# Contents

Assumptions About the Reader.................................................................................................................... 3
Advanced Features Overview......................................................................................................................... 3
Disclaimer.......................................................................................................................................................... 3
Understanding Memory Slots.......................................................................................................................... 4
Graphics Driver...................................................................................................................................................... 4
  Overview.......................................................................................................................................................... 5
  Concepts.......................................................................................................................................................... 5
    World Coordinate System.......................................................................................................................... 5
    Camera Coordinate System......................................................................................................................... 6
X-Sim extractor Graphics Driver Tutorial...................................................................................................... 7
  Scenario 1........................................................................................................................................................ 8
  Scenario 2........................................................................................................................................................ 8
  Scenario 3........................................................................................................................................................ 8
    Locating a Command Buffer That Contains Values..................................................................................... 9
    Determining if the Values in the Chosen Command Buffer are Accurate.................................................... 9
Creating an On/Off trigger For Auto-Start.................................................................................................... 14
Graphic Driver Game Overlay......................................................................................................................... 15
Saving Your Work.............................................................................................................................................. 16
Sharing a profile................................................................................................................................................ 16
Telemetry (Game) Plugins................................................................................................................................... 17
  Overview......................................................................................................................................................... 17
  Where to get Game Interface Plug-ins........................................................................................................ 17
  Installing a Telemetry (game) Plug-in.......................................................................................................... 17
Direct memory access (formerly force injector)............................................................................................... 18
Force Feedback Driver (Formerly Yoda)......................................................................................................... 19
Getting Help.................................................................................................................................................... 20
Credits and Version Information..................................................................................................................... 20
ASSUMPTIONS ABOUT THE READER

It is assumed that you have a basic understanding of the X-Sim extractor and converter as well as how they relate to each other. You should have a complete mastery of the X-Sim 3 "Generate a game profile" quick guide.

ADVANCED FEATURES OVERVIEW

The new X-Sim 3 extractor contains many powerful features that allow you to extract motion data from nearly any simulation title or PC game. In the past, game data was typically extracted from games via compiled game / telemetry plug-ins. This placed a large burden on the X-Sim developers each time a new game was released. Game / telemetry plug-ins are still supported and recommended, but will not be created for all games. At the time of this writing, most popular games with realistic simulation capabilities already have a game / telemetry plug-in, but many of the arcade style and first-person shooter games do not.

Moving forward, game data can be located and extracted by power users using the new Graphics Driver features. These power users can then share a profile containing this information which can be easily used by newcomers. Please note that the term ‘Graphics Driver’ simply means that motion data is derived from the games graphics engine. X-Sim in no way interferes with or replaces your video cards display driver.

In addition to easing the development burden and enabling power users to locate and share game information, the new Graphics Driver also provides more motion data than some of the existing plug-ins. This gives motion profile creator's increased design capabilities resulting in an even more realistic result.

The X-Sim extractor offers four game data extraction methods: Telemetry Plug-ins, Graphics Driver, Direct Memory Access and Force Feedback Driver. This manual will discuss Telemetry Plug-Ins and the Graphics Driver.

⚠️ It is important to understand that game data can be derived from more than one of these sources and sent to the X-Sim converter!

By the end of this manual, you should be able to extract data from a game that is not currently supported, save the configuration and share it online with others.

DISCLAIMER

This manual was written with an Alpha version of X-Sim. Many algorithms were not complete or not functioning at the time of this writing. Actual logic, screens and steps may vary from those in this manual. It is assumed that this manual will be fine tuned in the future.

You should use extreme caution when creating profiles and testing them. Unexpected simulator movements may occur during this process. It is highly recommended that you are not seated in your simulator when performing the steps in this manual!
UNDERSTANDING MEMORY SLOTS

A memory slot is simply a numbered location that game or gauge data can be assigned to when being passed from the X-Sim extractor to the X-Sim converter. X-Sims’ memory slots are organized into four groupings.

**Slots 1-20** – Data from Graphic Driver, Force Feedback Driver or Direct Memory Access
**Slots 21-40** – Game Telemetry Plug-in Motion Data Outputs
**Slots 41-55** – Joystick Axis and Button Status
**Slots 56-92** – Game Telemetry Plug-in Gauge Data Outputs

You can drag and drop available effects from the top list to the bottom list.

❗ It is recommended that you always export all effects in the order that they are shown in the top list! This will maximize compatibility with all motion profiles. This will ensure that the maximum number of motion profiles is compatible with your profile and that a consistent auto-start effect number is used by all.

GRAPHICS DRIVER
OVERVIEW

The Graphics Driver is a tool for power users that can be used to easily obtain motion data from a currently unsupported game. With this tool, you can obtain motion data either from the DirectX or Open GL based PC games. Most importantly, the X-Sim extractor will automatically save your settings and share them online in a big cloud so that basic users will be able to obtain the same motion data without having knowledge of this tool.

For those who are familiar with previous versions of X-Sim, you can think of this new tool as a method of creating a plug-in because it provides a simplified user interface to assist you with extracting data from a game and you then share the result with others much like plug-ins are shared.

NOTE: If a both a telemetry plug-in (that contains motion data) and a graphic driver solution exists for the same game, you should use the telemetry (game) plug-in because it will provide the best quality of motion in most cases and it is supported by the game developer in an online usage.

CONCEPTS

While it is not necessary to completely understand how the graphics driver or game graphics engine works in order to use it, it is important to understand some basic concepts about coordinate systems so that you can more effectively complete the steps in the following sections.

World Coordinate System

In a game, your player or vehicle is positioned on a game map. The game map is typically a track or level. The game keeps track of your player's position on the map in 3D space. This is not as complicated as it sounds. We live in 3D space. Points and objects are located in 3D space by coordinates. The location of a real world object can also be defined by coordinates.

Unlike the real universe, the "world" of a game has defined maximum points of each direction. This allows us to define an objects' location within the bounds of the "world".

The illustration below depicts the location of a fighter plane relative to its 3D world.

It is important to understand that game manufacturers may define X, Y & Z differently, but that they are always used to define coordinates in 3D space.
**Camera Coordinate System**

It is also important to understand that a game has a camera coordinate system. The camera location is really the view that you see on the screen. This camera view is relative to the X,Y & Z coordinates of the world view as shown below.

For the purposes of using the X-Sim graphics driver, the only thing you need to know is that a user's view consists of angles that are relative to the direction that he is pointing in the world view and that this view angle is described via X,Y & Z coordinates. The image below depicts an example camera view angles.

In the topics to follow, we will be using our knowledge of the coordinate systems above to orient the X-Sim graphics driver. This orientation is necessary because each game manufacturer utilizes the X, Y and Z axis differently. The entire point of this section is to provide you with enough knowledge of coordinate systems to be able to tell the X-Sim graphics driver which direction is up and to determine if the output from the Graphics Driver is correct.

To learn more about Polar Coordinate Systems, Euler angles and Cartesian Coordinate Systems please see the following Wikipedia articles:

http://en.wikipedia.org/wiki/Euler_angles
http://en.wikipedia.org/wiki/Cartesian_coordinate_system
http://en.wikipedia.org/wiki/Polar_coordinate_system
The graphics driver user interface can be found by clicking Settings Open Memory Driver and then clicking Graphic Driver in the X-Sim extractor application.

The Graphics Driver User Interface is broken into six steps. You will need to complete each of these six steps with a game loaded in order to accurately locate and send motion data from the game.

When using this user interface, you will either need to run the game in a window, or on a separate monitor. You will need to be able to see both the game and this window at the same time.

You can use the above dialog or you can read the more below chapter about the ingame game overlay which will lead you through the same steps as displayed in the graphic driver dialog.

**Important game setup before you start**

The graphic driver will reverse calculate the speed and accelerations out of your ingame position. Too many calculations per second will decrease your precision of the values because the real-time time measurement will get too low values for a computer value like a double float. To avoid this and to stabilize the result values you must use the game options that are called “VSYNC enabled”, in the game itself or as a fixed value in your graphic card options, to reduce the game updates to the update frequency (fps) of your monitor (LCD normally 60 times per second). Only very high end computers can ignore this.

You need always the incar or first person view. You cannot use the graphic driver with a look from behind. You have to insure a view out of the cockpit of your car or plane or you have a view which is anchored with your vehicle.

You should start a game after opening the motion driver dialog. The game MUST be started by pressing the ‘Play Game’ button if you like to additionally activate the graphic driver.
STEP 1 – CHOOSING A GRAPHICS ALGORITHM

Start now your game with the play button and go to the part of the game which is 3D, i.e. the racing track. You will need a map with a straight road where you can accelerate and brake without reducing your speed in curves. The following steps need your continuous support and decisions.

The graphics algorithm is a method of extracting motion data from a game via its graphics engine. The two most common graphics engines are DirectX and OpenGL. It is possible that more than one algorithm is available to extract data from a game.

In this case, a graphics algorithm is chosen via process of elimination. The simplest and first approach at narrowing our options is by comparing which algorithms have value that move in the menu vs. in the game itself.

If you find only one option that does not move in the menu, but does move in the game, then you are done and should choose this algorithm.

Step1: Choose a algorithm (algorithm usage displayed behind the type)

- D3D8 Transform
- D3D9 Transform
- D3D9Ex Transform
- D3 LookAt
- OpenGL LookAt
- OpenGL Matrix
- OpenGL Translate
- OpenGL Identity
- OpenGL Rotate
- OpenGL MulMatrix
- Vertex Float
- Pixel Float

Below are several sample scenarios and the logic for determining which algorithm to use.

Scenario 1

Only one algorithm shows changing values during game play.

You should choose this option because it is the only algorithm that could possibly provide motion data.

Scenario 2

One algorithm shows changing values in the menu, but not during game play. Another algorithm shows changing values during game play, but not in the menu.

You should choose the option that shows changing values during game play, but not in the menu because this option will be more stable and less likely to result in an unexpected simulator movement while in the game menu.

Scenario 3

Two or more algorithms show changing values both during game play and in the game menu.

A determination cannot be made at this step. Please select the first algorithm in the list and continue to Steps 2 and 3 to determine the quality of the values resulting from the chosen algorithm.
STEP 2 – DETERMINING WHICH COMMAND BUFFER CONTAINS THE CORRECT POSITIONING VALUES

In this step, we will step through the command buffers of the chosen algorithm in an attempt to locate the coordinates of our player or vehicle within the game map.

This step contains two tasks:

1. Locate a command buffer that contains changing values.
2. Determine if the values in this Command buffer are accurate.

Locating a Command Buffer That Contains Values

Find the first command buffer that contains changing X, Y & Z position values by clicking the up arrow on the numeric selector in step 2 of the dialog. Be sure to use the top numeric selector. (At the time of this writing, the remaining selectors are not functional)

In the example image below, we have selected command buffer 1

Look for values in steps three and four. If you see values here, then we can move along to determining if these values are accurate.

Determining if the Values in the Chosen Command Buffer are Accurate

The remaining steps all revolve around orienting the ‘Graphics Driver’ (telling it which direction is which) and determining the accuracy of the values that are in this command buffer.

At this point, we are only concerned with determining that the values change as we move around in the game and that they change in a logical manner. So for example, if you walk or drive forward and the value X moves, the value X should also move when you walk or drive in reverse.

Game developers often combine effects. You should expect this when evaluating the accuracy of values in steps 2-4. For example, when walking in Quake, the developer mixes in vertical walking effects in addition to the forward movement. Similar mixing is often done in vehicles. In short, it is not reasonable to expect that only one of the displayed values will move when you walk or drive forward, but it is reasonable to expect that one value will move more than the others.

If the values do not follow this logic, then you may have selected a command buffer that contains data for another player, replay system, etc.. You will need to return to Step 1 and try a different Command Buffer.

Once you have verified that the values follow the pattern of logic described above, you are ready to move to the next step.
STEP 3 – DETERMINING PLAYER OR VEHICLE POSITION ORIENTATION ON THE MAP

The goal of this step is to determine which direction is up for the player or the vehicle. Please do not confuse this with Step 4, which is determining the camera look direction. This step is necessary because most game manufacturers do not follow a standard for mapping the X, Y and Z Axis. The X-Sim graphics drivers needs to know which direction is which so that it can properly extract motion data. We accomplish this by determining which direction is up and selecting the correct radio button as shown below.

In each game you will need to do something slightly different to determine which direction is up. In a first person shooter such as Quake, this is easy because you can simply jump up and see which values change.

In a racing game, this isn't quite as simple because you can't easily make your car jump up and down. There are two different ways to determine which value is up. You can either drive up onto a tall hill or you can use process of elimination by driving straight, checking which values move, and then drive in another straight line, but 45 degrees to the right or left.

You can always look at the result table and look if the yaw value make sense.

Once you have selected the appropriate value for the up direction, you can move on to the next step.

The values shown in this section are map based. Since we cannot be sure that exactly where the X, Y & Z axis are, we cannot be sure that we are driving in a straight line along a given axis. Therefore, assuming that Y is up, it is reasonable to expect that both X and Z will move when we attempt to drive in a straight line or when we drive in a circle. Also, because of road surface, it is reasonable to expect that the vertical axis (Y in this example) may also move slightly, but not as much as X and Z.

If you see zeros here, then you have not selected an appropriate command buffer. Please select a different command buffer and start again.

STEP 4 – DETERMINING PLAYER OR VEHICLE CAMERA ORIENTATION

The goal of this step is to determine the orientation of the games camera system. If you haven't been following along, you should read the ‘Camera Coordinate System’ section above.

Again, in each game you will need to do something slightly different to determine which direction is up. In a first person shooter such as Quake, this is easy because you can simply look upward and see which value changes.

In most racing games, you will need to accomplish this via process of elimination by mapping the look left, look right and look back commands to keys on your keyboard and then using them to look around. After looking left, right and behind, you will know which value is up because it will be the only value that did not change.

If you see zeros here, then you have not selected an appropriate command buffer. Please select a different command buffer and start again.
buffer and start again.

**STEP 5 – EXAMINING THE RESULTS OF YOUR CHOICES IN STEPS 1-5**

Step 5: compare result values and fine tuning

| Speed X: 0.000000 | Acceleration X: 0.000000 | Roll: -0.0 |
| Speed Y: 0.000000 | Acceleration Y: 0.000000 | Pitch: -0.6 |
| Speed Z: 0.000000 | Acceleration Z: 0.000000 | Yaw: 84.4 |

If you are not moving, then your speed and acceleration values should be zero.

You should be able to move around the game and see proper Yaw, Pitch, Roll and Acceleration (G-Force) values.

You can test these values by performing normal driving operations such as accelerating, braking, driving up and down a hill, placing two wheels on a curb, etc., and watching to see what values change. Your values should be stable and only have small or no changes when your vehicle or player is stopped. If your values are not stable, you should start over at step 2 and try a different command buffer.

**Fine tune the speed until it has no jitter**

Setup the filter of the graphic driver to the common best startup values:

- Filter strength: 10
- Value factor: 1
- Speed filter type: delayed median
- Acceleration filter type: delayed square

You have to start the X-Sim converter application and add the longitudinal speed value with a G-Force plugin to a new generated axis. Use the auto detection feature of the G-Force plugin. You may delete this if the values are fidgety and the curve is less high.

You will see in real time the results of your graphic driver. If you accelerate the blue line of the math graph will show you a increasing speed. It is absolutely recommended that this line has no jitter if you want a good acceleration value in the next step. To get a perfect speed math graph you must increase the filter strength to about 5 to 10 and tryout the math graph again while you accelerate your vehicle. Change the “speed filter type” in your first steps to a “delayed median”.

If your speed is reacting in a too low speed to a brake event you must decrease the filter strength but not below 3. You can also try the other filter types of the speed value until you get a perfect behavior. If the math line is showing spikes or sudden throttle loss you will have big problems with the accelerations which will be calculated out of the changing speed value. You will detect a wrong setting in the increasing speed graph side (left) and with spikes if you brake. The speed graph
must be a straight increasing value without any jitter.

**Fine tune the acceleration**

Add another axis to the converter math dialog and insert a G-Force plugin with the longitudinal acceleration. Look again to the math graph and the resulting acceleration values. Change the Acceleration filter type to the "delayed square". Like to the speed value you must play around with the filter type until you get a acceleration that fits to your driving actions.

This graph shows the result of the speed graph if you increase the speed and brake.

![Graph showing constant acceleration, gear switch, and brake](image)

**Important hint: enable V-Sync**

The graphic driver will reverse calculate the speed and accelerations out of your ingame position. Too many calculations per second will decrease your precision of the values because the real-time time measurement will get too low values for a computer value like a double float. To avoid this and to stabilize the result values you must use the game options that are called "VSYNC enabled", in the game itself or as a fixed value in your graphic card options, to reduce the game updates to the update frequency (fps) of your monitor (LCD normally 60 times per second). Only very high end computers can ignore this.

⚠️ You should go back to the game menu and ensure that none of your values in Step 5 move while in the menu!! If your values change while you are in the menu then your simulator may unexpectedly auto-start while in a menu and cause injury!!!!!
The final step before saving your profile is to tell the extractor which values it should export in which memory slots. If you are unfamiliar with the concept of a memory slot, please refer to the ‘Understanding Memory Slots’ section above.

It is recommended that you export all effects in the order that they are shown in the top list! The map position effects are not useful for motion profile creation, but all other effects provide valuable information.

Beginners simply use the standard button. It is recommended to use this if you share a profile to get a high compatibility to other profile writers.

Exporting all possible effects also ensures that motion profile developers will have all possible data for their creative use.

Simply drag and drop motion data lines from the top table to the bottom table as shown in the example image.
CREATING AN ON/OFF TRIGGER FOR AUTO-START

Much like a game plug-in, the game profile that we are building has the ability to extract motion data from a game. However, it does not yet have the ability to tell the simulator when to auto-start.

Several options are available for auto-starting and more will be added in the future. At the time of this writing, the triggers for auto-starting can be a joystick button on a force feedback device, a trigger from a game plug-in, a force feedback effect or a changing value detection by the graphics driver.

If a developer plug-in is available for the game that you are building an profile for, then you should use this because it is the safest and most reliable method of auto-starting the simulator. Unfortunately, in most cases, we won’t be building profiles for games that already have a game plug-in, so we will need to consider the alternatives.

If you use the graphic driver only choose the graphic driver detection like the example below. If it does not fulfill your needs you may read the force feedback driver guide or use a joystick button as a switch.

![Motion driver setup diagram](image-url)
GRAPHIC DRIVER GAME OVERLAY

The game overlay menu (On Screen Display) contains the same Steps 1-5 as the graphics driver user interface shown above in the section ‘X-Sim extractor Graphics Driver User Interface’, but it does not include Step 6. After completing the last step in this overlay (Step 5), you will need to exit the game and go to the extractor Graphics Driver User Interface to complete Step 6 and then save your profile.

KEYBOARD

F1 + F2  Change OSD type, available in every menu

F3 - F10  OSD menu keys
          Scroll Lock  Enable/Disable OSD menu

LED shows OSD menu state

If you launched the game via the X-Sim extractor, then you can press the SCROLL-LOCK key to see the game overlay menu.

Pressing F4 will decrease the transparency of the overlay so that it is easier to read. Pressing F11 and F12 will move you forward and back in the overlay menus.

You can follow the guide to Steps 1-5 above when using the overlay menu.
SAVING YOUR WORK

To save your work simple create a new profile with the above button. It will add your work to your local database and you can select out of the left game listbox your game and load it again whenever you want.

Only members with the supporter edition can save and load their profile on the local harddisk in a file

The entire purpose of the X-Sim community is to create a worldwide network of individuals who create and share their profiles. If you have followed this tutorial and have created an profile, then PLEASE, continue to the next step and share it!

SHARING A PROFILE

All the profiles that are stored in the local database will be synchronized on the next extractor startup. You can recall your profiles with the “Show/edit your online clouded profiles” button or search for the game executable with the button below. If you like to delete a profile you must delete it out of the local database and also delete it out of the online cloud. If you

If you think your profile is worth to be fixed for the other members you can show your profiles, select your profile and set it as verified. This shows other users that this profile is a finished profile and it should work perfectly.
TELEMETRY (GAME) PLUGINS

OVERVIEW

A Telemetry plug-ins gathers motion data and gauge data from a game and provides it to the X-Sim extractor for submission to the converter. They are game specific.

Please note that telemetry plug-ins are also often referred to as Game Plugins because they are typically game specific.

If a both a telemetry plug-in (that contains motion data) and a graphic driver profile exists for the same game, you should use the telemetry (game) plug-in because it will provide the best quality of motion.

To learn more about telemetry (game) plug-ins such as what games are supported and what data is gathered by the plug-in, please read the x-sim introduction papers.

The process of developing a new telemetry plug-in C++ is documented here in the x-sim manual.

WHERE TO GET GAME INTERFACE PLUG-INS

In most cases, you will not need to download new game interface plug-ins because they are distributed with each release of the X-Sim software.

Pre-release game interface plug-ins can be sometimes be downloaded from http://www. x-sim.de/

INSTALLING A TELEMETRY (GAME) PLUG-IN

Telemetry plug-ins are installed in “C:\Program Files\X-Sim\Plugin” on 32-bit Operating Systems and “C:\Program Files (x86)\X-Sim\Plugin” on 64-bit Operating Systems.

To install a plug-in, simply copy it into the X-Sim\Plugin folder on your PC and restart the extractor.
DIRECT MEMORY ACCESS (FORMERLY FORCE INJECTOR)

Direct Memory Access is useful for determining an auto-start value or finding in game gauge values.

This manual does not currently cover the concept of Direct Memory Access. For more information on how to discover and utilize direct memory values, please have a look to the tutorials in the X-Sim community and on the X-Sim.de homepage.

DMA ADDRESS SCREEN
The force feedback driver will only work with a connected and working force feedback joystick or wheel. This driver will be between the wheel and the game and will present you every force feedback command that was sent by the game developer to the device. The game developer will not send a combined force command to the device. Instead Microsoft will send each game event as a short message to the device and the device combine this to a sine force, a periodic force or a constant force. Not all force feedback devices can handle all the command combinations which results in a bad feeling because most of the effects are simple ignored. You will see in the force feedback driver all the commands and you can use them. As an example you can use the constant left and right force of a wheel as a movement for your simulator. Like you work against the wheel you will then work against the moving chair. Another good feature is the curbs detection. Most of the game developer send a shake effect of you drive over a curbs and you can detect them with this dialog. These are always effects that comes directly from the game developer.

For this dialog you will find a special setup guide on the X-Sim.de homepage. It is a very complex search but if you have once done it you can share the result to all other members which are then able to use it.
GETTING HELP

You can get further assistance at http://www.x-sim.de, or from your simulator manufacturer.

CREDITS AND VERSION INFORMATION

<table>
<thead>
<tr>
<th>Version</th>
<th>Editor</th>
<th>Change</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>Berney Villers Jr</td>
<td>Original Draft</td>
<td>8/20/2010</td>
</tr>
<tr>
<td>1.1</td>
<td>Berney Villers Jr</td>
<td>Added Images, Disclaimers &amp; Table of Contents</td>
<td>9/02/2010</td>
</tr>
<tr>
<td>1.2</td>
<td>Berney Villers Jr</td>
<td>Updated Images &amp; Added Key Shortcut Descriptions</td>
<td>10/03/2010</td>
</tr>
<tr>
<td>1.3</td>
<td>Martin W.</td>
<td>Rewritten for X-Sim 3</td>
<td>04/26/2012</td>
</tr>
</tbody>
</table>

*Thanks to Martin Wiedenbauer for creating and sharing the X-Sim software with the world!*

**Thanks to Wikipedia for the images used to describe coordinate systems.**