins from that moment.

8.3 Destination

Default: empty, valid ICAO codes for airports only

If empty, airport parking takes place at the airports in your my_park_airports file.

If not empty, airport parking will take place at the destination airport only, and only when the conditions set by range and height above ground (see next) are met.

Place the cursor in the text box (indicated by light square brackets), type in or edit the value, and when you're ready, press the **Enter** key to activate the new value. Feedback about your input will be shown in the green line at the bottom of the GUI window.

8.4 Reserved stand

Default: empty, values will be set when you select a destination

Select one of the stands of your airport. See also the section about park airports (8.12).

8.5 (Within) range nm

Default **20**, range 5, 10, 15, 20, 25 and 30 nm

Parking at an airport starts if the user aircraft is within the range of the airport and below the height (see 7.9), and if the airport is in your park airports (see 7.16).

8.6 Below height (above ground)

Default: 18000, range 2000, 4000, 6000, 8000, 10000, 12000, 14000, 16000, 18000, 36000 and 48000

Parking at an airport starts if the user aircraft is within range of the airport (see 7.8) and below the height above ground, and if the airport is in your park airports (see 7.16).

8.7 RT parked only

Default: **off **)**

If checked, PSXT will use only the parked traffic info it receives from RealTraffic.

This way you may get fewer aircraft, but all of them are in the right place for that time of day. This, in combination with live parked traffic, may be a good option for some airports.

You will get all aircraft that are still parked at the airport up to 24 hours. You may probably see many from within, say, 6 hours and only a few from 20 or so hours ago (staying overnight, or a chartered aircraft waiting to return, for instance).

8.8 Max fit

Default: **24.0**, range **0.5 .. 24.0 hours **)**

If 0.5, only real, and soft (depending on Park Soft in 8.12), 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 > 0.5 it will search back for more time.

The Max fit value also determines how many RT parked aircraft are taken into account; value 24 implies all, value 0.5 implies current half-hour only, and so on.

8.9 Max park

Default: 500, range 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 100, 500

For every airport visited, this value will be the maximum number of parked (live + static) aircraft. This value is not stored in the learned airport file. Leave it at 500 if you are not using it.

8.10 Auto park

Default: **on **)**

If checked, PSXT will search for the best possible parameter settings of Max fit, Week and Soft (see further down) to deliver the target percentage of static parked.

8.11 Target parking percentage

Default: **30**, range 0 .. 100 % ****)**

This parameter sets the desired maximum percentage of parking positions at the airport that may be occupied by live or static parked aircraft. The value is expressed as a percentage of the total number of parking positions available.

- If the maximum has been reached and a new live aircraft needs to park, the oldest static aircraft will be removed first. If no static aircraft are available, the oldest live aircraft will be removed.
- Note: If you set this value to 0, only live aircraft will park (with no maximum limit).

Live aircraft that have parked but no longer receive RealTraffic data will remain at their stand for up to 3 hours (for heavies) or 90 minutes (for others), unless they start their return flight earlier.

PSXT automatically searches for the best allocation of static aircraft, based on parking options in the airport file that include day, hour, and real-world data. The Log.txt will indicate whether sufficient options were available for the selected hour.

The search is performed for the actual half-hour of the day. If not enough options are found, PSXT will step back in half-hour increments, up to 24 hours, until enough options are found. This process may cross into the previous day if needed.

Note: Static parked aircraft can be identified by a callsign ending with an asterisk (*)!

8.12 Week

Default: **off **)**

If checked: The day is ignored when selecting static aircraft options from the airport file. Only the maximum fit is checked. This can be useful at airports with a very low Qf, as it allows more static aircraft to be parked (though not according to real-world operations)

8.13 Soft

Default: **off **)**

If checked: Soft options from the airport file are used for static parking. These are less reliable options, meaning the aircraft are known to have visited the airport at that time of the week, but their exact parking stands are not recorded and therefore assigned randomly.

Flush button

If pressed, PSXT will remove all **static** parked aircraft and set ILO (8.12) to true. This will leave you with live (parked) aircraft only.

8.14 Flush button

Removes all static parked aircraft from the Sim.

9 Park airports

Park airports are airports where PSXT can park live and static aircraft. Parking at an airport takes place:

1. if you have entered that airport as your destination in the UI, or
2. when you start at an airport, or
3. in flight for airports you have defined to be **park airports**, as soon as you are within reach of the airport, and if it is the one closest by if there are more airports within that reach, or
4. if you land at an airport, not being one of your park airports, when you have come to a full stop.

@3 To begin with, all of your third-party add-on airports are taken as your park airports (in MSFS from 2020 and 2024).

However, in a file named `my_park_airports.txt`, placed in the folder `\PSXT\airports`, you may specify more airports, and (optionally) you may define a reserved stand for your aircraft at the airport (thereby excluding it from static and live parking)

Specify one airport per line. If you want to reserve a stand, you must add it after the airport ICAO code, separated by a comma. Enter the name of the stand as defined in the airport file; look for the `id` field as in `<parkpos id="GATE A12"`.

The airports are checked for validity when reading the `my_park_airports.txt` file at start-up.

An example `my_park_airports.txt`:

```
EFHK
EHAM,GATE B24
ESSA,PARKING 2
...
```

You can also let PSXT do that work for you, because:

- when you select Destination and Stand in the UI.
- when you start at an airport that is not one of your park airports, it will be **automatically** added to your `my_park_airports.txt`, and if you are parked at a stand, the stand will be added too.
- when you land at an airport that is not one of your park airports, the airport will be added.
- when you start at an airport that is one of your park airports, parked at a stand, your stand will be updated if applicable; when you exit your Sim or load a new flight while you are at a stand, the stand will be saved in your park airports.

In MSFS `my_park_airports.txt` is used in both modes (MSFS2020 and MSFS2024).

****)** *The value is stored in your learned airport file. The value in the learned file overrides (but does not change) its value in the `parameters.xml` file when the airport file is loaded. This way you can set different values for different airports.*

10 Airport files

Airport files are .xml files containing detailed information about an airport. A key part of this data concerns the parking positions (stands), including which aircraft park where, with which aircraft types, and on which day of the week.

Below is a shortened example for the airport EHRD:

```
<?xml version="1.0"?>
<root>
  <meta>
    <info name="author" value="PSXT" />
    <info name="version" value="14.5.0" />
    <info name="date" value="Feb 23, 2026" />
    <info name="copyright" value="(c) 2026 Nico W. Kaan, Netherlands, All Rights reserved" />
    <info name="sim" value="2" />
  </meta>
  <parameters>
    <parameter name="icao" value="EHRD" />
    <parameter name="airport_name" value="Rotterdam" />
    <parameter name="publisher" value="RDPresets" />
    <parameter name="version" value="1.0.0" />
    <parameter name="altitude" value="-16" />
    <parameter name="num_parkpos" value="73" />
    <parameter name="Qf_real" value="25" />
    <parameter name="Qf_soft" value="43" />
    <parameter name="RT_traffic_week" value="8" />
  </parameters>
  <parkpositions>
    <parkpos id="GATE A1" latitude="51.951387" longitude="4.432929" heading="327.0"
radius="18.0" >
      <opt reg="TC-RBD" day="2" hh="23:26" cs="PGT718P:PGT468H" />
      <opt reg="OO-GLM" soft="true" day="3" hh="25" cs="OOGLM" />
    </parkpos>
    <parkpos id="GATE A2" latitude="51.951622" longitude="4.433494" heading="327.0"
radius="18.0" >
      <opt reg="EI-IHJ" soft="true" day="3" hh="26" cs="PHDAG" />
    </parkpos>
    <parkpos id="GATE B1" latitude="51.950638" longitude="4.433715" heading="327.0"
radius="18.0" >
      <opt reg="G-LCYJ" day="1" hh="33:35" cs="CFE692:CFE35K" />
      <opt reg="G-LCAB" day="7" hh="32:33" cs="CFE692:CFE35K" />
      <opt reg="TC-OHG" day="7" hh="23:25" cs="TKJ301:TKJ302" />
    </parkpos>
    <parkpos id="GATE B3" latitude="51.951091" longitude="4.434862" heading="327.1"
radius="18.0" >
      <opt reg="PH-CDR" day="1" hh="16" cs="CND537P" />
      <opt reg="PH-HBJ" day="1" hh="29:30" cs="TRA087:TRA47Q" />
    </parkpos>
    ...
  </parkpositions>
</root>
```

This file is stored in the archive on my web server. It is automatically downloaded and loaded into PSXT (./airports/park_airport.xml) as soon as you reach EHRD, or when you start your flight at EHRD. PSXT then uses it to generate static parked aircraft that match the day of the week, the half-hour, and other relevant parameters.

PSXT comes with thousands of airport files on its web server, each already populated with ground traffic data from the most recent week. However, new versions of airports may be released, or entirely new airports may become available.

Such airports are detected during a **Scan for airports** (5.1). In that case, PSXT generates a new airport file containing all stand information, but without traffic data. You will then be asked to send

this file to me so I can generate traffic for it (this is called offline learning: TBLO) and add it to the archive (see the TBLO button 5.1.1).

PSXT will always learn **new options** for the airport **on-the-fly** (during your session), on top of what is already in the archive. The learned options will be saved in a file <ICAO>.xml in the local folders

\PSXT\airports\learned\stock, or
 \PSXT\airports\learned\simaddon, or
 \PSXT\airports\learned\thirdparty\<publisher>

depending on the publisher, and is not overwritten by each PSXT version update.

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

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 aircraft type (model) codes must comply with ICAO standards.

10.1 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 does not exist in the file in the archive.
Specify it in the same way as in the airport files, plus the attribute manual="true"
- c) You may create a complete airport file as an alternative to one in the archive.
Specify the parameters airport_name and publisher and add the parameter handmade="true".
There is no need to add the attribute manual to parking positions, as in b).

PSXT 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.

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

10.2 Editing park_airport.xml

Sometimes parking positions have incorrect parameters, such as the wrong type, an incorrect radius, or even inaccurate latitude and longitude coordinates. This can have a significant impact on parking behaviour. For example, PSXT will not allow a heavy aircraft to park at a stand that is classified as small.

If you do not leave the airport between two PSXT runs, PSXT will keep park_airport.xml as input (and does not overwrite it with a new file from the archive). Therefore it is possible to make changes and to tests it.

If your changes resolve the issue, you are encouraged to send the updated **park_airport.xml** file to leveled757@gmail.com so it can replace the version currently stored in the archive.

Here is an example of a parking position in ../airports/park_airport.xml.

```
parkpos id="GATE D52" typestr="GATE_MEDIUM" latitude="52.308937" longitude="4.772412"  
heading="3.0" radius="19.1"
```

The radius is in meters with one decimal.

The heading is in degrees with one decimal

The typestr values are:

- NONE
- RAMP_GA
- RAMP_GA_SMALL
- RAMP_GA_MEDIUM
- RAMP_GA_LARGE
- RAMP_CARGO
- RAMP_MIL_CARGO
- RAMP_MIL_COMBAT
- GATE_SMALL
- GATE_MEDIUM
- GATE_HEAVY
- DOCK_GA
- FUEL
- VEHICLE
- RAMP_GA_EXTRA
- GATE_EXTRA

11 Miscellaneous parameters

11.1 Random

Default: **true**

If set to false, there will be no randomness in livery matching and no randomness in the *stands* chosen for static aircraft.

You should leave this value at true. It is needed for users running **PSXT** on more than one computer at the same time (section 2.3) **only**.

11.2 Flight messages

Default: **true**

If set to true PSXT will display touch down and take off messages in the messages line at the bottom of the UI.

11.3 Aerowinx PSX (RTApp only)

Default: **false**

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

11.4 GUID changes

Default: **true (recommended)**

If set to **false**, live aircraft in the simulator will not be updated (a so-called “livery change”) to reflect new origin/destination parameters, updated callsign information, or reinitialised lighting. Although these updates are positive, they have the side effect of changing the object's **GUID**. As a result, if you are viewing that aircraft in spot mode (e.g. for a video stream), it may suddenly disappear from view — even though it still exists in the simulator at the same location.

Also note that if this setting is **false**, external programmes that retrieve data via SimConnect — such as **LittleNavMap** — will no longer receive up-to-date information.

11.5 Object ID list (MSFS only)

Default: **false**

Every minute, information about all active live aircraft is written to the file `\output\info\ObjectIdList.txt`.

See also 5.5 for a similar, more interactive, way to get the object ID of an aircraft.

Note: don't switch this on if you don't use it, it will save CPU time.

11.6 Nav Lights

Default: **true**

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

11.7 Port number (RTApp only)

Default: **10747**

With the hidden (see 3.5) parameter `PORT_NUMBER` you may set a different port number for the communication with the RealTraffic app (and change the one in RealTraffic accordingly!).

12 Time Shift [RT Direct only]

With these boxes you may enter a time shift up to 4 weeks ago. PSXT will read Traffic and Weather data with that time shift.

Ranges are: days: 0 .. 13, hours: 0 .. 23 and minutes: 0 .. 59. All these values will be set to 0 if you close PSXT; in a forced restart the values remain active.

Activate your selection by clicking the Set button.

Or alternatively, with the **Sim Time** checkbox (default off) the time in your Simulator determines the time in RT, but again, no more than 2 weeks ago is possible.

Whether all Time Shift parameters are kept between sessions or not depends on the setting of the Keep checkbox. At a forced restart the values are always kept.

13 Park airport information

13.1 First line

The ICAO code, airport name and publisher (plus version) from the loaded airport add-on.

(h)

If the file is handmade (see 10.1 section c) (h) will appear after the name of the publisher.

week=n n is the week number of the airport file (from the archive) (1 .. 52)

13.2 Second line

Info about whether static parked aircraft are being generated or how long it will take till the next update.

qf: <real[/soft]>

The quality factor of the airport file (0 .. 100), for both real and soft options. If real ≥ 35 , there are no soft options, so Qf soft is not shown. If there is no traffic data, the Qf is empty.

reserved stand: <>

If specified for this airport (see 7.7), the stand ID will appear here. If it is not a valid ID, it will be in yellow.

updates <n>

Gives the number of real and soft updates found during this session; they will be added to your learned file for that airport.

13.3 Third line

stands: <n>

The total number of parking positions (stands), defined in the airport file.

park live: <n>

The number of live aircraft that are parked at the airport.

Note that parked live aircraft are not included in the live count in the top right RT section of the UI.

live fit: <n.m>.

The average time -in hours- of parked live aircraft at the airport.

static: <n> (m)

The number of static aircraft that are parked at the airport. Plus -when parking is updated- the number of aircraft still to be parked (in yellow).

static fit: <n.m>.

The average time -in hours- of parked static aircraft at the airport. This value, together with max fit from the first line, is an indication of how "current" the parked static aircraft are.

This reading is not available if parking positions were used on days of the week other than the current day.

max park: <n>

The maximum allowed total number of parked live + static aircraft (based on your specified percentage and the number of parking positions).