Introduction

The osxAcousticSL software lets you implement a real-time sound source localization algorithm using two or four signals acquired from digital MEMS microphones to estimate the arrival direction of an audio source.

It is based on three different DOA algorithms exploiting cross correlation in the time domain, generalized cross correlation with phase transform and a matching pursuit routine using sparse representation framework.

The angle can be estimated over a 180 or 360 degree range, depending on the number of channels adopted and microphone placement.

The resolution of the computed value can be chosen at runtime, allowing you to determine the best tradeoff between localization precision and resource consumption.

The osxAcousticSL library is provided in binary format inside a software package with sample applications running on the X-NUCLEO-CCA02M1 expansion board connected to a NUCLEO-F401RE board.

The example package is an add-on for the X-CUBE-MEMSMIC1 package; it is based on STM32Cube technology and is easily ported to any STM32F4 microcontroller with an FPU unit.

Contents

1 Licensing information .................................................. 4
2 osxAcousticSL software library ........................................... 5
  2.1 Angle resolution ......................................................... 5
    2.1.1 XCORR cross-correlation algorithm ................................ 5
    2.1.2 GCC-PHAT algorithm ............................................... 6
    2.1.3 BMPH algorithm .................................................... 6
  2.2 Microphone geometry ................................................... 6
    2.2.1 Two-microphone scenario ........................................... 6
    2.2.2 4-microphone scenario ............................................. 7
3 osxAcousticSL software expansion for X-CUBE-MEMSMIC1 ....... 8
  3.1 Overview ............................................................... 8
  3.2 Architecture ............................................................ 8
  3.3 Folders Structure ....................................................... 9
  3.4 APIs ................................................................ 10
4 System setup guide ............................................................ 11
  4.1 Hardware setup ........................................................... 11
  4.2 Firmware and software .................................................. 11
    4.2.1 Additional package download and unpacking ......................... 11
    4.2.2 Library package installation ........................................ 12
    4.2.3 Node locking procedure ............................................. 15
    4.2.4 Compiling the example projects ..................................... 16
5 Sample application description .............................................. 18
  5.1 Firmware steps .......................................................... 18
  5.2 Example program execution ............................................. 18
6 Library Profiling ............................................................... 20
7 References ................................................................. 21
8 Revision history ........................................................... 22
List of figures

Figure 1: Two-microphone arrangement .................................................................6
Figure 2: Four-microphone arrangement ...............................................................7
Figure 3: Resulting software architecture ..............................................................9
Figure 4: Package folder structure .......................................................................9
Figure 5: Directory content after unpacking the X-CUBE-MEMSMIC1 package ..........12
Figure 6: License Agreement ..............................................................................12
Figure 7: Information and additional package requirements .................................13
Figure 8: Installation directory for the PC software ..............................................13
Figure 9: Installation directory for the firmware ..................................................14
Figure 10: OSX LicenseWizard tool installation ..................................................14
Figure 11: Resulting firmware workspace ............................................................15
Figure 12: Library selection ..............................................................................15
Figure 13: License request generation .................................................................16
Figure 14: Error while building the original osx_license.h file ..............................16
Figure 15: osx_license.h file after the required modifications ..............................17
Figure 16: STLink Virtual COM port detection ....................................................19
1 Licensing information

This software library is licensed under ST OpenSoftwareX Limited License Agreement, (the "OSX-LLA License"). You may obtain a copy of the OSX License at:

OpenSoftwareX license agreement

Some of the library code is based on the CMSIS DSP software library by ARM, a suite of common signal processing functions for use on Cortex-M processor based devices. Licensing terms are available in the attached release_note.html file, in the next lines of this document and on the web at:


ARM license note:

Copyright (C) 2009-2012 ARM Limited. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of ARM nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL COPYRIGHT HOLDERS AND CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
2 osxAcousticSL software library

This library uses audio signals from microphones to execute Sound Source Localization and estimate the angle of arrival of the audio source. It embeds three algorithms with differing RAM and MIPS footprints and overall performance [1]:

- **XCORR cross correlation algorithm**:
  - performed in the time domain
  - less resource intensive
  - lower resolution
  - requires microphones to be placed at greater distances

- **GCC-PHAT algorithm**
  - performed in the frequency domain
  - relatively resource intensive
  - higher resolution
  - performance should not be impacted by microphone distance

- **BMPH algorithm**
  - performed in the frequency domain
  - more resource demanding than XCORR, but less than GCC-PHAT
  - performance should not be impacted by microphone distance
  - performs a hangover step to enhance source stability

The basic versions of these algorithms use two audio signals to estimate an angle in a 180 degree range. A 360 degree range requires four audio streams and appropriate library configuration. The algorithm and the number of microphones are chosen by the user during library initialization. The software is designed to run in a real-time environment.

2.1 Angle resolution

Angle output resolution mainly depends on the chosen algorithm and how the result is computed:

2.1.1 XCORR cross-correlation algorithm

Cross-correlation is based on time-domain computation and the resolution depends on both the sampling frequency and the distance between microphones. For each combination of these parameters, the maximum delay between two microphones (in terms of samples) is:

\[
\text{DelayMax} = \text{Floor}\left(\frac{(\text{Distance}) \times (\text{Sampling\ Frequency})}{\text{Sound\ Speed}}\right)
\]

This value is strictly related to the maximum number of angles that can be discerned by the routine and thus determines the output resolution.

The minimum detectable angle over 180 degrees is thus:

\[
\text{MinimumDetectableAngle} = \text{Floor}\left(\frac{180}{2 \times \text{DelayMax}}\right)
\]

The larger the microphone distance, the higher the possible resolution, at a higher MIPS cost. If this algorithm is chosen, the achievable resolution is computed automatically by the initialization function and cannot be set by the user. In this case, the resolution parameter is ignored by the initialization function.
2.1.2 GCC-PHAT algorithm

The GCC-PHAT algorithm works with frequencies and generally offers a better resolution even in low inter-microphone spacing configurations. The user can set the desired resolution with a dedicated parameter in the initialization structure (see the chm help file in the Documentation folder). In this case, the higher the resolution, the higher the MIPS consumption.

Please note that the resolution value relates to the use of the library with two channels. When the library is used with four channels to perform source localization over a 360 degree range, two detected values are merged internally and the overall resolution may be different to the initial setting.

2.1.3 BMPH algorithm

The block matching pursuit with hangover (BMPH) algorithm also works with frequencies and the user can set the desired resolution with a dedicated parameter in the initialization structure.

The best resolution available for the BMPH algorithm is four degrees. While MIPS consumption and memory requirements of the BMPH algorithm are generally lower than for the GCC-PHAT algorithm, the algorithms exhibit similar performance because the algorithm adaptively analyzes the most relevant frequencies in the frame.

2.2 Microphone geometry

2.2.1 Two-microphone scenario

*Figure 1: “Two-microphone arrangement”* shows the plan view of microphones M1 and M2.

The minimum inter-microphone spacing depends on the algorithm:

- for XCORR, the distance must allow at least one sample delay between them
- GCC-PHAT has potentially no distance constraints

*Figure 1: Two-microphone arrangement*

The estimated angle is measured taking as a reference the imaginary line passing through the middle point of the segment joining the two microphones and orthogonal to it, as depicted in the picture.
2.2.2 4-microphone scenario

Figure 2: "Four-microphone arrangement" shows the arrangement of microphones M1, M2, M3 and M4. The same distance constraints as in the two-microphone scenario must be observed, but they are measured between opposite microphones; i.e., between M1-M2 and between M3-M4.
3 osxAcousticSL software expansion for X-CUBE-MEMSMIC1

3.1 Overview

osxAcousticSL software package expands the functionality provided by STM32Cube. It is designed as an add-on for X-CUBE-MEMSMIC1 pack and exploits its features for digital MEMS microphones acquisition, decimation and streaming. For further information regarding X-CUBE-MEMSMIC1, including the microphone acquisition process and software downloads, please refer to www.st.com.

For details on how to merge the two packages into a single package, refer to Section 6.2: “Firmware and software”.

The key features of this add-on package are:

- osxAcousticSL library middleware
- sample implementation available for X-NUCLEO-CCA02M1 board plugged on top of a NUCLEO-F401RE, based on the X-CUBE-MEMSMIC1 package
- detailed chm documentation of the library API

3.2 Architecture

The package extends X-CUBE-MEMSMIC1 by providing:

- an additional middleware component for the osxAcousticSL library
- a sample implementation that exploits the X-CUBE-MEMSMIC1 capabilities for microphone acquisition, decimation and streaming.

The two packages must therefore be merged in order to obtain a fully functional application based on the osxAcousticSL library, available to the user as a reference for software design and library integration.

The resulting application architecture includes the following layers needed to access the microphone data:

- **STM32Cube HAL layer**: consists of a set of simple, generic, multi-instance APIs (application programming interfaces) which interact with the upper layer applications, libraries and stacks. These generic and extension APIs are based on a common framework which allows any layers they built on, such as the middleware layer, to implement their functions without requiring specific hardware information for a given microcontroller unit (MCU). This structure improves library code reusability and guarantees easy portability across other devices.

- **Board Support Package (BSP) layer**: provides software support for the STM32 Nucleo board peripherals, excluding the MCU. These specific APIs provide a programming interface for certain board specific peripherals like LEDs, user buttons, etc and can also be used to fetch individual board version information. It also provides support for initializing, configuring and reading data.
### 3.3 Folders Structure

The following folders are included in the software package:

**Documentation**: contains a compiled HTML file generated from the source code, detailing the software components and APIs.

**Drivers**: contains the HAL drivers, the board specific drivers for each supported board or hardware platform, including those for the onboard components and the CMSIS layer which is a vendor-independent hardware abstraction layer for the Cortex-M processor series.

**Middlewares**: contains the osxAcousticBF library binary code, documentation and license information.

**Projects**: contains a sample application used to demonstrate the library, provided for the NUCLEO-F401RE platform with three development environments: (IAR) Embedded Workbench for ARM, RealView Microcontroller Development Kit (MDK-ARM) and System Workbench for STM32 (SW4STM32).
3.4 APIs

Detailed technical information fully describing the functions and parameters of the osxAcousticSL APIs can be found in the osxAcousticSL_Package.chm compiled HTML file located in the Documentation folder of the software package. In the same document, you can find general descriptions regarding all the osxAcoustic library APIs and the concepts behind their design.

osxAcousticSL is provided as a node-locked library which allows derivative firmware images to run on a specific STM32 Nucleo device only. Licensing activation codes must be requested from ST and included in the project (and become part of the build process) prior to attempting its usage. The resulting firmware binary image will therefore be node-locked.

For complete information about the open.AUDIO license agreement, please refer to the license file located in the Middlewares/ST/STM32_OSX_AcousticSL_Library folder.
4 System setup guide

4.1 Hardware setup
The library sample application needs the following hardware:

- NUCLEO-F401RE board
  Further details can be found at www.st.com
- Four external digital microphones based on the ST coupon daughterboard concept (part number: STEVAL-MKI129V* or STEVAL-MKI155V*). Find further details regarding possible choices at www.st.com
- X-NUCLEO-CCA02M1 expansion board

For this specific application, the board solder jumpers must be configured in order to acquire four microphones using the abovementioned external coupon boards. Note that this is not the default factory setting, so you may need to change solder jumper configurations and add strips line in order to host the coupon boards.

The correct setup is described in [2]. Refer to the same user manual for information regarding board connections and supplying power.

4.2 Firmware and software
The main steps required to setup the environment are:

- Additional packages download and unpack
- Library package installation
- Node locking procedure
- Compiling and running

4.2.1 Additional package download and unpacking
The application is designed as an add-on project for the X-CUBE-MEMSMIC1, which you can find at www.st.com.

After downloading, unpack the zip file in a directory of your choice. This will become the workspace used to test and work with the library. Avoid using paths names longer than 240 characters because it can cause problems in some tool chains during project build. In our example below, the workspace is placed in the “C:\Workspace” folder.
4.2.2 Library package installation

Double click the “osxAcousticSL_Setup” library installer available at www.st.com in order to start the installation procedure.

The first step is to accept the License Agreement:

Then the procedure details the additional packages required:
Now you can choose the directory where you want to install the library software. Note that this is not the path for the firmware, but where the PC software related information (uninstall and other information) shall be stored.

This folder does not necessarily have to be your workspace folder.

Then you choose the directory where the firmware will be unpacked.

This folder must be your desired workspace directory (“C:\Workspace” in our example).
Finally you can complete the installation procedure.

The LicenseWizard software is required for the next steps.

At this stage, your Workspace folder should resemble the image below.
4.2.3 Node locking procedure

Connect a NUCLEO-F401RE board to the PC, run the OSX LicenseWizard you installed in the previous steps and choose the library you want to activate.

Now use the appropriate buttons on the wizard to:

- identify the STM32 Nucleo board
- generate the license request
- send the request e-mail
- License request generation
The license mail should arrive after a few moments.

4.2.4 Compiling the example projects

Open the project with your favorite tool chain. In this example, the projects for three IDEs are located in “C:\Workspace\Projects\Multi\Applications\Acoustic_SL”.

Try to compile the project, you should receive the following error:

Open the osx-license.h file and substitute the dummy license array with the one you received by mail from the Open.Audio server. Also, comment the define directive describing the error.
You should now be able to compile and flash the board for which you requested the license in order to use the library example.
5 Sample application description

An example application using the osxAcousticSL library with NUCLEO-F401RE and X-NUCLEO-CCA02M1 boards is provided in the “Projects” directory. Ready to build projects are available for multiple IDEs.

The application is designed to perform:

- Acquisition of the four external microphones mounted on the X-NUCLEO-CCA02M1 board.
- Sound source localization running basing on those signals.
- Output of the localization result through the ST-Link embedded virtual COM port.
- Streaming of four audio channels via X-NUCLEO-CCA02M1 USB connector to a host PC.

5.1 Firmware steps

Follow these steps to obtain the desired behavior:

- Initialize and start the USB audio input driver and middleware. This allows a host PC to recognize the device as a standard multichannel USB microphone.
- Initialize microphone acquisition using the relevant BSP function.
- Initialize the osxAcousticSL library.
- Start the audio acquisition that will trigger library execution.
- The four omnidirectional microphones streams are be passed to USB driver each millisecond.
- The osxAcousticSL results are sent through ST-Link VCP to a host PC on each library run.

All the operations related to the osxAcousticSL library are performed in dedicated functions in the “audio_application.c” file.

Further details about library API can be found in the .chm help file in the Documentation folder.

5.2 Example program execution

Third party serial communication software is required to be able to collect localization values. One possible solution is the free Putty software available at: http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html.

To set up the demo:

- Connect NUCLEO-F401RE mini USB cable (USB type A to Mini-B USB cable).
Figure 16: STLink Virtual COM port detection

- Check the COM Port number assigned to the board in Windows Device Manager.
- Open the serial utility and ensure the connection parameters are set to:
  - Baudrate: 9600
  - Data bits: 8
  - Stop bits: 1
  - Parity: none
  - Flow Control: none
- Open the COM port

Now you should be able to see the localization results on the screen. Note that there is an audio energy threshold below which the library is not executed, in which case localization is not computed and nothing is sent to the VCP.

In order to record audio from the device, you need to install third party software such as the Audacity® freeware program to save or play the streamed signal.

Detailed information on how to setup the host system for audio recording can be found in the X-CUBE-MEMSMIC1 user manual, available at www.st.com.

Further information about Audacity® can be obtained at: http://audacityteam.org/?lang=en
6 Library Profiling

Profiling was performed in order to evaluate the library resource consumption in terms of MIPS, RAM and Flash. Detailed information can be found in the osxAcousticBF_Package.chm compiled HTML file located in the Documentation folder.
References

2. User manual, UM1900 - Getting started with the digital MEMS microphones expansion board based on MP34DT01-M for STM32 Nucleo. STMicroelectronics
8 Revision history

Table 1: Document revision history

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>02-Dec-2015</td>
<td>1</td>
<td>Initial release.</td>
</tr>
<tr>
<td>01-Sep-2016</td>
<td>2</td>
<td>Throughout document:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- added BMPH algorithm information</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- minor text and formatting changes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added Section 2.1.3: &quot;BMPH algorithm&quot;</td>
</tr>
</tbody>
</table>
IMPORTANT NOTICE – PLEASE READ CAREFULLY

STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST’s terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of Purchasers’ products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2016 STMicroelectronics – All rights reserved