FlightGear Manual

Michael Basler, Martin Spott,
Stuart Buchanan, Jon Berndt, Bernhard Buckel,
Roman Ludwicki, Cameron Moore, Curt Olson,
Dave Perry, Michael Selig, Darrell Walisser, and others

The FlightGear Manual
December 14, 2023
For FlightGear version 2020.3.19

Editor: Scott Giese

About the cover image:

Cessna 172P parked at Aosta Valley [LIMW] Italy, by Gilberto Agostinho, General Public License

The Cessna 172 Skyhawk is a four-seat, single-engine, high-wing fixed-wing aircraft. First flown in 1955 and still in production, more Cessna 172s have been built than any other aircraft.

Cessna 172P Contributors:

David Megginson (original author),
Gilberto Agostinho (gsagostinho), Wayne Bragg (wlbragg), Juan Vera del Campo (Juanvvc),
Daniel Dubreuil (Dany93), Jonathan Redpath (legoboyvdlp), Jonathan Schellhase (dg-505),
Tuomas Kuosmanen (tigert), Anders Gidenstam (AndersG), Waldo Kitty (wkitty42),
Jarl Arntzen (jarlarntzen), algefaen, Horacio, D-ECHO, onox, thevirtualfer

New in this Edition:

Chapter 4, Launcher instructions, Stuart Buchanan contributor
Chapter 8, New screenshots, Jonathan Redpath contributor
Chapter 8, Turn Coordinator, Scott Giese contributor

Contents

Chapter 1

Preface

FlightGear is an open source flight simulator developed cooperatively over the internet by a group of flight simulation and programming enthusiasts. This manual is meant to give beginners a guide in getting FlightGear up and running, and themselves into the air. It is not intended to provide complete documentation of all the features and add-ons of FlightGear but, instead, aims to give a new user the best start to explore what FlightGear has to offer.

This version of the document was written for FlightGear version 2020.3.19. Users of earlier versions of FlightGear will still find this document useful, but some of the features described may not be present.

This guide is split into three parts and is structured as follows:

Part I: Installation

Chapter 2, Want to have a free flight? Take FlightGear!, introduces FlightGear, and provides insight into the philosophy behind it and describes the system requirements.

Chapter 3, Preflight: Installing FlightGear. Here you will find instructions for installing the program and additional scenery and aircraft.

Part II: Flying with FlightGear

Chapter 4, Takeoff: How to start the program. This chapter includes instructions for using the integrated Launcher program as well as an overview on the numerous command line options available. Usage of the configuration files is also covered.

Chapter 5, In-flight: All about instruments, keystrokes and menus, describes how to operate the program, i.e., how to fly with FlightGear. This includes a (hopefully) complete list of predefined keyboard commands, an overview of the menu entries, detailed descriptions on the instrument panel and Head-Up Display, as well as hints on using the mouse functionality.

Chapter 6, Features, describes some of the special features that FlightGear offers to the advanced user.

Part III: Tutorials

Chapter 7, Tutorials, provides information on the many tutorials available for new pilots.

Chapter 8, A Basic Flight Simulator Tutorial, provides a tutorial on the basics of flying, illustrated with many examples of the appearance of instruments, knobs, and other gadgets within FlightGear.

Chapter 9, A Cross Country Flight Tutorial, describes a simple cross-country flight in the San Francisco area that can be run with a non-default installation.

Chapter 10, An IFR Cross Country Flight Tutorial, describes a similar cross-country flight making use of the instruments to successfully fly in the clouds under Instrument Flight Rules (IFR).

Chapter 11, A Helicopter Tutorial, includes a brief tutorial with the basics for the start-up, lift-off, flying and landing of helicopters in FlightGear.

Appendices

Appendix A, Missed approach: If anything refuses to work, we try to help you work through some common problems you may encounter while using FlightGear.

Appendix B, Landing: further thoughts before leaving the plane. We would like to highlight and recognize the many contributors of FlightGear that have earned our appreciation for their contributions to this amazing flight simulator. We will also sketch an overview of the development of FlightGear and point out what remains to be done.

Appendix C, Main Index. Includes the main index of the present manual.

Appendix D, Index of Command Line Options. Includes the index of the command line options of FlightGear.

Appendix E, List of Acronyms. Includes the list and description of all the acronyms used along the present manual.

Appendices F and G include, respectively, the List of Tables and List of Figures of the manual.

1.1 Condensed Reading

For those who don't want to read this document from cover to cover, we suggest reading the following sections in order to provide an easy way to get into the air:

Installation	Preflight: Installing FlightGear
Starting the simulator	Takeoff: How to start the program
Using the simulator	In-flight: All about instruments, keystrokes and menus

1.2 Instructions For the Truly Impatient

We know most people hate reading manuals. If you are sure the driver for your graphics card supports OpenGL (check documentation; for instance in general NVIDIA graphics cards do) and you are using Windows, macOS or Linux, you can probably skip at least Part I of this

manual and use pre-compiled binaries. These as well as instructions on how to set them up, can be found at:

https://www.flightgear.org/download/.

If you are running Linux, you may find that FlightGear is bundled with your distribution.

Once you have downloaded and installed the binaries, see Chapter 4 for details on starting the simulator.

1.3 Further Reading

While this introductory guide is meant to be self contained, we strongly suggest having a look into further documentation, especially in case of trouble:

- a handy **leaflet** on operation can be found in the base package (which is normally in a `FlightGear` or `fgdata` folder, see `FG_ROOT`) under `Docs/FGShortRef.pdf`;
- **additional documentation** on particular features and functions is available within the base package under the `Docs` directory (i.e., this is the `$FG_ROOT/Docs` folder);
- the official FlightGear **wiki** is available at https://wiki.flightgear.org;
- the official FlightGear **forum** can be found at https://forum.flightgear.org.

Part I

Installation

Chapter 2

Want to have a free flight? Take FlightGear!

2.1 Yet Another Flight Simulator?

Did you ever want to fly a plane yourself, but lacked the money or ability to do so? Are you a certificated pilot looking to improve your skills without having to incur the high expense associated with aviation? Do you want to try some dangerous maneuvers without the stress and inherent risks? Or do you just want to have fun with a more serious game without any violence? If any of these questions apply to you, flight simulators are just for you.

You may already have some experience using Microsoft's Flight Simulator or any other of the commercially available flight simulators targeted at the home consumer. As the price tag of those is usually within the $50 range, buying one of them should not be a serious problem given that running any serious flight simulator requires computer hardware within the $1500 range.

With so many commercially available flight simulators, why would we spend thousands of hours of programming and design work to build a free flight simulator? Well, there are many reasons, but here are the major ones:

- All of the commercial simulators have a serious drawback: they are made by a small group of developers defining features according to what is important to their goals and providing limited interfaces to end users. Anyone who has ever tried to contact a commercial developer would agree that getting your voice heard in that environment is a major challenge. In contrast, FlightGear is designed by the people and for the people with everything out in the open.

- Commercial simulators are usually a compromise of features and usability. Most commercial developers want to be able to serve a broad segment of the population, including serious pilots, beginners, and even casual gamers. In reality the result is always a compromise due to deadlines and funding. As FlightGear is free and open, there is no need for such a compromise. We have no publisher breathing down our necks, and we're all

volunteers that make our own deadlines. We are also at liberty to support markets that no commercial developer would consider viable, like the scientific research community.

- Due to their closed-source nature, access to source code and other resources associated with commercial simulators are tightly protected. This limits the amount of influence external developers, who may possess excellent ideas and skills, can contribute toward the development and direction of these products. With FlightGear, contributors of all skill levels have the potential to make a huge influence on the project. Contributing to a project as large and complex as FlightGear is very rewarding and provides the participant with a great deal of pride in knowing that we are shaping the future of a great simulator.
- Beyond everything else, it's just plain fun! I suppose you could compare us to real pilots that build kit-planes or scratch-builts. Sure, we can go out and buy a pre-built aircraft, but there's just something special about building one yourself.

The points mentioned above form the basis of why we created FlightGear. With those motivations in mind, we have set out to create a high-quality flight simulator that aims to be a civilian, multi-platform, open, user-supported, and user-extensible platform. Let us examine each of these characteristics:

- **Civilian:** The project is primarily aimed at civilian flight simulation. It should be appropriate for simulating general aviation as well as civilian aircraft. Our long-term goal is to have FlightGear FAA-approved as a flight training device. To the disappointment of some users, it is currently not a combat simulator; however, these features are not explicitly excluded. We just have not had a developer that was seriously interested in systems necessary for combat simulation.
- **Multi-platform:** The developers are attempting to keep the source code as platform-independent as possible. This is based on their observation that people interested in flight simulation utilize a wide variety of computer hardware and operating systems. The present code supports the following Operating Systems:
  - Linux (any recent distribution),
  - Windows 11/10/8/7/Vista (Intel/AMD platform),
  - BSD systems,
  - macOS.

  For optimum performance, it is recommended that you use a recent OS with a minimum of 4 GiB of system memory, current graphic drivers, and a GPU containing at least 1 GiB of memory.
- **Open:** The project is not restricted to a static or elite cadre of developers. Anyone who feels they are able to contribute is most welcome. The code (including documentation) is copyrighted under the terms of the GNU General Public License (GPL).

  The GPL is often misunderstood. In simple terms, it states that you can copy and freely distribute the program(s) so licensed. You can modify them if you like and even charge

as much money as you want for the distribution of the modified or original program. However, when distributing the software, you must make it available to the recipients in source code as well and it must retain the original copyright notices. In short:

"You can do anything with the software except make it non-free."

The full text of the GPL can be obtained from the FlightGear source code or from: https://www.gnu.org/copyleft/gpl.html.

- **User-supported and user-extensible:** Unlike most commercial simulators, FlightGear's scenery and aircraft formats, internal variables, APIs, and everything else are user accessible and documented from the beginning. Even without any explicit development documentation (which naturally has to be written at some point), one can always go to the source code to see how something works. It is the goal of the developers to build a basic engine to which scenery designers, panel engineers, maybe adventure or ATC routine writers, sound artists, and others can build upon. It is our hope that the project, including the developers and end users, will benefit from the creativity and ideation of the hundreds of talented "simmers" around the world.

Without a doubt, the success of the Linux project, initiated by Linus Torvalds, inspired several of the developers. Not only has Linux shown that distributed development of highly sophisticated software projects over the Internet is possible, it has also proven that such an effort can surpass the level of quality of competing commercial products.

Figure 2.1: Bad approach to PHNL (Daniel K. Inouye International, Honolulu)

2.2 System Requirements

In comparison to other recent flight simulators, FlightGear's system requirements are not extravagant. A medium-speed AMD or Intel 64 bit processor should be sufficient to handle Flight-

Gear pretty well, given that you have a proper 3D graphics card.

One important prerequisite for running FlightGear is a graphics card whose driver supports OpenGL. If you don't know what OpenGL is, the Khronos web site[1] says it best:

> "OpenGL® is the most widely adopted 2D and 3D graphics API in the industry…"

FlightGear does not run on a graphics board which only supports Direct3D/DirectX. Contrary to OpenGL, Direct3D is a proprietary interface, being restricted to the Windows operating system.

You may be able to run FlightGear on a computer that features a 3D video card not supporting hardware accelerated OpenGL – and even on systems lacking any 3D graphics hardware. However, the absence of hardware accelerated OpenGL support can bring even the fastest machine to its knees. The typical indication of lacking hardware acceleration are frame rates below 1 frame per second.

Any modern 3D graphics card featuring OpenGL support will do. Windows video card drivers that support OpenGL can be found by visit the home page of your video card manufacturer. You should note that sometimes OpenGL drivers are provided by the manufacturers of the graphics chip instead of by the makers of the board. If you are going to buy a graphics card for running FlightGear, a NVIDIA GeForce card is recommended, as these tend to have better OpenGL support than AMD/ATI Radeon. 1 GiB of dedicated graphics memory will be more than adequate – many people run FlightGear happily on less.

For the sound effects, any capable sound card should suffice. Due to its flexible design, FlightGear supports a wide range of joysticks and yokes as well as rudder pedals under Linux and Windows. FlightGear can also provide interfaces to full-motion flight chairs.

FlightGear is being developed primarily under Linux, a free UNIX clone (together with lots of GNU utilities) developed cooperatively over the Internet in much the same spirit as FlightGear itself. FlightGear also runs and is partly developed under several flavors of Windows. Building FlightGear is also possible on a macOS and several different UNIX/X11 workstations. Given you have a proper compiler installed, FlightGear can be built under all of these platforms. The primary compiler for all platforms is the free GNU C++ compiler (the Cygnus Cygwin compiler under Win32).

If you want to run FlightGear under macOS, you need to have macOS 10.9 or higher. All recent Apple hardware should run FlightGear, but discrete (as opposed to integrated) graphics will give much better performance (frame-rate). Note that FlightGear is not yet built for Apple Silicon machines, but will run on them using the automatic translation.

2.3 Choosing A Version

It is recommended that you run the latest official release, which is typically produced annually and is available from:

https://www.flightgear.org/download/

[1] https://www.khronos.org/opengl/

If you really want to get the most recent (and, at times, buggiest) code, you can clone the sources at:

https://sourceforge.net/p/flightgear/flightgear/ci/next/tree/

A detailed description of how to set this up for FlightGear can be found at:

https://wiki.flightgear.org/Git

2.4 Flight Dynamics Models

Historically, FlightGear was based on a flight model it inherited (together with the Navion airplane) from LaRCsim. As this had several limitations (most importantly, many characteristics were hard wired in contrast to using configuration files), there were several attempts to develop or include alternative flight models. As a result, FlightGear supports several different flight models, to be chosen from at runtime.

- Possibly the most important one is the JSB flight model developed by Jon Berndt. The JSB flight model is part of a stand-alone project called JSBSim:

  http://jsbsim.sourceforge.net/.

- Andrew Ross created another flight model called YASim for *Yet Another Simulator*. YASim takes a fundamentally different approach to many other FDMs (Flight Dynamics Models) by being based on geometry information rather than aerodynamic coefficients. YASim has a particularly advanced helicopter FDM.

- Christian Mayer developed a flight model of a hot air balloon. Curt Olson subsequently integrated a special "UFO" (Unidentified Flying Object) slew mode, which helps you to quickly fly from point A to point B.

- Finally, there is the UIUC flight model, developed by a team at the University of Illinois at Urbana-Champaign. This work was initially geared toward modeling aircraft in icing conditions, but now encompasses "nonlinear" aerodynamics, which result in more realism in extreme attitudes, such as stall and high angle of attack flight. Two good examples that illustrate this capability are the Airwave Xtreme 150 hang glider and the 1903 Wright Flyer. More details of the UIUC flight model can be found at:

  https://m-selig.ae.illinois.edu/apasim/Aircraft-uiuc.html

It is even possible to drive FlightGear's scene display using an external FDM running on a different computer or via named pipe on the local machine – although this might not be a setup recommended to people just getting in touch with FlightGear.

2.5 About This Guide

There is little, if any, material within this guide that is presented here exclusively. You could even say with Montaigne that we "merely gathered here a big bunch of other men's flowers,

having furnished nothing of my own but the string to hold them together". Most (but fortunately not all) of the information herein can also be obtained from the FlightGear web site located at:

https://www.flightgear.org

The FlightGear Manual is intended to be a first step towards a complete FlightGear documentation. The target audience is the end-user who is not interested in the internal workings of OpenGL or in building his or her own scenery. It is our hope that someday there will be an accompanying *FlightGear Programmer's Guide* a *FlightGear Scenery Design Guide*, describing the Scenery tools now packaged as TerraGear; and a *FlightGear Flight School* package.

We kindly ask you to help us refine this document by submitting corrections, improvements, suggestions and translations. All users are invited to contribute descriptions of alternative setups (graphics cards, operating systems etc.). We will be more than happy to include those in future versions of *The FlightGear Manual* (of course not without giving credit to the authors).

Chapter 3

Preflight: Installing FlightGear

To run FlightGear you need to install the binaries. Once you've done this you may install additional scenery and aircraft if you wish.

Pre-compiled binaries for the latest release are available for:

- Microsoft Windows
- macOS
- Linux.

To download them go to the link below and follow the provided instructions:

https://www.flightgear.org/download/

3.1 Installing scenery

Detailed FlightGear scenery is available for the entire world, allowing you to fly everywhere from the Himalaya mountains to rural Kansas. The FlightGear base package contains scenery for Keflavik, Iceland. To fly elsewhere you will need to download additional scenery. Once you have chosen a flight area, you can either download the Scenery files manually or enable automatic download via TerraSync.

3.1.1 Fetch Scenery as you fly (TerraSync)

FlightGear is able to fetch the Scenery as you fly, if you have a permanent Internet connection at your disposal. FlightGear will download scenery to a separate directory from the main installation. One major benefit of TerraSync is that it always fetches the latest and greatest Scenery from the FlightGear World (Custom) Scenery Project and therefore allows you to pick up incremental updates independent of the comprehensive World Scenery releases, which are generally synchronized with FlightGear releases.

You can enable TerraSync either through the GUI launcher or directly within the simulator itself:

- **Through the launcher**: enable option *Download scenery automatically* in tab *Settings*, Section *Downloads*.
- **Within the simulator**: Open menu *File* → *Scenery Download*. Then simply select the *Enable automatic scenery download* checkbox.

You can find more information about TerraSync in the Wiki:

https://wiki.flightgear.org/TerraSync

3.1.2 Manual Scenery Install

Each piece of scenery is packaged into a compressed archive, or tarball, in a 10 degree by 10 degree chunk. Each tarball is named after the 10x10 degree chunk it represents, for example w130n50.tgz.

You can download scenery outside of the simulator from a clickable map on the Internet or via the TerraMaster tool:

https://www.flightgear.org/download/scenery
https://wiki.flightgear.org/TerraMaster

Alternatively, you can support the FlightGear project by purchasing a complete set of world scenery from here:

https://store.flightgear.org

Once you have downloaded the tarball onto your computer, you need to find the `Scenery` directory of your FlightGear installation.

- For Windows, this directory is likely to be:

  `C:\Program Files\FlightGear\data\Scenery`.

- For Unices, it is usually:

  `/usr/local/share/FlightGear/data/Scenery`.

- For macOS, it is usually:

  `/Applications/FlightGear.app/Contents/Resources/data/Scenery`.

To install the scenery, uncompress the tarball into the `Scenery` directory. Most operating system provide tools to uncompress tarballs. If you cannot uncompress the tarball, install an extractor program such as 7-zip (https://www.7-zip.org).

Note that you should not decompress the numbered scenery files inside the tarball like 958402.gz – this will be done by FlightGear on the fly.

Once you have uncompressed the tarball, the `Terrain` and `Objects` directories will contain additional sub-directories with your new scenery inside.

To use the new scenery, simply select a starting airport within the new scenery. If you are using the FlightGear Launcher, you will need to press the **Refresh** button before you select your airport.

MS Windows Vista/7

If you are using Windows Vista or Windows 7, you may find that Windows installs downloaded scenery (and aircraft) to your *Virtual Store*:

`C:\Users\`*`Your Name`*`\AppData\Local\VirtualStore\Program Files\FlightGear\Scenery`

If it does this, you need to copy the `Terrain` and `Objects` directories manually to your real FlightGear `Scenery` directory as described above.

macOS

You may install the downloaded scenery data and aircraft using the GUI launcher. Pressing **Install add-on scenery** on the *Add-ons* tab opens up the file browser window. Selecting one or more scenery data files will install the scenery data into:

`/Users/(Your Name)/Library/Application Support/FlightGear/Scenery`

Acceptable formats for the scenery data are one of zip, tar.gz, tgz, tar, and extracted folder. If the installation via the GUI launcher fails for some reason, you still have an alternative way to install the data.

In the *Add-ons* tab, click the **Add** button for the *Additional scenery folders* section and select the folder where you store the downloaded and unpacked scenery. Likewise for adding aircrafts. Click the **Add** button for the *Additional aircraft folders* section and select the folder where you store downloaded and unpacked aircrafts.

FG_SCENERY

If you would prefer to keep your downloaded scenery separate from the core installation, you can do so by setting your FG_SCENERY environment variable.

This is where FlightGear looks for Scenery files. It consists of a list of directories that will be searched in order. The directories are separated by ":" on Unix (including macOS) and ";" on Windows.

For example, on Linux a FG_SCENERY environment variable set to:

```
/home/jsmith/WorldScenery:
      /usr/local/share/Flightgear/data/Scenery
```

first searches for scenery in:

`/home/jsmith/WorldScenery`

followed by:

`/usr/local/share/Flightgear/data/Scenery`.

On Windows, an FG_SCENERY environment variable set to:

`C:\Program Files\FlightGear\data\Scenery;C:\data\WorldScenery`

first searches for scenery in:

`C:\Program Files\FlightGear\data\Scenery`

followed by:

`C:\data\WorldScenery`

Setting up environment variables on different platforms is beyond the scope of this document.

3.1.3 Creating your own Scenery

If you are interested in generating your own Scenery, have a look at TerraGear – the tools that generate the Scenery for FlightGear:

https://wiki.flightgear.org/TerraGear

The actively maintained source tree of the TerraGear toolchain is located alongside the FlightGear project on SourceForge:

https://sourceforge.net/p/flightgear/terragear/ci/next/tree/.

3.2 Installing aircraft

The base FlightGear package contains only a small subset of the aircraft that are available for FlightGear. Developers have created a wide range of aircraft, from WWII fighters like the Spitfire, to passenger planes like the Boeing 747.

You can download aircraft from:

https://www.flightgear.org/download/download-aircraft/

Download the file and uncompress it into the `data/Aircraft` subdirectory of your installation. The aircraft are downloaded as `.zip` files. Once you have uncompressed them, there will be a new sub-directory in your `data/Aircraft` directory containing the aircraft. Next time you run FlightGear, the new aircraft will be available.

On all platforms, you may use the GUI launcher to install aircraft files.

3.3 Installing documentation

Most of the packages named above include the complete FlightGear documentation including a PDF version of *The FlightGear Manual* intended for pretty printing using Adobe Reader, available from https://get.adobe.com/reader/

Moreover, when installed, the HTML version can be accessed via FlightGear's *Help* menu entry.

Besides, the source code contains a directory `docs-mini` containing numerous ideas on and solutions to special problems. This is also a good place for further reading.

Part II

Flying with FlightGear

Chapter 4

Takeoff: How to start the program

4.1 Launching the simulator

Figure 4.1: Ready for takeoff – Startup position at Honolulu Intl., PHNL

FlightGear comes complete with an integrated launcher to start FlightGear. On Windows, double-click on the *FlightGear Launcher* Start Menu item or the icon on the Desktop. Alternatively, on any system, you can run

```
fgfs --launcher
```

from the command line. The launcher allows you to select your aircraft, the starting location (you can now even start 10 miles from the runway, in a nice place for an approach, or over a specific navaid!), time of day, enable or disable TerraSync or live weather, and lots of other settings.

When you first open the launcher, you will see a dialog box to set your FG_ROOT variable,

normally:

```
C:\Program Files\FlightGear\data or
C:\Program Files\FlightGear 2020.3.19\data
```

When you have set that, you will see the screen like in figure 4.2.

Figure 4.2: Summary of the launcher settings – Click on **Fly!** to start

The launcher defaults to starting a flight with the Cessna 172P at the default airport, which depends on the FlightGear release. Simply press the **Fly!** button to start the simulator. Alternatively, if you want to change any of your starting settings, select from the buttons on the left.

You can change your aircraft by navigating to the *Aircraft* tab on the left of the window. FlightGear comes pre-installed with the Cessna 172P, and a UFO. To fly other aircraft, the easiest way is to add a default FlightGear hangar with hundreds of aircraft. To do this, in the *Aircraft* tab, click the **Browse** button and then the **Add default hangar**. The list will fill up with lots of aircraft. You may download any of the other aircraft listed simply by clicking on the **Install** button. You can also download aircraft from the official website and private hangars.

Selecting *Location* allows you to select your starting position – tied down at a parking location, on the runway ready for take-off, on approach on an ILS approach, or relative to a VOR, FIX. By default, the current selected location is displayed. To select a completely different location, click the **Back** button and enter the name of where within the world you want to be.

FlightGear will automatically download any required scenery (assuming you have it selected in the *Settings* page).

The *Environment* page allows you to select the time of day, season and weather modeling. You can choose to use the current real world weather conditions, or select a specific weather scenario such as a high pressure region, or a thunderstorm.

The *Settings* page allows you to select a variety of simulation settings, such as multiplayer,

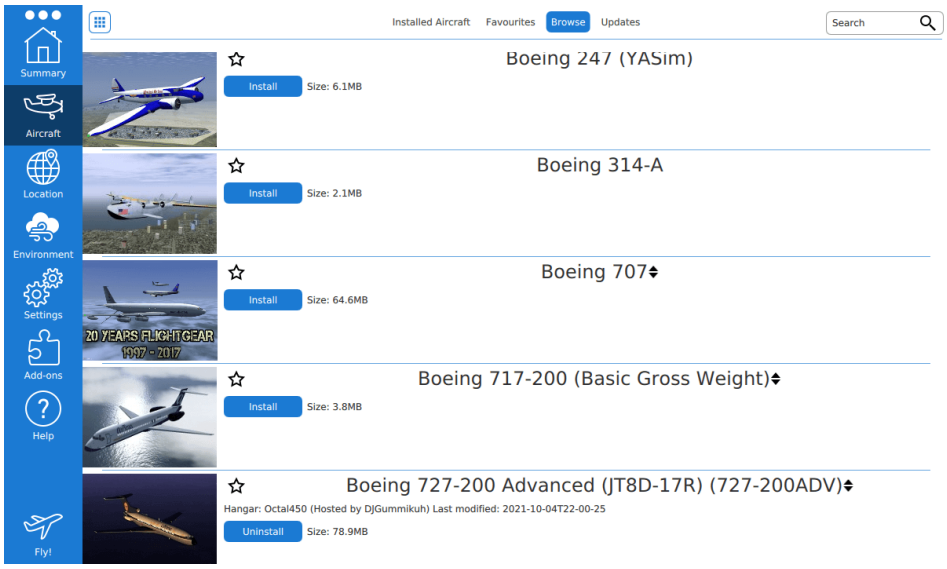

Figure 4.3: Aircraft Selection – Choose from a wide range of aircraft and download them automatically

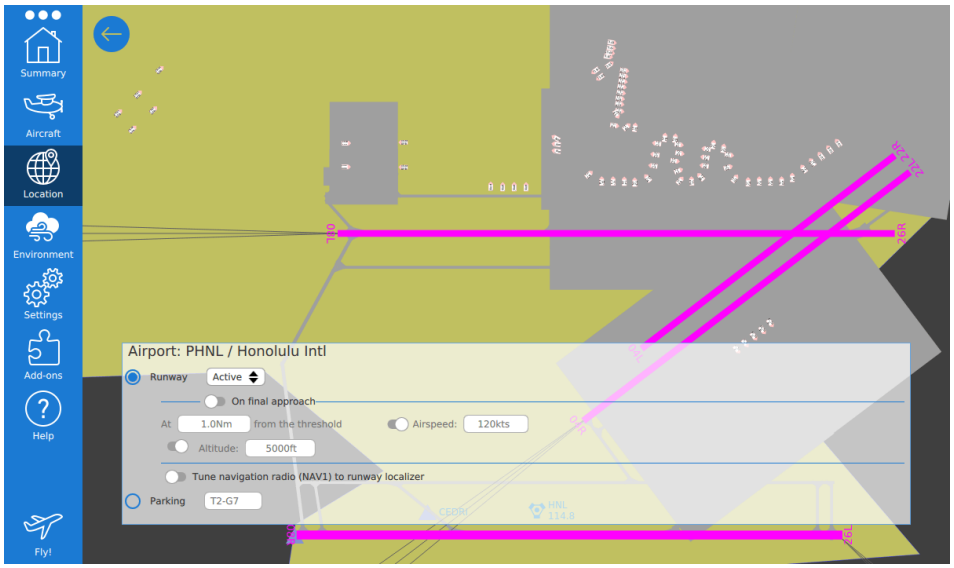

Figure 4.4: Starting Position – Choose a starting position on the ground or in the air

automatic scenery download and graphics options. Advanced options are available from **Show More** on the right.

Finally, the *Add-ons* page allows you to select different hangars of aircraft to download.

Once you are happy with your settings, click on the **Fly!** button to start the simulator.

4.2 Launching from the command line

Section 4.1 showed how to run FlightGear using its built-in launcher. Alternatively, you can run it from the command line. When doing so, you may pass options such as `--fg-root` and `--fg-scenery`, or set the corresponding environment variables named FG_ROOT and FG_SCENERY.

Note that doing such settings is not always necessary (for instance, if you compiled FlightGear yourself and passed `-D FG_DATA_DIR:PATH=`*path* to CMake, then your `fgfs` executable has this directory as its default FG_ROOT setting). However, if you are not using the built-in launcher and either FlightGear can't find its *base package* or you want to add directories for custom scenery, then passing options `--fg-root` or `--fg-scenery`, or alternatively setting FG_ROOT or FG_SCENERY as environment variables, is likely to be helpful.

How to set environment variables largely depends on the operating system and the shell in use, if any. We'll get into such platform-specific details after a few general comments on FG_ROOT and FG_SCENERY.

4.2.1 FG_ROOT

The value of the FG_ROOT setting is the path to a directory (folder) where FlightGear looks for its base data files (for instance, XML and Nasal files shared by several aircraft, navigational beacon locations, radio frequencies for contacting airports, shaders for graphical effects, translation files, etc.).

The contents of this directory comes directly from the FGData repository.[1] If you have installed FlightGear from a binary distribution on Windows or macOS, the directory name should be `data`. It can also be called `fgdata` (typically, after cloning the FGData repository), but other names are possible: for instance, the `flightgear-data-base` package in Debian ships it as `/usr/share/games/flightgear`. This directory contains files such as `version`, `defaults.xml` and `keyboard.xml` as well as subdirectories such as `Aircraft`, `Input` and `Nasal`.

In order to run, FlightGear absolutely needs to know where to find this directory. Users of the built-in launcher are prompted to select it from the graphical interface if FlightGear hasn't found it automatically. Unless the location recorded at compile-time points to a suitable directory, FlightGear command line users need to either set the FG_ROOT environment variable or pass the `--fg-root` option in order to specify an appropriate location (check that it has the aforementioned files and subdirectories).

If you are a user of the `download_and_compile.sh` script, the base data files are in `install/flightgear/fgdata` (relatively to the directory from which you ran the script),

[1] https://sourceforge.net/p/flightgear/fgdata/ci/next/tree/

but you don't need to use `--fg-root` nor to set FG_ROOT: the `run_fgfs.sh` script created by `download_and_compile.sh` automatically passes the `--fg-root` option to `fgfs` for you with the appropriate value.

In most Unix shell languages, $VAR can be thought of as "the value of variable VAR". In explanations and documentation related to FlightGear, we often extend this notation a little bit. Typically, $FG_ROOT should be understood as "your FG_ROOT setting, regardless of the way FlightGear obtains it: compile time default, FG_ROOT environment variable or `--fg-root` option."
The same kind of sloppy wording is sometimes applied to other settings such as FG_SCENERY to mean "the list of scenery paths FlightGear has been configured to use", however this can get confusing. Indeed, the list of scenery paths considered by FlightGear is *assembled* from several things, including the values passed to `--fg-scenery` options and that of the FG_SCENERY environment variable, if set… value which is obtained via $FG_SCENERY in typical Unix shells!

4.2.2 FG_SCENERY

The scenery shown by FlightGear can be assembled from several places, and the whole is sometimes referred to as "FG_SCENERY". This is however sloppy wording: FG_SCENERY is actually the name of an environment variable that *may* be used (but there are several other ways) to point FlightGear to some folders containing scenery. FlightGear can use scenery from folders listed in $FG_SCENERY as well as folders given to the `--fg-scenery` option, all in the same run.

The most straightforward way to obtain FlightGear scenery if you have a good Internet connection is TerraSync: with this piece of the FlightGear infrastructure, scenery for the relevant area is downloaded as you fly. TerraSync scenery is automatically updated when you fly again to the same place. FlightGear can also use folders containing scenery that isn't managed by TerraSync; this is how "custom scenery" is used.

For users of the built-in launcher, enabling or disabling TerraSync is done under *Downloads* in the *Settings* tab: from there, you can enable or disable automatic scenery download and choose where the resulting files are stored (they can easily take up several gigabytes). Scenery directories not managed by TerraSync can be set in the *Add-ons* tab under *Additional scenery folders*.

For people who start FlightGear from the command line, TerraSync can be enabled or disabled with `--enable-terrasync` and `--disable-terrasync`. In order to choose where TerraSync stores the resulting files, you can use `--download-dir` or `--terrasync-dir` (if both are set, the latter takes precedence).

If you start FlightGear from the command line with the simple command `fgfs --enable-terrasync`, scenery will be downloaded to the default location:

- `%USERPROFILE%\FlightGear\Downloads\TerraSync` on the Windows operating system (`%USERPROFILE%` may be `C:\Users\`*username*, but this may depend on the Windows version);

- a `TerraSync` folder stored in the same place as where the `fgfsrc` configuration file is looked for on other operating systems (see table 4.1).

This may be a good way to start FlightGear for the first time, if for some reason you don't want to use its built-in launcher. In case you don't have enough space in that location, simply specify another one using the `--download-dir` option.

Scenery directories not managed by TerraSync can be added using the `--fg-scenery` option and the FG_SCENERY environment variable. The value given to `--fg-scenery` or FG_SCENERY must be a list of directories that will be searched in order, separated by ":" on Unix or macOS and by ";" on Windows. For instance, this value could be used on Unix:

```
/home/joebloggs/my-fg-scenery:/home/joebloggs/.fgfs/TerraSync
```

and that one on Windows:

```
D:\MyScenery;C:\Users\joebloggs\FlightGear\Downloads\TerraSync
```

If you want to use both TerraSync-managed scenery and custom scenery, it is generally advised to give the TerraSync directory as the last item, as in the above examples. This way, custom scenery can override TerraSync-managed scenery. Actually, unless it is already mentioned in $FG_SCENERY or via `--fg-scenery`, the TerraSync directory is automatically appended to the list of scenery paths used by FlightGear (and followed by $FG_ROOT/Scenery in case neither FG_SCENERY nor `--fg-scenery` specify anything).

4.2.3 Launching the simulator under Windows

Open a shell (also known as "command prompt") and use the `cd` command to change to the directory containing your `fgfs.exe` executable. For instance:

```
C:
cd \Program Files\FlightGear\bin
```

Then try simply running `fgfs`. If it complains that it can't find the "base data files", you need to either pass the `--fg-root` option or set the FG_ROOT environment variable:

```
fgfs "--fg-root=C:\Program Files\FlightGear\data"
```

(quotes are needed here because of the spaces in the option value) or

```
SET FG_ROOT="C:\Program Files\FlightGear\data"
fgfs
```

In case you have custom scenery, you might also want to set FG_SCENERY (cf. section 4.2.2). The usual syntax for invoking the `fgfs` executable is:

```
fgfs --option1 --option2 . . .
```

where available options are described in section 4.4. Of course, you can create a batch file with a Windows text editor (like Notepad) using the above lines. For maximum performance, it is recommended that you to minimize (iconize) the text output window while running FlightGear.

4.2.4 Launching the simulator under Unix/Linux

There are several cases to distinguish: if you used the `download_and_compile.sh` script to compile FlightGear, simply run the generated `run_fgfs.sh` script as if it were `fgfs`, giving it any of the options mentioned in section 4.4. In this case, don't pass `--fg-root` to `run_fgfs.sh`: this is already handled inside the script.

We now assume that you have an `fgfs` executable that wasn't obtained by running the `download_and_compile.sh` script. If you have compiled FlightGear yourself and passed `-D FG_DATA_DIR:PATH=`*path* to CMake where *path* tells where your FGData resides, then your `fgfs` executable has this directory as its default FG_ROOT setting; therefore, you don't need to pass `--fg-root` nor to set the FG_ROOT environment variable: you can simply run the `fgfs` executable and give it any of the options mentioned in section 4.4.

In case the `fgfs` executable comes from your distribution, things may or may not be as in the previous paragraph. Try simply running `fgfs`. If it complains that it can't find the "base data files", you need to either pass the `--fg-root` option or set the FG_ROOT environment variable. Using the option is done this way:

```
fgfs --fg-root=/usr/share/games/flightgear
```

(replace `/usr/share/games/flightgear` with the directory containing your FGData files; the above value is only where Debian puts them). As with any shell command, use double or single quotes if the path contains spaces. An alternative to using the `--fg-root` option is to set the FG_ROOT environment variable. In Bourne-style shells, this can be done like so:

```
export FG_ROOT=/usr/share/games/flightgear
```

On the other hand, you would use something like this instead in C-style shells:

```
setenv FG_ROOT /usr/share/games/flightgear
```

Then simply run `fgfs`. In case you have custom scenery, you might also want to use the same technique to set FG_SCENERY (cf. section 4.2.2). Once these details have been sorted out, you can start FlightGear and give it any of the options mentioned in section 4.4:

```
fgfs --option1 --option2 . . .
```

4.2.5 Launching the simulator under macOS

You can also launch the simulator from the command line on macOS. In order to do so, you may open `Terminal.app` (located at `/Applications/Utilities`) and type the following commands:

```
cd /Applications/FlightGear.app/Contents/MacOS
./fgfs --option1 --option2 . . .
```

See section 4.4 for information on the available command line options. As long as you use a prebuilt binary package, you don't need to manually specify FG_ROOT.

4.3 `fgfsrc` configuration files

The FlightGear executable is called `fgfs` (`fgfs.exe` on Windows). This program accepts a large number of options which will be listed in section 4.4. One of these options, namely `--launcher`, has already been presented in section 4.1 and starts the FlightGear built-in launcher. When this option is used, one rarely needs to pass other options manually to `fgfs`, because most settings can be performed from the built-in launcher.

It is also possible to start FlightGear without going through its built-in launcher, i.e., without using the `--launcher` option. This is particularly useful when testing things that require one to restart FlightGear many times in the same or similar conditions. There are also external launchers which work this way, calling the `fgfs` executable with all kinds of options, except `--launcher`.

The rest of this section goes into some details and is safe to skip if you are starting with FlightGear, especially if you use its built-in launcher.

When FlightGear (or rather, the `fgfs` executable) is started without the `--launcher` option, it tries to read options from a well-defined set of configuration files before processing the options passed as arguments to `fgfs`. These configuration files are `fgfsrc` in a location described in table 4.1, as well as `.fgfsrc` and `.fgfsrc.`⟨*hostname*⟩ as explained in table 4.2. If you have options that you very often use, you can create one of these files and put the options there, one per line. Then you don't need to explicitly pass them to `fgfs` anymore. Such files can be created with any text editor (Notepad, Emacs, vi, etc.). If you don't know whether to create an `fgfsrc` or `.fgfsrc` file, we suggest the former: its expected location is tidier and its name not starting with a dot can make things easier in some situations.

Operating system	**Where `fgfsrc` is looked for**[a]
Windows	`%APPDATA%\flightgear.org` (typically: `C:\Users\`*`username`*`\AppData\Roaming\flightgear.org` on Vista or later, otherwise `C:\Documents and Settings\`*`username`*`\Application Data\flightgear.org`)
macOS	`~/Library/Application Support/FlightGear`
Unix/Linux	`~/.fgfs`

[a] That is, unless `FG_HOME` is set, in which case `fgfsrc` is looked for in the corresponding location; however, doing so is not advised unless you have a good reason.

Table 4.1: Location of the `fgfsrc` configuration file

Operating system	**Where `.fgfsrc` and `.fgfsrc.`⟨*hostname*⟩ are looked for**
Windows	`%USERPROFILE%` (typically: `C:\Users\`*`username`*)
macOS	`$HOME` (typically: `/Users/`*`username`*)
Unix/Linux	`$HOME` (typically: `/home/`*`username`*)

Table 4.2: Location of the `.fgfsrc` and `.fgfsrc.`⟨*hostname*⟩ configuration files

As the order of `fgfs` options can matter, let's be a bit more explicit about it. If `--launcher` has been passed, configuration files specified with the `--config` option are first processed, followed by other options from the command line. Otherwise, options are processed in the following order:

- those from `fgfsrc`;
- those from `.fgfsrc`;
- those from `.fgfsrc.`⟨*hostname*⟩ if ⟨*hostname*⟩ matches the host name (i.e., the name of the computer running FlightGear);
- those from `--config` options specifying non-XML files (in reverse order of these options, but in top-down order within each of the specified files);
- finally, the remaining options that were specified on the command line.

There can be exceptions, but usually, options that are processed later override those processed earlier.

The `fgfsrc`, `.fgfsrc`, `.fgfsrc.`⟨*hostname*⟩ and non-XML files loaded with `--config` all obey the same syntax rules: they should have one `fgfs` option per line. Lines starting with a `#` character are treated as comments, i.e. they are ignored. Each option found outside a comment line is treated as if it had been passed as an argument to the `fgfs` executable.

4.4 Command line parameters

Following is a complete list of the numerous command line options available in FlightGear (i.e., options that can be passed to the `fgfs` executable). For more convenience, options that you always want to use may be specified in a preferences file that is automatically processed when FlightGear is started (see section 4.3).

When using the built-in launcher, the default preferences files (`fgfsrc` and similar) are *not* read by FlightGear unless explicitly requested; however, applicable options may be entered in the *Additional Settings* box of the launcher's *Settings* tab.

4.4.1 General Options

- `--launcher`

  Start the launcher (described above).

- `--help`

  Display the most relevant command line options.

- `--help --verbose`

  Display all command line options.

- `--version`

  Display the current FlightGear version.

- `--fg-root`=*path*

  Tell FlightGear where to look for its base data files. If this option is not provided and the FG_ROOT environment variable is set, the latter is used. If neither the `--fg-root` option is passed nor the FG_ROOT environment variable set, FlightGear looks for its base data files in a place that was determined at compile time.

  Users of the FlightGear built-in launcher should not worry about passing `--fg-root` or setting the FG_ROOT environment variable: they can select the location of the base data files directly in the launcher, and it will be remembered.

  For more details, see section 4.2.1.

- `--fg-scenery`=*paths*

  List of paths to consider as scenery directories (it can be used to point FlightGear to custom scenery). These paths should be separated by ":" on Linux and macOS, ";" on Windows. If the FG_SCENERY environment variable is set (whose value should be a list of paths using the same syntax), the corresponding directories are added after those from `--fg-scenery` options.

  Let's call ℓ the resulting list. If ℓ is empty, the list of scenery paths considered by FlightGear is comprised of the TerraSync directory followed by `$FG_ROOT/Scenery`. Otherwise, the TerraSync directory is appended to ℓ unless it was already there, and this becomes the list of scenery paths considered by FlightGear.

  Users of the FlightGear built-in launcher should not worry about passing `--fg-scenery` or setting the FG_SCENERY environment variable: they can set the list of scenery paths directly in the launcher, and it will be remembered.

  For more details, see section 4.2.2.

- `--fg-aircraft`=*paths*

  List of paths where FlightGear will look for aircraft, in addition to `$FG_ROOT/Aircraft` and the directories listed in `$FG_AIRCRAFT`.

  The paths should be separated by ":" on Linux and macOS, ";" on Windows. This option may be specified multiple times—its effect is additive.

- `--data`=*path*

  Specify an additional base data directory (FGData), before the `$FG_ROOT` directory.

- `--addon`=*path*

  Specify a path to addon. Multiple instances can be used.

- `--download-dir`=*path*

  Store aircraft and scenery downloaded via the simulator under *path*.
  The TerraSync directory may be specifically set with the `--terrasync-dir` option.

- `--terrasync-dir`=*path*

  Set the path where the downloaded scenery will be stored.

- `--enable-terrasync`, `--disable-terrasync`

  Enable, disable automatic scenery downloads/updates. By default, TerraSync is turned off.

- `--language`=*code*

  Select the language for this session. The supported languages (codes) are: `ca`, `de`, `en`, `en_US`, `es`, `fr`, `it`, `nl`, `pl`, `pt`, `ru`, `tr`, `sk` and `zh_CN`.

- `--restore-defaults`

  Reset all user settings to their defaults.

- `--enable-save-on-exit`, `--disable-save-on-exit`

  Enable or disable saving of user-preferences on exit from the simulator.

- `--ignore-autosave`

  Ignore the user settings saved the previous time FlightGear was run. This option implies `--disable-save-on-exit`.

- `--read-only`

  Make `$FG_HOME` (the location where user-specific FlightGear data is stored) read-only.

- `--allow-nasal-read`=*paths*

  Allow Nasal aircraft scripts to read files from the directories listed in *paths* (separate multiple paths with colons, or semicolons on Microsoft Windows operating systems). By default, for security reasons, Nasal scripts can only read data from certain directories, such as `$FG_ROOT`, `$FG_HOME`, etc.

- `--browser-app`=*path*

  Specify location of your web browser. E.g:
  `--browser-app`=*"C:\Program Files\Internet Explorer\iexplore.exe"*
  (Note the "…" used because of the spaces!).

- `--config`=*path*

  Load additional properties from the given path. E.g.:
  `--config`=*./Aircraft/X15-set.xml*.

- `--no-default-config`

  Do not load any configuration files unless they were explicitly specified with `--config`.

- `--units-feet`

  Use feet as the unit of measurement.

- `--units-meters`

  Use meters as the unit of measurement.

- `--disable-gui`

  Enable headless mode.

4.4.2 Features

- `--enable-freeze`, `--disable-freeze`

  Control whether FlightGear starts paused or not. Defaults to not paused.

- `--enable-auto-coordination`, `--disable-auto-coordination`

  Switches auto-coordination between aileron and rudder on and off. Auto-coordination is recommended for users without rudder pedals or a "twist" joystick. Defaults to off.

- `--composite-viewer`=*{1|0}*

  Enable (1) or disable (0) CompositeViewer (extra view windows).

- `--enable-ai-models`, `--disable-ai-models`

  Enable, disable the internal AI subsystem, which is required for multiplayer gaming, AI traffic and many other animations.
  Disabling the internal AI subsystem is deprecated.

- `--enable-ai-traffic`, `--disable-ai-traffic`

  Enable, disable artificial plane traffic.

- `--ai-scenario`=*name*

  Enable a specific AI scenario (e.g. `--ai-scenario`=*vinson-demo*). May be used multiple times.

- `--load-tape`=*{file|URL}*

  Load recording of earlier FlightGear session. For *file*, if *file* ends with ".fgdata" it is treated as the local path of the recording file; otherwise we form the local path by prepending *file* with the tape directory (by default it is the directory `recordreplay.py.tapes` in the directory where the FlightGear binary is located) and appending ".fgtape".

  For *URL* (starting with "http://" or "https://") we download the remote recording (which must be a Continuous recording) in the background to a url-dependent filename while replaying it; if the url-dependent filename already exists it is assumed to be a truncated download and we only download any remaining data.

- `--restart-launcher`

  Open Launcher automatically when exiting FlightGear. Be careful when using this option from the command line, because if you don't specify `--launcher`, the simulator will restart instead of Launcher. Disabled as default.

4.4.3 Sound

- `--enable-sound`, `--disable-sound`

  Enable or disable sound.

- `--show-sound-devices`

  Show the available sound devices.

- `--sound-device`=*"device name"*

  Specify the sound device to use for audio.

4.4.4 Aircraft

- `--aircraft`=*aircraft*, `--vehicle`=*name*

  Load the specified aircraft, (e.g. `--aircraft`=*c172p*) For available choices check the directory `$FG_ROOT/Aircraft`, and look for files ending in `-set.xml`. When specifying the aircraft, drop the `-set.xml` from the filename.
  Alternatively, use the `--show-aircraft` option described below to list the available aircraft. For information on downloading additional aircraft, see Section 3.2, Installing aircraft.

- `--show-aircraft`

  Print a sorted list of the currently available aircraft types.

- `--min-status`=*{all|alpha|beta|early-production|production}*

  Display only those aircraft with a specified minimum declared status, one of `all`, `alpha`, `beta`, `early-production`, `production`. For use with `--show-aircraft`.

- `--aircraft-dir`=*path*

  Aircraft directory relative to the executable location.
  Defaults to `$FG_ROOT/Aircraft`.

- `--livery`=*name*

  Set the aircraft livery.

- `--state`=*value*

  Set the initial aircraft state to *value*. The states that can be used are aircraft-dependent.

4.4.5 Flight model

- `--fdm`=*{ada|acms|aisim|balloon|jsb|larcsim|magic|network|pipe|ufo|yasim|external|null}*

  Select the core Flight Dynamics Model (FDM). Options are:

  - `ada` – an externally-driven, special purpose flight dynamics model designed by the Aeronautical Development Agency of Bangalore, India.

- `acms` – the "Aircraft Condition Monitoring System" special purpose flight dynamics model intended to reproduce information stored in a flight data recorder.
- `aisim` – special purpose flight dynamics model for AI aircraft.
- `balloon` – special purpose flight dynamics model for hot air balloon simulation.
- `jsb` – the JSBSim flight dynamics model, which takes a data-driven approach to modeling: given the performance data for an aircraft (mass and balance, ground reactions, propulsions, aerodynamics, etc.), it assembles it together to produce the global aircraft dynamics.
- `larcsim` – the LaRCsim flight dynamics model, the original model used in FlightGear up to 2002, developed at NASA.
- `magic` – the "Magic Carpet" special purpose flight model. It is a simple flight model useful for scenery makers to be able to move around in the virtual world without having to worry about the laws of physics. Currently replaced with `ufo`.
- `network` – opens the network protocol (UDP) to retrieve flight dynamics data over the network from an external program. Usage template:
  `--fdm`=*network,host,port-out,port-in,port-cmd*,
  where *host* is the IP address/name of the computer to connect on the output port (*port-out*). Then we have a second port number as the input port (*port-in*) where FlightGear will accept connections. The last port number is the port to which the commands will be sent (*port-cmd*) via the HTTP protocol to the given *host*. Example of use: `--fdm`=*network,localhost,5501,5502,5503*.
- `pipe` – external model of flight dynamics, using the "named pipe" mechanism in UNIX systems. In other words, it is a FIFO queue (first in, first out) from which FlightGear can read data about properties (as a name=value pair), e.g. from a file, which in turn is fed with data by external flight dynamics model. At the same time, the external flight dynamics model can also ask FlightGear for certain properties – so the communication is bidirectional.

  This is an alternative to transferring the flight dynamics model over the network (see the `network` parameter for the `--fdm` option). The network is more nondeterministic (with the UDP protocol), where packets may be lost or arriving in a different order than they were sent, or arrive late, so this does not guarantee that the simulator will receive exactly what the flight dynamics model sends. Pipe does not have these inconveniences, but for pipe both processes (simulator and flight model) must run on the same machine.

  Usage template: `--fdm`=*pipe,name,protocol*, where *protocol* must be `binary` or `property`.
- `ufo` – another flight model for fast and free movement, useful for scenery makers. FlightGear comes with a UFO aircraft by default, which is just using this flight model.
- `yasim` – the YASim flight dynamics model, which given the physical and flying characteristics of an aircraft, attempts to solve for them.

- `external` or `null` – disabling the generation of the flight dynamics model by FlightGear in order to use an external flight dynamics model (the `external` option has the same meaning as `null` and is maintained for backward compatibility purposes).

This option can normally be ignored, as the `--aircraft` option will set the FDM correctly.

Such flight dynamics models as `ada`, `acms`, `aisim`, `balloon` and `magic` are of special purpose that should not be of interest to the common user.

- `--aero`=*aircraft*

  Specifies the aircraft aeronautical model to load. This option can normally be ignored, as the `--aircraft` option will set the aircraft model correctly.

- `--model-hz`=*n*

  Run the Flight Dynamics Model with this rate (iterations per second).

- `--speed`=*n*

  Run the Flight Dynamics Model this much faster than real time.

- `--trim`, `--notrim`

  Trim (or not) when initializing JSBSim. Defaults to trim.

- `--on-ground`, `--in-air`

  Start up at ground level (default), or in the air. If specifying `--in-air` you must also set an initial altitude using `--altitude`, and may also want to set an initial velocity with `--vc`. Note that some aircraft (notably the X15) must be started in mid-air.

- `--enable-fuel-freeze`, `--disable-fuel-freeze`

  Control whether fuel state is constant (e.g. frozen) or consumed normally (default).

4.4.6 Initial Position and Orientation

- `--airport`=*ICAO*

  Start at a specific airport. The airport is specified by its ICAO code,
  e.g. `--airport`=*KJFK* for JFK airport in New York. For US airport without an ICAO code, try prefixing the 3 character code with "K".

- `--parking-id`=*name*, `--parkpos`=*name*

  Start at a specific parking spot on the airport.

- `--runway`=*NNN*

  Start at the threshold of a specific runway (e.g. 28L). If no runway or parking ID is specified, a runway facing into the wind will be chosen for takeoff.

- `--vor`=*id*, `--ndb`=*id*, `--fix`=*id*

  Set the starting position relative to a VOR, NDB or FIX. Useful for practising approaches.

- `--vor-frequency`=*frequency*

  Set the frequency of the VOR. This option requires the `--vor` option to be present.

- `--ndb-frequency`=*frequency*

  Set the frequency of the NDB. This option requires the `--ndb` option to be present.

- `--carrier`=*name*

  Start on an aircraft carrier. See 6.2, Aircraft Carrier for details of carrier operations.

- `--carrier-position`=*{abeam|FLOLS|name}*

  Specify a starting position relative to the carrier where you can use the predefined `abeam` (start on downwind abeam) or `FLOLS` (start on final approach) values, or specify the name of the carrier's parking position. Must be used with `--carrier`. Defaults to a catapult launch position.

- `--lon`=*degrees*, `--lat`=*degrees*

  Start at a particular longitude and latitude, in decimal degrees (south, west negative).

- `--offset-distance`=*nm*, `--offset-azimuth`=*deg*

  Start at a particular distance (nautical miles) and heading from a position set using `--airport`, `--vor`, `--ndb`, `--fix`, `--carrier` or `--lat` and `--lon`.

- `--altitude`=*feet*

  Start at specific altitude. Implies `--in-air`. Altitude is specified in feet unless you also select `--units-meters`, in which case altitude is in meters. You may also want to set an initial velocity with `--vc` to avoid stalling immediately.

- `--heading`=*degrees*, `--roll`=*degrees*, `--pitch`=*degrees*

  Set the initial orientation of the aircraft. All values default to 0 – heading North, in straight and level flight.

- `--uBody`=*X*, `--vBody`=*Y*, `--wBody`=*Z*

  Set the initial speed along the X, Y and Z axes. Speed is in feet per second unless you also select `--units-meters`, in which case the speed is in meters per second.

- `--vNorth`=*N*, `--vEast`=*E*, `--vDown`=*D*

  Set the initial speed along the South-North, West-East and vertical axes. Speed is in feet per second unless you also select `--units-meters`, in which case the speed is in meters per second.

- `--vc`=*knots*, `--mach`=*num*

  Set the initial airspeed in knots or as a Mach number. Useful if setting `--altitude`, unless you want to stall immediately!

- `--glideslope`=*gradi*, `--roc`=*fpm*

  Set the initial glide slope angle in degrees or as a climb rate in feet per minute. May be positive or negative.

4.4.7 Environment Options

- `--metar`=*METAR STRING*

  Use a specific METAR string, e.g.
  `--metar`=*"XXXX 012345Z 00000KT 99SM CLR 19/M01 A2992"*.
  The METAR may be specified in most common formats (US, European).
  Incompatible with `--enable-real-weather-fetch`.

- `--enable-real-weather-fetch`, `--disable-real-weather-fetch`

  Control whether real-time weather information is downloaded and used.

- `--visibility`=*meters*, `--visibility-miles`=*miles*

  Set the visibility in meters or statute miles.

- `--wind`=*dir[:maxdir]@speed[:gust]*

  Specify the direction the wind blows from (dir) and its speed (speed knots), e.g.
  `--wind`=*180@10*. If the wind is not meant to blow from a fixed direction, but rather from a range of directions, specify the range as dir:maxdir, where dir and maxdir are the minimum and maximum angles in degrees. If you want the simulator to model wind gusts as well, set gust to their maximum intensity in knots, e.g. `--wind`=*180:220@10:15* – variable wind from 180 to 220 degrees, blowing at a speed of 10 knots in gusts up to 15 knots.

- `--random-wind`

  Sets random wind direction and strength.

- `--turbulence`=*n*

  Sets turbulence from completely calm (0.0) to severe (1.0).

- `--ceiling`=*FT_ASL[:THICKNESS_FT]*

  Sets an overcase ceiling at a particular height, and with an optional thickness (defaults to 2000 ft).

4.4.8 Rendering Options

- `--graphics-preset`=*{minimal-quality|low-quality|medium-quality|high-quality|ultra-quality}*

  Set graphic options from one of the presets. Allowed values are `minimal-quality`, `low-quality`, `medium-quality`, `high-quality` or `ultra-quality`.

- `--aspect-ratio-multiplier`=*n*

  Set a multiplier for the display aspect ratio. Default is `1.0`.

- `--bpp`=*depth*

  Specify the bits per pixel.

- `--enable-clouds`, `--disable-clouds`

  Enable (default) or disable cloud layers.

- `--enable-clouds3d`, `--disable-clouds3d`

  Enable (default), disable 3D clouds. Very pretty, but depend on your graphics card supporting GLSL Shaders, which some older, or less powerful graphics cards do not.

- `--enable-distance-attenuation`, `--disable-distance-attenuation`

  Enable or disable more realistic runway and approach light attenuation.

- `--enable-fullscreen`, `--disable-fullscreen`

  Enable, disable (default) full screen mode.

- `--enable-horizon-effect`, `--disable-horizon-effect`

  Enable (default), disable celestial body growth illusion near the horizon.

- `--enable-mouse-pointer`, `--disable-mouse-pointer`

  Enable, disable (default) extra mouse pointer. Useful in full screen mode for old Voodoo based cards.

- `--enable-panel`, `--disable-panel`

  Enable (default) the instrument panel.

- `--enable-random-buildings`, `--disable-random-buildings`

  Enable, disable (default) random buildings. Note that random buildings take up a lot of memory.

- `--enable-random-objects`, `--disable-random-objects`

  Enable (default), disable random scenery objects.

- `--enable-random-vegetation`, `--disable-random-vegetation`

  Enable (default), disable random vegetation such as trees. Requires a graphics card that supports GLSL Shaders, which some older, or less powerful graphics cards do not.

- `--enable-specular-highlight`, `--disable-specular-highlight`

  Enable (default), disable specular highlights.

- `--enable-splash-screen`, `--disable-splash-screen`

  Enable (default), disable the simulator splash screen while loading the aircraft/scenery.

- `--enable-wireframe`, `--disable-wireframe`

  Enable, disable (default) display of wireframes. If you want to know what the world of FlightGear looks like internally, try this!

- `--fog-disable`, `--fog-fastest`, `--fog-nicest`

  Set the fog level. To cut down the rendering efforts, distant regions vanish in fog by default. If you disable fog you'll see farther, but your frame rates will drop. Using `--fog-fastest` will display a less realistic fog, by increase frame rate. Default is `--fog-nicest`.

- `--fov`=*degrees*

  Sets the field of view in degrees. Default is 55.0.

- `--materials-file`=*file*

  Specify the materials file used to render the scenery.
  Default: `Materials/regions/materials.xml`.

- `--geometry`=*WWWxHHH*

  Defines the window resolution. E.g., `--geometry`=*1366x768*.

- `--max-fps`=*Hz*

  Limit the maximum frame rate of the simulator to frequency Hz (frames per second).

- `--shading-smooth`, `--shading-flat`

  Use smooth shading (default), or flat shading which is faster but less pretty.

- `--texture-filtering`=*value*

  Configure anisotropic texture filtering. Values are 1 (default), 2, 4, 8 or 16.

- `--view-offset`=*{LEFT|RIGHT|CENTER|own degrees}*

  Allows setting the default forward view direction as an offset from straight ahead. Possible values are LEFT (look left – 45°), RIGHT (look right – 315°), CENTER (look straight ahead – 0°), or a specific number of degrees. Useful for multi-window display.

- `--terrain-engine`=*{tilecache|pagedLOD}*

  Choose the terrain engine to use: *tilecache* is the "traditional" terrain engine (recommended); *pagedLOD* is a new, experimental terrain engine designed to minimize memory usage by loading more detailed versions of scenery objects on demand.
  The *pagedLOD* engine is available only if FlightGear was compiled with GDAL support.

- `--lod-levels`=*levels*

  Set the level of detail levels, where *levels* is a space-separated list of numeric levels. This option is available only if the terrain engine in use is *pagedLOD*.

- `--lod-res`=*resolution*

  Set the terrain mesh resolution. This option is available only if the terrain engine in use is *pagedLOD*.

- `--lod-texturing`=*{bluemarble|raster|debug}*

  Set the terrain texture method. This option is available only if the terrain engine in use is *pagedLOD*.

- `--lod-range-mult`=*multiplier*

  Set the range multiplier (the breakpoint from a low to a high level of detail). This option is available only if the terrain engine in use is *pagedLOD*.

- `--enable-texture-cache`, `--disable-texture-cache`, `--texture-cache-dir`=*path*

  Enable, disable (default) the texture cache for fast loading.
  Additionally, with `--texture-cache-dir` you can indicate the folder where the cache will be stored (the default is `$FG_HOME/TextureCache`).

4.4.9 HUD Options

- `--enable-anti-alias-hud`, `--disable-anti-alias-hud`

  Control whether the Head-Up Display (HUD) is shown anti-aliased.

- `--enable-hud`, `--disable-hud`

  Control whether the HUD is displayed. Defaults to disabled.

- `--enable-hud-3d`, `--disable-hud-3d`

  Control whether the 3D HUD is displayed. Defaults to disabled.

- `--hud-culled`, `--hud-tris`

  Display the percentage of triangles culled, or the number of triangles rendered in the HUD. Mainly of interest to graphics developers.

4.4.10 Aircraft System Options

- `--com1`=*frequency*, `--com2`=*frequency*

  Set the COM1/COM2 radio frequency.

- `--nav1`=*[radial:]frequency*, `--nav2`=*[radial:]frequency*

  Set the NAV1/NAV2 radio frequency and radial.

- `--adf1`=*[radial:]frequency*, `--adf2`=*[radial:]frequency*

  Set the ADF1/ADF2 frequency. You can optionally specify the rotation angle of its compass card by prefixing the frequency with the angle and a colon.

- `--dme`=*{nav1|nav2|frequency}*

  Set the DME to NAV1, NAV2, or a specific frequency and radial.

- `--failure`=*{electrical|pitot|static|vacuum}*

  Fail a specific aircraft system.
  Valid systems are `pitot`, `static`, `vacuum`, `electrical`. Specify multiple times to fail multiple systems.

4.4.11 Time Options

- `--enable-clock-freeze`, `--disable-clock-freeze`

  Control whether time advances normally or is frozen.

- `--start-date-gmt`=*yyyy:mm:dd:hh:mm:ss*,
  `--start-date-lat`=*yyyy:mm:dd:hh:mm:ss*,
  `--start-date-sys`=*yyyy:mm:dd:hh:mm:ss*

  Specify the exact startup time/date. The three functions differ in that they take either Greenwich Mean Time, the local time of your virtual flight, or your computer system's local time as the reference point.

  Incompatible with `--time-match-local`, `--time-match-real`.

- `--time-match-local`, `--time-match-real`

  With `--time-match-local` the simulator time is read from the system clock, and is used as is. When your virtual flight is in the same timezone as where your computer is located, this may be desirable, because the clocks are synchronized. However, when you fly in a different part of the world, it may not be the case, because there is a number of hours difference, between the position of your computer and the position of your virtual flight.

  The option `--time-match-real` (default) takes care of this by computing the timezone difference between your real world time zone and the position of your virtual flight, and local clocks are synchronized.

Incompatible with:
`--start-date-gmt`,
`--start-date-lat`,
`--start-date-sys`.

- `--time-offset`=*[+-]hh:mm:ss*

  Specify a time offset relative to one of the time options above.

- `--timeofday`=*{real|dawn|morning|noon|afternoon|dusk|evening|midnight}*

  Set the time of day. Valid parameters are `real`, `dawn`, `morning`, `noon`, `afternoon`, `dusk`, `evening`, `midnight`.

4.4.12 Network Options

- `--multiplay`=*{in|out},Hz,host,port*, `--callsign`=*name*

  Set multiplay options and aircraft call-sign. See Section 6.1, Multiplayer.

- `--httpd`=*port*, `--telnet`=*port*

  Enable HTTP server or telnet server on the specified port to provide access to the property tree.

- `--jpg-httpd`=*port*

  Enable screen shot HTTP server on the specified port. This option is deprecated. Instead, use `--httpd`, where screenshots will be available at `http://{IP}:{port}/screenshot`, where {IP} is the IP number of the computer on which FlightGear is running, {port} is the port number specified for `--httpd`.

- `--proxy`=*[user:password@]host:port*

  Specify a proxy server to use. Username and password are optional; if present, they should be given as MD5 hashes.

- `--enable-fgcom`, `--disable-fgcom`

  Enable, disable the FGCom (voice ATC) integration.

- `--disable-hold-short`

  In multiplayer mode, when we choose the starting position on the runway, by default we will be moved to the "hold short" position, The `--disable-hold-short` option allows you to disable this behavior, but keep in mind that this is undesirable as it may disrupt others taking off or landing.

- `--allow-nasal-from-sockets`

  Remove the security flag, which means that network connections will have full access to the simulator, including running any Nasal scripts. Make sure you have adequate security (such as a firewall that is blocking external connections).

- `--enable-sentry`, `--disable-sentry`

  Enable, disable crash and error reports to be sent to the development team for analysis. If enabled, data with crashes are sent to the Sentry.io platform.

4.4.13 Route/Waypoint Options

- `--wp`=*ID[@alt]*

  Allows specifying a waypoint for the GC autopilot; it is possible to specify multiple waypoints (i.e. a route) via multiple instances of this command.

- `--flight-plan`=*file*

  This is more comfortable if you have several waypoints. You can specify a file to read them from.

4.4.14 IO Options

These options are provided for the advanced user.

More detailed descriptions of the various IO parameters can be found in the README.IO file within the Docs directory of your FlightGear installation.

- `--atlas`=*params*

  Open connection using the Atlas protocol (used by Atlas and TerraSync).

- `--atcsim`=*params*

  Open connection using the ATC Sim protocol (atc610x).

- `--AV400`=*params*

  Open connection to drive a Garmin 196/296 series GPS.

- `--AV400Sim`=*params*

  Open connection to drive a Garmin 400 series GPS.

- `--AV400WSimA`=*params*

  Open connection for “A” channel using Garmin WAAS GPS protocol.

- `--AV400WSimB`=*params*

  Open connection for “B” channel using Garmin WAAS GPS protocol.

- `--flarm`=*params*

  Open connection using the FLARM protocol, which includes NMEA/GPS and traffic reporting messages.

- `--generic`=*medium,direction,hz,options,...*

  Open connection using a general protocol for which a specific protocol is defined in the XML file contained in the directory: `$FG_ROOT/Protocol`. This way we can easily extend I/O protocols by adding our own definitions in XML files. For more details on how to use this protocol, see the `$FG_ROOT/Docs/README.protocol` file.

- `--garmin`=*params*

  Open connection using the Garmin GPS protocol.

- `--igc`=*params*

  Open connection using the IGC (International Gliding Commission) protocol. Example of use: `fgfs --igc=file,out,1,output-file.igc`.

- `--joyclient`=*params*

  Open connection to an Agwagon joystick.

- `--jsclient`=*params*

  Open connection to a remote joystick.

- `--native-ctrls`=*params*

  Open connection using the FG native Controls protocol.

- `--native-fdm`=*params*

  Open connection using the FG Native FDM protocol.

- `--native-gui`=*params*

  Open connection using the FG Native GUI protocol.

- `--native`=*params*

  Open connection using the FG Native protocol.

- `--nmea`=*params*

  Open connection using the NMEA protocol.

- `--opengc`=*params*

  Open connection using the OpenGC protocol.

- `--props`=*params*

  Open connection using the interactive property manager.

- `--pve`=*params*

  Open connection using the PVE protocol.

- `--ray`=*params*

  Open connection using the RayWoodworth motion chair protocol.

- `--rul`=*params*

  Open connection using the RUL protocol.

4.4.15 Debugging options

These options are provided for the advanced user.

- `--console`

  Display a console window for simulator/aircraft debugging purposes.
  This option is recognized only on Microsoft Windows operating systems; on other systems, debug messages are always printed to standard output/standard error.

- `--developer`

  Enable developer mode.

- `--enable-fpe`

  Enable abort on a Floating Point Exception.

- `--fgviewer` *filename*

  Instead of loading the entire simulator, load a lightweight OSG viewer with the object specified in *filename*. Useful for checking aircraft models.

- `--jsbsim-output-directive-file`=*file*

  Log JSBSim properties in a CSV file. An output directives file contains an `<output type="CSV"></output>` element, within which should be specified the parameters or parameter groups that should be logged.

- `--json-report`

  Print a report in JSON format on the standard output. The report will give useful information for debugging purposes, such as the FlightGear version, the scenery/aircraft paths in use, the TerraSync and the data download directories and the paths to navigation data files.

- `--log-level`=*{bulk|debug|info|warn|alert}*

  Set the minimum logging level. Valid values are `bulk`, `debug`, `info`, `warn`, `alert`. Log messages having a severity greater than or equal to the specified value are recorded; the others are discarded.

- `--log-class`=*{none|terrain|astro|flight|input|gl|view|cockpit|general|math|event|aircraft|autopilot|io|clipper|network|atc|nasal|instrumentation|systems|ai|environment|sound|navaid|gui|terrasync|particles|headless|osg|undefined|all}*

Log only events belonging to the specified log classes (`all` logs all events, `none` logs none). Multiple classes can be specified by separating them with commas or pipes, for example: `--log-class`=*ai,flight*.

- `--log-dir`=*dir*

  Save the logs file in the given directory. If *dir* is `desktop`, the logs are saved on the Desktop. This option may be given several times, using a different directory each time. Inside the specified directory, the log file will be named `FlightGear_YYYY-MM-DD_num.log`, where `YYYY-MM-DD` is the current date and `num` is a progressive number starting at 0.

- `--prop`:[*type*:]*property*=*value*

  Set *property* to *value*.

  Example: `--prop`:/engines/engine[0]/running=*true* starts the simulator with running engines. Another example, which fills the Cessna for a short flight:

  `--aircraft`=*c172p*
  `--prop`:/consumables/fuel/tank[0]/level-gal_us=*10*
  `--prop`:/consumables/fuel/tank[1]/level-gal_us=*10*

  You may optionally specify the property *type* (`bool`, `double`, `float`, `int`, `long`, `string`).

- `--prop:browser`=*property*

  After starting the simulator, open the properties dialog immediately on the given property. If you need more dialogs, just add more browser indexes like browser[1], browser[2], etc., for example:

  `--prop:browser`=*/sim/presets* # browser[0] by default
  `--prop:browser`[1]=*/devices/status/keyboard/event*

  which will open two dialogs with the given properties.

- `--trace-read`=*property*

  Trace the reads for a property; multiple instances are allowed.

- `--trace-write`=*property*

  Trace the writes for a property; multiple instances are allowed.

- `--uninstall`

  Remove `$FG_HOME` directory. For Windows, it additionally removes `TerraSync`, `Aircraft` and `TextureCache` directories from download directory.

4.5 Joystick support

Could you imagine a pilot in his or her Cessna controlling the machine with a keyboard alone? For getting the proper feeling of flight you will need a joystick/yoke plus rudder pedals.

FlightGear has integrated joystick support, which automatically detects any joystick, yoke, or pedals attached. Simply plug in your joystick and start the simulator.

You can see how FlightGear has configured your joystick by selecting *File* → *Joystick Configuration* from the menu. This dialog shows the name of your joystick, and what each of the buttons and control axes have been set to. You can press a button or move the joystick to see exactly what control it maps to.

If you have a common joystick, there's every chance that someone has already set up FlightGear specific configuration for it, and you can simply go and fly! If you wish to change the configuration of a particular button/axis, simply edit it using the Joystick Configuration dialog.

If your joystick is more unusual, then FlightGear will by default use a simple joystick configuration for it. To change the configuration, simply use the Joystick Configuration dialog to select what you wish each button or movement to do. The configuration takes effect immediately and will be saved for your next flight.

Chapter 5

In-flight: All about instruments, keystrokes and menus

The following is a description of the main systems for controlling the program and piloting the plane. It is assumed that the reader is already familiar with flying, possibly from experience on other simulators. If you are completely new to flying, the tutorials in section 7, Tutorials are a better resource for learning to fly using FlightGear.

A short leaflet showing the standard keys and designed to be printed can be found under the installation directory:

```
FlightGear/Docs/FGShortRef.pdf
```

A reference to most of the keyboard controls can be found under the Help menu in the simulator.

5.1 Starting the engine

Depending on the type of aircraft, you may have to start the engine(s) before you can go flying. The instructions below are generic. Check the aircraft help or aircraft tutorials for more specific instructions.

Once you've started the engine, you should check whether the parking brakes are engaged. If so, press the **b** key to release them.

5.1.1 Piston Aircraft

For piston-engined aircraft, the magnetos are controlled by the **{** and **}** keys. On most aircraft, the starter is engaged using the **s** key. On multi-engined aircraft you can select which engines to control. Use **~** to select all the engines at once. Most magnetos have 4 positions – OFF, LEFT, RIGHT and BOTH. So, to start the selected engine, press the **}** key three times, then hold down **s**.

Note that the starting procedure for powerful WWII-era fighter aircraft is often more complex. See the aircraft help for details.

5.1.2 Turboprop Aircraft

Starting a turboprop engine generally requires simply moving the condition lever from Off to Idle, using the **m** key.

5.1.3 Jet Aircraft

Starting a jet aircraft is significantly more complex, and the controls vary between different aircraft.

1. Set cutoff to ON
2. Engage the starter
3. Once the engines spools up to approximately 5 % N1, set cutoff to OFF
4. Disengage the starter once the engine has reached operational speed

5.2 Keyboard controls

While joysticks, yokes and rudder pedals are supported, you can fly FlightGear using the keyboard alone or in conjunction with a mouse, described below. However you choose to control the aircraft, you will need to use the keyboard for at least some control actions.

These key bindings are not hard-coded, but user-adjustable. You can check these setting via the file `keyboard.xml` which can be found in the main FlightGear directory. This is a human-readable plain text file. We do not recommend modifying the files in the installation directory, but based on this file, you can create your own, in any location, and include it using a `--config`=*path/to/your-keyboard.xml* command line option. Although it is perhaps not the best idea for beginners to modify this file, more advanced users will find it useful to change key bindings according to their wishes, e.g. to match other simulators. Also remember that each aircraft can also have its own key bindings, which you should look for in the catalog of a particular aircraft.

5.2.1 Aircraft controls

In order to have full control of the plane via the keyboard you should ensure that `NumLock` is on, and the FlightGear window must be active (i.e. have focus.)

The keys shown in table 5.1 cause a deflection of the aircraft control surfaces or its throttle position.

Additionally, table 5.2 shows the keys that control the engines.

And the keys for the control of secondary aircraft systems are included in table 5.3.

Key	Action
9 / 3	Throttle
4 / 6	Aileron
8 / 2	Elevator
0 / Enter	Rudder
5	Center aileron/elevator/rudder
7 / 1	Elevator trim

Table 5.1: Primary aircraft controls

Key	Action
!	Select 1st engine
@	Select 2nd engine
#	Select 3rd engine
$	Select 4th engine
~	Select all engines
{	Decrease magneto on selected engine
}	Increase magneto on selected engine
s (hold)	Fire starter on selected engine(s)
M / m	Lean/Enrich selected engine mixture
N / n	Decrease/Increase selected propeller RPM

Table 5.2: Engine control keys

Key	Action
b (hold)	Apply all brakes
, / . (hold)	Apply left/right brake (useful for differential braking)
l	Toggle tail-wheel lock
B	Toggle parking brake
g / G	Raise/lower landing gear
Space	Push To Talk (PTT)
- / _	MP text chat menu/entry
[/]	Retract/extend flaps
j / k	Retract/extend spoilers
Ctrl-b	Toggle speed brakes

Table 5.3: Secondary aircraft controls

Numpad Key	View direction
Shift-8	Forward
Shift-7	Left/forward
Shift-4	Left
Shift-1	Left/back
Shift-2	Back
Shift-3	Right/back
Shift-6	Right
Shift-9	Right/forward

Table 5.4: View directions

5.2.2 Simulator controls

To change the view direction, you must de-activate **NumLock**. The available controls are listed in table 5.4.

Additionally, table 5.5 includes the keys that allow you to control the display.

Key	Action
P	Toggle instrument panel on/off
c	Toggle 3D/2D cockpit (if both are available)
S	Cycle panel style full/mini
Ctrl-c	Toggle panel/cockpit hotspot visibility
h	Toggle HUD
H	Change HUD brightness
i / I	Minimize/maximize HUD
x / X	Zoom in/out
Ctrl-x	Reset zoom to default value
v / V	Cycle view modes forth and back
Ctrl-v	Reset view modes to pilot view
z / Z	Increase/Decrease visibility (fog)
Ctrl-z	Reset visibility to default value
F10	Toggle menu on/off
Shift-F10	Toggle fullscreen mode on/off

Table 5.5: Display options

Finally, the general simulator controls are included in table 5.6.

5.2.3 Autopilot controls

FlightGear supports two types of autopilot – a generic autopilot that works with all aircraft (even those that would not normally have an autopilot), and aircraft-specific autopilots that

Key	Action
p	Pause simulator
a / A	Simulation speed up/slow down
t / T (hold)	Clock speed up/slow down
Ctrl-r	Instant replay
F3	Save screenshot
Esc	Exit program
Shift-Esc	Reset simulator

Table 5.6: General simulator controls

are controlled from within the cockpit.

The generic autopilot is controlled via the keys shown in table 5.7.

Key	Action
Backspace	Toggle autopilot
Ctrl-a	Toggle altitude lock
Ctrl-g	Toggle glide slope lock (NAV 1)
Ctrl-h	Toggle heading hold
Ctrl-n	Toggle NAV 1 lock
Ctrl-s	Toggle autothrottle
Ctrl-t	Toggle terrain follow (AGL) lock
Ctrl-u	Add 1000 ft to your altitude (emergency)
F6	Toggle autopilot heading mode
F11	Autopilot altitude dialog
8 / 2	Altitude adjust
4 / 6	Heading adjust
9 / 3	Autothrottle adjust

Table 5.7: Autopilot controls

Ctrl-t is especially interesting as it makes your aircraft behave like a cruise missile, and follow the terrain. **Ctrl-u** might be handy in case you feel you're just about to crash.

5.3 Mouse-controlled actions

As well as selecting menu items and clicking on controls in the cockpit, your mouse can be used for a variety of other valuable functions in FlightGear.

There are three mouse modes: Normal, Flight Control, and View Direction. These modes enable you to Interact (e.g. click things), fly the aircraft, and look around, respectively. You can change between modes by using the **Tab** key.

5.3.1 Normal mode

In normal mode, you can control the menus and the panel controls. This mode is indicated by a normal arrow cursor.

To change a switch or toggle, simply click on it with the left or middle mouse button.

To change a knob on a radio or linear control such as the throttle, click on the left-hand side to decrease the value, and the right-hand side to increase the value. Click with the left mouse button to make a small adjustment, or the middle button to make a large one. Some controls, such as radios, also support using the mouse wheel, while others, such as throttles, support dragging the mouse with the left button pressed.

Pressing **`Ctrl-c`** highlights the clickable hotspots of objects that may be acted upon.

5.3.2 Flight Control mode

In control mode you can control the aircraft flight controls by moving the mouse. This mode is indicated by a cross-hair mouse cursor.

In this mode, moving the mouse left or right controls the ailerons, which will roll the aircraft. Moving the mouse forwards and or backwards controls the elevator and changes the pitch of the aircraft.

Holding the left mouse button down changes the behaviour so that moving the mouse left/right controls the rudder. Holding the middle mouse button down and moving the mouse forwards/backwards controls the throttle.

Finally, the scroll-wheel may be used to set the elevator trim.

This mode is particularly useful if you do not have a joystick, as it provides for much finer control of the aircraft than using the keyboard alone. If you intend to use the mouse to control the aircraft regularly, it is recommended that you enabled auto-coordination, so that the ailerons are linked to the rudder. This can be done using `--enable-auto-coordination` when launching the simulator, or selecting *auto-coordination* from the launcher.

5.3.3 View Direction mode

In view mode you can look around using the mouse. This mode is indicated by a double-headed arrow cursor.

Simply moving the mouse pans and tilts your view in the current position. This is particularly useful for looking around the cockpit, or out a side window. The scroll-wheel can be used to zoom in or out. Clicking the left mouse button resets the view back to its initial position, usually straight ahead.

Holding down the middle mouse button and moving the mouse allows you to move the viewpoint itself left/right and up/down. Moving the mouse while both the middle button and **`Ctrl`** are held down allows you to move the viewpoint forwards and backwards.

5.4 Menu entries

The menu bar provides access to a variety of options for the simulator and the aircraft. Many aircraft have their own menu items, for changing their registration to automatically starting

their engines. These can be found at the end of the menu bar.

To display or hide the menu bar, press **F10**. You can also display the menu automatically by moving your mouse to the top of the screen.

The menu bar provides the following menus and options.

- **File**
  - **Reset (Shift-Esc)** the flight to the selected starting position. Comes in handy if you get lost or something goes wrong.
  - **Load Flight Recorder Tape (Shift-F1)** will load a pre-recorded flight to play back.
  - **Save Flight Recorder Tape (Shift-F2)** will save the current flight to play back later.
  - **Screenshot (F3)** saves a screen shot as `fgfs-YYYYMMDDHHmmSS.png`.
  - **Screenshot Directory** allows you to set the directory for screenshots to be saved to.
  - **Sound Configuration** configures the volume for various sound channels, and whether they are heard outside the aircraft.
  - **Mouse Configuration** configures the behaviour of the mouse.
  - **Joystick Configuration** allows you to configure your joystick from within the simulator, including axes and button assignments.
  - **Scenery Download** configures automatic download of scenery over the Internet using the TerraSync feature.
  - **Quit (Esc)** exits the program.
- **View**
  - **Toggle Fullscreen (Shift-F10)** toggles between displaying the simulator within a window and displaying full screen.
  - **Rendering Options** configures various graphical features. This allows you to trade eye-candy such as shadows, 3D clouds and specular reflections for frame-rate. To help you achieve a good balance, enable the "Show Frame Rate" option within the *Display Options* menu, which will show the current frame-rate in frames-per-second in the bottom right of the screen. Most people find a frame-rate of around 20 fps adequate for flying. The frame-rate is affected by the graphical features you have enabled, the current visibility (set by **Z**/**z**), the number of objects in view and their Level Of Detail (LOD).
  - **View Options** allow for configuring various display options, including which views are enabled, whether the 2D panel, frame rate and chat messages are to be displayed.
  - **Cockpit View Options** configures the view within the cockpit, the pilot's head movement, black-out due to high G, and redout due to negative G.

  - **Adjust LOD Ranges** sets the range at which different levels of detail are displayed. This affects the textures and objects displayed in the simulator.
  - **Adjust View Position** displays the current view offset. You can adjust this by provide new offsets values. Alternatively you can make small adjustments to your view-point using the mouse (see Section 5.3.3, View Direction mode.)
  - **Adjust HUD Properties** configures the Head-Up Display (HUD) settings, such as transparency and whether anti-aliasing is used to display it.
  - **Toggle Glide Slope Tunnel** displays a virtual tunnel to guide you down to the runway on a normal approach path. Useful if you are having difficulties setting up your approach for landing.
  - **Instant Replay (`Ctrl-R`)** controls the instant replay feature – a good tool for checking your landings!
  - **Earthview Orbital Rendering** controls the earthview orbital rendering based on the NASA Visible Earth image collection.
  - **Stereoscopic View Options** configures stereoscopic display, using Red-Green glasses or other display methods.

- **Location**

  - **Position Aircraft In Air** positions the aircraft at an arbitrary point in the air. You must select a known ground point, e.g. an airport, VOR, long/lat coordinates, and a position relative to that point, e.g distance, direction, altitude. You can also set your initial speed and heading. This is useful for practising approaches.
  - **Select Airport** positions the aircraft at an airport. You can search amongst all the airports that you have installed, view the current METAR, airfield information, airfield layout and select a runway or parking position.
  - **Random Attitude** Sets the aircraft with a random heading, speed and attitude. Useful for practising recovery from unusual attitudes.
  - **Tower position** configures the airport tower used for the Tower View and Tower View Look From views.

- **Autopilot** This menu is only available for aircraft that have the default autopilot configured. Some aircraft may have their own autopilot which is configured through the panel, in which case this menu is disabled.

  - **Autopilot Settings (`F11`)** configures the aircraft autopilot. You can set the autopilot up in a variety of different ways – from simply keeping the wings level, to following an ILS.
  - **Route Manager** opens a window to plan and manage your flight route. Here you can specify the origin airport, destination airport (along with runways and SID and STAR procedures), cruise altitude and speed, and you can add or remove any waypoints such as VOR, NDB, FIX, airport, geographic coordinates. Optionally, for each

waypoint, you can specify a restriction on how high you can fly over due waypoint. The heading, distance and time to the current waypoint may be displayed in the HUD.

- **Previous Waypoint** selects the previous waypoint from the route list.
- **Next Waypoint** selects the next waypoint from the route list.

- **Environment**
  - **Weather** allows you to set the current weather, select a weather scenario, or use the weather reported by the closest weather reporting station (usually an airport) via METAR. You may either use Basic weather, which sets conditions explicitly, or Advanced weather which simulates larger weather systems, at the expense of local accuracy.
  - **Environment Settings** allows you to configure the seasonal textures in use (summer or winter), and ground conditions such as snow level, snow thickness, ice, dust, humidity, vegetation and aurora effect. These are just graphical effects that do not affect the weather.
  - **Time Settings** allows you to set the current time in the simulator, speed up the simulation, and change the rate at which time passes in the simulator. Also displays UTC and local time.
  - **Wildfire Settings** configures whether aircraft crashes will create realistic wildfires, that may spread and be extinguished using an appropriately equipped water bomber aircraft.
  - **Volcanoes** allow for enabling active volcanic activity in the vicinity of your aircraft.

- **Equipment**
  - **Map** displays a moving map, showing airports, navigational beacons etc.
  - **Map (Canvas)** displays an alternative moving map on which different layers (Canvases) can be applied, such as topographic maps (OpenStreetMaps, STAMEN), OpenAIP, VFR, etc.
  - **Map (opens in browser)** displays an alternative moving map within a web browser (requires the internal web server to be running which can be done by `--httpd`=*8080* command line option.)
  - **Stopwatch** displays a simple stopwatch. Useful for instrument approaches.
  - **Fuel and Payload** allows you to set the fuel levels and current payload within the aircraft. Only available on some aircraft.
  - **Radio Settings (`F12`)** sets the frequencies and radials being used by the radios and navigational equipment.
  - **GPS Settings** configures waypoints and views course information for the GPS.

  - **Instrument Settings** allows you to set the altimeter pressure (both in hPa and in inHg) and Heading Indicator offset. Here you also have information about the magnetic declination at your current location.
  - **Random Failures** configures random failure of various aircraft systems and instruments, by setting a Mean Time Between Failures (MTBF) or Mean Cycles Between Failures (MCBF).
  - **System Failures** configures random failure of aircraft systems, such as vacuum system, static pressure, pitot tube, electrical system, control surfaces, landing gear, flaps, brakes aerodynamic, engines, etc.
  - **Instrument Failures** configures random failure of specific aircraft instruments, such as speedometer, altimeter, VSI, turn coordinator, gyro heading indicator, compass and all radio navigation.

- **AI**

  - **Traffic and Scenario Settings** allows you to enable/disable AI (Artificial Intelligence) traffic and the active AI scenarios.
  - **ATC Services in Range** displays the airport communications frequencies for nearby airports.
  - **Wingman Controls** allows you to control AI wingmen (depending on aircraft).
  - **Tanker Controls** allows you to dynamically create an air-to-air refueling tanker, if your aircraft supports it. See Section 6.8, Air-Air Refuelling for further details.
  - **Jetway Settings** allows you to control jetways on some airports.
  - **AI Objects** lets you select an AI object to bring up its own control dialog.

- **Multiplayer**

  - **Multiplayer Settings** allows you to enable Multiplayer by setting a callsign and server.
  - **FGCom Settings** allows you to configure voice communications with other Multiplayer users.
  - **Chat Dialog** allows you to chat with other aircraft in the multi-player environment.
  - **Chat Menu (-)** allows you to send common chat messages to other aircraft or ATC in the multi-player environment. Some menus contain sub-menus of options.
  - **Pilot List** displays list of the other multi-player pilots within range, along with their distance, heading and altitude.
  - **MPCarrier Selection** displays a list of the available MPCarriers.
  - **Lag settings** opens the network lag correction dialog.

  - **swift connection** opens the dialog to start a Swift server to connect to a swift client application previously installed on your computer, which allows you to connect to other networks like VATSIM.

- **Debug** menu contains various options outside the scope of this guide.
- **Help**
  - **Help (opens in browser)** opens the help system in a browser window.
  - **Documentation browser** opens an in-sim documentation reader.
  - **Aircraft Help** displays information specific to the aircraft.
  - **Aircraft Checklists** displays specific checklists for this aircraft, if available.
  - **Common Aircraft Keys** lists the basic keys for controlling the aircraft.
  - **Basic Simulator Keys** lists the basic keys for the controlling the simulator.
  - **Tutorials** allows you to start an in-simulator tutorial for the current aircraft. This is only available on some aircraft. See Tutorials below for details.
  - **About** displays information about the current version of FlightGear, subsystem versions, graphics engine version, build and revision number and graphics card information.

5.5 The Instrument Panel

Figure 5.1: The 3D cockpit of the Cessna 172P Skyhawk (1982)

Aircraft within FlightGear can have both a 2-dimensional and a 3-dimensional instrument panel within the cockpit. The 3-dimensional cockpit provides a much more realistic pilot-eye view, but can be difficult to read with small monitors.

The default Cessna 172P (c172p) has both a 3-dimensional and 2-dimensional cockpit. The 3D cockpit is activated by default when you start FlightGear, but you can overlay the 2D instrument panel by selecting: *View* → *Display Options* → *Show 2D Panel* from the menu, or by pressing **P**.

All panel levers and knobs can be operated with the mouse. To change a control, just click with the left/middle mouse button on the corresponding knob or lever. For controls that have a range of positions, use the middle mouse button for larger adjustments. In general, clicking on the right side of a control will increase the value, while clicking the left side of the control will decrease the value.

Some instruments (particularly radios) also support use of a mouse scroll-wheel to change values. Other controls (such as throttle) can also be dragged using the left mouse button.

5.6 The Head-Up Display

FlightGear also provides a HUD (Head-Up Display). HUDs are generally found in military aircraft and some very advanced jets. However, FlightGear also allows you to use a HUD on many GA aircraft. To activate the HUD, press **h** key.

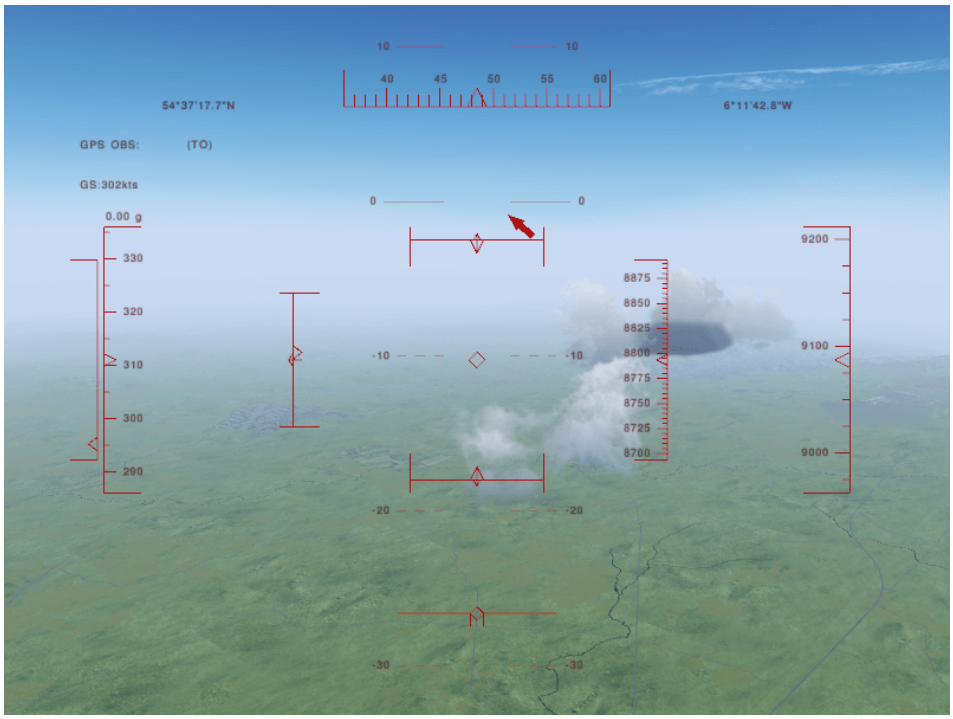

Figure 5.2: The HUD, or Head-Up Display

The HUD shown in Fig. 5.2 displays all main flight parameters of the plane. In the center you find the pitch indicator (in degrees) with the aileron indicator above and the rudder indicator below. A corresponding scale for the elevator can be found to the left of the pitch scale

along with a pitch trim indicator. On the bottom there is a simple turn indicator.

There are two scales at the extreme left: The inner one displays the speed (in kts) while the outer one indicates position of the throttle. The two scales on the extreme right display your height – the left one shows the height above ground while the right of it displays height above sea-level, both being displayed in feet.

Besides this, the HUD delivers some additions information in the upper area. On the left you will find GPS, Ground Speed, and G-Force. The top also displays your current position, in latitude and longitude.

The large arrow points to the active control tower.

You can change color of the **HUD** using the **`H`** or **`h`** key. Pressing the toggle **`i`**/**`I`** minimizes and maximizes the HUD.

Chapter 6

Features

FlightGear contains many special features, some of which are not obvious to the new user. This section describes how to enable and make use of some of the more advanced features.

Many of the features are under constant development, so the information here may not be completely up-to-date. For the very latest information (and new features), see the FlightGear Wiki, available from https://wiki.flightgear.org.

6.1 Multiplayer

FlightGear supports a multiplayer (MP) environment, allowing you to share the air with other flight-simmers. For server details and to see who is online (and where they are flying), have a look at the excellent multiplayer map, available from:
http://mpmap02.flightgear.org

Click on the "server" tab to see a list of multiplayer servers.

6.1.1 Quick Start

You can connect to the MP environment from the *Multiplayer Settings* dialog under the *Multiplayer* menu. Simply select the server closest to you from the list, enter a callsign (which will be seen by other players), and click "Connect".

To see a list of other pilots are in the area, select *Pilot List* from the *Multiplayer* menu.

All the standard MP servers are interconnected, so there is no need to be on the same server as people you are flying with.

6.1.2 Other Methods

If you are connecting to a non-standard server, or the above method does not work, you can also connect using the methods below.

Using the FlightGear Launcher

The *Settings* screen of the FlightGear Launcher has a section for Multiplayer. You can select an existing server from the drop-down list, or configure access manually by selecting *Custom server*. The latter requires you to provide a hostname and port in the format `hostname:port`.

You should also configure the callsign to identify yourself. This can be up to 7 characters in length.

Using the Command Line

The basic arguments to pass to fgfs for multiplayer are these:

```
--multiplay=out,10,<server>,<portnumber>
--multiplay=in,10,<client>,<portnumber>
--callsign=<callsign>
--enable-ai-models
```

where

1. `<portnumber>` is the TCP port number of the server e.g. 5000.
2. `<server>` is the name of the multiplayer server e.g. mpserver01.flightgear.org.
3. `<client>` is the name of your computer, or the IP address ip address of the network interface being used by FG to connect to the server – even if that's a local 192.168 type address. e.g. 192.168.0.1
4. `<callsign>` is the call sign to identify yourself, up to 7 characters, e.g. N-FGFS.

Once the simulator has loaded, you should see yourself displayed on the map. If you don't, check the console for error messages and see the Troubleshooting section below.

6.1.3 Troubleshooting

To get multiplayer to work, we need information about the IP address of our computer and the ability to communicate with the server. How to get this information depends on your configuration and is described below.

Those using a USB modem to connect to the Internet

Firstly, you will need to know the IP address of the network you are going to be running FG multiplayer over. If your Internet connection is via an ADSL modem that plugs directly into your computer with a USB connection, you should be able to find your IP address by visiting https://www.whatismyip.com. Please note that this address may very well change every now and again – if MP stops working, check this first.

Those using some kind of Ethernet router to connect to the Internet

Otherwise, your connection is likely via some kind of router that connects to your computer via an RJ-45, or "Ethernet" connector (similar shape to most Western telephone plugs), or by a wireless link. You need to find the IP address of that network interface.

Under Linux, this can be found by logging in as root and typing "ifconfig". You may find more than one interface listed, beginning with "lo" – ignore that one. You should have something like "eth0" or "wlan0" also listed – look through this block of text for "inet". This will be followed directly by the number you're looking for, e.g. "inet 192.168.0.150"

Under Windows 10, click menu Start and type "cmd". In the terminal window which appears, type "ipconfig". This should show you your IP address – note it down.

If It Still Doesn't Work

You MUST give your local, behind-the-router IP address for MultiPlayer to work. Trust me on this one!

You should check that your firewall is not causing problems – either turn it off temporarily or add an exception to allow incoming connections on port 5000.

If it's still just not working for you, ask nicely on the FlightGear IRC channel and someone should be able to assist.

6.2 Aircraft Carrier

FlightGear supports operations on several carriers, both in the US (such as Nimitz and Vinson in San Francisco) and the Mediterranean sea (such as Eisenhower and Foch). The carriers are equipped with working catapult, arrester wires, elevators, TACAN and FLOLS.

Note that several FG aircraft are carrier-capable, but the seahawk is possibly the easiest to fly to begin with.

6.2.1 Starting on the Carrier

To position your aircraft on the carrier at startup, use the following command line options:

```
--carrier=Nimitz --aircraft=seahawk
```

Please note the uppercase "N" in "Nimitz".

If you are using the launcher, you can select a carrier from the Location screen. Click on the "boat" button on the top right (to the right of the search bar). This will display a list of available carriers, and allow you to set the start position – either on deck or in the air.

If you are using the Windows or OS X launcher to run FG, you should find a text entry box in the gui that allows you to specify command line options, add the above options there.

6.2.2 Launching from the Catapult

Once FlightGear has started, you should ensure that the parking brakes are off and press and hold **L** to engage the launchbar. You must hold down **L** until the launch bar has engaged.

You should notice the aircraft being pulled into alignment with the catapult and see the strops appear and hold down the aircraft. This will only happen if your aircraft is close enough to the correct spot on the catapult; as a rough guide, for the default parking position the seahawk's nose should be roughly level with the deck observation bubble.

To get the carrier into as good a position as possible for launch, open the *AI Objects* dialog from the *AI* menu, select the carrier you are at (in this case, Nimitz), and select the option *Turn to launch course*. You should now notice the carrier begin to pick up speed and turn into the wind, and naturally the deck may tilt somewhat as it turns. You should wait for this maneuver to finish and the deck to return to level before moving on to the next stage.

Being attached to the catapult, you should spool up the engines to full power, ensure the brakes are off and that all flight controls are in a suitable position for launch (stick held right back with the seahawk). When ready, press **C** to release the catapult. Your aircraft will be hurled forward off the deck, and you should be able to raise the undercarriage and climb slowly away, being careful to avoid stalling.

6.2.3 Finding the Carrier – TACAN

Actually finding the carrier in a vast expanse of open water can be very difficult, especially if visibility is poor. To assist with this task, Nimitz is equipped with TACAN (TACtical Air Navigation), which allows a suitably-equipped aircraft (including the Seahawk) to obtain a range and bearing to the carrier. First, you must set the appropriate TACAN channel, 029Y in this case, in the radios dialogue (**F12** or choose *Equipment* → *Radio Settings* from the FG menubar). You should, if within range, notice the DME (Distance Measuring Equipment) instrument show your distance from the carrier, and the ADF (Automatic Direction Finder) instrument (next to the DME in the seahawk) should indicate a bearing to the carrier. Turn to the indicated heading and you should see the DME dial indicate your closing in on the carrier.

6.2.4 Landing on the Carrier

This is the most difficult part of the operation, as in real life. You might well find Andy Ross' tutorial on operating the A4 Skyhawk useful. It is available from here:
https://wiki.flightgear.org/A-4F_Skyhawk_Operations_Manual

Once you have used the TACAN to locate the carrier, you must line up with the rear of the flight deck. As this part of the deck is at an angle to the course of the vessel, you may need to correct your alignment often. Ensure that the aircraft is in the correct configuration for approach (the *Help* → *Aircraft Help* menu should contain useful data for your aircraft) and that the gear and the arrestor hook are down.

As you approach you should see, on the left hand side of the deck, a set of brightly coloured lights – called the Fresnel Lens Optical Landing System (FLOLS). This indicates your position on the landing glideslope. You will see a horizontal row of green lights, and when approximately on the glideslope, an orange light (known in some circles as the "meatball") approximately in line with the green lights. When approaching correctly, the meatball appears in line with the green lights. If you are high it is above, and when low it is below. If you are very low the meatball turns red. If you fly to keep the meatball aligned you should catch number 3 wire.

Carrier landings are often described as "controlled crashes" and you shouldn't waste your time attempting to flare and place the aircraft gently on the deck like you would with a conventional landing – ensuring that you catch the wires is the main thing.

Immediately your wheels touch the deck, you should open the throttles to full power, in case you have missed the wires and need to "go around" again; the wires will hold the aircraft if you have caught them, even at full power.

If you wish, you can then raise the elevators (through the *AI* → *AI Objects* → <*your carrier*> menu), taxi onto one of the elevators, lower it (uncheck the box on the menu) and taxi off into the hangar.

Don't be discouraged if you don't succeed at first – it's not an easy maneuver to master. If after a little practice you find the Seahawk too easy, you could move on to the Seafire for more of a challenge!

6.3 Atlas

Atlas is a "moving map" application for FlightGear. It displays the aircraft in relation to the terrain below, along with airports, navigation aids and radio frequencies.

Further details can be found on the Atlas website:
http://atlas.sourceforge.net

6.4 Multiple Displays

FlightGear supports multiple displays. Using some straightforward XML, you can configure multiple "slave cameras" that are offset from the main view, so you can use multiple monitors to display a single view of the simulator. For example, you can have one display showing the view straight ahead, while two additional displays show the view to either side.

Information on configuring multiple displays can be found in the README.multiscreen file in the docs directory of your FlightGear installation.

6.5 Multiple Computer

FlightGear allows you to connect multiple instances of the program using the very flexible I/O subsystem, and display completely different views and controls on different computers. This can be used in combination with the Multiple Display support to create a more sophisticated environment with separate cockpit panel displays and even a separate control station allowing an instructor to fail instruments, change the weather etc.

An example of this is the 747 cockpit project:
http://geoffair.org/fg/site/Projects/747-JW/

6.5.1 Basic Concepts

Each instance of FlightGear can support a single display. Due to the complexity of the FDM and graphics, FlightGear is very processor-intensive, so running multiple instances of FlightGear

on a single machine is not recommended.

You will therefore need a computer for each view of the simulation you wish to display, including the panel. The computers obviously must be networked and for simplicity should be on the same subnet.

One computer is designated as the primary machine. This computer will run the FDM and be connected to controls such as yokes, joysticks and pedals. As the machine is running the FDM, it usually only displays a simple view, typically the main panel, to maximize performance.

All other computers are designated as secondary. They are purely used for display purposes and receive FDM information from the primary.

6.5.2 Basic Configuration

Creating a basic configuration for multiple displays is straightforward. The primary computer needs to broadcast the FDM and control information to the secondary machines. This is done using the following command line options:

```
--native-fdm=socket,out,60,<IP-secondary>,5505,udp
--native-ctrls=socket,out,60,<IP-secondary>,5506,udp
```

You should change `<IP-secondary>` to the IP of your secondary machine. Finding out the IP of your machine is dependent on your OS; how to do it you can see in Section 6.1.3. If you have multiple secondaries, you can copy these two options in the primary's command line as many times as required, changing the `<IP-secondary>` field appropriately for each secondary and keeping the rest of the fields identical.

After starting the primary, you can start the secondary computers. These need to listen for the information provided by the primary, and also need to have their own FDMs switched off:

```
--native-fdm=socket,in,60„5505,udp
--native-ctrls=socket,in,60„5506,udp
--fdm=null
```

6.5.3 Advanced Configuration

The options listed above will simply show the same view on both machines. You will probably also want to set the following command-line options on most of the computers:

```
--enable-fullscreen (full screen for sdl or windows)
--prop:/sim/menubar/visibility=false (hide menu bar)
--prop:/sim/ai/enabled=false (disable AI ATC)
--prop:/sim/ai-traffic/enabled=false (disable AI planes)
--prop:/sim/rendering/bump-mapping=false
```

If using the primary to display a panel only, you may wish to create a full-screen panel for the aircraft you wish to fly (one is already available for the Cessna 172), and use the following options on the primary:

```
--enable-panel (enable the display of the 2D cockpit panel)
--disable-hud (disable the HUD)
```

```
# switch off rendering the world:
--prop:/sim/rendering/draw-mask/terrain=false
--prop:/sim/rendering/draw-mask/aircraft=false
--prop:/sim/rendering/draw-mask/models=false
--prop:/sim/rendering/draw-mask/clouds=false
```

6.6 Recording and Playback

As well as the Instant Replay feature within the simulator, you can record your flight for later analysis or replay using the I/O system. Technical details of how to record specific FDM information can be found in the `$FG_ROOT/Docs/README.protocol` file.

To record a flight, use the following command line options:

```
--generic=file,out,20,flight.out,playback
```

This will record the FDM state at 20 Hz (20 times per second), using the playback protocol and write it to a file flight.out. To play it back later, use the following command line options:

```
--generic=file,in,20,flight.out,playback
--fdm=external
```

The `playback.xml` protocol file does not include information such as plane type, time of day, so you should use the same set of command line options as you did when recording.

6.7 Text to Speech with Festival

FlightGear supports Text To Speech (TTS) for ATC and tutorial messages through the festival TTS engine (http://www.cstr.ed.ac.uk/projects/festival/). This is available on many Linux distros, and can also be installed easily on a Cygwin Windows system. At time of writing, support on other platforms is unknown.

6.7.1 Installing the Festival system

1. Install festival. Some Linux distros (such as Ubuntu 20.04) have packages available to install from their package managers. Otherwise, you can install it from http://www.cstr.ed.ac.uk/projects/festival/.

2. Check if Festival works. Festival provides a direct console interface. Only the relevant lines are shown here. Note the parentheses!

   ```
   $ festival
   festival> (SayText "FlightGear")
   festival> (quit)
   ```

3. Check if MBROLA is installed. Again, you may be lucky and find it in your distro's package manager. Otherwise, the official page of the project is:

https://github.com/numediart/MBROLA

MBROLA is optional, so you can skip it if you wish. But then you can't use the more realistic voices :(

Run MBROLA and marvel at the help screen. That's just to check if it's in the path and executable.

```
$ mbrola -h
```

6.7.2 Running FlightGear with Voice Support

First start the festival server:

```
$ festival --server
```

Now, start FlightGear with voice support enabled. You can do that by adding the following option to your command line:

`--prop:/sim/sound/voices/enabled=`*`true`*

Of course, you can put this option into your personal configuration file. This doesn't mean that you then always have to use FlightGear together with Festival. You'll just get a few error messages in the terminal window, but that's it. You cannot enable the voice subsystem when FlightGear is running.

To test it is all working, contact the KSFO ATC using the `'` key. You should hear **your** voice first (and see the text in yellow color on top of the screen), then you should hear ATC answer with a different voice (and see it in light-green color).

6.7.3 Troubleshooting

On some Linux distros, festival access is restricted, and you will get message like the following:

```
client(1) Tue Feb 21 13:29:46 2006 : \
  rejected from localhost.localdomain
not in access list
```

Details on this can be found from:
http://www.festvox.org/docs/manual-2.4.0/festival_28.html.

You can disable access restrictions from `localhost` and `localhost.localdomain` by adding the following to a `.festivalrc` file in `$HOME`:

```
(set! server_access_list '("localhost"))
(set! server_access_list '("localhost.localdomain"))
```

Or, you can just disable the access list altogether:

```
(set! server_access_list nil)
```

This will allow connections from anywhere, but should be OK if your machine is behind a firewall.

6.7.4 Installing more voices

I'm afraid this is a bit tedious. You can skip it if you are happy with the default voice. First find the Festival data directory. All Festival data goes to a common file tree, like in FlightGear. This can be `/usr/local/share/festival/` on Unices. We'll call that directory `$FESTIVAL` for now.

1. Check which voices are available. You can test them by prepending `voice_`:

```
$ festival
festival> (print (mapcar (lambda (pair) (car pair)) \
                                    voice-locations))
(kal_diphone rab_diphone don_diphone us1_mbrola \
                  us2_mbrola us3_mbrola en1_mbrola)
nil
festival> (voice_us3_mbrola)
festival> (SayText "I've got a nice voice.")
festival> (quit)
```

2. Festival voices and MBROLA wrappers can be downloaded here:

   http://festvox.org/packed/festival/1.95/

   The don_diphone voice is not the best, but it is comparatively small and well suited for ai-planes. If you install it, it should end up as directory

   $FESTIVAL/voices/english/don_diphone/. You also need to install festlex_OALD.tar.gz for it as $FESTIVAL/dicts/oald/ and run the Makefile in this directory.
   (You may have to add `--heap 10000000` to the festival command arguments in the Makefile.)

3. Quite good voices are "us2_mbrola", "us3_mbrola", and "en1_mbrola". For these you need to install MBROLA (see above) as well as these wrappers:

```
festvox_us2.tar.gz
festvox_us3.tar.gz
festvox_en1.tar.gz
```

   They create directories `$FESTIVAL/voices/english/us2_mbrola/` etc. The voice data, however, has to be downloaded separately from another site (see point 4).

4. MBROLA voices can be downloaded from the MBROLA download page (see above). You want the voices labeled "us2" and "us3". Unpack them in the directories that the wrappers have created: `$FESTIVAL/voices/english/us2_mbrola/` and likewise for "us3" and "en1".

6.8 Air-Air Refuelling

As the name suggests, Air-Air Refueling (AAR) involves refueling an aircraft (typically a short-range jet fighter) from a tanker aircraft by flying in close formation. FlightGear supports several tankers, each of them supporting one of two systems. The first one comprises a boom that connects to a receiver on the refueling aircraft. The second deploys a hose into which the refueling aircraft inserts a probe.

A number of aircraft support AAR, including the T-38, Lightning, A-4F, Vulcan, Victor (which can also act as a tanker) and A-6E. You can tell if a particular aircraft supports AAR by looking at the AI menu. If the *Tanker Controls* item is enabled, the aircraft supports AAR.

6.8.1 Setup

To set up AAR, simply start FlightGear with an AAR-enabled aircraft, take-off and climb to about 15 000 ft. Once cruising at this altitude, open menu *AI* → *Tanker Controls*, select a tanker from the drop-down list and click on **Request**. This will spawn a new tanker in clear air.

FlightGear will report the altitude, heading, speed, and TACAN identifier of the tanker. Program your TACAN with the TACAN identifier reported by the tanker (through the *Equipment* → *Radio Settings* dialog, or your cockpit controls). Depending on your aircraft, the tanker may also appear on your radar. If you require more help to find the tanker, you can click on **Get Position** to be told the tanker location relative to yourself.

Turn to an appropriate heading, guided by the TACAN bearing (you should try a "leading" approach to close in on the tanker) and look for the tanker on the radar or nav. screen. Around 5 nm away, you should reduce your speed to around 20 kts faster than the tanker – a "slow overtake". Bigger tankers will be visible from about 10 nm, but smaller ones may be visible just over 1 nm. If you find yourself overshooting, deploy your airbrakes.

Close to within 50 ft of the tanker (don't get too close, or you may collide with the tanker itself). You should see indication in the cockpit that you are receiving fuel (e.g. the A4 fuel gauge has a green light), and you should see the indicated tank load increase. If you still have issues telling whether you are connecting to the tanker, you can turn on option *Report refueling* in the *Tanker controls* window. If you still can't engage with the tanker, increase the *Contact radius* to make the maneuver more resilient to "disconnects".

Once your tanks are full, or you have taken as much fuel as you wish, close the throttle a little, back away from the tanker and continue your intended flight.

Successfully refueling is not easy and can take a lot of practise, as in real life. Here are some helpful hints for making contact.

1. Approach the tanker slowly. It is very easy to overshoot and be unable to spot where the tanker has gone.

2. If you are having difficulty matching the speed of the tanker due to the throttle being too sensitive, try deploying your airbrakes. This will require more power to achieve the same speed and will reduce the throttle sensitivity.

3. To reduce your workload, you may be able to use the autpilot to stay at the correct

altitude and/or speed. This is technically cheating, though NASA recently demonstrated that an advanced autopilot can perform AAR without pilot intervention.

4. Bear in mind that as you receive fuel your aircraft will become heavier and the center of gravity will move, affecting the trim of the aircraft.

5. The tanker aircraft flies a clock-wise "race-track" pattern in the sky. While it is possible to stay connected during these turns, you may find it easier to wait until the tanker has settled on its new course before refueling. The tanker aircraft provides warnings when they are intending to turn.

6.8.2 Multiplayer Refueling

Refuelling is possible within a Multiplayer session using the KC135 or Victor. The pilot of this aircraft should use the callsign "MOBIL1", "MOBIL2" or "MOBIL3". Other numbers are acceptable, but only these three have A-A TACAN channels assigned. These are 060X, 061X and 062X respectively.

If the receiving aircraft uses a YASim FDM, there are no further complications. Should the receiving aircraft be JSBSim-based, the user must make sure that there are no AI tankers in their configuration. This means disabling (commenting out) all refuelling scenarios in the relevant `aircraft-set.xml` and in `defaults.xml`.

MP refuelling works in exactly the same way as AI refuelling and is a fun challenge. It is best to ensure that your network connection is as free from interruptions as possible; the MP code does a degree of prediction if there is a "blip" in the stream of packets and this can make close formation flight very difficult or even impossible.

Part III

Tutorials

Chapter 7

Tutorials

If you are new to flying, an advanced simulator such as FlightGear can seem daunting: You are presented with a cockpit of an aircraft with little information on how to fly it.

In reality, when learning to fly you would have an instructor sitting next to you to teach you how to fly and keep you safe. While we cannot provide a personal instructor for every virtual pilot, there are a number of tutorials available that you can follow to become a proficient virtual pilot.

7.1 In-flight Tutorials

FlightGear contains an in-flight tutorial system, where a simulated instructor provides a virtual lesson. These vary between aircraft, from simple tutorials teaching you how to start the engines on the aircraft, to full lessons teaching you how to fly for the first time. To access tutorials, Select *Help* → *Tutorials* from the menu and click **Start Tutorial**.

The tutorial system works particularly well with the Festival TTS (Text To Speech) system (6.7, Text to Speech with Festival).

For simplicity, run tutorials with AI aircraft disabled. Otherwise, ATC (Air Traffic Control) messages may make it difficult to hear your instructor. Via *AI* → *Traffic and Scenario Settings*, uncheck option *Enable AI traffic*.

Each tutorial consists of a number of discrete steps which you must complete. Your instructor will provide instruction on how to complete each step, observe you perform them, and provide you with additional guidance when required.

Within a tutorial, you may request your instructor to repeat any instructions, by tapping **+**. You may pause the tutorial at any time using the **p** key. To stop the tutorial, select **Stop Tutorial** from the *Help* → *Tutorials* menu.

7.1.1 Cessna 172P Tutorials

If this is your first time flying, a number of tutorials exist for the Cessna 172P designed to teach you the basics of flight, in a similar way to a real flight school.

The tutorials are based around Hilo International Airport (PHTO), Hawaii and Daniel K Inouye International Airport (PHNL), Honolulu, Hawaii. Both these airports are provided in

the base package. To start the tutorials, select the *Cessna 172P* aircraft, and a starting airport of PHTO or PHNL, using the Launcher, or the command line:

```
fgfs --aircraft=c172p --airport=PHTO
```

When the simulator has loaded, select *Help* → *Tutorials* from the menu. You will then be presented with a list of the available tutorials. Select a tutorial. A description of the tutorial is displayed. Press **Start Tutorial** to begin.

7.2 FlightGear Tutorials

The following chapters provide FlightGear specific tutorials to take the budding aviator from their first time in an aircraft to flying in the clouds, relying on their instruments for navigation. If you have never flown a small aircraft before, following the tutorials provides an excellent introduction to flight.

Outside of this manual, there is an excellent tutorial written by David Megginson – being one of the main developers of FlightGear – on flying a basic airport circuit specifically using FlightGear. This document includes a lot of screen shots, numerical material etc., and is available from:

https://web.archive.org/web/20070403233149/http://www.flightgear.org:80/Docs/Tutorials/circuit/

7.3 Other Tutorials

There are many non-FlightGear specific tutorials, many of which are applicable. First, a quite comprehensive manual of this type is the Aeronautical Information Manual, published by the FAA, and available at:

https://www.faa.gov/air_traffic/publications/

This is the Official Guide to Basic Flight Information and ATC Procedures by the FAA. It contains a lot of information on flight rules, flight safety, navigation, and more. If you find this a bit too difficult, you may prefer the *FAA Training Book*,

http://avstop.com/AC/FlightTraingHandbook/,

which covers all aspects of flight, beginning with the theory of flight and the working of airplanes, via procedures like takeoff and landing up to emergency situations. This is an ideal reading for those who want to learn some basics on flight but don't want to spend bucks on getting a published pilot handbook.

While the handbook mentioned above is an excellent introduction on VFR (Visual Flight Rules), it does not include flying according to IFR (Instrument Flight Rules). However, an excellent introduction into navigation and flight according to Instrument Flight Rules written by Charles Wood can be found at:

Flight Simulator Navigation (PDF)[1].

[1] https://static1.1.sqspcdn.com/static/f/1003521/14592887/1318406663403/Flight+Navigation.pdf

Another comprehensive but yet readable text is John Denker's "See how it flies", available at:

https://www.av8n.com/how/

This is a real online text book, beginning with Bernoulli's principle, drag and power, and the like, with the later chapters covering even advanced aspects of VFR as well as IFR flying.

Chapter 8

A Basic Flight Simulator Tutorial

8.1 Foreword

Aviation is about extremes:

- An airplane is quite fragile and flies at high speeds. Yet it is one of the safest forms of transport.
- Pilots must constantly follow rules and procedures. Yet an airplane is a symbol of freedom.
- With a little bit of training, flying a small aircraft becomes easy. Yet if a problem occurs, you must be able to resolve it within a few seconds.
- Many flight tutorials are written with a lot of humor. Yet not taking flying seriously will bring you down to earth prematurely.

The aircraft used in this tutorial is the Cessna 172P (see figure 8.1). This is the aircraft used in many real life flight schools and it is a great airplane to fly.

The following articles complement this tutorial and will answer most questions that may arise as you read through. The first one in particular is a good introduction to the airplane's main components and controls:

- https://www.gleimaviation.com/resources/free-downloads/
- http://www.pilotfriend.com/training/flight_training/aero/principa.htm
- https://en.wikipedia.org/wiki/Aircraft
- https://en.wikipedia.org/wiki/Flight_control_surfaces
- https://en.wikipedia.org/wiki/Airplane_flight_mechanics
- https://en.wikipedia.org/wiki/Aircraft_engine_controls

Figure 8.1: The Cessna 172P

- Flight Simulator Navigation (PDF)[1]
- http://www.free-online-private-pilot-ground-school.com

This tutorial is accurate to the best of my knowledge, but will inevitably contain some mistakes. I apologize in advance for these.

8.2 Starting Up

There are a number of different ways to start FlightGear based on your platform and the distribution you are using. I will refer you back to Chapter 4, Takeoff: How to start the program.

8.2.1 Using the Launcher

FlightGear has a Launcher in which you can choose your aircraft and starting postion. From along the left edge, select the *Aircraft* tab and then choose the *Cessna 172P Skyhawk* airplane (see figure 8.2). Click on the **Back** button if you need to choose a different aircraft.

You can start from any airport for this tutorial, but I will assume that you will start from Daniel K. Inouye International airport at Honolulu (PHNL, see figure 8.3).

From the *Location* tab, once you have selected PHNL (click on the **Back** button if you need to change it), you can then set any number of options for the simulator. For your first flight, I suggest setting the `Time of day` to `Noon` within the *Environment* tab (see figure 8.4).

[1] https://static1.1.sqspcdn.com/static/f/1003521/14592887/1318406663403/Flight+Navigation.pdf

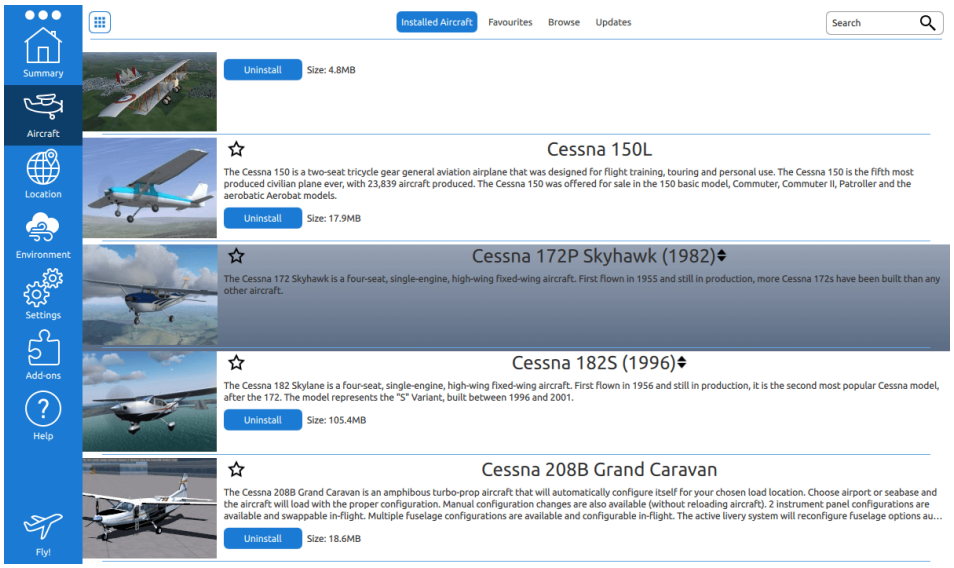

Figure 8.2: Choosing the aircraft

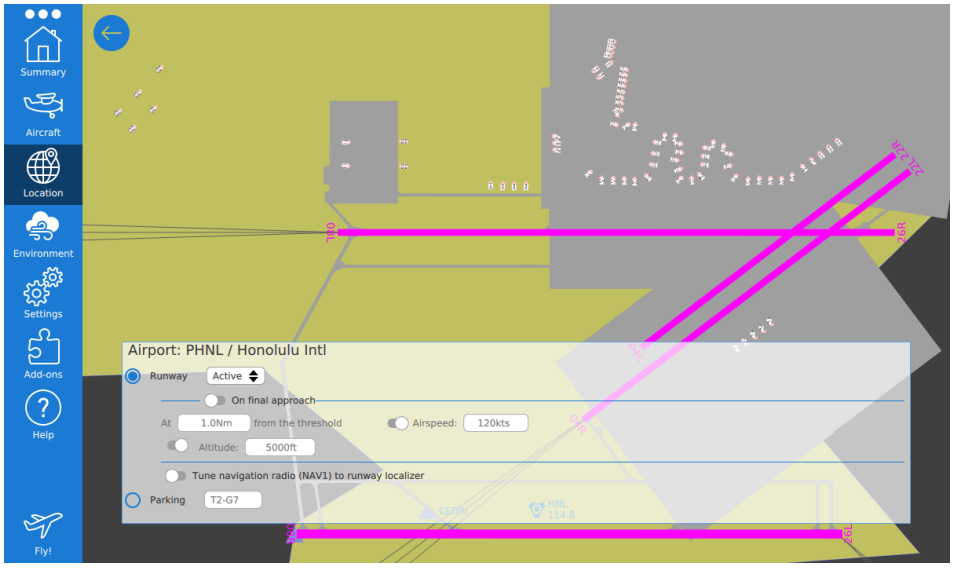

Figure 8.3: The airport diagram for PHNL in the built-in launcher

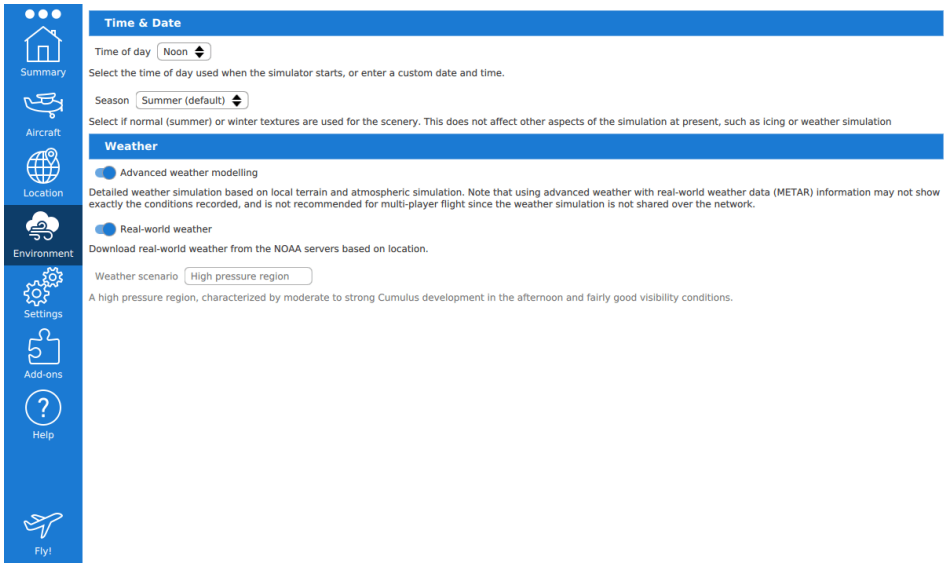

Figure 8.4: The *Environment* tab of the built-in launcher

I also recommend that you start with a small resolution of 800×600. You can change the resolution via the *Settings* tab by clicking on the **Show more** button of section **View and Window** (see figure 8.5). Later on, you can play around with the options and use a higher resolution, but this obviously adversly affects performance. Click on the **Fly!** button and FlightGear will start with the options you have selected.

If you have problems running the latest version FlightGear on your system, you may want to try an earlier version with lower graphics requirements. You can find previous releases on the FTP mirrors on the web page: https://www.flightgear.org/download/mirror/.

8.3 The First Challenge – Flying Straight

Once FlightGear is started, you will see the scene shown in figure 8.6. Your first task is to get the engine running, which is most easily performed via the menu bar *Cessna C172P* → *Autostart*.

The aircraft is at the start of the runway with the engine running at low power. With the propeller now spinning, the airplane may occasionally tremble a little, but it should not move (see figure 8.7).

About the keyboard

- In this tutorial, a lowercase letter key indicates you should simply press that key. An uppercase means you must press **`Shift`** (marked with blue in figure 8.8) and that key. In other words: if you are told to type "v", simply hit the **`v`** key briefly. If you are told to type **`V`**, first press the **`Shift`** key and hold it down; while it is held down, press the **`v`** key, release it then release the **`Shift`** key. (In short: **`V`** is the same as **`Shift-v`**.)

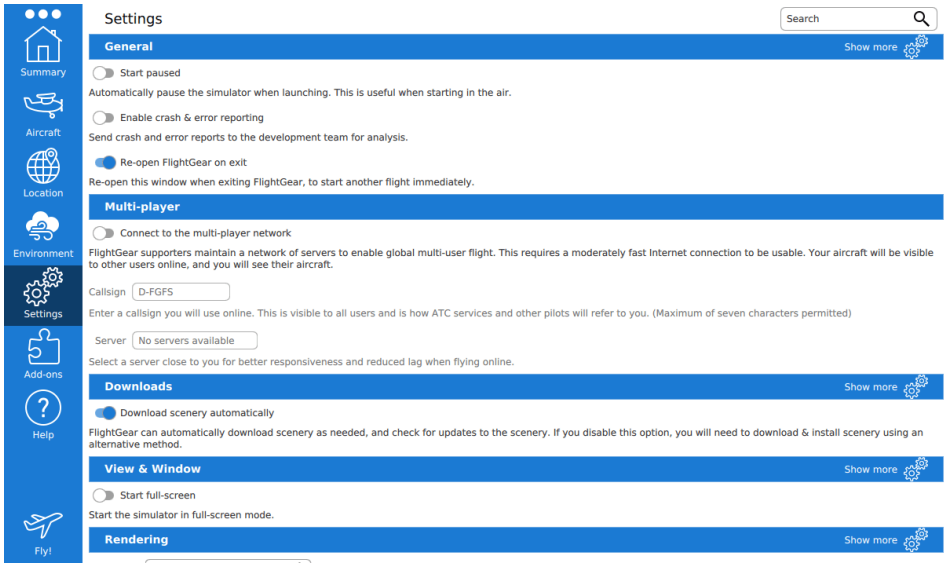

Figure 8.5: The *Settings* tab of the built-in launcher

Figure 8.6: Automatic starting of the Cessna 172P

Figure 8.7: Cessna 172P on runway with engine started

- To be sure that the keys will work regardless of the platform on which you run FlightGear, this tutorial will assume you have the **NumLock** key on. When this is the case, you should see a small green LED at the right of your keyboard. Press the **NumLock** key once to turn the LED on in case it is switched off (marked with green in figure 8.8).

  Having NumLock turned on, we can use the **Home**/**End** (for trim) and **PageUp**/**PageDown** (throttle control) keys, located above the arrow keys (marked with red in figure 8.8). Normally for the Cessna 172P, you should be able to use these keys from the numeric keypad as well, but depending on the platform and operating system it may be different.

Press **v** to view the aircraft from the outside, as shown in figure 8.9. Type **v** repeatedly to cycle through a number of different available views until you return to the cockpit. Pressing **V** will cycle backwards through the views. **Ctrl-v** will return you directly back to the cockpit view.

In real life, we would have inspected all around the plane to check that everything is working, to confirm that nothing is hampering the moving parts, and that nothing is obstructing the instrument openings. In the simulator, this is already done for us before we start.

Hold the **PageUp** (or **NumPad-9**) key down for about eight seconds. You will hear the engine sound rise as the throttle is advanced to full power.

The airplane will start accelerating down the runway. As it does so, it will drift to the left, before finally taking off, banking to the left, falling to the ground and crashing (probably).

You can see a replay of the crash using the *View* → *Instant Replay* menu. Click the **Replay** button at the bottom of the dialog window, then use **v** and **V** to see the airplane from the outside. Figure 8.10 shows the end part of the flight. You can take a snapshot by typing the **F3** key. You can also use the **F10** key to toggle the menu bar on or off.

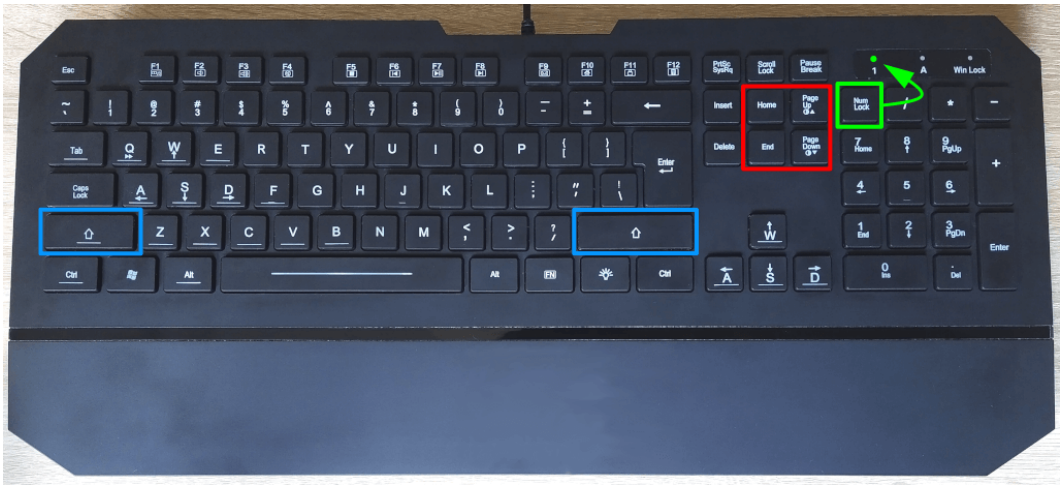

Figure 8.8: The **`Shift`** keys (boxed in blue), **`NumLock`** key (boxed in green), **`Home`**, **`End`**, **`PageUp`** and **`PageDown`** keys (boxed in red)

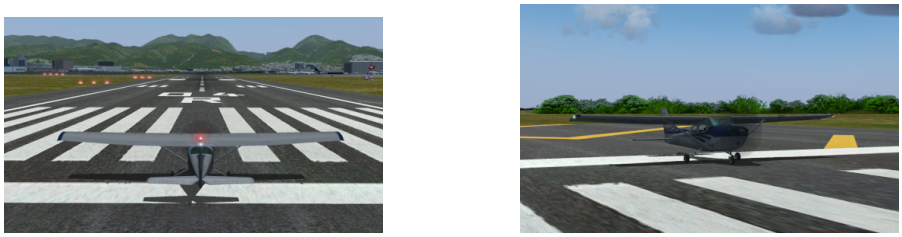

Figure 8.9: External views

Figure 8.10: Viewing the crash using *Instant replay*

Having observed your crash, reset FlightGear using *File* → *Reset* (or the keyboard shortcut **Shift-Esc**), to return to the pre-crash condition.

In order to fly straight, we will need to use the airplane control yoke (see figure 8.11).

Figure 8.11: The yoke

You can control the yoke using a joystick, or by moving the mouse. To use the mouse you need to be in mouse yoke mode. Get into that mode by pressing **Tab**. The mouse cursor becomes a + symbol. Move the mouse and see the yoke moving accordingly. Type **v** to see the plane from the outside. If you move the mouse again you will see movement of the rudder, elevator (tail) and the ailerons (wings). If your viewpoint is too far from the aircraft to see any movement, type **x** a few times to zoom in. Type **X** to zoom back out. **Ctrl-x** returns the view to the default zoom level. Use **Ctrl-v** to change the view back into the cockpit.

Pressing **Tab** again gets you into mouse view mode. In this mode the mouse cursor will be a ↔ symbol. This allows you to look around easily by moving the mouse. Clicking the left mouse button will re-center the view. You can also change your view direction in the normal and yoke modes by holding down the right mouse button and moving the mouse. A further press of **Tab** will return you to the normal mouse mode.

To summarize, the **Tab** key cycles the mouse through three modes:

- *Normal mode*. This mode allows you to click on the menu and on the instrument panel.
- *Yoke mode*. The mouse controls the yoke (+ pointer).
- *View mode*. The mouse controls the view direction (↔ pointer).

Try taking off again using the mouse to control the yoke. Press **Tab** to put the mouse in yoke mode (+ pointer) and raise the engine throttle to maximum by holding the **PageUp** key down. Do not try too hard to keep the airplane rolling straight on the runway using the mouse/yoke. It is okay to let it drift a bit leftwards. Wait until it rises into the air. Then use

the mouse to try and get the airplane to fly straight. (If you want to control the airplane on the ground see section 8.5.)

You will find that you must prevent the airplane from banking to the left or to the right (cf. figure 8.12)… or from plunging toward the ground (figure 8.13). Try to fly more or less straight, with the horizon line slightly above the airplane nose (see figure 8.14).

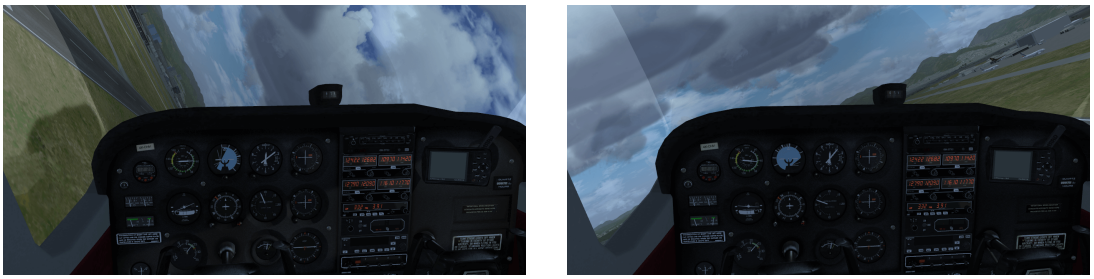

Figure 8.12: Left or right banking

Figure 8.13: An intentional descent?

Whatever your skills at video games or simpler simulators, you will probably not succeed at first. The airplane will crash, probably quite soon after take-off. This is the moment where it is common for the novice pilot to begin to despair and abandon trying to fly a simulator or a real aircraft. Just hold tight and keep trying. Eventually you will develop a feel for the subtle control inputs required.

Common Mistakes

The most common errors are moving the controls too much, which lead to the aircraft getting out of your control, or moving the mouse forward in an attempt to bring the nose up.

You will learn that small yoke movements have a fairly large effect on the aircraft's reaction. Go easy on the controls by using small yoke movement. You may always add a touch more control input if you see the aircraft is responding too subtly. Inversely, too much yoke movement can cause a severe response from the aircraft and then you find yourself in a poor

Figure 8.14: Straight and level flight

situation as you desperately attempt to regain control. If over-control is especially difficult for you, then you may find that decreasing your mouse sensitivity could help during your initial training.

Remember, `pull` the nose up and `push` the nose down. You must pull the yoke aft, by moving the mouse backwards, to raise the nose. Equally, when you want to lower the airplane's nose, you must move the mouse forwards. This can seem odd, but all airplane control yokes are designed that way. With time, you will wonder how you every thought it worked any other way.

If you have difficulty visualising this, the following analogy may help. Imagine a soccer ball is on your desk and you have "glued" your hand to the top of it. If you move your hand forward the ball will roll forward and your fingers will point toward the desk. If you move your hand backward the ball will roll backward and your fingers will now point up at the ceiling. Your hand is the airplane.

Another common mistake is the assumption that the control inputs directly match airplane bank. In other words, you believe if the control yoke is level, the airplane will fly level. This is not true. The yoke controls the *rate* at which the airplane banks. If the airplane is banked 20° to the left and the control yoke is level, the airplane will stay banked at 20° left until some other force affects it. If you want to return the airplane to level flight, you have to turn the control yoke slightly to the right (move the mouse slightly rightwards) and keep it slightly to the right for a while. The airplane will turn slowly rightwards. Once it is level with the horizon, bring the control yoke level too. Then the airplane will stay level (until some other force changes its orientation).

A third error is trying to find "the right position" for the yoke/mouse. Naturally, you will want to find the fine tuning that will leave the airplane fly straight. Actually there is no such

ideal yoke position. The airplane is inherintely unstable in the air. You must constantly correct the airplane's attitude and keep it flying straight with tiny movements of the mouse. This may seem to take all your concentration intially, but just like driving a car, keeping the aircraft straight and level will soon become second nature. For longer flights, you will eventually use the autopilot to keep the airplane level, but this is outside the scope of this tutorial.

To help fine-tune your senses to the control inputs required, keep your eyes on the outside scenery and not get fixated on the instruments or the yoke. Check the angle of the horizon and its height above the airplane's nose. The horizon line and the airplane engine cover are your main flight instruments. Look at the instrument panel only once in a while.

While the mouse is in yoke control mode (+ pointer), do not move it close to the FlightGear window edge. Once the mouse leaves the window, it stops controlling the aircraft, often at the worse possible moment! If you wish to use the mouse outside of the window, first go back to standard mouse mode by pressing **Tab** twice. Or use FlightGear in full screen mode.

You can also control the yoke using the four **keyboard arrow** keys or the keypad **8**, **2**, **4** and **6** keys. While initially this may seem easier than the mouse, you cannot make the very fine adjustments required for accurate flying, so it is much better to persevere with the mouse.

You may hear beeping sounds while flying around the airport. These are landing aid signals. Don't worry about them for the moment.

You will know that you have mastered this when you can make the aircraft climb steadily in the air. The next step is to learn to keep the aircraft at a constant altitude, or to make it ascend or descend slowly and under your control.

Keeping the aircraft at a constant altitude involves observing the altimeter and making small changes with the mouse forwards or backwards to stop the aircraft ascending or descending respectively.

The altimeter instrument is at the middle top of the instrument panel. The long needle shows hundreds of feet, the medium needle shows thousands of feet, the short needle shows tens of thousands of feet. The altimeter represented in figure 8.15 shows an altitude of about 1800 ft, i.e. approximately 549 meters.

As you ascend or descend the altimeter will change accordingly, turning anti-clockwise as you descend, and clockwise as you gain height. If you see the altimeter "unwinding" you will be able to tell that you are losing height and move the mouse backwards slightly to raise the nose. After a while you will notice that when flying level the nose of the aircraft is always in the same position relative to the horizon. This is the aircraft attitude for level flight. By putting the nose in that same position, you will achieve almost level flight without having to reference the instruments. From there you can fine-tune your altitude.

Beware: an altimeter does not automatically show the absolute altitude above sea level. You must adjust for the local air pressure. The little black knob on the lower left side of the altimeter allows you to adjust the altimeter. Start FlightGear and stay on the ground. Click (in normal mouse mode) inside the black knob and drag the mouse left or right. A drag toward the left makes the altimeter decrease in indicated altitude. Dragging toward the right will result in an increase in indicated altitude. An even more convenient method is to use the mouse wheel on the black altimeter knob. Use that little knob to set the altimeter to the field elevation of your airport. The principle is that you use the knob when you know your present altitude. If you know you are at 1100 ft altitude, tune in 1100 ft on the altimeter. Type **Ctrl-c** to see the

Figure 8.15: The altimeter

knob highlighted so that you understand the allowable click region.

To make settings the altimeter easier, airports advertise their altitude in various ways. They may provide a radio service, called ATIS (Automatic Terminal Information Service) in the USA, to broadcast the current air pressure at sea level. This is expressed in inches of mercury. The altimeter contains a small scale inside which is calibrated in this way. You can set your altimeter using this scale. Alternatively, if you are on the ground and know the altitude of the airport, you can simply adjust your altimeter until it displays the correct altitude.

Note that there is an important difference between "altitude above mean sea level" and "altitude above ground level". If you fly near Mount Everest at an altitude of 24 000 ft above Mean Sea Level (MSL), your altitude Above Ground Level (AGL) will be much less. Knowing the altitude of the ground around you is obviously useful.

8.4 Basic Turns

While, if you had enough fuel, you could return to the same airport by flying straight head for thousands of miles, being able to change direction will make your flying more enjoyable and useful.

Once you are able to fly more or less straight, it is time to learn to turn. The principle is simple:

- When the airplane is banked to the left, it turns to the left.
- When banked to the right, it turns to the right.

To turn, you do not need high levels of bank. 20° is more than enough for a safe and steady turn. Besides, passengers find excessive bank angles very disheartening. They don't like it!

The turn coordinator does NOT represent bank angle.

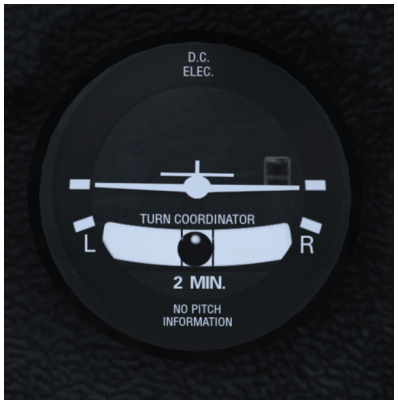

Figure 8.16: Turn coordinator

The inquisitive pilot may be interested in the fact that the bank angle required to establish a 3°, standard rate turn varies with True Air Speed (TAS). You could calculate an approximation with the formula:

$$\text{angle} \approx \frac{\text{TAS(knots)}}{10} + 7$$

TAS (knots)	formula	angle
80	$8 + 7$	15°
100	$10 + 7$	17°
120	$12 + 7$	19°

The turn coordinator represents Rate of Turn. The standard rate of turn is defined as 3° of travel per second. Why is this important? Frankly, because it can save your life! If you even find yourself suddenly in poor visibility, your smart reaction is to turn around 180°. In poor visibility, how are you suppose to know when you have turned halfway around? Hopefully, the standard rate of turn is now making sense to you. If you perform a standard rate turn, covering 3° of the turn every second, then a 180° turn will complete in one minute. After two minutes, you would have turned in a full circle and returned back to where you started your turn. This explains the "2 minute" placard you often see printed on the face of the instrument.

With this knowledge, let us explore how to properly utilize the turn coordinator instrument. As you bank the aircraft to the right, notice how the airplane depicted on the instrument lowers its right wing. Likewise, banking the aircraft to the left is duplicated on the face of the turn coordinator. When the wingtip aligns with the tickmark along the rim, at that point nuetralize the control yoke to maintain that angle. Glance quickly at the clock and roll out of the turn one minute later. I know you were curious as to why airplanes had clocks.

Try the following: keep the airplane banked in a standard rate turn for a few minutes and keep your eyes outside the aircraft. You will see the same ground features appear again and again, every 120 seconds. This demostrates that you need 120 seconds to make a 360° turn (or 60 seconds for a 180° turn.) This is particularly useful when navigating. Whatever speed the

airplane is flying, if you hold a bank at the standard rate you always need 60 seconds to make a 180° turn in the Cessna 172P (or any other aircraft).

So, by banking the airplane to the left or to the right, you make it turn to the left or to the right. Keeping the airplane level, in relation to the horizon, establishes you into a straight and level flight path.

The little ball, at the bottom of the turn indicator, shows the sideways forces acting upon the aircraft. In real life you would feel these forces as you turn. However, it is not possible to simulate these, so you must simply keep an "eye on the ball". Think of the ball as representing the position of the tail of the airplane. If you turn neatly (a.k.a. a coordinated turn), the ball will remain centered. If the ball is pushed, say rightwards, this means that you the pilot too are pushed rightwards. During a coordinated turn, even a strong turn, the plane occupants do not endure a sideways force. They are only pushed a little harder into their seats by the centrifugal force. To maintain coordination, simply "step on the ball". When you see the ball deflected toward the right, push on the right pedal to return the ball to the center. Likewise, the left pedal will return the ball to the center when deflected to the left.

By experimenting you will notice you can make much steeper turns by banking the airplane to high angles and pulling back on the yoke. Turns at over 60° bank angle are the realm of aerobatics and military flying, and dangerous to aircraft such as the Cessna.

8.5 Taxiing on the ground

Figure 8.17: Tachometer

While FlightGear is able to start you off conveniently lined up on the runway, you may be wondering how to get your aircraft from its hangar, along the taxiways, to the runway. This is taxiing.

Figure 8.17 shows the tachometer instrument. It displays how fast the engine is turning in hundreds of Revolutions Per Minute (RPM).

Tap the **PageUp** key a few times, until the tachometer is showing 1,000 RPM (as shown above). If required tap the **PageDown** key to decrease the engine speed.

At roughly 1,000 RPM, the airplane will move forward on the runway, but it will not accelerate and take off.

Tap the "**.**" key (**Shift-;** on Azerty keyboards). The airplane will make a sharp turn to the right. If you keep the "**.**" key down the airplane will halt. When you tap the "**.**" key, you are activating the brake on the right wheel of the airplane.

To activate the brake on the left wheel, use the "**,**" key.

The "**,**" and "**.**" keys simulate two brake pedals located at your feet on a real airplane. Using the throttle and the brake pedals you can control the speed of the aircraft and cause it to turn on the ground.

The brakes can be very useful when taxiing slowly on the ramp and taxiways. You can also steer the nose-wheel of the aircraft. In a real airplane this is done by pushing the rudder pedals with your feet. You push with your feet on the side you want to turn toward. If you don't have real rudder pedals, there are two ways to control the virtual rudder pedals:

- Using the keypad **0** and **Enter** keys. If you type the keypad **Enter** key say seven times, you will see the airplane firmly turns to the right and stays turning that way. Type the keypad **0** key seven times to get the airplane back rolling (almost) straight.
- Using the mouse. While the mouse is in yoke control mode (+ pointer), if you hold the left mouse button down, the mouse controls the rudder pedals instead of the yoke. The rudder pedals are connected to both the rudder and nose-wheel. This method is much more precise.

Start the simulator, Type **v** or **V** to view the airplane from the outside and keep **x** down a couple of seconds to zoom in on the airplane. Look at the front wheel and keep keypad **0** down. Then keep keypad **Enter** down. See the front wheel turn. Press **Tab** to get in yoke control mode (+ pointer). Keep the left mouse button down to get in rudder control mode and move the mouse to the left and to the right. Note that the rudder, that big vertical control surface at the rear of the plane, moves together with the front wheel.

I tend to control the rudder pedals using the mouse while the front wheel is on the ground and use the keypad **0** and **Enter** keys once it has lifted off. In other words: I keep the left mouse button down while the front wheel is on the ground. This allows for a precise and easy rudder control on the ground. Then I simply release the left mouse button once the front wheel lifts off the surface.

8.5.1 Airspeed

Just like driving a car, it is good to know how fast you are traveling. The aviation equivalent of a speedometer is the Air Speed Indicator (ASI), calibrated to nautical miles per hour (knots).

One nautical mile is the distance covered by 1 arc minute of latitude (1852 meters.)
One knot is equivalent to the speed required to travel one nautical mile per hour.
(60 knots = 60 nm/hour)

A knot is 1.852 km/h. So, if you want to have a rough idea of your speed in flight expressed in km/h, multiply the knots displayed by 2. A knot is 1.15115 miles per hour, so very roughly,

Figure 8.18: Airspeed Indicator

1 knot is 1 mph. Note that some aircraft ASIs (in particular the Piper J3 Cub) display mph instead of knots.

The airspeed indicator displays the speed of the aircraft compared to the surrounding air, not the speed compared to the ground like a car speedometer does. If the plane is halted on the ground and there is a 10 kt wind blowing from straight ahead, the airspeed indicator will display 10 knots airspeed, although the plane will not be moving relative to the ground.

When the airplane rolls over the runway at more than 40 knots, you must prevent the front wheel from touching the ground. The nosewheel is not designed for high speeds and in real life would shimmy and wear out.

During take off, once over 40 knots you can make the front wheel leave the ground by pulling back gently on the control yoke. Don't turn sharply at high speed on the ground. Doing so may cause the aircraft to tip over.

Figure 8.19 shows the front wheel slightly lifted. Don't overdo this. Keep the airplane's white nose cover well below the horizon. You just need to lift the plane's nose very slightly.

Question: if the front wheel no longer touches the runway, how do you steer the airplane? Answer: still using the rudder pedals. As mentioned above, the rudder pedals are linked to both the nose-wheel and the tail rudder, that big vertical moving part at the tail of the plane shown in figure 8.20. At airspeeds above 40 knots, the rudder has enough airflow over it to steer the airplane.

Note the front wheel and the tail rudder don't make the airplane turn at exactly the same rate. So when the rudder takes over the front wheel, you must adapt the rudder pedals angle. That means fast typing keypad **0** and keypad **Enter** (or hold the left mouse button down and tightly control the rudder with the mouse.)

Once you've become familiar with the nose-wheel and rudder, you can use these new controls to keep the airplane straight on the runway during take-off.

Say the airplane is heading too much to the right. You type keypad **0** a few times to make it turn back to the left. Do not wait till the aircraft has straightened up completely. Type keypad **Enter** before the aircraft reaches the direction you wish to travel. Otherwise you will find that you will over-correct and have to turn back again. If you use the mouse, such corrections

Figure 8.19: The front wheel slightly lifted

Figure 8.20: The tail rudder

are much easier and more precise.

To summarise: two methods exist to steer the airplane on the ground: the differential brakes on the main wheels and the rudder pedals. This control redundancy is very common in aviation. If one method fails, you still have another method available to perform the task.

You may be wondering why the aircraft drifts to the left when it rolls on the ground, forcing you to compensate with a little push on the right rudder pedal? The main reason is engine torque. As the propeller rotates clockwise, an opposite force is rolling the aircraft counter-clockwise. This transfers a force onto the left main gear and tire. This force results is a slight increase in drag and a slightly smaller tire radius. A secondary reason is the flow of air produced by the propeller. It blows along the airplane body, but also corkscrews around the airplane fuselage. The upper part of that slight vortex pushes the vertical tail to the right. This causes the front of the aircraft to yaw a bit to the left.

You can center all yoke and rudder controls by typing **5** on the keypad. This is a good preflight precaution. Sometimes it can "save your life" in flight if you find yourself with controls all over the place!

8.6 Advanced Turns

As with turning on the ground, there are two methods of turning in the air. You can use the wing ailerons (steered by the yoke/mouse) as described above or you can use the tail rudder (steered by the rudder pedals/the keypad keys **0** and **Enter**).

Why two ways? Partially for redundancy, but mainly because they are complementary. The main effect of the rudder is yaw (rotation around the vertical axis), while the main effect of the ailerons is roll (rotation around the longitudonal axis.)

- When flying close to the ground, it is better not to bank the airplane in order to turn. The rudder is used more instead. Acting on the rudder pedals allows you to turn the airplane without excessive banking.
- When the plane is just above the runway, the two side wheels need to be at the same height above the runway for landing. That means the wings must be level with the horizon. The plane is not allowed to bank. You keep the plane wings level with the horizon by using the yoke/mouse/ailerons. Note this does not need to be perfect. A bank of a few degrees is harmless.
- In flight, especially at high speed, the rudder is an in-efficient way to turn the aircraft:
  - It causes the airplane to present its flank to the airstream, increasing drag.
  - The airplane turns very slowly.
  - You will lack control while turning.
  - At high flight speed the centrifugal force will be disturbing or even dangerous.

  Using the yoke/mouse/ailerons allows for efficient, fast, reliable and comfortable turns.

- The rudder can be vital when the wings are stalled. Indeed, during a stall the wing ailerons become less effective or even useless. (Note that some airplanes can go in a very dangerous stall if you overdo the rudder control at low speed.)

When you turn in flight, using the ailerons, you still need the rudder a little bit. You add a little bit of rudder. This allows you to compensate for the adverse yaw created when you roll using the ailerons. In a real aircraft, you can feel this sideways motion. In the simulator, you can check this visually on the turn coordinator. In figure 8.21, the little ball is pushed rightwards during a strong turn to the left using the ailerons. That means you the pilot endure a rightwards force too. You can compensate this by pushing the right rudder pedal (type the keypad **`Enter`** key a few times.) In normal flight, you should use the rudder to keep the little ball centered.

Figure 8.21: Turn coordinator

So, in normal flight use the ailerons to turn, while close to the ground at low speed use the rudder. However, one method never completely cancels out the other. You still need the rudder at high altitudes and speeds. Reciprocally you have to use the ailerons a little bit when close to the ground, to keep the wings level with the horizon.

Even when taxiing, you should use the ailerons. Otherwise, strong winds can blow the aircraft onto its side. To counteract this, your should turn the ailerons into the wind. This raises the aileron in the wind, helping to keep the wing down.

You should avoid making quick and agressive movements of the rudder. On the ground at high speed this can make the airplane turn too sharply. In flight at low speed it can cause a very dangerous type of stall. In flight at high speed it can cause all kinds of aerodynamic and physical discomfort. Instead, make gentle movements of the rudder.

I recommend you practise turning with the rudder in flight. Fly at a low speed of about 70 knots. Try to keep the altitude stable by increasing and decreasing the engine power. Use the rudder to turn towards a ground feature and maintain a heading, then turn the aircraft towards a new heading. See how the plane yaws. Learn to anticipate rudder control. Don't try to make steep turns. Use the yoke/ailerons to keep the wings level constantly.

8.7 A Bit of Wieheisterology

Wieheisterology comes from the German phrase "Wie heißt Er" – "What's that name". This section is about gauges, switches and controls of the aircraft. While in the simulator you can

take off and land a basic airplane with just the engine throttle and the yoke, but you will need all the controls to perform securely and efficiently.

8.7.1 Engine control

An airplane engine is designed for simplicity, reliability and efficiency. Rather than use advanced electronic ignition and fuel injection systems found in modern cars, they instead use older technology that doesn't rely on electrical power. That way, the plane can still fly even if it suffers complete electrical failure.

Magneto

On the bottom left, below the instrument panel, you will find the magneto switch and engine starter (see figure 8.22).

Figure 8.22: Magneto

To better view the magneto switch, click the place where the yoke column enters the cockpit to hide the yoke. You can also press **x** key several times to zoom out (**X** or **Ctrl-x** to zoom back in.)

You can move the switch with the **{** and **}** keys (use the **Alt Gr** key on Azerty keyboards).

You are probably aware that the fuel inside a car engine is ignited by electric sparks. Modern car engines use electronic ignition. An airplane engine uses a more old-fashioned (but more reliable) magneto ignition instead. For redundancy, it contains two such magnetos: the "left" one and the "right" one. When you change the magneto switch to OFF, both magnetos are switched off and the engine will not run. With the magneto switch on L you are using the left magneto. On R you are using the right magneto. On BOTH you use both. In flight you will use BOTH.

Given that you use both magnetos in flight, why have the switch? The reason is that during your preflight checks you will verify that each of the magnetos is working correctly. To do this, increase the RPM to about 1500 then switch the magneto switch to L and observe the tachometer. You should observe a slight drop in RPM. If the engine cuts out, the left magneto is broken. If you do not see an RPM drop, then the switch may be faulty, as both magnetos are still switched on. You can then perform the same test on the right magneto. Of course, in the simulator, the magnetos are unlikely to fail!

Should one of the two magnetos fail in flight, the other one will keep the engine running. The failure of one magneto is rare, the failure of both simultaneously is almost unheard of.

You may have typed **{** to shut the engine down. To start the engine again after doing so, type **}** three times in order to put the magneto switch on BOTH. Then use the starter motor by pressing the **s** for a few seconds, till the engine is started.

You can also turn the magneto switch with the mouse by simply clicking the left and middle mouse button. Click and hold near "START" to start the engine.

If you turn the switch to OFF, the engine noise stops. If you quickly turn the switch back to L, the engine starts again as the propeller is still turning. If you wait for the propeller to stop, placing the switch on L, R or BOTH won't start the engine. (Once the engine is halted, always place the magneto switch to OFF.)

Throttle

Figure 8.23: Throttle & mixture

You already know that you increase the engine power by pushing that throttle rod in (**PageUp** key). You decrease the power by pulling the control out (**PageDown** key). You can also use the mouse wheel while its cursor is on the throttle, or you can hold down the left mouse button and drag the mouse.

What does "increase the power" actually mean? Does it mean you increase the amount of fuel delivered to the engine? Yes, but this is not enough to fully understand what you are doing. You need to be aware that the engine is also fed with a huge amount of air. The engine cylinders burn a mixture of fuel and air. Fuel alone wouldn't burn. Only a mixture of fuel and air can detonate and move the engine pistons. So when you push the throttle in, you increase both the fuel and the air fed to the engine.

Mixture

The amount of air compared to the amount of fuel is critical. The proportion of the two has to be tuned closely. This is the purpose of the mixture lever. Figure 8.23 displays the mixture lever (in red color).

When the mixture lever is fully pushed in, you feed the engine with an lots of fuel and little air. This is known as a "rich" mixture. When the lever is pulled out completely, there is an excess of air, known as a "lean" mixture. The correct position to produce maximum power is in between these two extremes, usually quite close to fully pushed in.

When you start the engine and when you take off, you need a fuel-rich mixture. That means the mixture lever should be pushed in (**m** key). A fuel-rich mixture allows the engine to start easily. It also makes the engine a little more reliable. The drawback is that a part of the fuel is not burned inside the engine. It is simply wasted and pushed out the exhaust. This makes the engine more polluting, it decreases the energy the engine can deliver and it slowly degrades the engine by causing deposits of residues inside the cylinders.

Once in normal flight, you have to pull the mixture lever a little, to get a more optimal mixture. Check this out by doing the following. Start the simulator. Put the parking brakes on with key **B**. Push the throttle in to its maximum. The engine RPM should now be close to the maximum. Slowly pull on the mixture lever (using the mouse in normal pointer mode or **M** key.) You will see the RPM increases a little. You get more power, without increasing the fuel intake. You waste no fuel and it pollutes less. If you continue to pull the mixture lever, the RPM will decrease back away, because now there is too much air. The excess of air slows the explosions down inside the cylinders and decreases the explosion temperature, hence the thermodynamic yield decreases. You have to tune in the optimal mixture. For thermodynamic reasons, the best mixture isn't exactly at maximum power – it is better for the engine to be running very slight richer or leaner than maximum power. This also avoids the possibility of the fuel detonating explosively damaging the engine. You can find the maximum power point by the fact you get the highest RPM. (Another method is to check the engine exhaust temperature. Roughly, this is the point at which you get the highest temperature.)

The mixture control allows you to burn less fuel for the same speed and distance, and therefore fly farther and pollute less. However, if you mis-manage it, it can cause serious problems. Suppose you go flying at high altitude and pull out the mixture lever accordingly. At high altitude there is less oxygen available so the correct mixture will be quite lean – i.e. with little fuel being used. Then you descend back in order to land. If you forget to push the mixture lever in as you descend, the fuel/air mixture will become far too lean and the engine will simply halt.

When landing, you have to tune back in a mixture that is a little too rich in fuel. This means pushing the mixture lever in. That way the engine becomes a little more reliable and will be better adapted to a decrease in altitude.

I wrote above that placing the magneto on OFF is not the right way to stop the engine. The right method is to pull the mixture lever. First pull the throttle out completely, to get the engine to minimum power and fuel consumption. Then pull the mixture lever, till the engine stops because the mixture contains too much air. This ensures the engine doesn't get choked by waste fuel residues. Finally, turn the magneto switch to OFF to ensure the engine won't start again accidentally.

An important warning: you may think the RPM indicator reflects the engine power. Wrong. Two things make the RPM increase: the engine power and the airplane speed. To check this, fly to a given altitude then pull the engine power to minimum. Try out diving to the ground then rising back to altitude. You will see the RPM varies significantly as does your airspeed. It rises while diving and decreases while climbing.

One pitfall of this is when you intend to tune the engine power in for landing. Suppose you're descending towards the airport, flying fast. You know the ideal RPM for landing is around 1,900 RPM. So you pull the throttle till you get 1,900 RPM. You think you tuned in the

appropriate RPM. You think you shouldn't bother any more about it. But when you level off, the plane's speed starts to decrease, along with the RPM. A few minutes later, you get the low flight speed you wanted. You don't see the RPM is now far too slow. You will either lose altitude or stall. Or both. Be cautious with the throttle and with the RPM indicator. Either pull on the throttle more steadily or be mentally prepared to push it back in quickly.

8.7.2 Wings and speed

Say you are flying with full engine power. Dropping the nose a little makes you lose altitude and raising the nose a little makes you gain altitude. You may think this is quite straightforward. The plane travels in the direction it is heading; the direction the propeller is heading. This is not the best way to think about it. This model would be fine for a rocket, but not for an airplane. A rocket is lifted by its engine, while a plane is lifted by its wings. That's a huge difference.

Get a big rigid square of cardboard, hold it horizontally in your hand with your arm stretched out and make it do fast horizontal movements while rotating your torso. When the cardboard moves flat through the air, it experiences no lift force. If you twist your arm slightly to give the cardboard a slight upward angle, you will feel it tends to lift in the air. There is an upward force acting on the cardboard. That's the way a wing holds a plane in the air. The wings have a slight upward angle and lift the airplane. The more angle you give the cardboard, the more lift force. (Till you give it too steep an angle. Then you will rather feel a brake force. The cardboard is "stalling" (see below).)

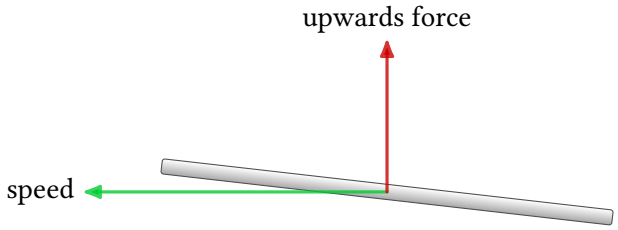

Figure 8.24: Lift

- When you pull the yoke, the airplane's nose rises up. Hence the wings travel through the air at a steeper angle. Hence the lift force on the wings is stronger. Hence the plane rises in the air.
- When you push the yoke, the airplane's nose dives. Hence the wings travel through the air with less angle. Hence the lift force on the wings decreases. Hence the plane descends.

What matters is the angle the wings travel through the air. This is the angle of attack.

I wrote above that when the wings travel through the air with no angle of attack, they don't produce lift. This is false. It would be true if the wings were a flat plate like the cardboard. But they aren't. The wings are a slightly curved airfoil. This makes them create lift even when traveling through the air at no angle of attack. Actually, even with a little negative angle of

attack, they still create a lift force. At high speed, the airplane flies with the wings slightly angled towards the ground!

Figure 8.25: Airfoil

The angle at which the wings travel through the air matters. Something else matters too: the speed. Take the cardboard again in your hand. Hold it with a given slight angle and don't change that angle. Move it at different speeds through the air. The faster you move the cardboard, the more upward force it experiences.

- When you increase the engine power, the plane increases speed, the lift force on the wings increases and the plane gains altitude.
- When you decrease the engine power, the plane decreases speed, the lift force on the wings decreases and the plane loses altitude.

To make things a little more complicated: when rising in the air, the airplane tends to lose speed. When descending, it tends to gain speed.

That's all a matter of compromises. If you want to fly at a constant altitude and at a given speed, you will have to tune both the engine power and the yoke/elevator (or better: the trim—see section 8.7.5), till you get what you want. If you want to descend yet keep the same speed, you have to push the yoke a little and decrease the engine power. And so on. You constantly have to tune both the engine power and the yoke. However, during a normal flight you can simplify this by simply choosing a comfortable engine power level then relying on the yoke and trim for altitude.

A very interesting exercise you can perform with the simulator is to fly straight with full engine power. Get maximum speed while keeping in horizontal flight. Then decrease the engine power to minimum. Pull steadily on the yoke to keep the plane at constant altitude. The plane slows down steadily, meanwhile you have pull more and more on the yoke to stay level. Since the speed decreases the lift from the wing will decrease, but you compensate the loss of speed by increasing the wing angle of attack. This proves the plane does not necessarily travel in the direction its nose is heading. In this experiment we make the nose rise in order to stay at constant altitude. Once the plane is flying very slowly, and the nose is very high, you may hear a siren yell. That's the stall warning (see below.) This indicates that the angle of attack is too high for the airfoil to produce lift. The wings are no longer producing lift and the plane quickly loses altitude. The only way to correct this is push the yoke forwards to reduce the angle of attack, making the nose drop, then apply full power to gain speed and finally bring the yoke carefully back to level flight.

Question: is it better to control the airplane's speed and altitude with the yoke or with the throttle? Answer: it depends on what exactly you intent to do and on the situation you are in. In normal flight, as said above, you tend to set a comfortable engine power level, forget about it and rely on the yoke and trim. During take off and landing the procedures are quite strict

about the use of yoke and throttle. You do the opposite: control the speed with the yoke and trim, control the altitude and descent speed with the engine throttle. This will be discussed further below.

8.7.3 The flaps

The flaps are situated at the rear of the wings, either side of the aircraft fuselage. You deploy the flaps and retract them back in by using the flaps control lever (see figure 8.26).

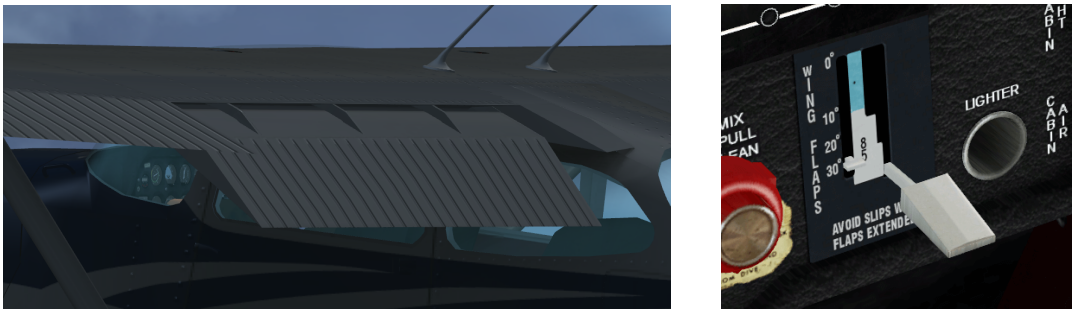

Figure 8.26: Flaps & flaps lever

You can either click on it with the mouse or use the **[** and **]** keys. Key **[** to retract the flaps one step, **]** to deploy them one step at a time. Type **v** to view the plane from the outside and try out **[** and **]**. (On the schematic instrument panel, the flaps lever is located at the lower right.)

In the Cessna 172P, there are four flaps settings:

- 0° – for normal flight and normal take off.
- 10° – for short field take off, when you want to gain altitude while flying slowly. Or during the first stage of an approach to land.
- 20° – to slow the aircraft and lose altitude quickly, for example when descending towards the runway to land.
- 30° – to lose altitude even more quickly.

The flaps are somewhat delicate. Do not deploy the first step of flaps above 110 knots. Do not deploy the second or third stage of flaps above 85 knots.

The flaps create large amounts of drag on the aircraft and brake the plane at high speed. This is one more reason not to forget to pull the flaps back in once you fly above 85 or 110 knots.

To check the flaps position visually, either use the mouse view mode to look at the back of the wing, or type **Shift-**→ to shift the view to the right and then quickly **Shift-**↑ to get back to front view.

Flaps increase wing lift by altering the shape of the airfoil. The wing lifts more at a given speed with the first stage of flaps set. Hence you will get in the air a little sooner during take off. It also has the effect to make the plane fly with a lower nose attitude. This is useful as it provides a better view of the runway when taking off or landing.

The flaps also increase drag on the aircraft. The second and third stage of flaps produce much more drag than lift, so they are used to brake the plane. This is particularly useful when landing, because the airplane glides very well. If you cut down the engine power completely, the plane will descend, yet but too slowly. You need to deploy two or three flaps steps in order to brake and really descend towards the ground.

The fact that the flaps brake during landing makes you need more engine power during the landing. This can seem odd. Why not simply throttle the engine down to minimum and use less flaps steps? The answer is that it is better to have a strongly braking plane and lots of engine power, as the plane reacts faster to your commands. Should the engine fail, then just retract flaps as needed and glide to the runway.

What can you do if you have full flaps extended and need to increase your rate of descent further? Slowly push the rudder pedals on one side. This will make the plane present its flank to the air stream and brake. Compensate the turning by using the ailerons (yoke). This is known as side-slipping, and is a very effective way to lose height progressively as it is easy to stop at any point.

8.7.4 The stall

An aircraft relies on the smooth flow of air over the surface of the wing to produce lift. However, if the wing is at too high an angle of attack, this flow is broken, and the wing no-longer produces lift. With no lift, the aircraft cannot fly, and quickly drops back to earth. This is known as a stall.

A stall is an emergency situation. While it can happen at any speed, it commonly occurs in slow flight. A given aircraft has a specific stall speed, at which no angle of attack can produce enough lift. You should always keep your aircraft well above the stall speed. To help, aircraft are equipped with stall sirens that sound when the angle of attack is approached.

If you encounter a stall, the remedial action is to immediately drop the nose, and apply full power, bringing the nose level when flying speed has been attained again. However, doing so will cause the aircraft to lose altitude, which you may not have to spare when landing or taking off!

A spin occurs when one wing stalls before the other, which can occur in a steep turn at low speed. As one wing is still flying, the aircraft turns around the stalled wing, spinning tighter and tighter. To get out of a spin, you need to apply rudder to straighten out the spin into a normal stall, then recover as above.

Aircraft like the Cessna 172 and Piper Cub, have benign stalls, and are unlikely to enter a spin. High performance jets, such as the F16 have much more agressive stalls, and can easily enter a spin.

To practise this in the simulator, do the following:

- Fly at constant altitude and attitude.
- Reduce engine power, raising the nose to avoid entering a descent.
- Continue to reduce power until the stall begins.
- Try to control the plane while it stalls and descends to the ground.

- Keep the yoke pulled to the maximum and the plane in a steady attitude, the wings parallel with the horizon. Try to change direction.
- Recover by lowering the nose, applying full power, and correcting the attitude once flying speed has been regained.

You can also experiment with stalls with different flap settings, and high speed stalls by making abrupt attitude changes.

Experiment with different aircraft. Compared with the Cessna 172 the Cessna Citation jet, stalls much more agressively and with little warning.

8.7.5 The trim

Figure 8.27: Trim

The trim is the dark big vertical wheel with gray dots located at the middle below the instrument panel (see figure 8.27).

In FlightGear, the keys **Home** and **End** adjust the trim. **Home** rolls the wheel upwards while the **End** rolls the wheel downwards. You can also left-click the trimmer wheel to rotate it up, or middle-click to rotate it down.

In first approximation, the trim does the same as the yoke: it acts on the elevator. Turning the trim wheel downwards is the same as pulling on the yoke. Yet there is a key difference between the trim and the yoke. The trim remains in position after you make a change, while the yoke only continues to affect the elevator while you apply pressure and returns the elevator to neutral when you release it.

During cruise flight, the required elevator position to keep the aircraft at constand altitude will not be completely neutral – it will vary depending on the air outside the aircraft, the current fuel level, and the payload. Obviously, holding the yoke continually to retain a constant attitude would quickly become tiring. By using the trim to "trim out" the elevator force required for cruise flight, the yoke can be kept neutral.

During take off the trim should be neutral. Otherwise you may find that it either refuses to take-off with the normal level of yoke control, or takes off too quickly.

During landing, try to get the yoke/mouse/elevator towards neutral position by tuning the trim. This makes making small adjustments to your attitude and position easier. On the Cessna 172p this means trim on neutral. On the Cherokee Warrior II this means the trim a little "pulled".

The trim wheel movement is much slower than the yoke, allowing for delicate changes in trim. Be patient.

8.7.6 What direction am I flying?

Knowing the direction you are going is obviously a good idea. There are three basic ways to determine the direction you are flying:

- Look through the windows. If you are flying regularly from the same airport, you will learn to recognize the ground features such as roads, hills, bridges, cities, forests. In a simulator, you only have a narrow view of the virtual outside world. Several ways exist to allow you to pan your virtual head inside the airplane:
  - Use **`Shift`** and the four arrow keys to look to the front, rear, left and right.
  - Use **`Shift`** and the keypad keys to look in the four directions mentioned above and in four diagonal directions in-between.
  - Hold down the right mouse button in normal or yoke mode and move the mouse to change the view direction.
  - Use the mouse in view mode (**`Tab`**, ↔). This allows you to look in every direction, including up and down. Click the left mouse button to bring the view back to straight ahead.
- The magnetic compass. This is located above the instrument panel. The compass is simple, but is affected by the acceleration of the aircraft, and magnetic abnormalities on the ground. Also, the compass points towards magnetic north rather than true north. This deviation varies depending on your location.
- The directional gyro or "heading indicator" (see figure 8.28). The gyro is powered by a vacuum system. The gyro is set to match the magnetic compass, and is not affected by magnetic issues, or aircraft movement. However, due to gyroscopic precession and friction in the instrument, over time it drifts and must be reset by reference to the magnetic compass on ocassion. To reset the HI, during cruise flight, use the black knob on the bottom left of the instrument (normal mouse pointer mode, use the mouse wheel, in addition, you can hold **`Shift`** key to move faster, **`Ctrl-c`** to highlight clickable areas.) Remember to adjust the gyro heading indicator only when flying straight, at a constant speed. Then you will be sure that the magnetic compass will not have any distortions. (The red knob, bottom right, is used to tell the autopilot what direction you wish to fly (**HDG** = "heading".))

Figure 8.28: Magnetic compass & heading indicator

8.7.7 A look around the panel

Finally, let's have a look at the instrument panel, combining the instruments described above with some new ones.

The six-pack

Figure 8.29: Attitude indicator

Let us start with the most important instruments any simulator pilot must know, these are known as the "holy six" or the "six-pack". In the center of the instrument panel (fig. 5.1), in the upper row, you will find the artificial horizon (attitude indicator) displaying pitch and bank of your plane. It has pitch marks as well as bank marks at 10, 20, 30, 60, and 90 degrees (see figure 8.29).

Left of the artificial horizon, you'll see the airspeed indicator shown in figure 8.18. Not only does it provide a speed indication in knots but also several arcs showing characteristic velocity

ranges you have to consider. At first, there is a green arc indicating the normal operating range of speed with the flaps fully retracted. The white arc indicates the range of speed with flaps in action. The yellow arc shows a range which should only be used in smooth air. The upper end of it has a red radial indicating the speed you must never exceeded, unless you want to break up the plane in mid-flights...

Below the airspeed indicator, you can find the turn indicator (cf. figure 8.16). The airplane in the middle indicates the roll of your plane. If the left or right wing of the plane is aligned with one of the marks, this would indicate a standard turn, i.e. a turn of 360 degrees in exactly two minutes.

Below the plane, still in the turn indicator, is the inclinometer. It indicates whether the rudder and ailerons are co-ordinated. During turns, you always have to operate aileron and rudder in such a way that the ball in the tube remains centered; otherwise the plane is skidding. A simple rule says: "Step on the ball", i.e. step onto the left rudder pedal when the ball is on the left-hand side.

If you don't have pedals or lack the experience to handle the proper ratio between aileron and rudder automatically, you can start FlightGear with the option:

```
--enable-auto-coordination
```

To the right-hand side of the artificial horizon, you will find the altimeter showing the height above sea level (not ground!) in hundreds of feet (cf. figure 8.15). Below the altimeter is the VSI (Vertical Speed Indicator) (see figure 8.30) indicating the rate of climbing or sinking of your plane in hundreds of feet per minute. While you may find it more convenient to use than the altimeter in certain cases, keep in mind that its display usually has a certain time-lag.

Figure 8.30: Vertical Speed Indicator

Below the turn coordinator is the propellor tachometer, or RPM (Revolutions Per Minute) indicator, which displays the rotations per minute in hundreds (cf. figure 8.17). The green arc marks the optimum region for cruise flight.

The group of the main instruments further includes the gyro compass being situated below the artificial horizon. Besides this one, there is a magnetic compass sitting on top of the panel; both are shown in figure 8.28.

Four of these gauges being arranged in the from of a "T" are of special importance: The air speed indicator, the artificial horizon, the altimeter, and the compass should be scanned regularly during flight.

Supplementary Instruments

Beside the six-pack, there are several supplementary instruments. To the very left you will find the clock, obviously being an important tool for instance for determining turn rates. Below the clock there are several smaller gauges displaying the technical state of your engine. Certainly the most important of them is the fuel indicator – as any pilot should know.

The ignition switch is situated in the lower left corner of the panel (cf. Fig. 8.22). It has five positions: "OFF", "L", "R", "BOTH", and "START". The first one is obvious. "L" and "R" do not refer to two engines (as the Cessna 172 only has one) but the two magnetos, providing redundancy in the case of a failure. The two switch positions can be used for test purposes during preflight. During normal flight the switch should point on "BOTH". The extreme right position is for starting the engine using a battery-powered starter (operated with the **s** key).

The handle below the yoke is the parking brake. In the vertical position, the parking brake is ON. The parking brake is operated with the **B** key.

Radios

The right hand side of the panel is occupied by the radio stack. Here you find two VOR receivers (NAV), an NDB receiver (ADF) and two communication radios (COMM1/2) as well as the AutoPilot (AP).

The communication radio is used for communication with air traffic facilities; it is just a usual radio transceiver working in a special frequency range. The frequency is displayed in the LEDs. Usually there are two COM transceivers; this way you can dial in the frequency of the next controller to contact while still being in contact with the previous one.

The COM radio can be used to listen to the current weather conditions at an airport, known as ATIS. To do this, simply dial in the ATIS frequency of the relevant airport. You can find this by selecting *AI* → *ATC Services in Range* from the menu, and selecting the 4-letter ICAO (International Civil Aviation Organization) code of a nearby airport.

Each COM radio has two frequencies configured – an "active" frequency which the pilot is transmitting and receiving on, and a "standby" frequency which may be changed. In this way, you can continue to listen on one frequency while tuning another one.

You can change the radio frequency using the mouse. For this purpose, click left/right to the circular knob below the corresponding number. The corresponding switch left to this knob can be used for toggling between the active/standby frequency.

Use of the autopilot and radio navigation equipement is covered in later tutorials. For the moment you can ignore these radio instruments as long as you are strictly flying according to VFR (Visual Flight Rules).

8.8 Let's Fly

By now you will be able to keep on runway while taking off by using the rudder and you're able to fly straight, descend, climb and make gentle turns. This section will describe a slightly more realistic approach to taking off and landing, and introduce some of the more subtle concepts you should be aware of.

8.8.1 A realistic take off

The following general rules apply during a normal take-off:

- The nose-wheel should be lifted from the runway at approximately 40 knots.
- Immediately after take-off, you should accelerate to 70 knots, to stay well above stall speed, in case of a gust of wind, or engine failure.
- Don't fly much above 75 knots to ensure you gain height as quickly as possible.
- Follow the runway heading until at 500 ft. This way, if you suffer an engine failure, you can easily land back on the runway you left.
- Don't over-fly buildings until at least 1000 ft.
- Close to the ground, turns should be gentle and well coordinated using the rudder.

So, you need to take off and rise in the air at a steady speed of around 75 knots. However, when you raise the nose slightly at 40 knots, the aircraft will probably take-off at around 55 knots. To accelerate quickly to 75 knots, lower the nose slightly immediately on take-off, then raise it once 75 knots has been achieved. You are using the yoke to control the speed of the aircraft.

Putting this all together with what you have learned previously, a normal take-off using the mouse will consist of the following:

1. Adjust the altimeter to the correct altitude, based on the airport altitude. For reference, KSFO is at sea level – 0 ft.
2. Check aileron and elevator are neutral by observing the yoke position.
3. Change the mouse to control mode by pressing **`Tab`**.
4. Hold the left mouse button down to control the rudder.
5. Apply full power (hold **`PgUp`** until the throttle is fully in.)
6. As the aircraft accelerates down the runway, keep it in the center by making small adjustments using the mouse.
7. As 40 kts is reached, release the left mouse button, and pull back slightly to raise the nose-wheel. You are now controlling the yoke with the mouse.

8. The aircraft will fly off the runway at approximately 55 knots.
9. Lower the nose slightly to accelerate to 70 knots.
10. Keep alignment with the runway.
11. Use the yoke to keep the ASI at 70 knots as you climb. If the airspeed is dropping, lower the nose. If the airspeed is increasing, raise the nose slightly.
12. Once you reach 500 ft, turn to your required heading, staying away from buildings until you are over 1000 ft.

8.8.2 Landing

The rules for landing are almost identical to that of take-off, but in reverse order:

- Close to the ground, turns should be gentle and well coordinated using the rudder.
- Stay above 500 ft until on final approach to the runway.
- Approach the runway at approximately 70 knots.
- Touch down on the two main wheels at 55 kts.
- Let the nosewheel touch down at 40 kts.

Landings are much easier if you have an aiming point picked out on the runway. By observing the aiming point, you can easily tell if you are descending too fast or too slowly. If the aiming point appears to move upwards, you are descending too fast.

Obviously, you need to be lined up with the runway. That means your flight direction has to match the middle line of the runway (see figure 8.31 (a)). In order to arrive at this, don't aim at the start of the runway (b). Rather aim at a fictitious point well in front of the runway (c). And begin to turn gently towards the runway well before you reach that fictitious point (d). Note the turns and bankings you make for these flight corrections are often very soft. You wouldn't even notice them on the turn coordinator. This is one example where you better rely on the outside horizon line than on the inside flight instruments.

A straight in landing using the mouse would consist of the following:

1. 1500 ft above the airport, and a couple of miles out, an in-line with the runway, reduce power to approximately 1500 RPM. This will slow you down somewhat and start a gradual descent.
2. Once below 115 knots, apply one step of flaps (**]**). This will increase lift, but also slow you down.
3. Re-trim the aircraft so you continue to descend.
4. At around 1000 ft, apply another step of flaps (**]**). This increase drag significantly, but also improve the view over the nose.

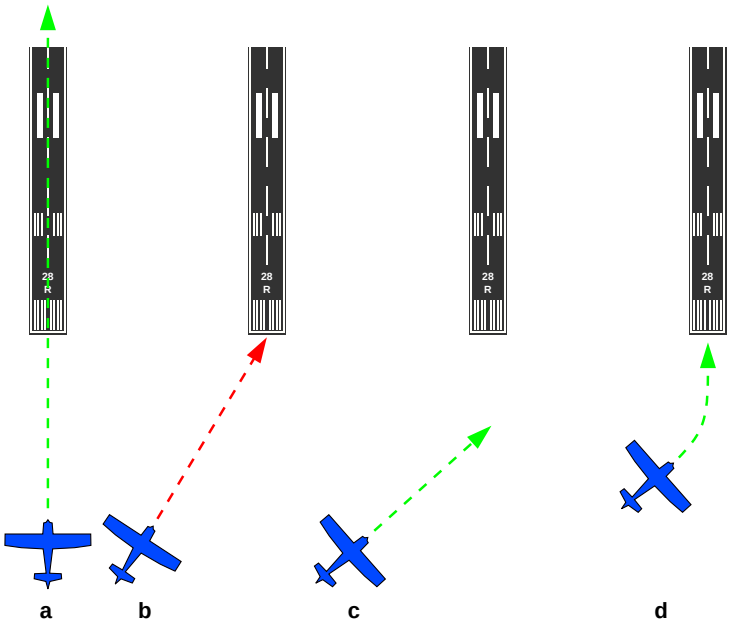

Figure 8.31: Alignment to the runway

5. Tune the speed using the elevator and trim: push the yoke if you are flying below 70 kt, pull the yoke if you are flying above 70 knots. If using a joystick, use the trim to relieve any pressure on the elevator.

6. Tune the altitude using the engine throttle. Add power if you are descending too fast, reduce power if you are too high. It is much easier to work out if you are too high or too low by observing the numbers on the runway. If they are moving up the screen, you are descending too fast – increase power. If they are moving down, you are too high and need to reduce power.

7. Make minor adjustments to heading to keep aligned with the runway.

8. At about 500 ft, apply the final step of flaps. (]). This increase drag significantly, so be prepared to increase power to keep your descent constant.

9. When you are just above the runway, reduce power to idle, and use the yoke to gently pull back the aircraft to horizontal. This is the "round-out" and should result in the aircraft flying level with the runway a couple of feet above the surface. Performing the round-out at the correct height is a difficult task to master. To make it easier, observe the horizon rather than getting fixated on the aiming point.

10. Keep the wings level using small inputs with the yoke. We want both wheels to touch down at the same time.

11. Continue pulling back on the yoke to lifting the nose slightly – this is the "flare". The main wheels should touch down at about 55 knots.

12. As you touch down, be ready to use the rudder to keep the aircraft straight (keypad **0** and keypad **Enter**)
13. Once you are below 40 knots, lower the nose-wheel to the ground.
14. Hold down the left mouse button to control the nosewheel/rudder using the mouse.
15. Once below 30 knots, use the brakes **b** to slow the aircraft further.

Once the plane is halted or at very low speed, you can release the **b** key and add a little engine power to taxi to the parking or hangar.

Figure 8.32: Landing

8.8.3 Engine Shutdown

To shut the engine down:

- Set the parking brake, type **B**.
- Engine throttle to minimum (hold **PageDown** down for a while.)
- Pull the mixture lever to halt the engine (mouse in normal pointer mode, click on the left of the red mixture lever to pull it out.)
- Rotate the magneto switch to OFF (a few hits on **{**).

8.8.4 Aborted Landing

You must be mentally prepared to abort landing any time the landing does not look good, or due to external factors such as:

- an order from the control tower.
- incorrect speed or landing angle when there is insufficient time to correct it.
- a strong gust blow of wind.
- birds flying over the runway.

To abort the landing, apply full power (hold **PgUp**), raise the nose to climb, and once you are climbing, retract the flaps (**[**).

Landing is much harder than taking off. Landing on a large runway, such as KSFO (San Francisco) is much easier than smaller runways such as KHAF (Half Moon Bay, about 10 miles to the south west of KSFO.)

To practise landings, use the command line below in a terminal window to start the simulator in flight and heading for the runway. The airplane is placed 5 miles ahead of the runway, at an altitude of **1500** feet and a speed of about 120 knots.

```
fgfs --offset-distance=5 --altitude=1500 --vc=120 --timeofday=noon
```

Approaching to land at 65 knots instead of 70 knots allows to use a much shorter runway length. However, this requires better control, particularly as it is much closer to the stall speed. It is quite different from landing at 70 knots.

8.9 Dealing with the Wind

Consider a hot air balloon. Think of it as being in the middle of a gigantic cube of air. The cube of air may move at high speed compared to the ground, but the balloon itself is completely static in the middle of the cube. Whatever the wind speed, persons aboard a hot air balloon experience not a breath of wind.

In the same way, an aircraft flies in the middle of a gigantic cube of air and flies relative to that air mass. The motion of the cube of air relative to the ground has no effect on the aircraft.

You, the pilot, on the contrary, are interested in the speed of the surrounding air compared to the ground. It can make you drift to the left or to the right. It can make you arrive at your destination much later or much sooner than planed.

When the wind blows in the same direction as you fly, the speed of the wind adds itself to the airspeed of the plane. Hence you move faster compared to the ground. You will arrive earlier at your destination.

When the wind blows in the opposite direction (towards the nose of the plane), the speed of the wind subtracts itself from the airspeed of the plane. Hence you move slower compared to the ground. You will arrive later at your destination and have more time to enjoy the landscape.

The two cases above are quite simple. More complex is when the wind blows towards the side of the airplane. Consider figure 8.33.

- On picture (a) there is no wind. The pilot wants to reach the green hill situated to the North. He heads for the hill, towards the North, and reaches the hill after a while. When there is no wind, you just head towards your destination and everything's fine.

- On picture (b), the pilot keeps heading to the North. Yet there is wind blowing from the left; from the West. The airplane drifts to the right and misses the hill.
- On picture (c), the pilot keeps heading towards the hill. This time he will arrive at the hill. Yet the plane flies a curved path. This makes the pilot lose time to get to the hill. Such a curved path is awful when you need to make precise navigation.
- Picture (d) shows the optimal way to get to the hill. The plane is directed to the left of the hill, slightly towards the West and the wind. That way it compensates for the wind and remains on a straight path towards the hill.

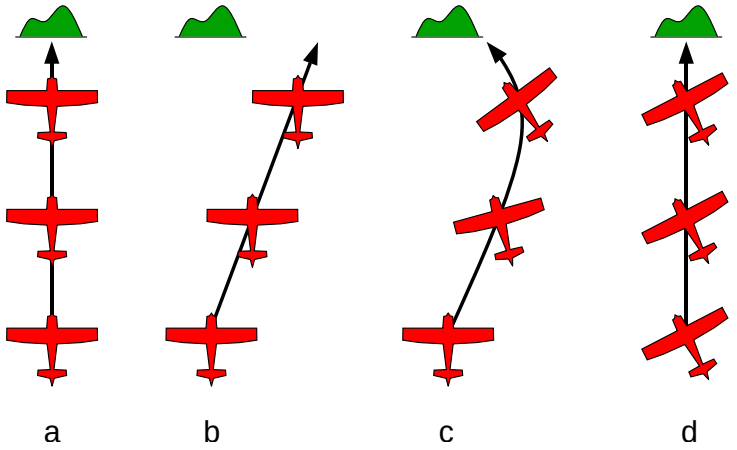

Figure 8.33: Side wind

How much to the left or to the right of the object must you head? At what angle? Serious pilots use geometry and trigonometric computations to calculate the correct angle. You need no computations at all to fly roughly straight. The trick is to choose an aiming point in the direction you wish to fly, then observe how it moves. You will become aware if you are drifting leftwards or rightwards. Then let your instinct slowly head the plane to the right or to the left to compensate the obvious drift. To begin with, you may need to think about what you are doing. Very soon this will become automatic, just like when you learned to fly straight. You will no more keep the plane headed towards the object. You will rather keep it flying towards the object.

The faster the flight airspeed compared to the wind speed, the less correction you will need.

8.9.1 Crosswind Take Off

Taking off when the wind is coming from the side is tricky. Airport designers avoid this by placing runways so that they face into the prevailing wind. Often airports have multiple runways, placed such that there will be a runway facing straight into wind as much of the time as possible.

Taking off with a wind blowing straight towards the nose of the aircraft makes life easier as it is the speed of the wing relative to the air that causes lift. When there is no wind, the

aircraft must accelerate to 55 knots to take off. However, if there is a 10 knot head-wind, the aircraft has an airspeed of 10 knots standing still and only has to accelerate to 45 knots relative to the ground to take off. This shortens take-off distances.

Just as a headwind shortens take-off, a tail-wind increases take-off length. Anything more than a knot or two makes a huge difference to take-off distance. As (most) runways can be flown from either end, you can easily take off from the other end of the runway and benefit from the headwing.

The main way to know the wind direction and speed is to go to the control tower or ask the control tower by radio. A necessary and complementary tool are the windsocks at both ends of the runway. They show the wind direction and speed. The longer and the stiffer the windsock, the more wind there is. The windsock on figure 8.34 shows an airspeed of 5 knots.

Figure 8.34: Windsock

Unfortunately, sometimes there isn't a runway facing the wind, and you have to take off when the wind is blowing from the side.

The technique is as for a normal take-off with two changes:

- During the take-off roll, the aircraft will try to "weather-cock" into wind. You must react by using the rudder to keep the aircraft running straight. You will have to apply the rudder at quite a strong angle to stay aligned with the runway. You will need to keep applying rudder throughout the take-off.
- As you take off, the aircraft will react to the rudder and try to turn. You will need to correct for this using the ailerons. Once the aircraft is in the air, you can reduce the rudder pressure and aileron, then correct for the wind, to keep aligned with the runway as described above.

8.9.2 Crosswind Landing

Landing in a crosswind is very similar to the take-off:

- Stay aligned with the runway by compensating for the crosswind.
- As you begin to round-out, use the yoke to straighten the aircraft so it is pointed down the runway. Apply rudder to stop the aircraft turning.
- The aircraft will land on one wheel first. Use the rudder to keep the aircraft pointed straight down the runway as the other wheel touches down.

The technique described here is the slip landing. Another crosswind landing technique is the crab landing.

8.9.3 Taxiing in the Wind

Under 10 knots wind the Cessna 172p seems not to need particular precautions when taxiing. Yet any sudden increase in wind speed can tilt it and tumble it over. So best apply the recommendations whenever there is wind.

To train taxiing on the ground when there is wind, configure the simulator for a strong wind, like 20 knots. Such a wind can tilt the plane and blow it away tumbling any moment. One single error during taxiing and the plane is lost.

Main rule is you must *push the yoke towards the wind*. This deserves some physical explanation:

- When the wind is blowing from 12 o'clock, this is quite logical. The yoke is pushed (towards 12 o'clock) and the elevator makes the tail rise a little. That's the most stable position to avoid the plane be tilted by the wind.
- When the wind comes from 10 o'clock, pushing the yoke towards 10 o'clock means that the elevator is almost neutral, while the left aileron is upward and the right aileron is downward. This pushes the left wing down and lifts the right wing. Again, that's the most stable position to avoid the plane be tilted by the wind.
- When the wind blows from 8 o'clock, you would think you should invert the position of the ailerons, to keep the left wing being pushed down. Hence you should push the yoke to 4 o'clock. Wrong! Keep pushing the yoke to 8 o'clock. The reason is the downward position of the aileron on the right wing makes it act like a slat. This increases the lift on the right wing and this is all we want. Symmetrically, the upward position of the left aileron decreases the lift of the left wing.
- When the wind comes from the rear, from 6 o'clock, the yoke is pulled (toward 6 o'clock). The upward position of the elevator tends to make the tail be pushed down. Once again this is the best. Strong wind can push the tail against the ground. This is impressive but the tail is conceived to withstand this.

If you want to move towards the wind, you will need more engine power. When the wind blows from the rear you may need no engine power at all. Always keep the engine power to the minimum needed.

Especially when turning, move very slowly. Make little changes at a time. Take your time and closely survey the yoke angle. Constantly keep it pushed towards the wind. Constantly try to reduce the engine power. Keep in mind using the brakes too firmly may shortly tilt the plane at an angle that allows the wind to tilt it and blow it away.

8.10 The autopilot

Figure 8.35: Autopilot

An AutoPilot (AP) is not an "intelligent" pilot. It just takes over simple tasks for the pilot. You still are the pilot aboard and have to keep aware of everything. Be prepared to shut the autopilot down as they often go wrong, both in real life, and in the simulator.

The autopilot is mounted to the right of the yoke.

Switch it on by pressing the **AP** button. The autopilot then controls the roll of the aircraft. It keeps the wings level with the horizon. This is displayed in figure 8.35 by the **ROL** marking. To switch the autopilot off press **AP** again.

If you press the **HDG** button the autopilot will try to keep the plane flying towards the direction tuned on the directional gyro by the red marking (see section 8.7.6.) **HDG** stands for "heading". Press again on the **HDG** button to get back to roll control mode (or **AP** to switch the autopilot off).

The buttons **ALT**, **UP** and **DN** are used to tell the autopilot either to control the vertical speed **VS** or the altitude **ALT**. For more advanced use of the autopilot, see the reference document for the autopilot modelled in the Cessna 172: Bendix King website.

8.11 What Next?

This tutorial has introduced you to the basics to flight in the Cessna 172. From here you can explore the many features that FlightGear has to offer.

Once you have mastered the content of this tutorial, you may want to look at the other tutorials in this Manual, covering flying to other airports, flying using instruments when clouds obscure the ground, and flying helicopters.

This tutorial has skipped over a number of topics that a real-life pilot would have to consider:

- How to follow real checklists.
- How to make emergency landing on very short fields, after and engine failure.
- How to navigate with regard to the laws of the air, charts, laws, radio beacons and weather conditions.
- How to create a flight plan and fly it accurately.
- How to place people, fuel and baggage in an airplane to get a correct center of gravity.
- How to deal with the control tower and with other airplanes.
- How to deal with several fuel tanks and their systems.
- How to deal with the failure of every possible part of the plane.

This tutorial has also not covered features of more advanced aircraft, including:

- retractable landing gear
- variable pitch propellers
- multiple engines
- jet engines.

8.12 Flying Other Aircraft

I cross-checked all the data about the Cessna 172p, a pilot friend verified I did not write too much rubbish and I made numerous virtual test flights. This section contains less reliable data about other airplanes based on my experience in the simulator. You may find it useful as an introduction to those airplanes but bear in mind my only goal was to make flights that seem OK and acquire basic knowledge.

8.12.1 How to land the Cherokee Warrior II

The Cherokee Warrior II has some advantages upon the Cessna 172p. Thanks to its low wings it is far less sensitive to crosswind. Fully extended flaps provide more braking and allow it to land on a much shorter distance.

Take off is the same as for the Cessna 172p in FlightGear. In real life their take off checklists are not exactly the same.

You have to get used to some minor differences of the Cherokee Warrior II for the landing:

- During the steady horizontal flight before landing, the trim must be pulled a little below neutral in order to get the yoke around neutral.

- The optimal tachometer RPM during landing is at a lower RPM than the tachometer green zone. Roughly, keep the needle vertical.
- Only put use two steps of flaps during landing. Don't decrease the engine throttle too much.
- If you remain at two flaps deployed during landing, the round-out and flare will be similar to the Cessna 172p. However, using the third set of flaps will slow the aircraft down dramatically. It will very quickly touch the runway then come to a near halt. Be prepared to lower the front wheel very soon. (It is possible to use the third flaps step during the descent towards the runway, instead of tuning the engine power down. Oscillating between two steps and three steps allows to aim the runway start. Yet keep two flaps steps and tune the engine seems easier. An interesting stunt is to fly stable till nearly above the runway start, then tune the engine to minimum and deploy three flaps steps. The plane almost falls to the runway. It's impressive but it works.)

In real life, an advantage of the Cessna 172p upon the Cherokee Warrior II is the fuel reservoirs of the Cessna are located in the wings close above the center of the plane and higher than the engine. What's more an automatic system switches between the reservoirs. That means you almost don't have to bother for the way the fuel gets to the engine in flight. On the contrary, on the Cherokee Warrior II the reservoirs are located separately, on both wings and lower than the engine. That means you have to constantly switch between the two reservoirs in flight. Should one reservoir become much lighter than the other, this would destabilize the airplane. The fact the reservoirs are lower than the engine means you have to control the fuel pumps and the backup fuel pumps.

Some links:

- https://en.wikipedia.org/wiki/Piper_Cherokee
- http://freechecklists.net/Resources/Piper/PA-28-151+Warrior/

8.12.2 How to take off and land the Piper J3 Cub

The Piper J3 Cub is a very different airplane from the Cessna 172p and the Cherokee Warrior II. The Cessna 172p and the Cherokee Warrior II are nose-wheel airplanes, while the Piper J3 Cub is a tail wheel airplane. Take off and landing with tail wheel airplanes is more difficult. You have to tightly use the rudder pedals when rolling over the runway. The yoke often needs to be pulled backwards to the maximum. The Piper J3 Cub is a good introduction to tail-wheel aircraft and it is quite easy to take off and land provided you follow an appropriate procedure. Stall speed seems to be a little below 40 mph (the airspeed indicator is in mph) (about 27 knots). Take-off is below 50 mph.

My take off procedure for the Piper Cub is to fully pull the yoke backwards then throttle the engine to maximum. Once the front wheels clearly rises from the ground, gently push the yoke back to neutral, towards a normal flight close above the runway. Let the plane accelerate to 50 mph. Then pull the yoke to keep a little more than 50 mph while rising in the air.

The landing procedure is quite different to that of 172, as the aircraft is very light, and has no flaps.

1. Fly at say 500 ft constant altitude and "exactly" 52 mph speed towards the runway. Let the engine cover eat up the runway start. The engine cover will hide the runway completely. To see where the runway is, push the yoke/mouse very shortly then stabilize again in normal flight.

2. Once the runway start matches with the set of instruments (if you could see through the instrument panel), reduce the throttle to a near minimum and begin the dive towards the runway start. Keep 52 mph using the yoke. Add some throttle if you are going to miss the runway edge. (Keep in mind just a little wind is enough to change things a lot for the Piper J3 Cub.)

3. Make the rounding and pull the throttle to minimum. Do not pull steadily on the yoke. Instead let the wheels roll on the runway immediately.

4. Once the wheels roll on the runway, *push* firmly on the yoke, to its maximum. This rises the tail in the air. You would think the propeller will hit the runway or the airplane will tilt over and be damaged. But everything's fine. The wings are at a strong negative angle and this brakes the plane. (Don't push the yoke this way on other airplanes, even if their shape seems close to that of the Piper J3 Cub. Most of them will tumble forwards.)

5. The yoke being pushed in to its maximum, push the left mouse button and keep it pushed to go in rudder control mode. Keep the plane more or less centered on the runway. This is quite uneasy. One tip is to stop aiming the rudder to say the left already when the plane just starts to turn to the left.

6. Once the speed is really low (and the rudder control stabilized), you will see the tail begins to sink to the ground. Release the left mouse button to go back to yoke control. Pull the yoke backwards completely, to the other extreme. The tail now touches the ground and the nose is high up. Now you can use the wheel brakes (**b**). (If you use the brakes too early, the plane nose will hit the ground.)

The take off procedure mentioned above is symmetrical to the first landing procedure. There exists a second take off procedure, symmetrical to the second landing procedure. Yet I don't succeed it properly so I won't write about it.

8.12.3 How to take off and land a jet

Take off on a jet is easy but you must have fast reflexes. My favorite jet on FlightGear is the A-4 Skyhawk. You get it with the `--aircraft=`*a4-uiuc* parameter, provided it is installed.

This is the "calm" procedure to take off:

- Ask for a red and full HUD by typing **h** twice. The engine throttle indicator is the leftmost on the HUD.

- The airspeed indicator is the one labeled "KIAS" on the upper left side of the instrument panel. You can also use the airspeed indicator on the HUD, of course.

- Tune in half of the engine power.

- Keep the yoke pulled in halfway (see the arrow on the right side of the vertical line shown in figure 8.36).

Figure 8.36: Yoke indicator on the HUD

- It is not mandatory to use the rudder to keep on the runway. The airplane will take off before it drifts off the runway. (For sure it is better and more "secure" to keep in the middle of the runway. But using the rudder can make things hectic for a beginner.)

- Once above about 160 knots, the plane rises its nose in the air. Immediately push the yoke back to neutral or almost and stabilize at 200 knots airspeed (which makes a fair climb angle) (I've no idea whether 200 knots is the right climb speed for a real A-4. What's more I suppose one should rather use the AoA – see below.)

- Retract the landing gear using key **g**.

- Either maintain the engine power at ½ and a speed of 200 knots to get above the clouds, or reduce the engine power to less than ¼ and fly normally. (Off course you can "fly normally" with full engine power. Great fun.)

The "nervous" take off procedure is the same but you push in full engine power. The plane takes off quickly and you need to settle a very steep climb angle to keep 200 knots. Best retract the landing gear immediately.

You don't land a jet the same way you land a little propeller airplane. My way to land the A-4, inspired by some texts I found on the Web, is this:

- Really far from the runway, keep below 2000 ft and get the speed below 200 knots. Then lower the landing gear (key **G**) and I deploy full flaps (all three steps, by hitting **]** three times.)

- Keep a steady altitude of about 1000 ft and a speed of "exactly" 150 knots. Use the mouse/yoke/elevator to tune the altitude and the engine throttle to tune the speed. (The opposite from the Cessna.)

- Try to align with the runway.

- When do you know the dive towards the runway must begin? For this you need the HUD; the full default HUD with lots of features. Look at the figure 8.37. When you see the "distance" between the green 0° lines and the runway start is 25 % the distance between the green 0° lines and the green −10° dotted line, it is time to dive, aiming at the runway start. (In the figure 8.37, that "distance" is 64 %, far too much to start a landing.)

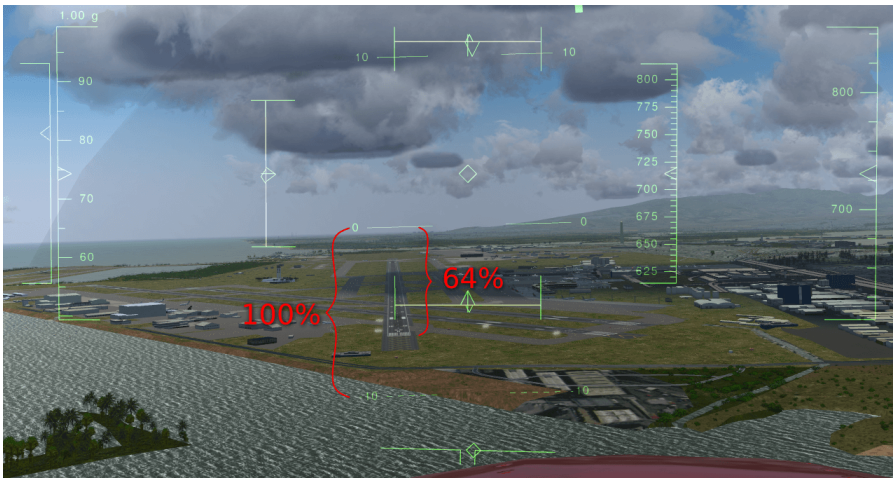

Figure 8.37: Aiming at the threshold of the runway with the HUD

Let's explain this. The two horizontal lines labeled "0" show the horizon line. Rather they show where the horizon would be if the Earth was flat. When your eyes aim at those "0" lines, you are looking horizontally. Look at the dotted green lines labeled "−10". A feature on the ground situated there is situated 10° below the ideal horizon. In other words: when you look to objects "hidden" by the lines labeled "0", you have to lower your eyes of 10° to look at objects "hidden" by the dotted lines labeled "−10". This implies, and it is very important, that a person in a rowboat, "hidden" by the dotted lines labeled "−10", has to rise his eyes up 10° to look at your plane. He sees you 10° above the horizon. In figure 8.37, the start of the runway is situated at 64 % of the way towards the green "−10" dotted lines. That means you have to lower your eyes of 6.4° to look at the runway start. This also means that if you start now to descent towards the runway start, the descent path will be of 6.4° (too steep). So, the HUD allows to measure precisely the angle of the descent path. On a jet plane you need an angle of 2.5° (up to 3°), that is 25 % of −10° (up to 30 %).

- Once descending towards the runway start, aim at it using the yoke/mouse. And keep 150 knots speed using the engine throttle lever.

- Keep measuring the angle between the ideal horizon and the runway start. It must keep 2.5° (that is 25 % of −10°):

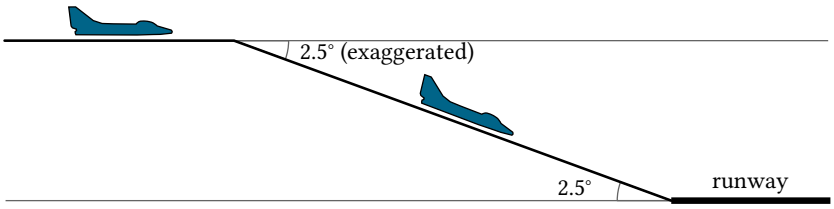

Figure 8.38: Glideslope

  - If the angle increases above 2.5°, you are above the desired path and you must loose altitude faster. Both decrease the engine power and dive the nose a little.
  - If the angle decreases below 2.5°, you are under the desired path. I wouldn't say you should gain altitude, rather you should loose altitude less fast. Both add a little engine power and rise the nose a little.

- Once very close to the runway start, do no rounding. Don't pull steadily on the yoke like you would for the Cessna 172p. Simply let the plane touch the ground immediately, at high speed. Let it smash on the runway, so to say. All three wheels almost together. Just throttle the engine down to minimum. (If you try to pull steadily on the yoke and hover over the runway while the plane nose rises steadily, on a F-16 you would scrape the plane rear and probably destroy it.)

- Keep the key **b** down to brake and use the rudder to stay aligned with the runway. Make only very little tunings with the rudder, otherwise the plane will tumble on one of its sides.

The HUD in a real jet contains a symbol to show towards what the airplane is moving. It is shown in figure 8.39. When you are flying at constant altitude, that symbol is on the ideal horizon line. Once you dive towards the runway start, you simply have to place that symbol on the runway start. This is quite an easy and precise way to aim at the runway start. (The diamond in the center of the FlightGear HUD sometimes can help but it does not have the same purpose. It shows towards what the airplane nose is pointing. For example, if you descend towards the ground at low speed, the symbol would be somewhere on the ground while the FlightGear diamond will be up in the sky.) (By the way, the HUD on the virtual B-52 on FlightGear has that symbol. It is great to use while landing.)

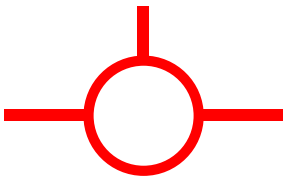

Figure 8.39: Aim point symbol

Also, a real HUD shows a dotted line at −2.5°, to help find the correct descent path. Simply keep that dotted line on the runway thresh-hold.

In additional to airspeed, military fast jet pilots rely on using the correct angle of attack during approach. The Angle of Attack (AoA) is the angle at which the wings are pitched against the relative airflow. The advantage of keeping to an optimal AoA is that the optimal AoA for landing does not depend on the plane load, while the optimal airspeed speed does. By ensuring that the AoA is correct for every landing, you will land at the correct speed, whatever the plane load.

The Angle of Attack is displayed within the HUD, and/or as a set of three lights (see figure 8.40). When the upper ∨ is lit, your Angle of Attack (AoA) is too high and you need to pitch down. When the lower ∧ is lit, your AoA is too low and you need to pitch up. The center ◯ indicates your the AoA is OK. Obviously, as you pitch up or down your airspeed and descent rate will change, so you will need to change your throttle setting appropriately.

Figure 8.40: The HUD in F-14B

The Cessna 172 and the A-4 Skyhawk are two extremes. Most other airplanes are in-between these two extremes. If you trained them both (and one or two tail wheel airplanes), you should be able to find out how to take off and land most other airplanes.

160 knots seems an appropriate landing speed for the F-16 Falcon. Also you need to throttle down the engine to minimum just before the plane should touch the runway. Otherwise it will hover over the runway. Don't bother for the flaps. It seems they are deployed automatically with the landing gear. (Read the section 8.7.4 about the stall.)

140 up to 150 knots and all 8 flaps steps deployed seem appropriate to land the virtual Boeing 737. But don't trust me especially on that one. I just made a few experiments and didn't search for serious data. The landing speed varies a lot depending on the plane load, I suppose 140 knots is for a plane with no load. The Boeing 737 seems to like a gentle rounding before the wheels touch the runway. Start the rounding early.

In the take off procedure for the Cessna 172 and the A-4 Skyhawk I recommend you pull the yoke/mouse/elevator halfway, from the start on. This seems to be a bad practice on the Pilatus PC-7. Keep the elevator neutral. Let the plane accelerate and wait till the speed gets over 100 knots. Then pull calmly on the yoke. During landing, deploy full flaps once you start plunging to the runway but don't decrease the engine throttle. Decrease it only when the hovering above the runway starts. 100 knots seems a good landing speed.

For the Cessna 310 too you better leave the elevator neutral during the acceleration on the

runway. The plane will raise its nose by its own provided you deployed one flaps step. (If you keep the yoke pulled from the start on, the nose will rise sooner and you will get yawful yaw problems.)

(Some virtual airplanes, like some big airliners or fast aircraft, need faster physical computations. Then add the `--model-hz=`*480* parameter to the command line. If the plane is difficult to control during landings, try this.)

The angle at which you land a Cessna 172p is far steeper than the narrow 2.5° for a jet. Nevertheless you are allowed to land the Cessna at a narrow angle too. (Provided the terrain around the runway allows for this, of course.) If you have passengers who have ears problems with the variation of air pressure…

8.12.4 How to take off and land the P-51D Mustang

Should you ever get a chance to pilot a P-51 Mustang, just say no. It is quite dangerous to take off and land. That's the kind of airplane you fly only when your country is in danger. You need a lot of training. Yet once in the air the P-51 Mustang seems no more dangerous to its pilot than other common military airplanes. It is quite easy to pilot.

At low and medium altitude the P-51 wasn't better than the Spitfire and the Messerschmitts. The big difference was at high altitude. The P-51 kept efficient and maneuverable while enemy fighters were just capable to hang in the air. This was an advantage at medium altitude too because the P-51 was able to plunge towards enemy airplanes from high altitude. Another key difference was the P-51 is very streamlined. Hence it was capable to fly much further than the Spitfire. These two differences let the P-51 Mustang fulfill its purpose: escort Allied bombers all the way to their targets in Germany. This allowed the bombings to be much more efficient and contributed to the defeat of the Nazis.

To get the P-51D Mustang, use the `--aircraft=`*p51d* command line parameter.

To take off the P-51D Mustang in FlightGear, deploy one flaps step, pull and keep the yoke completely backwards, push the engine throttle to maximum and keep the left mouse button pressed to control the rudder and keep on the runway. Once you reach exactly 100 mph, suddenly push the rudder ⅓ of its total way to the right. Immediately release the left mouse button and push the yoke to rise the tail (don't push it too much, as the sooner the wheels leave the ground the better.) From now on, keep the left mouse button released. Only make very short adjustments to the rudder. Let the plane rise from the runway and get to altitude at a speed of say 150 mph. Don't forget to retract the landing gear and the flaps.

Don't make too steep turns. You would loose control on the plane and crash.

To land, deploy full flaps and lower the landing gear from the start on. 130 mph speed seems fine, up to 140 mph. Make an approach from 1000 ft altitude and a dive at a low angle, like for a jet. Once over the runway, shut the engine down completely (key **{**). Don't hover over the runway. Get the wheels rolling soon (like for a jet). Hold the left mouse button down to steer the plane using the rudder. Once the tail sinks in, briskly pull the yoke (left mouse button shortly released) to force the tail on the runway. Go on steering the plane using the rudder. Now the tail is firmly on the ground, use the brakes if you want.

8.12.5 How to take off and land the B-52 Stratofortress

The B-52F bomber implemented in FlightGear is a success. It is one of my favorite airplanes. I'm sorry it was conceived to terrify me. One single B-52 bomber can wipe out every main town of my country and rise a nightmare of sicknesses and children malformation for centuries. All B-52 bombers united can wipe out mankind and almost every kinds of plants and animals on Earth.

The differences between the virtual B-52F bomber and the Cessna 172p are these:

- The B-52F starts with the flaps deployed and the parking brakes set.
- There are only two flaps steps: retracted and deployed. When deployed they are only meant to make the wings lift more, not to brake. If you want to brake, you need the spoilers. They deploy on the upper side of the wings. Use the key **k** to deploy the spoilers and the key **j** to retract them. There are seven steps of spoilers.
- The main landing gear of the Cessna 172p is composed of two wheels, one on each side of the airplane. In order for these wheels to leave and touch the ground altogether, you need to keep the wings parallel with the ground. The main landing gear of the B-52F is composed of a set of wheels at the front and a set of wheels at the rear. This implies that in order for these wheels to leave and touch the ground altogether, you need to keep the airplane body parallel with the ground.

This is my procedure to take off the virtual B-52F:

- *Push* the yoke 1/3 of the total way.
- Push the engine throttle to maximum.
- Release the parking brakes (key **B**).
- Push down the left mouse button to control the rudder pedals and keep the airplane on the runway.
- The whole runway length is needed till the B-52F rises from the ground (KSFO).
- Once the B-52F leaves the ground, around 190 knots seems appropriate to get to altitude.
- Retract the flaps and the landing gear.

To land, the B-52F's HUD offers that great airplane-shaped symbol I talked about in the section about jets. So you just have to put that symbol on the airplane threshold (a few pixels further seems optimal) and keep the runway start 2.5° below the ideal horizon line. 130 up to 140 knots seems a good landing speed. (Instead of the speed you can make use of the AoA indicator displayed on the schematic instrument panel (**P**).) Simply keep the AoA at 3°. I must confess I prefer to tune the speed rather than the AoA.) If the plane gets to the runway at 130 up to 140 knots, simply "let it smash" on the runway. Otherwise, if the speed is higher, make a rounding and a short hover. The brakes seem to be very effective **b**). They allow to stop the B-52F on roughly the same short runway length as the Cessna 172p.

Replays of the flights are a delight. They allow to check the plane body left the runway and landed back parallel with it. One of the points of view is situated inside the B-52F rear turret, which allows you to be your own passenger and to compare what you see with what you experienced as a passenger in airliners. The key **K** allows to visualize the airplane trajectory.

To cause an accident with the B-52 do this:

- Make a steep turn with a very strong bank; the wings nearly perpendicular to the ground.
- Try to get the plane back level. It will obey but very slowly. You will get aware that the turn will go on for a while and that you will turn further than your intended flight direction.
- Do something that accelerates the stabilization on some airplanes: push the rudder to an extreme, opposite to the current turn. This will suddenly make the airplane drop from the sky.

8.13 Thanks

I wish to thank:

- Benno Schulenberg who corrected lots of mistakes in my English in this tutorial.
- Albert Frank who gave me key data on piloting and corrected technical errors.
- Vassilii Khachaturov who taught me new things about FlightGear.
- Roy Vegard Ovesen for pointing me to the official Autopilot Pilots Guide.
- Dene Maxwell for his solution for problems under Windows ME.
- Mark Akermann and Paul Surgeon for their remarks.
- Michael "Sam van der Mac" Maciejewski who made the translation in Polish and converted the tutorial for use in the manual.
- The FlightGear mailing list users for their hearty welcome.
- 4p8 webmaster my friend Frédéric Cloth for the web space used by this tutorial.

Chapter 9

A Cross Country Flight Tutorial

9.1 Introduction

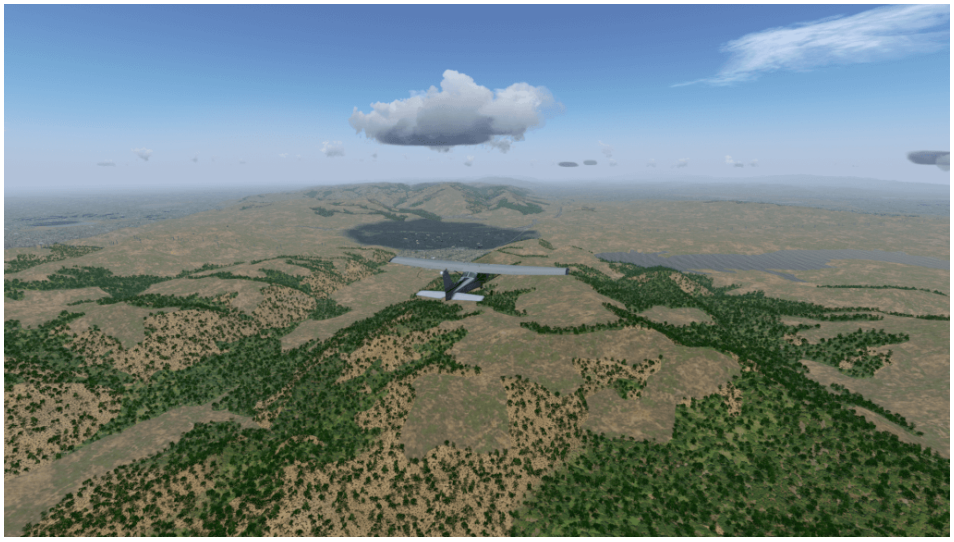

Figure 9.1: Flying over the San Antonio Dam to Livermore

This tutorial simulates a cross-country flight from Reid-Hillview (KRHV) to Livermore (KLVK) under Visual Flight Rules (VFR). Both airports are not included in the standard FlightGear package, therefore make sure that in the Launcher you have enabled the option *Download scenery automatically*, which is in the *Settings* tab.

I'll assume that you are happy taking off, climbing, turning, descending and landing in FlightGear. If not, have a look at the tutorials in chapters 7 (Tutorials) and 8 (A Basic Flight Simulator Tutorial). This tutorial is designed to follow on from them and provide information on some of the slightly more complicated flight systems and procedures.

9.1.1 Disclaimer and Thanks

A quick disclaimer. I fly microlights rather than Cessnas in real life. Most of this information has been gleaned from various non-authoritive sources. If you find an error or misunderstanding, please let me know. Mail me at stuart_d_buchanan -at- yahoo.co.uk.

I'd like to thank the following people for helping make this tutorial accurate and readable: Benno Schulenberg, Sid Boyce, Vassilii Khachaturov, James Briggs.

9.2 Flight Planning

Before we begin, we need to plan our flight. Otherwise we'll be taking off not knowing whether to turn left or right.

First, have a look at the Sectional for the area. This is an aeronautical map which shows airports, navigational aids, and obstructions. There are two scales of sectionals for VFR flight – the 1:500 000 sectionals themselves, and a number of 1:250 000 VFR Terminal Area Charts which cover particularly busy areas.

They are available from pilot shops, or on the web from various sources. You can access a Google-map style interface here:

https://skyvector.com

Simple click "Flight Plan" and enter KRHV to "Departure" field. An extract from the chart is shown in Figure 9.2.

If you want a map of the entire area showing exactly where the plane is, you can use Atlas. This is a moving-map program that connects to FlightGear. See Section 6.3 for details.

An even simpler option is to use "Phi", which is a map in the browser. Just run FlightGear with the command line option `--httpd`=*8080* and then in the simulator choose *Equipment* → *Map (opens in browser)* from main menu.

So, how are we going to fly from Reid-Hillview to Livermore?

We'll be taking off from runway 31R at KRHV. KRHV is the ICAO code for Reid-Hillview airport, and you can find them in Launcher in *Location* tab. (It is marked on the sectional as RHV for historic reasons. To get the ICAO code, simply prefix a "K".)

The 31 indicates that the magnetic heading of the runway is around 310 degrees, and the R indicates that it's the runway on the right. As can be seen from the sectional, there are two parallel runways at KRHV. This is to handle the large amount of traffic that uses the airport. Each of the runways can be used in either direction. Runway 31 can be used from the other end as runway 13. So, the runways available are 13R, 13L, 31R, 31L. Taking off and landing is easier done into the wind, so when the wind is coming from the North West, runways 31L and 31R will be in use. The name of the runway is written in large letters at the beginning and is easily seen from the air.

Once we take off we'll head at 350° magnetic towards Livermore (KLVK). We'll fly at about 3500 ft about sea-level. This puts us at least 500 ft above any terrain or obstructions like radio masts on the way.

We'll fly over the Calaveras Reservoir then the San Antonio Reservoir. These are both large bodies of water and we can use them as navigation aids to ensure we stay on the right track.

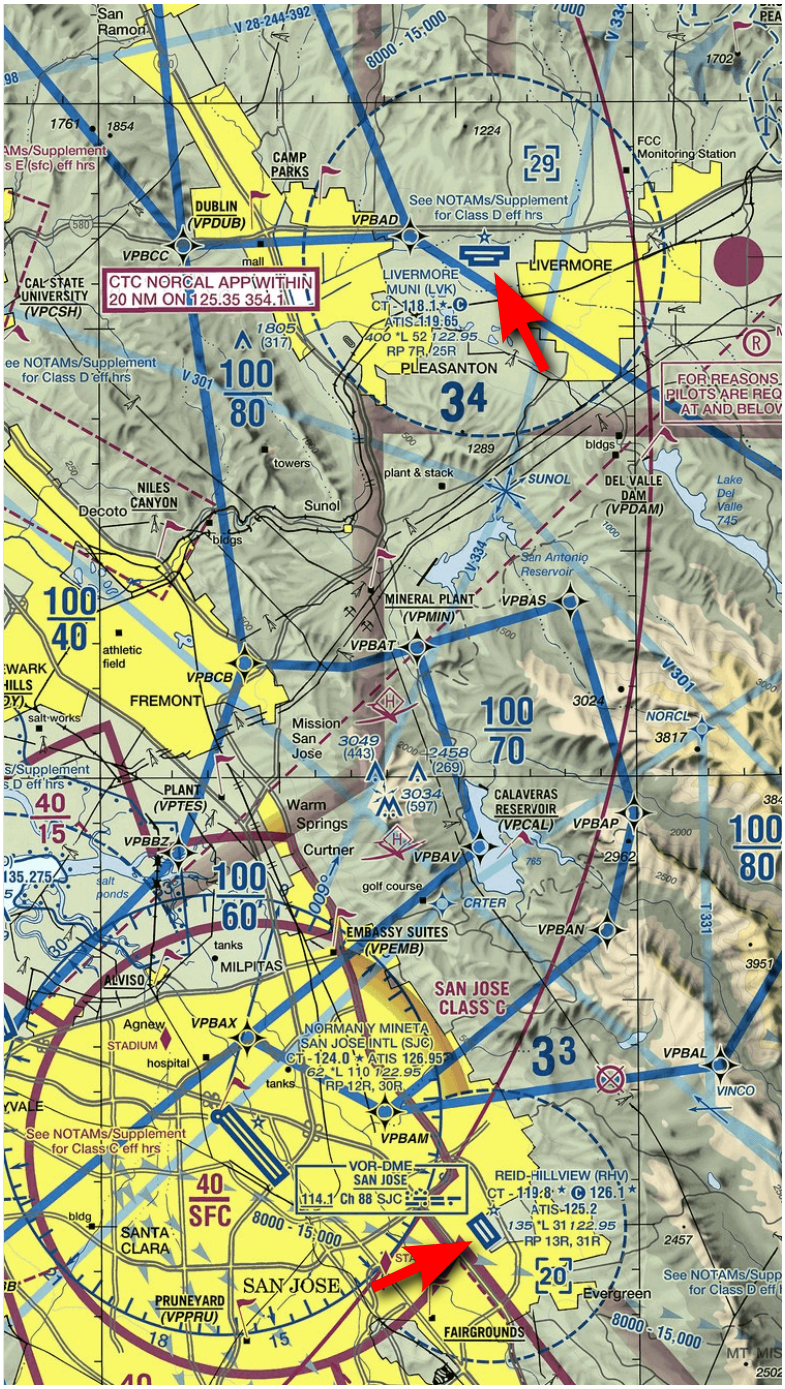

Figure 9.2: Sectional extract showing Reid-Hillview and Livermore airports

Once we get about 10 miles out of Livermore (above the San Antonio Reservoir), we'll contact the Livermore Air Traffic Control (ATC) to find out where we should land. We'll then join the circuit and land.

9.3 Getting Up

OK, we know where we're going and how we'll get there. Time to get started.

Start FlightGear using the launcher (or command-line if you prefer). We want to use a C172P and take off from runway 31R at Reid-Hillview of Santa Clara County (KRHV). Dawn is a nice time to fly in California.

If you want, you can fly in the current weather at KRHV (which is recommended to be able to listen the ATIS.) To do this, in the Launcher, go to the *Environment* tab and check the *Real-world weather* option.

Figure 9.3: On the runway at KRHV

9.3.1 Preflight

Before we take off, we need to preflight the aircraft. In the real world, this consists of walking around the aircraft to check nothing has fallen off, and checking we have enough fuel. If you want, you can do this by selecting *Walker* → *Toggle Walker outside* from the main menu.

However, in our case, we'll take the opportunity to check the weather, set our altimeter and pre-set things that are easier to do when you're not flying.

The weather is obviously important when flying. We need to know if there is any sort of cross-wind that might affect take-off, at what altitude any clouds are (this is a VFR flight – so we need to stay well away from clouds at all times), and any wind that might blow us off course.

We also need to calibrate our altimeter. Altimeters calculate the current alititude indirectly by measuring air pressure, which decreases as you ascend. However, weather systems can affect the air pressure and lead to incorrect altimeter readings, which can be deadly if flying in mountains.

9.3.2 ATIS

Conveniently, airports broadcast the current sea-level pressure along with useful weather and airport information over the Automatic Terminal Information Service (ATIS). This is a recorded message that is broadcast over the radio. However, to listen to it, we need to tune the radio to the correct frequency.

The ATIS frequency is displayed on the sectional (look for "ATIS" near the airport), but is also available from within FlightGear. To find out the frequencies for an airport (including the tower, ground and approach if appropriate), use the *AI → ATC Services in Range* menu. Then, in the dialog box, click the button with the ICAO code of our airport, i.e. KRHV. The various frequencies associated with the airport are then displayed. Duplicates indicate that the airport uses multiple frequencies for that task, and you may use either.

Either way, the ATIS frequency for Reid-Hillview is 125.2 MHz.

9.3.3 Radios

We now need to tune the radio. The radio is located in the Radio Stack to the right of the main instruments. There are actually two independent radio systems, 1 and 2. Each radio is split in two, with a communications (COMM) radio on the left, and a navigation radio (NAV) on the right. We want to tune COMM1 to the ATIS frequency.

Figure 9.4: The C172 communications stack with COMM1 highlighted

The radio has two frequencies, the active frequency, which is currently in use, and the

standby frequency, which we tune to the frequency we wish to use next. The active frequency is shown on the left 6 digits, while the standby frequency is shown on the right. We change the standby frequency, then swap the two over, so the standby becomes active and the active standby. This way, we don't lose radio contact while tuning the radio.

Figure 9.5: COMM1 adjustment knob

To change the frequency, use the mouse wheel on the black knob below the standby frequency (highlighted in the Figure 9.5), just to the right of the "STBY". Turning the mouse wheel over the small (inner) knob changes the decimal point (kHz), and turning the wheel over the large (outer) knob changes the number before the decimal place (MHz). Turnning the wheel in both ways – you will see the values go up and down. Alternatively, you can click each knob. Left click mouse button increasing the frequency, and middle click decreasing.

As for the COMM radio, note that the frequency before the decimal point (MHz) can be set in the range from 118 to 136, in 1 MHz step. However, the frequency after the decimal point can work in two standards, the older one with a spacing of 25 kHz, or the new one with a spacing of 8.33 kHz. The default is 8.33 kHz spacing, giving you more frequencies to choose from. If you want you can switch the radio between 25 kHz and 8.33 kHz spacing, just hold down the **`Shift`** key and click the little knob. When you look closely, you will notice that a small, inner knob is pushed in.

If you are having difficulty clicking on the correct place, press **`Ctrl-c`** to highlight the hot-spots for clicking.

Once you have tuned the frequency to 125.2 MHz, press the white button with arrows, located between the words "COMM" and "STBY" to swap the active and standby frequencies (highlighted in Figure 9.6). After a second or so, you'll hear the ATIS information. If you can't hear ATIS, make sure the "COMM1" switch above the frequencies is in the *Speaker* or *Phone* position (see Figure 9.7).

Figure 9.6: COMM1 transfer key to swap the frequencies

Figure 9.7: Speakers switch for COMM1

9.3.4 Altimeter and Heading Indicator

Figure 9.8: Altimeter calibration knob

Listen for the "Altimeter" setting. If you are not using real weather, the atmospheric pressure will depend on the weather conditions set, according to the selected weather scenario. Then listening to ATIS will be impossible. Therefore, to check the pressure, go to menu *Environment* → *Weather*. In the bottom of new window, there is a METAR (Meteorological Aerodrome Report) message with atmospheric pressure. If you don't know how to read the METAR please use the "METAR Description" button.

As you can see, the atmospheric pressure can be different every time. Therefore, we need to set the altimeter to the correct value. To do this, use the knob at the bottom left of the altimeter (circled in red in Figure 9.8), in the same way as you changed the radio frequency. Turning the knob changes the value in the little window on the right of the altimeter. You must set the same value as given by ATIS (or METAR). Notice that the altitude indicated by the altimeter is changing too.

The other way to set the altimeter is to match it to the elevation above sea-level of the airport. The elevation is listed on the sectional. For KRHV it is 133 ft. This means you can double-check the pressure value reported over ATIS.

As you may have noticed, the pressure on the Cessna altimeter can only be set in units of inHg. Fortunately, we fly in the USA, where ATIS also gives us the pressure in inHg. But when flying in Europe, for example, the pressure will be given in hPa. Then we have to convert hPa to inHg, or simply use the "METAR Description" window mentioned above. Another way is to use the menu *Equipment* → *Instrument Settings*, where you can set the pressure for the altimeter, both in hPa and inHg.

We will also take the opportunity to set the heading bug on the heading indicator to 350° – our bearing from KRHV to KLVK. To do this, use the amber knob on the heading indicator

Figure 9.9: Heading adjust knob

housing (highlighted in Figure 9.9), just as you've done before. For faster rotation, hold down the **`Shift`** key to increase the increment value by 5°. The value of 350° is just anti-clockwise of the labeled value of N (North – 000°).

9.3.5 Take-Off

OK, now we've done that we can actually take off! In my case this usually involves weaving all over the runway, and swerving to the left once I've actually left the ground, but you'll probably have better control than me. Once above 1000 ft, make a gentle turn to the right to a heading of 350°. As we've set the heading bug, it will be easy to follow. We're aiming for a fairly prominent valley.

Continue climbing to 3500 ft at around 500-700 fpm. Once you reach that altitude, reduce power, level off to level flight and trim appropriately. Check the power again and adjust so it's in the green arc of the RPM gauge. We shouldn't run the engine at maximum RPM except during take-off.

9.4 Cruising

OK, we've taken off and are on our way to Livermore. Now we can make our life a bit easier by using the autopilot and our plane more fuel efficient by tuning the engine. We'll also want to check we're on-course.

9.4.1 The Autopilot

We can make our life a bit easier by handing some control of the aircraft over to "George" – the autopilot.

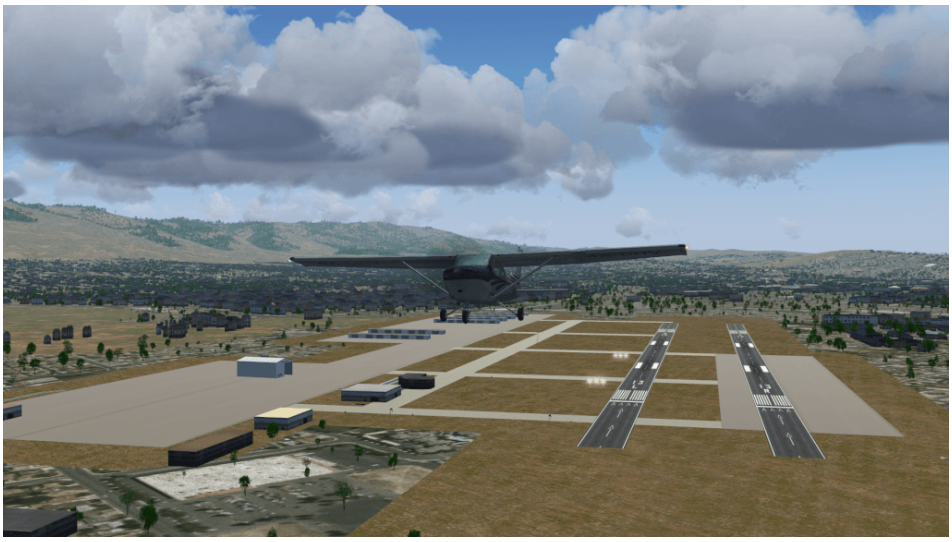

Figure 9.10: Take-off from KRHV

Figure 9.11: The C172 Autopilot

The autopilot panel is located towards the bottom of the radio stack (highlighted in Figure 9.11). It is easily distinguishable as it has many more buttons than the other components on the stack. It can work in a number of different modes, but we are only interested in one of them for this flight – HDG. As the names suggest, HDG will cause the autopilot to follow the heading bug on the heading indicator, which we set earlier.

To set the autopilot, press the AP button to switch the autopilot on, then press the HDG button to activate heading mode. While the autopilot is switched on, it will use the ailerons controls to keep the plane on the heading. You can change the heading bug, and the autopilot will maneuver appropriately. However, the autopilot doesn't make any allowances for wind speed or direction, it only sets the heading of the airplane. If flying in a cross-wind, the plane may be pointed in one direction, but be travelling in quite another.

You should use the trim controls to keep a level flight. You can use the autopilot for this, but it is a bit more complicated.

Once the aircraft has settled down under the autopilot's control, we can pay more attention to the outside world and higher level tasks.

9.4.2 Navigation

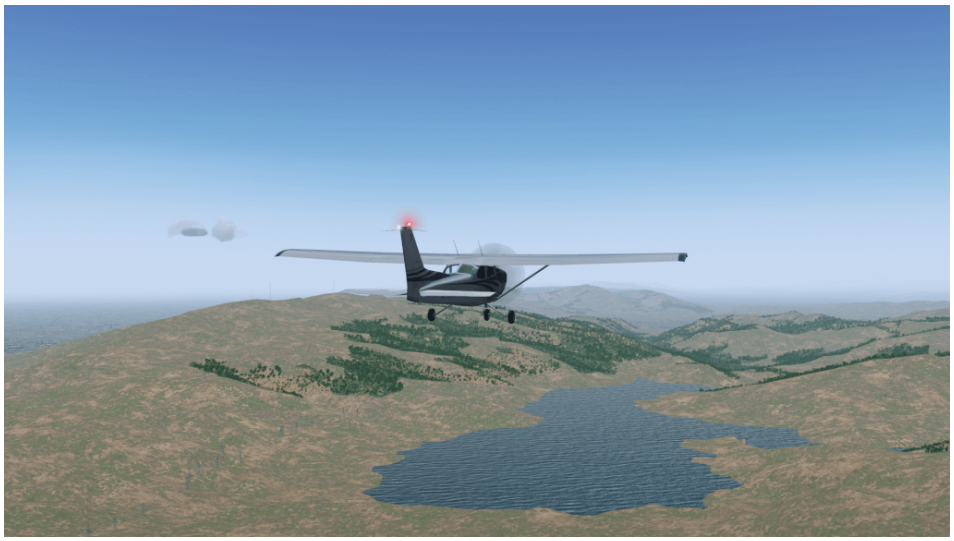

Figure 9.12: The Calaveras Reservoir

As we noted above, we're going to be travelling over a couple of reservoirs. When you leveled off, the first (Calaveras) was probably right in front of you. You can use them to check your position on the map. If it looks like you're heading off course, twist the heading bug to compensate.

9.4.3 Mixture

As altitude increases, the air gets thinner and contains less oxygen. This means that less fuel

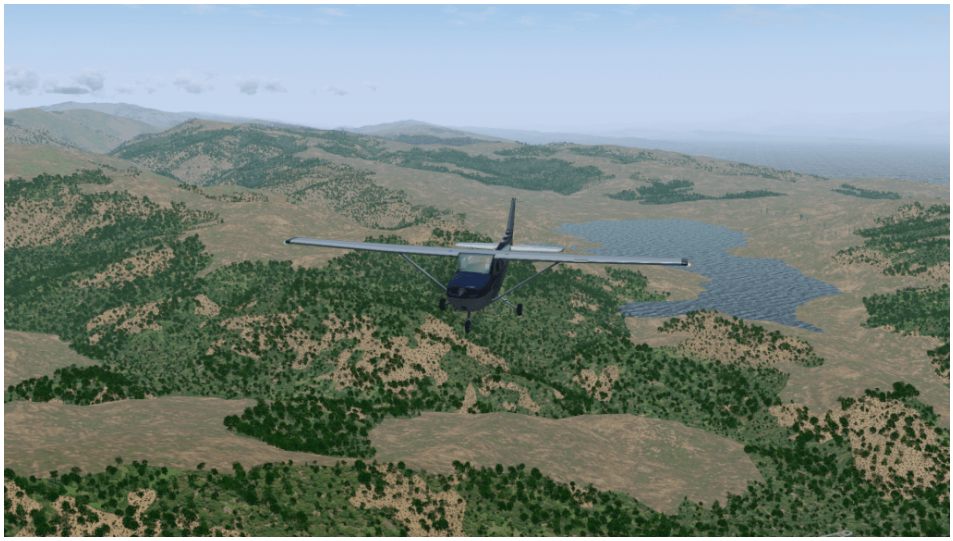

Figure 9.13: The Calaveras Reservoir

can be burnt each engine cycle. The engine in the C172 is simple and doesn't automatically adjust the amount of fuel to compensate for this lack of oxygen. This results in an inefficient fuel burn and a reduction in power because the fuel-air mixture is too "rich". We can control the amount of fuel entering the engine every cycle using the mixture control. This is the red lever next to the throttle (Figure 9.14). By pulling it out, we "lean" the mixture. We don't want the mixture too rich, nor too lean. Both these conditions don't produce as much power as we'd like. Nor do we want it perfect, because this causes the fuel-air to explode, rather than burn in a controlled manner, which is a quick way to trash an engine.

The mixture is controlled by the red lever to the right of the yoke. You may need to pan your cockpit view to see it.

To pan the cockpit view, hold down the right mouse button. Moving the mouse now pans the view. Once you can see the mixture lever clearly, release the right mouse button.

Pull the mixture lever out slowly (use **Ctrl-c** to see the hot spots), leaning the mixture. As you do so, you'll see various engine instruments (on the left of the panel) change. Fuel flow will go down (we're burning less fuel), EGT (Exhaust Gas Temperature) will go up (we're getting closer to a "perfect mixture") and RPM will increase (we're producing more power). Pull the mixture lever out until you see the EGT go off the scale, then push it in a bit (see Figure 9.15). We're now running slightly rich of peak. While at 3500 ft we don't need to lean much, at higher altitudes leaning the engine is critical for performance.

9.5 Getting Down

Once you reach the second reservoir (the San Antonio Reservoir), we need to start planning our descent and landing at Livermore. Landing is a lot more complicated than taking off, assuming you want to get down in one piece, so you may want to pause the simulator (press **p**) while

Figure 9.14: Mixture Control

Figure 9.15: EGT gauge

reading this.

9.5.1 Air Traffic Control

In the Real World, we'd have been in contact with Air Traffic Control (ATC) continually, as the bay area is quite congested in the air as well as on the ground. ATC would probably provide us with a "flight following" service, and would continually warn us about planes around us, helping to avoid any possible collisions. The FlightGear skies are generally clear of traffic, so we don't need a flight following service. If you want to change the amount of traffic in the sky, you can do so from the *AI* menu.

Livermore Airport is Towered (towered airports are drawn in blue on the sectional), so we will need to communicate with the tower to receive instructions on how and where to land.

Before that, we should listen to the ATIS, and re-adjust our altimeter, just in case anything has changed. This is quite unlikely on such a short flight, but if flying hundreds of miles, it might make a difference. To save time when tuning radios, you can access the Radio Frequencies dialog from the *Equipment* → *Radio Settings* menu. The Livermore ATIS frequency is 119.65 MHz.

An ATIS message also has a phonetic letter (Alpha, Bravo, … Zulu) to identify the message. This phonetic is changed each time the recorded message is updated. When first contacting a tower, the pilot mentions the identifier, so the tower can double-check the pilot has up to date information.

Besides the altitude and weather information, the ATIS will also say which runway is in use. This is useful for planning our landing. Normally, due to the prevalent Westerly wind, Livermore has runways 25R and 25L in use.

Once you've got the ATIS, tune the radio to Livermore Tower. The frequency is 118.1 MHz. Depending on the level of AI traffic you have configured on your system, you may hear Livermore Tower talking to other aircraft that are landing or departing.

Once the frequency goes quiet, press the **'** key. This will bring up the ATC menu. Click on the radio button on the left to select what you wish to say (you only have one option), then click **OK**.

Your transmission will be displayed at the top of the screen. It will indicate who you are (type and tail number), where you are (e.g. 6 miles south), that you are landing, and the ATIS you have.

After a couple of seconds, Livermore Tower will respond, addressing you by callsign and telling you what runway to use, which pattern is in use and when to contact them, for example:

> "Golf Foxtrot Sierra, Livermore Tower, Report left downwind runway two five left."

To understand what this means, we'll have to describe the Traffic Pattern.

9.5.2 The Traffic Pattern

With the number of aircraft flying around, there have to be standard procedures for take-off and landing, otherwise someone might try to land on-top of an aircraft taking off.

The Traffic Pattern is a standard route all aircraft must follow when near an airport, either taking off or landing. The traffic pattern has four stages (or "legs"), shown in Figure 9.16. The "downwind" mentioned above refers to one of these, the one with the number 3.

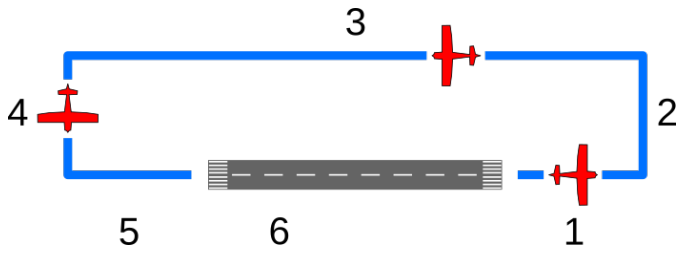

Figure 9.16: The Traffic Pattern

1. Aircraft take off from the runway and climb. If they are leaving the airport, they just continue climbing straight ahead until clear of the pattern and then do whatever they like. If they are returning to the runway (for example to practise landing), they continue climbing until they reach a couple of hundred feet below "pattern altitude". This varies from country to country, but is usually between 500 ft and 1000 ft Above Ground Level (AGL). This is called the *upwind* leg.

2. The pilot makes a 90 degree left-hand turn onto the *crosswind* leg. They continue their climb to "pattern altitude" and level out.

3. After about 45 seconds to a minute on the crosswind leg, the pilot again makes a 90 degree left turn onto the *downwind* leg. Aircraft arriving from other airports join the pattern at this point, approaching from a 45 degree angle away from the runway.

4. When a mile or so past the end of the runway (a good guide is when the runway is 45 degrees behind you), the pilot turns 90 degrees again onto the *base* leg and begins the descent to the runway, dropping flaps as appropriate. A descent rate of about 500 fpm is good.

5. After about 45 seconds the pilot turns again onto the *final* leg. It can be hard to estimate exactly when to perform this turn. Final adjustments for landing are made. I usually have to make small turns to align with the runway properly.

6. The aircraft lands. If the pilot is practising take-offs and landings, full power can be applied and flaps retracted for takeoff, and the aircraft can take off once more. This is known as "touch-and-go".

Most patterns at left-handed, i.e. all turns are to the left, as described above. Right-hand patterns also exist, and are marked as "RP" on the sectional. ATC will also advise you what pattern is in use.

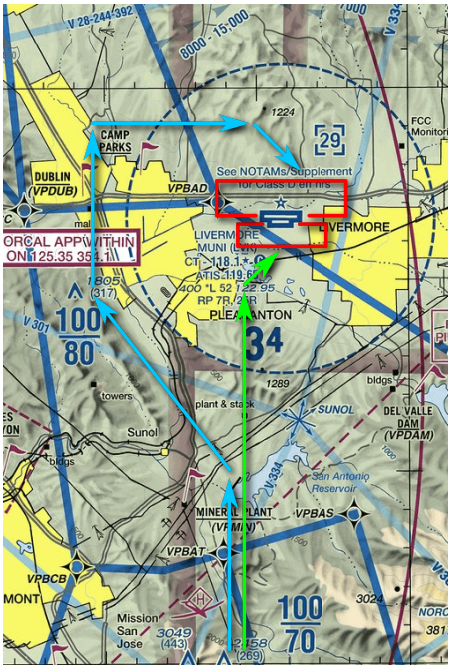

Figure 9.17: Sectional extract showing approaches to Livermore

9.5.3 Approach

We're approaching Livermore airport from the South, while the runways align East to West. Due to the prevailing Westerly wind, we'll usually be directed to either runway 25R or 25L. 25R uses a right-hand pattern, while 25L uses a left-hand pattern. Both the patterns are illustrated in Figure 9.17. Depending on the runway we've been assigned, we'll approach the airport in one of two ways. If we've been asked to land on runway 25R, we'll follow the blue line in the diagram. If we've been asked to land on runway 25L, we'll follow the green line.

We also need to reduce our altitude. We want to end up joining the pattern at pattern altitude, about 1000 ft AGL (Above Ground Level). Livermore airport is at 400 ft ASL (Above Sea Level), so we need to descend to an altitude of 1400 ft ASL.

We want to begin our maneuvers well before we reach the airport. Otherwise we're likely to arrive too high, too fast, and probably coming from the wrong direction. Not the best start for a perfect landing :).

So, let's start descending immediately.

1. First switch off the autopilot by pressing the AP switch.

2. Return mixture to fully rich (pushed right in). If we were landing at a high airport, we'd just enrich the mixture slightly and re-adjust when we reached the pattern.

3. Apply carb-heat. This stops ice forming when the fuel and air mix before entering the cylinder, something that can often happen during descent in humid air. The carb-heat lever is located between the throttle and mixture. Pull it out to apply heat.

4. Reduce power quite a bit. Otherwise we might stress the airframe due to over-speeding.

5. Drop the nose slightly to start the descent.

6. Trim the aircraft.

Use your location relative to the airport and the two towns of Pleasanton and Livermore to navigate yourself to the appropriate pattern as shown in the figure 9.17.

Once you're established on the downwind leg, you'll need to report to ATC again. Do this in the same way as before. They will then tell you where you are in the queue to land. "Number 1" means there are no planes ahead of you, while "Number 9" means you might want to go to a less busy airport! They'll also tell you who is ahead of you and where. For example "Number 2 for landing, follow the Cessna on short final" means that there is a single aircraft in front of you that is currently on the final leg of the pattern. When they land and are clear of the runway, they'll tell ATC, who can then tell you "Number 1 for landing".

9.5.4 VASI

Figure 9.18: On Final at Livermore with VASI on the left

Once on final, you'll notice two sets of lights on the left of the runway (enhanced in Figure 9.18). This is the VASI (Visual Approach Slope Indicator) and provides a nice visual clue as to whether you're too low or too high on approach. Each set of lights can either be white or red. White means too high, red means too low. White and red together means just perfect. On a Cessna approaching at 60 kts, a descent rate of about 500 fpm should be fine. If you are too high, just decrease power to increase your descent rate to 700 fpm. If you are too low, increase power to decrease your descent rate to 200 fpm.

Figure 9.19: Missed approach at Livermore

9.5.5 Go Around

If for some reason it looks like you're going to mess up the landing you can abort the landing and try again. This is called a "Go Around". To do this:

1. Apply full power.
2. Wait until you have a positive rate of climb – i.e. your altitude is increasing according to the altimeter.
3. Raise your flaps to 10 degrees (first-stage).
4. Tell ATC you are "going around".
5. Climb to pattern height.
6. If you aborted on final approach, continue over the runway to re-join the pattern on the crosswind leg. If on base, fly past the turn for final, then turn and fly parallel to the runway on the opposite side from downwind to rejoin on the crosswind leg.
7. Fly the complete pattern, telling ATC when you are on downwind, and try again.

9.5.6 Clearing the Runway

Once you're on the ground, you should taxi off the runway, then tell ATC you are clear. At high-altitude airports, you would lean the engine to avoid fouling the spark-plugs with an over-rich mixture. Find somewhere nice to park, shut down the engine by pulling mixture to full lean, then throttle off and magnetos to off (knob on the bottom left of the panel). Switch off the avionics master switch, tie down the aircraft, then go get that hamburger!

I hope this tutorial is of some use. If you have any comments, please let me know at stuart_d_buchanan {at} yahoo.co.uk.

Chapter 10

An IFR Cross Country Flight Tutorial

10.1 Introduction

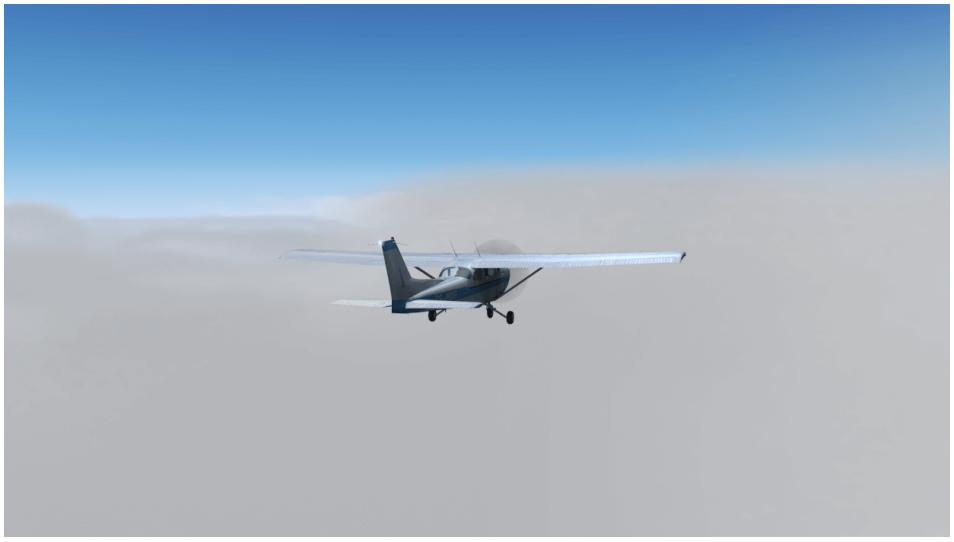

Figure 10.1: Flying over the San Antonio Dam to Livermore (I think)

In the cross country flight tutorial (see chapter 9), you learned about VFR flight, and in the course of the flight you were introduced to most of the flight instruments in the C172P. Now we're going to do an Instrument Flight Rules (IFR) flight. In this flight you'll be introduced to the remaining instruments, learn a bit about IFR flight, and learn many, many TLAs (Three-Letter Acronyms).

We'll fly the same flight, from Reid-Hillview (KRHV), runway 31R, to Livermore (KLVK), runway 25R, only this time we'll do it in IFR conditions: a ceiling 200 ft above ground level, and 800 metre visibility. This tutorial assumes you've completed the cross country flight tutorial.

10.1.1 Disclaimers

This is not intended to teach you how to fly IFR. Rather, it is meant to give a flavour of what IFR flying is like, and remove the mystery of the panel instruments not covered by the cross country flight tutorial.

I'm not a pilot. Like the previous tutorial, this information has been gleaned from various non-authoritative sources. If you find an error or misunderstanding, please let me know. Mail me at bschack-flightgear -at- usa -dot- net.

This flight was flown using FlightGear 3.0. Newer or older versions of FlightGear might be slightly different (the screenshots have been updated to version 2020.3.11. The approach map for Livermore has been changed, but FlightGear still offers withdrawn navaids, so the tutorial can be followed.)

10.2 Before Takeoff

We need to tell FlightGear about our flight conditions. There are different ways to set our "desired" weather, but we'll use the global weather menu. After launching FlightGear, click *Environment* → *Weather* to bring up the weather dialog. In the **Weather Conditions** list, select **CAT I minimum**.

This will give us a low ceiling and reduced visibility. Unfortunately, it will also give us rather stiff winds. If you don't want to deal with them, then you can easily turn off the winds:

- Click on **Weather Conditions** again, and select **Manual input**.
- In the METAR string at the bottom, change "15015KT" (15 knot winds coming from 150°) to "15000KT" (0 knot winds coming from 150°).

Hit **OK** to make FlightGear accept the changes and close the dialog.

10.2.1 Flight Planning

When you look out the window, you'll see something like Figure 10.2. Those clouds don't look very friendly, and it's hard to even see past the end of the runway. Maybe we should just *drive* there in the Cessna. We had been planning to practice ground steering anyway ...

So how do you get from A to B when you can't see? There are a variety of ways that have evolved over the years, with various advantages and disadvantages. Our flight will use all of the navigation instruments the standard Cessna C172P has, just to give a taste of what's possible.

Our entire route, and the aids we'll be using, are shown in Figure 10.3. Our route is in green, the navigational aids blue and red. The route looks a bit crazy – in fact, you might wonder if we're *more* lost using our fancy equipment than just flying by the seat of our pants – but there is a method to the madness. Rather than overwhelming you with details by explaining it all now, I'll explain it bit by bit as we go along.

Figure 10.2: On runway 31R at KRHV

10.2.2 VHF Omnidirectional Range

The first bit will involve VOR[1] navigation, and will get us to a point about 5 nm (nautical miles) south of Livermore.

VOR stations are indicated on the sectional by a big bluish-green circle with compass markings around the outside. I've helped you by marking their centers with a big blue dot as well. Reid-Hillview is very close to one, San Jose, which you can see in Figure 10.3. Near the centre of the circle, in a bluish-green rectangle, is the station information. According to the station information, it's a VOR-DME station (I'll explain DME later), its name is San Jose, its frequency is 114.1 MHz (or Channel 88, which is an alternative way to say the same thing), and its identifier, or "ident", is SJC (which in Morse code is ··· ·––– –·–·).

To tune into a VOR station, we use one of the NAV receivers, which are paired with the COMM receivers (see Figure 10.4). And we navigate using the corresponding VOR gauge. We'll choose the NAV1 receiver (and VOR1 gauge) in this case (NAV2 would have worked just as well). Before setting the frequency, check out the VOR1 gauge. It should look like VOR1 on the left in Figure 10.5. The important thing is the red "NAV" flag. That means there's no VOR signal, so we can't trust the gauge.

The NAV receiver has an active frequency, a standby frequency, and a tuning knob, just like the COMM receiver.[2] Tune it to 114.1, and press the swap button. If you look at VOR1, you should notice that the red "NAV" flag has disappeared, to be replaced with a "TO" flag, as shown on the right of Figure 10.5. That means we're receiving a signal. But is it the correct one? What if we accidentally set the wrong frequency?

NAV1 ⇒ 114.1[4]

[1] See https://en.wikipedia.org/wiki/VHF_omnidirectional_range for more information.

[2] Operation of the COMM receivers was covered in the cross country flight tutorial.

[4] All important actions and events will be given in the margin. This should provide a nice summary of the flight, uncluttered by the verbiage of the text.

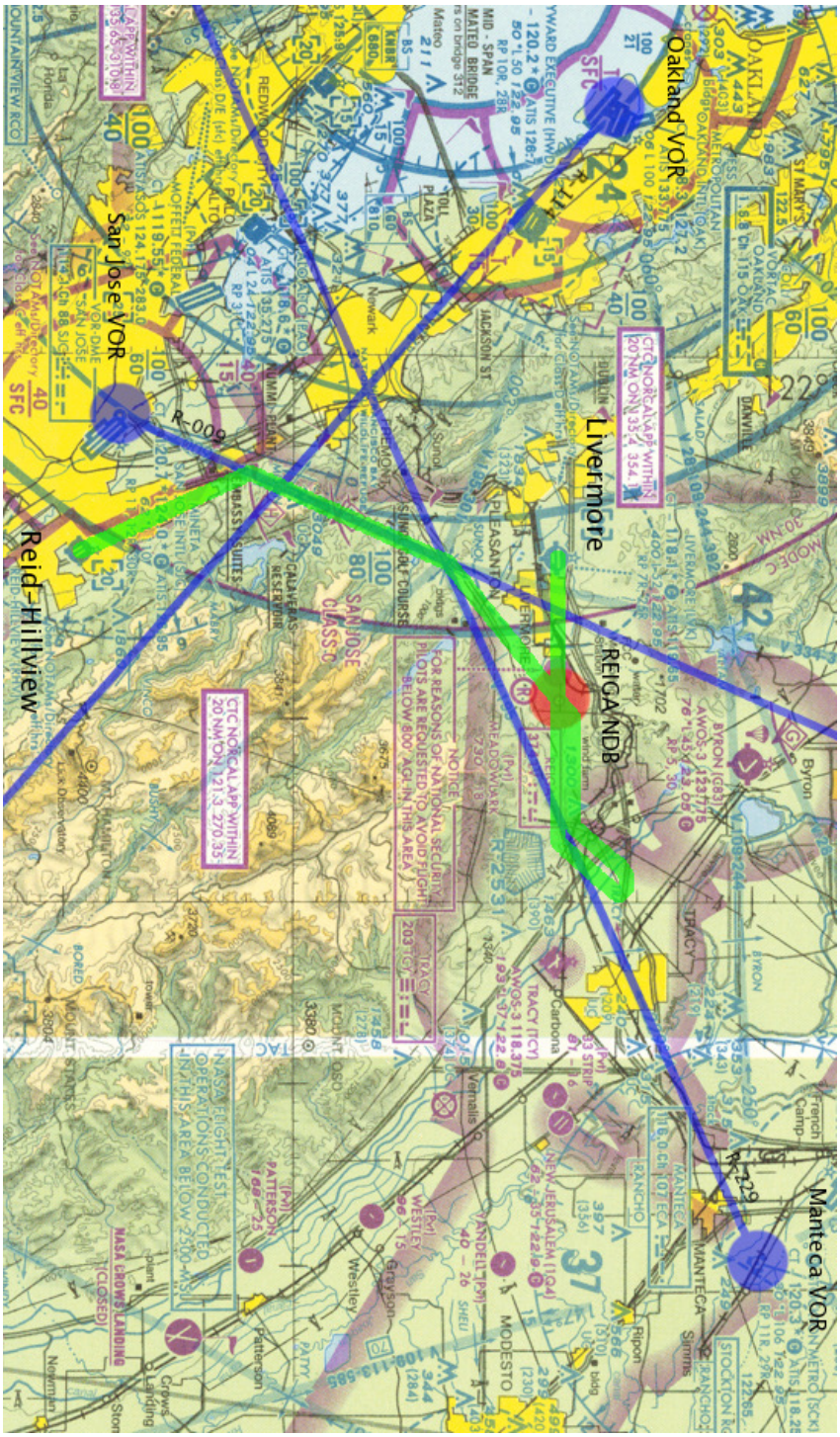

Figure 10.3: Green: our route, Blue: VORs and radials, Red: NDBs

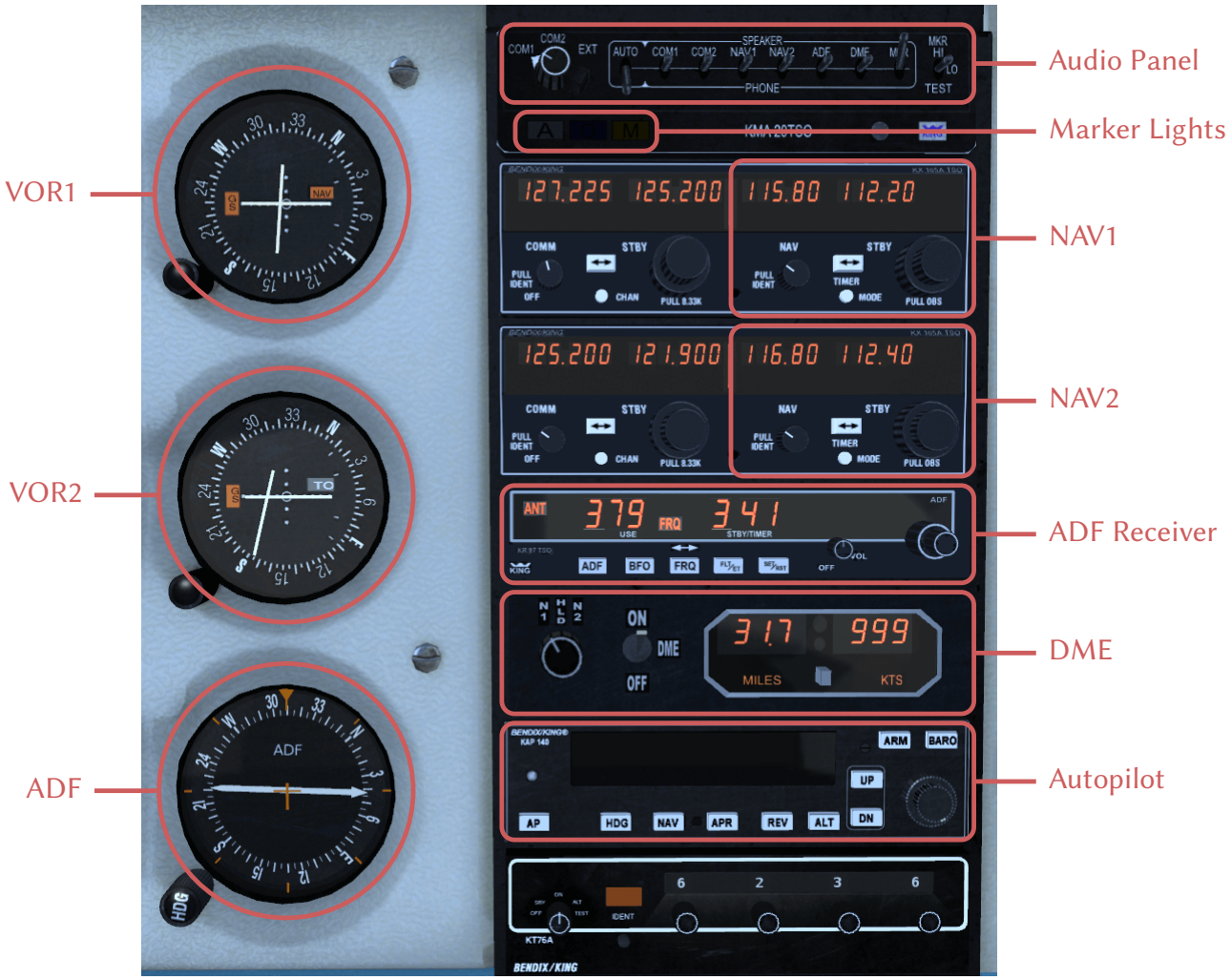

Figure 10.4: IFR navigation instruments

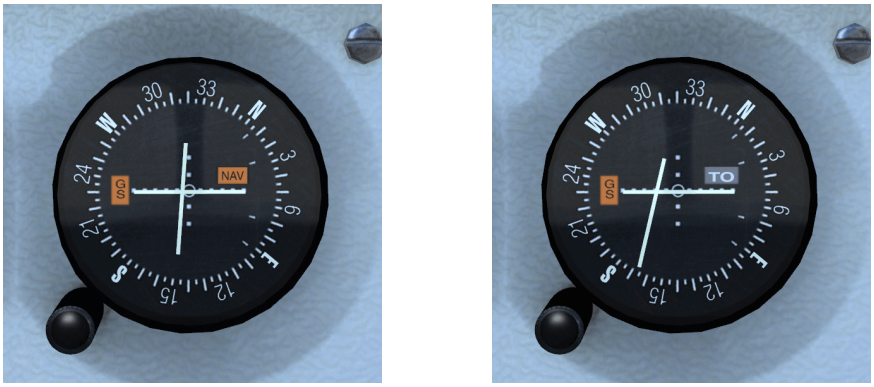

Figure 10.5: VOR1, before and after tuning

To confirm that we're tuned into the correct VOR, we listen for its ident. If you can't hear the ident, or if it doesn't match the chart, don't trust the needle. So far, you probably haven't heard a thing. Why? Check the audio panel (see Figure 10.4). You'll note there's a switch for all the instruments that produce useful sounds, and NAV1 is one of them. Flip the switch up (or down – it doesn't matter), and you should hear this: ··· ·––– –·–·.[5] Nice. Flip the switch back to the centre when you get tired of listening to dots and dashes.

Back to VOR1. There's a knob on the lower left, called the OBS (Omni Bearing Selector). As the name vaguely suggests, it is used to select a bearing. If you turn it, you should see the vertical needle, called the CDI (Course Deviation Indicator) move.[6] Try to center the needle. It should center when the little arrow at the top points to somewhere around 277. That number, and the TO flag means: "Flying at a heading of 277° will lead you directly *to* the station".

That's great, except, according to our route, we don't want to go *to* the station. We actually want to intercept the light blue line labelled "009°" (the "9 degree radial") coming *from* the station. How do we do that? Simple. Set the OBS to 9. When we fly across the radial, the needle will center, and the flag will say FROM. This tells us: "flying at a heading of 009° will lead you directly away *from* the station", which is what we want. At that point we'll turn right to a heading of 009°.

VOR1 OBS ⇒ 009

One final thing – set the heading bug on the directional gyro to our current runway heading (about 310°).

Heading bug ⇒ 310

10.2.3 How High Are We Really?

One effect of our changing the weather conditions is that the barometric pressure is no longer the standard value of 29.92 inHg. Our altimeter needs to know the correct value, otherwise it will report the wrong altitude. This isn't critical at takeoff, but it can make a *huge* difference when descending through the clouds (can you say "controlled flight into terrain"?).

As described in the cross-country flight tutorial, we get the current barometric pressure via ATIS. To recap, click *AI* → *ATC Services in Range*, select our airport, and look up the ATIS frequency (it should be 125.2 MHz). Dial this frequency into COMM1 or COMM2 (remembering to flip the appropriate switch on the audio panel), listen to the ATIS report, and set the altimeter to the given barometric pressure.

We are going to be using the autopilot (see Figures 10.4 and 10.6) to hold an altitude (more on that later), so it also needs to know the barometric pressure. To do so, click the **BARO** button on the autopilot. You should see "29.92" displayed – this is what the autopilot thinks the barometric pressure is. Before the "29.92" disappears (within about 3 seconds), rotate the big dial to change it to the correct value.

[5] Still can't hear it? Check the volume control on the NAV1 receiver. If that has no effect, click *File* → *Sound Configuration* and adjust the settings. If that doesn't work, check the volume on your computer. If that doesn't work, and you have external speakers, adjust the volume on the speakers. And if that doesn't work, check your ears.

[6] The horizontal needle is used in ILS landings, which will be explained later.

10.3 Takeoff

We're ready to take off. There are other preparations that we should have made, but again, in the interests of not overwhelming your brains, I'm only feeding you a bare minimum of information, and feeding it in trickles. This brings us to the most important control you have – the **p** key. Use this often, especially when a new concept is introduced.

Take off; climb on runway heading

Okay. Take off, keeping a heading of 310° for now. Establish a steady rate of climb. We plan to climb to 4000 ft. There's just one problem though – those ugly-looking clouds are standing in our way.

10.4 In the Air

If this is your first attempt at IFR flight, you will find it impossible to fly once you enter the clouds. When you enter the clouds, you will be momentarily disconcerted by the lack of visual cues. "No matter", you then think. "I'll just keep things steady". In a few moments, though, you'll probably notice dials and needles spinning crazily, and without knowing it, you'll be flying upside down, or diving towards the ground, or stalling, or all three.

It takes practice to get used to flying without external visual clues, although it's a skill that you definitely *must* master if you want to fly IFR. For now though, we'll use "George", the autopilot, to make this part of flying easier.

10.4.1 George I

Once you've established a steady rate of climb and heading, engage the autopilot by pressing the **AP** button. You should see "ROL" displayed on the left to show that it's in "roll mode" – it is keeping the wings level. In the middle it will display "VS", to show it is in "vertical speed" mode – it is maintaining a constant vertical speed. On the right it will *momentarily* display that vertical speed (in feet per minute). Initially, the value is your vertical speed at the moment the autopilot is turned on. In the case of Figure 10.6, the autopilot has set the vertical speed to 300 feet per minute.

Figure 10.6: Autopilot after engaging

When you engage the autopilot, CHECK THIS CAREFULLY. It may occur that you engage the autopilot at a time when you had an irrational rate of climb, such as 1800 feet per minute. Our little Cessna cannot sustain this, and if the autopilot tries to maintain this (and it will), you will stall before you can say "Icarus".

We want a vertical speed of around 500 to 700 feet per minute. Hit the up and down (**UP** and **DN**) buttons to adjust the vertical speed to a nice value. Take into account the airspeed as well. We want a sustainable rate of climb.

Engage autopilot; set vertical speed; engage heading mode

Finally, once you're climbing nicely, hit the heading (**HDG**) button. On the display, "ROL" will change to "HDG", and the autopilot will turn the airplane to track the heading bug. Since you set the heading bug to the runway heading, and you took off straight ahead (didn't you?), it shouldn't turn much.

10.4.2 MISON Impossible

It's around 8 nm to the 009 radial intercept, so we've got a bit of time. Since there's no scenery to admire (eg, see Figure 10.7), we might as well prepare for the next phase of the flight.

Figure 10.7: Typical IFR scenery

If you look along our route, just after we intercept the 009 radial and turn north, we pass by a point labelled MISON (see Figure 10.8 for a closeup of that section of the chart without my fat blue and green lines drawn on top. MISON is in the lower right). Just above and to the left of MISON are two crossed arrows. MISON is an intersection. We're actually going to pass east of MISON, but the radial passing roughly from northwest to southeast through MISON (and our route) is of interest to us. We're going to use it to monitor our progress.

Noting our passage of that radial isn't strictly necessary – we can just keep flying along the 009 radial from San Jose until we need to turn. But it's useful for two reasons: First, it's nice to know exactly where we are. Second, it confirms we are where we think we are. If we fly and fly and never cross the radial, alarm bells should start going off.

Looking at the sectional, we see that the radial is the 114 radial from the Oakland VORTAC (VOR TACAN, where TACAN stands for TACtical Air Navigation). Oakland's frequency is 116.8, and its ident is OAK (--- ·- -·-). NAV2 should already be tuned to Oakland, but if it isn't, do it now. Turn on NAV2 in the audio panel and make sure you're getting the correct

NAV2 ⇒ 116.8

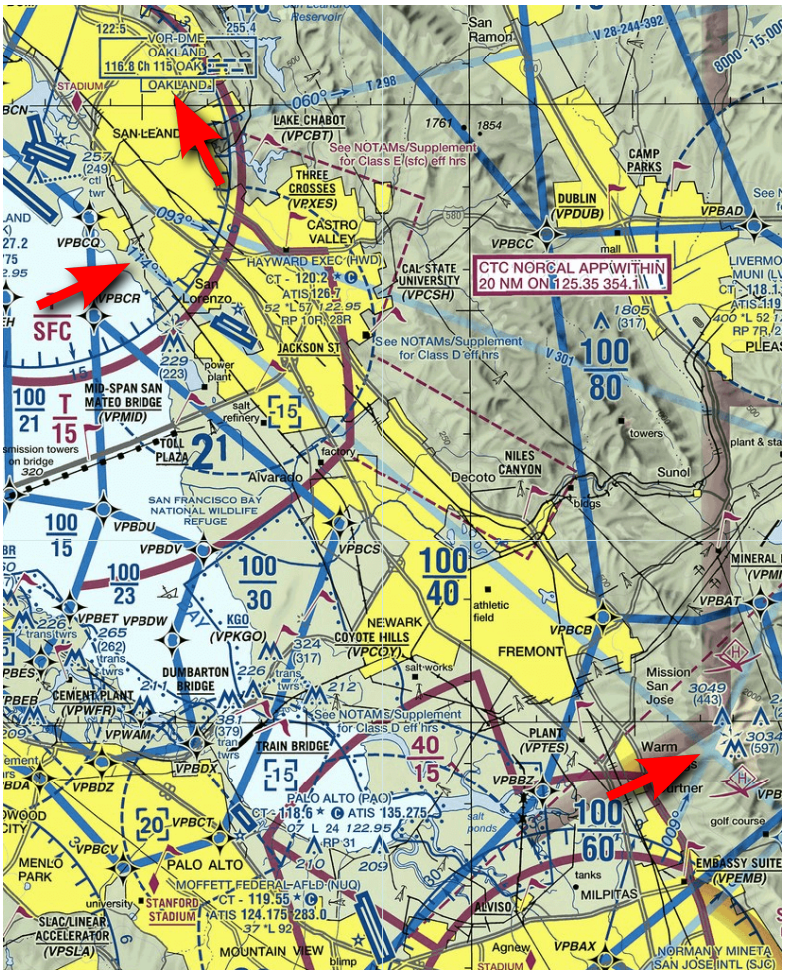

Figure 10.8: Oakland VOR and 114 radial to MISON intersection

ident.

We need to adjust the OBS, to tell VOR2 which radial we're interested in. Set the OBS to 114.[7] See if you can guess whether the flag should read TO or FROM when we cross the 114 radial. And see if you can guess whether the needle will move from left to right or right to left as we cross the radial.

VOR2 OBS ⇒ 114

A final note: For our purposes, there's nothing magical about the 114 radial – we could have used 113, or 115, or 100, or 090. The reason I chose 114 is because there was a line on the map already drawn along the 114 radial, which saved me the trouble of drawing a line myself.

10.4.3 George II

As we continue towards the 009 radial intercept, let's look a bit more closely at the autopilot. First of all, if you aren't in the habit of trimming the airplane, you'll probably notice a flashing "PT" with an arrow on the autopilot. This means you need to trim the plane, which the autopilot will do for you automatically. The arrows indicate the trim direction, the up arrow – nose up, down arrow – nose down.

Also, on the right there's a big knob, the altitude select knob, which we can use to dial in a target altitude. We're going to use it. Turn it until you see our desired cruising altitude, 4000 ft, displayed on the right. When you started turning it, "ALT ARM" should have appeared in the autopilot display (as in Figure 10.9). This indicates that you've selected a target altitude. The autopilot will maintain the current rate of climb until reaching that altitude, at which point it will level off and change from vertical speed (VS) mode to altitude hold (ALT) mode. In altitude hold mode it maintains an altitude (in this case our target altitude of 4000 ft).[8] It will also politely beep 5 times when you cross 3000 ft to remind you that you're within 1000 ft the armed altitude.

Set autopilot altitude to 4000

Figure 10.9: Autopilot with altitude armed

Don't forget that the autopilot won't adjust the throttle, so when it levels out, the airplane (and engine) will speed up. You'll need to adjust the throttle to get a proper cruise.

10.4.4 Staying the Course

At some point you'll intercept the 009 radial (the VOR1 needle will centre). Turn to a heading of

[7] If you get tired of clicking on the knobs, much of this can be done more easily using the *Equipment → Radio Settings* dialog.

[8] Of course, you don't really have to do this – you could just watch the altimeter, and when it gets to 4000 ft, reduce the vertical speed to 0, or press the ALT button to enter altitude hold mode. But by using the altitude select knob, we've demystified one more mystery button.

009. You can do this using the heading bug on the directional gyro if you're using the autopilot. Turn to 009° upon VOR1 intercept

Unless you're good or lucky, the needle probably won't be centered. We need to adjust our course. The CDI needle (the vertical needle on the VOR) tells us where to go. If it's to the left, that means the radial is to the left, so we need to go left. Ditto for right.

It's quite easy in theory, although in practice you may find that it's hard to keep the needle centered, and that you are slaloming down the radial. The key is to realize this: the *position* of the needle tells us where we *are*, the *motion* of the needle tells us what to *do*.

I'll explain. If the needle is to our left, then, yes, the radial is definitely to our left.[9] But if the needle is *moving* towards us, that means we're going to cross the radial, sooner or later, so our situation is improving, and we probably just need to wait for the needle to center. On the other hand, if the needle is *moving* away, we need to turn towards it to stop, and reverse, its motion.

Note that the amount we need to turn is difficult to guess correctly at first, so experiment. Try 10°. If the needle moves too fast, cut it down to 5° (ie, turn back 5°). If, on the other hand, the needle moves too slowly, double it to 20° (ie, add another 10°), and see what happens.

10.4.5 Yet More Cross-Checks

Cross-checking your position is always a good thing. The intersection with the Oakland 114 radial is one way. Ahead of that lies the SUNOL intersection. If you look closely, 5 separate radials join at the point, so we have an embarrassment of choices with regards to the intersecting radial. Because it will come in useful later, we're going to use the one coming in from the upper right. Another check of the sectional reveals that this is the 229 radial of the Manteca VORTAC, 116.0 MHz, ident ECA (· -·-· ·-). Cross OAK 114 radial

You should know the drill by now: Tune NAV2 to 116.0, set the OBS to 229, and check the ident to confirm the station. NAV2 ⇒ 116.0
VOR2 OBS ⇒ 229

Meanwhile, let's introduce another piece of gear on the panel that will cross-check the SUNOL passage. Some VOR stations have a distance capability, called DME (Distance Measuring Equipment)[10]. For example, San Jose does (remember it's a VOR-DME station), as do Oakland and Manteca (VORTACs have DME capabilities).

Using DME, you can find out how far you are, in straight-line distance, from the VOR station. In our scenario, the DME isn't necessary, but we'll use it anyway, just to see how it works, and to reconfirm our position.

The DME is the instrument below the autopilot (refer to Figure 10.4). Make sure it's turned on. The selector to the left of the on/off switch is probably set to N1, where "N1" means "listen to NAV1". Since NAV1 is tuned to San Jose, it's telling us the distance to the San Jose VOR-DME. Switch the DME to N2. It now shows us the distance to the Manteca VOR. DME ⇒ N2

The DME shows you 3 things: the distance in nautical miles to the station, your speed towards or away from the station, and estimated time to the station at the current speed. Note that the distance is the direct distance from your plane to the station (called the "slant distance"), not the ground distance. Note as well that the speed is relative to the station, so unless you're flying directly to or from the station, it will probably be lower than your true groundspeed.

[9] Unless you're heading in the opposite direction, but that's another story.

[10] See https://en.wikipedia.org/wiki/Distance_Measuring_Equipment for more information.

For example, the speed from San Jose, which is directly behind us, should be greater than the speed towards Manteca, which is off to the right.

If we look up information about the SUNOL intersection,[11] it tells us that it is 33.35 nm (as measured by a DME receiver) from ECA on the 229.00 radial (that's what "ECAr229.00/33.35" means).

Now we have two ways to confirm the SUNOL intersection: The VOR2 needle will center, and the DME will read 33.4 or so. Note that the DME doesn't provide us with a very precise fix here because Manteca is at such an oblique angle. But it does give us a good warning of SUNOL's impending arrival. Moreover, if it has an unexpected value (like 30), it should raise a few alarm bells.

You may be wondering what "HLD" means (the setting between N1 and N2 on the DME). It stands for "hold", and means "retain the current frequency, regardless of whether NAV1 or NAV2 are retuned". For example, if we switch from N2 to HLD, the DME will continue to display (and update) information to Manteca. Even if we retune NAV2, the DME will remain tuned to Manteca. This is handy, because it basically represents a third independent receiver, and in IFR flight two receivers just never seem like enough.

10.5 Getting Down

We're getting close to SUNOL, flying along the 009 radial from San Jose, monitoring our position with the DME. At SUNOL we'll be less than 5 nm from Livermore, somewhere down there in the clouds. Perhaps if we just descended to 700 ft or so (Livermore is at 400, the ceiling is at 750) and headed more or less directly north after SUNOL, we'd get there? A recipe for disaster my friend, and you know it.

10.5.1 Instrument Approach Procedures

As you recall from the previous tutorial, when flying VFR, you don't just point your airplane to the nearest runway to land. You need to fly a pattern. This helps you line up, and helps prevent planes from crashing into one another, which is a Good Thing.

Similarly with IFR landings. There's a procedure to follow. In fact, there are *procedures* to follow. Because of the complexity of landing in IFR conditions, there's no single procedure for all airports. You need to check for your particular airport. In fact, you usually need to check for your particular airport, runway, and navigation equipment.

Our airport is Livermore (KLVK). Let's check the information for that airport. Go to https://www.airnav.com/airport/KLVK to see what they've got. Down near the bottom, we have IAPs (Instrument Approach Procedures). There are three listed for runway 25R. The first is an ILS (Instrument Landing System) approach, the second is a GPS (Global Positioning System) approach and the third is a LOC (Localizer). For this tutorial, let's choose the ILS approach (which I'll explain later).

Although Livermore only has three different instrument approach procedures, big airports have many many more. If you look at nearby San Francisco, you'll see they have a *slew* of

[11] For example, from https://www.airnav.com/airspace/fix/SUNOL.

procedures. There are ILS procedures, GPS procedures, LDA (Localizer-type Directional Aid) procedures, VOR procedures, … I wouldn't be surprised if they had a procedure for someone with a sextant and an hourglass in there. To learn IFR flight, you'll need to master all of them.

Back to Livermore. If you download the procedure, you'll see something like Figure 10.10 (except for the colour). It's pretty overwhelming at first – it compresses a lot of information in a small space. We'll ignore as much as we can, restricting ourselves to the three parts that have been coloured in. And we'll do those parts on a "need to know" basis – we'll only look at them when we really have to.

Where to start? At the beginning of course. An IAP will have one or more Initial Approach Fixes (IAFs). These are your entry points to the approach procedure and can be found in the "plan view", which I've coloured purple in Figure 10.10. Our IAP lists two, one in the middle and one on the right (see Figure 10.11 for a close-up).

An IAF is a *fix*, and a fix is an identifiable point in space. In fact, we've already encountered another kind of fix, namely a VOR intersection. Fixes are also usually named (eg, MISON, SUNOL). The IAF on the right is named TRACY, and consists of a radial, a distance, and an altitude. Specifically, it's 15 DME (15 nm as measured by a DME receiver) along the 229 radial from the ECA (ie, Manteca) VOR.

10.5.2 Non-directional Beacons

However, we're not going to use TRACY as our IAF. We're going to use the IAF in the middle, which is a marker (LOM stands for Locator Outer Marker). We'll worry about what an outer marker is later. For now let's concentrate on the locator part. The locator in an LOM is an NDB (Non-Directional Beacon)[12]. It's a bit like a VOR, in that it can be used to determine your heading and navigate from place to place. Like a VOR, it has a name (REIGA, in this case), a frequency (374 kHz), and an ident (LV, or ·–·· ···– in Morse). NDBs also appear on sectionals, as fuzzy red circles with a small circle in the middle, with their identification information placed in a red box nearby. (see Figure 10.12 for a closeup. Don't confuse the NDB, which is fuzzy, with the solid red circle on the left, nor the circle below with the "R" inside).

An NDB station basically broadcasts a signal that says "I'm over here", and the receiver on the plane can receive that signal and tell you, the pilot, "the station is over there". You just need to tune the receiver and monitor the correct instruments. The receiver, labelled ADF (Automatic Direction Finder) Receiver, and the corresponding instrument, also labelled ADF, are shown in Figure 10.4.

To tune into REIGA, turn the tuning knob on the receiver until 374 is displayed as the standby (STDBY) frequency. Use the big knob to change the frequency every 100 kHz, and the small knob to change the frequency every 1 kHz. Additionally, as always, you can hold down the **`Shift`** key to increase the pitch, respectively every 600 kHz for a large knob and every 10 kHz for a small knob. Then hit the swap button (labelled "FRQ"). The 374 is now displayed as the selected (USE) frequency. The needle on the ADF should swing around, eventually pointing ahead to the right, to REIGA. But it might not. Why? Because the receiver might be in antenna

ADF ⇒ 374

[12] See https://en.wikipedia.org/wiki/Non-directional_beacon for more information.

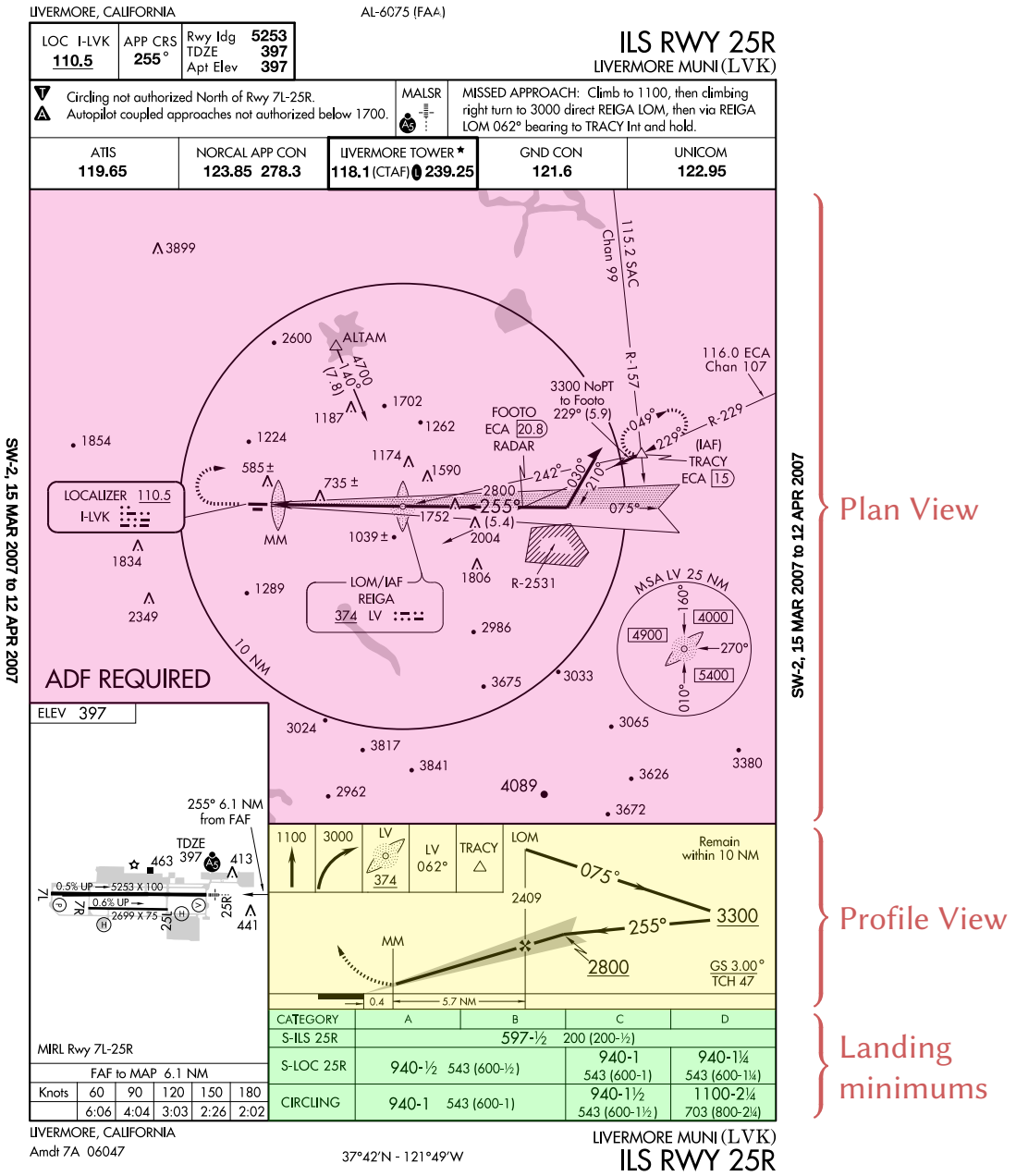

Figure 10.10: ILS approach plate for Livermore runway 25R

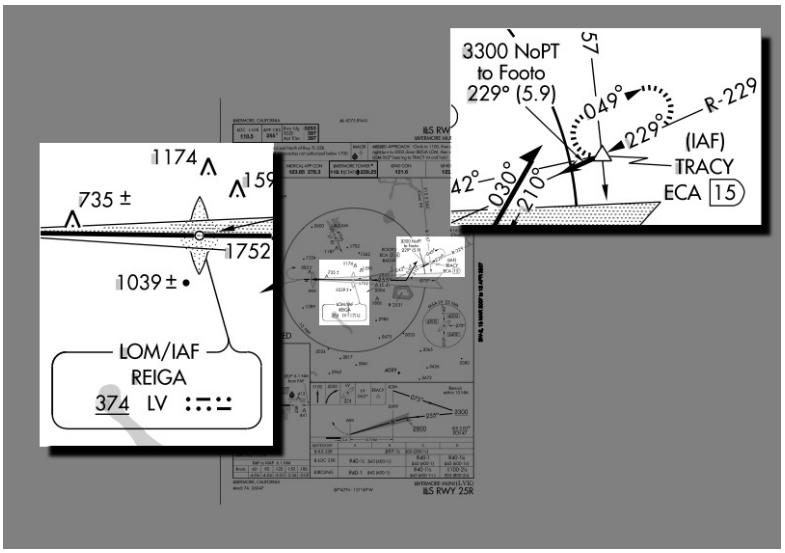

Figure 10.11: Initial approach fixes

Figure 10.12: REIGA non-directional beacon

mode (as show by the "ANT" in the upper-left portion of the display).[13] If it is in antenna mode, hit the **ADF** button so that "ADF" shows. Now the needle should swing to point to REIGA. Like VORs, to be sure we're really tuned into the right station, we need to hear the ident as well, so hit the ADF switch on the audio panel and check.

Notice there's no OBS to set for an ADF – the needle just points to the station, which is nice. This leads us to our first rule for ADFs:

> **ADF Rule 1:** The needle points to the station.

Pretty simple. In fact, you may not think it merits a "rule", but it's important to emphasize the difference between ADFs and VORs. A VOR, remember, tracks a single radial, which you specify by turning the OBS. An ADF has a knob, and a identical-looking compass card, so it's tempting to believe it acts the same way. It doesn't. Turn the ADF heading knob (labelled "HDG") and see what happens. The compass card moves, but the arrow doesn't. It just *points to the station.*

In our current situation, where we just want to fly to REIGA, that's all we need to know to use the ADF. If the needle points "over there", then we'll fly "over there", and eventually we'll pass over REIGA. However, for the sake of practice, and because it will be necessary later, I'm going to give the second rule for ADFs, which explains what the compass card is there for:

> **ADF Rule 2:** *If* the compass card reflects our current heading, then the needle gives the bearing *to* the station.

In other words, the compass card gives "over there" a number.

Now we're ready to head to REIGA. Rotate the ADF heading knob until our current heading is at the top (basically, the ADF should match the directional gyro). When we pass the SUNOL intersection, look at the ADF needle, and set the HDG bug to that heading (I assume you're using the autopilot. If not, just turn to that heading). At the end of the turn, the ADF needle should point straight ahead. And if it doesn't, adjust your heading so that it does.[14]

Cross SUNOL; turn to REIGA

By the way, the closer you get to REIGA, the more sensitive the needle becomes to changes in your heading. Don't go crazy trying to keep the needle centered as you get close. Maintain a steady heading, and get ready for the …

10.5.3 Procedure Turn

So, once we hit REIGA, do we just turn left and head down to the runway? Ah, if only life were so simple. No, we turn right, *away* from the airport, and do a *procedure turn.* We know there's a procedure turn because of the barbed arrow in the plan view (see Figure 10.13). As you can see if you follow the arrow, we need to fly away, on a heading of 075°, then turn left 45° to a heading of 030°. We do a U-turn (to the right, *away* from the airport – that's one of the rules about procedure turns) to come back at 210°, then a 45° right turn to 255°, heading straight towards the runway. All of this turning gives us time to set ourselves correctly on course, at the right altitude, to land on 25R.

[13] Antenna mode, by the way, is usually used to ident an NDB, because it gives better audio reception. While in antenna mode, however, the ADF will *not* point to the station – the needle will be parked pointing directly right.

[14] Which is actually bad technique in the presence of a crosswind, but I'm ignoring the wind to simplify the tutorial.

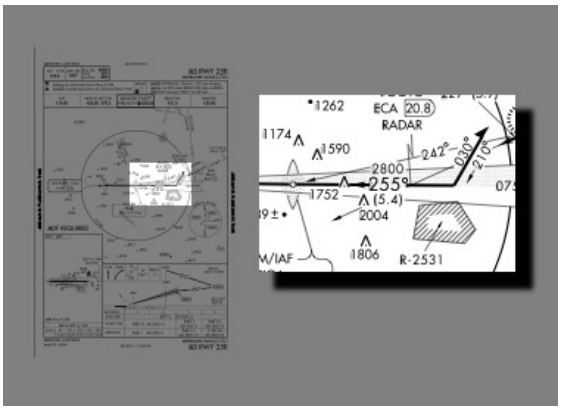

Figure 10.13: Livermore ILS procedure turn

Hmmm. I mentioned "right altitude", but how do we know that is? That's down below, in the profile view (the yellow part of Figure 10.10). You can see that at the top is the LOM, our IAF. Now follow the arrows. After the IAF, we head out at 075°. During the procedure turn we can descend to 3300 ft, but *no lower* (that's what the line *under* the 3300 means). After we finish our procedure turn and are heading back at 255°, we can descend to 2800 ft, but *no lower*, until we intercept the glide slope.

One thing the instrument approach procedure does *not* tell you is the length of the procedure turn. The only constraint is that you must not fly more than 10 nm away from the NDB. You'll notice there's a 10 nm circle drawn around it in the plan view, and a note in the profile view saying "Remain within 10 NM". They're not kidding. So, since we fly at around 110 knots, two minutes on each leg is reasonable – two minutes at 075°, and two minutes at 030°. On the way back we don't care about times – we just want to intercept 255°.

So, after we pass REIGA, turn right to 075°. Our ADF receiver has a built-in timer, so we'll use that to time our two-minute leg. Hit the "FLT/ET" (flight time/elapsed time) button. The "FRQ" in the middle of the display will disappear, "FLT" will appear on the right, and the standby frequency will be replaced by a time. This is the total flight time, and cannot be changed, except by cycling the power. Hit "FLT/ET" again. Now you'll see "ET" displayed, and a time, probably the same as the flight time. To reset the elapsed time, hit the next switch, labelled "SET/RST". The timer should reset to 0, then start counting up (see Figure 10.14).[15] Each time you hit "SET/RST", the elapsed time will resets to 0. If you want to see the standby frequency again, hit "FRQ" once. The timers will continue to run.

10.5.4 Chasing the Needle

When we approached REIGA, we weren't particularly concerned about our course – we just aimed for REIGA. Now, however, our course is important. We want to be flying directly away from REIGA *on a course of* 075°.

Cross REIGA;
Fly at 075° away from REIGA for two minutes

[15] The timer can also be set to count down from a time you specify – except that feature has not yet been implemented.

Figure 10.14: ADF with timer running

Now, in an ideal world, after we turned to 075°, the ADF needle would be pointed directly behind you (ie, we'd be on course). Probably it isn't, so we need to adjust our course. The key to adjusting our course is ADF Rule 2. If we've set the compass card correctly, then the needle shows us the current NDB bearing. If we turn and fly until we intercept the 255 bearing, then turn to 075°, we'll be right on course.

Figure 10.15 shows what I mean. In the figure, the plane, flying along the green line, is initially off course.[16] The heading is correct, 075°, but the station is at 225°, not 255°. To correct this, we turn right (remembering to adjust the ADF compass card to match our new heading). As we fly on this new heading, we get closer to the correct position, crossing the 235 and 245 bearings (shown in red). Finally, when we the ADF needle points to 255°, we turn left to 075°, and readjust the ADF compass card.[17] We are now on course.

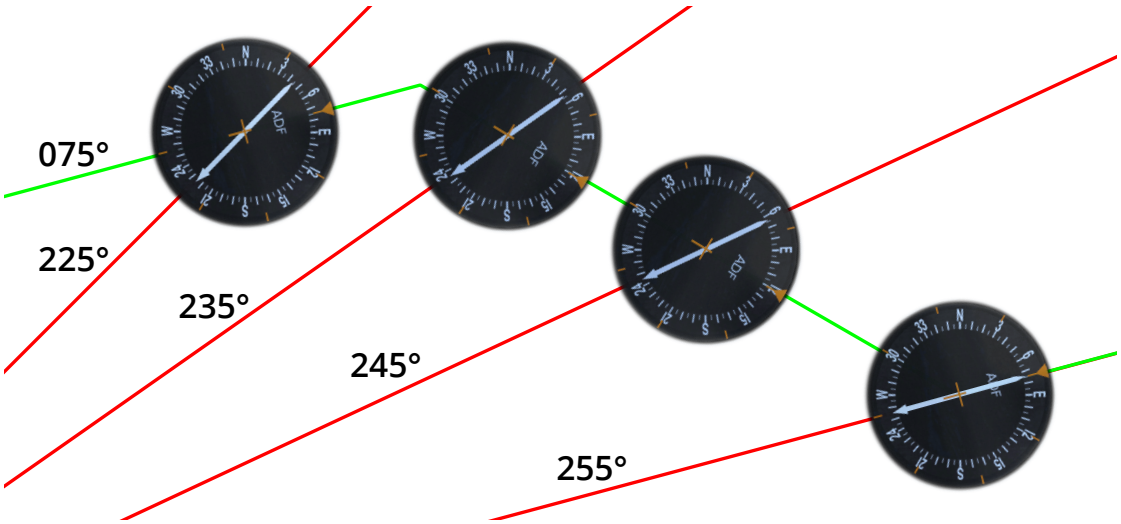

Figure 10.15: Getting back on course

Of course, even when you get back on track, that won't be the end of the story. Your airplane drifts; your mind drifts; your compass drifts; the wind pushes you around. What you find is that you will be constantly making little corrections. That's okay, as long as we're close. And anyway, before long (2 minutes actually), we'll turn left 45° to 030° as part of our procedure

[16] *Way* off course, actually. I've exaggerated the angles to make the explanation clearer.

[17] You might be thinking "Wouldn't it be nice if there was an ADF where the compass card rotated automatically?" Well, such an ADF does exist, and it comes with its own acronym – RMI (Radio Magnetic Indicator).

turn, at which point we'll just ignore the NDB anyway. Sigh. All that effort for just 2 minutes. Hardly seems worth it.

10.5.5 FOOTO Time

While you're flying outbound, take an occasional look at VOR2, tuned to Manteca, and the DME. Assuming the OBS is still at 229, and the DME still tuned to N2, at some point the needle should center, meaning you've crossed the 229 radial, and, if you're on course, at the same time the DME should read 20.8. How do I know that? If you look at the approach plate (Figure 10.10), you'll notice an intersection, named FOOTO. FOOTO is on the approach, and is defined to be 20.8 DME from ECA. Although this intersection is not strictly necessary for us, it comes for free, and provides good confirmation of our position both outbound and, later, inbound.

Depending on how fast you're flying, you'll probably pass FOOTO close to the time your two minutes at 075° are up. At the end of two minutes, turn left 45° to 030°. Reset the timer, and fly for another two minutes on this heading.

10.5.6 George III

This leg is relatively uneventful, so we'll take advantage of the lull in the action to descend to 3300 ft. Before descending, check the KLVK ATIS (it should be 119.65 MHz) and make sure your altimeter is correct.

Turn left to 030°; fly for two minutes while descending to 3300

Assuming you're using the autopilot, you will need to do a few things to descend:

1. If you're in altitude hold (ALT) mode, you need to get back into vertical speed (VS) mode. Press the **ALT** button – the "ALT" in the middle of the display should change to "VS", and your current vertical speed (probably 0) should be displayed momentarily on the right.

2. Click the **DN** button until you get a vertical speed of -500 feet per minute.

3. If you want to set the target altitude, like before, rotate the big knob on the right until "3300" shows up on the right side of the display. "ALT ARM" should appear on the bottom.

Note that if you're using the autopilot to descend, it will just push the nose down, like a bad pilot, so the airplane will speed up. We want to go down, but we don't want to speed up, so we need to reduce engine RPMs to keep the speed at 110 knots. Later, when you level off at 3300 ft, you'll have to increase power again.

If you're flying manually, then you just need to adjust the engine to get the descent rate you want – the plane should stay magically at 110 knots if it's already trimmed for 110.

10.5.7 ILS Landings

While descending, we also need to start considering how we're going to intercept 255° on the way back and follow it down to the runway. You might think we're going to use the NDB like we did on the outbound leg, but at this point, the NDB is not good enough. This is an ILS

landing, a so-called "precision" landing, and an NDB is just not precise enough. It can get us close to the runway, but not close enough.

So, we're going to switch over to our ILS system. It is much more accurate horizontally. As well, it offers vertical guidance, something which the NDB does not give at all. And hey, it also gives you something else to learn in our few remaining minutes so that you don't get bored.

As with NDB and VOR navigation, the ILS system[18] has a transmitter (or *transmitters* – a localizer *and* a glide slope) on the ground, and a receiver and a gauge in the aircraft. The receiver, it turns out, is just a NAV receiver, of which we have two. The gauge is like a VOR indicator, but it has an added glide slope indicator, which is a horizontal (you hope) needle. Like a VOR, the vertical needle shows whether you're left or right. The horizontal needle shows whether you're high or low. Our ILS gauge is our old friend VOR1.

As you might have guessed, the localizer has a frequency and ident associated with it (there's no need to tune the glide slope separately. If you tune the localizer, you've tuned the glide slope). This is shown on the approach plate in two places: at the top left corner, and in the plan view by the runway (see Figure 10.16). As we can see, the frequency is 110.5 MHz, and the ident is I-LVK (·· ·–·· ···– –·–).

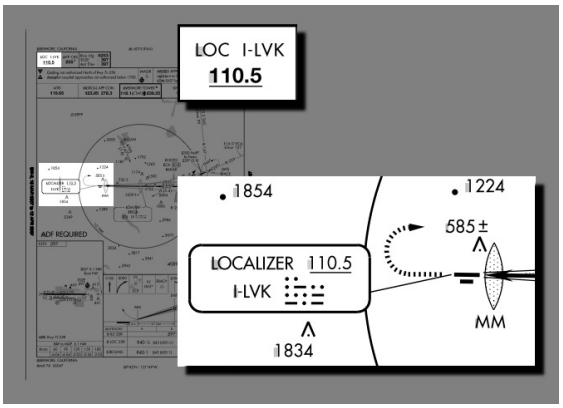

Figure 10.16: Livermore 25R localizer

NAV1 ⇒ 110.5

If you look at VOR1 now, it should be showing a red "GS" flag (this can be seen in Figure 10.5). This indicates that there is no glideslope signal. Now tune NAV1 to 110.5. The red "GS" flag should disappear. Check for the ident. Sounds lovely, doesn't it? That localizer is going to save your bacon and get you out of this interminable soup. When you tuned into the localizer, you'll also have noticed the ILS needles move. And the OBS? Well, it's useless. Try moving it. No matter how you turn it, the needles don't move in response. That's by design. A localizer is basically a VOR with *one* radial, the approach heading. We don't care about any others, so we don't need an OBS to declare interest in any others. However, it does serve as a useful reminder, so move the OBS to 255, our desired heading.

VOR1 OBS ⇒ 255

[18] See https://en.wikipedia.org/wiki/Instrument_Landing_System for more information.

10.5.8 Intercepting the Localizer

We're now ready to intercept the ILS localizer. When the two minutes on the 030° leg have passed, make your U-turn to the right to 210°. Soon after you complete your turn, the vertical (localizer) needle on the ILS will begin to move. And it will move *fast*, much faster than the ADF and VOR needles did. A localizer is 4 times as sensitive as a VOR, relatively small movements of the aircraft make big changes in the needles. You'll probably overshoot, but don't worry, because we have around 5 or 10 minutes to get things straightened out.

Turn right 180° to 210°

Intercept localizer

Just remember: don't chase the needles. That mantra is now more important than ever. Those needles are sensitive – if you just turn left when the localizer needle is to the left and right when it's to the right, you'll be flying like a drunken sailor. If you're lucky, the runway will be passing underneath you as you swing across the track for the umpteenth time. Luck, though, is something we should not be relying on. Determine on how the needles are moving before making your move.

Now that you're heading back inbound at 255°, slow to 75 knots, drop a notch of flaps, and descend to 2800 ft (but no lower). And check for the inbound passage of FOOTO to confirm your position. And pat your head and rub your stomach.

Slow to 75 knots; drop a notch of flaps; descend to 2800

10.5.9 Intercepting the Glide Slope

As we fly towards the runway, don't forget to look at the horizontal needle, the glide-slope needle. When we intercepted the localizer, it should have been high above us, because we were actually under the glide slope. As we levelled out at 2800, the glide slope started coming "down" to us. Eventually, you should see the needle start to move down. When the needle is horizontal, that means you're on the glide slope.[19] And, soon after we intercept the glide slope, we should pass over the outer marker. Several things will happen more or less simultaneously, all of which confirm your position:

1. You'll hear a continuous series of dashes.
2. The blue light labelled "O" above COMM1 will flash.
3. The ADF needle will swing around.

Once on the glideslope, we need to start descending. What's a good rate? It depends on our groundspeed. In our case, we're going at 75 knots (there's almost no wind, so our airspeed and groundspeed are the same), and it turns out that we need to descend at around 400 feet per minute. With the autopilot, that's pretty easy – just dial in -400, and you're set (but remember to reduce power to keep our speed at 75 knots, or you'll hit the runway going pretty fast, and be prepared to adjust things if you drift above or below the glide slope).

Intercept glideslope; cross outer marker; drop second notch of flaps

Without the autopilot, it's also pretty easy – just reduce power. How much? In this case, with our plane, to around 1700 RPM. Again, it depends on many things – plane, elevation, winds, weight, ..., so you'll have to adjust things if you see the glide-slope needle start to

[19] Maybe. There can be false glideslopes, and FlightGear models these, so we have to make sure we're on the real one. One purpose of the procedure turn is to get you in the correct position, at the correct altitude, to intercept the true glideslope.

move up or down. Like the localizer needle though, … (are you ready?) DON'T CHASE IT. Watch how it's moving, then make your adjustment.

Since we're on final approach, you might want to drop a second notch of flaps. This will affect your trim, and you'll have to adjust power a bit as well.

10.5.10 Touchdown, Almost

After all the excitement of the procedure turn, it will seem like a long way down to the runway from the outer marker. There's not much to do but stare at those needles. In fact, you'll probably stare at them like you've never stared at them before. Take a look around at the other gauges too, though – they have useful things to tell you. Is our airspeed okay? We don't want to stall. RPMs about right? If flying manually, you'll want to constantly check the attitude indicator and directional gyro. This being a simulator, we don't have to worry about oil pressure and engine temperature, but you might want to glance over there anyway, just to get into the habit. And I hope you've done things like set the mixture to full rich (you did lean it out while cruising, didn't you?). If you want, you can lower the flaps completely as you get closer.

10.5.11 A Confession

I've actually made you do more work than you have to. We've been using the autopilot as a fancy steering wheel, but it's capable of more than that. You may have noticed that the autopilot has some buttons I haven't explained – NAV, APR, and REV. Well, using those buttons, the autopilot can:

NAV: Track a VOR radial or localizer.

APR: Do a direct ILS approach, tracking both the localizer *and* the glideslope.

REV: Intercept the ILS before the procedure turn (ie, head *away* from the localizer.

So, in fact, even more of the work you've done could have been done by the autopilot. After takeoff, you could have asked it to track the 009 radial from SJC all the way to SUNOL in NAV mode; at SUNOL, you could have asked it to fly the "back-course approach" from I-LVK in REV mode; done the procedure turn in HDG mode; finally, tracked the localizer and glideslope in APR mode.

However, I didn't give you this information for two reasons. First, flying by hand (even with the autopilot gently holding your hand, as we've been doing) gives you a better idea of what's happening. Second, the autopilot doesn't behave quite as the official manual says it should for some of these functions – best stick to the features that are known to work well.

10.5.12 Touchdown, Not

Although ILS approaches can get us close to the runway, closer than VFR, NDB, or VOR approaches can, we still need *some* visibility to land,[20] so we need a way to decide if landing is possible or not. That's what the landing minimums section of the procedure plate is for

[20] Well, unless it's a Category IIIC ILS approach.

(coloured green in Figure 10.10). In the category labelled "S-ILS 25R" (that's us), you'll see "597-1/2 200(200-1/2)". This tells us that we can track the glide slope down to an altitude of 597 feet (200 ft above the runway). At 597 ft we make our decision – if we can't see the runway, then we have to execute a missed approach. 597 ft is our *Decision Height* (DH).

In addition to the altimeter, this particular approach also has another indication that we're close – a Middle Marker (MM). This marker will sound – in this case, a dot dash series – and the yellow light labelled "M" above COMM1 will flash. Passage over the middle marker should coincide with reaching decision height.[21]

So, what if you can't see the runway at decision height? As you might have expected, just as you can't land willy-nilly, you can't just go around willy-nilly. There's a Procedure. A Missed Approach Procedure. This is shown in several places on the approach plate (see Figure 10.17): At the top, where it says "MISSED APPROACH", in the plan view, where you can see a dashed arrow coming off the end of the runway and a dashed oval on the right, and in the profile view, where a series of boxes shows graphically what to do. In our case, these all tell us to:

1. Climb straight ahead to 1100 ft
2. Make a climbing right turn to 3000 ft
3. Fly to REIGA
4. Fly outbound from REIGA at 062°
5. Fly a holding pattern at the TRACY intersection

The holding pattern, as you might have guessed, is a place where you can "park" while sorting things out, and has its own set of procedures and techniques which we won't go into here, because …

10.5.13 Touchdown

In our ideal simulator world, you probably won't have to execute a missed approach. Assuming you stayed on the glide slope, you should have popped out of the murk at the decision height, and with 800 metre visibility, the runway should have been in view soon after. With the runway in sight, you could then turn wildly to get on course[22] (it's very hard to be lined up perfectly) and land "normally" (which for me involves a lot of bouncing around and cursing). Park the plane, then stagger out of the cockpit and have another hamburger!

Sight runway; disengage autopilot; cross mid-marker

Land; eat hamburger

[21] As you may have guessed, the remaining light – white, and labelled "A" – indicates passage of the inner marker. Our approach doesn't have one, but San Fransisco's runway 28R does. While passing over it, you should hear a rapid, high-pitched series of dots. Why is it labelled "A" and not "I"? Because in ancient times, it was also used to identify passage over "airway" markers along radio range tracks.

[22] Remembering, of course, to disengage the autopilot.

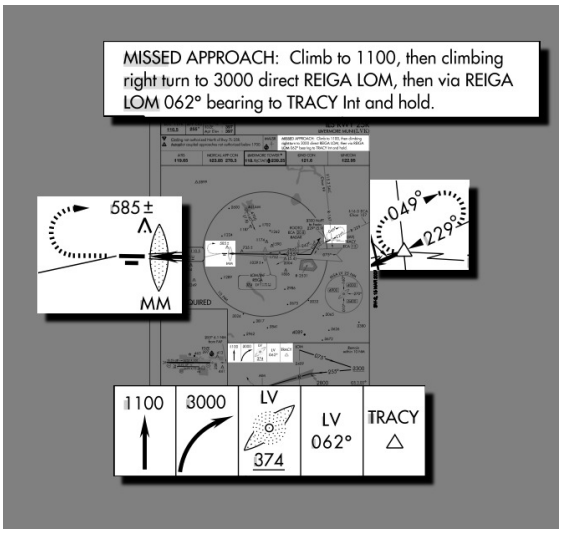

Figure 10.17: Missed approach procedure

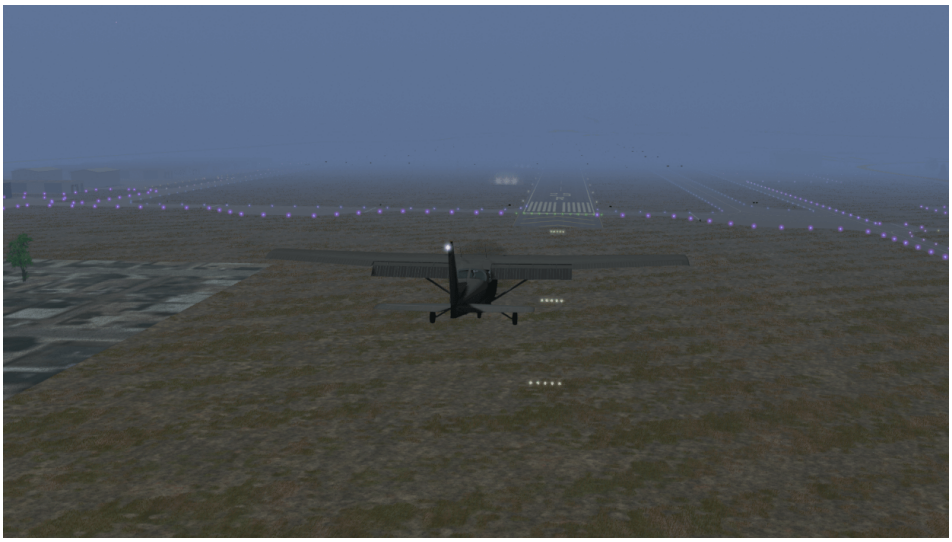

Figure 10.18: On course, runway in view. We're going to live!

10.6 Epilogue

That was a lot of information in a short time, a rather brutal introduction to IFR flying. Hopefully, instead of turning you off, it has whetted your appetite for more, because there *is* more. Some of the major issues I've ignored are:

Wind This is a big one. Flying IFR in a crosswind affects everything you do, and you need to be aware of it or your navigation will suffer.

Flying without the autopilot George tries his best, but he's not completely trustworthy. You have to be prepared to go it alone.

DG precession The Directional Gyro (DG) in the C172P is not perfect. Over time, the values it gives you are less and less reliable – it *precesses*. It needs to be periodically calibrated against the compass (using the OBS knob on the DG to adjust it).

IFR charts We used sectionals, which are really intended for VFR flight. There are a whole set of charts devoted exclusively to IFR flight.

ATC The other people out there need to know what you're doing. As well, they'll probably tell you what to do, including to ignore the approach plate you so fastidiously studied.

SIDs/DPs, Airways, and STARs This tutorial introduced IAPs, which are standard ways to make approaches. In IFR flight, there are standard ways to *leave* airports (Standard Instrument Departures – SIDs, or Departure Procedures – DPs), standard ways to travel *between* airports (airways), and standard ways to go from airways to IAPs (Standard Terminal Arrival Routes – STARs).

Holding Patterns Most missed approaches end in a holding pattern somewhere, so you'd better know how to fly them.

GPS Nowadays most small aircraft are equipped with GPS, which is rapidly replacing radio-based navaids.

If you want to learn more, try the following resources:

- *Flight Simulator Navigation*, written by Charles Wood. It covers everything from basic navigation to ILS approaches, with lots of examples and practice flights to improve your skills. Everything is linked together by an entertaining storyline in which you are the pilot for a fictional charter service.

  Two caveats, though. First, it is Microsoft Flight Simulator 98 based, so you'll have to translate into "FlightGear-ese" as appropriate. Second, it is a bit out of date, and things in the real world have changed since it was written. NDB beacons have been decommissioned, new approaches have replaced old ones – even an airport has disappeared (!). Treat this as a learning opportunity. You'll get better at finding more up to date information, and learn not to blindly trust your charts, just as you have learned not to blindly trust your instruments.

- If you're *really* keen and want to hear it straight from the horse's mouth, there's the official *FAA Instrument Flying Handbook*. It's big and detailed, and there's *no* interesting storyline in which you're a pilot for a fictional charter service. More documents can be found at their *Aviation Handbooks & Manuals* page.

- If you'd like practice deciphering what the instruments are telling you, without the bother flying (or even virtual flying), you can try *luizmonteiro.com*, which has Flash tutorials of various instruments, including a VOR and an ADF.

- Another simulated instrument site is *Fergo IFR Simulator*, where you can learn navigation by playing NAVAIDs finding missions.

- If it's navigation information you're after, an excellent site is *AirNav.Com*, which I've used extensively in the course of this tutorial. It has detailed airport, navaid, and fix information, and links to IAPs. Unfortunately, the information is only for the USA.

- *FlightSim.Com* has a very informative series of articles entitled "How To … Use Approach Plates". It starts with a *very, very* dense tutorial on how to read an approach plate, then follows with a set of approaches at Kodiak, Alaska. These are an excellent supplement to the approaches given in Charles Wood's *Flight Simulator Navigation* (see above).

  Most interesting, though, is section two – "Dangerous Approaches." Approaches at six airports around the world, from Penticton, BC to Kathmandu, Nepal, are described. Fly them if you dare!

  Warning – this series is even more Microsoft Flight Simulator (2000) focused than Charles Wood's, and some of it is out of date (some outside links are broken, and some of the approaches have changed).

- Also from *FlightSim.Com* is "Golden Argosy", a description of a flight from New York to Rome by Tony Vallillo, an American Airlines 767 captain. It gives some interesting information about navigation that doesn't appear in the other sites mentioned here, such as the North Atlantic Tracks. However, its main appeal is that it gives a good answer to the question "What's it *really* like to be a pilot?" The author's love of flying is evident throughout the article.

- For those who are interested in the ATC side of things, and want information from an authoritative source, check out Michael Oxner's "Aviation Topic of the Week", a series of articles about flying "in many types of airspaces in many situations." Michael Oxner is a professional controller and private pilot who obviously can't get enough of airplanes, because in his spare time he's also an on-line controller with VATSIM. Particularly interesting are a set of articles describing a complete IFR flight and a complete VFR flight.

Chapter 11

A Helicopter Tutorial

11.1 Preface

First: in principle everything that applies to real helicopters, applies to FlightGear. Fundamental maneuvers are well described here: https://www.cybercom.net/~copters/pilot/maneuvers.html. Some details are simplified in FlightGear, in particular the engine handling and some overstresses are not simulated or are without any consequence. In FlightGear it is (up to now) not possible to damage a helicopter in flight.

Figure 11.1: In the Bo105 helicopter cockpit

The helicopter flight model of FlightGear is quite realistic. The only exceptions are "vortex ring conditions". These occur if you descend too fast and perpendicularly (without forward speed). The heli can get into its own rotor downwash causing the lift to be substantially reduced. Recovering from this condition is possible only at higher altitudes. On the Internet you can find a video of a Seaking helicopter, which got into this condition during a flight demon-

stration and touched down so hard afterwards that it was completely destroyed.

For all FlightGear helicopters the parameters are not completely optimized and thus the performance data between model and original can deviate slightly. On the hardware side I recommend the use of a "good" joystick. A joystick without springs is recommended because it will not center by itself. You can either remove the spring from a normal joystick, or use a force feedback joystick, with a disconnected voltage supply. Further, the joystick should have a "thrust controller" (throttle). For controlling the tail rotor you should have pedals or at least a twistable joystick – using a keyboard is hard. FlightGear supports multiple joysticks attached at the same time.

11.2 Getting started

The number of available helicopters in FlightGear is limited. In my opinion the Bo105 is the easiest to fly, since it reacts substantially more directly than other helicopters. For flight behavior I can also recommend the S76C. The S76C reacts more retarded than the Bo.

Once you have loaded FlightGear, take a moment to centralize the controls by moving them around. In particular the collective is often at maximum on startup.

Figure 11.2: Sikorsky S76C ready to start

The helicopter is controlled by four functions. The stick (joystick) controls two of them, the inclination of the rotor disc (and thus the inclination of the helicopter) to the right/left and forwards/back. Together these functions are called "cyclic blade control". Next there is the "collective blade control", which is controlled by the thrust controller. This causes a change of the thrust produced by the rotor. Since the powering of the main rotor transfers a torque to the fuselage, this must be compensated by the tail rotor. Since the torque is dependent on the collective and on the flight condition as well as the wind on the fuselage, the tail rotor is also controlled by the pilot using the pedals. If you push the right pedal, the helicopter turns to the

right (!). The pedals are not a steering wheel. Using the pedals you can yaw helicopter around the vertical axis. The number of revolutions of the rotor is kept constant (if possible) by the aircraft.

11.3 Lift-Off

First reduce the collective to minimum. To increase the rotor thrust, you have to "pull" the collective. Therefore for minimum collective you have to push the control down (that is the full acceleration position (!) of the thrust controller). Equally, "full power" has the thrust controller at idle. Start the engine with **}**. After few seconds the rotor will start to turn and accelerate slowly. Keep the stick and the pedals approximately centered. Wait until the rotor has finished accelerating. For the Bo105 there is an instrument for engine and rotor speed on the left of the upper row.

Figure 11.3: Eurocopter EC135 lift-off

Once rotor acceleration is complete, pull the collective very slowly. Keep your eye on the horizon. If the heli tilts or turns even slightly, stop increasing the collective and correct the position/movement with stick and pedals. If you are successful, continue pulling the collective (slowly!).

As the helicopter takes off, increase the collective a little bit more and try to keep the helicopter in a leveled position. The main challenge is reacting to the inadvertent rotating motion of the helicopter with the correct control inputs. Only three things can help you: practice, practice and practice. It is quite common for it to take hours of practice to achieve a halfway good looking hovering flight. Note: The stick position in a stable hover is not the center position of the joystick.

11.4 In the air

To avoid the continual frustration of trying to achieve level flight, you may want to try forward flight. After take off continue pulling the collective a short time and then lower the nose a slightly using the control stick. The helicopter will accelerate forward. With forward speed the tail rotor does not have to be controlled as precisely due to the relative wind coming from directly ahead. Altogether the flight behavior in forward flight is quite similar to that of an badly trimmed airplane. The "neutral" position of the stick will depend upon airspeed and collective.

Transitioning from forward flight to hovering is easiest if you reduce speed slowly by raising the nose of the helicopter. At the same time, reduce the collective to stop the helicopter from climbing. As the helicopter slows, "translation lift" is reduced, and you will have to compensate by pulling the collective. When the speed is nearly zero, lower the nose to the position it was when hovering. Otherwise the helicopter will accelerate backwards!

11.5 Back to Earth I

To land the helicopter transition to a hover as described above while reducing the altitude using the collective. Briefly before hitting the ground reduce the rate of descent slowly. A perfect landing is achieved if you managed to zero the altitude, speed and descent rate at the same time (gently). However, such landing are extremely difficult. Most pilots perform a hover more or less near to the ground and then decent slowly to the ground. Landing with forward velocity is easier, however you must make sure you don't land with any lateral (sideways) component to avoid a rollover.

Figure 11.4: Bo105 landing on the roof of VU Medical Center in Amsterdam

11.6 Back to Earth II

It is worth mentioning autorotation briefly. This is a unpowered flight condition, where the flow of air through the rotors rotates the rotor itelf. At an appropriate altitude select a landing point (at first in the size of a larger airfield) and then switch the engine off by pressing **{**. Reduce collective to minimum, place the tail rotor to approximately 0° incidence (with the Bo push the right pedal about half, with As350 the left). Approach at approximately 80 knots. Don't allow the rotor speed to rise more than a few percent over 100 %, otherwise the rotor will be damaged (though this is not currently simulated). As you reach the ground, reduce the airspeed by lifting the nose. The descent rate will drop at the same time, so you do not need to pull the collective. It may be the case that the rotor speed rises beyond the permitted range. Counteract this by raising the collective if required. Just above the ground, reduce the descent rate by pulling the collective. The goal is it to touch down with a very low descent rate and no forward speed. With forward speed it is easier, but there is a danger of a roll over if the skids are not aligned parallel to the flight direction. During the approach it is not necessary to adjust the tail rotor, since without power there is almost no torque. If you feel (after some practice), that autorotation is too easy, try it with a more realistic payload via the *Equipment* → *Fuel and Payload* menu.

Figure 11.5: Bo105 landing

Part IV

Appendices

Appendix A

Missed approach: If anything refuses to work

In the following section, we tried to sort some problems according to operating system, but if you encounter a problem, it may be a wise idea to look beyond "your" operating system – just in case. If you are experiencing problems, we would strongly advise you to first check the FAQ maintained by Cameron Moore at:

https://wiki.flightgear.org/Frequently_asked_questions

Moreover, the source code contains a directory `docs-mini` containing numerous ideas on and solutions to special problems. This is also a good place to go for further reading.

A.1 FlightGear Problem Reports

The best place to look for help is generally the mailing lists, specifically the **[Flightgear-User]** mailing list. If you happen to be running a Git version of FlightGear, you may want to subscribe to the **[Flightgear-Devel]** list. Instructions for subscription can be found at:

https://wiki.flightgear.org/Mailing_lists

It is often the case that someone has already dealt with the issue you are dealing with, so it may be worth your time to search the mailing list archives at:

https://sourceforge.net/mailarchive/forum.php?forum_name=flightgear-devel
https://sourceforge.net/mailarchive/forum.php?forum_name=flightgear-users

You should also consider searching the FlightGear forums for help, instructions and archives at:

https://forum.flightgear.org

There are numerous developers and users reading those lists and forums, so questions are generally answered. However, messages of the type *FlightGear does not compile on my system.*

What shall I do? are hard to answer without any further detail given, aren't they? Here are some things to consider including in your message when you report a problem:

- **Operating system:** (Linux Fedora 17…/Windows Seven 64 bits…)
- **Computer:** (Pentium Dual Core, 2.3 GHz…)
- **Graphics board/chip:** (ATI Radeon HD 770 XT/NVidia GeForce GTX 590…)
- **Compiler/version:** (GCC version 4.6.3…)
- **Versions of relevant libraries:** (PLIB 1.8.5, OpenSceneGraph 3.0.1…)
- **Type of problem:** (Linker dies with message…)
- **Steps to recreate the problem:** Start at KSFO, turn off brakes…

In order to analyze what happened during the last FlightGear session, the following command can be used (the `~/.fgfs` path corresponds to `$FG_HOME` on Unix-type systems; it needs to be adapted on other systems, or you can use option `--log-dir` to make FlightGear write the log elsewhere):

```
less ~/.fgfs/fgfs.log
```

The default log level is `alert`; it corresponds to the `fgfs` option `--log-level=`*alert*. If you pass option `--log-level=`*debug* to `fgfs`, FlightGear will write a lot more messages to its log file.

Some messages, *a priori* originating from FlightGear's dependencies, aren't written to the log file. One way to capture all messages directly or indirectly coming from FlightGear, is to redirect its standard output and standard error streams:

```
fgfs --log-level=debug >log.txt 2>&1
```

One final remark: please avoid posting binaries to these lists or forums! List subscribers are widely distributed, and some users have low bandwidth and/or metered connections. Large messages may be rejected by the mailing list administrator. Thanks.

A.2 General problems

- FlightGear runs SOOO slow.
  When FlightGear indicates that it is running with something like 1-fps (frame per second) or below you typically don't have working hardware-OpenGL support. There may be several reasons for this. First, there may be no OpenGL hardware drivers available for older cards. In this case it is highly recommended to get a new board.

  Second, check if your drivers are properly installed. Several cards need additional OpenGL support drivers besides the "native" windows ones.

- Either the `configure` or the `make` script display errors due to missing PLIB headers or libraries.
  Ensure you have the latest version of PLIB compiled and installed. The headers, like `puAux.h`, have to be within a `plib` subfolder (e.g. `/usr/include/plib`) and its libraries, like `libplibpu.a`, are typically at `/lib`. Double-check that there is not a second set of PLIB headers and libraries sitting elsewhere! Check carefully the error messages of `configure`. In several cases, it specifies what is missing.

A.3 Potential problems under Linux

Since we don't have access to all possible flavors of Linux distributions, here are some thoughts on possible causes of problems. (This Section includes contributions by Kai Troester.)

- Wrong library versions
  This is a rather common cause of grief especially when you prefer to install the libraries needed by FlightGear by hand. Be sure that especially the Mesa library contains support for the 3DFX board and that GLIDE libraries are installed and can be found. If a `ldd ‘which fgfs‘` complains about missing libraries you are in trouble.

  You should also be sure to *always* keep the *latest* version of PLIB on your system. Lots of people have failed miserably to compile FlightGear just because of an outdated PLIB.

- Non-default install options
  FlightGear will display a lot of diagnostics while starting up. If it complains about bad looking or missing files, check that you installed them in the way they are supposed to be installed (i.ewith the latest version and in the proper location). The canonical location FlightGear wants its data files under `/usr/local/lib`. Be sure to grab the latest versions of everything that might be needed!

- Compile problems in general
  Make sure you have the latest (official) version of gcc. Old versions of gcc are a frequent source of trouble! On the other hand, some versions of the RedHat 7.0 reportedly have certain problems compiling FlightGear as they include a preliminary version of gcc.

A.4 Potential problems under Windows

- The executable refuses to run.

  You may have tried to start the executable directly either by double-clicking `fgfs.exe` in Windows Explorer or by invoking it within a MS-DOS shell. Double-clicking via Explorer never works (unless you set the environment variable `FG_ROOT` in `autoexec.bat` or otherwise). Rather double-click `fgrun`. For more details, check Chapter 4.

  Another cause of grief might be that you did not download the most recent versions of the base package files required by FlightGear, or you did not download any of them at

all. Have a close look at this, as the scenery/texture format is still under development and may change frequently. For more details, check Chapter 3.

Next, if you run into trouble at runtime, do not use Windows utilities for unpacking the `.tar.gz`. If you did, try it in the Cygnus shell with `tar -xvzf` instead.

- FlightGear ignores the command line parameters.

  There can be a problem with passing command line options containing a "=" on the command line. Instead create a batch job to include your options and run that instead.

- I am unable to build FlightGear under MSVC/MS DevStudio.

  By default, FlightGear is build with GNU GCC. The Win32 port of GNU GCC is known as Cygwin. For hints on *Makefiles* required for MSVC or MSC DevStudio have a look into:

  https://wiki.flightgear.org/Building_using_CMake_-_Windows

  In principle, it should be possible to compile FlightGear with the project files provided with the source code.

- Compilation of FlightGear dies.

  There may be several reasons for this, including true bugs. However, before trying to do anything else or report a problem, make sure you have the latest version of the Cygwin compiler. In case of doubt, start `setup.exe` anew and download and install the most recent versions of bundles as they possibly may have changed.

Appendix B

Landing: some further thoughts before leaving the plane

B.1 A Sketch on the History of FlightGear

History may be a boring subject. However, from time to time there are people asking for the history of FlightGear. As a result, we'll give a short outline.

The FlightGear project goes back to a discussion among a group of net citizens in 1996 resulting in a proposal written by David Murr who, unfortunately, dropped out of the project (as well as the net) later. The original proposal is still available and can be found under:

https://groups.google.com/g/rec.aviation.simulators/c/ny8HFBE5_T8/m/OdtIiGNGJc8J

Although the names of the people and several of the details have changed over time, the spirit of that proposal has clearly been retained up to the present time.

Actual coding started in the summer of 1996 and by the end of that year essential graphics routines were completed. At that time, programming was mainly performed and coordinated by Eric Korpela from Berkeley University. Early code ran under Linux as well as under DOS, OS/2, Windows 95/NT, and Sun-OS. This was found to be quite an ambitious project as it involved, among other things, writing all the graphics routines in a system-independent way entirely from scratch.

Development slowed and finally stopped in the beginning of 1997 when Eric was completing his thesis. At this point, the project seemed to be dead and traffic on the mailing list went down to nearly nothing.

It was Curt Olson from the University of Minnesota who re-launched the project in the middle of 1997. His idea was as simple as it was powerful: Why invent the wheel a second time? There have been several free flight simulators available running on workstations under different flavors of UNIX. One of these, LaRCsim (developed by Bruce Jackson from NASA), seemed to be well suited to the approach. Curt took this one apart and re-wrote several of the routines such as to make them build as well as run on the intended target platforms. The key idea in doing so was to exploit a system-independent graphics platform: OpenGL.

In addition, a clever decision on the selection of the basic scenery data was made in the

very first version. FlightGear scenery is created based on satellite data published by the U.S. Geological Survey. These terrain data are available from:

https://www.usgs.gov/core-science-systems/ngp/tnm-delivery/gis-data-download

for the U.S., and

https://www.usgs.gov/centers/eros/science/usgs-eros-archive-digital-elevation-global-30-arc-second-elevation-gtopo30

resp., for other countries. Those freely accessible scenery data, in conjunction with scenery building tools included with FlightGear, are an important feature enabling anyone to create his or her own scenery.

This new FlightGear code – still largely being based on the original LaRCsim code – was released in July 1997. From that moment the project gained momentum again. Here are some milestones in the more recent development history.

B.1.1 Scenery

- Texture support was added by Curt Olson in spring 1998. This marked a significant improvement in terms of reality. Some high-quality textures were submitted by Eric Mitchell for the FlightGear project. Another set of high-quality textures was added by Erik Hofman ever since.

- After improving the scenery and texture support frame rate dropped down to a point where FlightGear became unflyable in spring 1998. This issue was resolved by exploiting hardware OpenGL support, which became available at that time, and implementing view frustum culling (a rendering technique that ignores the part of the scenery not visible in a scene), done by Curt Olson. With respect to frame rate one should keep in mind that the code, at present, is in no way optimized, which leaves room for further improvements.

- In September 1998 Curt Olson succeeded in creating a complete terrain model for the U.S. The scenery is available worldwide now, via a clickable map at:

  https://www.flightgear.org/download/scenery

- Scenery was further improved by adding geographic features including lakes, rivers, and coastlines later. Textured runways were added by Dave Cornish in spring 2001. Light textures add to the visual impression at night. To cope with the constant growth of scenery data, a binary scenery format was introduced in spring 2001. Runway lighting was introduced by Curt Olson in spring 2001. Finally, a completely new set of scenery files for the whole world was created by William Riley based on preparatory documentation by David Megginson in summer 2002. This is based on a data set called VMap0 as an alternative to the GSHHS data used so far. This scenery is a big improvement as it has world wide coverage of main streets, rivers, etc., while its downsides are much less accurate coast lines. FlightGear's base scenery is based on these new scenery files since summer 2002.

- There was support added for static objects to the scenery in 2001, which permits placing buildings, static planes, trees and so on in the scenery.
- The world is populated with random ground objects with appropriate type and density for the local ground cover type since summer 2002. This marks a major improvement of reality and is mainly thanks to work by D. Megginson.
- Today, the effort is still going on, with the use of TerraSync, the tool for on-the-fly scenery download, the powerful mapserver and scenemodels infrastructure laying behind, as well as webforms enabling to quickly add or update objects. Scenery generation tools have been updated in order to use more accurate data such as the 8.50 apt.dat format, as well as external data such as OpenStreetMap, when the license is adequate.

B.1.2 Aircraft

- A HUD (Head-Up Display) was added based on code provided by Michele America and Charlie Hotchkiss in the fall of 1997 and was improved later by Norman Vine. While not generally available for real Cessna 172, the HUD conveniently reports the actual flight performance of the simulation and may be of further use in military jets later.
- A rudimentary autopilot implementing heading hold was contributed by Jeff Goeke-Smith in April 1998. It was improved by the addition of an altitude hold and a terrain following switch in October 1998 and further developed by Norman Vine later.
- Friedemann Reinhard developed early instrument panel code, which was added in June 1998. Unfortunately, development of that panel slowed down later. Finally, David Megginson decided to rebuild the panel code from scratch in January 2000. This led to a rapid addition of new instruments and features to the panel, resulting in nearly all main instruments being included until spring 2001. A handy minipanel was added in summer 2001.
- Finally, LaRCsims Navion was replaced as the default aircraft when the Cessna 172 was stable enough in February 2000 – a move most users will welcome. There are now several flight model and airplane options to choose from at runtime. Jon Berndt has invested a lot of time in a more realistic and versatile flight model with a more powerful aircraft configuration method. JSBSim, as it has come to be called, did replace LaRCsim as the default Flight Dynamics Model (FDM), and it is planned to include such features as fuel slosh effects, turbulence, complete flight control systems, and other features not often found all together in a flight simulator. As an alternative, Andy Ross added another flight dynamics model called YASim (Yet Another Flight Dynamics Simulator) which aims at simplicity of use and is based on fluid dynamics, by the end of 2001. This one bought us flight models for a 747, an A4, and a DC-3. Alternatively, a group around Michael Selig from the UIUC group provided another flight model along with several planes since around 2000.
- A fully operational radio stack and working radios were added to the panel by Curt Olson in spring 2000. A huge database of Navaids contributed by Robin Peel allows IFR

navigation since then. There was basic ATC support added in fall 2001 by David Luff. This is not yet fully implemented, but displaying ATIS messages is already possible. A magneto switch with proper functions was added at the end of 2001 by John Check and David Megginson. Moreover, several panels were continually improved during 2001 and 2002 by John and others. FlightGear now allows flying ILS approaches and features a Bendix transponder.

- In 2002 functional multi-engine support found its way into FlightGear. JSBSim is now the default FDM in FlightGear.

- Support of "true" 3D panels became stable via contributions from John Check and others in spring 2002. In addition, we got movable control surfaces like propellers etc., thanks to David Megginson.

B.1.3 Environment

- The display of sun, moon and stars have been a weak point for PC flight simulators for a long time. It is one of the great achievements of FlightGear to include accurate modeling and display of sun, moon, and planets very early. The corresponding astronomy code was implemented in fall 1997 by Durk Talsma.

- Christian Mayer, together with Durk Talsma, contributed weather code in the winter of 1999. This included clouds, winds, and even thunderstorms.

B.1.4 User Interface

- The foundation for a menu system was laid based on another library, the Portable Library PLIB, in June 1998. After having been idle for a time, the first working menu entries came to life in spring 1999.

  PLIB underwent rapid development later. It has been distributed as a separate package by Steve Baker with a much broader range of applications in mind, since spring 1999. It has provided the basic graphics rendering engine for FlightGear since fall 1999.

- In 1998 there was basic audio support, i. eàn audio library and some basic background engine sound. This was later integrated into the above-mentioned portable library, PLIB. This same library was extended to support joystick/yoke/rudder in October 1999, again marking a huge step in terms of realism. To adapt on different joystick, configuration options were introduced in fall 2000. Joystick support was further improved by adding a self detection feature based on xml joystick files, by David Megginson in summer 2002.

- Networking/multiplayer code has been integrated by Oliver Delise and Curt Olson starting fall 1999. This effort is aimed at enabling FlightGear to run concurrently on several machines over a network, either an Intranet or the Internet, coupling it to a flight planner running on a second machine, and more. There emerged several approaches for remotely controlling FlightGear over a Network during 2001. Notably there was added support for

the "Atlas" moving map program. Besides, an embedded HTTP server developed by Curt Olson late in 2001 can now act a property manager for external programs.

- Manually changing views in a flight simulator is in a sense always "unreal" but nonetheless required in certain situations. A possible solution was supplied by Norman Vine in the winter of 1999 by implementing code for changing views using the mouse. Alternatively, you can use a hat switch for this purpose, today.

- A property manager was implemented by David Megginson in fall 2000. It allows parsing a file called `.fgfsrc` for input options. This plain ASCII file has proven useful in submitting the growing number of input options, and notably the joystick settings. This has shown to be a useful concept, and joystick, keyboard, and panel settings are no longer hard coded but set using *.xml files since spring 2001 thanks to work mainly by David Megginson and John Check.

During development there were several code reorganization efforts. Various code subsystems were moved into packages. As a result, code is organized as follows at present:

The base of the graphics engine is **OpenGL**, a platform independent graphics library. Based on OpenGL, the Portable Library PLIB provides basic rendering, audio, joystick etcroutines. Based on PLIB is SimGear, which includes all of the basic routines required for the flight simulator as well as for building scenery. On top of SimGear there are (i) FlightGear (the simulator itself), and (ii) TerraGear, which comprises the scenery building tools.

This is by no means an exhaustive history and most likely some people who have made important contributions have been left out. Besides the above-named contributions there was a lot of work done concerning the internal structure by: Jon S. Berndt, Oliver Delise, Christian Mayer, Curt Olson, Tony Peden, Gary R. Van Sickle, Norman Vine, and others. A more comprehensive list of contributors can be found in Chapter B as well as in the `Thanks` file provided with the code. Also, the FlightGear Wiki contains a detailed history worth reading of all of the notable development milestones at

https://wiki.flightgear.org/Category:FlightGear_changelogs

B.2 Those, who did the work

Did you enjoy the flight? In case you did, don't forget those who devoted hundreds of hours to that project. All of this work is done on a voluntary basis within spare time, thus please bear with the programmers in case something does not work the way you want it to. Instead, sit down and write them a kind (!) mail proposing what to change. Alternatively, you can subscribe to the FlightGear mailing lists and contribute your thoughts there. Instructions to do so can be found at

https://wiki.flightgear.org/Mailing_lists

Essentially, there are two lists: one of which being mainly for the developers and the other one for end users. Besides, there is a very low-traffic list for announcements.

The following names the people who did the job (this information was essentially taken from the `Thanks` file accompanying the code).

A1 Free Sounds

Granted permission for the FlightGear project to use some of the sound effects from their site.

Syd Adams

Added clipping for 2D instruments, ATC volume control and created a wide variety of aircraft.

Raul Alonzo

Mr. Alonzo is the author of Ssystem and provided his kind permission for using the moon texture. Parts of his code were used as a template when adding the texture. Ssystem Homepage can be found at:

http://openuniverse.sourceforge.net

Michele America

Contributed to the HUD code.

Michael Basler

Author of Installation and Getting Started. Flight Simulation Page at:

https://www.flusi.info

Jon S. Berndt

Working on a complete C++ rewrite/reimplimentation of the core FDM. Initially he is using X15 data to test his code, but once things are all in place we should be able to simulate arbitrary aircraft. Jon maintains a page dealing with Flight Dynamics at:

http://jsbsim.sourceforge.net

Special attention to X15 is paid in separate pages on this site. Besides, Jon contributed via a lot of suggestions/corrections to this Guide.

Paul Bleisch

Redid the debug system so that it would be much more flexible, so it could be easily disabled for production system, and so that messages for certain subsystems could be selectively enabled. Also contributed a first stab at a config file/command line parsing system.

Jim Brennan

Provided a big chunk of online space to store USA scenery for FlightGear!

Bernie Bright

Many C++ style, usage, and implementation improvements, STL portability and much, much more. Added threading support and a threaded tile pager.

Stuart Buchanan

Updated various parts of the manual, wrote the initial tutorial subsystem, developed random vegetation and buildings.

Bernhard H. Buckel

Contributed the README.Linux. Contributed several sections to earlier versions of Installation and Getting Started.

Gene Buckle

A lot of work getting FlightGear to compile with the MSVC++ compiler. Numerous hints on detailed improvements.

Ralph Carmichael

Support of the project. The Public Domain Aeronautical Software web site at

https://www.pdas.com/

has the PDAS CD-ROM for sale containing great programs for astronautical engineers.

Didier Chauveau

Provided some initial code to parse the 30 arcsec DEM files found at:

https://www.usgs.gov/centers/eros/science/usgs-eros-archive-digital-elevation-global-30-arc-second-elevation-gtopo30.

John Check

John maintains the base package CVS repository. He contributed cloud textures, wrote an excellent Joystick Howto as well as a panel Howto. Moreover, he contributed new instrument panel configurations.

Dave Cornish

Dave created new cool runway textures plus some of our cloud textures.

Oliver Delise

Started a FAQ, Documentation, Public relations. Working on adding some networking/multi-user code. Founder of the FlightGear MultiPilot.

Jean-Francois Doue

Vector 2D, 3D, 4D and Matrix 3D and 4D inlined C++ classes. (Based on Graphics Gems IV, Ed. Paul S. Heckbert)

https://www.animats.com/simpleppp/ftp/public_html/topics/developers.html

Dave Eberly

Contributed some sphere interpolation code used by Christian Mayer's weather data base system.

Francine Evans

Wrote the GPL'd tri-striper we use:

https://www3.cs.stonybrook.edu/~stripe/

Oscar Everitt

Created single engine piston engine sounds as part of an F4U package for FS98. They are pretty cool and Oscar was happy to contribute them to our little project.

Bruce Finney

Contributed patches for MSVC5 compatibility.

Olaf Flebbe

Improved the build system for Windows and provided pre-built dependencies.

Melchior Franz

Contributed joystick hat support, a LED font, improvements of the telnet and the HTTP interface. Notable effort in hunting memory leaks in FlightGear, SimGear, and JSBSim.

Jean-loup Gailly and Mark Adler

Authors of the zlib library. Used for on-the-fly compression and decompression routines:

https://zlib.net.

Mohit Garg

Contributed to the manual.

Thomas Gellekum

Changes and updates for compiling on FreeBSD.

Neetha Girish

Contributed the changes for the XML-configurable HUD.

Jeff Goeke-Smith

Contributed our first autopilot (Heading Hold). Better autoconf check for external timezone and daylight variables.

Michael I. Gold

Patiently answered questions on OpenGL.

Habibe

Made RedHat package building changes for SimGear.

Mike Hill

For allowing us to concert and use his wonderful planes for FlightGear.

Erik Hofman

Major overhaul and parameterization of the sound module to allow aircraft-specific sound configuration at runtime. Contributed SGI IRIX support (including binaries) and some really great textures.

Charlie Hotchkiss

Worked on improving and enhancing the HUD code. Lots of code style tips and code tweaks.

Bruce Jackson (NASA)

Developed the LaRCsim code under funding by NASA which we use to provide the flight model. Bruce has patiently answered many, many questions.

Maik Justus

Added helicopter support, gear/ground interaction and aerotow/winch support to the YASim FDM.

Ove Kaaven

Contributed the Debian binary.

Richard Kaszeta

Contributed screen buffer to ppm screen shot routine. Also helped in the early development of the "altitude hold autopilot module" by teaching Curt Olson the basics of Control Theory and helping him code and debug early versions. Curt's "Boss" Bob Hain also contributed to that. Further details available in "Flight Gear Autopilot: Altitude Hold Module" on archive.org.[1]

Rich's Homepage is at

http://www.kaszeta.org/rich/

Tom Knienieder

Ported the audio library first to OpenBSD and IRIX and after that to Win32.

Reto Koradi

Helped with setting up fog effects.

Bob Kuehne

Redid the Makefile system so it is simpler and more robust.

Kyler B Laird

Contributed corrections to the manual.

Roman Ludwicki

Contributed corrections and the Polish translation of this manual, refreshed all screenshots using a current version of FlightGear (in 2021), improved or remade other illustrations (e.g., using vector formats), and helped to get the LaTeX markup of this manual cleaner.

David Luff

Contributed heavily to the IO360 piston engine model.

Sam van der Mac

Contributed to the manual by translating HTML tutorials to LaTeX.

Christian Mayer

Working on multi-lingual conversion tools for fgfs as a demonstration of technology. Contributed code to read Microsoft Flight Simulator scenery textures. Christian is working on a completely new weather subsystem. Donated a hot air balloon to the project.

[1] https://web.archive.org/web/20030803143303/http://www.menet.umn.edu:80/~curt/fgfs/Docs/Autopilot/AltitudeHold/AltitudeHold.html

David Megginson

Contributed patches to allow mouse input to control view direction yoke. Contributed financially towards hard drive space for use by the FlightGear project. Updates to README.running. Working on getting fgfs and ssg to work without textures. Also added the new 2D panel and the save/load support. Further, he developed new panel code, playing better with OpenGL, with new features. Developed the property manager and contributed to joystick support. Random ground cover objects.

Cameron Moore

FAQ maintainer. Reigning list administrator. Provided man pages.

Eric Mitchell

Contributed some topnotch scenery textures being all original creations by him.

Anders Morken

Former maintainer of European web pages.

Alan Murta

Created the Generic Polygon Clipping library.

https://en.wikipedia.org/wiki/General_Polygon_Clipper

Phil Nelson

Author of GNU dbm, a set of database routines that use extendible hashing and work similar to the standard UNIX dbm routines.

Alexei Novikov

Created European Scenery. Contributed a script to turn fgfs scenery into beautifully rendered 2D maps. Wrote a first draft of a Scenery Creation Howto.

Curt Olson

Primary organization of the project.
First implementation and modifications based on LaRCsim.
Besides putting together all the pieces provided by others mainly concentrating on the scenery subsystem as well as the graphics stuff.

Brian Paul

We made use of his TR library and of course of Mesa:

https://www.ssec.wisc.edu/~billh/bp/TR.html,
https://www.mesa3d.org

Tony Peden

Contributions on flight model development, including a LaRCsim based Cessna 172. Contributed to JSBSim the initial conditions code, a more complete standard atmosphere model, and other bugfixes/additions.

Robin Peel

Maintains worldwide airport and runway database for FlightGear as well as X-Plane.

Alex Perry

Contributed code to more accurately model VSI, DG, Altitude. Suggestions for improvements of the layout of the simulator on the mailing list and help on documentation.

Friedemann Reinhard

Development of an early textured instrument panel.

Petter Reinholdtsen

Incorporated the GNU automake/autoconf system (with libtool). This should streamline and standardize the build process for all UNIX-like platforms. It should have little effect on IDE type environments since they don't use the UNIX make system.

William Riley

Contributed code to add "brakes". Also wrote a patch to support a first joystick with more than 2 axes. Did the job to create scenery based on VMap0 data.

Andy Ross

Contributed a new configurable FDM called YASim (Yet Another Flight Dynamics Simulator), based on geometry information rather than aerodynamic coefficients.

Paul Schlyter

Provided Durk Talsma with all the information he needed to write the astro code. Mr. Schlyter is also willing to answer astro-related questions whenever one needs to.

https://stjarnhimlen.se/english.php

Chris Schoeneman

Contributed ideas on audio support.

Phil Schubert

Contributed various textures and engine modeling.

Jonathan R. Shewchuk

Author of the Triangle program. Triangle is used to calculate the Delaunay triangulation of our irregular terrain.

Gordan Sikic

Contributed a Cherokee flight model for LaRCsim. Currently is not working and needs to be debugged. Use `--fdm`=*larcsim* `--aero`=*cherokee* options to run the Cherokee instead of the Cessna.

Michael Smith

Contributed cockpit graphics, 3D models, logos, and other images. Project Bonanza.

Martin Spott

Co-Author of The Manual.

Jon Stockill

Maintains a database of objects and their location to populate the worldwide scenery.

Durk Talsma

Accurate Sun, Moon, and Planets. Sun changes color based on position in sky. Moon has correct phase and blends well into the sky. Planets are correctly positioned and have proper magnitude. Help with time functions, GUI, and other things. Contributed 2D cloud layer. Website at

http://www.durktalsma.nl.

UIUC – Department of Aeronautical and Astronautical Engineering

Contributed modifications to LaRCsim to allow loading of aircraft parameters from a file. These modifications were made as part of an icing research project.

Those did the coding and made it all work:
Jeff Scott
Bipin Sehgal
Michael Selig

Moreover, those helped to support the effort:
Jay Thomas
Eunice Lee
Elizabeth Rendon
Sudhi Uppuluri

U.S. Geological Survey

Provided geographic data used by this project.

https://www.usgs.gov/core-science-systems/ngp/tnm-delivery/gis-data-download

Mark Vallevand

Contributed some METAR parsing code and some Win32 screen printing routines.

Gary R. Van Sickle

Contributed some initial GameGLUT support and other fixes. Has done preliminary work on a binary file format.

Norman Vine

Provided numerous URLs to the FlightGear community. Many performance optimizations throughout the code. Many contributions and much advice for the scenery generation section. Lots of Windows related contributions. Contributed wgs84 distance and course routines. Contributed a great circle route autopilot mode based on wgs84 routines. Many other GUI, HUD and autopilot contributions. Patch to allow mouse input to control view direction. Ultra hires tiled screen dumps. Contributed the initial `goto airport` and `reset` functions and the initial HTTP image server code.

Roland Voegtli

Contributed great photorealistic textures. Founder of European Scenery Project for X-Plane.

Carmelo Volpe

Porting FlightGear to the Metro Works development environment (PC/Mac).

Darrell Walisser

Contributed a large number of changes to porting FlightGear to the Metro Works development environment (PC/Mac). Finally produced the first Macintosh port. Contributed to the Mac part of Getting Started, too.

Ed Williams

Contributed magnetic variation code (implements Nima WMM 2000). We've also borrowed from Ed's wonderful aviation formulary at various times as well. Website at:

https://edwilliams.org

Jim Wilson

Wrote a major overhaul of the viewer code to make it more flexible and modular. Contributed many small fixes and bug reports. Contributed to the PUI property browser and to the autopilot.

Jean-Claude Wippler

Author of MetaKit – a portable, embeddible database with a portable data file format previously used in FlightGear. Please see the following URL for more info:

https://www.equi4.com/metakit/

Woodsoup Project

While FlightGear no longer uses Woodsoup services we appreciate the support provided to our project during the time they hosted us. Once upon a time, they provided computing resources and services so that the FlightGear project could have a real home.

Robert Allan Zeh

Helped tremendously in figuring out the Cygnus Win32 compiler and how to link with .dll's. Without him, the first run-able Win32 version of FlightGear would have been impossible.

Pablo Barrio and Gonzalo Pesquero

Contributed to the Spanish translation of the manual and the FlightGear GUI.

Others

The following individuals have contributed to the scenery object database: Jon Stockill, Martin Spott, Dave Martin, Thomas Foerster, Chris Metzler, Frédéric Bouvier, Melchior Franz, Roberto Inzerillo, Erik Hofman, Mike Round, Innis Cunningham, David Megginson, Stuart Buchanan, Josh Babcock, Esa Hyytia, Mircea Lutic, Jens Thoms Toerring, Mark Akermann, Torsten Dreyer, Martin C. Doege, Alexis Bory, Sebastian Bechtold, Julien Pierru, Bertrand Augras, Gerard Robin, Jakub Skibinski, Morten Oesterlund Joergensen, Carsten Vogel, Dominique Lemesre, Daniel Leygnat, Bertrand Gilot, Morten Skyt Eriksen, Alex Bamesreiter, Oliver Predelli, Georg Vollnhals, and Paul Richter.

B.3 What remains to be done

If you read (and, maybe, followed) this guide up to this point you may probably agree: FlightGear even in its present state, is not at all for the birds. It is already a flight simulator that supports even several flight models, many planes with 3D cockpits, HUD, terrain scenery with realistic buildings, roads, rivers, water awkens, all basic controls with many joysticks support and real weather.

Despite, FlightGear needs – and gets – further development. Except internal tweaks, there are several fields where FlightGear needs basics improvement and development. A first direction is adding airports, buildings, and more of those things bringing scenery to real life and belonging to realistic airports and cities. Another task is further implementation of the menu system, which should not be too hard with the basics being working now. A lot of options at present set via command line or even during compile time should finally make it into menu entries. Finally, FlightGear doesn't have a good ATC so far.

There are already people working in all of these directions. If you're a programmer and think you can contribute, you are invited to do so.

Acknowledgements

Obviously this document could not have been written without all of the contributors mentioned above for making FlightGear a reality.

First, I was very glad to see Martin Spott entering the documentation effort. Martin provided not only several updates and contributions (notably in the OpenGL section) on the Linux side of the project but also several general ideas on the documentation in general.

I would like to express a special thanks to Curt Olson, whose numerous scattered READMEs, Thanks, Webpages, and personal emails were of special help to me and were freely exploited in the making of this booklet.

Next, Bernhard Buckel wrote several sections of early versions of that Guide and contributed at lot of ideas to it.

Jon S. Berndt supported me by critical proofreading of several versions of the document, pointing out inconsistences and suggesting improvements.

Moreover, I gained a lot of help and support from Norman Vine. Maybe, without Norman's answers I would have never been able to tame different versions of the Cygwin – FlightGear couple.

We were glad, our Mac expert Darrell Walisser contributed the section on compiling under macOS. In addition, he submitted several Mac-related hints and fixes.

Further contributions and donations on special points came from John Check, (general layout), Oliver Delise (several suggestions including notes on that chapter), Mohit Garg (OpenGL), Kyler B. Laird (corrections), Alex Perry (OpenGL), Kai Troester (compile problems), Dave Perry (joystick support), and Michael Selig (UIUC models).

Besides those whose names got lost withing the last-minute-trouble we'd like to express our gratitude to the following people for contributing valuable "bug fixes" to this version of The FlightGear Manual (in random order): Cameron Moore, Melchior Franz, David Megginson, Jon Berndt, Alex Perry, Dave Perry, Andy Ross, Erik Hofman, and Julian Foad.

Appendix C

Main Index

Appendix D

Index of Command Line Options

Page numbers in italics indicate the reference page for the option.

Appendix E

List of Acronyms

AAR Air-Air Refueling

ADF Automatic Direction Finder

AGL Above Ground Level

AI Artificial Intelligence

AoA Angle of Attack

AP AutoPilot

ASI Air Speed Indicator

ASL Above Sea Level

ATC Air Traffic Control

ATIS Automatic Terminal Information Service

CDI Course Deviation Indicator

DG Directional Gyro

DH Decision Height

DME Distance Measuring Equipment

DP Departure Procedure

EGT Exhaust Gas Temperature

FAA Federal Aviation Administration

FDM Flight Dynamics Model

FLOLS Fresnel Lens Optical Landing System

GPS Global Positioning System

HUD Head-Up Display

IAF Initial Approach Fix

IAP Instrument Approach Procedure

ICAO International Civil Aviation Organization

IFR Instrument Flight Rules

IGC International Gliding Commission

ILS Instrument Landing System

LDA Localizer-type Directional Aid

LOC Localizer

LOD Level Of Detail

LOM Locator Outer Marker

MCBF Mean Cycles Between Failures

METAR Meteorological Aerodrome Report

MM Middle Marker

MP MultiPlayer

MSL Mean Sea Level

MTBF Mean Time Between Failures

NAV NAVigation

NDB Non-Directional Beacon

OBS Omni Bearing Selector

PTT Push To Talk

RMI Radio Magnetic Indicator

RPM Revolutions Per Minute

SID Standard Instrument Departure

STAR Standard Terminal Arrival Route

TACAN TACtical Air Navigation

TAS True Air Speed

TLA Three-Letter Acronym

TTS Text To Speech

UFO Unidentified Flying Object

VASI Visual Approach Slope Indicator

VFR Visual Flight Rules

VOR VHF Omnidirectional Range

VSI Vertical Speed Indicator

Appendix F

List of Tables

Appendix G

List of Figures

https://www.flightgear.org