NAO

KEY FEATURE
AUDIO SIGNAL PROCESSING

Abstract

In robotics, due to the limited computational power available on embedded processors, it can be useful to deport some calculations to a remote desktop or server. This especially true for audio signal processing; for example, speech recognition can often be done more efficiently - faster and more accurately - on a remote processor, and so most modern smartphones process voice recognition remotely.

Users may also wish to apply their own signal processing algorithm directly on the robot. For example, the user may want to analyse the input signal to detect sound events such as music, ring-tones, or voice...

This paper describes how the audio modules are organized on the NAO robot, how to access NAO's microphone sound date, and how to send data onto NAO's loudspeakers either locally or remotely.

Related work

- Due to the constant growth of the internet, much research has been done to make it possible to send large amounts of data (audio or video) over the network, and many protocols have been developed to achieve this goal and optimize data transmission. The NAOqi framework uses one of these protocols (SOAP - Simple Object Access Protocol) to send and receive audio signals over the web.
- Sound is produced and recorded on NAO using the ALSA (Advanced Linux Sound Architecture) library.

Principles

NAOqi currently contains six audio modules which are interdependent and organized as described in the **FIGURE 1**.

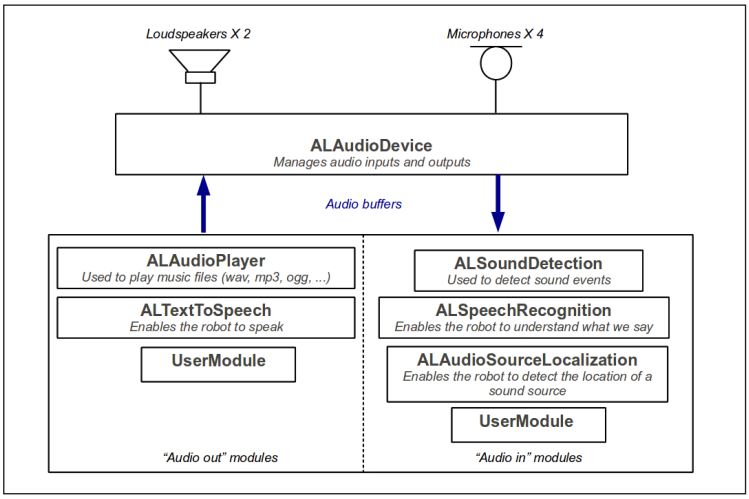

FIGURE 1
Architecture of the NAOqi audio modules

The ALAudioDevice module manages the audio inputs and outputs. Consequently, every module which is intended to send sound onto NAO's loudspeakers or to process sound coming from NAO's microphones must communicate with the ALAudioDevice module.

Real-time sound processing

In order to do real-time signal processing on the microphones input signals, the first step is to create an "Audio in" module. Then, this module must subscribe to the ALAudioDevice module, which will send the input buffer via a callback function.

ALAudioDevice sends the buffers sequentially, i.e. the input buffer is sent to the first subscribed module, and then it will be sent to the second one once the first one has finished its processing, etc...

This working mode is illustrated on the **FIGURE 2**.

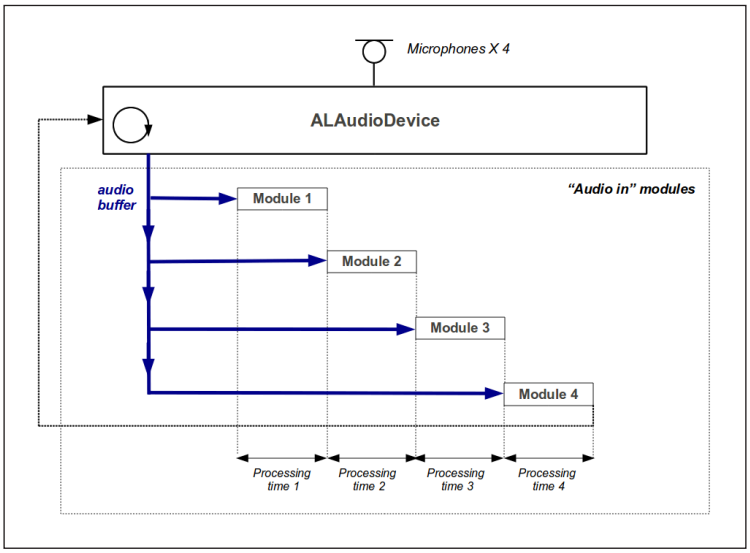

FIGURE 2
Working mode of the input buffers sending and processing.

As ALAudioDevice reads its input buffers at a regular time interval fixed by the inputBufferSize parameter and the input sample rate (48kHz), the total processing times of the subscribed modules should not exceed this time interval, or audio buffers will be missed and processing will no longer be real-time.

Performances

The advantage of processing sound remotely is that the user can easily debug or optimize their sound processing algorithms, and that these algorithms can take more CPU load than on the robot. However, the transfer speed of the audio buffers will be faster if the user module runs directly on NAO.

Limitations

- The maximum input buffer size which can be obtained with ALAudioDevice is 65536 (16384 16-bit samples per microphone channel).
Therefore, the maximum total processing time available for a buffer is 341ms.
- The maximum buffer size which can be sent through ALAudioDevice is 32768 (16384 16-bit stereo samples).
- Due to the 48kHz input sample rate, the remote sound processing needs at least a 384 kBits/s connection bitrate. If the user's connection speed is too low, or if the link quality is poor, audiobuffers will be missed.

How does it work?

- **Accessing sound data from NAO's microphones**

To access to the sound data coming from NAO's microphones, the user first has to create a NAOqi module (in C++ or Python language) which contains a callback function called «processSound» or «processSoundRemote» depending on whether the module is executed locally or remotly.

Then, to start receiving the audio buffers, this module simply has to call the subsribeLocalModule or subsribeRemoteModule function of the ALAudioDevice module (depending on whether the module is executed locally or remotely).

Every time an audio buffer is ready for processing, the ALAudioDevice module will send it through the processSound or processSoundRemote function. The received buffer contains the 4 16-bit microphone signals sampled at 48kHz. These samples are interleaved, that is to say that the buffer contains s1m1, s1m2, s1m3, s1m4, s2m1, s2m2, ... where simj is the sample number i of microphone j. The order of microphones is as follow: 1: left microphone / 2: right microphone / 3: front microphone / 4: rear microphone.

There is an example of this in the SDK: /modules/src/examples/sounplayback.
Or you can download it at:
http://aldebaran-robotics.com/multimedia/soundplayback.zip

- **Sending sound to NAO's loudpseakers**

To send sound to NAO's loudpseakers, the user just has to call the sendLocalBufferToOutput or sendRemoteBufferToOutput function of the ALAudioDevice module, depending on whether his module is executed locally or remotely.

The buffer given in argument of this function must contain 16-bit stereo sound samples. The output sample rate can easily be adjusted by a call to the setParameter function of the ALAudioDevice module.

There is an example here:
http://aldebaran-robotics.com/multimedia/rms_level_calculation.zip

How are people using it?

Using NAO's audio capacities, a wide range of experiences and research can be done in the fields of Human-Robot Interaction and communication. For instance, NAO may be used as a communication device. The user could interact with NAO (talk and hear) as if he were talking to another human being.

Signal processing is of course an interesting example. Thanks to the audio module, you can get the raw audio data from the microphones in real time and then process it with your own code. For example, the LIMSI lab is working on detecting emotions in human voice (for more information: **http://gdr-isis.fr/uploads/comptes-rendus/reunion-123/04_Devillers.pdf**).