Appendix A: Development of the Installation Procedure

CHAPTER I. DEFINITIONS AND CONVENTIONS

Throughout the discussion that follows:

- "Player" refers to the Player server.
- "Gazebo" refers to the Gazebo high-fidelity robot simulator.
- The versions of relevant applications and source are documented herein.
- When a specific version of a documented or undocumented dependency was required by installation instructions, that version is documented herein. If no specific version was required, "-none-" is recorded.
- Development (header and library) files for documented or undocumented dependencies were installed where available.

CHAPTER II. METHODOLOGY

II.A. Use current, stable, and release versions of applications and source

Because the potential simulation targets required unknown potential changes to Player, Gazebo, and ODE, the author developed an installation procedure to ensure a reproducible simulation environment and establish a reliable baseline from which to proceed with any changes.

Ideally, the author would have used versions of applications and source available during the 2004 or 2005 GCE. With a few exceptions, they are no longer available. Although the author has versions of Player and Gazebo dating from 2004 and 2005, specifically Player 1.6 and Gazebo 0.5.1, the author concluded attempting to base the simulation environment on these versions of Player and Gazebo would ultimately be unproductive due to problems encountered during the development of the installation procedure.

As a result, the simulation environment was based on current, stable, release versions of applications and source, with one exception: Gazebo. As noted, ideally the author would have used versions of applications and source available during the 2004 and 2005 GCE. However, the author asserts an increase in processing power was a key factor during the 2005 GCE, and contends that, if ten percent of the development cost of team challenge vehicles had been invested in improvements to the simulation environment, resulting in more rapid development, the same increase in processing power may have been a key factor during the 2005 GCE by allowing the use of more realistic simulation, for example, to compare the performance and capabilities of various SICK LIDAR

sensors or to effectively visualize the interaction of the challenge vehicle with the environment.

The author concluded an exception must be made for Gazebo when repeated attempts to compile the current, stable, release version of Gazebo ("gazebo-0.9.0.tar.bz2") failed due to "undefined reference" errors to "FreeImage_Rescale" and "FreeImage_ConvertFromRawBits".

These errors also occurred during the initial attempt to develop the installation procedure (herein "initial attempt"). Although the author concluded the problem was caused by linking errors, and most probably by a missing or inaccessible (for whatever reason) library, attempts to manually compile past this point, modify configuration files, and engage the community directly were unsuccessful. The author was unable to resolve the errors and eventually abandoned the initial attempt. Similar errors occurred with OpenGL and FFmpeg (specifically libavcodec) during the initial attempt, which were successfully resolved.

As a result, the author downloaded the latest revision (revision 8443) of the Gazebo 0.9.0 source code using `svn`:

```
$ svn co https://playerstage.svn.sourceforge.net/
svnroot/playerstage/code/gazebo/trunk gazebo
```

II.B. Use documented installation instructions, when available

Based on comments made by teams participating in the 2004 and 2005 GCE, the author concluded most teams wanting to use Player and Gazebo as a simulation environment would not have had the time required to troubleshoot an installation

procedure, and would have relied on documented installation instructions, where available. As a result, that author installed applications or source code in accordance with installation instructions documented by the application's developer(s) or through the use of an automated tool when possible.

As used herein, “documented installation instructions” means installation instructions included with applications and source code (“packaged documentation”) or installation instructions available through online documentation (“online documentation”). When both packaged documentation and online documentation was available, the author favored online documentation because he believed it would be more up-to-date than packaged documentation. As noted throughout the paragraphs that follow, this was true for some applications but not others.

Reliance on documented installation instructions caused several problems:

- No comprehensive installation procedure was available for Gazebo. Gazebo required several third-party libraries and the lack of a comprehensive installation procedure was one of the two significant problems resolved while developing this installation procedure. [78] published an alternate comprehensive installation procedure based on Fedora 9, in lieu of openSUSE 11.2, and earlier versions of Player, Gazebo, and their dependencies. See paragraph II.D.
- Installation instructions for some applications were incomplete.
- Installation instructions for some applications were incorrect.
- No reliable list of dependencies was available for some applications, and, in fact, the definition of a “dependency” varied from application to application.

Configuration of some packages failed because “optional” libraries were not installed. This was the cause of one of the two significant problems resolved while developing this installation procedure. See paragraphs II.C.1. and III.G.4.

- There were conflicts between packaged documentation and online documentation which had to be resolved on a case basis.

II.C. Troubleshoot the installation procedure

The initial attempt to develop an installation procedure for the simulation environment failed. Attempting to resolve errors in documentation, in particular inadequately documented dependencies and conflicting installation instructions, consumed several weeks during which more productive research was delayed.

Problems encountered during the initial attempt and later successful attempt to develop an installation procedure, and their resolutions, are documented herein. Because the author revised the order of installation, used a later revision of Gazebo, and maintained inadequate records of the initial attempt, problems encountered during the initial attempt may not be reproducible using this order of installation and revision of Gazebo.

However, the author developed a method for troubleshooting the installation procedure as a result of experimentation and review of packaged and online documentation which was used to develop the installation procedure:

II.C.1. “Optional” libraries

During the initial attempt, configuration of some applications and source failed because “optional” libraries were not installed. Because configuration should not have

failed because optional libraries were not installed, the author considered these undocumented dependencies, and resolved the problem when it occurred by installing the undocumented dependency.

However, due to problems encountered during the initial attempt ordering the installation procedure and determining which applications and source were true dependencies, the author decided to limit the use of "optional" libraries to simplify the installation procedure to the extent possible during the development of the installation procedure. Any resulting errors were dispositioned on a case basis as either configuration errors or evidence of undocumented dependencies. In general, configuration errors were resolved by disabling some feature during configuration and undocumented dependencies were resolved by installing the undocumented dependency. Resolution is documented throughout this Appendix.

II.C.2. Order the installation procedure

Because of inadequately documented dependencies, the author used 5" X 8" index cards to record application name, package or source file name, documented dependencies, undocumented dependencies as they were noted, specific configuration options, installation instructions used to compile applications from packages or source, and any information necessary to troubleshoot installation for each application. This allowed the author to easily re-order the steps of the installation procedure as necessary.

II.C.3. Maintain a record of errors encountered

The initial attempt ultimately failed when the author was unable to resolve many "undefined reference" errors to OpenGL, avcodec, and FreeImage functions while

attempting to build the Gazebo executable. The specific step during which the "undefined reference" errors occurred was "Linking CXX executable gazebo".

Attempts to determine the exact cause of the errors by reviewing the Gazebo mailing list archives were unsuccessful. An attempt by the author to solicit help by engaging the Player Project's community more directly was also unsuccessful. A reply to a post to the playerstage-gazebo mailing list on 25 September 2009 stated: "The latest SVN version of Gazebo should have this problem fixed." However, the author was unable to determine the cause of the problem from the response and the problem was not resolved. Attempts to compile the "latest SVN version" of Gazebo failed with the same errors.

While attempting to resolve the errors, openSUSE 11.2 was released. The author upgraded to openSUSE 11.2 from 11.1, and subsequently developed the installation procedure documented by this Appendix. The initial attempt was not preserved, and the author retained only a few records of errors encountered while attempting to build a reproducible simulation environment. The author concluded this was a mistake which made it more difficult to troubleshoot specific errors encountered during the initial attempt because it was impossible to determine the effects of specific actions taken to identify or resolve errors without being able to compare output from one attempted solution to the next.

This resulted in a change in methodology. During the development of this installation procedure, standard output and standard error from `./bootstrap`, `./configure`, `make`, `cmake`, and `make install` commands were re-directed to

files to document any errors and their successful resolution.

II.D. Comparison between the installation procedure and alternate installation procedures

II.D.1. First alternate comprehensive procedure

[78] published an alternate comprehensive installation procedure ("alternate procedure") for Gazebo 0.7.0 and 0.8.0, which was based on [79], and which pre-dates [80] and [81]. The alternate procedure for Gazebo 0.8.0 ([82]) was based on Fedora 9, in lieu of openSUSE 11.2. Based on the initial attempt, the author determined there were several additional problems with the use of the alternate procedure for Gazebo 0.8.0, in addition to those noted above:

II.D.1.a. Problem: The alternate procedure refers, incorrectly, to the NVIDIA Cg library as an "OGRE dependency"

[82] states: "OGRE dependency: nVidia Cg Toolkit". The author considers this to be additional evidence supporting the author's assertion that the definition of "dependency" varies from application to application, which is discussed in more detail throughout this Appendix.

II.D.1.b. Resolution: Re-evaluate the installation of Cg

The author re-evaluated the installation of the NVIDIA Cg library ("Cg"), and revised the installation procedure to include step "Install Cg". See paragraph III.H.

II.D.1.c. Problem: The alternate procedure does not require the installation of CEGUI

[82] does not require the installation of CEGUI.

II.D.1.d. Resolution: Re-evaluate the installation of CEGUI

The author determined the only purpose of CEGUI in the installation procedure was to provide support for OGRE demos. The successful installation of Gazebo was a research objective. The author's only interest in OGRE was as a dependency for Gazebo. As a result, the author re-evaluated the installation of CEGUI and revised the installation procedure to delete step "Install CEGUI". See paragraph III.G.4.

II.D.1.e. Confirm path environment variables

After the author successfully installed Gazebo using this installation procedure, the first attempt to run Gazebo resulted in the following error:

```
error while loading shared libraries:
libavformat.so.52
```

This was similar to errors encountered during the initial attempt. The command:

```
export LD_LIBRARY_PATH=/usr/local/lib:$LD_LIBRARY_PATH
```

appeared to resolve the error, but the author did not modify `.bashrc`, `.profile`, or any variants thereof and subsequent attempts to run `gazebo <worldfile>` from a new shell were successful without exporting `LD_LIBRARY_PATH`. This error did not recur during the verification of the installation procedure. As a result, this error was not reproducible.

However, as a result of this error and similar errors encountered during the initial attempt, the author re-evaluated the need to either confirm the following path environment variables include the following paths prior to verification of the installation procedure, or export them as necessary, as noted by the alternate procedure:

```
export PATH=/usr/local/bin:$PATH
export CPATH=/usr/local/include:$CPATH
export LIBRARY_PATH=/usr/local/lib:$LIBRARY_PATH
export PKG_CONFIG_PATH=/usr/local/lib/pkgconfig:
$PKG_CONFIG_PATH
```

The author revised the installation procedure to include step “Path environment variables”.

II.D.2. Second alternate installation procedure

Shortly after verification of the installation procedure in accordance with Appendix C, the author located a second alternate installation procedure available via [38] while reviewing documentation for Gazebo world files and Player configuration files. The installation instructions are hosted on the project wiki, which provides a kind of version control documented by the page “history”. Review of the history indicates an earlier version of the second alternate installation procedure was available while the author was attempting to develop the installation procedure documented in Appendix B in accordance with this Appendix. As a result, the earlier revision ([83], which is dated March 3, 2009) is referred to as the “second alternate installation procedure” herein. The later revision of this procedure ([84], which is dated January 12, 2010) is referred to as the “revised second alternate installation procedure” herein.

The second installation procedure identified the following documented dependencies:

- `pkg-config`
- `zziplib`

- FreeImage
- OGRE
- ODE

The author noted the following problems with the second installation procedure. The second installation procedure:

- provides no installation instructions for Gazebo
- does not refer to the installation of development (header and library) files, for example `zziplib-devel` for OGRE and `libxml2-devel` for Gazebo
- does not identify compatible versions of documented dependencies
- does not identify `libxml2` as a dependency
- does not identify OIS as a dependency
- does not identify CEGUI as a dependency
- does not identify Cg as a dependency, although it does recommend the use of the `--disable-cg` flag to disable building Cg support when configuring OGRE
- does not require Player to be installed before Gazebo

The revised second installation procedure identified the following documented dependencies:

- FreeImage >= 3.10
- OGRE 1.70
- ODE 0.11.1

- boost >= 1.35
- FLTK 1.1

The revised second installation procedure provided installation instructions for Gazebo, conforming in general to [81], and includes a section titled “Troubleshooting the Install”. The author noted the following problems with the revised second installation procedure. The revised second installation procedure:

- does not identify `zziplib` as a documented dependency of Gazebo
- does not refer to the installation of development (header and library) files, for example `zziplib-devel` for OGRE and `libxml2-devel` for Gazebo
- does not identify `libxml2` as a dependency
- does not identify OIS as a dependency
- does not identify CEGUI as a dependency, and does not recommend the use of the `--disable-ogre-demos` flag to force `./configure` to continue without building the OGRE demos when configuring OGRE
- does not identify Cg as a dependency, and does not require the use of the `--disable-cg` flag to disable building Cg support when configuring OGRE
- does not require Player to be installed before Gazebo

Section “Troubleshooting the Install” states, in part: “If gazebo doesn't compile... Check the output of the configure step. Resolve all errors by installing the necessary 3rd party packages.” However, as noted throughout this Appendix, during the initial attempt,

configuration of some applications and source failed because optional libraries were not installed. Although installation of Gazebo did not result in any configuration errors due to optional libraries, installation of OGRE, which is a documented dependency of Gazebo, failed due to configuration errors. These errors initially caused the author to install CEGUI, leading to one of the two significant problems encountered.

Section "Troubleshooting the Install" also states, in part: "If gazebo doesn't compile... Make sure that the 3rd party packages are the correct versions." However, installation of Gazebo failed because a later version of a documented dependency was used (OGRE version 1.6.4 was used in lieu of 1.6.3 during the initial attempt). In addition, to configure Gazebo a later version of a documented dependency was required than was documented (OGRE version 1.6.3 in lieu of 1.4.4. See paragraph III.L.2.a.).

II.D.2.a. Conformance of the installation procedure to the second alternate installation procedure

Aside from the use of `scons`, which was the build system used by Gazebo 0.8.0, the alternate procedure generally conforms to the installation procedure developed by the author, with the following exceptions:

- The order of installation of all applications and source is similar but not identical. However, the order of installation of the group of OGRE dependencies is relatively unimportant since they are not interdependent unless CEGUI is also installed. As a result, the true order of installation is: OGRE dependencies, OGRE, Player, and Gazebo. ODE, having no dependencies other those provided by the base installation, may be installed at any time prior to Gazebo.

- The names and versions of installed applications and source are different between Fedora 9 and openSUSE 11.2.
- The alternate procedure uses `yum` in lieu of YaST as a package manager.
- A base installation of Fedora 9 provides different installed packages than openSUSE 11.2, requiring the installation of different dependencies. For example, both the alternate procedure and the installation procedure require the installation of `mesa` and `mesa-devel` (as noted above, the names of installed applications and source are different), but the alternate procedure does not require the installation of `libxml2-devel`, which is a documented dependency of Gazebo.

Therefore, although the order of installation of dependencies selected by the author differs, the elimination of configuration errors which caused various installation failures for the three interdependent packages OIS, CEGUI, and OGRE by revising the "Install OGRE" step to use the `--disable-ogre-demos` flag to force `./configure` to continue without building the OGRE demos resolved one of the two significant problems the author encountered by eliminating the interdependence between OIS, CEGUI, and OGRE.

Resolving this problem during the initial attempt may have eliminated weeks of troubleshooting, and may have resulted in a working installation of Gazebo months before the author was able to verify the installation procedure. Although the author includes detail in this Appendix to provide a more comprehensive history of the

development of the installation procedure, including the installation of CEGUI, the author cannot stress enough that if the successful installation of Gazebo, not OGRE, is a research goal, CEGUI should, under no circumstances, be installed.

II.D.2.b. Resolution of problems noted during review of the second and revised second alternate installation procedures

Because the author successfully verified the installation procedure documented in Appendix B in accordance with Appendix C, problems noted during review of the second and revised second alternate procedures were not dispositioned individually in paragraph II.D.2., but rather as documented below:

- Installation instructions for Gazebo documented by [84] confirm the installation instructions documented by Appendix B, step “Install Gazebo”.
- Development (header and library) files for documented and undocumented dependencies should be installed where available.
- There are no “correct” versions of “3rd party packages”. Versions of applications and source other than those documented by [84] may be used, for example OGRE version 1.6.4 in lieu of 1.7.0 and FLTK version 1.1.9 in lieu of 1.1.
- Packages `zziplib` and `zziplib-devel` are documented dependencies of OGRE, not Gazebo. The author concluded [83] is in error.
- Packages `libxml2` and `libxml2-devel` are documented dependencies of Gazebo. Output of the `./cmake ..` command stated, in part:

```
--checking for module 'libxml-2.0'
--  found libxml02.0, version 2.7.3
```

The author concluded [84] is in error.

- [85] states OIS is a documented dependency of Gazebo. The author concluded this may be an error. Output of the `./cmake ..` command does not confirm `cmake` checks for the presence of OIS when configuring Gazebo. The author did not revise the installation procedure to test this, and notes OIS may be an undocumented dependency of OGRE if CEGUI is also installed due to the interdependence between OIS, CEGUI, and OGRE. See paragraphs III.D.1.a., III.G.2.a., and III.I.1.d.
- CEGUI is not a dependency of OGRE, but OGRE must be compiled with the `--disable-ogre-demos` flag to eliminate a configuration error which may be misinterpreted as a statement that CEGUI is a "necessary 3rd party package" when configuring OGRE.
- Cg is not a dependency of OGRE, but OGRE must be compiled with the `--disable-cg` flag to eliminate a configuration error which may be misinterpreted as a statement that Cg is a "necessary 3rd party package" when configuring OGRE.
- Player must be installed before Gazebo.
- Installation instructions for Gazebo documented by [84] confirm `boost-devel` was an undocumented dependency of Gazebo. See paragraph III.L.2.g.

Overall, the author concluded neither [83] nor [84] were comprehensive, or represented sufficient improvement over the installation procedure documented by

Appendix B to cause the author to revise Appendix B.

CHAPTER III. DEVELOPMENT OF THE INSTALLATION PROCEDURE

III.A. Base installation

Package (source package name)	Description (Version)
openSUSE (openSUSE-11.2-NET-i586.iso)	openSUSE Linux distribution (11.2)
Base Development	A minimal set of tools for compiling and linking applications (11.2)
C/C++ Development	Tools and libraries for software development using C/C++ and other derivatives of the C programming language (11.2)
nvidia_gfx_kmp_default	NVIDIA graphics driver kernel module for GeForce4 GPUs (96.43.11_2.6.31.5_0.1-20.2)
x11_video_nvidia	NVIDIA graphics driver for GeForce4 GPUs (96.43.11-21.4)

Based on familiarity with several generations of the SUSE Linux distribution, in particular YaST (“Yet another Setup Tool”) for installation of software and management of software updates, the author installed openSUSE 11.2 using the “Internet Installation Boot Image” (“openSUSE-11.2-NET-i586.iso”) disk image.

The base installation included the following package groups and packages:

- the default selections for a KDE-based desktop
- “Base Development” package group (YaST pattern view)
- “C/C++ Development” package group (YaST pattern view)
- NVIDIA graphics driver kernel module

  (“nvidia-gfx-kmp-default 96.43.11_2.6.31.5_0.1-20.2”)
- NVIDIA graphics driver (“x11-video-nvidia 96.43.11-21.4”)

The NVIDIA graphics driver kernel module and graphics driver were installed from the NVIDIA repository (http://download.nvidia.com/opensuse/11.2) in accordance with [86] and [87] using YaST. All other packages were installed from the openSUSE repository (http://download.opensuse.org/).

Before installing any additional applications from packages or source, the author archived the base installation using the YaST "System Backup" utility. The archive took several hours to complete.

III.B. Path environment variables

Confirm the following path environment variables include the following paths, or export them as necessary:

```
export PATH=/usr/local/bin:$PATH

export CPATH=/usr/local/include:$CPATH

export LIBRARY_PATH=/usr/local/lib:$LIBRARY_PATH

export PKG_CONFIG_PATH=/usr/local/lib/pkgconfig:

$PKG_CONFIG_PATH
```

III.C. FreeImage

Package (source package name)	Description (Version)
FreeImage (FreeImage3130.zip)	Open source image library (3.13.0)

III.C.1. Dependencies

Documented dependency	Description (Version)
-none-	

Undocumented dependency	Description (Version)
-none-	

III.C.2. Installation instructions

[88] provided documented installation instructions.

III.C.3. Install FreeImage

Compile and install FreeImage as follows:

- `$ make`
- `$ su`
- `$ make install`
- `$ exit`

III.D. Object-oriented Input System (OIS)

Package (source package name)	Description (Version)
OIS (ois_1.2.0.tar.gz)	Cross-platform object-oriented library for handling input devices (1.2.0)

III.D.1. Dependencies

Documented dependency	Description (Version)
X11	X Window System (-none-)
OGRE (GLX Platform)	Object-oriented Graphics Rendering Engine (-none-)
CEGUI (if building CEGUIOgre OIS Demo)	Library providing windowing and widgets for graphics APIs / engines (0.4.0)
"Newer Linux Kernel" (for Event API, otherwise use --disable-joyevents)	Linux kernel (2.6+ ?)

Undocumented dependency	Description (Version)
-none-	

III.D.1.a. Problem: OIS, CEGUI, and OGRE are interdependent

The following dependencies were documented by [89]:

- X11
- Ogre (GLX Platform) & CEGUI 0.4.0 If building CEGUIOgre OIS Demo
- Newer Linux Kernel (2.6+?) for Event API - else, use --disable-joyevents

OGRE is a documented dependency for OIS if building the "CEGUIOgre OIS demo". OIS is an undocumented dependency for CEGUI if building "OGRE CEGUI demos". OGRE cannot be installed before CEGUI, therefore OGRE cannot be installed before OIS. See paragraphs III.G.2.a. and III.I.1.d.

III.D.1.b. Resolution: None.

This problem had no known impact on the successful installation of OIS because the author had no interest in either the "CEGUIOgre OIS demo" or "OGRE CEGUI demos".

III.D.2. Installation instructions

[89] provided documented installation instructions.

III.D.2.a. Problem: Option `--disable-joyevents` is an unrecognized option.

The author configured the installation of OIS using:

```
./configure --disable-joyevents
```

which resulted in the following error:

```
configure: WARNING: unrecognized options: --disable-joyevents.
```

III.D.2.b. Resolution: None. Option `--disable-joyevents` is a valid option.

[89] states option `--disable-joyevents` is a valid configuration option. This problem had no known impact on the successful installation of OIS. The author considers this a configuration error.

III.D.3. Install OIS

Configure, compile, and install OIS as follows:

- `$ ./bootstrap`
- `$ ./configure --disable-joyevents`

- `$ make`
- `$ su`
- `$ make install`
- `$ exit`

III.E. Open Dynamics Engine (ODE)

Package (source package name)	Description (Version)
ODE (ode-0.11.1.tar.gz)	Open source library for simulating rigid body physics (0.11.1)

III.E.1. Dependencies

Documented dependency	Description (Version)
-none-	

Undocumented dependency	Description (Version)
-none-	

III.E.2. Installation instructions

[90] provided documented installation instructions.

III.E.3. Install ODE

Configure, compile, and install ODE as follows:

- `$ ./configure`
- `$ make`
- `$ su`

- `$ make install`
- `$ exit`

III.F. Fast Light ToolKit (FLTK)

Package (source package name)	Description (Version)
FLTK	Cross-platform GUI toolkit (1.1.9)

III.F.1. Dependencies

Documented dependency	Description (Version)
X11 header and library files	X Window System files and libraries required for development (-none-)
OpenGL (or Mesa) header and library files	Open Graphics Library files and libraries required for development (-none-)
JPEG header and library files	JPEG files and libraries required for development (-none-)

Undocumented dependency	Description (Version)
-none-	

III.F.1.a. Install Mesa-devel

Package `Mesa 7.6-3.1` was installed as part of the base installation. To satisfy a documented dependency, the author installed package `Mesa-devel 7.6-3.1` using YaST. Package `libdrm-devel 2.4.14-2.1` was installed by YaST to resolve a dependency.

III.F.2. Installation instructions

[91] provided documented installation instructions.

III.F.2.a. Problem: FLTK failed to compile because of an “invalid conversion” error

The exact error is reproduced below:

```
filename_list.cxx: In function 'int
fl_filename_list(const char*, dirent***, int (*)
(dirent**, dirent**))':
filename_list.cxx:70: error: invalid conversion from
'int (*)(const void*, const void*)' to 'int(*)(const
dirent**, const dirent**)'
filename_list.cxx:70: error:   initializing argument 4
of 'int scandir(const char*, dirent***, int (*)(const
dirent*), int(*)(const dirent**, const dirent**))'
```

III.F.2.b. Resolution: Install FLTK using YaST

Because the teams did not have the time required to troubleshoot an installation procedure (see paragraph II.B.) and because FLTK was available for installation from the openSUSE repository using YaST, the author did not attempt to resolve the error, and instead installed packages `fltk 1.1.9-36.1` and `fltk-devel 1.1.9-36.1` using YaST.

III.F.3. Install FLTK

1. Install `Mesa-devel` (see paragraph III.F.1.a.).

2. Install `fltk` and `fltk-devel` from the openSUSE repository using YaST.

III.G. CrazyEddie's GUI System (CEGUI)

Package (source package name)	Description (Version)
`CEGUI` `(CEGUI-0.6.2b.tar.gz)`	`CrazyEddie's GUI System (CEGUI)` `(0.6.2b)`

III.G.1. Dependencies

[92] provided a list of documented dependencies.

Documented dependency	Description (Version)
`FreeType2`	`Software font engine` `(-none-)`
`PCRE`	`Perl-Compatible Regular Expressions library` `(-none-)`

III.G.1.a. Install `pcre-devel`

Packages `freetype2 2.3.9-2.2`, `freetype2-devel 2.3.9-2.2`, and `pcre 7.9.0-2.3.1` were installed as part of the base installation. To satisfy a documented dependency, the author installed package `pcre-devel 7.9.0-2.3.1` using YaST. Packages `libpcre0 7.9.0-2.3.1`, `libpcreposix0 7.9.0-2.3.1`, `libpcrecpp0 7.9.0-2.3.1` were installed by YaST to resolve a dependency.

Undocumented dependency	Description (Version)
`-none-`	`-none-`

III.G.2. Installation instructions

[93] provided documented installation instructions.

III.G.2.a. Problem: OIS, CEGUI, and OGRE are interdependent

The initial attempt was based on documented installation instructions and used documented dependencies to establish the installation procedure. OIS is a documented dependency of Gazebo, however OIS is not a documented dependency of CEGUI. Because OIS was not installed, configuration of CEGUI resulted in a warning which stated, in part:

```
You do not have OIS installed.  This is required to
build Ogre CEGUI demos.
```

and continued:

```
If you do not want to build the demos, you can safely
ignore this.
```

During the initial attempt, the author installed OIS before continuing with the installation of CEGUI.

III.G.2.b. Resolution: None.

The author revised the installation procedure to install OIS before CEGUI, thus resolving the problem. The author does not consider this a configuration error because configuration of CEGUI continued and was successfully completed.

Based on a review of online documentation, the author determined that the normal installation procedure for CEGUI and OGRE is: CEGUI, OGRE, then CEGUI again for

the “OGRE CEGUI demos”. OIS is required to be installed both before CEGUI and OGRE, so perhaps the normal installation procedure should be: OIS, CEGUI, OGRE, OIS (for the “CEGUIOgre Demo”), and finally CEGUI (for the “OGRE CEGUI demos”).

The author had no interest in CEGUI except as a dependency for OGRE, and so did not attempt to determine the correct order of installation to build the “CEGUIOgre Demo” or “OGRE CEGUI demos” using OIS, CEGUI, and OGRE. See paragraphs III.D.1.a. and III.I.1.d.

III.G.2.c. Problem: `install` failed when attempting to overwrite an existing just-created file

The very first line output by `./configure` is:

```
checking for a BSD-compatible install...
/usr/bin/install -c.
```

However, `man install` states the `-c` flag is `(ignored)` and `info install` states the `-c` flag is: `Ignored; for compatibility with old Unix versions of 'install'`. As a result, `./configure` reported success:

```
Now you can do make && make install.  Good luck!
```

But attempts to `make install` failed as a result of “will not overwrite just-created” file errors.

Based on a search of the CEGUI wiki ([93]) for similar problems reported by other users, the author initially determined this problem is due to a difference between the `-c` and `-C` options, but based on an evaluation of the results of attempting to `install` files twice in a single `install` invocation with and without the `--compare (-C)`

option finally concluded the error is due to the fact that install will attempt to copy a file onto itself, and fail when unsuccessful unless the `--compare (-C)` option is used.

By default, `install` fails when attempting to overwrite an existing just-created file. However, the `-C` flag causes `install` to ignore subsequent attempts to overwrite an existing just-created file. `man install` states the -C flag causes `install` to "compare each pair of source and destination files, and in some cases, do not modify the destination at all" but does not explain how this is prevented "in some cases".

The author reviewed the GNU Core Utilities ("`coreutils`") source code ([94]) to determine when the `-C` option prevents file copying. As a result of that review, the author concluded there is some confusion concerning the intended use of the `-C` option. The "ChangeLog" included with coreutils (`coreutils 7.1` was installed as part of the base installation) states functions `have_same_content` and `need_copy` were added to `install.c` when it was modified to recognize the `--compare (-C)` option to install files only when necessary.

`man install` recognizes the `--compare (-C)` option, but also states:

```
The full documentation for install is maintained as a
Texinfo manual.  If the info and install programs are
properly installed at your site, the command
     info coreutils 'install invocation'
should give you access to the complete manual.
```

However, the manual doesn't recognize the `--compare (-C)` option. The manual states (of `install`): "It refuses to copy files onto themselves." As

demonstrated below, this is misleading. `install` will copy a file onto itself, but not if the file name occurs more than once in the same invocation.

The "ChangeLog" states function `copy_file` was revised to: "Skip file copying if not necessary." However this is also misleading, and is not the observed behavior. Function `copy_file` and, by extension `install,` attempts to copy even if not necessary unless the `--compare (-C)` option is invoked.

After reviewing the relevant source code (`install.c` and `copy.c`), the author concluded the intended use of the `--compare (-C)` option is to cause `install` to check to see if there is a difference between two files by several methods and prevent copying if there is no difference between them if the option is used, but to otherwise allow copying to fail due to a "will not overwrite just-created" error if copying a file onto itself twice in the same invocation. As noted above, `install` will copy a file onto itself, and also fail when attempting to copy a file onto itself in the same invocation unless the `--compare (-C)` option is used.

The `make install` output indicates `install` failed on a line which attempted to install the files `CEGUIListHeader.h` and `CEGUIListHeaderProperties.h` twice. The author confirmed `install source destination` will copy a source file to the destination. Subsequent attempts to install the source file to the same destination will copy the source file to the destination after first removing the existing file at destination:

```
$ install -v test.h ./test
$ 'test.h' -> './test/test.h'
```

```
$ install -v test.h ./test
$ removed './test/test.h'
$ 'test.h' -> './test/test.h'
```

Attempting to copy the file twice with a single install command fails with a “will not overwrite just-created” file error if the file does not already exist at the destination:

```
$ install -v test.h test.h ./test
$ install: will not overwrite just-created
'./test/test' with 'test.h'
```

If the file already exists at the destination, attempting to copy the file twice with a single install command results in the file first being overwritten, then install failing as above with a “will not overwrite just-created” file error:

```
$ install -v test.h test.h ./test
$ removed './test/test.h'
$ 'test.h' -> './test/test.h'
$ install: will not overwrite just-created
'./test/test' with 'test.h'
```

Attempting to copy the file multiple times with a single `install` command when using the `-C` flag results in the file being installed at the destination, but `install` ignores subsequent attempts to overwrite the file:

```
$ install -C -v test.h test.h test.h ./test
$ 'test.h' -> './test/test.h'
```

If the file already exists at the destination, install ignores subsequent attempts to overwrite the file when using the `-C` flag:

```
$ install -C -v test.h ./test
$ install -C -v test.h test.h ./test
$ install -C -v test.h test.h test.h ./test
```

III.G.2.d. Resolution: revise `./configure` so that `install -C` is invoked in lieu of `install -c`

The author resolved this problem by modifying `./configure` to invoke `install -C` in lieu of `install -c` on lines 2325, 2329, 2333, 2408, and 2644. The author notes that the duplicate file names could also have been deleted from each line which invoked `install`.

III.G.3. Install CEGUI

1. Install `pcre-devel`.
2. Revise `./configure` to invoke `install -C` in lieu of `install -c`.
3. Configure, compile, and install CEGUI as follows:
   - `$ ./configure`
   - `$ make`
   - `$ su`
   - `$ make install`
   - `$ exit`

III.G.4. After developing the installation procedure, the author determined CEGUI was not an undocumented dependency

Neither [95] nor [85] refer to CEGUI as a dependency. [95] refers to CEGUI as

an “optional” library, and [85] identifies OGRE as a dependency. Despite being an “optional” library, the author concluded CEGUI was an undocumented dependency for OGRE based on output from `./configure` when attempting to install OGRE during the initial attempt. Resolution of problems identified during the installation of OIS, CEGUI, and OGRE during the initial attempt, in particular the required installation order made effective troubleshooting more difficult.

However, as noted in paragraph II.C. above, based on failure of the initial attempt the author developed a method for troubleshooting the installation procedure which was used to develop the installation procedure herein. Maintaining a record of errors encountered allowed the author to determine CEGUI was not an undocumented dependency during the development of this installation procedure. Also, because the author had no interest in building either the “CEGUIOgre Demo” when installing OIS or “OGRE CEGUI demos” when installing CEGUI, the author concluded CEGUI should not be installed.

As a result, the author revised the installation procedure to delete step “Install CEGUI” in its entirety, revised step “Install OGRE” to use the `--disable-ogre-demos` flag to force `./configure` to continue without building the OGRE demos, and confirmed OGRE installation using Cg during verification of the installation procedure.

The author considers this a configuration error. Configuration of OGRE should not have failed because the “optional” CEGUI library was not installed. However, the author does not consider CEGUI an undocumented dependency of OGRE because

CEGUI support could be disabled.

III.H. Cg

Package (source package name)	Description (Version)
`Cg`	`Compile and runtime libraries for the Cg graphics language (2.2)`

III.H.1. Dependencies

Documented dependency	Description (Version)
`-none-`	

Undocumented dependency	Description (Version)
`-none-`	

III.H.2. Installation instructions

[95] states only that Cg is an "optional" library. Installation instructions for Cg were unavailable from either [95] or [96]. During the initial attempt the author downloaded Cg (version 2.2: "Cg-2.2_April2009_x86.tgz") from NVIDIA ([96]) and extracted the file using the following command into the `ogre` directory:

```
sudo tar xzf ./Cg-2.2_April2009_x86.tgz
```

III.H.2.a. Problem: after extracting Cg into the ogre directory during the initial attempt, it was not available to `./configure`

Attempts to configure OGRE failed because Cg must be extracted into the root directory.

III.H.2.b. Resolution

The author deleted the directory containing the files extracted into the `ogre` directory.

III.H.2.c. Problem: extracting Cg into the root directory during the initial attempt preserved the existing user identification and group identification of all files in the archive

The author extracted the files using the following command into the root directory:

```
sudo tar xzf ./Cg-2.2_April2009_x86.tgz
```

However, the existing user identification (uid) of 2402 and group identification (gid) of 30 of all files in the archive were preserved. The author notes that this is the default behavior for the root user, unless the option `--no-same-owner` is used.

III.H.2.d. Resolution

The author identified affected files as follows:

```
ls -a | grep 2402
```

The author modified the attributes of the affected files as follows:

```
chown root <filename>
chgrp root <filename>
```

Due to the failure of the initial attempt, the author decided to limit the installation of "optional" libraries to the extent possible and did not install the NVIDIA Cg library ("Cg") during the development of this installation procedure, and configured OGRE with

the command:

```
./configure --with-platform=GLX --disable-cg
```

to force `./configure` to continue without Cg support.

However, after successfully building Gazebo, the author re-evaluated the installation of Cg based on the similarity of the alternate procedure to the installation procedure. Because the author was only able to successfully install Cg from source with difficulty (see paragraphs III.H.2.a. and III.H.2.c., above), the author confirmed that Cg was available for installation from the openSUSE repository using YaST, and installed packages `cg 2.2-1.1.1` and `cg-devel 2.2-1.1.1` using YaST.

The author then revised the "Install OGRE" step of the installation procedure to not disable Cg support using the `--disable-cg` flag and confirmed OGRE installation using Cg during the verification of the installation procedure.

III.H.3. Install Cg

1. Install `cg` and `cg-devel` from the openSUSE repository using YaST.

III.I. OGRE

Package (source package name)	Description (Version)
`OGRE` `(ogre-v1-6-4.tar.bz2)`	`Object-oriented Graphics Rendering Engine` `(1.6.4)`

III.I.1. Dependencies

Documented dependency	Description (Version)
automake	A program for generating makefiles (1.9.5 (1.6+ required))
autoconf	A tool for configuring source code (2.59a (2.50+ required))
make	The make command (3.80)
libtool	A tool to build shared libraries (1.5.6 (1.4+ required)
pkg-config	A library management system (0.17.2)
gcc	The system C compiler (3.3.5)
g++ (gcc-c++)	The system C++ compiler (3.3.5)
cpp	The system preprocessor (3.3.5)
FreeType2	Software font engine (2.1.x+)
zziplib	Zip compression library (0.13.x+)
FreeImage	Open source image library (-none-)

III.I.1.a. Install zziplib and zziplib-devel

To satisfy a documented dependency, the author installed package zziplb 0.13.56-2.1 and zziplib-devel 0.13.56-2.1 using YaST.

Undocumented dependency	Description (Version)
GLEW	OpenGL Extension Wrangler library (-none-)

III.I.1.b. Problem: GLEW is an undocumented dependency

The first attempt to compile OGRE failed due to a “No such file or directory”

error, a portion of which is reproduced below:

```
from OgreGLXGLSupport.cpp:35:
../../../../RenderSystems/GL/include/GL/glew.h:1128:20
: error: GL/glu.h: No such file or directory
```

The error suggested the problem was related to GLEW, which was not installed as part of the base installation. Because `./configure` did not warn or fail as a result, the author considers this an undocumented dependency.

III.I.1.c. Resolution: install `glew` and `glew-devel`

To satisfy an undocumented dependency, the author installed packages `glew 1.5.1-2.1` and `glew-devel 1.5.1-2.1` using YaST. Package `libGLEW1_5 1.5.1-2.1` was installed by YaST to resolve a dependency.

III.I.1.d. Problem: OIS, CEGUI, and OGRE are interdependent

The initial attempt was based on documented installation instructions and used documented dependencies to establish the installation procedure. OGRE is a documented dependency for Gazebo, however neither OIS nor CEGUI are documented dependencies of OGRE. During the initial attempt, configuration of OGRE failed because neither OIS nor CEGUI were installed.

III.I.1.e. Resolution: None.

The author revised the installation procedure to install OIS before CEGUI and CEGUI before OGRE, thus resolving the problem. Based on a review of online documentation, the author determined that the normal installation procedure for CEGUI and OGRE is: CEGUI, OGRE, then CEGUI again for the “OGRE CEGUI demos”. OIS

is required to be installed both before CEGUI and OGRE, so perhaps the normal installation procedure should be: OIS, CEGUI, OGRE, OIS (for the "CEGUIOgre Demo"), and finally CEGUI (for the "OGRE CEGUI demos").

The author had no interest in CEGUI except as a dependency for OGRE, or OGRE except as a dependency for Gazebo, and so did not attempt to determine the correct order of installation to build the "CEGUIOgre Demo" or "OGRE CEGUI demos" using OIS, CEGUI, and OGRE. See paragraphs III.D.1.a. and III.G.2.a.

III.I.2. Installation instructions

[95] provided documented installation instructions.

III.I.2.a. Problem: configuration failed because the NVIDIA Cg library was not installed. The NVIDIA Cg library is optional.

The first attempt to configure OGRE with the command:

```
./configure --with-platform=GLX
```

failed, resulting in an error which stated, in part:

```
You do not have the nVidia Cg libraries installed.
```

and continued:

```
You can disable the building of Cg support by
providing --disable-cg to this configure script but
this is highly discouraged as this breaks many of the
examples.
```

The author considers this a configuration error. `./configure` should not have failed because the "optional" NVIDIA Cg library ("Cg") was not installed. However, the author does not consider Cg an undocumented dependency because building Cg support

could be disabled.

III.I.2.b. Resolution: Install Cg using YaST

See paragraph III.H.

III.I.3. Install OGRE

1. Install `zziplib` and `zziplib-devel`.
2. Install `glew` and `glew-devel`.
3. Configure, compile, and install OGRE as follows:
   - `$ ./bootstrap`
   - `$ ./configure --with-platform=GLX --disable-ogre-demos`
   - `$ make`
   - `$ su`
   - `$ make install`
   - `$ exit`

III.J. FFmpeg

FFmpeg was neither a documented nor undocumented dependency of Gazebo, but was installed to provide access to libavcodec.

Package (source package name)	Description (Version)
`FFmpeg` `(ffmpeg-0.5.tar.bz2)`	`command line tool to convert multimedia files between formats (0.5)`

III.J.1. Dependencies

Documented dependency	Description (Version)
`make`	`The make command (3.81+)`

Undocumented dependency	Description (Version)
`-none-`	

III.J.2. Installation instructions

[97] provided documented installation instructions.

III.J.2.a. Problem: shared libraries were not enabled by default

The first attempt to configure, compile, and install FFmpeg was successful. However, a subsequent attempt to compile Gazebo resulted in failure due to numerous "undefined reference" errors to functions beginning with "av_". The author determined these functions were provided by libavcodec, which is provided by FFmpeg.

III.J.2.b. Resolution: enable shared libraries

The author uninstalled FFmpeg and configured the build to enable shared libraries:

```
./configure --enable-shared
```

The author then compiled and installed FFmpeg successfully. This resolved the "undefined reference" errors encountered when attempting to compile Gazebo.

III.J.3. Install FFmpeg

Configure, compile, and install FFmpeg as follows:

- `$ ./configure --enable-shared`
- `$ make`
- `$ su`
- `$ make install`
- `$ exit`

III.K. Player

Package (source package name)	Description (Version)
`Player` `(player-3.0.0.tar.gz)`	`the Player server` `(3.0.0)`

III.K.1. Dependencies

Documented dependency	Description (Version)
`-none-`	

Undocumented dependency	Description (Version)
`cmake` `(cmake-2.6.4-3.3)`	`Cross-platform, open source make system` `(2.6.4)`

III.K.1.a. Problem: `cmake` was an undocumented dependency

Online documentation for the Player Project was not up-to-date. The application cmake was an undocumented dependency. The installation procedure given by [98] (“Standard install procedure”) was:

- Download the latest Player source tarball (player-<version>.tgz) from Sourceforge.

- Uncompress and expand the tarball:

  ```
  $ tar xzvf player-<version>.tgz
  ```

- 'cd' into Player's source directory:

  ```
  $ cd player-<version>
  ```

- To configure Player with default settings:

  ```
  $ ./configure
  ```

- Compile Player:

  ```
  $ make
  ```

- Install Player. By default, Player will be installed in /usr/local so you need to become root for this step. Remember to return to your normal user ID afterwards.

  ```
  $ make install
  ```

The installation procedures given by [99] (“Installation”) and [100] (“Out-of-source Build”) were essentially the same:

`$ cd player` (this step is omitted by [99])

```
$ mkdir build
$ cd build
$ cmake ../
```

`$ make` (this step is omitted by [99])

```
$ make install
```

The installation procedure given by [98] was incorrect. Player 3.0.0 was released on September 7, 2009. An announcement made to the playerstage-users mailing list

stated: an “entirely new build system using CMake” was a feature of the new release. As a result, the author was aware the installation instructions for Player 3.0.0 had changed and concluded online documentation was not updated to document the use of `cmake` to configure Player 3.0.0.

Because online documentation was initially favored (see paragraph II.B.), the author considers `cmake` an undocumented dependency.

III.K.1.b. Resolution: install `cmake`

The author installed package `cmake 2.6.4-3.3` using YaST. During development of the installation procedure, the author also installed package `cmake-gui 2.6.4-3.3` using YaST to provide a more usable front-end for `cmake`. However, the author did not re-install `cmake-gui` during the verification of the installation procedure because it was less useful than anticipated.

III.K.2. Installation instructions

[98], [99], and [100] provided documented installation instructions. The installation procedure given by [98] was incorrect.

III.K.2.a. Problem: the environment variables `PYTHON_INCLUDE_PATH` and `PYTHON_LIBRARY` were set to `NOTFOUND`

Configuration of Player failed because the environment variables `PYTHON_INCLUDE_PATH` and `PYTHON_LIBRARY` were set to `NOTFOUND`. As noted in paragraph III.A. above, the author installed the “Base Development”and “C/C++ Development” package groups, but did not install the “Python Development” package group.

III.K.2.b. Resolution: Set `BUILD_PYTHONC_BINDINGS` to `OFF`

The author used `ccmake` to review the configuration options and set "`BUILD_PYTHONC_BINDINGS: Build the Python bindings for the C client library`" to `OFF`. The configuration option "`BUILD_PYTHONCPP_BINDINGS`" was then set to `OFF` by `ccmake` by default. The author considers this a configuration error. `./configure` should not have failed because an "optional" library was not installed. The author does not consider Python an undocumented dependency because building Python bindings for the C client library could be disabled.

III.K.3. Install Player

1. Install `cmake` (see paragraph III.K.1.b.).
2. Set `BUILD_PYTHONC_BINDINGS` to `OFF` (see paragraph III.K.2.b.).
3. Configure, compile, and install Player as follows:
   - `$ mkdir build`
   - `$ cd build`
   - `$ cmake ..`
   - `$ make`
   - `$ su`
   - `$ make install`
   - `$ exit`

III.L. Gazebo

[80] and [81] provide contradictory installation instructions. Neither is complete

or correct. The initial attempt to compile and install Gazebo failed with numerous errors. [80] states: “If things go wrong, please check the archives of the Gazebo mailing list.” The author found online documentation to be of limited utility. Review of available forums including the playerstage-gazebo mailing list reveals it is not uncommon for several people to ask the same question or describe the same problem, often with no recorded resolution. Several of the problems the author encountered during the initial attempt and later successful attempt were also identified by others before and after the development of this installation procedure, without resolution.

Package (source package name)	Description (Version)
`Gazebo` `(-none-)`	`Gazebo` `(0.9.0 rev. 8443)`

III.L.1. Problems encountered before installation

III.L.1.a. Problem: installation instructions included with online documentation are incorrect.

[80] states the following will extract Gazebo:

```
$ tar xvjf gazebo-<version>.tar.gz
```

However, Gazebo 0.9.0 is distributed as a “bz2” file (“gazebo-0.9.0.tar.bz2”). As a result, attempting to follow these instructions results in the following error:

```
gzip: stdin: not in gzip format
tar: Child returned status 1
tar: Error exit delayed from previous errors
```

III.L.1.b. Resolution: None.

All other releases of the Gazebo package hosted by [38] are ".tar.gz" files. The author concluded [80] was not updated to provide installation instructions for the current Gazebo 0.9.0 package. The author used Ark 2.10.999 to extract Gazebo.

III.L.1.c. Problem: installation instructions included with packaged documentation are incomplete.

[81] stated:

```
Installation
------------
Read the installation instructions in the online
manual for generic instructions.  For most people, the
following sequence will suffice:

$ mkdir build (inside the gazebo-trunk directory)
$ cd build
$ cmake ..
$ make

Uninstallation
--------------
Read the installation instructions in the online
manual for generic instructions.  For most people, the
following sequence will suffice:
```

```
$ make install (inside the gazebo-trunk/build
directory)
```

This appears to be an editorial error. To install Gazebo, it would be necessary for a user to read, then follow, the uninstallation instructions. No uninstallation instructions were provided.

III.L.1.d. Resolution: None.

The author concluded installation instructions provided by [81] were incomplete. Based on the installation instructions for Player, the author was familiar with the installation procedure using `cmake`.

III.L.1.e. Problem: installation instructions provided by online documentation do not match installation instructions provided by packaged documentation.

Installation instructions provided by [80] do not match the installation instructions provided by [81] (the Gazebo package contains no "INSTALL" file). [80] stated:

```
$ tar xvzf gazebo-<version>.tar.gz
$ cd gazebo-<version>
$ scons
Note that scons will fail if any of the required
packages are missing. Once Gazebo has been built, it
can be installed:
$ su
```

```
$ scons install
$ exit
```

Installation instructions provided by [81] are documented above.

III.L.1.f. Resolution: None.

Based on the result of trying to install Gazebo using `scons`, the author concluded [80] was incorrect. See paragraph III.L.3.a.

III.L.1.g. Problem: online documentation does not provide the latest installation instructions.

[81] also stated:

```
On-line installation instructions
---------------------------------
The latest installation instructions can be found on-
line, at
  http://playerstage.sourceforge.net/doc/
```

However, attempting to access this URL results in a “403 error”.

III.L.1.h. Resolution: None.

The author concluded the latest installation instructions are included with packaged documentation, not online documentation.

III.L.1.i. Problem: Online documentation directs users to the socalwifi-iptv mailing list archive in lieu of the playerstage-gazebo mailing list archive.

[80] stated: “If things go wrong, please check the archives of the Gazebo mailing

list. Please read the instructions below carefully before reporting posting [*sic*] to the mailing list." However, the hyperlink to the Gazebo mailing list ("http://sourceforge.net/mailarchive/forum.php?forum_id=33909") is a hyperlink to the mailing list archive for the socalwifi-iptv project on SourceForge.net. The playerstage-gazebo mailing list ("http://sourceforge.net/mailarchive/forum.php?forum_id=27052") is accessible from the Player Project "Support" page.

III.L.2. Dependencies

[85] provided a list of documented dependencies.

Documented dependency	Description (Version)
`scons`	`Replacement for make (0.97 or greater)`
`fltk`	`Cross-platform GUI toolkit (1.1.7 or greater)`
`OGRE`	`Object-oriented Graphics Rendering Engine (1.4.4)`
`ODE`	`Open source library for simulating rigid body physics (0.8)`
`OIS`	`Cross-platform object-oriented library for handling input devices (1.0)`
`libxml2`	`A library to manipulate XML files (2.6.29 or greater)`

III.L.2.a. Problem: a later version of OGRE was required to compile Gazebo than that documented by the installation instructions

The author downloaded and installed a version of OGRE other than version 1.4.4 which [85] states is a prerequisite. The author concluded OGRE version 1.6.3 or greater is required to compile Gazebo based on the following output of the `cmake ../`

command:

```
-- checking for module 'OGRE>=1.6.3'
--   found OGRE, version 1.6.4
```

III.L.2.b. Resolution: None

The author concluded [85] was incorrect.

III.L.2.c. Problem: `libxml2-devel` was not installed by default

Package `libxml2` was installed as part of the base installation. However, the development (header and library) files were not installed. Attempts to compile Gazebo resulted in the following error:

```
Error: libxml2 and development files not found.
```

III.L.2.d. Resolution: install `libxml2-devel`

The author installed package `libxml2-devel 2.7.3-2.2` using YaST. Package `readline-devel 6.0-18.3` was installed by YaST to resolve a dependency.

Undocumented dependency	Description (Version)
-none	

III.L.2.e. Problem: during the initial attempt the author concluded packages `freeglut` and `openal` were undocumented dependencies

Based on errors encountered during the initial attempt, the author concluded `freeglut` and `openal` were undocumented dependencies of Gazebo.

As a result, the author installed packages `freeglut`, `freeglut-devel`,

`openal-soft`, `openal-soft-devel`, and `libopenal1-soft`.

During development of the installation procedure, package `freeglut 090301-3.1` was installed as part of the base installation, and the author installed packages `freeglut-devel 090301-3.1`, `openal-soft 1.9.616-1.1.1`, `openal-soft-devel 1.9.616-1.1.1`, and `libopenal1-soft 1.9.616-1.1.1` using YaST.

III.L.2.f. Resolution: packages freeglut and openal are optional libraries, not undocumented dependencies

Based on the author's decisions to disposition errors on a case basis as either configuration errors or evidence of undocumented dependencies, and to limit the use of "optional" libraries to simplify the installation procedure to the extent possible, the author re-evaluated the installation of these packages, determined they were optional libraries, not undocumented dependencies, and did not install them during verification of the installation procedure.

III.L.2.g. Problem: `boost-devel` is an undocumented dependency.

During the initial attempt, attempts to compile Gazebo failed because package `boost-devel` was not installed.

III.L.2.h. Resolution: None.

During the initial attempt, the author installed package `boost-devel 1.36.0-9.5` using YaST. This problem did not recur during the development of the installation procedure because `boost-devel 1.39.0-3.4.1` is part of the openSUSE 11.2 "C/C++ Development" package group.

III.L.3. Installation instructions

[80] and [81] provided documented installation instructions.

III.L.3.a. Problem: `scons` is no longer used to configure or compile Gazebo

As noted in Chapter II., the author first attempted to follow documented installation instructions. [80] states `scons` is used to configure, make, and install Gazebo. The author installed package `scons 1.2.0-2.2` using YaST, then attempted to configure and compile Gazebo using `scons`, resulting in the following error:

```
scons: *** No SConstruct file found.
```

The SConstruct file is required.

III.L.3.b. Resolution: None.

[81] provided alternate installation instructions. The author concluded [80] was incorrect.

III.L.3.c. Problem: an attempt to compile Gazebo resulted in a "cannot convert" error

After successfully configuring the build of Gazebo, the author attempted to compile Gazebo, resulting in the following error:

```
gazebo/server/controllers/audio/Audio.cc: In member
function 'void
gazebo::AudioController::PutAudioData()':
gazebo/server/controllers/audio/Audio.cc:160: error:
cannot convert 'gazebo::Time' to 'double' in
assignment
```

III.L.3.d. Resolution: revise file `Audio.cc` to eliminate the source of the error

The author revised line 160 of file `Audio.cc`:

```
this->audioIface->data->head.time =
Simulator::Instance()->GetSimTime();
```

as follows:

```
this->audioIface->data->head.time =
Simulator::Instance()->GetSimTime().Double();
```

The author submitted bug report number 2909192 on December 5, 2009 to report this problem to the Player Project.

III.L.4. Install Gazebo

1. Install `libxml2-devel`.
2. Revise line 160 of file `Audio.cc`:

   ```
   this->audioIface->data->head.time =
   Simulator::Instance()->GetSimTime();
   ```

   as follows:

   ```
   this->audioIface->data->head.time =
   Simulator::Instance()->GetSimTime().Double();
   ```

3. Configure, compile, and install Gazebo as follows:
   - `$ mkdir build`
   - `$ cd build`
   - `$ cmake ..`

- `$ make`
- `$ su`
- `$ make install`
- `$ exit`