System Overview

The AMT(M) application is a monophonic audio to MIDI translator designed for use with an electric guitar or violin. It analyzes a single audio input and translates the audio stream into MIDI notes.

If your instrument has a divided pickup and you wish to have polyphonic translation to MIDI, then please refer to the AMT(P) application and the associated audio breakout boards. That system is available from my site CoolLutherie.com for you to download and build.

This application, AMT(M), is designed to be simple, yet functional; a stepping stone toward MIDI integration for a stringed instrument. As a monophonic processor, it serves well for lead presets with most software synthesizers. The output is MIDI version 1 which is supported by any MIDI-compatible software.

For visual artists, AMT(M) integrates with TouchDesigner by sending the MIDI message information over the network as JSON through UDP. A fully-functional TouchDesigner example is available to download from CoolLutherie.com. TouchDesigner is a real time graphics design environment and display engine that is available to download from derivative.ca.

Learn how to make or modify this program

I provide the AMT(M) application as Open Source so that you can modify it for your performance needs. To learn how to make or modify the application, a set of how-to videos with corresponding downloads are available at CoolLutherie.com. Through a step-by-step approach, the videos cover the concepts of audio to MIDI translation in Max. Max, a premier development environment for cross-platform audio applications, is the product of the company Cycling '74. (cycling74.com)

My tutorials cover the concepts and techniques required to create a stand-alone application in Max 9. I take you beyond basic functionality by teaching topics such as the incorporation of Javascript to add critical processing capability to streaming data, how to convert data to JSON for UDP broadcasting, saving settings to an embedded file within the standalone application and more. These topics can be applied to any Max project. It's my way of sharing knowledge that I have gained throughout the years of performing R&D.

Software Overview

With the instrument connected to your computer through an audio interface, the AMT(M).app will be able to read the audio channel and translate the inbound audio stream to MIDI notes. It is important to set the audio preferences correctly and to tune the input to match your playing style. There are visual controls to assist you.

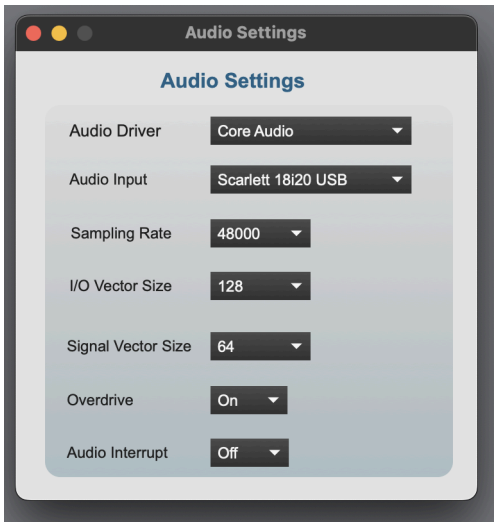
Translating stringed instrument audio to MIDI notes, accurately, is difficult. This is because the frequencies generated by a vibrating string are complex. The software implements a variety of technical solutions to achieve superior results, and it is helpful to understand some of the issues so that your playing can complement them. First, and foremost, your instrument must be in tune. Being in tune dramatically reduces the translation error rate. Secondly, while a string continues to ring, it is difficult to sense the attack of a new note. Since this is a monophonic application that processes each note as it arrives in sequence, you will not be able to play chords; only single notes. Hammer-ons are easy to sense, pull-offs are not. The cleaner you play, the better the resulting translation.

For input adjustment, there is a Gain setting and a Noise Cut setting. The higher the gain, the more sensitive the translation. However, high gain also introduces noise to the signal which can slow the processing. The noise is reduced by increasing the Noise Cut control. The blue LED will light when audio is being processed, and the green LED will light when a MIDI note is being sent. Every guitar and audio interface is different. It is important to tune the application to your style of playing for best results.

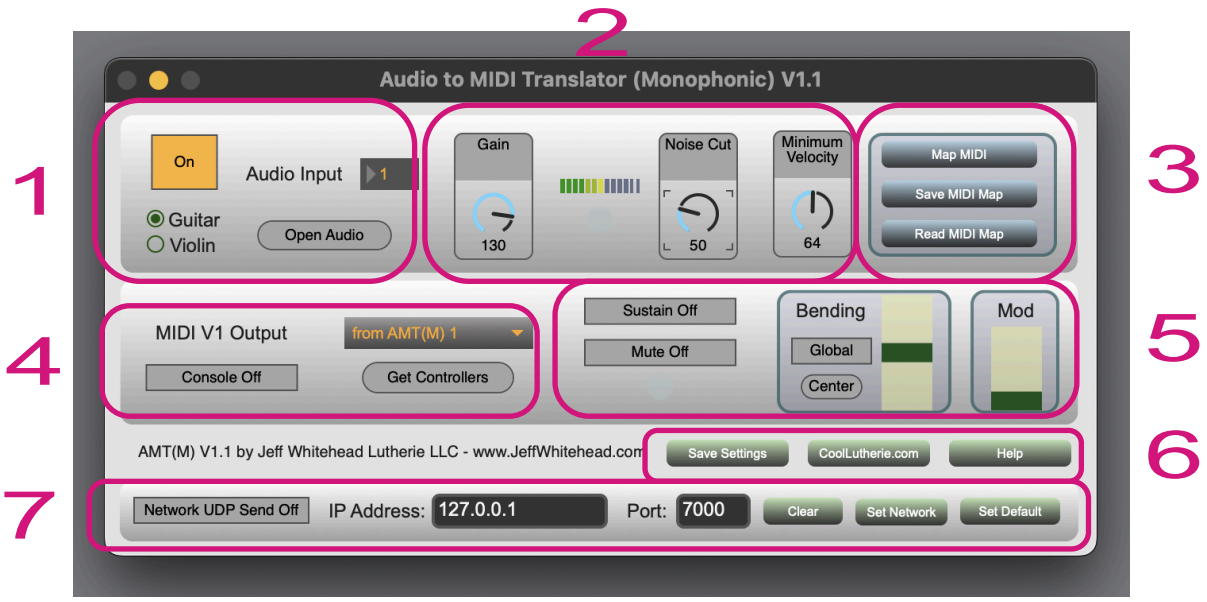
Translation occurs through digital sampling of the audio stream. Sampling rates of 44.1 and 48K are supported. Anything higher than that will not work. Since the process is one of MIDI translation, high sampling rates have no positive effect on the results anyway. Therefore, you do not need an expensive audio interface to achieve excellent results.

The audio interface is configured through a popup panel. A Sampling Rate of 48000, I/O Vector Size of 128, and Signal Vector Size of 64 have worked well on Macs with Intel or M1 chip sets. It is important to have the Scheduler in Overdrive. The Audio Interrupt is normally set to off. These settings will optimize the audio to MIDI translation, reducing latency. To learn more about how Max runtime works, please refer to the documentation by Cycling ‘74. (cycling74.com)

Every computer and audio interface setup is a little different. You will need to try different settings to discover which are best for your system.



Description of the Input and Output Controls



1: Audio Interface. Here, you set the audio interface through which the guitar or violin's audio signal is being sent into the computer. The audio input channel is entered into the field. The On button activates the audio input and defaults the application to its saved settings. The Open Audio button enables you to configure the audio interface. The instrument toggle sets the app to guitar or violin mode. For the selected instrument, the note range is set to standard tuning for audio to MIDI translation, which is the standard for hardware guitar synthesizers.

2: String Audio Controls. There are three adjustments available for matching your playing style to accurate MIDI translation. Gain sets the inbound audio level. Noise Cut sets a volume threshold for the system to ignore inbound audio in order to eliminate false note triggering. Minimum Velocity sets the minimum MIDI velocity for every note being translated.

3: Mapping MIDI controls. An external controller (such as a foot pedal) can be mapped to the app's controls. Once mapped, press the Escape key to leave the mapping mode. Maps can be saved to, and recalled from, an external file.

4: MIDI Output Settings. Set the MIDI output controller from the drop-down menu. The app provides two of its own MIDI output controllers, "from AMT(M) 1" and "from AMT(M) 2." These are the recommended controllers to use. If these controllers are not visible in the menu, the Get Controllers button will refresh the controller list from your computer. MIDI version 1 is sent through the selected controller over channel #1. The Console button is used to toggle output on and off to the Max Console for debugging and R&D purposes.

5: Interactive controls. The Sustain, Mute, Bending, and Mod controls affect all output. MIDI Bend and Mod messages are sent when the sliders are moved. Bending sends the standard pitch bend MIDI 1 message, and the Mod controller sends the standard CC #1 MIDI modulation message. The Bending button toggles between string bending and global bending. Global bending is performed with the slider. String bending and Sustain cannot be used concurrently and will be toggled automatically.

6: Settings. Settings can be saved within the app. To save the current settings, click the Save button. The app will default to the saved settings when the audio button is set to On.

7: Network broadcast. Monophonic data can be sent as JSON across the network via UDP for integration with other software, such as TouchDesigner.

Operating as a MIDI controller for software synthesizers

AMT(M).app becomes a MIDI controller in your environment. For this example, I am using ODC 2800 by Cherry Audio (cherryaudio.com); an incredible synthesizer with a multitude of presets and an intuitive interface that pays homage to, yet expands upon, the gorgeous ARP instruments.

Configuration is simple. With AMT(M) running, click the settings icon on the ODC 2800. Then, from within the Audio/MIDI tab, activate the MIDI input that you selected for MIDI output in AMT(M). Set the audio output in ODC 2800 to your preferred settings. Note that the larger the audio buffer size, the greater the latency in producing sound. So, the smallest audio buffer size setting without producing audible issues like pops and clicks is usually best.

Your guitar or violin will now perform as a MIDI controller for the ODC 2800. In addition to playing notes, AMT(M) sends pitch bend and modulation MIDI messages, just like a keyboard controller.

Since the Pitch Bend and Mod sliders in AMT(M) send the standard MIDI messages, they will interact with those same controls in the ODC 2800 automatically.

To connect the Mod slider in AMT(M) to another control within the ODC 2800 that allows MIDI mapping, control-click the control and select “MIDI Learn” from the pop-up menu. The control will be highlighted in a box, indicating that it is waiting for a MIDI signal. Then, move the Mod slider in AMT(M) to complete the connection.

To remove the connection, control-click the control and select “Unlearn” from the popup menu. The connection will be removed.

AMT(M) sliders interoperate with software automatically through standard pitch bend and modulation MIDI messages.

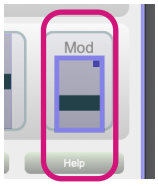
Operating an AMT(M) control from another controller

Just like utilizing MIDI Learn within the ODC 2800, it can also be used within AMT(M) to allow control from another MIDI device. For example, you may wish to have a foot pedal from another controller move the Mod slider that is connected to the ODC 2800's modulation.

Doing so is similar to the operations within the ODC 2800.

Click the "Map MIDI" button. The controls that can accept a MIDI mapping will be visible while everything else will be disabled. Select the control to be mapped. In this example, that is the Mod slider. It will be highlighted in a box, indicating that it has been selected for MIDI learning.

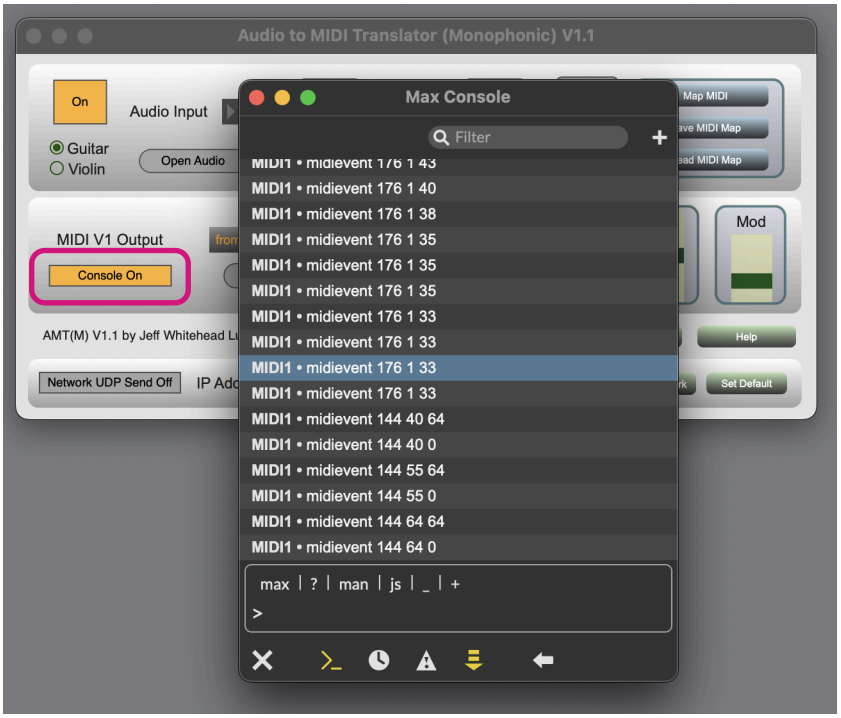
Then, move the foot pedal (or whatever MIDI control you wish to map) and you will see the slider respond. A small block will appear on the slider indicating that it has been assigned to the MIDI control.



Undoing the mapping is simple; with Map MIDI active, control-click the slider and select "Delete Mapping." You will see the small block disappear and the mapping will be dropped.

To leave the MIDI mapping mode, press the Escape key.

Using the Max Console to view MIDI output



Being a Max application, AMT(M) can take advantage of the real time signal processing and data display that the Max environment provides. One of the most important utilities from Max is the Console. The Max Console displays information in real time that the developer wishes to post. For AMT(M), that information is the MIDI 1 messages that are being sent from the application through the selected controller output.

These messages include notes on and off with velocity and channel information. Pitch bend and Mod messages are also displayed.

If you are having difficulty integrating with another MIDI application, it can be helpful to see what is being sent from AMT(M) in real time by using the Console.

If you are a visual artist using the MIDI messages to control live performances, this a valuable tool to view the actual data.

To launch the Max Console from the AMT(M) main menu, select "Window" then "Max Console."

To send the message output to the console, click the Console On/Off button to On.

Recommended Practices for Optimal Performance

For accurate audio to MIDI translation, there are techniques for setting up the guitar or violin, the audio interface, and the AMT(M) software to achieve optimal performance.

Before any other activity, ensure that your instrument is in tune. The AMT(M) app is sampling audio wave forms from a stringed instrument that are very complex. A properly tuned instrument will contribute to accurate translation with minimal latency.

The guitar pickup produces a relatively weak audio signal. An audio interface with an input volume control is best because that can adjust the signal. Increase the volume to a point just below clipping. A line input without amplification for the output of a pickup will not provide an adequate audio signal for accurate MIDI translation.

Within the AMT(M) app, increasing the Gain control until the blue LED flashes *while a note is not being played*, indicates the maximum acceptable input volume. At that point, the app is reading only noise from the audio signal and attempting to perform audio to MIDI note translation. Reducing the gain until the LED goes off brings the signal to below the noise threshold and to its most sensitive sampling state.

Playing a note should trigger the blue LED. Depending upon your playing style, such as finger picking versus using a plectrum or bowing, you may adjust the gain until the LED flashes at the moment of attack. The illuminated blue LED indicates that there is an audio signal available for translation. When the note is actually being translated, the green LED will be illuminated.

If note triggering is overly sensitive, then the Noise Cut control may be increased to filter out the ambient noise. Between the adjustment of the Gain and Noise Cut controls, an optimal setting can be achieved for your instrument, interface, and playing style.

Sensing a duplicate note from a stringed instrument without creating false duplicate notes is a complicated matter. Related in complexity is the ability to distinguish a sustained note from a re-triggered note. If you wish to rapidly re-play a note, it is best to dampen the note when played so as to reduce the sustained vibrations.

Lastly, the Minimum Velocity control sets the minimum MIDI note velocity for all notes being played. The numeric value of the Minimum Velocity control is a MIDI note velocity, ranging from 0 to 127. Setting a high value will result in stronger note attack by the MIDI instrument when the note is played. Some synthesizer software is more responsive when the note velocity is higher.

After the settings have been made, it is important to save them. Click the Save button to save them as the application's default. The settings will be saved within the app. The settings will be used when the audio is turned on.

Note: If the MIDI controllers that are active on your computer change, the MIDI controller that you saved may need to be re-selected when the app is re-launched. This is because the index value of the saved controller will no longer match the name of the controller that you wish to be using. Click the Get Controllers button to refresh the list, then reselect the controller that you wish to use.

Recommended Synthesizers for new and experienced players

I cited the ODC 2800 synthesizer as an example. All of the Cherry Audio synthesizers perform brilliantly and are available as downloads with a free trial period. Go to: cherryaudio.com to learn and explore. Each synthesizer comes with a large selection of presets. With AMT(M), lead and arp presets work very well due to their monophonic nature. You will discover that an incredible palette of tone awaits. Try them all!

SWAM solo instruments by Audio Modeling are incredible virtual instruments that emulate the expressiveness of strings, woodwinds, and brass. SWAM instruments require a controller for expression. Without one, there will be no sound from the instrument. The AMT(M) Mod slider can be mapped as that controller. Using the Mod slider will enable you to play SWAM instruments without purchasing any other hardware, such as a breath controller, for expression. Audio Modeling provides software downloads on a free trial basis.

AMT(M), ODC 2800, SWAM Bassoon, and TouchDesigner running concurrently.

Network broadcasting of monophonic data

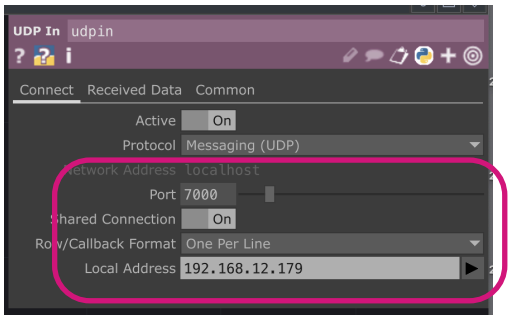
TouchDesigner is a real time graphics design environment and display engine that is available to download from derivative.ca. For integration with TouchDesigner, AMT(M) converts the monophonic MIDI information to JSON and transmits it over the network as a UDP broadcast. Within TouchDesigner, this JSON can be read in real time and translated to data that manipulates visual performances. TouchDesigner can be running on a Mac or Windows computer within the same network.

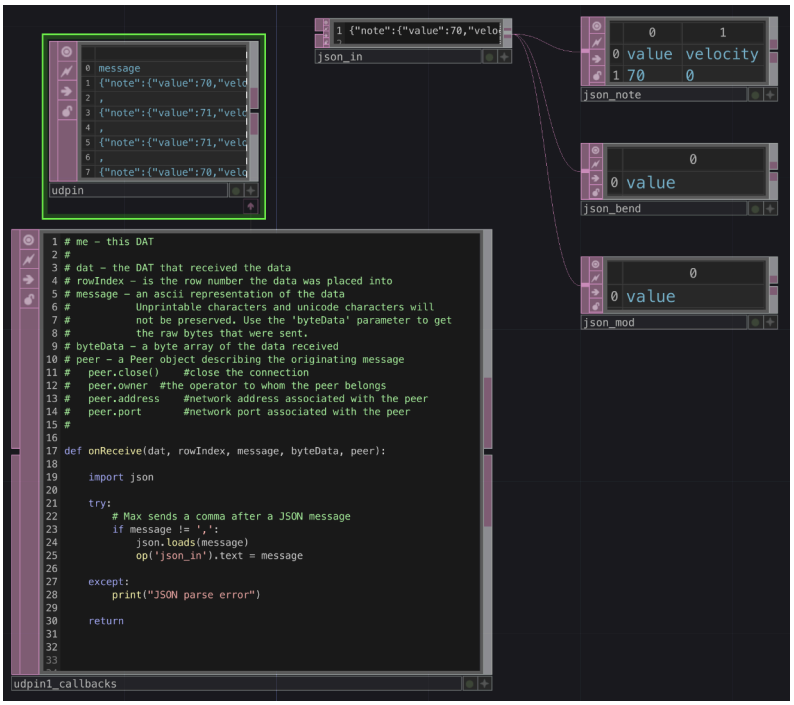
If TouchDesigner and AMT(M) are running on the same computer, network broadcasting can be used to intercommunicate by entering the computer's local TCP/IP address of 127.0.0.1 and selecting a Port value that is not in use by other applications. The default value for the Port is 7000.

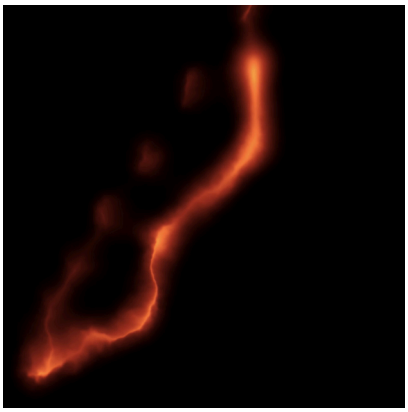
When the applications reside on separate machines, the network TCP/IP Address and listening Port of the TouchDesigner computer are set within both applications.

A fully-functional TouchDesigner example is available at CoolLutherie.com. Drop the code into your own project, set the values for Local Address and Port within the TouchDesigner "UDP In" object, and your network will be instantly connected to AMT(M). If not, check to ensure that the computers allow UDP. Sometimes a firewall rule must be added.

A polyphonic version of the JSON/TouchDesigner example is available for the AMT(P) application. AMT(P) broadcasts MIDI 2 messages that take advantage of MPE, including concurrent notes with their respective bends and global aftertouch. The only difference between the two TouchDesigner examples is availability of the additional MIDI 2 data when in polyphonic mode. Otherwise, the integration is identical.







A note-manipulated animation in TouchDesigner.

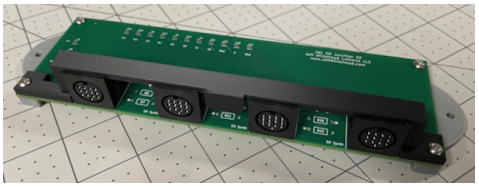
Jeff Whitehead Lutherie LLC

I have been making stringed instruments for many years. It started in my youth, along with an interest in electronics. During my middle life, I was a software engineer and entrepreneur. So, the natural progression was to bring instrument design and computer processing together. This Audio to MIDI Translator application is an important part of that journey and I am delighted to be able to share it with you.

My instruments are enjoyed by professional musicians, collectors, and enthusiasts. Each is unique, part of an evolving process that involves a lot of experimentation; something that I enjoy immensely. My instruments can be viewed at JeffWhitehead.com.

To share the R&D experience that I have acquired through the years of instrument building and software development, I created the site, CoolLutherie.com.

At Cool Lutherie, you may watch videos about this and other projects that showcase fun and interesting things in the world of electronic stringed instruments and computer integration. From there, you can learn from, and download, helpful content. Once you are comfortable working with MIDI, you may want to move on to polyphony with the audio breakout board. The audio breakout board splits the analog output from a divided pickup to separate audio jacks. This is the critical piece of hardware to experience true, polyphonic processing. I have designed a variety of PCB's to account for most applications. These range from a simple board that supports a single, 13-pin socket to a junction that can interconnect up to three hardware synthesizers to run in parallel with the audio processing. These PCB's and other related parts can be purchased through Cool Lutherie.



It is important to me that this information is provided as open source, enabling you to create your own hardware and software to meet your artistic needs. Therefore, this and other projects that I provide are done so under the Creative Commons licensing. This also means that my work is supported by donations. So, if you find my content helpful and interesting, please make your donation at CoolLutherie.com.

The reason that I focused on MIDI integration with stringed instruments, in the first place, was that I found hardware guitar synthesizers to be limited in capability when compared to the exploding software market. I wanted a simple, polyphonic interface that allowed me to use the cool stuff that was being enjoyed by keyboard players. And… I wanted to develop something that could be easily shared with other artists. So, I created this system.

Once you begin working with MIDI integration, new artistic opportunities present themselves, such as MIDI-triggered visual performances, stacking software synthesizers for a huge sound, combining computer with analog audio, adding additional modulation through controllers, and more; a whole new world of possibilities for your studio and live performances shall be open to you.

Where will MIDI lead you?

Enjoy the journey!

Important Information

About this application

This application is written in Max, a premier development environment for cross-platform audio applications. It is the product of the company Cycling ‘74. Therefore, you may refer to the Cycling ‘74 forums to learn more about the Max environment and applications created in Max. Go to: cycling74.com/forums for more information.

This Audio to MIDI Monophonic Translator application, AMT(M).app, written by Jeff Whitehead, was compiled within Max 9 to run as a standalone app on a Mac computer and is available as an official distribution only from CoolLutherie.com.

If you have not received this software from CoolLutherie.com then it may not be an official distribution.

Source code and tutorials for audio to MIDI translation are available through CoolLutherie.com and can be modified within the Max software development environment, a product of Cycling ‘74. (cycling74.com)

Intended Use

The AMT(M) app is written by Jeff Whitehead Lutherie LLC and is available as a DIY project through CoolLutherie.com.

The AMT(M) app performs monophonic translation of audio to MIDI notes by analyzing the audio stream that is received through a computer’s audio interface. It is designed to support electric guitars and violins with standard tunings.

Download Contents

The archive download for this Mac standalone version includes the following files:

AMT(M).app - this application

AMT(M)_readme.txt - important information about this software

AMT(M)_JSON_Example.toe - a TouchDesigner example network for JSON integration

Warranty

License

To learn how to distribute your work under this license, please refer to:

https://creativecommons.org/licenses/by-nc-sa/4.0/

This license enables reusers to distribute, remix, adapt, and build upon the material in any medium or format for noncommercial purposes only, and only so long as attribution is given to the creator. If you remix, adapt, or build upon the material, you must license the modified material under identical terms. CC BY-NC-SA includes the following elements:

BY: credit must be given to the creator, Jeff Whitehead Lutherie LLC, at JeffWhitehead.com

NC: Only noncommercial uses of the work are permitted.

SA: Adaptations must be shared under the same terms.

Adaptations must include the attribution to the original and preceding authors and describe all changes that have been made to the preceding work.