

# 13 Parallel Rendering and Virtual Reality

EnSight Gold 7.6 supports general parallel rendering for increased performance, increased display resolution, and arbitrary screen orientations. Combined with support for 6 DOF (degree-of-freedom) input devices, EnSight Gold provides an immersive virtual reality interface. This chapter describes the configuration file format and command-line parameters required for parallel rendering and 6D input. Using hardware accelerated rendering, the features described are supported on SGI Irix 6.5, Sun Solaris 8, and HP-UX 11.0. Using software rendering (`ensight7 -x`) these features are supported on all Unix/Linux platforms. Six DOF input is supported on all Unix platforms, with pre-compiled trackd support for Irix, Solaris, and Linux (x86).

In order to make use of parallel rendering with EnSight, the user must create a configuration file. This file is specified on the command line using the argument `-dconfig <file>`. If `<file>` is not a fully-qualified path EnSight will search for the file in the following directories:

1. `~/.ensight7/dconfig/`
2. `$CEI_HOME/ensight76/site_preferences/dconfig/`

These options allow for user-level and site-level configurations, respectively.

There are two logical displays which can be configured in EnSight. The *GUI display* is always active, and consists of the main rendering window embedded in the user-interface. The *detached display* is external to the user-interface, and may consist of 1-16 regions configured to form a large continuous display. The configuration file contains information about both the GUI display and the detached display, as well as tracking calibration information and options for using 6D input devices. The remaining sections will address each of the capabilities related to parallel rendering and VR. The sample configurations described in this chapter can be found in the directory `$CEI_HOME/ensight76/src/input/dconfig`. There are also examples of ‘simulated’ configuration files, which allow you to simulate display to multiple graphics pipes on a single display.

## CONFIGURATION FILE FORMAT

### MULTI-PIPE PARALLEL RENDERING

### DISPLAY WALLS

### IMMERSIVE DISPLAYS

### TRACKING

### ANNOTATIONS

### STEREO DISPLAY

### TIPS

## CONFIGURATION FILE FORMAT

Configuration files are entirely text-based beginning with the line:

```
PRsD 2.0
# after the first line, anything following a '#' is a comment
```

The remainder of the file consists of one or more sections describing the displays and options. In describing the format of the file, portions which are optional will be surrounded by [].

## MULTI-PIPE PARALLEL RENDERING

The default mode of EnSight uses only a single graphics pipe for rendering to the GUI display. When run on a multi-pipe machine, EnSight can be configured to use the additional pipes to accelerate the display through parallel rendering. The format of the configuration for the GUI display is as follows:

```
guidisplay
  worker <p1>
  worker <p2>
  ...
  worker <pn>
```

where:

<p<sub>i</sub>> = an X display (i.e. localhost:0.2)

### Example 1:

```
PRsD 2.0
# This configuration uses 4 graphics pipes to
# accelerate rendering
guidisplay
  worker :0.1
  worker :0.2
  worker :0.3
```

In the example above, there is one X server (:0) which manages four graphics pipes. Note that the configuration file does not include the pipe to which the EnSight GUI is displayed. The GUI is always displayed on the pipe indicated by the `$DISPLAY` environment variable, and it is not necessary to specify this screen in the configuration file. Parallel software rendering is available on all Unix/Linux platforms with the `-X` option. The same configuration file format is used in this case, although the displays themselves are not actually opened.

There are two convenience mechanisms for common configurations. When running EnSight on an X server with multiple screens, it is possible to configure EnSight to use all of the pipes for accelerating the GUI display using the command-line option: `-dconfig mpipe`. This will detect how many pipes are available and configure them appropriately. This auto-configuration option will not be able to configure multiple screens which belong to multiple X servers (i.e. :0.0, :1.0, :2.0).

When using software rendering, if the named file is not found and “<file>” is a number, this number will be interpreted as the number of parallel rendering workers. For example:

```
ensight7 -X -batch -dconfig 3 -p <cmdfile>
```

will run a batch session with a total of 4 threads performing parallel rendering.

## DISPLAY WALLS

Another type of parallel rendering available in EnSight allows for the use of multiple graphics pipes to create large, flat tiled displays. Commonly referred to as display walls, this is the first example of a “*detached display*” supported by EnSight. The advantage of a display wall configuration is that the file specification is easy to create. The disadvantage is that display walls cannot be used for tracking and 6d input. In order to use tracking, it will be necessary to use the more general immersive configuration format described later.

The specification for a display wall consists of:

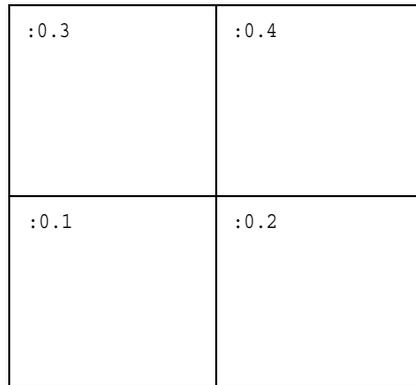
```
wallresolution
  <x-res> <y-res>
numpipes
  <num>
pipe
  xserver <p1>
  resolution <x-res> <y-res>
  wallorigin <wall-x> <wall-y>
  [ xorigin <x0> <y0> ]
  [ lefteye
    or
    righteye
  ]
  [ worker <p2>
    ...
    worker <pn> ]
[ repeat 'pipe' section (num-1) more times ]
```

The `wallresolution` section gives the total pixel resolution of the display wall. The `numpipes` parameter specifies how many separate regions will be configured. For each region, there will be a `pipe` section that describes the size (resolution) and position (`wallorigin`) of the region within the global display. The `xserver` parameter specifies the X display (i.e. :0.1). The `xorigin` is an optional parameter to specify the origin of the window on the given pipe (default (0,0)). Note that `xorigin` is a position relative to the origin of a given xserver, while `wallorigin` is a position relative to the origin of the global display. Changing `wallorigin` will change the region of the wall that is visible in a given window, while changes to `xorigin` simply move the window on the screen without changing the contents. Example 5a will demonstrate a situation when the use of `xorigin` is useful. The `lefteye/righteye` optional designation can be used for passive stereo displays, in which separate graphics pipes render the left and right images. Note that each pipe in a detached

display can have one or more `worker` pipes configured to accelerate the rendering, just as described in the previous section.

### Example 2

In this example there is one X server with five graphics pipes. The GUI is displayed on pipe `:0.0`, with the other four pipes used for the detached display. Four projectors are configured in a 2x2 array to form a large continuous wall as illustrated::



```
PRSD 2.0
#
# conference room display wall
#
wallresolution
    2560 2048
numpipes
    4
pipe # lower-left
    xserver :0.1
    resolution 1280 1024
    wallorigin 0 0
pipe # lower-right
    xserver :0.2
    resolution 1280 1024
    wallorigin 1280 0
pipe # upper-left
    xserver :0.3
    resolution 1280 1024
    wallorigin 0 1024
pipe # upper-right
    xserver :0.4
    resolution 1280 1024
    wallorigin 1280 1024
```

### Example 3

It is not uncommon for displays walls to use overlapping images with *edge-blending* to smooth the otherwise sharp transition between projector images. The edge-blending is performed by the projectors directly. This is easily configured as

a detached display by specifying pipes with overlapping pixel regions. Consider an example of two pipes at 1280x1024 resolution each, with an overlap of 128 pixels.

```
PRsD 2.0
#
# edge-blending example
#
wallresolution
    2432 1024
numpipes
    2
pipe # left
    xserver :0.1
    resolution 1280 1024
    wallorigin    0    0
pipe # right
    xserver :0.2
    resolution 1280 1024
    wallorigin 1024    0
```

Note that in this case the total resolution of the wall in the x direction is decreased by the amount of overlap.

#### **Example 4**

Passive stereo displays achieve stereo by projecting overlapping polarized images from multiple projectors. This can be achieved using detached displays with a distinct rendering region for each screen and eye. Consider for this example a single screen with two projectors. For illustration purposes we will assume that we have five graphics pipes. One pipe (:0.0) renders the GUI and is not listed. Two pipes use parallel rendering to render the left eye image, and two pipes render the right eye image.

```
PRsD 2.0
#
# passive stereo display
#
wallresolution
    1280 1024
numpipes
    2
pipe # left-eye
    xserver :0.1
    resolution 1280 1024
    wallorigin    0    0
    lefteye
    worker :0.2
pipe # right-eye
    xserver :0.3
    resolution 1280 1024
    wallorigin    0    0
```

```
righteye
worker :0.4
```

Note that the `lefteye/righteye` parameters are NOT necessary when using traditional quad-buffered stereo to drive the projectors. Some systems have a signal splitter which takes the frame-sequential stereo signal and generates separate signals for left and right eye. In this case a conventional configuration file without the “eye” designations will work fine. Passive stereo displays are always in stereo mode.

## IMMERSIVE DISPLAYS

True immersive display requires more information than is present in the display wall configuration files previously described. The key factors are that (1) immersive displays are often not flat and (2) the rendered images must be co-registered with the coordinates of a 6d input tracking system.

The basic syntax of the immersive display configuration is going to look very similar to the display wall format:

```
numpipes
  <num>
pipe
  xserver <p1>
  resolution <x-res> <y-res>
  [ xorigin <x0> <y0> ]
  [ bottomleft <x> <y> <z>
    bottomright <x> <y> <z>
    topleft <x> <y> <z>
  ]
  [ lefteye
    or
    righteye
  ]
  [ worker <p2>
    ...
    worker <pn> ]
  [ repeat 'pipe' section (num-1) more times ]
```

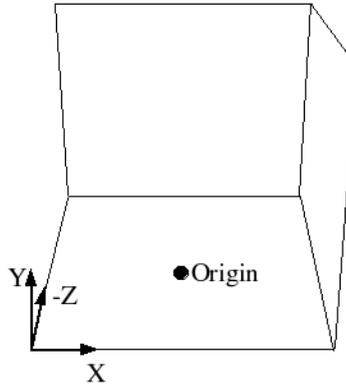
The important difference is that the position of the screen is measured in 3d physical coordinates, rather than 2d pixel coordinates. Note that all 3d coordinates given in the file are unit-less, but they must be consistent and in the same frame of reference, which is referred to as "display coordinate space".

The designations bottom/top refer to the minimum Y/maximum Y of the region, and left/right refer to the minimum X/maximum X of the region. In some cases 'bottom' may be near the ceiling, and 'top' may be near the floor, such as when a projector is mounted in an inverted position.

When determining the proper coordinates to use it is invaluable to sketch out the display environment, label the corners of each screen, and mark the location of the origin of the coordinate system. While it is possible to choose a coordinate system arbitrarily, it is usually easier to make the display coordinate system and the tracking coordinate system the same.

### Example 5

For the purpose of illustration consider the following example. Two projectors are pointed at screens which form a right angle, as illustrated below. The projected images are 10 feet wide by 7.5 feet high. The tracking system is calibrated in units of feet with the origin on the floor in the middle of the room.



```
PRsD 2.0
numpipes
  2
pipe
  xserver :0.2
  resolution 1024 768
  bottomleft -5 0.0 -5
  bottomright 5 0.0 -5
  topleft -5 7.5 -5
pipe
  xserver :0.1
  resolution 1024 768
  bottomleft 5 7.5 -5
  bottomright 5 7.5 5
  topleft 5 7.5 -5
```

Without head-tracking, this example is not yet very useful. The default position of the viewer is at (0,0,0), which is on the floor in the chosen coordinate system. There is an optional `view` section that can be inserted before the `numpipes` keyword of the configuration file to change these defaults:

```
view
  [ origin <x> <y> <z> ]
  [ zaxis <nx> <ny> <nz> ]
  [ yaxis <nx> <ny> <nz> ]
  [ center <x> <y> <z> ]
  [ scale <factor> ]
  [ eyesep <d> ]
```

The `origin` specifies the position of the viewer, and is only used if head-tracking has not been enabled. The `zaxis` and `yaxis` are unit vectors that allow the specification of a default orientation for objects placed in the scene. The

default values are (0,0,-1) for `zaxis` and (0,1,0) for `yaxis`. From the origin vantage point, it is useful to think of `zaxis` as the direction that the viewer is looking and `yaxis` as the 'up' direction.

The `center` and `scale` parameters allow you to position and size the scene for your display. If these parameters are not given, EnSight will compute a bounding box from the 3d coordinates given in the `bottomleft`, `bottomright`, and `topleft` parameters for the screens. The default center will be at the center of this box and the default scale will be computed so that your EnSight scene will fill the 3d space. Specifying a scale factor of 1.0 may be useful if your display coordinates were designed to coincide with your model coordinates. This will allow you to view your models life-sized.

The `eyesep` parameter allows an exact setting of the stereo separation between the eyes.

### Example 5a

Extending our example, we can position the viewer at the opposite corner of the room at a height of 5.75 feet:

```
PRsd 2.0
view
  origin -5 5.75 5
numpipes
  2
pipe
  xserver :0.2
  resolution 1024 768
  bottomleft -5 0.0 -5
  bottomright 5 0.0 -5
  topleft -5 7.5 -5
pipe
  xserver :0.1
  resolution 1024 768
  bottomleft 5 0.0 -5
  bottomright 5 0.0 5
  topleft 5 7.5 -5
```

### Example 5a-sim

It is relatively straightforward to test large displays and VR environments on a smaller system with a different number of graphics pipes. This can be accomplished by creating a configuration file that maps the pipes to smaller regions on a single monitor. As an example we will take the immersive configuration from Example 5a and modify it to run on a single display, with the modified regions shown in bold text.

```
PRsd 2.0
view
  origin -5 5.75 5
numpipes
  2
```

```

pipe
  xserver :0.0
  resolution 320 240
  bottomleft -5 0.0 -5
  bottomright 5 0.0 -5
  topleft -5 7.5 -5
pipe
  xserver :0.0
  xorigin 320 0
  resolution 320 240
  bottomleft 5 0.0 -5
  bottomright 5 0.0 5
  topleft 5 7.5 -5

```

Note that this method makes use of the `xorigin` parameter so that the resulting windows do not overlap. The default value for `xorigin` is (0,0) for each pipe. In a similar manner it is also possible to simulate large display walls on a single pipe.

## TRACKING

EnSight supports tracking and input with 6 DOF devices through a defined API. Pre-built libraries are provided to interface with `trackd` ((C) VRCO, Inc., [www.vrco.com](http://www.vrco.com)) on SGI, Sun, and Linux(x86), or the user may write a custom interface to other devices or libraries. The tracking library is specified with the `ENSIGHT7_INPUT` environment variable. To select `trackd`, use:

```
setenv ENSIGHT7_INPUT trackd (for csh or equivalent users)
```

The value of `ENSIGHT7_INPUT` can either be a fully-qualified path and filename or simply the name of the driver, in which case EnSight will load the library `libuserd_input.so` from directory:

```
$CEI_HOME/ensight76/machines/$CEI_ARCH/lib_input/
$ENSIGHT7_INPUT/
```

For the `trackd` interface you will also need to set:

```
ENSIGHT7_TRACKER_KEY <num>
ENSIGHT7_CONTROLLER_KEY <num>
```

in order to specify the shared-memory keys for the input library to interact with `trackd`. You should be able to find these values in your `trackd.conf` configuration file. For information on the API which allows you to interface to other tracking libraries or devices, please see the `README.v2` file in `$CEI_HOME/ensight76/src/input`.

With the environment variables set, you are ready to activate tracking. There are two parts to this. First, `trackd` operates as a daemon that is run independent of EnSight. If your input interface includes a separate program, you can run it at this time. For `trackd` users, it is often useful during configuration to invoke `trackd` with the `-status` option, so that you can see the information on your input devices. Once any external programs are started, you can enable tracking in

EnSight. From the 'Preferences->User Defined Input' menu, there is a toggle button which turns tracking on and off.

The trackd driver shipped with EnSight also has a debug mode that can be activated as follows:

```
setenv ENSIGHT7_TRACKD_DEBUG 1
```

This is similar to the trackd -status option, but it reports the input as seen by the EnSight trackd interface.

Once the EnSight client has been correctly interfaced to a tracking system you can add a section to the configuration file in order to calibrate the tracking with the display frame and customize the behavior of various interactions. The syntax for the section is:

```
tracker
  [ origin <x> <y> <z> ]
  [ zaxis  <x> <y> <z> ]
  [ yaxis  <x> <y> <z> ]
  [ headtracker <i> ]
  [ cursortracker <i> ]
  [ selectbutton <i> ]
  [ rotatebutton <i> ]
  [ transbutton <i> ]
  [ zoombutton <i> ]
  [ xformbutton <i> ]
  [ transxval <i> ]
  [ transyval <i> ]
  [ transzval <i> ]
```

The `origin`, `zaxis` and `yaxis` parameters allow you to calibrate the tracker to your defined display coordinate space. Many tracking libraries (including trackd) have options to perform similar transformations, and you may omit these parameters if you have defined your display coordinate frame in terms of the native tracker coordinate frame. The `zaxis` and `yaxis` vectors need not be unit length. If your tracker coordinates are in inches but you find it more convenient to specify your display coordinate in centimeters, you might include:

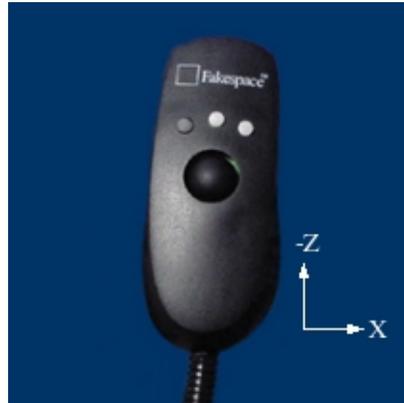
```
tracker
  zaxis 0.00 0.00 -2.54
  yaxis 0.00 2.54 0.00
```

The `headtracker` and `cursortracker` parameters allow you to specify which tracking device is tracking head position and which is tracking the controller. At this time only two devices can be tracked by EnSight – one for the head position and one for the position of the controller. All button/valuator input is interpreted as having come from the controller. Note that the EnSight API for input devices uses 0-based indices for trackers, buttons, and valuator. Trackd uses 1-based indices, and other libraries may differ as well.

The remaining options allow you to customize the behavior of buttons and valuators on the 6D input device. The input device can be used for:

1. Selecting items from the 3D GUI, which includes the heads-up macro (HUM) panel, the part list, variable list, and value slider.
2. Performing transformations on the geometry in the scene.
3. Manipulating the cursor, line, plane, and quadric tools.

The input device has a local coordinate system which is relevant for some forms of 6d interaction:



The default mode defines button 0 as the select button. When the 3D GUI is visible, you can point at the 3D buttons and the item that you are pointing at will be displayed in a highlight color. When you press the select button you will activate the current selection. For the HUM panel, this means that you will activate the macro that is defined for the selected button. You will find example macro files and additional instructions in `$CEI_HOME/ensight76/src/input/README.v2`. Clicking on an item in the part list will select or unselect the item in the list. Combined with macros in the HUM, this will allow you to modify visibility or other attributes on a part or collection of parts. If there are many parts in the part list, you can also select the scrollbar and move the controller up and down to scroll through the list. Similarly, the part-value slider can be used to modify part attributes for certain part types. For isosurfaces you can select the part slider and move left to right to change the isovalue. When no parts are selected, the part-value slider can be used to modify the time in a transient simulation.

The `rotatebutton`, `transbutton`, and `zoombutton` allow you to perform the selected transformations using gestures with the 6d input device. The `xformbutton` allows you to link a button to the current transformation mode, similar to the mouse button configurations for the main GUI interactions. You may want to add buttons on the heads-up-macro (HUM) panel to switch between modes. This is useful for 6D input devices with a smaller number of buttons. Note that it is possible (and encouraged) to re-use the `selectbutton` for a transformation. The `selectbutton` is only used when you are pointing at a heads-up menu. When you are not pointing at a menu, the same button could be used as the `xformbutton`, for example.

All 6d transformations have a 'sensitivity' which can be set to control the speed at which the transformation occurs. These values can be set from the 'Edit->Preferences->User Defined Input' dialog. There are also two forms of rotation

available. In ‘Mixed Mode’, the 6d device acts similar to a mouse for rotation. Once you click the rotatebutton, your movement is tracked in the X-Y plane of the input device. Your translation in this space is mapped to a rotation in the 3D space. In ‘Direct Mode’ it is the orientation of the device, rather than the position of the device, which controls the rotation.

The `transxval`, `transyval`, and `transzval` parameters configure the valuators to allow for translation of the scene by pressing the valuator in a given direction. The 'x', 'y', and 'z' designations refer to a local coordinate system which is fixed to the controller input device. As you hold the device in your hand, positive x is to the right, positive y is up, and positive z is toward the viewer. This local coordinate system depends on the orientation of the tracking device attached to the input device. It may be necessary to align the tracking device properly or modify the trackd (or other tracking library) configuration to achieve the proper orientation.

### Example 6

For the most basic configuration with head-tracking and a 6d input device, there are only three lines added to Example 5 to create the `tracking` section:

```
PRsD 2.0
numpipes
  2
view
  origin -5 5.75 5
tracker
  headtracker 0
  cursortracker 1
pipe
  xserver :0.2
  resolution 1024 768
  bottomleft -5 0.0 -5
  bottomright 5 0.0 -5
  topleft -5 7.5 -5
pipe
  xserver :0.1
  resolution 1024 768
  bottomleft 5 0.0 -5
  bottomright 5 0.0 5
  topleft 5 7.5 -5
```

### Example 6a

There are many different input devices available, and some have additional buttons and valuators that can be used for navigation and selection in immersive environments. In this example the configuration file is extended to use different buttons for rotation, translation, zoom, and selection. We also configure a ‘thumbwheel’ input to provide translation in the X-Z plane.

```
PRsD 2.0
numpipes
  2
view
```

```

    origin -5 5.75 5
tracker
  headtracker 0
  cursortracker 1
  selectbutton 4
  rotatebutton 0
  transbutton 1
  zoombutton 2
  xtransval 0
  ztransval 1
pipe
  xserver :0.2
  resolution 1024 768
  bottomleft -5 0.0 -5
  bottomright 5 0.0 -5
  topleft -5 7.5 -5
pipe
  xserver :0.1
  resolution 1024 768
  bottomleft 5 0.0 -5
  bottomright 5 0.0 5
  topleft 5 7.5 -5

```

## ANNOTATIONS

Annotations in EnSight include the heads-up macro panel, text, lines, logos, legends, and plots. In the GUI display these items appear as an overlay which is fixed in screen space. In an immersive display environment it is useful to be able to specify the locations of these objects. In EnSight 7.6, these items continue to occupy a plane in the 3D world. By default, this plane will coincide with the first pipe in the configuration file. The user may choose to specify the position and orientation of this plane with the following addition to the configuration file:

```

annot
  [ pipe <n> ]
    OR
  [
    center <x> <y> <z>
    zaxis <x> <y> <z>
    yaxis <x> <y> <z>
    xscale <float>
    yscale <float>
  ]

```

### Example 7

To continue with Example 6, suppose that the user would prefer for the annotations to appear on the right wall instead of the left wall. The following configuration file defines an annot section with the appropriate parameters to do this:

```

PRSD 2.0
numpipes

```

```

2
view
  origin -5 5.75 5
tracker
  headtracker 0
  cursortracker 1
  selectbutton 4
  rotatebutton 0
  transbutton 1
  zoombutton 2
  xtransval 0
  xtransval 1
annot
  pipe 1
pipe
  xserver :0.2
  resolution 1024 768
  bottomleft -5 0.0 -5
  bottomright 5 0.0 -5
  topleft -5 7.5 -5
pipe
  xserver :0.1
  resolution 1024 768
  bottomleft 5 0.0 -5
  bottomright 5 0.0 5
  topleft 5 7.5 -5

```

Fixing the annotations to a pipe is merely provided as a convenience. Internally this is identical to using the explicit form:

```

annot
  center 5 3.75 0
  zaxis 1 0 0
  yaxis 0 1 0
  xscale 10
  yscale 7.5

```

## STEREO DISPLAY

When using a detached display (either a wall or an immersive configuration) the created windows will be monoscopic by default. If you want the display to be initialized in stereo, you can simply add the keyword to the configuration file between any of the other sections.

```
stereo
```

This keyword is not necessary for passive stereo displays, which are always in stereo.

## TIPS

1. Use the `-bbox` command-line option when using a detached display. This will cause EnSight to draw only bounding boxes in the GUI window, which

will improve performance.

2. Make sure that the displays that are in your configuration files are valid. An X display identifier looks like: `<host>:<server>.<screen>`. If you have constructed a valid X display you should be able to set the `$DISPLAY` environment variable with the given string and run `xterm` or another X11 application.
3. Tracking is the most difficult part of configuring a system. Make sure that you are confident in the display configuration before you activate tracking. It may be useful to manually position the view origin in several different locations in order to verify that the display coordinate system is as you expected. In the examples installed from your CD-ROM example 5 is extended with several different view origins to demonstrate this technique.

