# RobotX task: Underwater Acoustic Detection System

Sam Liu last modified on Nov. 16, 2019.

The objective of this tutorial is to introduce a **RobotX task: Underwater Acoustic Detection Hardware & Software system.**

*Any sentence behind “****$****” means those are commands that typed in the terminal*

*(****$*** *代表terminal的指令)*

## Hardware and Software Setup

| **##### Both laptop and Raspberry pi need to do these instruction first #####**  $ cd ~/ && git clone <https://github.com/RobotX-NCTU/robotx-underwater-task.git>  $ cd robotx-underwater-task  --------------------- In the container ---------------------  $ catkin\_make -C catkin\_ws # Compile  $ exit # Exit from docker container |
| --- |

For acoustic source control:

* Raspberry Pi 3 model B or newer
* Docker (version 19.03 or newer)
* USB sound card
* Underwater pinger hardware (audio amplifier, underwater speaker, relay control)

For acoustic analysis:

* Laptop (amd64 architecture)
* Docker (version 19.03 or newer)
* Acoustic recording hardware (Hydrophone array, pre-amplifier, audio recorder)

[**RobotX task: Underwater Acoustic Detection System**](#_jkop8kfo2r4s) **1**

[Hardware and Software Setup](#_rhtwi3gigp91) 1

[Overview](#_4pknuww40csp) 2

[Topics and Activities](#_oizs47ynqr4f) 3

[Topic/Activity 1 Acoustic source system (Pinger)](#_1nmm7g8819ns) 3

[Topic/Activity 2 Acoustic recording system](#_9m68ywphc0z) 4

[Topic/Activity 3 Realtime data visualization](#_yhq00hsek0cg) 7

[Topic/Activity 4 Sound analysis](#_6ccbhzqoh29r) 8

[DEMO](#_8p49024eoxmd) 9

[Trouble shooting](#_61no9c946nc0) 10

[Reference](#_6fdhrutdoo5m) 10

## Overview

To set the underwater acoustic source control system, you have to prepare the equipments shown below, included the laptop, the high power Wi-Fi hotspot and three acoustic source hardware (We will call “pinger” in later paragraph).

On the other hand, the acoustic recording system is much simpler than acoustic source control system. We will introduce it in Topic 2.

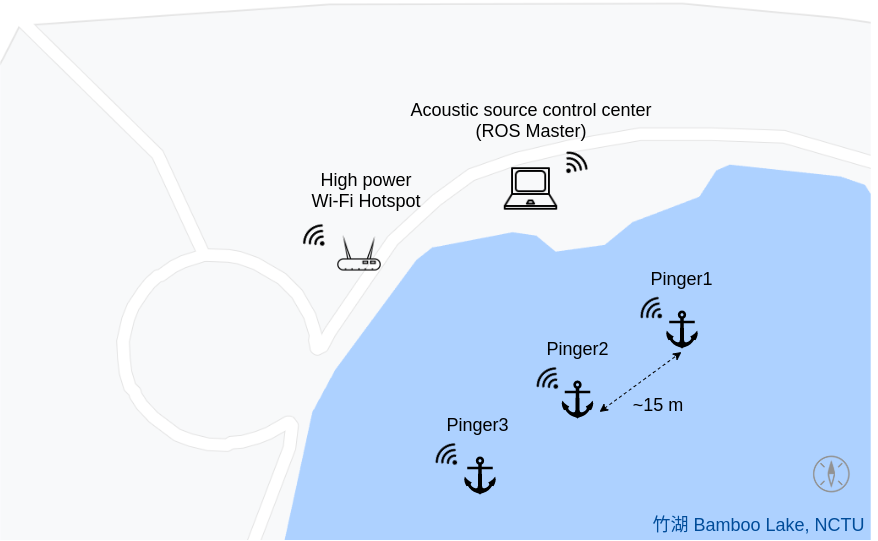


Fig. Underwater Pinger(acoustic source) System Setup

After completing this tutorial you should

* understand how to set up the enviroment for underwater sound detection
* be able to run Time Difference of Arrival (TDOA) algorithm

## 

## Topics and Activities

### Topic/Activity 1 Acoustic source system (Pinger)

This acoustic source system is controlled by Raspberry Pi model B. Because of the frequency response limit of USB sound card and audio amplifier. The pinger only play the audio under 20kHz. In the contrast.

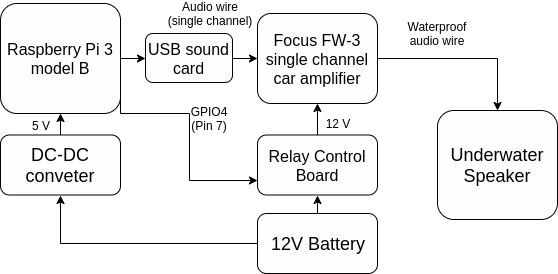


Fig. Underwatet Pinger(acoustic source) system architecture

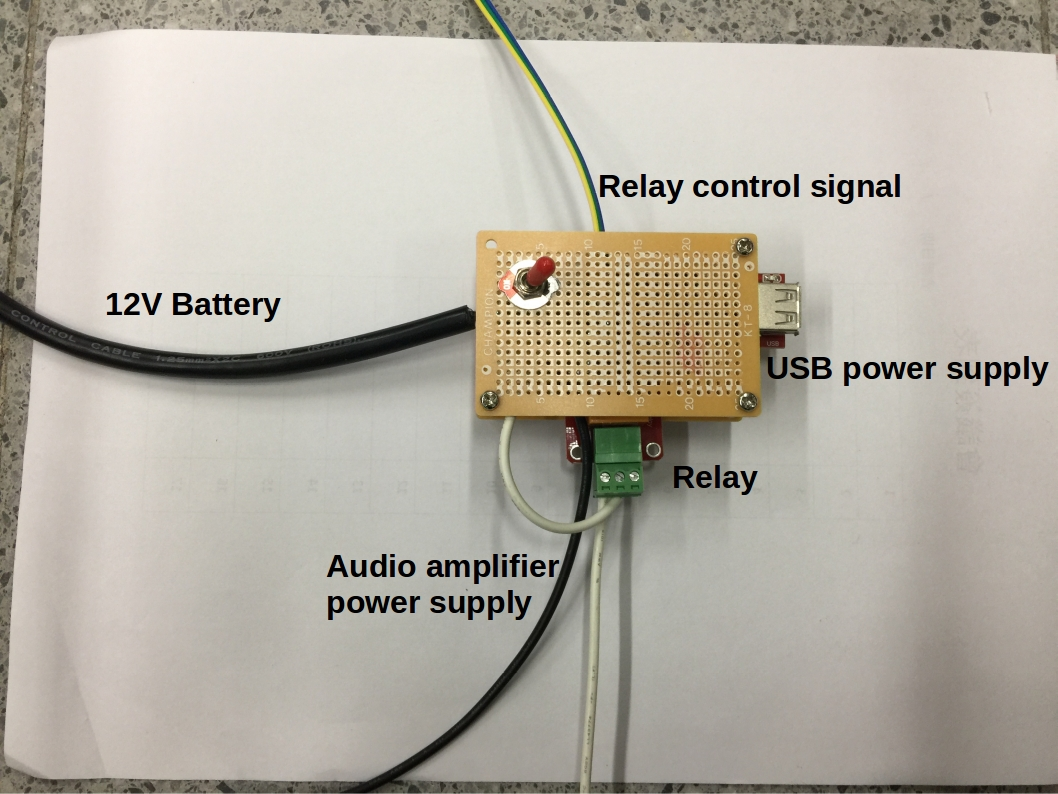


Fig. Relay Control Board

Fig. A set of Underwater Pinger(acoustic source) hardware

How to run

1. Make sure your Raspberry Pi in Pinger system is connected to the shore side Wi-Fi hotspot, as well as your laptop.
2. Login your raspberry pi via SSH command.
3. Follow the [steps](#_rhtwi3gigp91) to git clone the repo.
4. Follow the instructions shown below. We will launch Pinger control node on your raspberry PI and trigger the sound by the laptop via ROS service(std\_srv**(Recommend: Use “byobu” terminal)**

| $ cd robotx-underwater-task  $ source docker\_run\_rpi.sh  --------------------- In the docker container ---------------------  $ catkin\_make -C catkin\_ws # Compile  $ export ROS\_IP=[**YOUR\_RPI\_IP**]  $ export ROS\_MASTER\_URI=http://**[ROS\_MASTER\_IP]**:11311  $ source catkin\_ws/devel/setup.sh  $ roslaunch acoustic\_source\_control **[pinger1.launch | pinger2.launch | pinger3.launch]**  --------------------- On your laptop (after [git clone the main repo](#_rhtwi3gigp91)) ---------------------  $ cd robotx-underwater-task  $ source docker\_run\_laptop.sh  $ export ROS\_IP=**[YOUR\_LAPTOP\_IP]**  $ export ROS\_MASTER\_URI=http://**[ROS\_MASTER\_IP]**:11311  $ source catkin\_ws/devel/setup.sh  **# The following command will trigger pinger to make sine wave sound.**  $ rosservice call /pinger1 "{}" # Your can change the pinger number |
| --- |

### Topic/Activity 2 Acoustic recording system

The acoustic recording system include a audio recorder, aluminum support frame, two set of hydrophone hardware and computation unit. For fitting in the RobotX underwater task (25k - 40kHz of acoustic source frequency). The sampling frequency of the audio recorder must be higher than 100kHz, and the audio recorder must support **ALSA driver** that allow people to use Linux systems to control the recorder. We used Focusrite Scarlett series audio recorder which supports both features described above in RobotX Challenge 2018.

For the computation unit, here we prepare only an amd64 architecture docker image in this tutorial. Actually we used TX2 (arm64) to estimate the direction of acoustic sources during the competition.

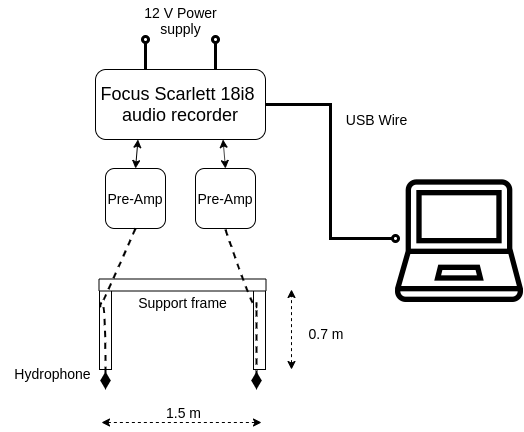


Fig. Acoustic recording system architecture

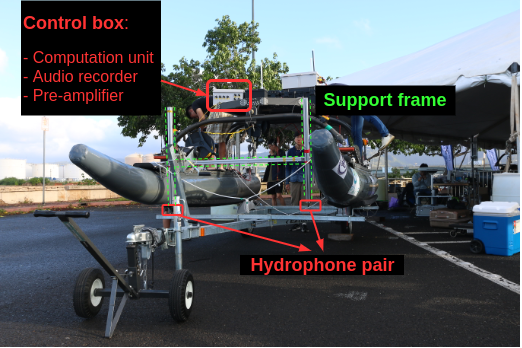
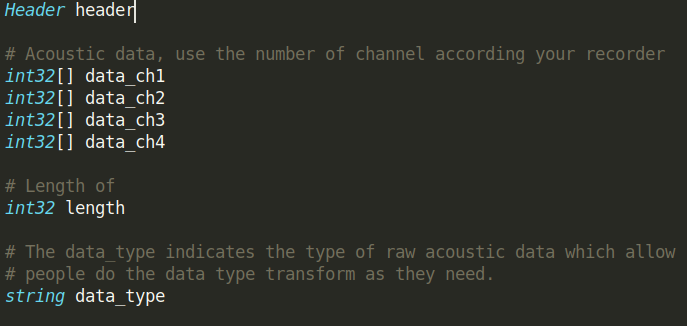


Fig. Acoustic recording hardware

The following instructions allow you to sample the hydrophone data with 2-channel track and with **sampling frequency 192kHz** as default, then publish the custom ROS topic called **/hydrophone\_data** with 0.5Hz frequency. **So you will get 96000×(2 channel) samples data every 0.5 second** if there is no data transmission or networking delay. The ROS message (robotx\_msgs/HydrophoneData) type is shown below.



| $ cd robotx-underwater-task  $ source docker\_run\_laptop.sh  --------------------- In the docker container ---------------------  # Don’t worry about that the warning message pop up when you launch the docker container  $ export ROS\_IP=**[YOUR\_LAPTOP\_IP]**  $ export ROS\_MASTER\_URI=http://**[ROS\_MASTER\_IP]**:11311  $ source catkin\_ws/devel/setup.sh  **# According your recorder product series, set the device parameter, the 18i8 is default.**  **# Use command line “arecord -l” to check hardware pcm ID, e.g. pcm\_id:=hw2,0**  $ roslaunch acoustic\_sampling sampling.launch device:=**[18i8 | 2i2]** pcm\_id:=[**PCM\_ID**]  **# Save to CSV**  $ rosrun acoustic\_sampling save\_to\_csv.py  **# Realtime data visualization**  $ rosrun acoustic\_sampling data\_visualization.py  **# Or use Matlab to visualize data from .csv file** |
| --- |

### Topic/Activity 3 Realtime data visualization

192k sampling frequency

Time domain data is shown on left (Normailzed to -1 ~ 1)

Freq domain data is shown on right

rosrun acoustic\_sampling data\_visualization.py

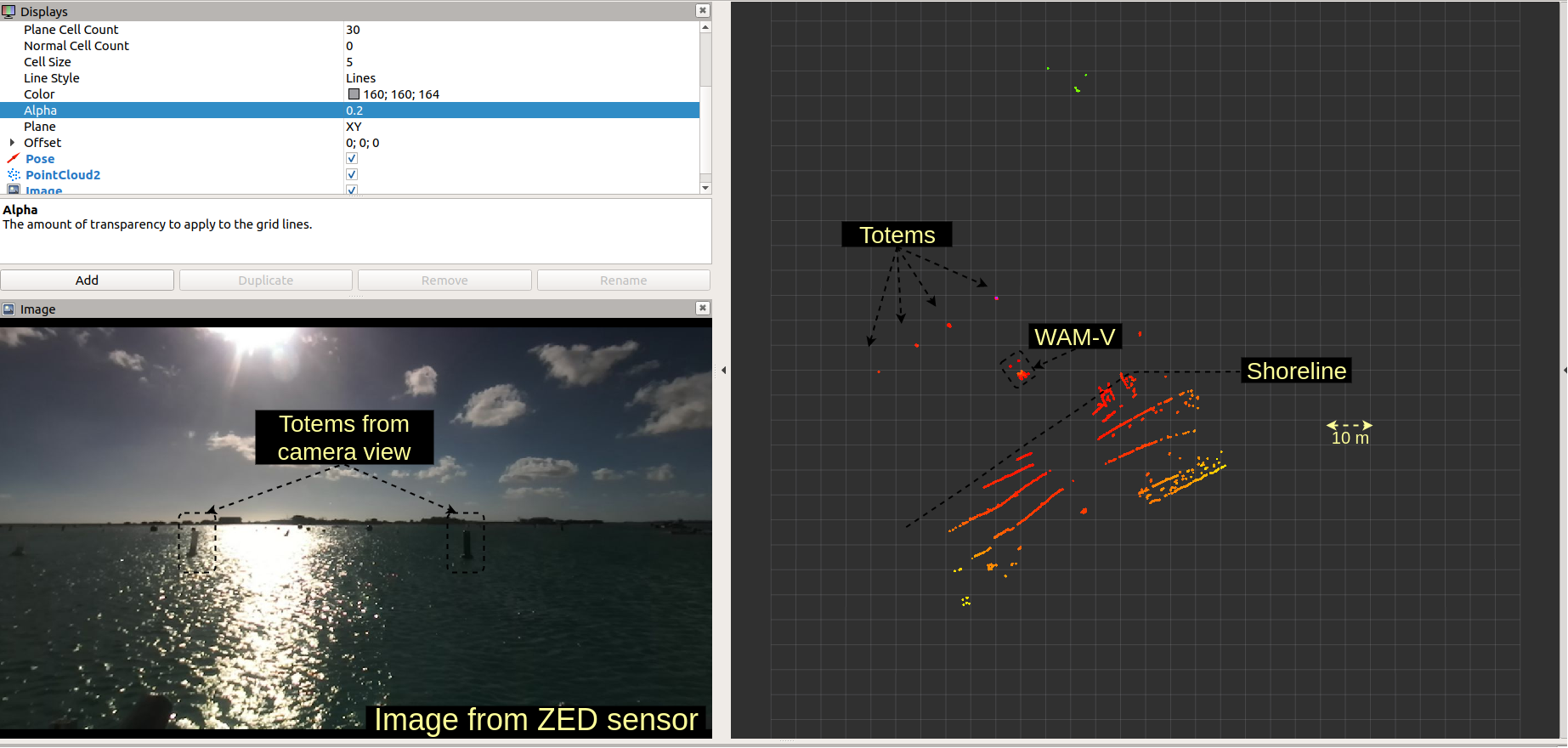
# This program will subscribe **/hydrophone\_data** and update the figures every 0.5 second

# The length of time domain data is 192k samples = samples within a second

### 

### Topic/Activity 4 Sound analysis

In this section, we just introduce the instructions to apply DOA algorithm on the acoustic topic. In addition, this section will allow you to download the rosbag and try to run the rosbag section which has been recorded in Hawaii. If you want to learn more about TDOA or the parameters of this package, please send the Email to [Sam Liu](mailto:c52647.ece07g@nctu.edu.tw), who has been kicked out from ARG Lab by NW. The rosbag includes point cloud data from Velodyne VLP-32 LiDAR, two images topic from different field of views and two channels acoustic data from hydrophone system.



Analysis data from audio recording system if you are using recorder then continue from Topic 2:

| ------------- New terminal Terminal 1 ( in the robotx-underwater-task directory) --------------  $ source docker\_run\_laptop.sh **same**  $ export ROS\_IP=**[YOUR\_LAPTOP\_IP]**  $ export ROS\_MASTER\_URI=http://**[ROS\_MASTER\_IP]**:11311  $ source catkin\_ws/devel/setup.sh  $ roslaunch doa\_estimation tdoa\_v2.launch # launch tdoa estimation node  # It will publish /guess\_angle(std\_msgs) if the acoustic data samples has been collected enough |
| --- |

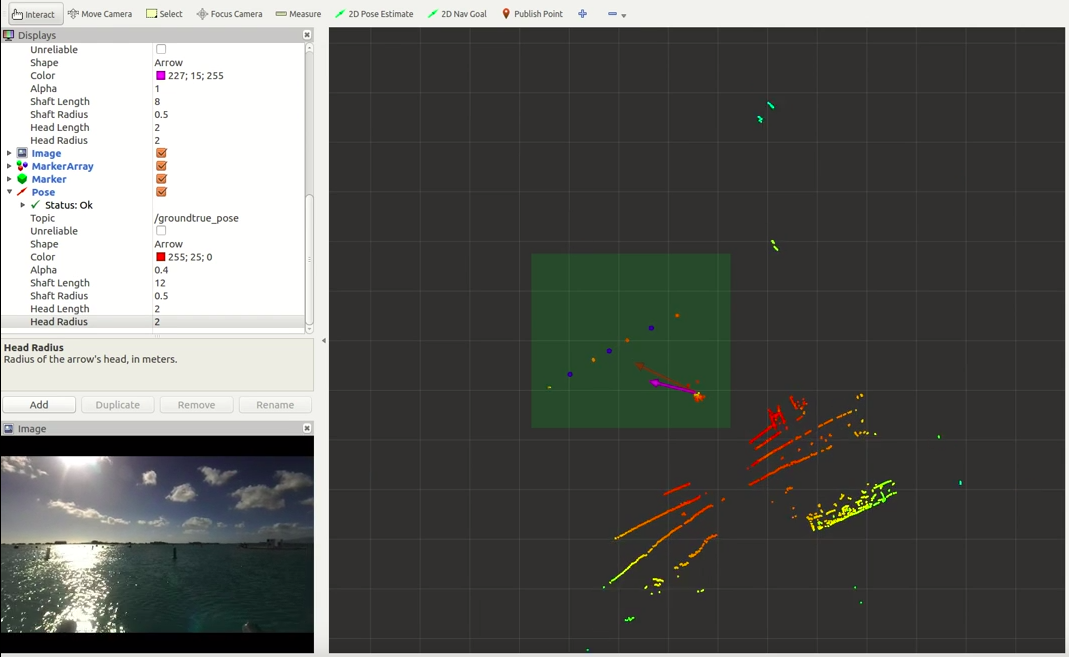
Analysis data from rosbag:

1. Download the acoustic [rosbag](https://drive.google.com/open?id=1C_8Mye66iDj1ELFruyMOyQV_dcM0sa-w) to ***robotx-underwater-task/bags*** folder
2. Follow the instructions shown below.

| --------------------- Terminal 1 ---------------------  $ cd robotx-underwater-task  $ source docker\_run\_laptop.sh  $ roscore  ----------- Terminal 2 (make sure you are in the robotx-underwater-task directory) -----------  $ source docker\_run\_laptop.sh **same**  $ source catkin\_ws/devel/setup.sh  $ roslaunch doa\_estimation tdoa\_v2.launch # launch tdoa estimation node  # It will publish /guess\_angle(std\_msgs) if the acoustic data samples has been collected enough  ----------- Terminal 3 (make sure you are in the robotx-underwater-task directory) -----------  # play the rosbag  $ source docker\_run\_laptop.sh **same**  $ rosbag play bags/entrance\_and\_exit\_sound\_2018-12-14-09-12-19\_1.bag |
| --- |

### DEMO

This demonstration use the same as rosbag from Hawaii as the one I shown you above. Externally, I added some visualization method to complete the [video](https://drive.google.com/open?id=1OIDGfENPX_Glo8RuUonfMF1uIzAvnG8-).



### Trouble shooting

### Reference

* C-T. Hung, S. Liu, Y. Huang, C. Chan and H. Wang, "Localization of an Underwater Beacon in Task2 of Maritime RobotX," *2019 IEEE Underwater Technology (UT)*, Kaohsiung, Taiwan, 2019, pp. 1-4. doi: 10.1109/UT.2019.8734328