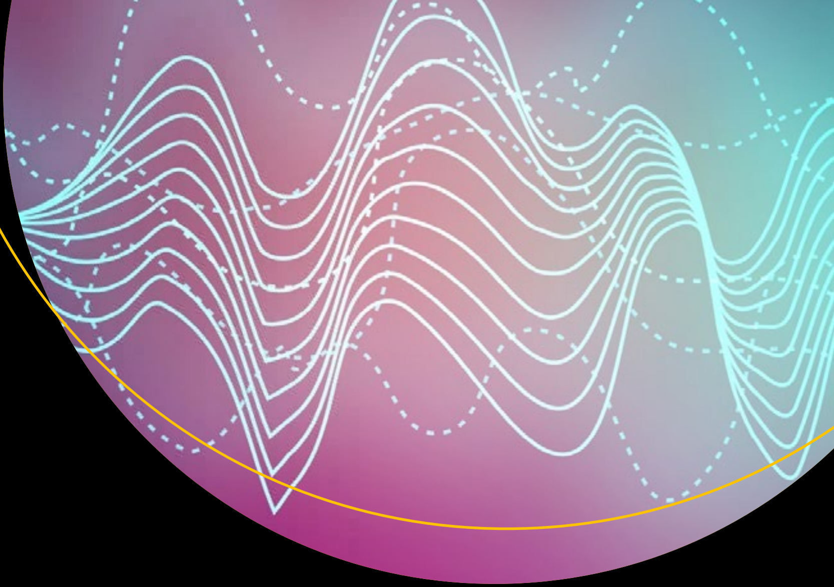


Audio Visualization Using ThMAD

Realtime Graphics Rendering for Ubuntu Linux

Peter Späth

Apress®

Audio Visualization Using ThMAD

Realtime Graphics Rendering for Ubuntu Linux

Peter Späth

Apress®

Audio Visualization Using ThMAD: Realtime Graphics Rendering for Ubuntu Linux

Peter Späth
Leipzig, Germany

ISBN-13 (pbk): 978-1-4842-3167-8 ISBN-13 (electronic): 978-1-4842-3168-5
https://doi.org/10.1007/978-1-4842-3168-5

Library of Congress Control Number: 2017960214

Cover image designed by Freepik

Managing Director: Welmoed Spahr
Editorial Director: Todd Green
Acquisitions Editor: Natalie Pao
Development Editor: James Markham
Technical Reviewer: Massimo Nardone
Coordinating Editor: Jessica Vakili
Copy Editor: Kezia Endsley
Compositor: SPi Global
Indexer: SPi Global
Artist: SPi Global

Contents

About the Author

Dr. Peter Späth has worked as an IT consultant with heavy focus on Java related development for over 15 years. Recently, Peter has decided to focus on his work as an author and working in a self-paced manner on software.

About the Technical Reviewer

Massimo Nardone has more than 22 years of experience in security, web/mobile development, cloud and IT architecture. His true IT passions are security and Android.

He has been programming and teaching people how to program with Android, Perl, PHP, Java, VB, Python, C/C++, and MySQL for more than 20 years.

He holds a Master's of Science degree in computing science from the University of Salerno, Italy.

He has worked as a project manager, software engineer, research engineer, chief security architect, information security manager, PCI/SCADA auditor, and senior lead IT security/cloud/SCADA architect for many years.

His technical skills include security, Android, Cloud, Java, MySQL, Drupal, Cobol, Perl, Web and Mobile development, MongoDB, D3, Joomla, Couchbase, C/C++, WebGL, Python, Pro Rails, Django CMS, Jekyll, Scratch, and more.

He currently works as the Chief Information Security Office (CISO) for Cargotec Oyj.

He worked as a visiting lecturer and supervisor for exercises at the Networking Laboratory of the Helsinki University of Technology (Aalto University). He holds four international patents (in the PKI, SIP, SAML, and Proxy areas).

Massimo has reviewed more than 40 IT books for different publishing companies and he is the coauthor of *Pro Android Games* (Apress, 2015).

This book is dedicated to Antti Jalonen and his family, who are always there when he needs them.

Introduction

Sound visualization is about capturing the sound coming from outside, using a microphone or the line-in jack of your sound card, or coming from inside if a sound file is played on your computer. The software suite ThMAD allows for graphically building a sound pipeline, yielding real-time graphics to produce beautiful and interesting output from that sound. The introductory chapter depicts such a system, describes the targeted audience, and gives you some hints about text conventions and the preferred way of reading this book.

About Sound Visualization and ThMAD

You can buy a CD or listen to a radio station, or listen to music provided by a stream or other means. If you like the music it will give you an emotional and/or intellectual feeling, similar to how a good movie makes you feel. Concerning movies, there is an obvious marriage of musical sound and vision—movies have sound tracks and the combined pleasure is higher the more the sound track matches the story on-screen. Making a movie usually means you first have the scenes and speech assembled, and then you add the sound track. This is of course a high art if you want to make a good movie and a good combination of the two senses—sight and sound. If it is really good, one will amplify the other in either way.

As for audio visualization, you have a combination of visual and aural input to your senses. But it happens to be done differently compared to movies. The order is different and the coupling happens in an apparently more automated way. For audio visualization, the sound comes first and, based on it, the visual part of the joint artwork is generated by some program. As for the second assumption, I intentionally say apparently automated, because once you have a rendering program, you can start the program and enjoy the visualization of your music while you sit on a chair and do nothing. There is a third difference: the soundtrack of a movie is based on the plot, and it is by no means possible to change the plot but use the same soundtrack. For audio visualization the sound input can be different for the same visualization and you still get an interesting net result.

In order to get to a nice visualization and to have it become really good, you have to construct a rendering pipeline in some computer language, which could be an art itself. This has to be done manually. The good thing about it is that you don't need to be a master artist, nor a master software developer, from the beginning, since getting into it is not overly complicated and you'll be able to get your feet wet using the right program.

The Thinking Machine Audio Dreams (ThMAD, pronounced Thee-Mad) software suite is such a program. Once you learn the basics, you can start with some easy working rendering setups, and if you have enough time to learn, experiment, and build up sophisticated rendering pipelines, you will end up with magnificent audio visualizations.

ThMAD basically consists of a program named Artiste for generating a rendering pipeline, and a program named Player that you can use to run the pipeline in a visualization performance. While working inside Artiste, you see the realtime result of what you are doing, promptly and appropriately reacting to incoming sound according to changes you made, which gives a tremendous boost to your productivity. ThMAD and the surrounding system are depicted in Figure 1.

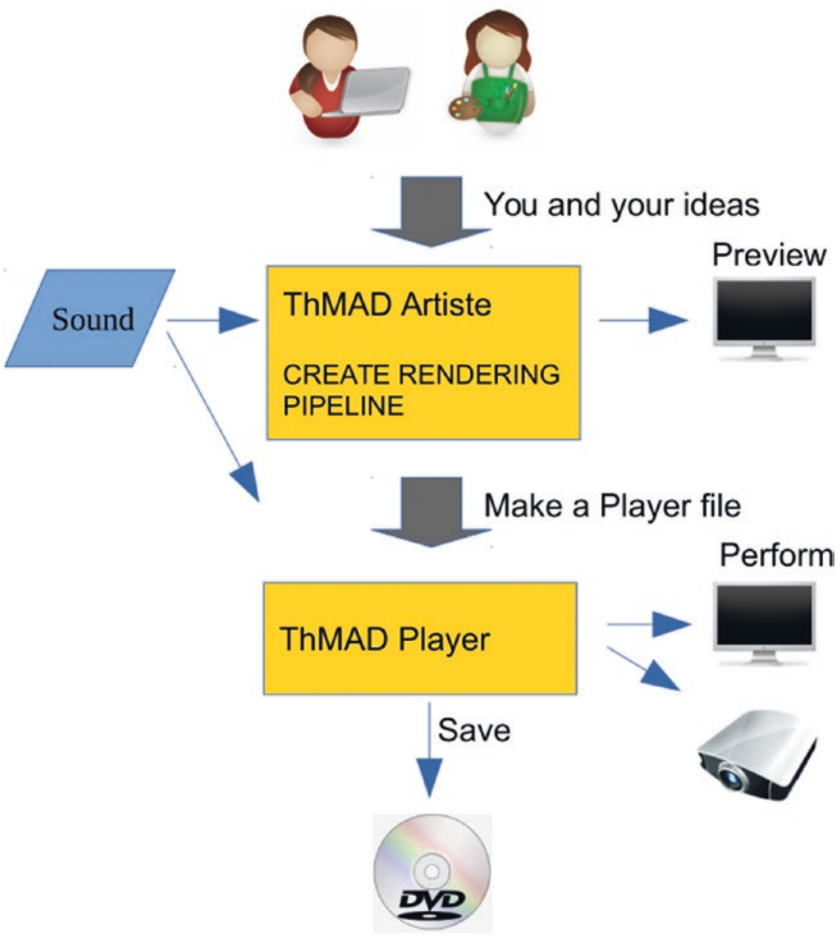


***Figure 1.** ThMAD and the surrounding system*

The ThMAD software is based on the open source GNU GPL licensed (`https://www.gnu.org/licenses/gpl-3.0.txt`) software VSXu from Vovoid Media Technologies®. The fork has been taken from version 0.5.0. A considerable amount of work has been done on it since then, but some code has been introduced to allow for VSXu states to be loaded and interpreted correctly.

ThMAD is GNU GPL licensed, so the source is open to everyone. You cannot sell it or use it for a product you sell according to the GNU GPL. But you can use it for private or professional pleasure or work on it and give it or your own work away (only!) as a GNU GPL licensed software itself. You can find the source here: `https://sourceforge.net/projects/thmad/`.

■ **Note** VSXu is a registered or unregistered trademark of Vovoid Media Technologies AB, Sweden. Dr. Peter Späth is independent of Vovoid Media.

ThMAD is for Linux, more precisely Ubuntu. You can however run Ubuntu inside a virtual machine under Windows or whatever and use it there for development and experiments. But don't expect such a setup to give you high performance. For serious applications, it is strongly suggested you get an Ubuntu box. It is not hard nowadays to buy a PC or laptop with preinstalled Ubuntu, or install Ubuntu yourself. The latter might be tricky sometimes and under certain circumstances, you should expect to read a lot of documentation and blogs before you get it running smoothly. Of course, the Ubuntu homepage will help, and you can start with a trial version running from a bootable CD. Linux distributions other than Ubuntu will work as well, but you have to compile them yourself from the source. As far as this book is concerned, ThMAD was tested only with Ubuntu.

Under the hood the software OpenGL is used, and it's a widely adopted industry standard for sophisticated graphics rendering, including manufacturing and computer games. You will be able to use a wide range of graphics cards to see ThMAD running. Be cautious with cheap hardware though, especially for older graphics cards and older or overly downsized systems. You cannot expect each and every rendering pipeline to run smoothly, and some of the more advanced ones may even fail. Do yourself a favor and spend some extra money on at least a medium-grade computer and decent graphics hardware.

This book is based on ThMAD version 1.0.0. The associated OpenGL version as of the time of writing is 3.0.

The Book's Targeted Audience

This book is for artists with some IT background, or developers with artistic inclinations. Development experience is not required, but is surely helpful for some advanced features like shader programming. It is however not necessary to use ThMAD at all, and this book helps you with samples you can use as a working basis if the shader source code is involved.

This book is not a development guide, so hints about the actual implementation are given rather infrequently and only as anchor points for people who might feel an inclination to get into the development.

In order to fully understand ThMAD, and OpenGL based software in general, you need to know basics about 2D and 3D graphics and coordinate systems. However, most geometric concepts are explained to some fair extent.

Installation

The installation process of ThMAD is discussed in Chapter 2, “Visualization Basics”.

Conventions Used in This Text

Working with ThMAD involves using its modules, which are organized in a tree-like structure.

Modules are usually named maths → converters → 4float_to_float4. Or in short with fixed width font, `4float_to_float4` if the module position inside the module tree is clear from the context.

State is the common notion for a rendering pipeline while constructing it. Finished states are also called *visuals*. References to sample states, including associated code provided with the installation, as well as informational hints in general, are highlighted like so:

■ **Note** This sample is as a source available under `A-3.2.1_Visualization_basics_basic_samples_basic_2d_sample` inside the `TheArtOfAudioVisualization` folder.

With the “Note” label replaced with “Tip” for informational hints. By folder in this context I mean a folder as showing up in the module lister or browser. Important notes and pitfalls are marked as follows:

■ **Caution** Due to the backfeeding it might easily happen… …

… … …

Code and script snippets, as well as terminal input and output, usually show up in monospace font like here:

```
apt-get install libc6 libfreetype6 libgcc1 libpulse0 libstdc++6 libglfw3
```

Very small code snippets appear directly inside the text. If a longer line does not fit into a line, a trailing ┐ at each line of code signifies that while actually writing it, the ┐ must be removed and the subsequent line break must be discarded. For example,

```
echo "cmd [...] rectangle ┐
abc [...]"
```

should be entered as

```
echo "cmd [...] rectangle abc [...]"
```

At many places, an asterisk * is used as a wildcard to denote any string. This frequently happens to refer to all the files inside a folder, or to file name patterns. Upon first startup, ThMAD Artiste creates a data folder for all your states and visualizations at

```
/home/[USER]/.local/share/thmad
```

and a symbolic link at

```
/home/[USER]/thmad
```

This points to the aforementioned If you are referring to the data folder inside this book, the link location is used.

How to Read This Book

This book can be read sequentially. Chapters 1 to 4 serve as an introduction, with Chapter 2, "Visualization Basics," perhaps being the most important, since it describes the basic system setup and two important simple visualization examples.

Chapters 5, "Stories—Basic Level" and 6, "Stories—Advanced Level" contain a collection of independent stories or tutorials that you can work through in any order.

Chapters 7 and 8 are references that you can consult whenever you have doubts or questions while working through the stories, and of course you can use them to deepen your knowledge about modules and to get ideas for your own visualizations.

CHAPTER 1

Sound Input

In this chapter, we deepen the knowledge of how the computer can be prepared to capture incoming sound or produced sound, how the sound is represented internally, and how the data arrives at ThMAD. We distinguish between the obvious air pressure elongation versus time representation, and the power versus frequency representation, or *spectrum*.

Preparing for Sound Input

After you purchase a PC or laptop with Ubuntu Linux, or after you have Ubuntu Linux installed on your PC, you have basically two options—you can use external audio sources or you can let the sound play from the computer. As for the latter, you could use the CD player of your PC, you could play some files from a USB stick or your hard disk, or you could stream audio files via the Internet or some other means.

Note that other Linux distributions might work as well. Give it a try—chances are good that you'll find similar programs, tools, and settings to accomplish the same thing.

If you want to use external audio sources, you need a microphone or a sound card with a line-in to connect to. Especially for laptops, the built-in microphones are not of the highest quality, but they might be enough for your purposes. You actually don't want to accurately reproduce the sound, but react to it, and for this aim, having perfectly linear input curves is not too important. On the other hand, if you don't want to lose important impulses from the basses, which can happen with cheap microphones, getting yourself a decent microphone might help you avoid surprises. Also, bear in mind that audio visualizations might be brittle to the structure of the incoming sound under certain circumstances.

Usually you want to avoid that and the overall outcome should be interesting for any kind of music input. This is easy enough to check with different recordings. But if, for example, the basses never make their way through the audio hardware to a suitable extent, because your microphone misses the basses, your rendering pipeline might lack reactiveness to an important part of the incoming sound. If instead for external sound input you just connect some audio source to the line-in jack of your computer, you are automatically on the safe side.

If you want to play CDs using your computer's CD player, or play audio files or streamed audio contents, e.g., using your browser, chances are good you don't have to do anything but start suitable programs or let the operating system do it for you automatically. For larger sound file collections, a program for administering them might be handy. RhythmBox, which is preinstalled on Ubuntu, is a good option.

The current version 1.0.0 of ThMAD primarily depends on PulseAudio, which is an audio routing server that handles all sound streams inside your computer. It knows everything that's captured or recorded, and everything that's played. Ubuntu Linux comes with PulseAudio preinstalled and automatically started; for other Linux distributions you may have to install it first.

ThMAD can also connect to ALSA, which is a low-level technology that talks to the sound hardware, and it can connect to JACK, which is a sound server that music professionals usually prefer. It is, however, considerably more difficult to use those options compared to PulseAudio, so we will as a sort of standard case use PulseAudio in the text.

For a graphical description of the standard PulseAudio sound chain, see Figure 1-1.

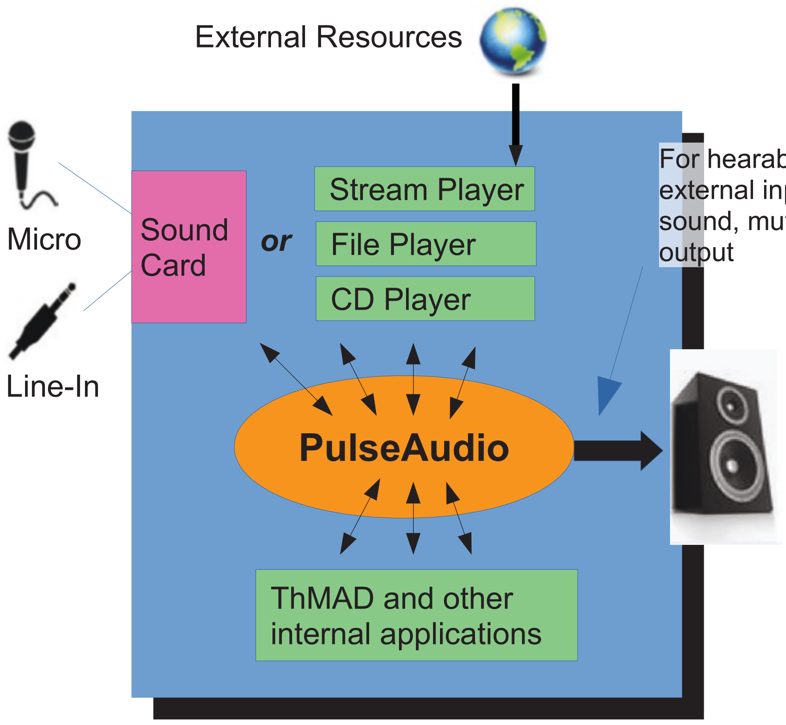


Figure 1-1. *The PulseAudio sound server inside Ubuntu*

Understanding Sound Structure

Sound is about air pressure oscillations that are received by your ears. From a mathematical or physical point of view, there are different representations for sound—the time-elongation (or time-pressure) representation and the frequency-power representation. Both of these are discussed in the following sections.

Time-Elongation Representation

On a diagram with the x-axis denoting the time and the y-axis denoting the pressure, the time-elongation representation might, for a sine wave, look like Figure 1-2.

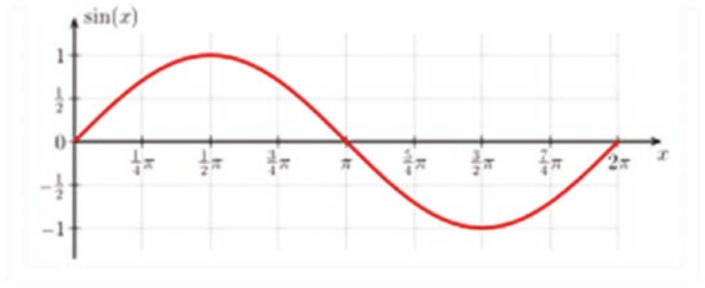


Figure 1-2. *A sine wave*

In computer systems, we need a digital representation for this sine wave. The idea is as follows: we divide the time into small time steps, say 44,100 steps per second, and for each time step, we write down the current air pressure, or y-value, and save it inside an array. This is sometimes called *analog-to-digital conversion* (ADC). Note that 44,100 is a widely adopted industry standard, for example, it's used with music CDs. Because we have two ears and like stereo, we do that conversion twice, for the left ear and the right ear. By that means, we end up with 88,200 numbers, which digitally represent one second of stereo sound. For the pressure or y-value representation, we use integer values (-32,768 up to 32,767), with the lower value representing a negative pressure offset, so maybe 0.997 bars, and the higher value representing a positive pressure offset, say 1.003 bars.

According to a scaling we can freely define, these could be mapped to 0.997 bars → y=-1000 and 1.003 bars → y=+1000. All the other numbers are mean pressures between these values. Of course we could use number ranges other than -32,768 ... 32,767 , but the range we chose here is internally represented by exactly two bytes of data, and computers like that very much. It is also a trade-off: fewer different values means less resolution and poorer quality, and more different values means higher storage need.

Of course, in reality music is stored in lots of different formats, including MP3, Ogg Vorbis and others, mainly for reasons of saving space. The 88,200 numbers per second add up quite rapidly. But in case of letting an application like ThMAD access PulseAudio data, it will receive the data in an uncompressed and untransformed, raw format. This is nice, since then ThMAD doesn't need extra logic to handle different sound formats.

Frequency-Power Representation

A practically less obvious representation of sound consists of writing down the frequency distribution at each instant of time.

Consider the time range [10s;10.1s] when listening to some music. Instead of reporting the air pressure amplitudes at each instant, e.g., 10.000s, 10.001s, 10.002s, we report the frequency mixture of the tones that arrive in our ears during some time range [10s;10.1s]. A · 100Hz + b · 200Hz + c · 300Hz + ..., where x Hz means a sine oscillation frequency of x per second, and the a, b, c,... are weights or power coefficients. The lower the number, the smaller the contribution and the higher, the larger the contribution.

Doing this in a mathematically concise way is called *Fourier Transformation*, and it turns out that it is a perfectly equivalent way of describing sound. In fact, if we have sound in a pressure versus time representation, we can apply a Fourier Transformation to transfer this into a power versus frequency representation without losing any information. That means the process is reversible and we have something like an *Inverse Fourier Transformation* to go back the other way. We don't show the mathematical details here; you can find a lot about that in other books and on the Internet.

Why we are mentioning Fourier Transformation here is that it turns out to be important for our audio visualization aim. Because instead of letting our visualization react on the elongation, which we *could* do but which bears the danger of losing things since changes happen so fast, we could also react to the powers of frequencies. Just think of reacting to bass beats in one way and reacting to the treble melody in a totally different way. Because beats happen at a much less frequent rate, maybe twice a second or something like that, the influence of the sound on the visualization is much more perceptible compared to the fast sound pressure oscillations.

To use such a representation inside ThMAD, the input from PulseAudio undergoes a Fourier Transformation and the frequency related powers we get from that can serve as an input to the visualization setup. We'll show you the details about that later.

Input Data Taken from the Sound Card

In any case, what the application will first receive from the sound driver, which is the software counterpart of the sound card, is an array of size *N* of elongation data. For example, if at an instant t0, we request 512x2 samples from the sound driver at 44,100 Hz, with the "2" multiplier because we want to capture two-channel stereo data, this means we have 512/44,100 = 0.0116 seconds of data from t0 -0.0116 to t0 . This implies that, if we don't want to lose any data, we need at least 0.0116 seconds before we ask for the next chunk of data from the sound driver, and so on.

While for the first sound representation, pressure versus time, we are done by just providing the input array acquired from the sound driver to the visualization pipeline, for the power versus frequency representation we pass this array through the Fourier Transformation. We get an array with powers for the frequencies $1 \cdot f_0$, $2 \cdot f_0$, $3 \cdot f_0$, ... , where for the case of a 44,100 Hz sampling frequency and 512 · 2 acquired samples, f0 calculates to f0 = 44,100 / 512 = 86.13 Hz. For some Fourier Transformation algorithm intrinsics, the power for the highest frequency will not be for 44,100 Hz, but only half of it—22,100 Hz. So, for an input of size 512, the Fourier Transformation will yield 256 power values.

Summary

You learned how sound data arrives at your system and how that sound is represented internally. You learned that ThMAD primarily depends on Ubuntu's internally running standard sound server, called *PulseAudio*.

In the next chapter, you will investigate the toolchain necessary to construct audio visualizations. You will also learn how to install ThMAD on your system and what needs to be configured to have everything running smoothly. The chapter starts with two basic examples to be run inside ThMAD.

CHAPTER 2

Visualization Basics

This chapter is about the toolchain used for an audio visualization project and presents two basic audio visualization samples using ThMAD.

If you want to become an artist, one of the first things you'll have to do is learn how to use the tools to accomplish an oeuvre. You don't start with the most complicated setups though, apart maybe from whetting your appetite, you begin with simple things. Then you improve your proficiency step by step, maybe learn about a more and more complicated example of other artists' work, and in the end hopefully you can accomplish your own ideas no matter how complicated they are. Well, some people prefer to mix the stages and learn from the inside with more complicated setups. It is up to you. But learning the tools is inevitable in any case.

Toolchain

The toolchain tells us which program is needed for the complete work and presentation setup, starting from input, which is sound in this case, to output, which is graphics. The latter might show up in the monitor, or in the beamer, or inside a video file if sound and graphics are merged.

Actually, you don't need too much. Ubuntu Linux provides you with quite a lot. To play something or capture something from the outside, everything is already there. And PulseAudio, the sound server running inside Ubuntu, already knows about it and provides the sound data to programs that need it, like ThMAD.

What is left is a connection from PulseAudio to ThMAD, ThMAD itself, which is providing the output to a monitor or beamer, and a program to record and produce a video if you like.

Installing ThMAD

After you downloaded ThMAD from `https://sourceforge.net/projects/thmad/` as a Debian package that Ubuntu understands (Debian is the mother distribution of Ubuntu) with suffix `.deb`, you have to make sure the dependencies are fulfilled.

© Peter Späth 2017
P. Späth, *Audio Visualization Using ThMAD*, https://doi.org/10.1007/978-1-4842-3168-5_2

A future version might do this automatically, but for now you do it manually. ThMAD depends on the following packages, where entries marked with a ▪ are most probably already installed on your Ubuntu Linux system from the beginning:

- libglfw3 (≥ 3.1)
- ▪libc6 (≥ 2.17)
- ▪libfreetype6 (≥ 2.2.1)
- ▪libgcc1 (≥ 1:4.1.1)
- ▪libgl1-mesa-glx (≥ 11.2.0) (or libgl1)
- ▪libglew1.13 (≥ 1.13.0)
- ▪libglu1-mesa (≥ 9.0.0) (or libglu1)
- ▪libjpeg8 (≥ 8c)
- ▪libpng12-0 (≥ 1.2.13-4)
- ▪libpulse0 (≥ 0.99.1)
- ▪libstdc++6 (≥ 5.2)

To install these packages, log in as root inside a terminal. Press Ctrl+Alt+T and then enter `sudo su` at the terminal. You'll be asked to enter your password. Do so then enter the following:

```
apt-get install libc6 \
libfreetype6 libgcc1 \
libgl1-mesa-glx libglew1.13 \
ibglu1-mesa libjpeg8 libpng12-0 \
libpulse0 libstdc++6 libglfw3
```

where the \ means the following newline (you press Enter or Return) gets ignored. If you want to enter the command in a single line, just ignore the backslashes.

Don't worry if the output says you already have `...` installed; it will not hurt if you try to install packages again, instead the command will simply ignore packages you already have on your system.

To install ThMAD itself, say you have downloaded it via browser and it ended up in folder `Downloads` in your home directory, you will install it, still as root, via

```
dpkg -i /home/[USER]/Downloads/thmad_1.0.0_amd64.deb
```

or any other version you get. `[USER]` should be replaced with your Linux username. All files will end up in `/opt/thmad`. After that, log off as root by pressing Ctrl+D. This is important for subsequent actions to not mess up your system.

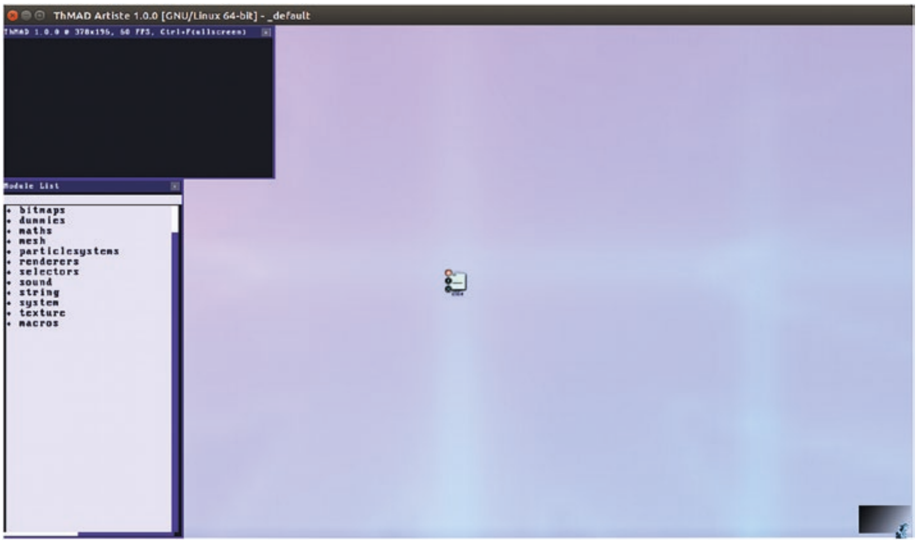
For your convenience, launchers are available; you can place them via these commands on your desktop:

```
cp /opt/thmad/share/applications/thmad-artiste*.desktop\
~/Desktop/
cp /opt/thmad/share/applications/thmad-player*.desktop\
~/Desktop/
```

To see whether everything works, use the launcher for Artiste, or on the terminal, enter this:

```
/opt/thmad/thmad_artiste
```

A window should show up, as shown in Figure 2-1. Congratulations! ThMAD is now running on your system.

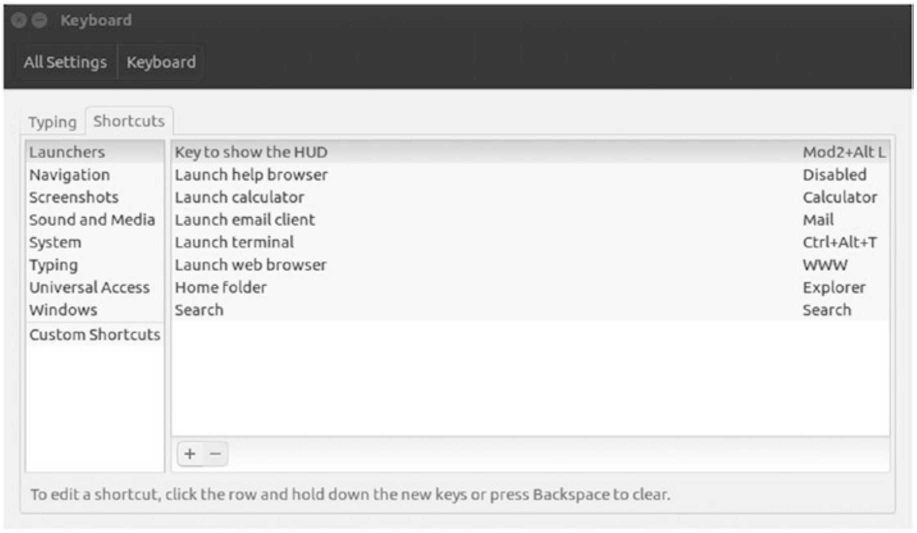


ThMAD Artiste 1.0.0 [GNU/Linux 64-bit] - _default
THMAD 1.0.0 • 378x195, 60 FPS, Ctrl-F(ullscreen)
Module List
bitmaps
dummies
maths
mesh
particlesystems
renderers
selectors
sound
string
system
texture
macros

Figure 2-1. *The ThMAD Artiste startup window*

While the main installation folder can be renamed, the launchers depend on the installation residing in `/opt/thmad`. You could however edit the launchers appropriately, if you think a different installation folder is a better option for you.

As a last preparation step before actually using ThMAD Artiste, you might consider releasing the Alt key from the operating system. The default Ubuntu window manager, Unity, uses the Alt key to start the *Heads Up Display*, HUD, but ThMAD uses it as well for various GUI actions. To disable Ubuntu using the Alt key for HUD, or to change the key binding, go to the Keyboard section of the preferences, go to the Shortcuts tab, then to the Launchers menu. Select the Key to Show the HUD entry and press Backspace to disable it, or choose a new key or key combination to change the binding. See Figure 2-2.



Launchers	
Navigation	
Screenshots	
Sound and Media	
System	
Typing	
Universal Access	
Windows	
Custom Shortcuts	

Key to show the HUD	Mod2+Alt L
Launch help browser	Disabled
Launch calculator	Calculator
Launch email client	Mail
Launch terminal	Ctrl+Alt+T
Launch web browser	WWW
Home folder	Explorer
Search	Search

To edit a shortcut, click the row and hold down the new keys or press Backspace to clear.

Figure 2-2. *Releasing the Alt key in Ubuntu Unity*

After clicking on Key to Show the HUD and pressing Backspace, it will be disabled. Or you can enter a different key or key combination to change the binding.

As an internal detail, for those who are interested in it, ThMAD version 1.x.x depends on OpenGL version 3.x.x. To find out which OpenGL version your system has, you can install the package `mesa-utils`:

```
sudo apt-get install mesa-utils
```

and then enter:

```
glxinfo | grep 'version'
```

This will give you something like

```
server glx version string: 1.4
client glx version string: 1.4
GLX version: 1.4
    Max core profile version: 3.3
    Max compat profile version: 3.0
    Max GLES1 profile version: 1.1
    Max GLES[23] profile version: 3.0
```

```
OpenGL core profile version string: 3.3 (Core Profile) Mesa 12.0.6

OpenGL core profile shading language version string: 3.30

OpenGL version string: 3.0 Mesa 12.0.6

OpenGL shading language version string: 1.30

OpenGL ES profile version string: OpenGL ES 3.0 Mesa 12.0.6

OpenGL ES profile shading language version string: OpenGL ES GLSL ES 3.00
```

The `OpenGL core profile version string` line points you to the OpenGL version in use.

Connect PulseAudio Sound to ThMAD

In order to connect ThMAD to Ubuntu's sound server PulseAudio, a couple of things are worth mentioning.

To see how PulseAudio connects things, the application `pavucontrol` comes in handy. It needs to be installed and, in order to do that from a terminal, open a terminal window via Ctrl+Alt+T and enter

```
sudo apt install pavucontrol
```

You will be prompted for your password. You can instead use the installer launcher and enter `pavucontrol`. After the installation, you can run it from the starter. Press the Windows key, then enter `pavucontrol` and click on the launcher icon.

PulseAudio internally handles the following objects, reflected inside the `pavucontrol` program:

- *Playback*. Applications sending sound data toward PulseAudio are playback objects. Without running a music-playing application, you still will have one entry, called System Sounds, which belongs to the operating system. With a music-player at work, say RhythmBox (it is installed with a standard Ubuntu installation), it will look like Figure 2-3.

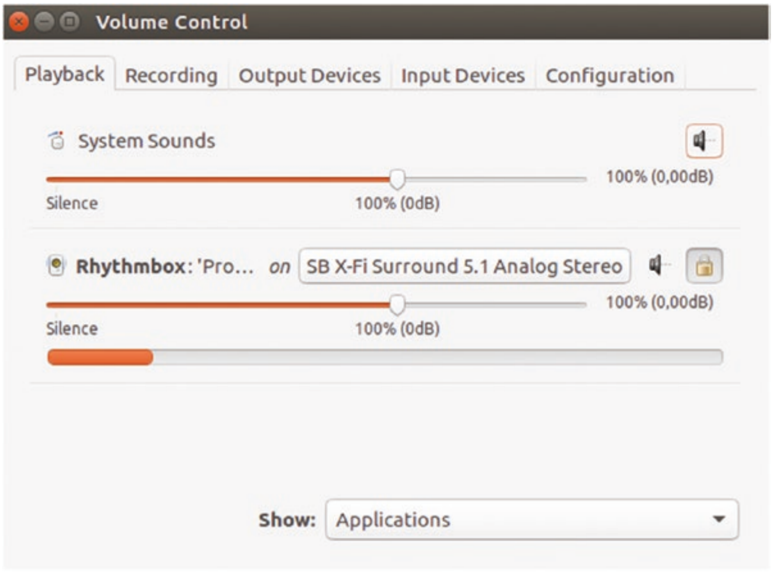


Figure 2-3. *PulseAudio playback objects*

You can see RhythmBox is currently sending something to my sound card called SB X-Fi by looking at the orange meter bar below it.

- *Recording*. Applications reading sound data will be listed here. ThMAD will later show up here.
- *Output Devices*. The sound cards installed on your system show up here, more precisely their playback channels. For me it lists, among others, SB X-Fi Surround 5.1 Analog Stereo.
- *Input Devices*. The sound cards installed on your system shows up here, more precisely their recording channels.

Input channels are microphones, but also *monitors*, and they plug to the soundcard's output, say speakers or earphones. For me it lists, among others, Monitor of SB X-Fi Surround 5.1 Analog Stereo. See Figure 2-4. This is important, because that is where ThMAD connects.

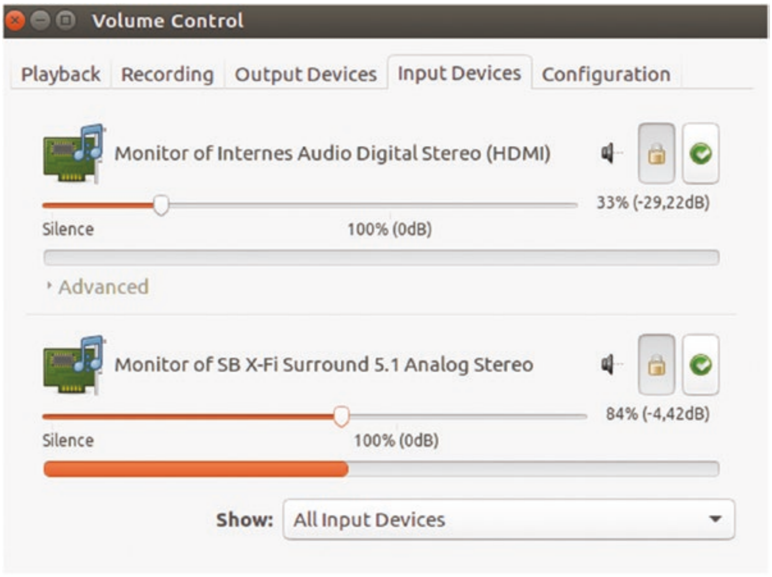


Figure 2-4. *PulseAudio input device objects*

You can see in Figure 2-4 that PulseAudio is currently providing sound data to the monitor Monitor of SB X-Fi by looking at the orange meter bar below it.

To actually do the connection, start ThMAD Artiste and drag the menu item `input_visualization_listener` from the Sound section from inside the left modules menu to the canvas. To make it appear, click on Sound inside the left menu to open the submenu. The module `input_visualization_listener` will then appear and you can drag it to the canvas. See Figure 2-5. You can use the mouse wheel to zoom the canvas in or out in order to make the module symbols appear bigger.

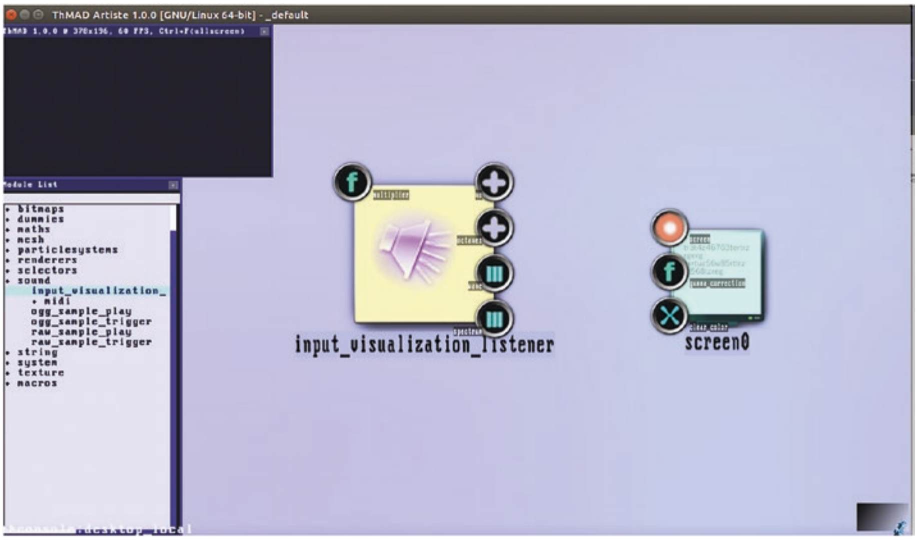


Figure 2-5. *Open the ThMAD sound module*

Then open `pavucontrol` if it's not already opened. Go to the *Recording* section and connect `thmad` to a monitor of your audible sound card output channel, as shown in Figure 2-6.

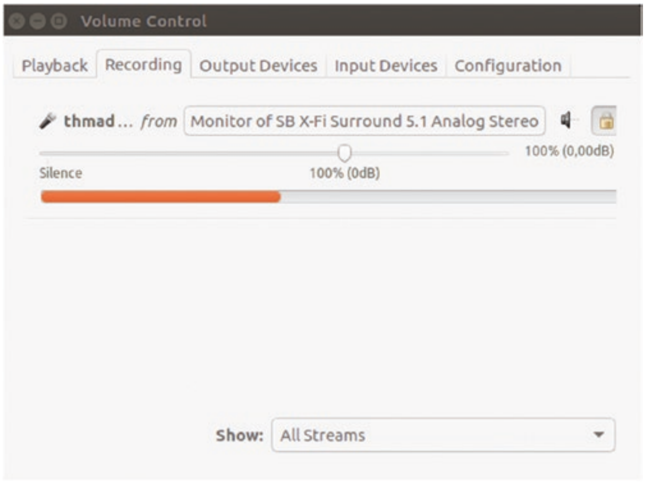


Figure 2-6. *Connecting ThMAD to a PulseAudio monitor*

You can see its monitor called Monitor of SB X-Fi sends some sound data to ThMAD by looking at the orange meter bar below it.