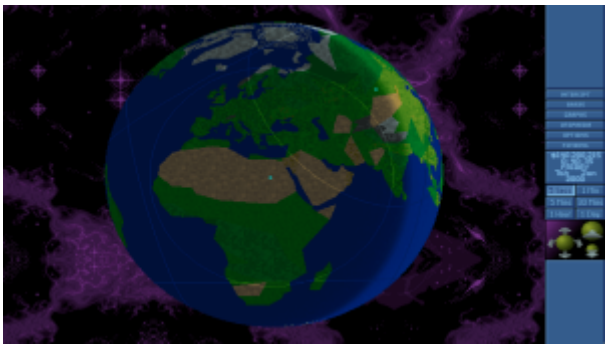
OpenXcom

Infos

OpenXCom is basically nothing else than X-COM UFO defense or Terror from the Deep reprogrammed again. However, the game depends on the original graphics data, so you still need UFO in the original. You can still find it at GOG and Steam. However, since the original practically doesn't run on the current systems anymore, you can easily get over it. But the fact that the game is also available for Linux is really cool.

Also some changes have been made to improve the gameplay. For example, you can make the setting to display the path the soldier will take and the time unit cost before executing a Go command. Doors can now be opened, as in TFTD, without necessarily going through them. You can walk sideways, you can run, you can choose the base layout when starting a new game. The possibility to display the radar range and the course a craft will take is also ingenious.



You can also instruct the soldier to shoot at the target regardless of the missing line of sight. Especially with the auto shot you can shoot away obstacles with the chance that your remaining shots will hit the target. Also at last you can define the order of the soldiers in a spaceship and their weapon layouts will be saved (so you don't have to reorder everything before every battle). You can also choose to have spaceships take off even though their fuel and ammo aren't 100%. The scroll wheel on the mouse now also has a feature, namely switching levels. You can also use the keyboard to scroll and change soldiers. The number of savegames is no longer limited, because the savegames are now by file name. These are all things that I personally find very powerful.

Also some bugs have been fixed. For example, that Avengers never used more than 50% of the fuel is no longer the case. You can finally move the tanks really freely, without any restrictions due to the errors in the mouse control. Also you can now, if you don't feel like starting a (new) campaign, simply generate a battle directly, with selection of which UFO, which race, which level of technology, with what you show up yourself. Good beginner training for everyone who wants to play UFO2000. 😀

Installation

There is a relatively old installation package with OpenXcom 1.0 stable, so not Nightly. Even if „stable“ suggests that this would be the better choice, the nightlies are recommended for various reasons:

- more bugs fixed,
- more functions,
- Mods are now always based on the folder structure of the nightlies, so they don't work with the stable
- there is also TFTD as shipped mod.

The game *requires* the original data from X-COM. If your original data medium is gone by now, you can find the files on the net. OpenXcom needs either version 1.4 of the DOS version or the Collectors Edition. If you are a non-owner and want to proceed correctly, you can buy the game from Steam. There is now also a patch developed by the community that fixes map bugs and the like. It can be found at http://openxcom.org/download/extras/universal-patch.zip.

Windows

Under Windows, the game is simply extracted from the archive into an appropriate folder. The original data has to be copied into the directory UFO or TFTD.

Linux

Via Package Sources (deb)

As already written, it is not recommended to take the stable ones, but rather the nightlies:

```
sudo add-apt-repository ppa:knapsu/openxcom
sudo apt-get update
sudo apt-get install openxcom
```

Then put the original files in the right place, directly into the directory `/usr/local/share/openxcom/UFO/`. For TFTD it is the folder `/usr/local/share/openxcom/TFTD/`. OpenXcom searches there for the original files. It is also possible to place them somewhere else, but it makes sense to stay consistent here to avoid problems.

Mods don't belong in the game directory anymore, but into `$HOME/.local/share/openxcom/mods`, one directory deeper, than where the saved games are. After that you should be able to play.

Compiling

In principle, the game is also available via the package management. However, the packages are not exactly most up-to-date. You can also do this manually:

Prerequisites

Some directories are needed from the original game to make it work at all. The X-COM version is either 1.4 (=DOS), or the Collectors-Edition (=Windows). If somebody doesn't have the game, you can still buy it today via Steam, or you can google it (if you lost your installation disks...) There are at least some abandon ware pages that offer it. So it's advisable to search for the Collectors Edition right away, because you don't always know which version of the DOS versions you really received. For X-COM 1 you need the following directories:

- `GEODATA`,
- `GEOGRAPH`,
- `MAPS`,
- `ROUTES`,
- `SOUND`,
- `TERRAIN`,
- `UFOGRAPH`,
- `UFOINTRO` and
- `UNITS`.

For X-COM 2, the folders are:

- `ANIMS`,
- `FLOP_INT`,
- `GEODATA`,
- `GEOGRAPH`,
- `MAPS`,
- `ROUTES`,
- `SOUND`,
- `TERRAIN`,
- `UFOGRAPH` and
- `UNITS`.

Note: Since Linux is case-sensitive for file names, the folders *must* be written in capital letters. Otherwise the game won't find the files under Linux.

You should install the community patch. If you want to compile yourself more often in the future, you can also simply pack these files together with the original game. First unpack the above mentioned folders into a directory, then unpack the directories from the patch archive there and overwrite them if necessary. The result can be stored in an archive for later use.

You may need to install additional packages to use it, especially for compiling:

- git (only for compiling)
- SDL
- SDL Mixer
- SDL Image
- SDL gfx Version 2.0.22 or higher
- `yaml-cpp` Version 0.5 or higher
- `boost` (dependency for yaml-cpp)
- if necessary `cmake` and `cmake-curses`

Or as a command to install under Ubuntu:

```
sudo apt-get install git libsdl1.2-dev libsdl-mixer1.2-dev libsdl-image1.2-dev libsdl-gfx1.2-dev libyaml-cpp-dev libboost-dev cmake cmake-curses-gui
```

The `-dev` packages are only required for compiling, their compiled versions should actually be installed when installing OpenXcom from a PPA.

Compiler Run

You may have to compile the game yourself, e.g. because newer mods also require newer OpenXcom versions. As a Ubuntu user you have to use the compiler yourself for nightlies. But that's not very difficult either. In principle, you can automate this for the most part.

Download source files. These can always be found at https://github.com/SupSuper/OpenXcom, on the righthand side below the link „Download ZIP“. Unzip it into a directory of your choice (you won't need it afterwards). Or you can do it with git:

```
git clone https://github.com/SupSuper/OpenXcom.git OpenXcom-master
```

With git, the advantage is that the version string also uniquely identifies which build was compiled, e.g. if you want to submit bug reports. This will pretty much narrow down which version the problem occurs with.

1. Copy/unpack the original files mentioned above and the patch files into the folder `bin/UFO/` of the currently unpacked source files. If you still have TFTD around, copy it to `bin/TFTD/`.
2. Adjust the version string if necessary: Normally, the version 1.0 is always specified as the version in these source files, which has relatively little meaning, because this way, for example, it is impossible to determine what version you have installed, one from the PPAs, a nightly, and if the latter, from when. Therefore you can change the string `#define OPENXCOM_VERSION_SHORT "1.0"` in the file `src/version.h`, e.g. to `#define OPENXCOM_VERSION_SHORT "1.0 Nightly 2015-06-05-12-13"`, for the 5.6.2015, 12:13 pm. With Git there is no need for it, but it is also possible if you want more information in the version. For example, Windows precompiled nightlys from the website contain an automatic entry of Git.
3. Create the `build` folder in the source directory and switch to it on the console.
4. `cmake ..` or alternatively `cmake -DCMAKE_BUILD_TYPE=Release ..` in case you want to install a release into the default directories anyway. This saves the next step. But you can also specify the folders, if you like, by passing the paths with `-DVariable=Variablevalue` (see

points under 6).
5. `ccmake …`, this step and the following one are optional if you executed `cmake` with the options passed directly.
6. Make settings:
    1. `CMAKE_BUILD_TYPE` to `Release`, alternatively to `Debug`.
    2. `CMAKE_INSTALL_PREFIX` to the path where the OpenXcom binary is to be installed later. Default is `/usr/local/`, which will put the game in the folder `/usr/local/bin`.
    3. Set `DATADIR` to the path of the target folder for the data files (originals). The files copied to the sources above will then be copied there during the subsequent process. If no path is specified, the data ends up in `/usr/local/share/openxcom/data/`.
    4. press c for configure
    5. e to get out of the configure screen.
    6. Press g to generate and quit `ccmake`.
7. `make -j3`, you can also specify values higher than 3, e.g. `10`, to speed up the compilation process. However, it is not recommended that you do not specify any values in order for it to run at maximum speed. Unforeseen conditions may occur that prevent compilation!
8. `make install`
9. If you have specified a folder in the path `CMAKE_INSTALL_PREFIX` or `DATADIR` for which you do not have write permissions, you must compile under a user who has the permissions. The default path therefore requires root privileges:
10. `sudo make install`

This finishes the compilation (unless any errors occurred…). In this case you should be able to start with standard paths. If you have edited the paths, e.g. to get the game to run as a user somewhere, i.e. without proper installation, you can write a start script:

```
#!/bin/sh
"Pathto/openxcom" -data "Pathto/data"
```

with the respective paths to OpenXcom or the original directory. Then save the script under `$HOME/bin/` and name it `openxcom`, make it executable and you can play OpenXcom without admin rights for the installation in your own home folder by simply entering `openxcom` in the console. Of course, the script can also be located somewhere else, e.g. on the desktop, where you can start it elsewhere.

Desktop file

If you really value optics, you can also create a desktop file for the just created start script, in order to integrate the just compiled OpenXcom e.g. into the starter of Gnome:

```
[Desktop Entry]
Name=OpenXcom
Exec=<pathname/to/OpenXcom-script>
Type=Application
StartupNotify=true
Path=<path/to/OpenXcom-script>
Icon=<path/to/OpenXcom-folder>/common/openxcom.png
```

Save this file e.g. under the name `OpenXcom.desktop` and make it executable. This file can be

located anywhere, but for Ubuntu with Gnome it is recommended to store it under `~/.local/share/applications`, because the starter is searching there too. For all users the folder `/usr/share/applications` is more suitable, because everyone can find the symbol.

Important: The newly installed files in the original folder may not be used if a UFO or TFTD folder is found in another one of the searched paths. Therefore it is recommended to either delete or at least rename all other original folders (e.g. `data old` or something similar). You may be able to find other existing original folders with a simple command:

```
sudo find / -name BACKPALS.DAT
```

`sudo` is necessary, because otherwise you might not be able to find all paths, or error messages may constantly interrupt the output due to missing rights. The file `BACKPALS.DAT` is part of the UFO directory and should theoretically only occur once on the disk, unless you have stashed it somewhere else as well. If you can find several paths with `data/GEODATA/BACKPALS.DAT` at the end, these are exactly such possible problems.

Automated Compilation

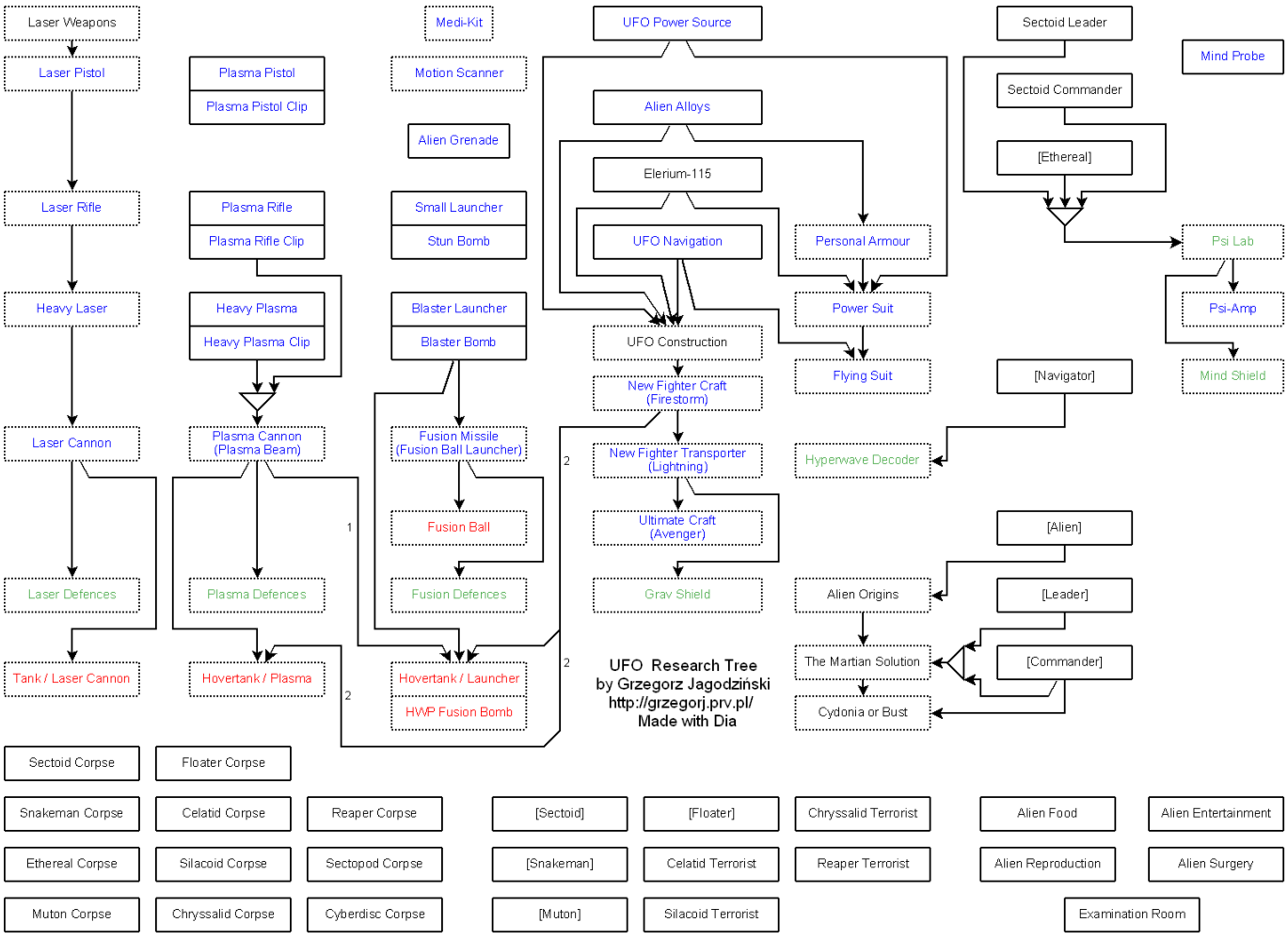
```
echo "Loading source files from github..."
wget https://github.com/SupSuper/OpenXcom/archive/master.zip
echo 'Decompressing "master.zip"...'
unzip master.zip
echo "Changing to newly created sources folder..."
cd OpenXcom-master
echo 'Decompressiong patch and vanilla files into folder "bin/UFO/"...'
<decompression or copy command of vanilla and patch files to ./bin/UFO/>
d=`date +%Y-%m-%d-%H-%M`
echo Changing version string from \"1.0\" to \"1.0 Nightly $d\"
perl -pi -e 's/SHORT "1.0"/SHORT "1.0 Nightly '$d'"/g' src/version.h
echo 'Creating folder "build"...'
mkdir build
echo 'Changing to "build"...'
cd build
echo 'Configuring make as "Release"...'
cmake -DCMAKE_BUILD_TYPE=Release ..
echo 'Compiliing OpenXcom...'
make -j10
echo 'Installiing OpenXcom...'
sudo make install
echo 'Removing not required "master.zip"...'
rm ../../master.zip
```

The sample script downloads the sources, unpacks them, copies/unpacks the original/patch files into the UFO directory, changes the version string and compiles a release build including installation. The no longer needed zip file with the sources will also be removed. **Hint**: It is not recommended to run the script twice, if it is possible that the folder structure has changed. Better delete the old OpenXcom master folders before running! (Can of course be automated as well, like many other things….) Of course you can work yourself to the limit here.

Tech Tree

At this point a hint: For someone who doesn't know X-COM yet, the tech tree should not matter for now. At least once you should have played X-COM 1 or 2 without hints. Otherwise the fun factor might be lost faster than necessary. For the tech tree it is true, that it can show quite considerable changes to the original, up to a conversion to a completely different game. The original tech tree looks like this:

Complete



Savegame Information

- Alien bases in OpenXcom savegames: `alienBases` starts list of alien bases. If one of them says `discovered: true`, it is visible.

Back to the games database