NERC Marine Renewable Energy Knowledge Exchange Program

Appendix 2: Software Instructions

How to run the PLABuoy electronics and analyse collected data

Jamie Macaulay*[1], Jonathan Gordon[1], Doug Gillespie[1], Chloe Malinka[1], Mark Johnson[1], Simon Northridge[1]

[1]Sea Mammal Research Unit, Scottish Oceans Institute, University of St Andrews, St Andrews, Fife, KY16 8LB, UK

*jdjm@st-andrews.ac.uk

1 Introduction

Here we describe two sets of software. The first (sections 2 - 4) are programs related to the collection of data in the field. This primarily focuses on how to set up the cRio DAQ system to record multi-channel WAV files. The second (section 5) is a tutorial on to the use of PAMGUARD routines for post processing multi-channel recordings and to determine georeferenced 3D tracks of odontocetes.

2 Field Software

2.1 Setting up the cRio

The National Instruments cRio computer forms the core of the PLABuoy's recording system. Two groups of software are required to set up the cRio. These are;

- **NI-RIO device drivers and NI DAQMX** – allows users to communicate with cRio, change IP addresses and install software on device. The NI utility program DAQmx also contains useful software and we recommend installing it.
- **PuTTy and/or Eclipse with Remote Systems Explorer add on** – both programs can be used to access the NI Real Time Linux OS on the device, navigate through directories and send/receive commands. PuTTy is simple and reliable and hence we recommend its use here.

Download software from relevant websites and install on a computer.

- PuTTy: http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html
- Eclipse: http://www.ni.com/download/labview-real-time-module-2014/4846/en/
- cRio Drivers: http://www.ni.com/download/ni-rio-14.5/5129/en/
- NIDAQMX:http://www.ni.com/nisearch/app/main/p/bot/no/ap/tech/lang/en/pg/1/sn/catnav:du,n8:3478.41.181.5495,ssnav:ndr/

After the NI-RIO drivers are installed, install NI Linux Real Time OS and set up the cRio. Comprehensive instructions can be found on the National Instruments website.

https://www.ni.com/getting-started/set-up-hardware/compactrio/controller-software

A good forum for any issues and also a source for comprehensive tutorials on many aspects of NI Linux Real Time is at:

https://decibel.ni.com/content/groups/ni-linux-real-time?view=documents

Once the Linux Real Time OS has been installed on the cRio it can be configured to acquire data through one or more NI 9222 cards.

Several files need to be added and code compiled for the FPGA and a daemon must initiated in order to configure the cRio for recording. Note: NI 9222 modules should be installed onto the chassis by this point.

There is an online repository within source forge which currently holds all programs related to the cRio. This is located at: https://sourceforge.net/projects/plabuoy/.

In the online repository for the PLABuoy there is a zip name 'cRio DAQ' which contains four files;

- **cRioDAQC** - the compiled C code for the PLABuoy which handles the recording, communication and watchdog processes involved in the cRio.
- **cRioDAQSettings.txt** - the settings file which allows users to change basic settings on the cRio. (Note this is currently not used but maybe utilised in newer version of cRioDAQC)
- **cRioDaemonScript** – a shell script which automatically runs cRioDAQC when the cRio is started and redirects output messages to a file.
- **libsndfile.a** – a library for recording .wav files (originally from http://www.mega-nerd.com/libsndfile/)
- **.lvbitx** – two compiled programs which run on the cRio's FPGA. Note each is for a different model, one for a cRio 9068 and the other for a cRio 9067.

Create a directory named **/home/cRioKE** and copy all the files listed above into it.

(Several programs can be used for accessing the Linux Real Time OS on the cRio. PuTTy allows remote terminal access and Eclipse remote systems explorer provides an easy to use GUI. Both are freely available online. They require the IP address of the device which can be found using NIMAX software.)

The cRio is now ready to run the cRioDAQC program; this will record acoustic and GPS data to a hard drive and allow users to interact with the system through a tablet or laptop.

To begin, simply navigate to the folder containing the program e.g. in the console enter **cd cRioKE.** Enter **ls** to check you are in the correct directory.

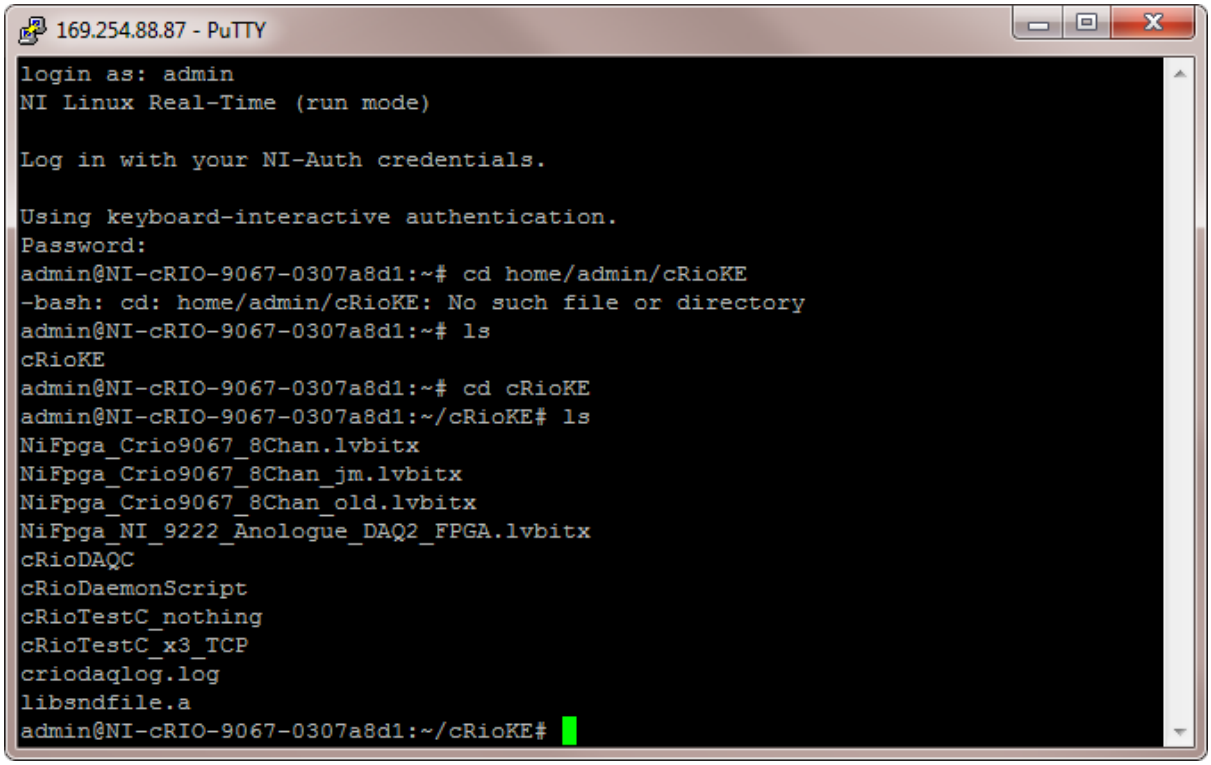


Figure 1. Using PuTTy to log into the cRio Linux Real Time OS and show contents of the folder containing code to run the cRioDAQC program.

Then type**. /cRioDAQC** to start the cRioDAQC program. The program will ask for a command.

Type **help** to see a list of commands. (Commands can also be sent to the cRio through its network ports.)

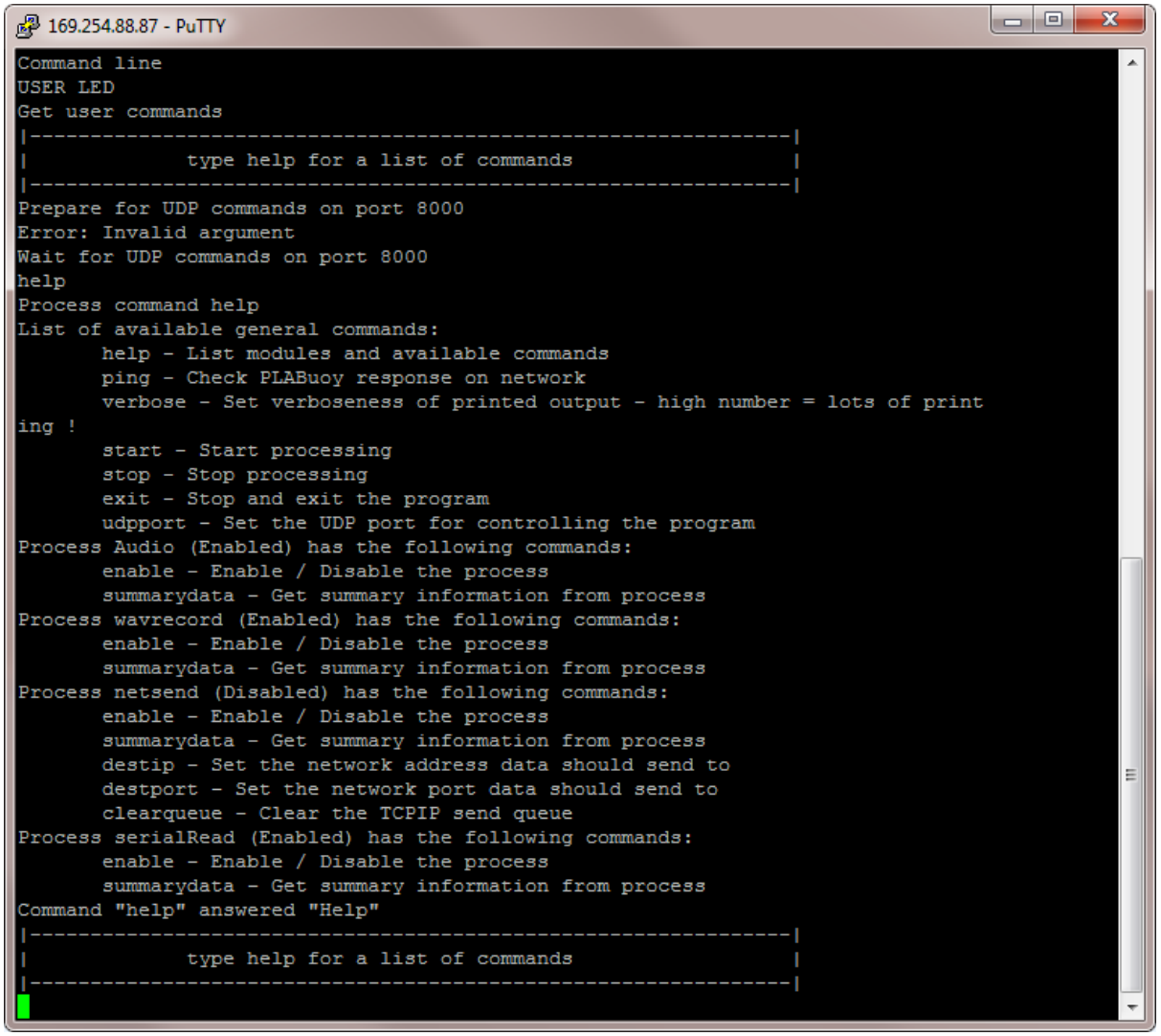


Figure 2. The cRioDAQC program can be controlled through a console. Commands can also be sent over a network, *i.e.* by using PLABuoyInterface software.

To start recording simply type **start.**

You should see frequent updates begin to scroll down the screen. Error messages indicate an issue, such as the hard drive not being plugged in or the occurrence of FPGA faults.

Note: a good indication the device is working is to check that the hard drive light (if one exists) is blinking *i.e.* data are being written to the drive.

2.2 Setting up the runtime Daemon

When using the PLABuoy in the field, it is impractical to start a C binary from a console each time the cRio is switched on and off. Thus, it is necessary to run the C code (cRioDAQC) as a Daemon, i.e. a program that automatically starts when the cRio starts. To do this it is necessary to install **cRioDaemonScript** (in folder **/home/cRioKE**) as a start-up script. Log into the cRio using PuTTy or equivalent and perform the following commands.

1) Navigate to the directory containing the start-up script, e.g. **cd cRioKE**. Check the start-up script is actually there using **ls.**
2) Make the start-up script executable **chmod +x cRioDaemonScript** and check it works by typing **/home/cRioKE/cRioDaemonScript start.** The program should start outputting to a log file in the same folder.

   To stop enter **/home/cRioKE/cRioDaemonScript stop.** A message should appear confirming the application has stopped.

   (If located in a folder other than /home/cRioKE or the script name is altered then substitute se the directory the script is located in and/or appropriate script name *e.g.* **Directory/script_name start)**
3) The next stage is to install the script. The script should be installed so that it is nearly the last thing to start on cRio start up.

   Enter **cp cRioDaemonScript /etc/init.d** to copy the script to the script directory. Next enter **/usr/sbin/update-rc.d -f cRioDaemonScript defaults 99 0.** The script is now installed and will start cRioDAQC on start up.

*Tip. Occasionally you may want to start the cRio and stop the script from running. To do this enter***/etc/init.d/cRioDaemonScript stop.** *To uninstall the script completely use* **/usr/sbin/update-rc.d -f cRioDaemonScript remove.** *The NI website contains a general tutorial on installing start up scripts 'Installing Startup Scripts on NI Linux Real-Time' (https://decibel.ni.com/content/docs/DOC-38960)*

The cRio is now ready for deployment. Note that the cRioDAQC program does not start recording automatically, it simply starts and waits for commands. To initiate recording you need to either send a '*start'* command to the program or through a network to the cRio. The PLABuoyInterface program will do this automatically.

3 Setting up the cRio Development Environment

In order to alter the source code on the cRio two programs are required

- **National Instruments LabView and Labview FPGA module** – a graphical programming language which allows users to compile code for the FPGA.
- **Eclipse**. A free and open source IDE which can be used to alter the C/C++ source code of cRioDAQC program.

3.1 LabView: Programming the FPGA

The cRio contains an FPGA chip, a specialised processor for signal processing. FGPA chips are ideal for the extremely fast digital signal processing tasks, such as recording multiple channels of high frequency acoustic data or calculating FFTs. The FPGA in the cRio communicates directly with the DAQ modules and then passes data to programs running on the ARM chip.

In the case of the PLABuoy the FPGA needs to request data from the NI9222 cards, package samples into an appropriate format and send to a FIFO buffer which can be accessed from a the cRioDAQC software running on the Linux OS. It also reads temperature sensors and controls LEDs.

National Instruments have created LabView FPGA, a high level design tool to program the cRio's FPGA. This allows for relatively inexperienced users to program the FPGA. As part of the PLABuoy project we have created a LabView FPGA program which records data from two NI9222 cards. Operators can use this without a LabView licence. This guide demonstrates how to set up a LabView system to compile FPGA code for a cRio for those needing a different configuration. It should be noted that we have used cRio 9068 and 9067 models as development machines. Other cRio models are very similar but LabView code may need to be slightly altered to run on them.

3.2 Setting up the FPGA

The online code repository for the PLABuoy contains a folder named 'LabView FPGA'. Inside this are several files, one of which is a LabView project **cRio_Project.lvproj**. Open this in LabView FPGA.

The project contains two potential targets (cRio devices), a cRio 9068 and cRio 9067. You can add additional targets/cRio models easily (Figure 3).

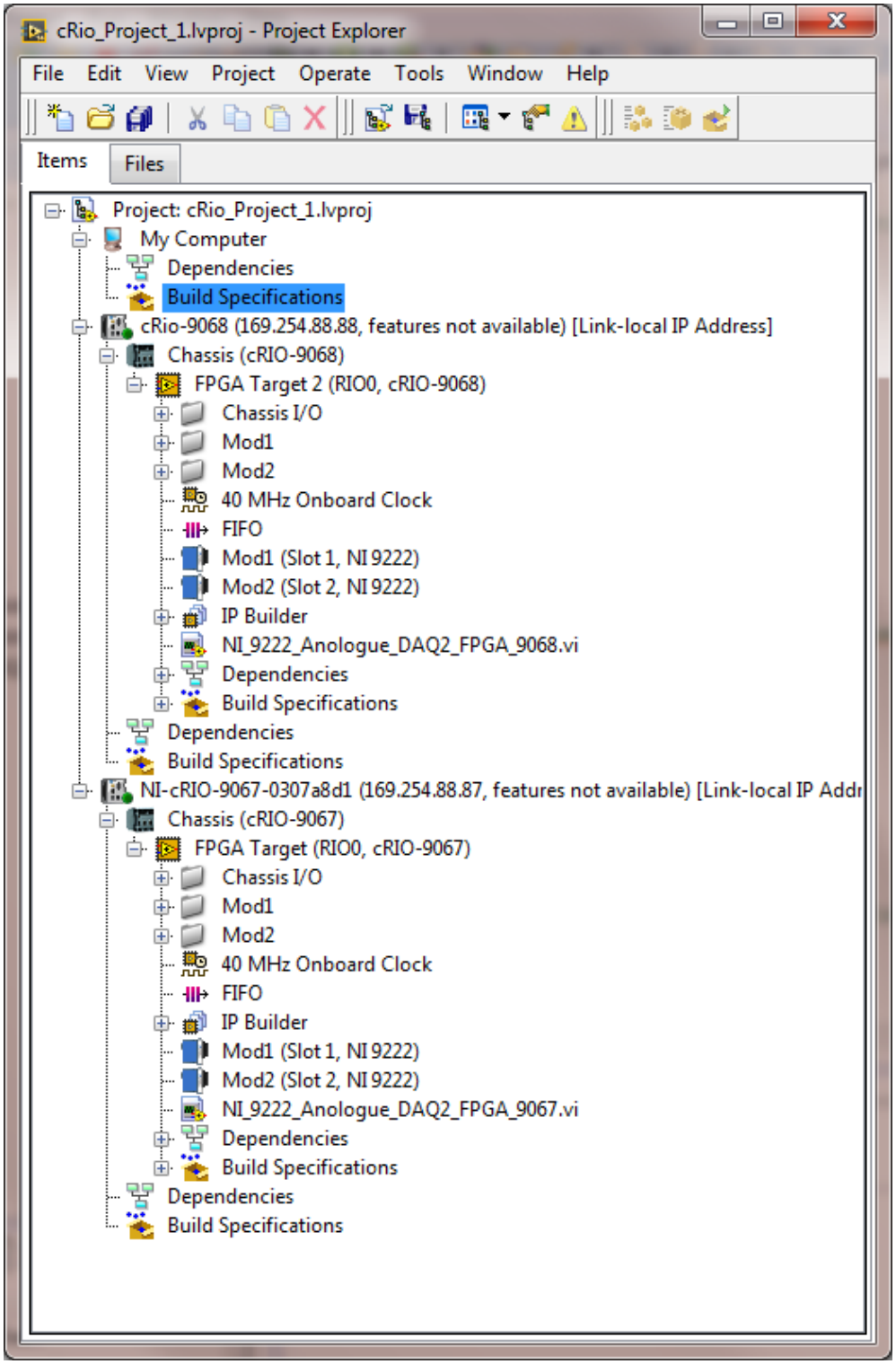


Figure 3. A Labview FPGA project containing two targets, a cRio 9067 and cRio 9068.

Within each target is a .vi file (e.g. NI_9222_Anologue_DAQ2_FPGA_9067.vi) which contains the code that runs on the FPGA. The code is very basic. It simply acquires measurements from the NI9222 DAQ cards at a specified sample rate, combines the measurements into an array and sends this to a FIFO buffer. This FIFO buffer can then be accessed by code running on the cRio's ARM processor. Opening the .vi file will bring up the LabView code used to program the FPGA as shown in Figure 4.

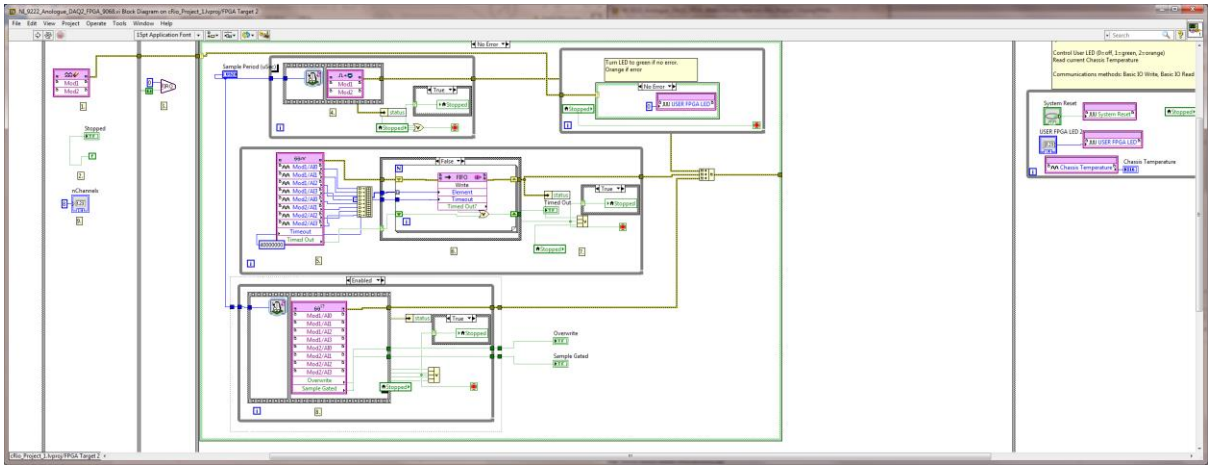


Figure 4. Example of the block diagram for the cRio's FPGA.

Different cRio devices may require slight alterations to the code. For example, the LEDs on the 9068 and 9067 require slightly different inputs to change colour. Therefore a separate .vi is provided for the cRio 9067 and 9068, although the core code is almost identical. If using a model other than 9067/8 the .vi may require slight alterations; however the main body of code will remain much the same. A good explanation of how the general FPGA code works can be found at http://www.ni.com/tutorial/14532/en/ (31/08/2015).

To compile code simply click the arrow button (Figure 5). You will then have the choice to use a local compile server or the National Instruments cloud based system (Note, you will need have installed Xilinx compilation tools from NI to use a local compile server. Otherwise a subscription for NI cloud based services will be required). Choose your preference and click OK. The compile process will then begin (Figure 5).

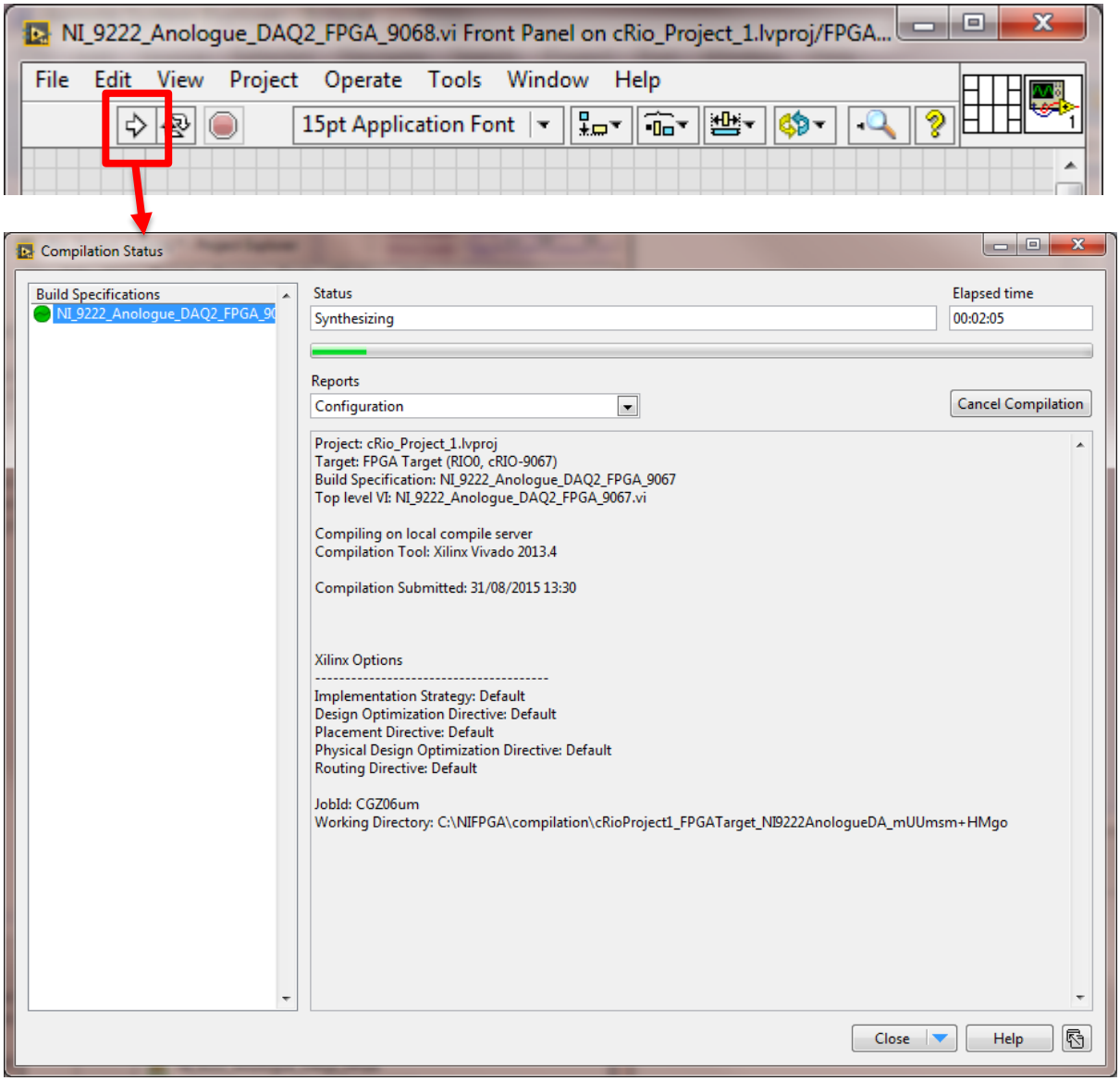


Figure 5. Compiling LabView code for the FPGA. This can take upwards of 30 minutes.

During the compile process a .lvbitx file will be generated You can view the file by expanding the ***Build Specifications*** section in the project. The .lvbitx file is essentially the compiled program for the FPGA and is required for the cRio to run cRioDAQC. For the FPGA to interact with the compiled C program a C library is also required. This is generated by simply right clicking on NI_9222_Anologue_DAQ2_FPGA_9067.vi and selecting ***Launch C API generator*** (Figure 6).

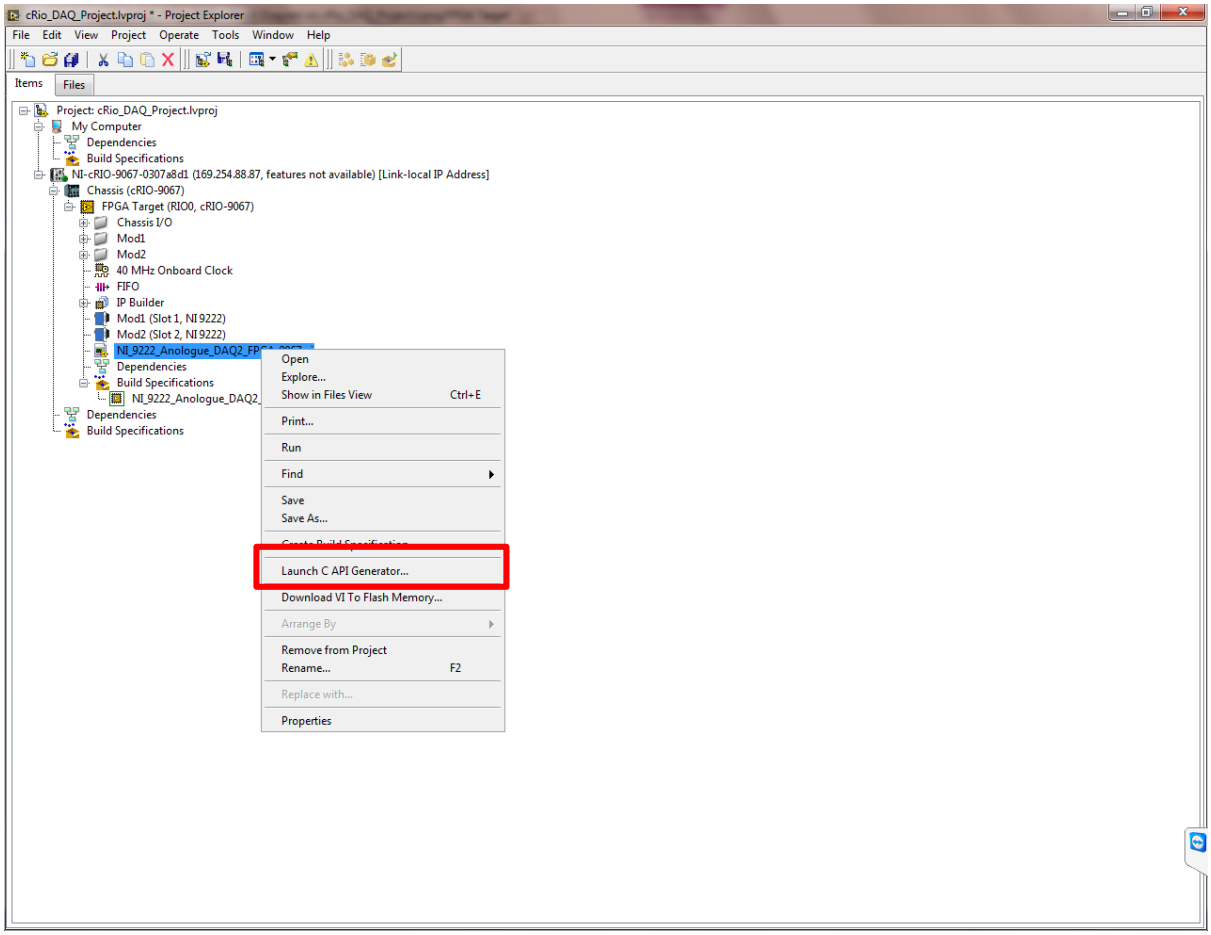


Figure 6. The LAbView software can generate user specific C libraries which interact with the Labview FPGA code.

The C API generator will generate 4 files. Ignore NiFpga.c and NiFpga.h. These have already been integrated into the C code in cRioDAQC and are identical for all generated C APIs. Open the remaining C header file e.g. NiFpga_NI_9222_Anologue_DAQ2_FPGA_9067.h. This should look something like Figure 7.

Tip: If using another cRio model and therefore a slightly different .vi you will need to hardcode this header file into the cRioDAQC program.

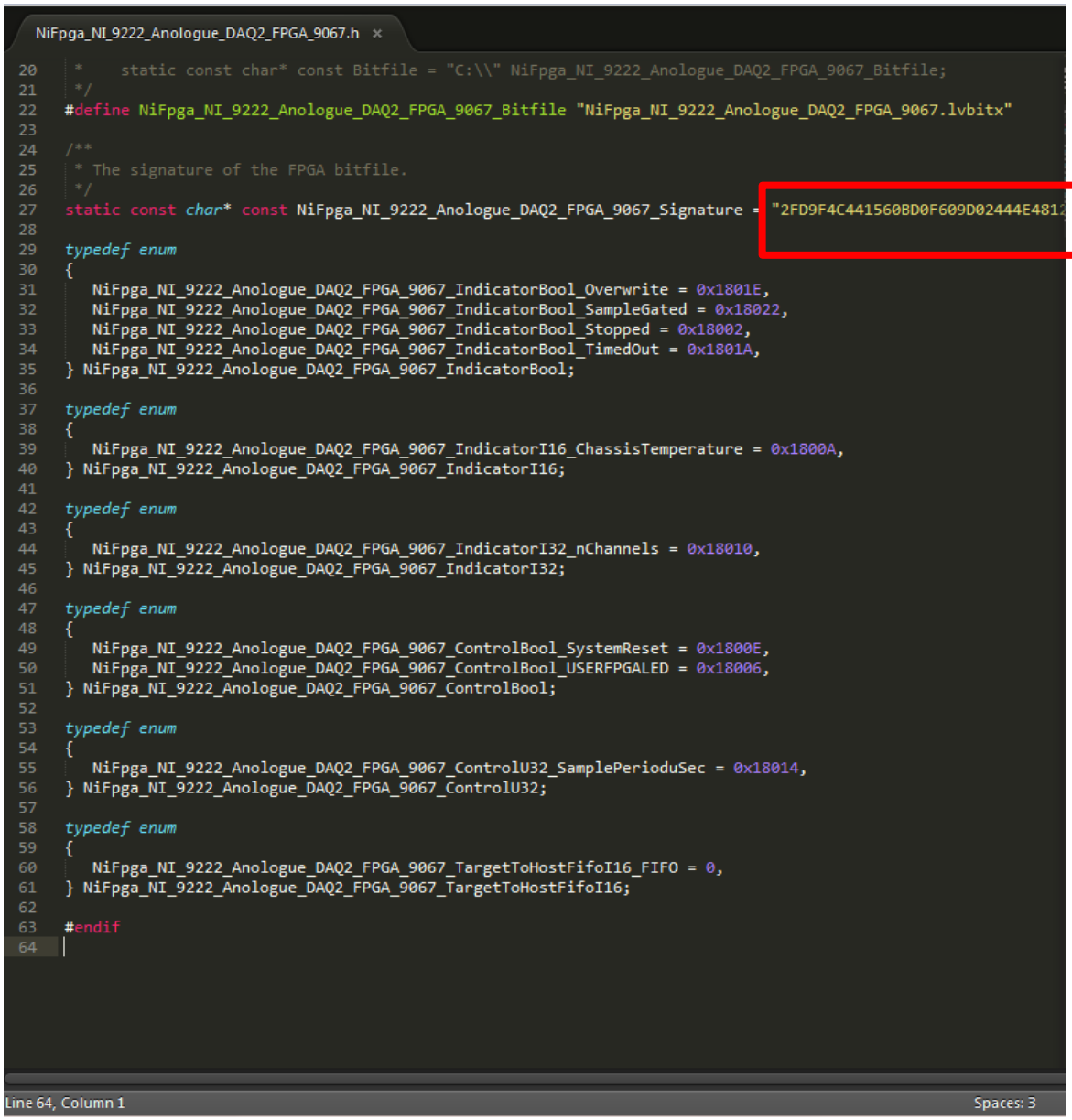


Figure 7. The generated C library header. A newly-compiled FPGA .lvbitx file will have a unique serial number which should be copied into the cRio's settings file.

The only item needed form this file is the FPGA signature, highlighted in red in Figure 7. Copy this into the settings file.

Remember to copy the .lvbitx file onto the cRio e.g. **/home/cRIoKE** and make sure its path corresponds to that in the cRio settings file.

3.3 Programming C/C++ for cRioDAQC

National Instruments maintain a bespoke version of Eclipse which can be used to develop C/C++ applications for the cRio. This can be downloaded from the National Instruments website. Once this has been downloaded a project for a cRio can be created by following a tutorial on the National Instruments website.

In order to access the most recent source code for the cRioDAQC project SVN tools must be added to eclipse and the project downloaded from the online repository.

The next section describes how to get a project up and running in Eclipse and then compile for a cRio target.

3.3.1 Download cRioDAQC from SVN

Install SVN tools.

In Eclipse go to **Help->Install New Software…**In the dialog which appears select the appropriate update site for eclipse e.g. **Kepler – http://download.eclipse.org/releases/kepler**. In the list of optional add-ons which appears, expand **Collaboration** and select all SVN tools (Figure 8).

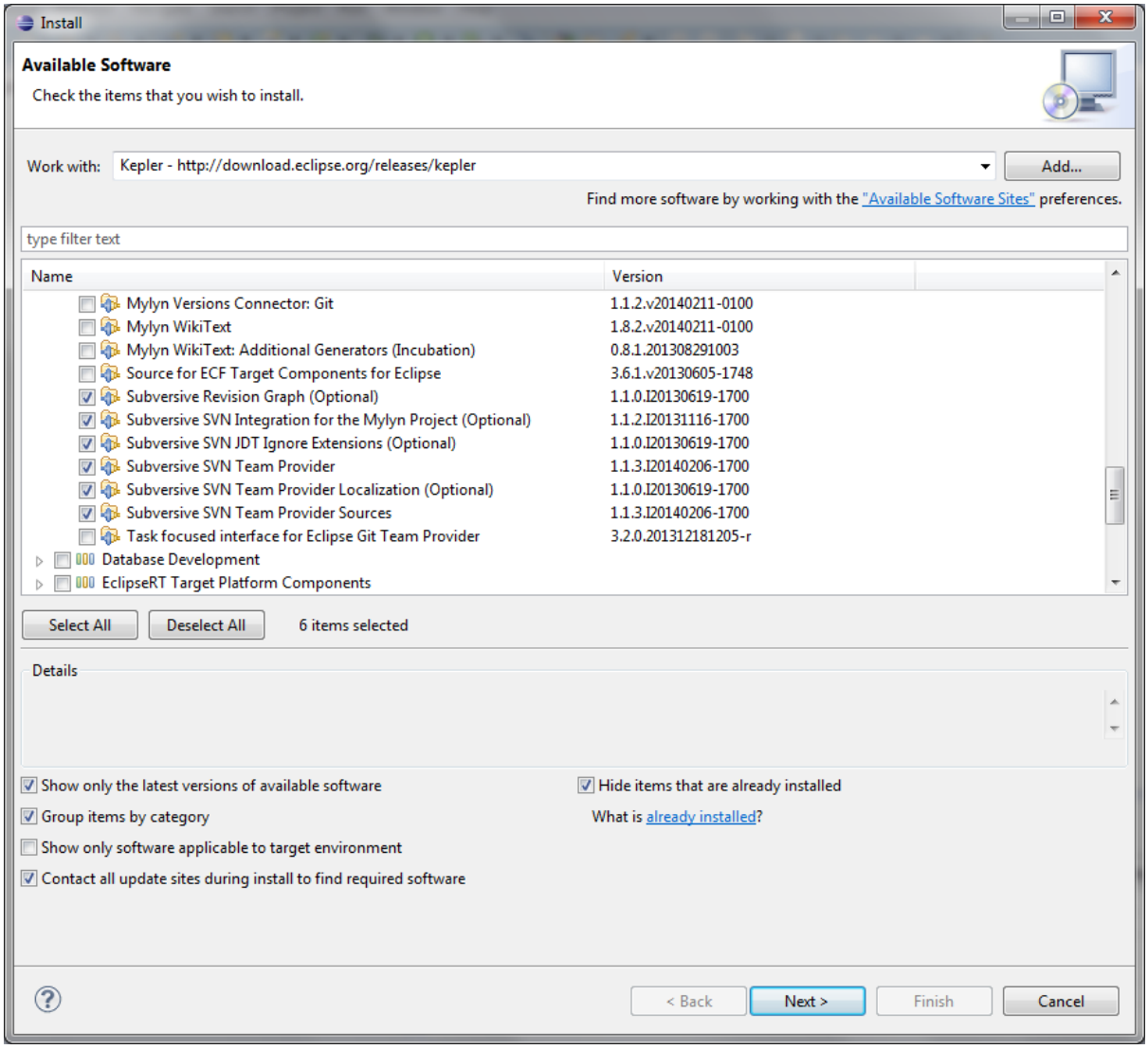


Figure 8. The Eclipse software installer. You need to download the Eclipse SVN software in order to download source code form the PLABuoy SVN repository.

Click **Next** and install the selected add-ons. Eclipse should prompt for a restart, if not restart Eclipse anyway.

Once eclipse has restarted exit the welcome screen. Enter the C/C++ view and right click on the Project Explorer panel and select **New->Other…** In the Wizard dialog expand SVN and select **Project from SVN** (Figure 9).

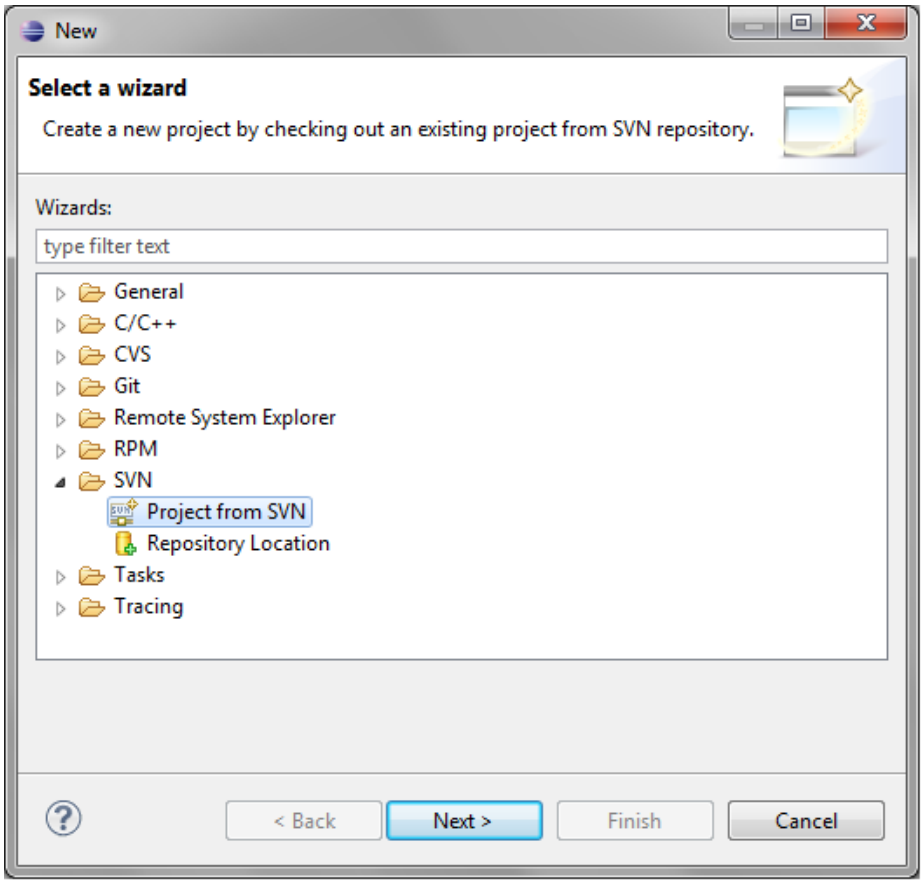


Figure 9. Once SVN tools have been installed create a new project from SVN.

SVN will then prompt further installation of software (Figure 10). Click through the installation process and Eclipse should restart again; if this does not occur manually restart Eclipse.

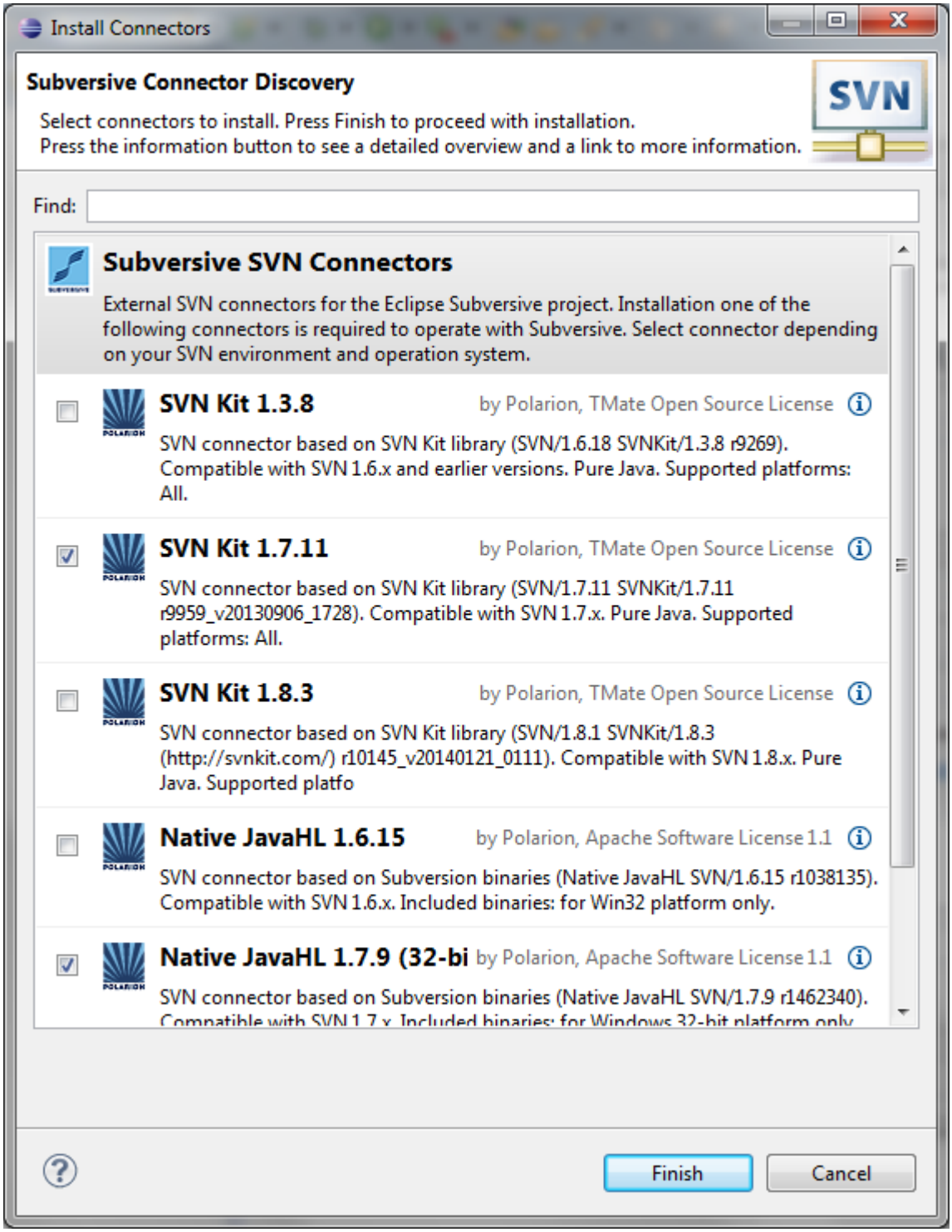


Figure 10. Installing SVN plugin for Eclipse. This allows users to download source code from the online PLABuoy repository.

Create a new SVN project again using the same process as in the previous step. This time all SVN should have been installed properly. Create a new repository location in Eclipse (Figure 11). The repository location is

- **svn+ssh://svn.code.sf.net/p/plabuoy/svn-code/cRio_Daq_cpp** (*Read/Write)*
- **svn://svn.code.sf.net/p/plabuoy/svn-code/cRio_Daq_cpp** (*Read only)*

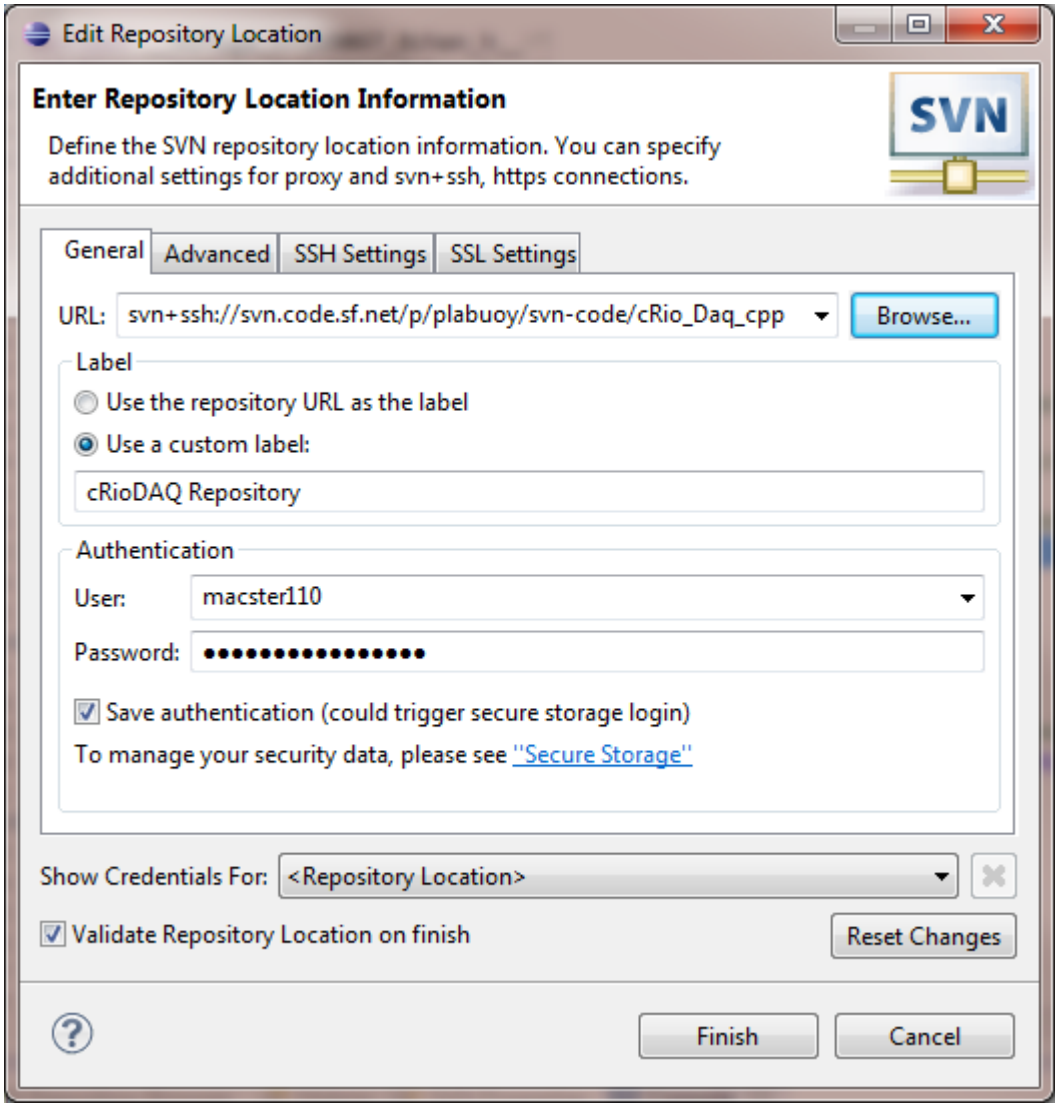


Figure 11. Setting the repository location in Eclipse to create a new project.

Click **Next**. Eclipse will validate the repository. (Note: For read and write access you must have a Source Forge account and be added as an administrator to the project. Please contact SMRU (jdjm@st-andrews.ac.uk) if these access privileges are required.)

Once validated the **Select Resource** dialog will appear (Figure 12).

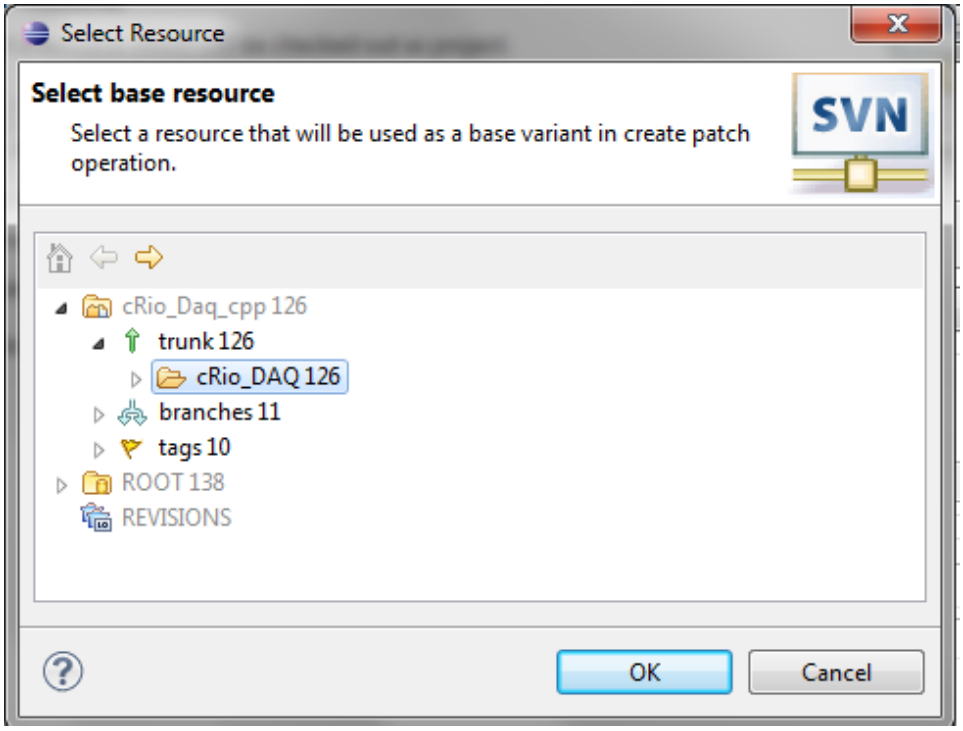


Figure 12. Selecting the correct folder to download C++ code for PLABuoy,

Click Browse, expand **cRio_Daq_cpp**, expand **trunk** and select the **cRio_DAQ** project. Click **OK** and then click **Finish**. In the next dialog choose an appropriate name for the project and click **Finish**. The project will now download.

3.3.2 Set up the C++ Build Environment

Now we need to change some build settings.

Right click the project and select **Properties.** This will bring up the project properties page.

The first thing to do is to connect up the relevant external libraries. In the project properties dialog expand **C/C++ Build** and select **Settings**. Under **Cross GCC Compiler** select **Includes** (Figure 13).

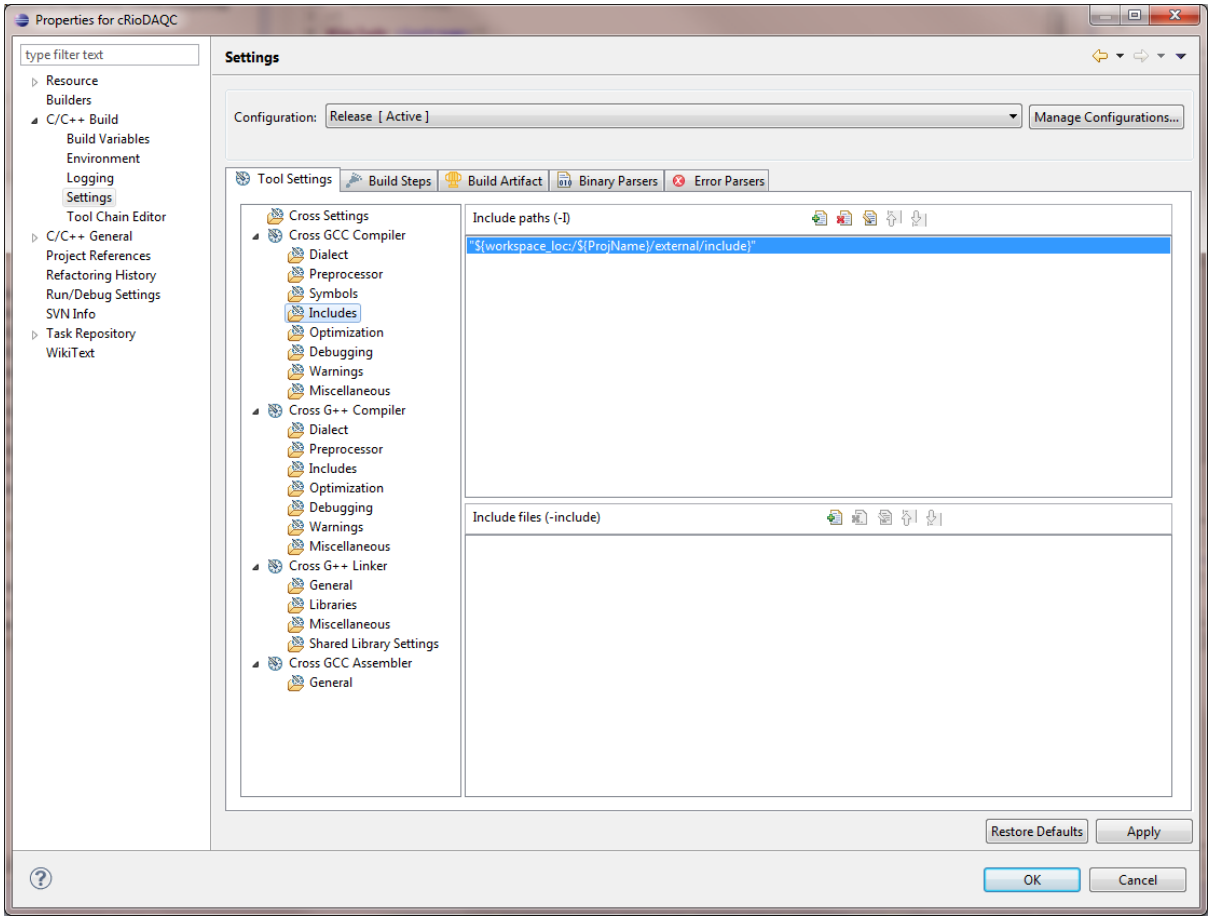


Figure 13. To get the C++ code to compile properly in Eclipse some build settings have to be changed, .

In the include paths (-I) table click the add button.

In the dialog select **Workspace…** and within the Folder selection dialog browse to **cRioDaqC/external/include.**

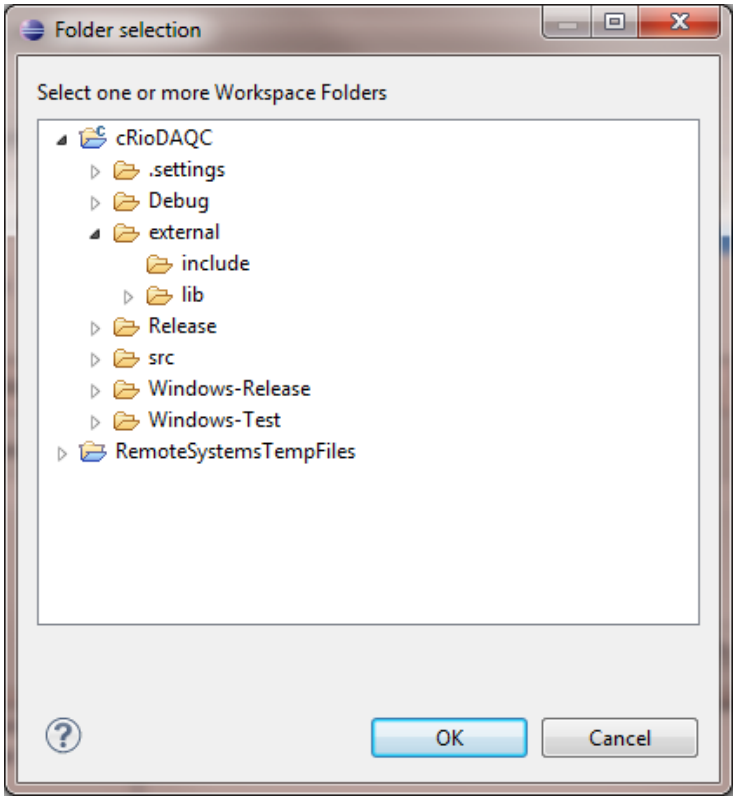


Figure 14. External libraries need to be linked to.

Click OK and close the dialog. Repeat the process for **Cross G++ Compiler.** Under **Cross G++ Linker** select **Libraries.** In the **Library search path (-L)** table click the add button, select Workspace… and in the Folder selection dialog navigate to **cRioDaqC/external/lib** (Figure 14)**.**

Next in **Cross GCC Compiler->Symbols** add **CRIO** to the **Defined Symbols (-D**) table and **WINDOWS** to the **Undefined Symbols (-U)** table (Figure 15). Do the same in **Cross G++ Compiler -**

>Preprocessor.

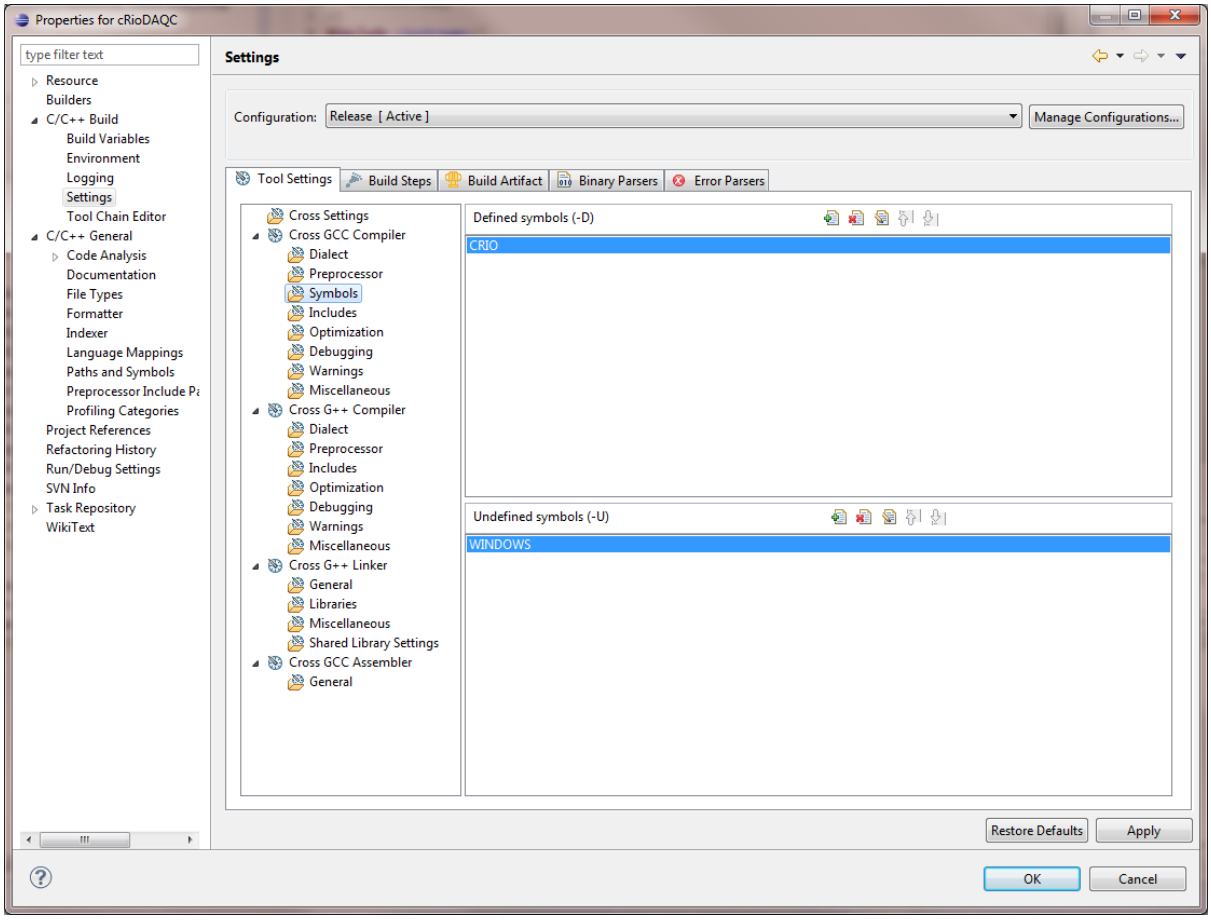


Figure 15. Make sure Cross GCC Compiler->Symbols settings looks like this screenshot.

Now tell the compiler to include libraries from the lib folder.

Go to **Cross G++ Linker** -> **Libraries**. In the **Library search path (-L)** table select the add button and add a new path to **cRioDAQC/external/lib**. In the **Libraries (-l)** table add **pthread** and **sndfile** if not already there (Figure 16).

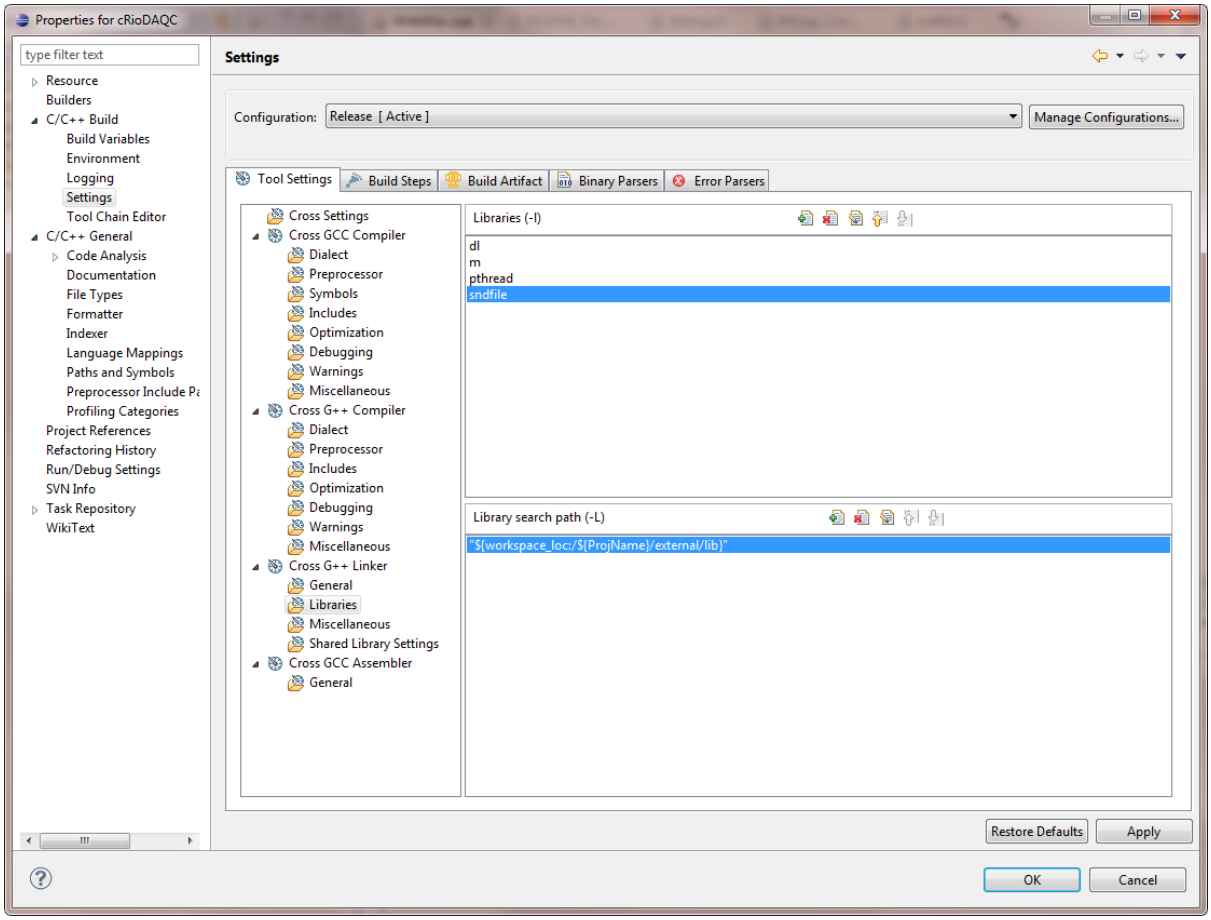


Figure 16. Make sure the compiler knows which external libraries to link to.

3.3.3 Connecting the cRio

Connect the cRio to the computer, switch on and open NIMAX. In NIMAX expand **Remote Systems** and select the cRio. Click the **Network Settings** tab and note the IP Address (Figure 17). Copy the IP address.

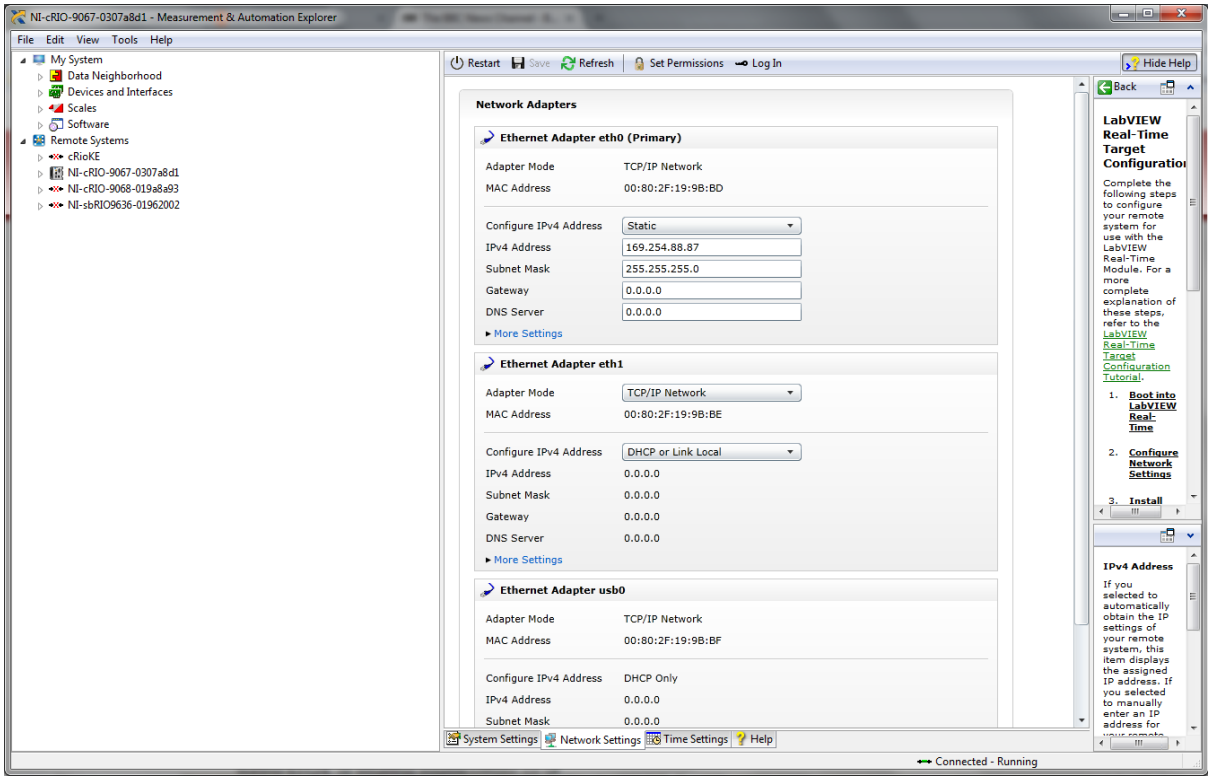


Figure 17. NI Max software allows you to chage the IP address settings of the cRio. It also has a host of handy features to communicate with cRio devices.

Next the cRio needs to be connected to Eclipse. Select the **Remote System Explorer** view and click **Define a connection to remote system (**Figure 18)

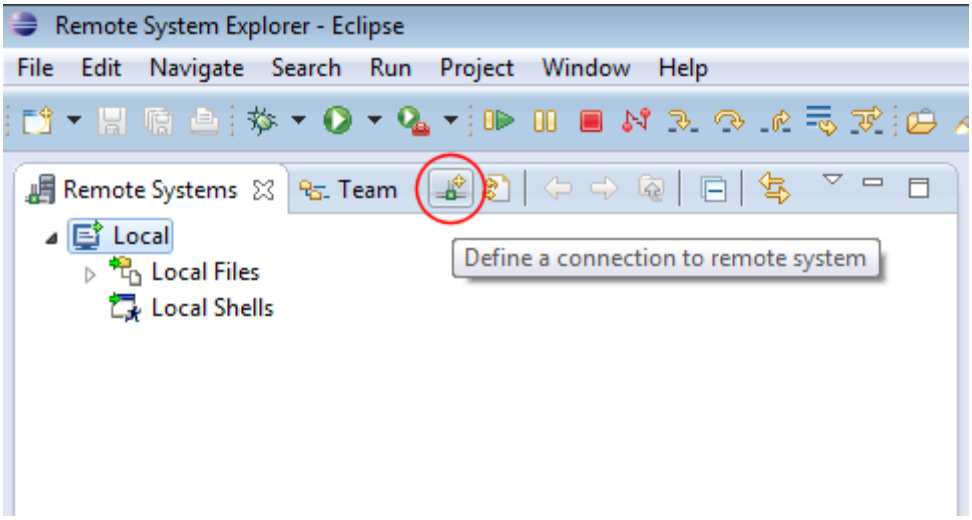


Figure 18. Eclipse can remotely connect to a cRio. You can use the eclipse remote system explorer to connect to difference devices.

In the dialog which appears select SSH only (Figure 19). Linux can also be selected but SSH shortens the setup process.

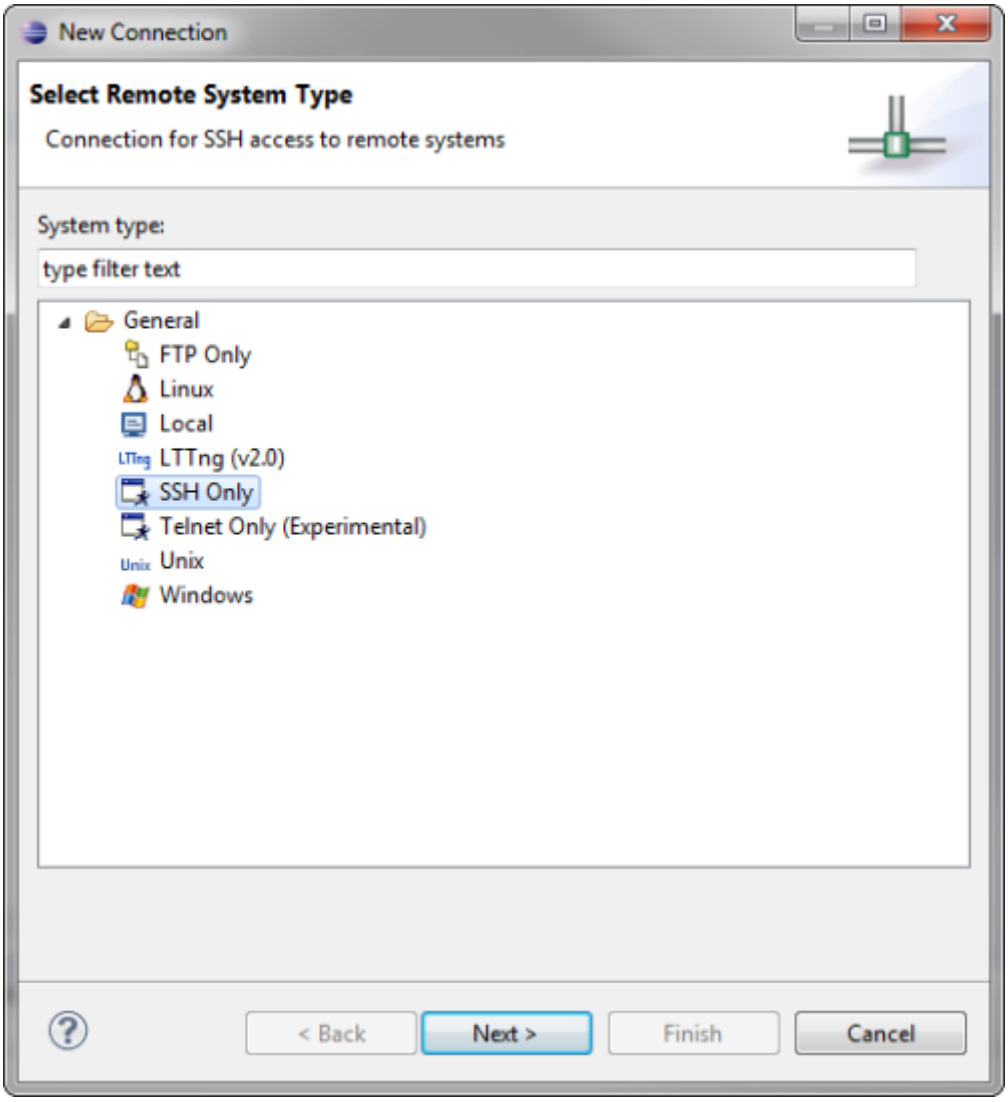


Figure 19. Connecting to a cRio. Select SSH Only. (You can also select Linux but this goes through some unnecessary steps)

In the new Connection Dialog copy the IP address into the **Host name** text field. Note that you can also use the hostname of the cRio instead of the IP address (Figure 20). Click **Finish.**

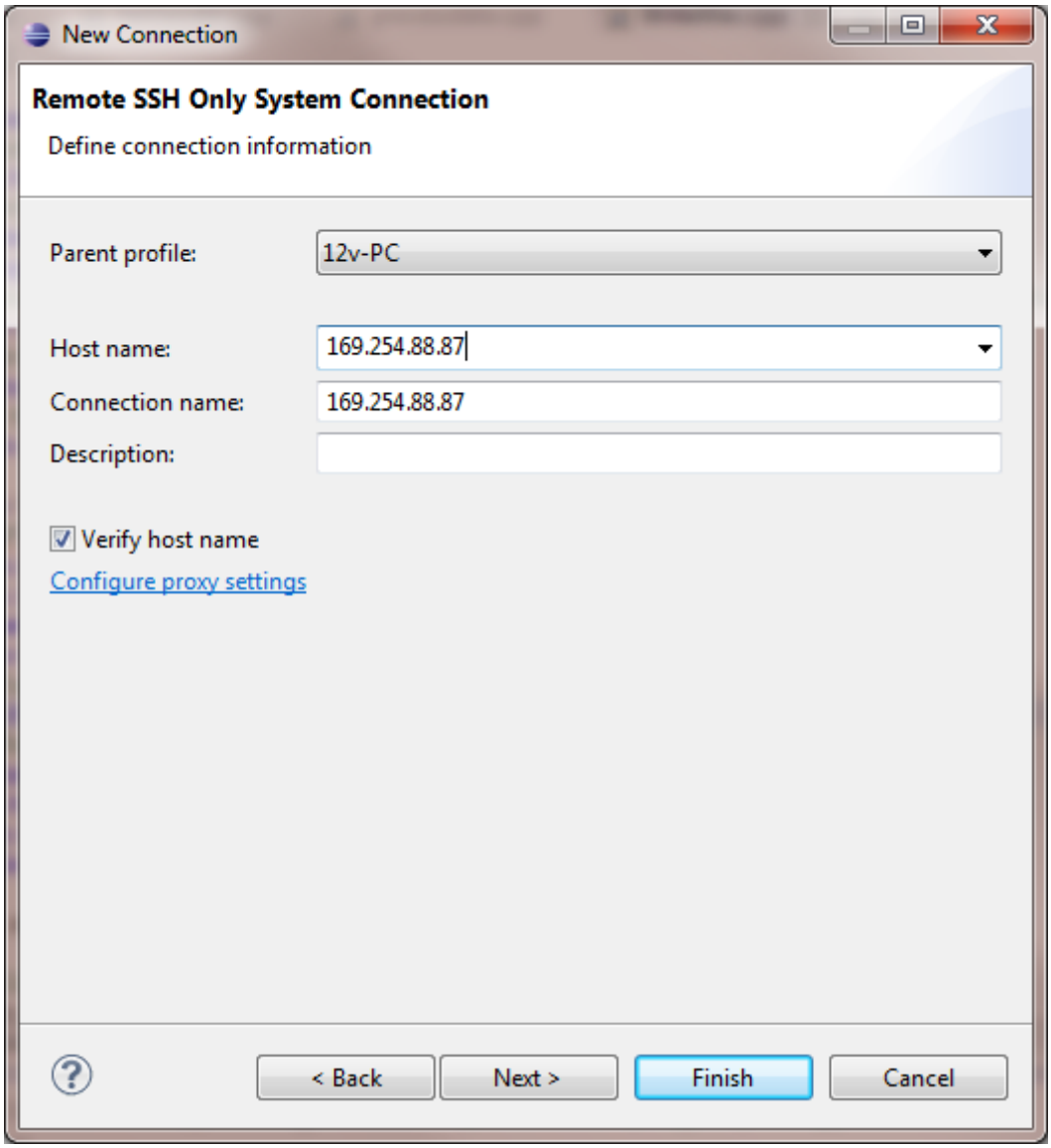


Figure 20. Eclipse needs to know the IP address of the cRio to connect remotely. The IP address can be easily found using NIMAX software.

In the Remote Systems pane you should now be able to browse through the cRio's file system (Figure 21). Note that you may need to enter a password and username. The default username is

admin and default password is blank. You can change the password in NIMax software.

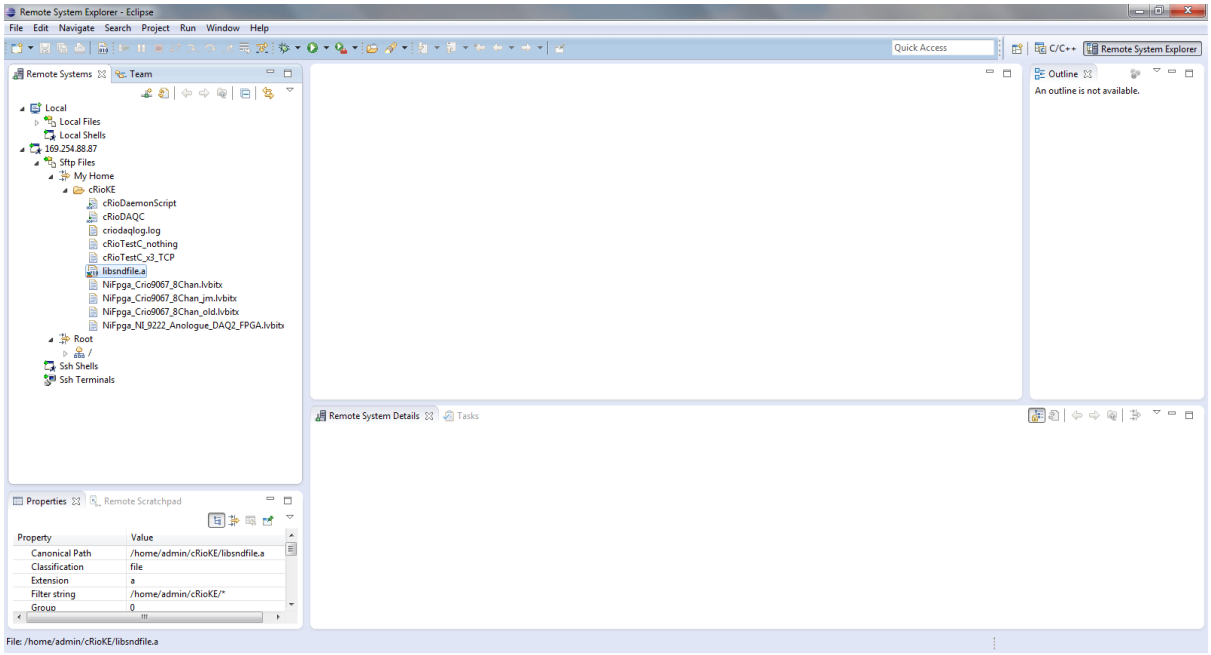


Figure 21. The remote system explorer is a GUI which allows users to browse through the cRio folder system. There is also a console which can be used instead of PuTTy.

Next we create a run configuration which will run a C executable on the cRio. At this stage you should have created a cRioKE folder with files from the online repository as detailed in Section 0.

In Eclipse, select **Run->Run Configurations…**

In the Run Configurations dialog right click on **C/C++ Remote Application** and select **New**…

In the **Connection** drop down menu select the remote connection and fill in the dialog as shown in Figure 22**.**

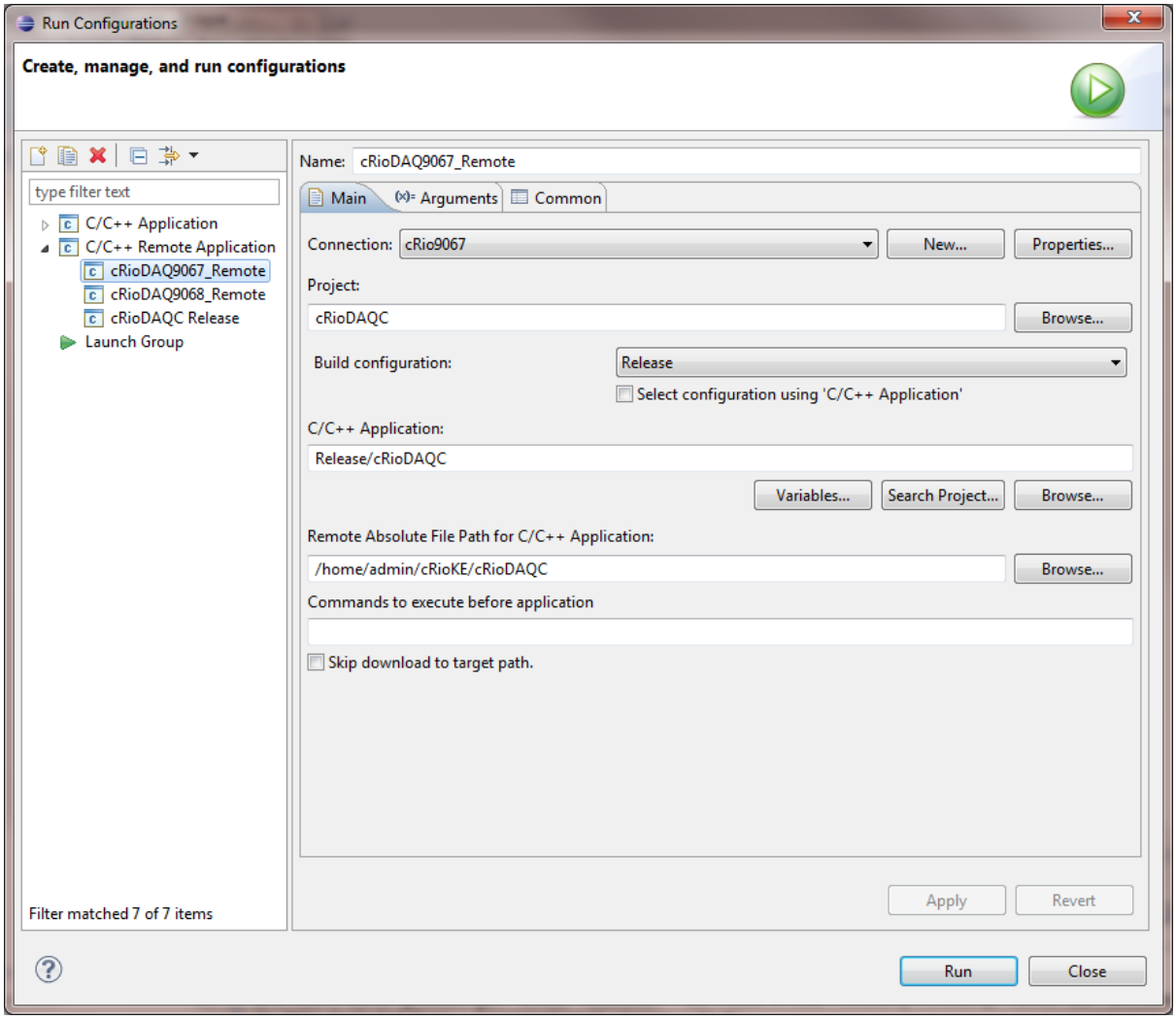


Figure 22. The run configuration dialog allows users to configure the C executable to ruin the cRio rather than the development machine.

Click **Apply** and then **close.**

Next build the project by selecting **Project-> Build Project**. Finally to run, click on the run button. The application should start running on the cRio. In order to begin recording, make sure a hard drive is

attached to the cRio and type **start** into the console.

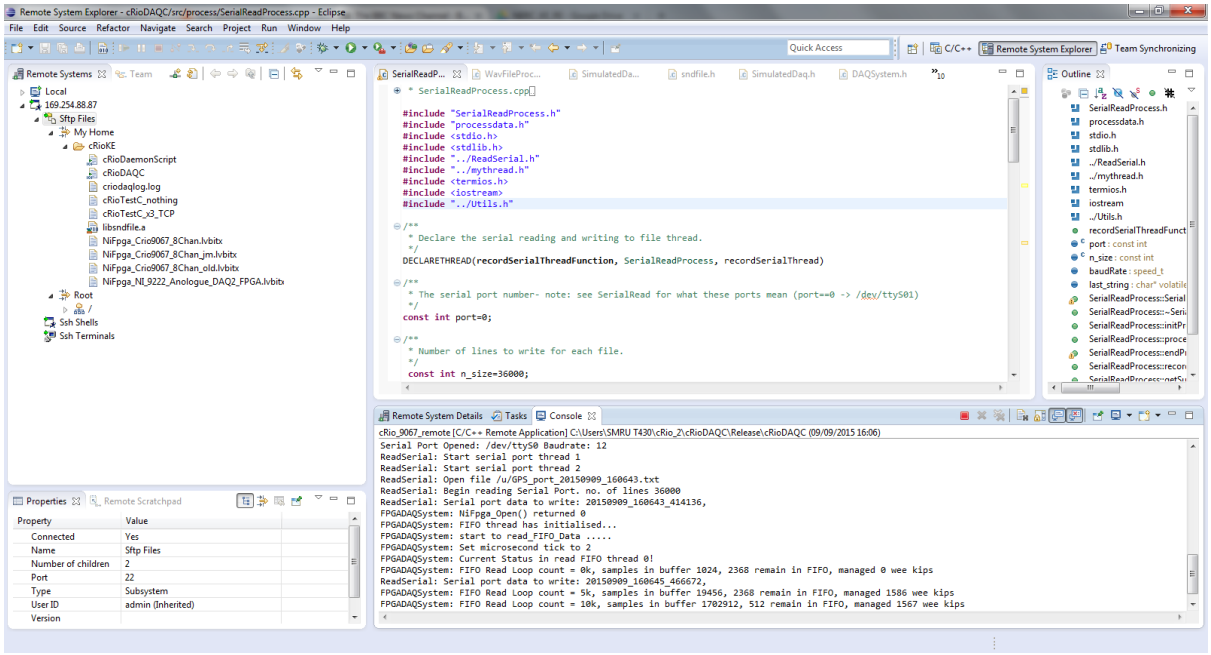


Figure 23. The cRioDAQC source code in Eclipse. The Console is used to type commands and view status output.

You are now ready to begin adding to or improving the cRioDAQC source code (Figure 23).

4 Data Analysis Tutorial

The field data collected using the PLA Buoy will consist of multi-channel (8 in this case) .wav files, GPS txt files collected by the cRio and Open Tag data in a series of .DSG files.

Analysis is split into three stages.

1) **Modelling the movement of the array.** The positions of hydrophones are modelled and a time series of hydrophone positions is created. The heading, pitch and roll of the tetrahedral array will be required to determine accurate bearings to animals.
2) **Extraction of clicks**. Acoustic data are analysed for animal vocalisations. Our examples are of porpoises but dolphin clicks and some whistle vocalisation can also be localised with useful accuracy.
3) **Localisation**. The modelled hydrophone positons and detected vocalisations are analysed together to determine the locations of animals.

4.1 Modelling array Movement

4.1.1 Calibration of IMU units

4.1.1.1 Movement Calibration

Before any analysis is undertaken the OpenTag or other IMU sensors must be calibrated. Movement sensors require a magnetometer calibration to compensate for distortions in the magnetic field around the IMU, a gyroscope calibration to zero gyroscope sensors and pitch calibration to compensate for the fact that IMU units may not be completely vertical when the array is vertical through misalignment.

The magnetometer calibration is performed in the field, the IMU units must be spun through a range of angles in order to calibrate the magnetometer. Ideally this would be performed underwater, with the IMU units attached the array. As this is impractical, the units should be spun close to their attachment point away from any electrical wires or metal objects which could distort results. It is often difficult to get away from such items on a small vessel so it may be sensible to perform calibration on land nearby, *e.g*. on a beach or in a field (stay away from power lines). Several programs to calculate and compensate for magnetic distortions are available. We used the OpenTag MATLAB library and also created our own our own library in Java.

A gyroscope calibration is performed by leaving the device perfectly stationary. All gyroscope sensors should read zero and calibration values are therefore determined by simply taking the average reading for each sensor over the period the IMU was stationary. This must be done on land and should be repeated every few days.

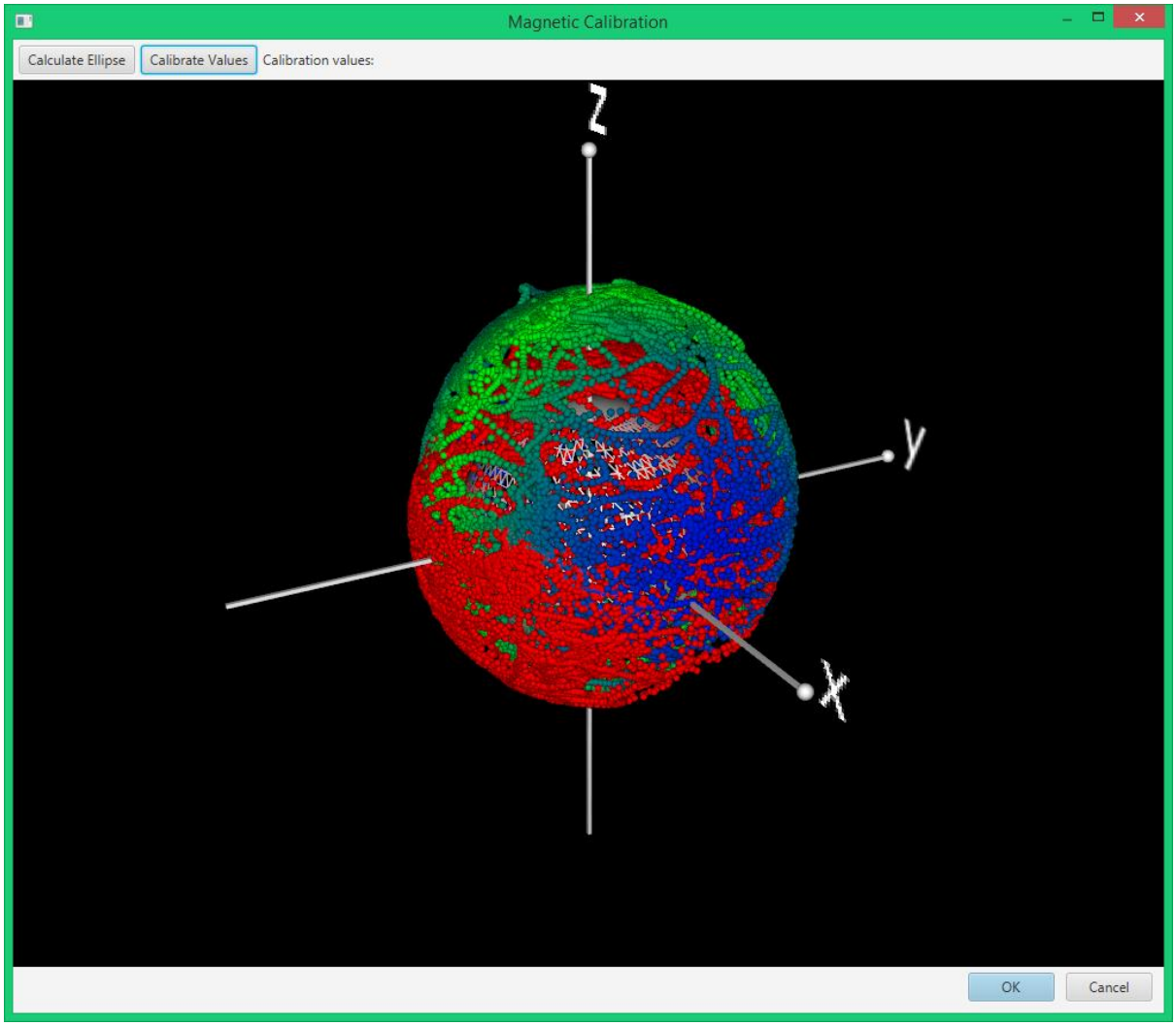


Figure 24. Magnetic calibration of an Open Tag performed in PLABuoy Hydrophone software. The green/blue represent uncalibrated values and the red dots, which should form a sphere, represent calibrated values.

Once the magnetometer and gyroscope calibrations have been performed the Euler angles of the IMU can be calculated. We used a custom MATLAB script from Madgwick (Madgwick et al. 2011) (http://www.x-io.co.uk/open-source-imu-and-ahrs-algorithms/) to calculate Euler angles from raw sensor data. We also created our own Java library for the Madgwick algorithm and this is available online (http://www.x-io.co.uk/open-source-imu-and-ahrs-algorithms/). Euler angles, the heading pitch and roll of the tags are used to model the orientation of the array.

Because the circuit boards or the IMU components themselves may be attached at a slight angle to the array, the pitch and headings are likely to contain some systematic offsets. As heading is the most inaccurate measurement, it is likely any offset is trivial compared to the units poor precision, however this not the case for errors in pitch.

To measure pitch offset, the tag should be oriented perfectly vertically while measurements are taken. We achieved this by hanging portions of the vertical array heavily weighted in a stairwell. It is

important that the tag is in the exact configuration it would be if deployed as even minor offsets of a few degrees could contribute to large localisation errors.

4.1.1.2 Time Calibration

The IMU units are not connected to the cRio; they archive data to an SD card. This means they use a separate clock to that on the cRio. Any clock drifts slightly and hence, after a period of operation, the cRio and IMU units clocks will no longer be synchronised. It is important to ensure that this drift is measured, and if large, compensated for.

To measure the drift simply start the cRio and IMU units before the first deployment and tap each IMU unit against the hydrophone. This will register on the sound files recorded by the cRio and also the gyroscope on the IMU. Once deployment is complete tap the IMU units against the hydrophones again.

To determine the time drifts open up the .wav files containing the taps in Audacity (**http://audacityteam.org/**) or other suitable sound editing program. Navigate to the taps and record the precise time of the first tap (Figure 25). Next open the gyroscope data on the IMU (*e.g.* in MATLAB) and record the precise time of the first tap which corresponds to the first tap in the wav file (Figure 26). Do this for all taps made throughout the survey. Plot PLABuoy time against gyroscope times and calculate the slope of the line to determine the time drift.

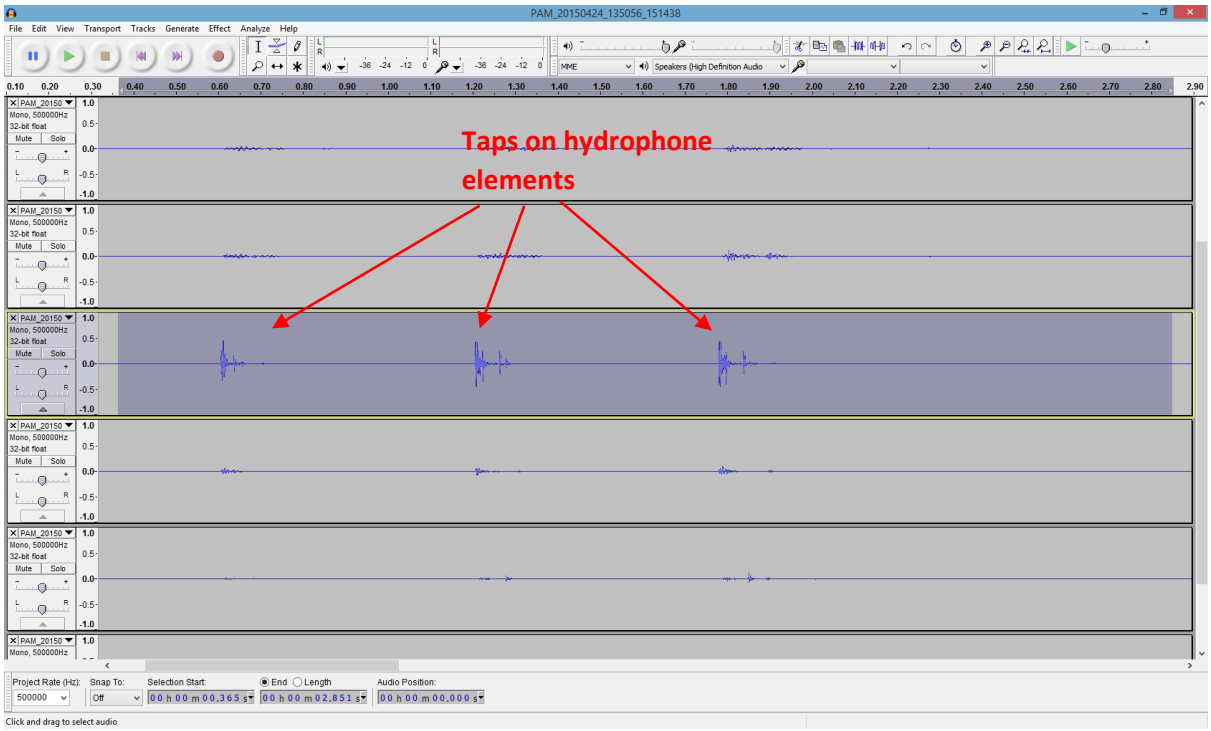


Figure 25. The taps of the IMU unit on the hydrophone register in the sound files.

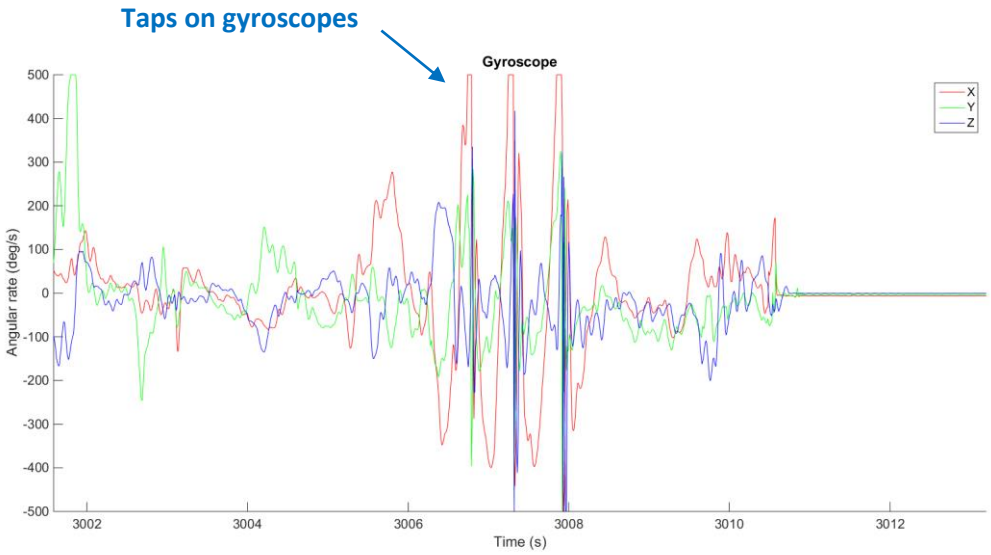


Figure 26. Taps also register on the gyroscope on the IMU. By comparing the time of taps on the IMU and hydrophone it is possible to determine the time drift.

4.1.2 Movement Modelling

After calibrating the IMUs the next stage in modelling array movement is to determine whether a model is actually needed at all.

It is fairly simple to work out the error introduced by the array moving off the vertical angle. *e.g.* If a porpoise is located d meters and the array and θ ° off vertical then the error in depth is simply

$$\epsilon = \sin\theta \times \mathrm{d}$$

So for a porpoise 100 m from the array and the array 5° off vertical then the error in depth would be 8.74 meters. Open tags can be used to determine the average vertical offset angle, and, based on this, an analyst can assess whether errors of this magnitude would be significant for a particular application.

4.1.3 Calculating hydrophone positions

There are two possible strategies to model the array movement

1) **Completely ignore it**. If movement is deemed insignificant then hydrophone positions can be simply input into the PAMGUARD array manager as shown in Figure 27. It is assumed that the main axis of the array, x is 0 and y is 0 and z is the depth of each hydrophone. (The PAMGUARD convention is y points north, x points east and depth points downwards *i.e.* 10 m underwater is depth = 10.) Note that, although very easy, this also will not rotate the tetrahedral array in the correct direction. Therefore co-ordinates will not be properly geo referenced, although localised dive depths and ranges to animals will be correct assuming the array has remained near vertical.

2) **Model positions of both the vertical array and tetrahedral array.** A MATLAB script has been written which can rotate the tetrahedral array and model the position of hydrophones on the vertical array based on measurements stored on OpenTags or other IMUs. A time series of hydrophone positions is produced which can be loaded into PAMGuard as detailed in section 5.2.7.

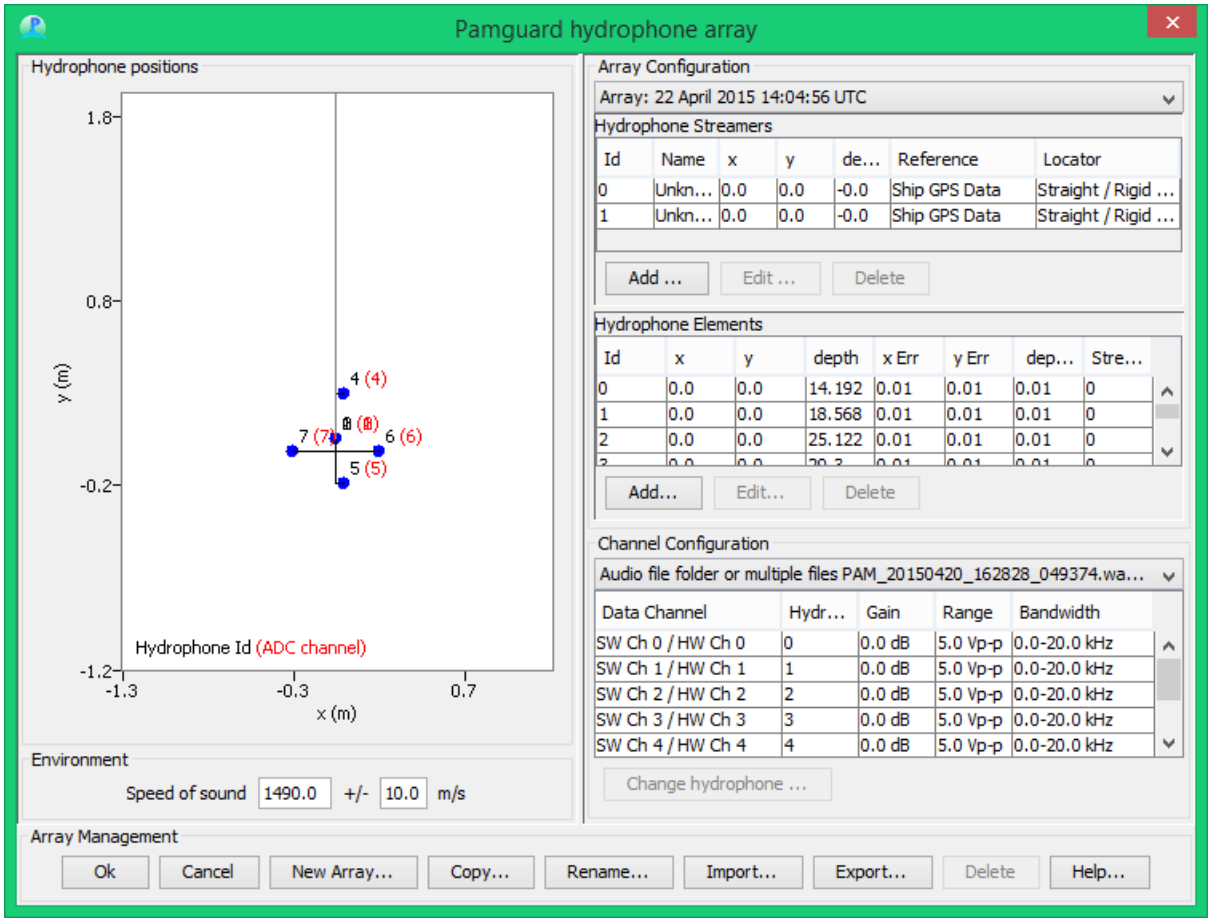


Figure 27 The PAMGuard array manager can be used to input hydrophone positions.

If movement modelling is needed (i.e. approach 2) then this can be performed in MATLAB utilising the OpenTag MATLAB library or by using the PLABuoyHydrophone software. The PLABuoyHydrophone software is capable of modelling array movements, and producing PAMGuard-compatible outputs. It is also more user friendly and does not require any additional coding. A configuration for the PLABuoy configuration that we used can be loaded into the PLABuoyHydrophone software. This can be loaded and then adjusted and saved, e.g. if hydrophone array spacing is different. An example of PLABuoyHydrophone software modelling shown in Figure 28.

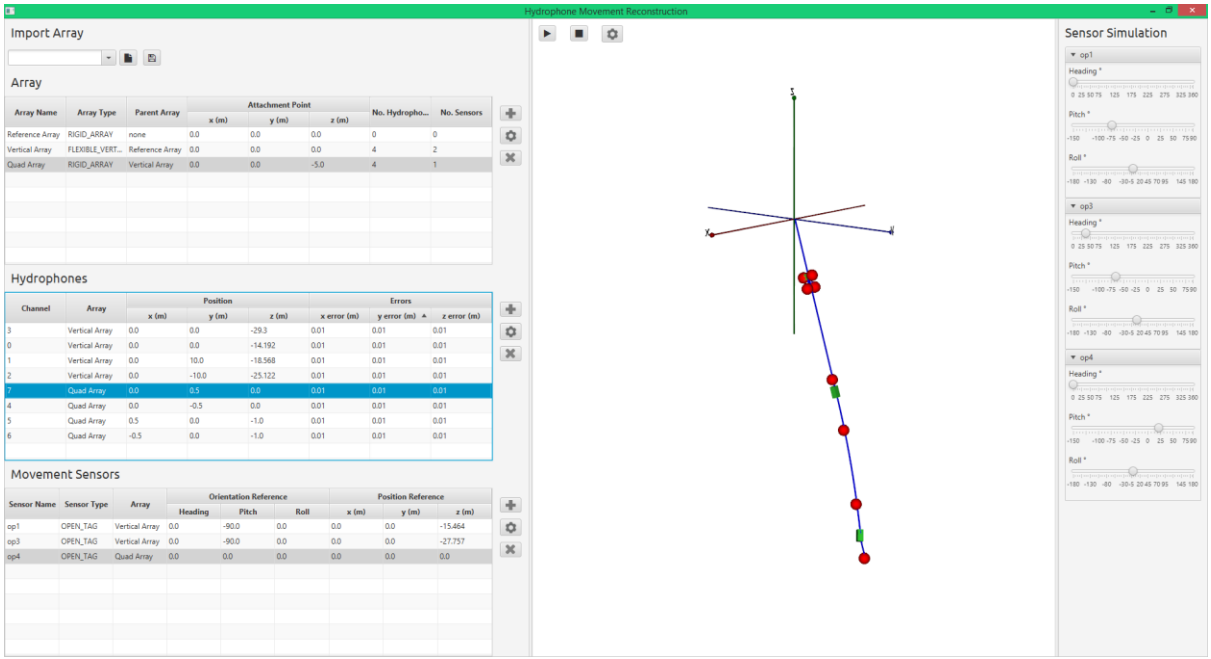


Figure 28. The PLABuoyHydrophone software can be used to model array movement.

5 PAMGUARD Analysis

PAMGuard is the primary analysis software for the PLABuoy. It is used to extract short transient sounds, classify which of these sounds are likely to be small cetacean clicks and then localise the position of animals. The following is a tutorial which familiarises users with the PAMGuard modules used to analyse PLABuoy data. It starts by extracting porpoise clicks automatically from multi-channel WAV files, moves to localisation from a simple vertical array and finally demonstrates how to incorporate array movement into analysis and use a more complex array, such as the PLABuoy.

5.1 Extracting Clicks

The first stage in localising the position of porpoises is to extract porpoise vocalisation from the raw .wav files recorded by the buoy.

5.1.1 Create the Module Structure

Start PAMGUARD in **normal mode.**

Select **File->Add Modules->Sound Processing->Sound Acquisition** to add the sound acquisition module.

This will allow the raw .wav files to be analysed in PAMGUARD.

Next add a click detector. Go to **File->Add Modules->Detector->Click Detector.** The click detector module should now appear.

Finally a database (**File->Add Modules->Utilities->Database**) and binary storage (**File->Add Modules->Utilities->Binary Store**) are needed.

Now you have added the modules, the PAMGAURD data model should look like Figure 29. The next stage is to parametrise the relevant modules.

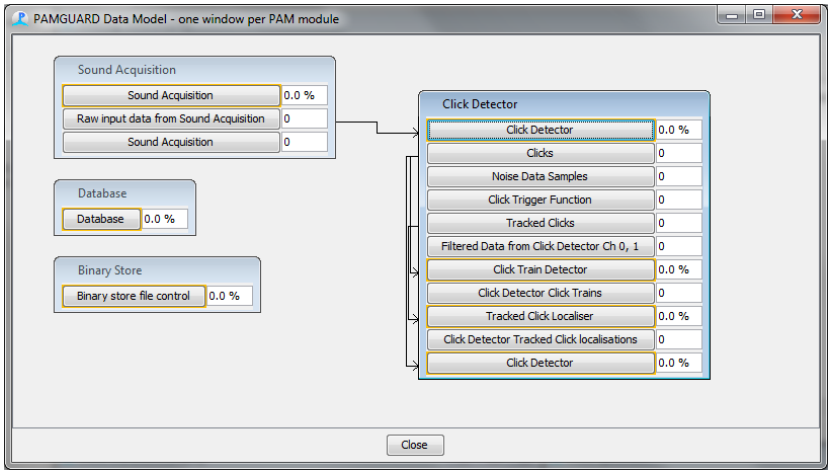


Figure 29. The data model in PAMGuard should look something like this.

When inputting settings in PAMGuard always start with modules on the left and move towards modules on the right. In this instance, start with sound acquisition and then move on to the click detector.

Select Detection->Sound Acquisition… In the dialog box set the audio source to **Audio file or multiple file,** select **Folder or Files**… and choose the folder containing the porpoise .wav files. Now that the sound acquisition knows it is dealing with 8 channels of data it will require an 8 channel hydrophone array to be created in the array manager.

Select **File-> Hydrophone Array…** and use **Import** to select the eight channel array file for this porpoise data. For a standard PLABuoy the hydrophone array manager dialog box should look like Figure 30.

(The xyz coordinates for each hydrophone element can be adjusted as required.)

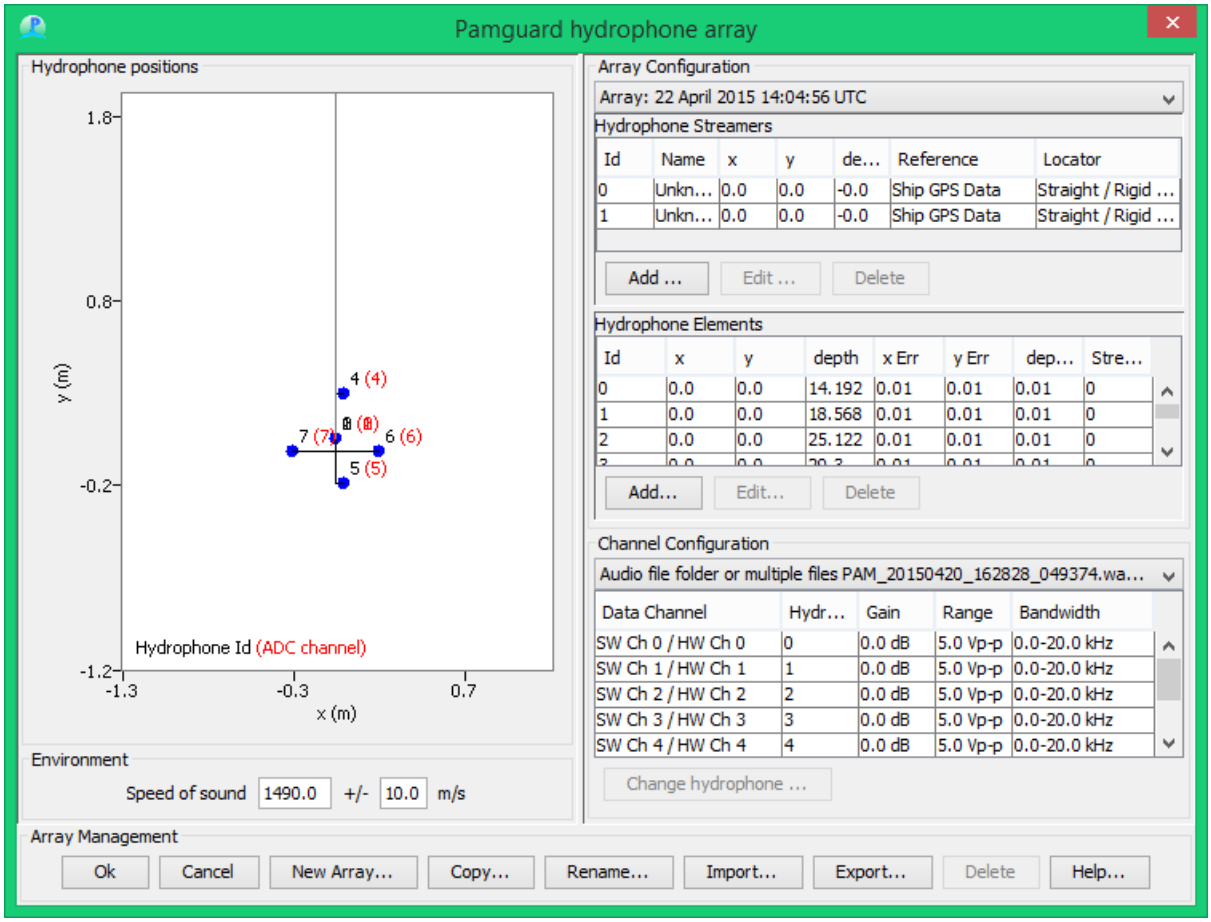


Figure 30. Hydrophone array manager in PAMGuard.

5.1.2 Set the Click Detector Parameters.

Now set up the click detector. Go to the **Click Detection** menu on the main screen and select **Detection Parameters…**. Clicks need to be detected on all channels so select the check box for all eight channels. Make sure that no grouping is selected. This is usually a sensible choice for wide aperture arrays (see note on groups in 5.1.5).

Select the **Trigger** tab and ensure check boxes are selected for every channel. This ensures that the click detection algorithm is run on each channel individually. (Figure 31). Click **OK.**

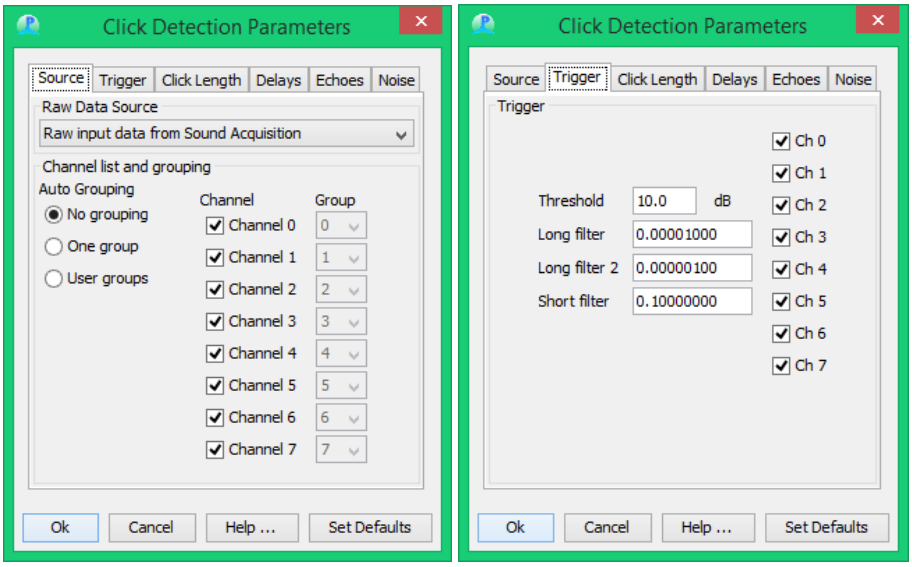


Figure 31 Make sure the channels are not grouped and all selected in the trigger tab.

5.1.3 Create filters to optimise for Porpoise Click Detection

Next, configure filters to optimise detection of porpoise clicks. There are two filters in the click detector, a pre filter and digital trigger filter. The pre filter filters the raw sound data used by the click detector module. To remove low frequency noise a high pass filter set at 20 kHz is ideal. Select **Click Detection-> Digital Pre Filter….** In the dialog which pops up create a high pass Butterworth filter at 20000Hz (Figure 32).

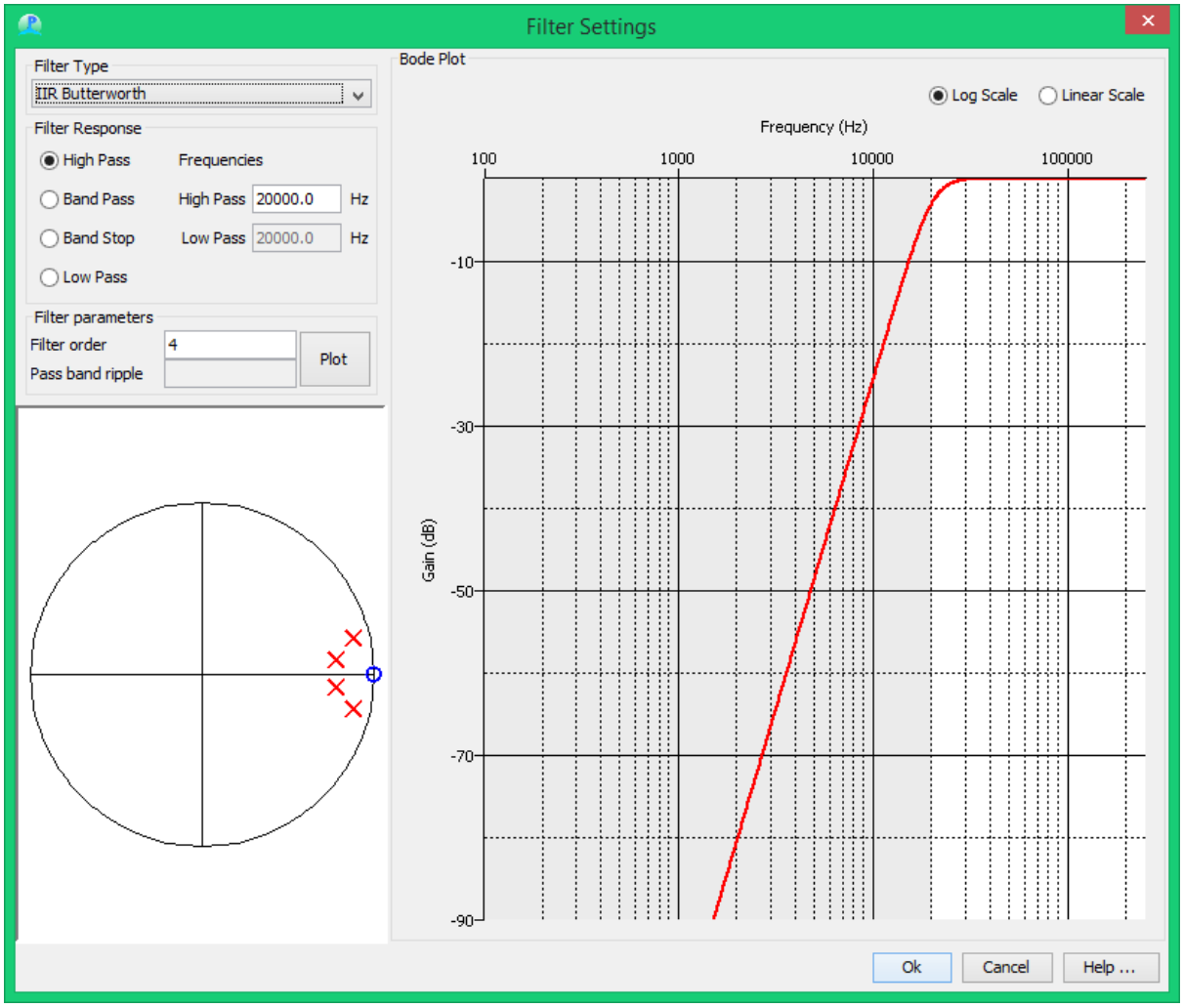


Figure 32. The digital pre filter should be set to 20kHz, high pass.

After the pre filter, the digital trigger filter should be configured. The digital trigger filter is only used by the algorithm which searches for clicks. The final saved click waveforms and spectra are taken from the raw sound data filtered by the pre filter. Since porpoises have narrow band high frequency clicks it is sensible to trigger on clicks within a narrow frequency band. Hence the digital trigger filter will be a band pass, from 100 kHz to 150 kHz. Select **Click Detection-> Digital rigger filter…** In the dialog which pops up create a band pass Chebyshev filter between 100000 and 150000Hz (Figure 33).

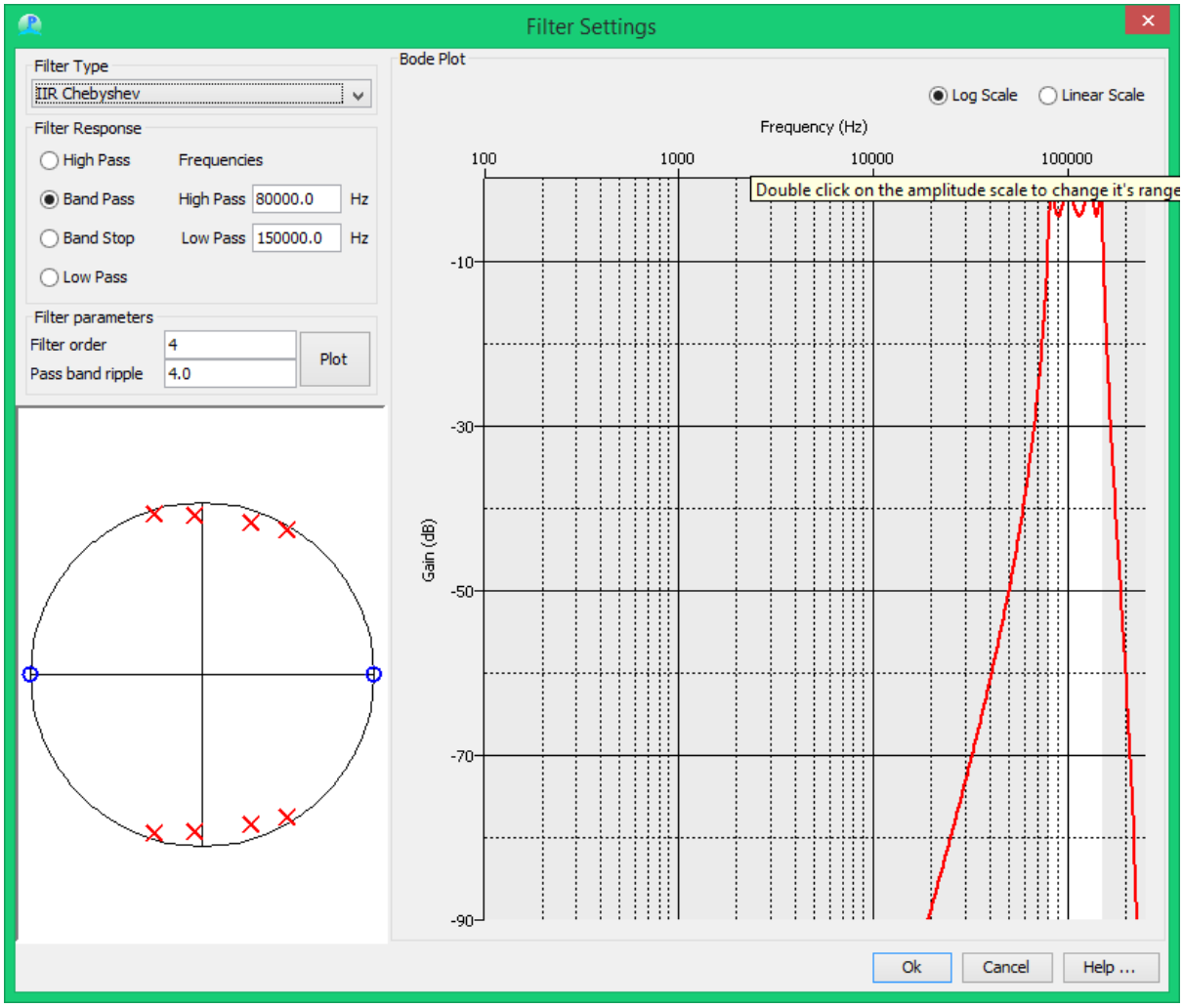


Figure 33. The digital trigger filter should be a band pass between 100 and 150kHz.

5.1.4 Create a porpoise click classifier

The click detector is now set up to detect clicks in the porpoise frequency range. Even though the click detector has been optimised for porpoise like vocalisations it will also detect many non-porpoise transient sounds. To distinguish porpoise vocalisations, click detections from other transient sounds need to be classified using the PAMGUARD click classifier.

Go to **Click Detection-> Click Classification.** In the **Click Classifier Selection** drop down box select **Classifier with frequency sweep.** Select **New** to bring up a new click classifier settings dialog. In the bottom of the dialog box use the **Set Defaults** button to create default porpoise parameters (Figure 34). Click OK.

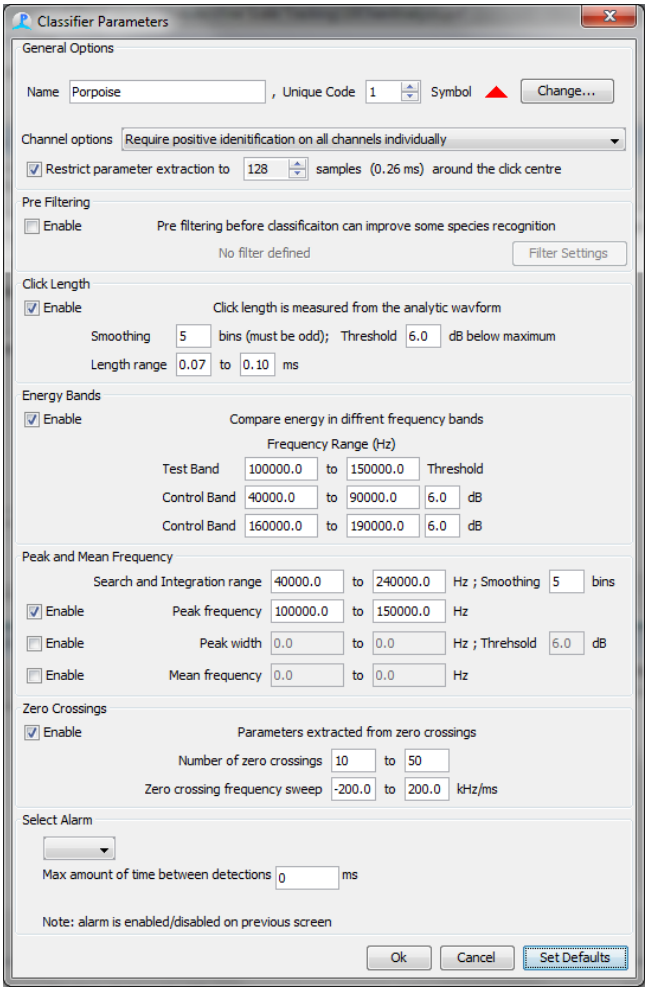


Figure 34. The click classifier has default settings for porpoise clicks.

A porpoise click classifier has now been created. To ensure that classifier is applied to data make sure that the **Run classification online** and **Enable** check boxes are selected (Figure 35).

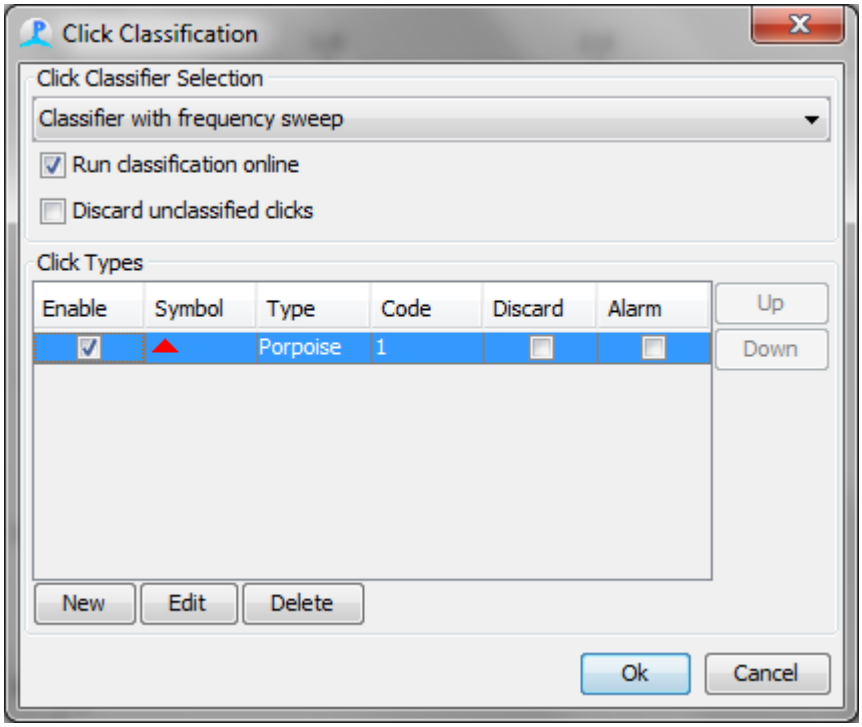


Figure 35 Click classifier manager.

5.1.5 Running

Setup is now complete and PAMGuard is ready to detect porpoise clicks.

To run select **Detection->Start** and PAMGuard will start analysing data (Figure 36).

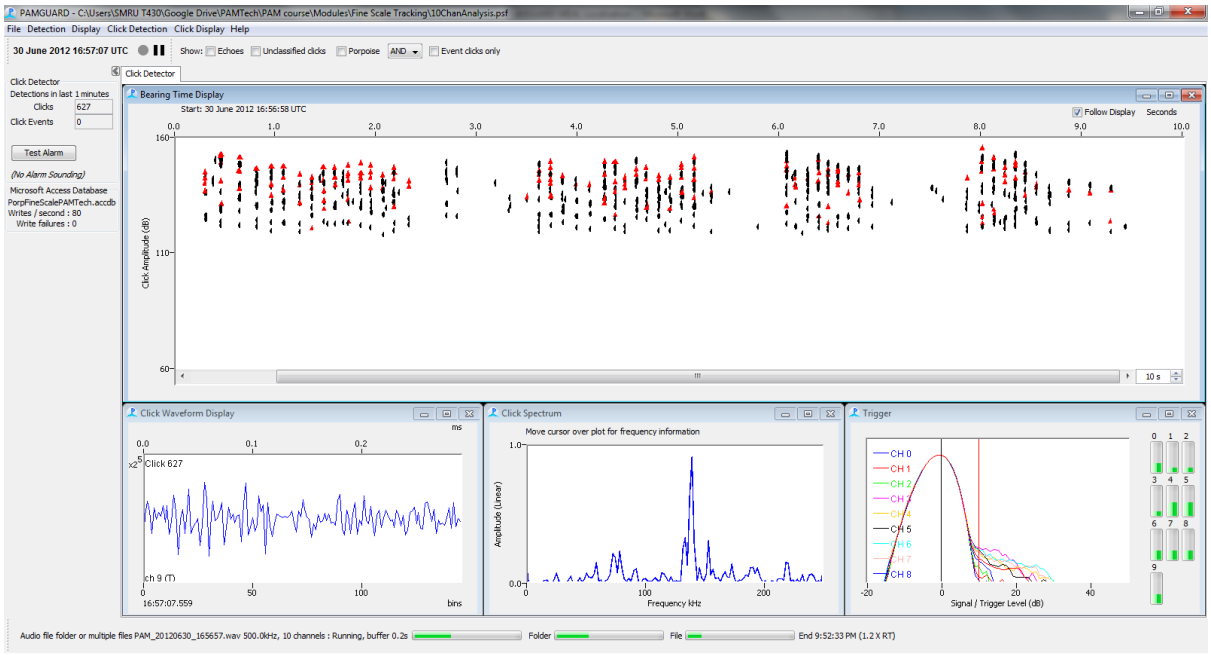


Figure 36 The click detector running through the ten channel .wav files should look something like the above.

A note on groups. *In many detectors there is the option of 'grouping channels'. Grouping instructs PAMGuard to automatically calculate extra information for those channels which have been selected as belonging to a group. Grouped channels are saved as a single data unit containing information from each channel. For example, a simple towed array might contain two closely spaced hydrophones. These two channels are usually grouped in PAMGUARD. If a click is detected on one of the hydrophones then a waveform clip is also automatically recorded from the other channel and a data unit is saved containing information on both channels. PAMGUARD can calculate additional information for any data unit which contains more than one channel of information. For a towed array, two clicks allow bearing information to be calculated for bearing time displays and target motion analysis. In general, widely spaced hydrophones are not grouped because the waveform clips, which have to be at least as long as the time for sound to travel between the elements, would be too long. This would use up memory and also mean waveform clips contained echoes, and possibly other clicks from the same or different animals.*

5.1.6 Viewing Data

Thus far in this tutorial, PAMGuard has been used to run through eight channels of raw .wav data and pick out candidate porpoise clicks. All clicks, including classified porpoise clicks, have been saved in PAMGuard binary files by the binary file module added earlier. These files are around 0.01% of the size of .wav files and yet contain all information needed to browse through click data and perform complex tasks, such as classification or localisation, however they no longer contain the raw data (hence the reduction in size). This section describes how to open Binary files in PAMGuard viewer mode so that data can be visualised and further analysed.

Start PAMGuard in viewer mode (if using Windows, type 'pamguard viewer' into search/Cortana if you do not have shortcut set up). PAMGuard will ask you to select a database. Select the database you used in analysing the .wav files.

Next PAMGuard will ask you to select binary files (Figure 37).

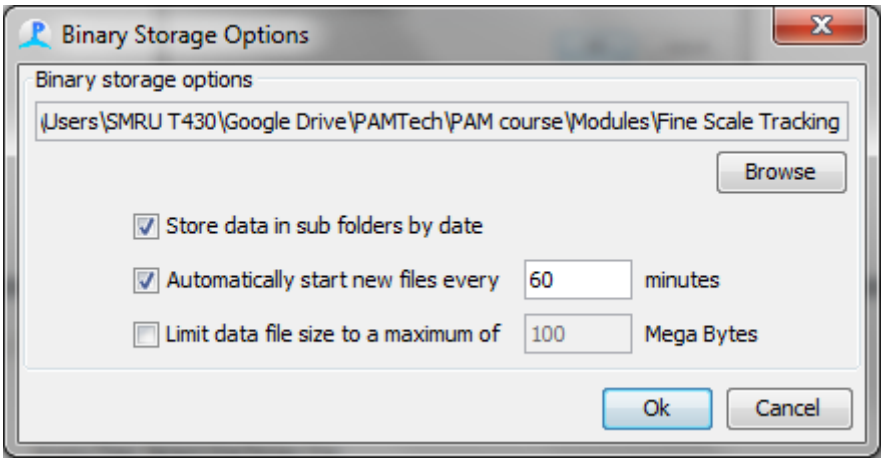
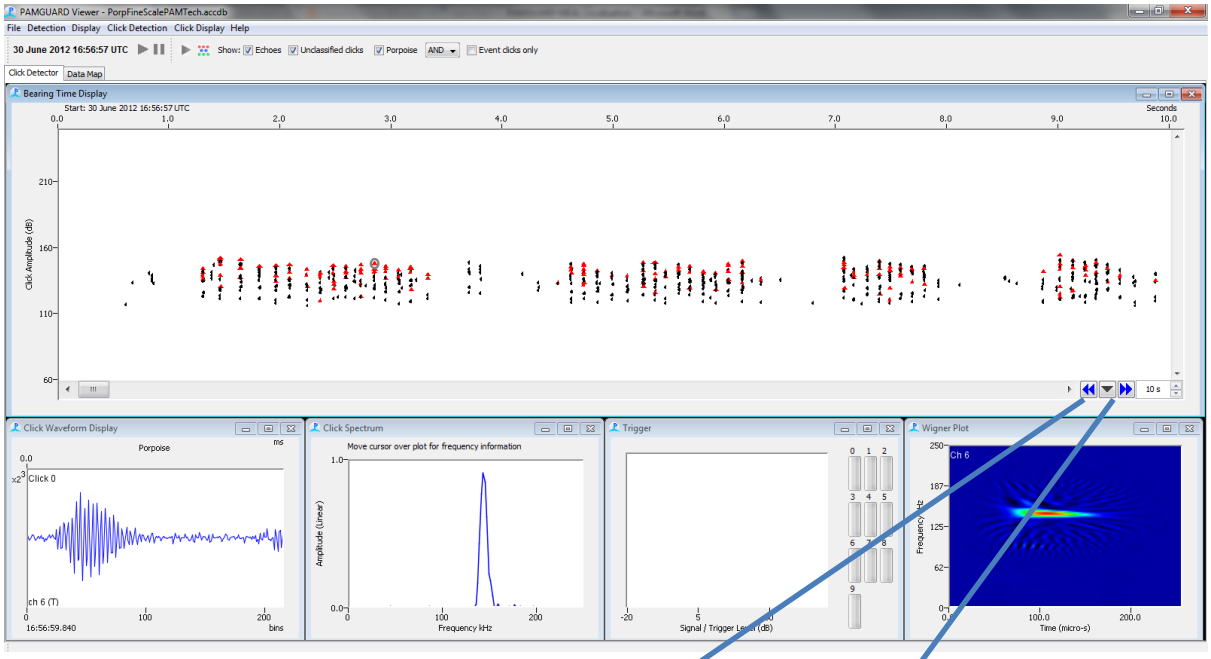


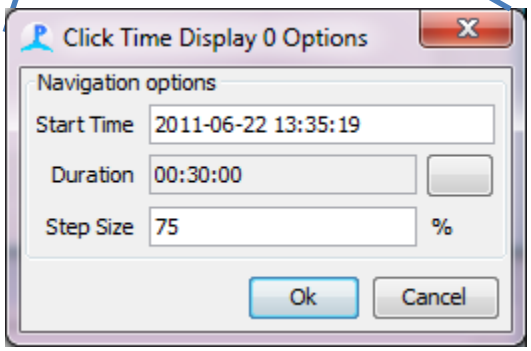
Figure 37. Open the binary files you saved.

Select the folder in which binary files were saved when PAMGuard was used in standard mode (section above). PAMGUARD should now open and a click detector window should be visible.



Extra scroll bar controls are present in many primary displays in PAMGUARD Viewer Mode. Click detections within a certain time period are loaded into the computer's memory at any one time and only these detections are visible in the click detector display. To move to a time period before or after the currently loaded data press the blue buttons.

*The middle button can be pressed to bring up a dialog box. Here you can define the period of data (**Duration**) to be loaded into memory. The **Step Size** defines the 'jump' to take to the next section of data when the blue buttons are pressed. For example 75% means that when a new section of data is loaded, 25% of the previous section is included. The **Start Time** allows you to manually select where to load the data from.*



The display is very similar to that of the Standard PAMGuard program, however, there several important differences. The scroll bars allow users to navigate through and load new sections of data. New displays are also available as is additional functionality to annotate clicks and add to events.

5.1.7 Reclassifying clicks.

It may be the case that, the classifier used in section 5.1.4 has missed a significant number of porpoise clicks. For the purposes of localisation it is often better to have more false positives than false negatives Thus, it may be desirable to reclassify click detection using a slightly more lenient classifier. Go to **Click Detection-> Re Analyse Click Types…** This will bring up the click analysis dialog box (Figure 38).

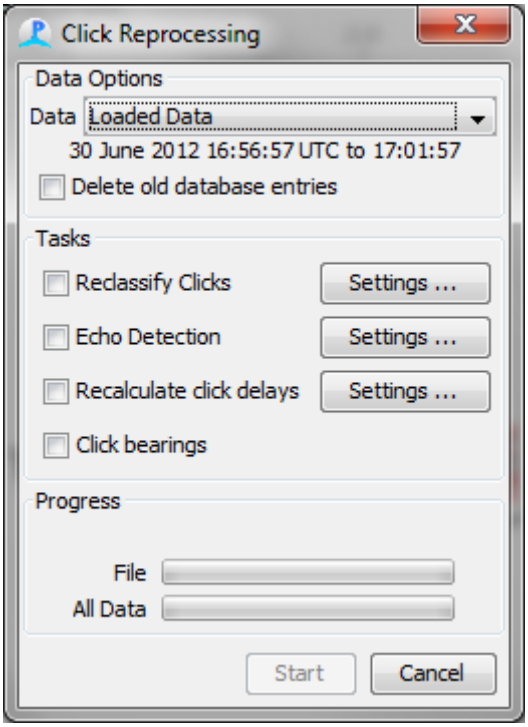


Figure 38. Reprocessing clicks allows you to tweak classifier settings.

Tick the **Reclassify Clicks** check box and click on **Settings…** This will bring up the familiar click classifier manager from which you can access your classifier settings (Figure 39).

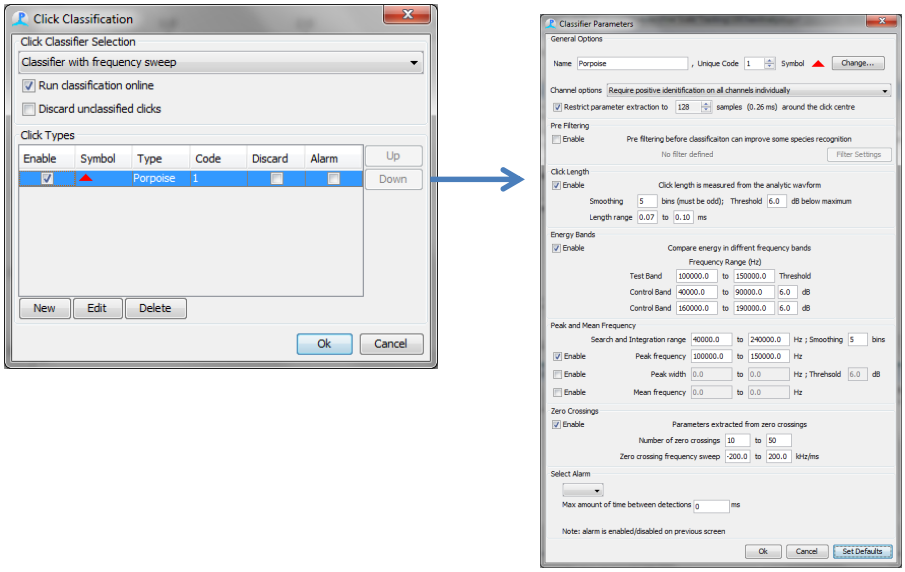


Figure 39. Changing the click classification settings. In viewer mode it is possible to reclassify clicks.

In the click classifier disable the **Click Length** and **Zero Crossing** settings. Click OK on the classifier and Classifier manager to return the **Click Analysis dialog**. In the **Data Options** drop down menu select **All Data** (Figure 40)**.** This will perform data processing on all binary files rather than just the click detections currently loaded into memory. Click **Start** and wait for the processing to finish. The specified click detections will have been reprocessed with the revised classification parameters.

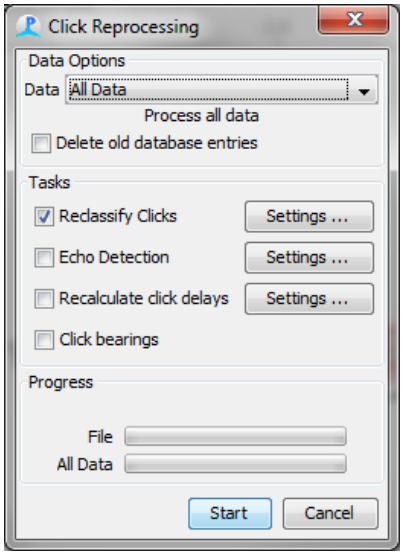


Figure 40. The batch processing dialog allows users to change re classify and recalculate basic localisation information for detected clicks

5.2 Localising

5.2.1 Set up the localiser

The Large Aperture Localiser module can be configured to localise animals using click data from any large hydrophone array, such as the PLABuoy. Because localisation is computationally intensive it is best to use the large aperture localiser in subsets of data in viewer mode rather than applying it during the initial analysis run described in the previous section.

First add the **Large Aperture Localiser** to in PAMGuard viewer mode: go to **File->Add Modules->Localisers->MEAL localiser.** This will add the **Large Aperture Localiser** to viewer mode.

A new localiser tab will appear (Figure 41)

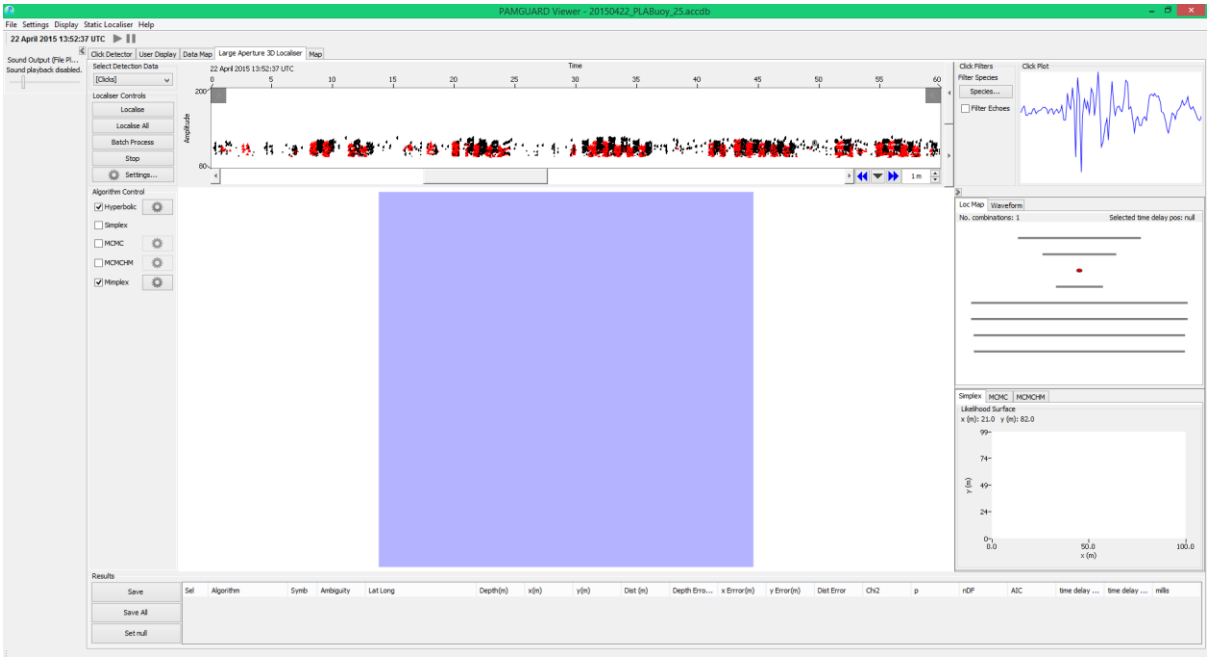


Figure 41. The MEAL localiser should initially look something like this.

The module display has several different panels (Figure 42)

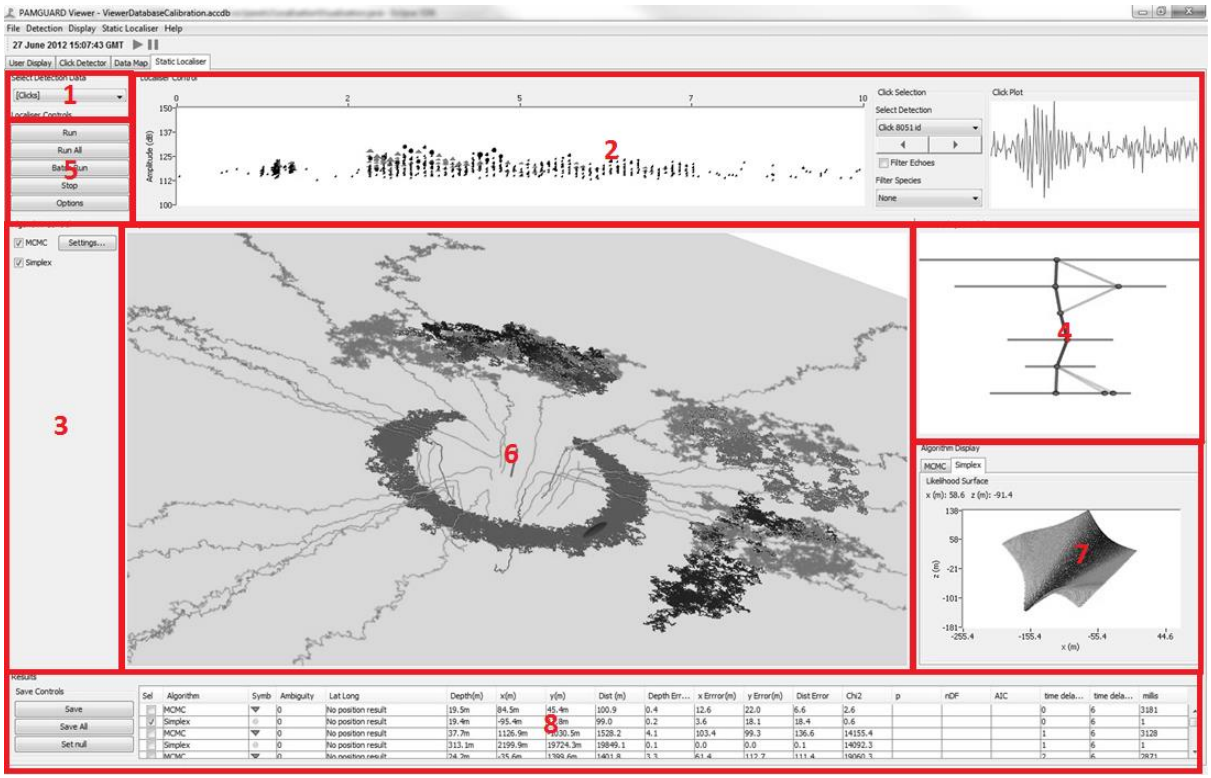


Figure 42. A diagram of the Large Aperture Loclalisation module in PAMGuard. Numbers denote panels in the graphical user interface whos function are explained below.

1] Data block selection menu.

This allows users to select which data block to localise detections within. Note: currently only clicks can be localised.

2] Click time display

Click bearing, click amplitude or ICI (inter click interval) versus time displays can be selected. Users can manually select the detections to localise from this display.

3] Algorithm selection

Allows users to select the algorithm to use to determine the position of an animals. These include a fast Hyperbolic Localisation method and more computationally intensive MCMC (Markov Chain Monte Carlo) methods.

4] Detection match

The detection match display shows which click detections on each hydrophone could belong to the same animal. In some case there may be some ambiguity as to which detections to include in the localisation calculation. Users can use this display to select which combination of detections to use in the localisation.

5] Localiser controls

Allows users to localise a selected detection. Also allows users to batch process either selected time intervals or all the data.

6] Map

3D representation showing hydrophone array and localisation results.

7] Algorithm Display

Algorithm specific display showing information on performance of each localisation algorithm.

8] Results

Shows a table of results from all selected localisation algorithms

The top Click Time Display panel contains, by default, a miniature bearing time display. However, because clicks were not analysed as a group by PAMGuard in the previous section, there is no bearing information to display. Thus, we can only view amplitude or ICI values on the y axis.

Click on the array on the top left of the display and select **Amplitude** in the window which appears (Figure 43).

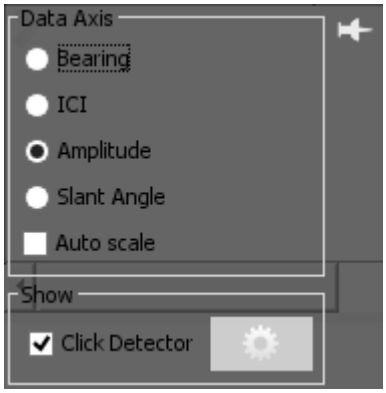


Figure 43. Because in this case clicks are not grouped, data blocks data can only be viewed on an amplitude vs time display.

The Click time display panel will now look similar to the main Click Detector bearing time display (provided that amplitude was also selected for the y axis on that display).

Select Porpoise in the **Filter Species** box. This tells the localiser to only use classified porpoise clicks in calculations.

The Map panel displays localisation results and shows the hydrophone array. Right click to drag the map, left click to rotate and use the mouse wheel to zoom in and out. The display should now resemble Figure 44

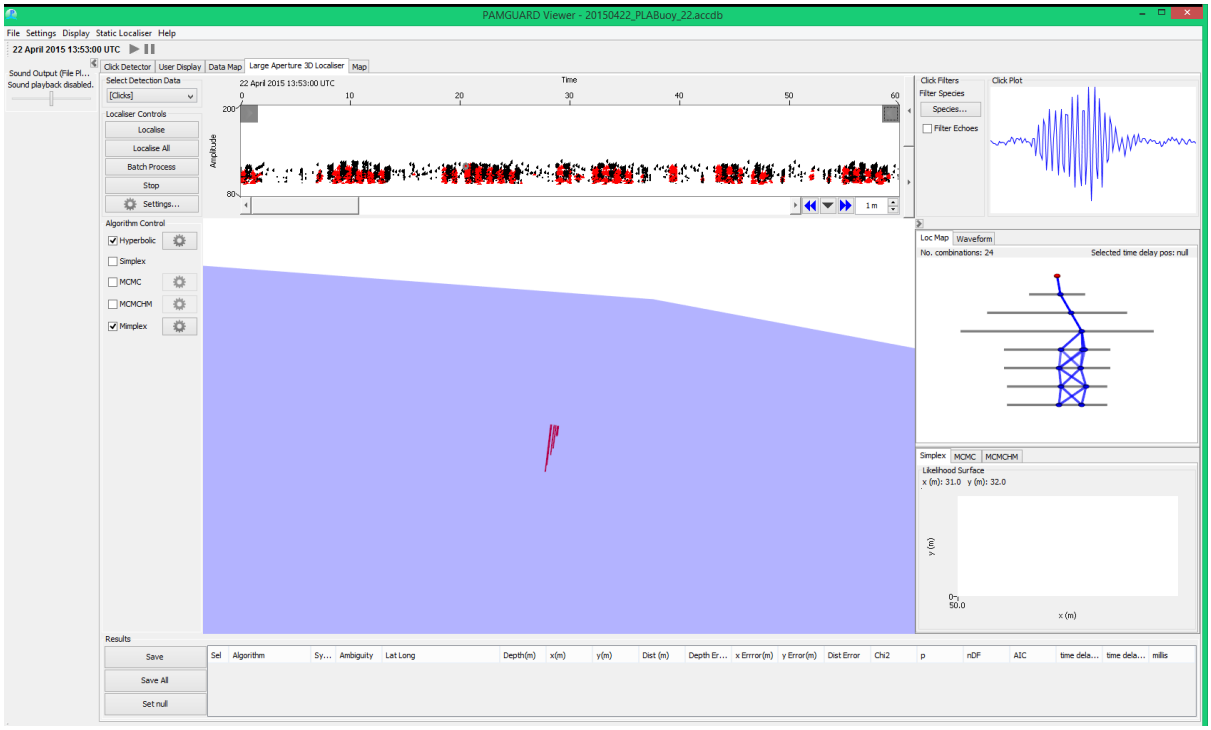


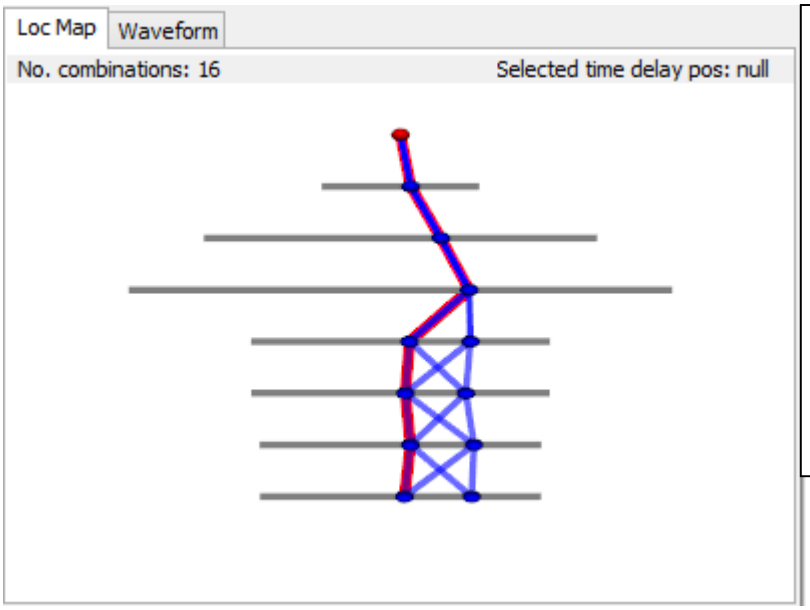
Figure 44. Use the mouse left click, right click and wheel to manipulate the 3D map.

Select a porpoise click by clicking on a click in the Click Time display. The Detection match panel will change showing a network of possible click combinations. The Detection match panel contains

information on the number of possible time delay combinations. The red circle represents the selected detection, in this case whichever click has been selected in the Click time Display. This selected detection is the **primary** detection and the hydrophone on which it was detected is designated the **primary hydrophone.**

In order to localise the position of an animal this same detection needs to be found on other hydrophones; to do this PAMGuard searches a time window before and after the time of the primary detection; time windows are represented in the panel by grey lines. Each hydrophone will have a different time window depending on how far it is away from the primary hydrophone. It is possible that a time window will contain more than one detection.

In the Localisation Map (Figure 45) the **blue dots** represent the possible matches between the primary click and other detections within the appropriate time window for each hydrophone. Each **blue line** represents a possible combination of detections of which only one will contain only detections which correspond to the primary detection. The number of possible combinations is shown at the top of the window.



Reasons for multiple time delay possibilities

-echoes

-multiple animals

-rapid vocalisations

-widely spaced elements

Figure 45. The number of possible time delay combinations is displayed in the Detection match panel. Lines represent each combination and the red dot represents the currently selected detection.

In the next section of the tutorial we will use the vertical component of the PLABuoy to determine an animal's position. This relatively simple procedure, will reduce the number of time delay combinations thus computation time and provide information on depth and range to vocalising animals.

In the localiser controls panel select **Options and** in the **Channels** tab deselect channels 6, 7, 8 and 9. In the **Map** tab make sure that **Use high res. Plot symbols** is selected. Click **OK**.

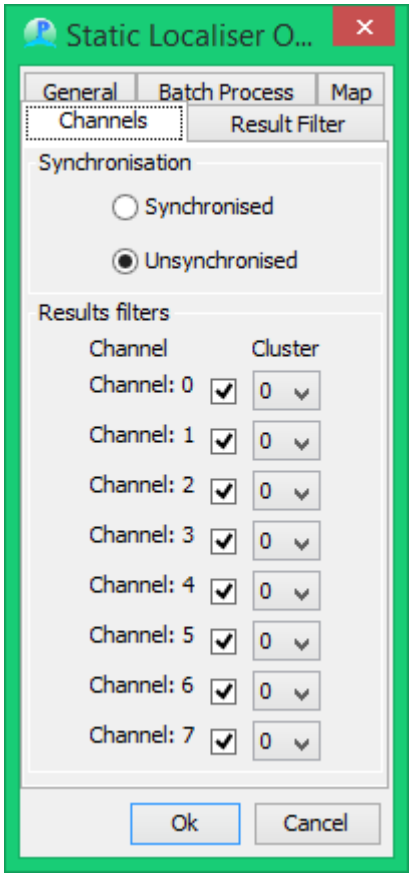


Figure 46. For this exercise we only need channels 0-5 selected.

5.2.2 Localise a click.

A localisation algorithm must be selected before a localisation can be performed.

Select MCMC in the Algorithm selection panel. The **Localise**, **Localise All**, and **Stop** buttons will become enabled. Find a click which contains corresponding detections on most other channels and which has more than one time delay possibility. In the detection match panel such a click should look something like Figure 47. Remember that only the vertical component of the array is being used and thus only four channels are displayed.

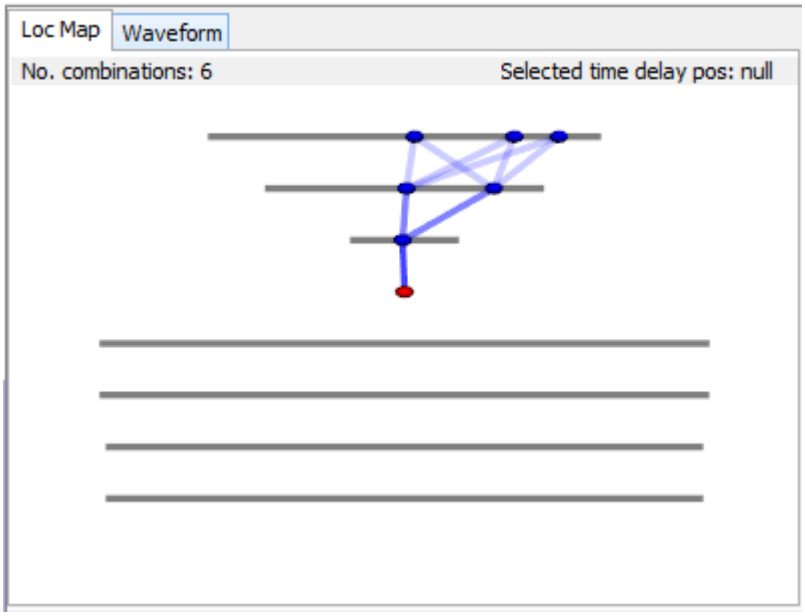


Figure 47. Select a click which has time delay possibilities roughly similar to this.

The large aperture localiser module allows users to localise using only one of the possible time delay combination or to localise all combinations and select the best one; that is the combination of time delays which provides the best results (best fit to the localisation algorithm) and makes most physical sense.

Click the **Localise** Button to localise only one combination. You can also select individual combination in the **Detection Match** panel.

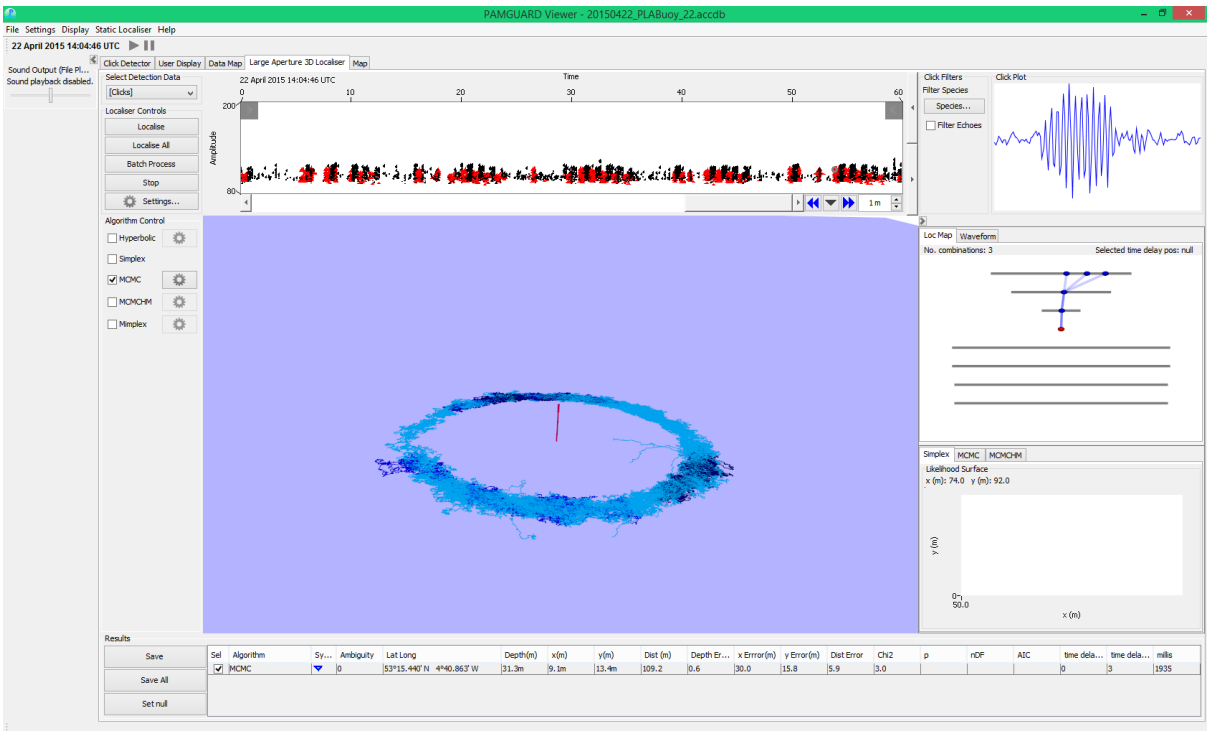


Figure 48. MCMC produces a circular probability distribution. This is exactly what would be expected from a linear array.

A circular ‘cloud’ will appear on the map similar to that in Figure 48. This cloud corresponds to the probability distribution of the porpoises’ location calculated by the MCMC algorithm. The cloud is circular because a linear array will only provide values for **depth** and **range.**

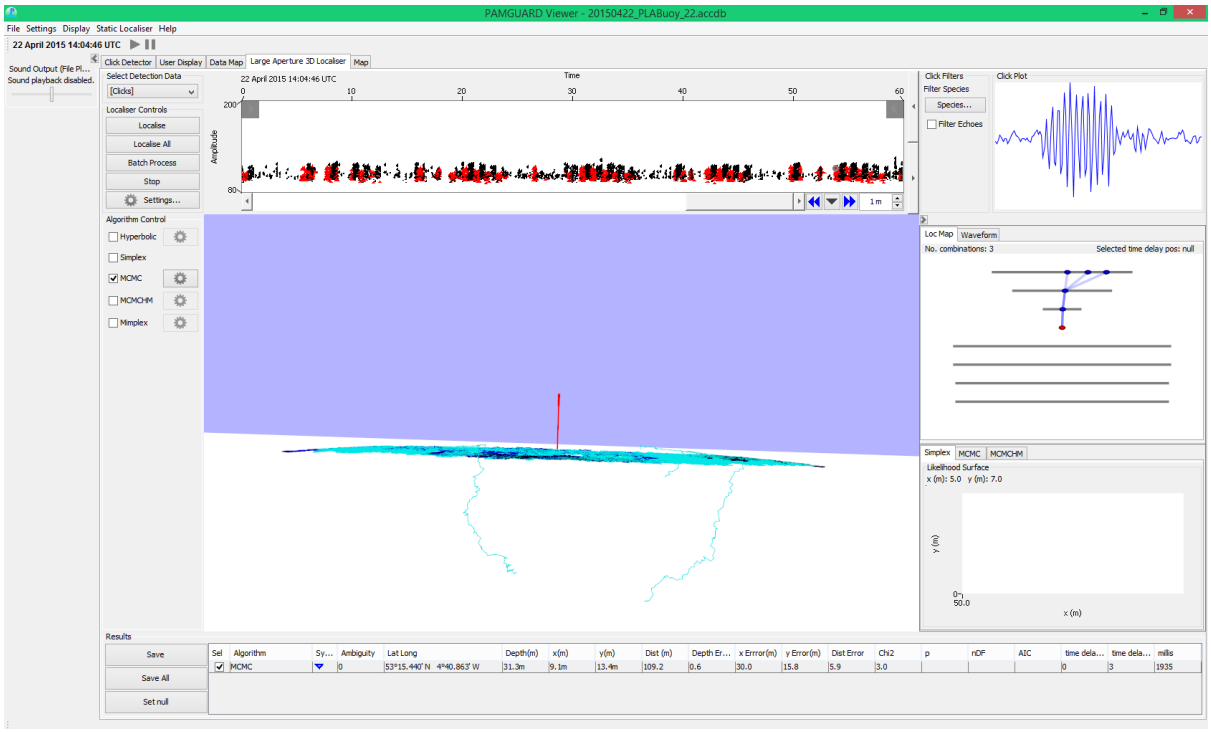


Figure 49. Using the map you can view the circle side on. Note it has a well-defined depth.

Align the map horizontally as shown in Figure 49 and it will become apparent that the cloud has a well-defined and restricted depth. Range to the sound source corresponds to the radius of the circle. The ability of the MCMC method to indicate these probability distributions is one of its primary strengths.

The results panel displays summary information on the position of the animal including its depth, range and corresponding errors.

5.2.3 Localise all Combination

Next try to localise all possible combinations of time delays. PAMGuard will indicate which combination is most likely to be correct. This is typically how the localiser will operate when run in automated mode.

Choose **Localise All.** The localiser will now localise all the possible combinations. Once the localiser has stopped the localiser module should look something like Figure 50

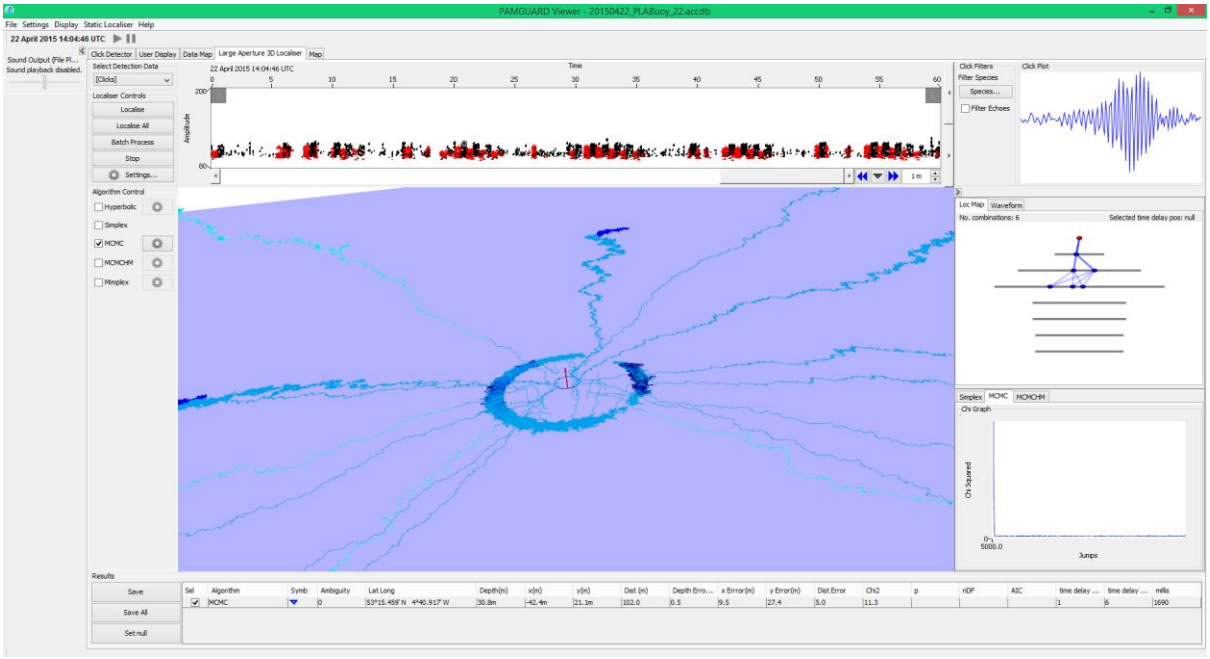
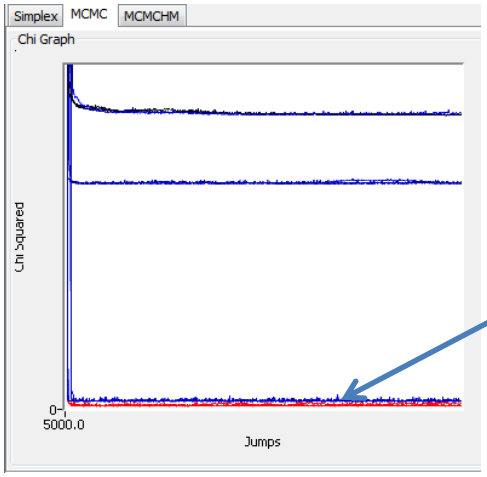


Figure 50. You can localise all possible time delay combination for a given detection. The result which has the lowest chi squared value is also likely the correct combination of detections.

The display now shows multiple differently coloured localisation clouds (Figure 50), some which have the expected circular shape and others that do not. Simply by observing these it will be clear that some combinations produce better results than others.

Click on one of the clouds. It will turn red along with its corresponding time delay combination in the Detection Match panel. Clicking on a symbol on the map will also highlight the corresponding localisation result in the Results panel. The algorithm display panel has an MCMC tab (Figure 51). When selected this will show a graph of χ^2 versus 'jump iteration'.



The highlighted series here has the lowest average χ^2 (chi squared) value and hence is the best fit result. χ^2 is a measure of how similar the observed time delays are to the theoretical delays which would be produced from a source at a specific location, in this case the localised position of an animal. By selecting it in this panel the corresponding cloud in the map and time delay combination in the detection match panel will be highlighted. Selecting the line with lowest final value of χ^2 will usually correspond to the correct localisation result.

Figure 51. The χ^2 values of all jumps for all attempted MCMC localisations.

The more quickly this graph reaches a low χ^2 value and the lower the final value, the better the time delay information has fit the algorithm. You can also use this panel to select and highlight

corresponding detection match lines and plot symbols. This is a convenient way of picking the combination which has the lowest final χ^2 value (Figure 52).

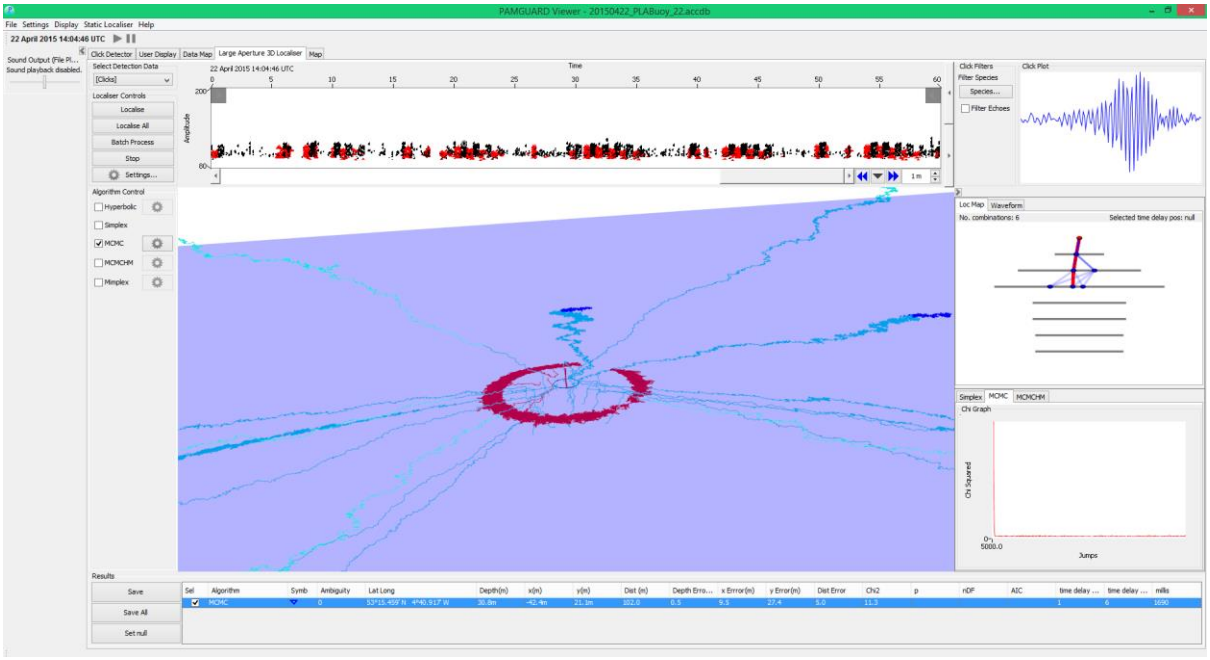


Figure 52. You can select 3D shapes on the map and time delay possibilities in the detection match panel.

In the results panel, the result with lowest **X^2 (chi squared) value** is ticked by default. This result is the best fit to the algorithm, *i.e.* usually the most sensible result which is likely to have been calculated using the correct combination of time delays. The chosen result can be saved by pressing **Save.**

5.2.4 Simplex Localisation

The Simplex algorithm is similar to MCMC however does not show true probability distributions. It is, however much faster.

Select **Simplex** in the Algorithm selection panel and compare it to the MCMC algorithm by localising a few more clicks. The **millis** column in the results table shows the computational time of each algorithm (Figure 53).

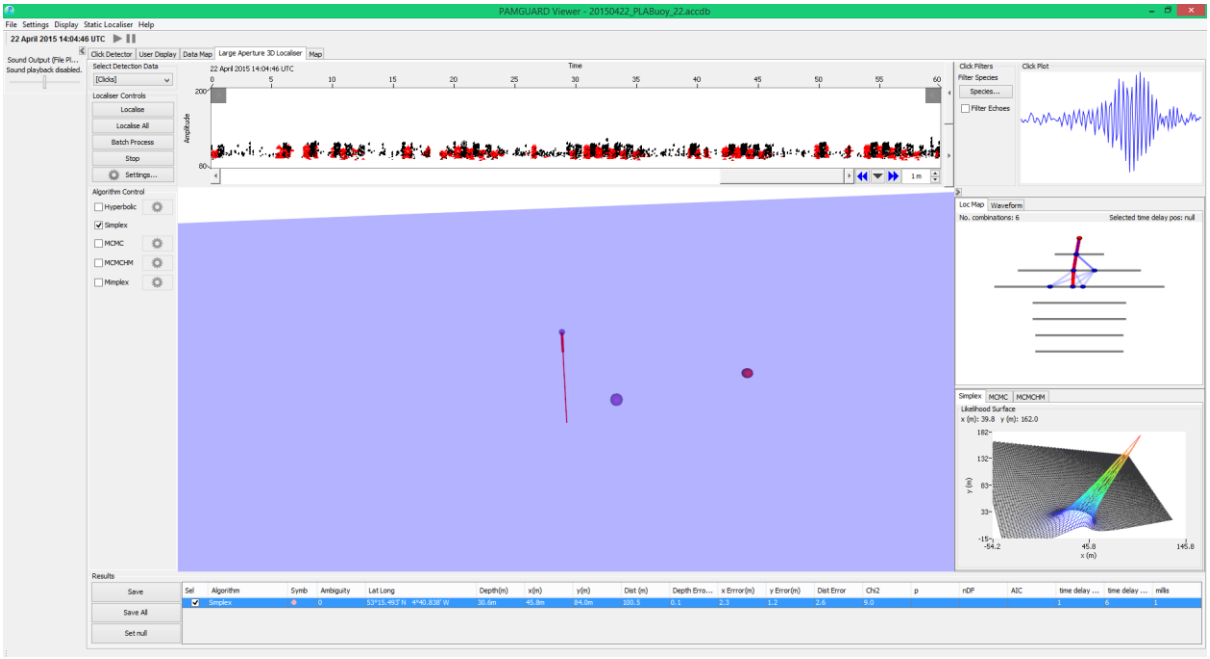


Figure 53. The simplex algorithm is much faster than MCMC but can be unstable, falling into the wrong position or iterating to infinity and beyond.

5.2.5 Batch Process Data

The large aperture localiser was designed to automatically process large datasets. To do this Select **Batch Process** and tick the **Batch Localise box** and ensure **All Data** is selected in the **Data Options** drop down menu**.** Next select **Settings** (cog symbol)

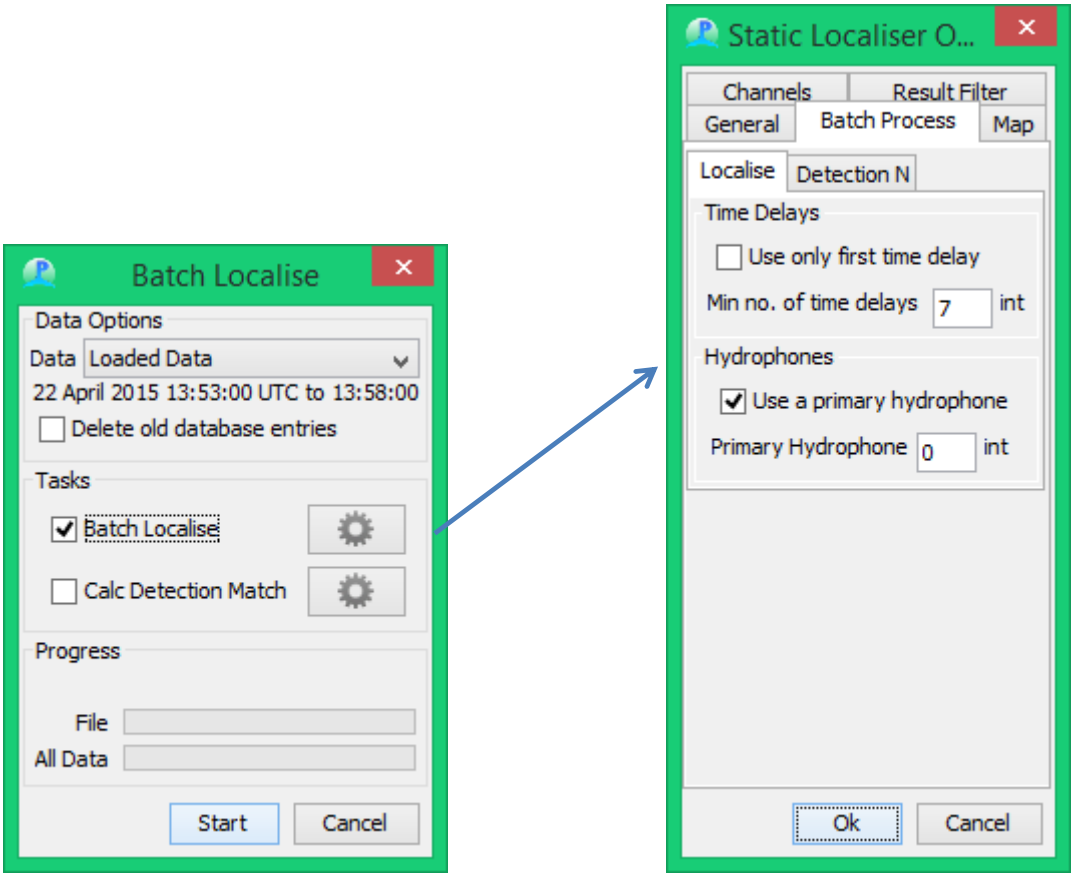


Figure 54. In viewer mode batch processing can be performed to reanalyse data or perform complex computational tasks.

In the **Batch Process** Tab select **Use a primary hydrophone** box and set the **Primary Hydrophone** to be the mid depth hydrophone in the array (Figure 54). Using a primary hydrophone means that only one channel is used for primary detections. This ensures that localisations are not repeated across different channels.

Set the **Min no. of time delays** to 4.

Min. number of delays specifies the minimum number of channels which must have a click detection within the possible time window for the localisation calculation to proceed?????

Note the number of time delays is related to the number channels by

$$N_{ch} = 1 + \frac{\sqrt{1 + 8N_{\tau}}}{2}$$

were N_{τ} is the number of time delays and N_{ch} is the number of channels. There is no point in localising a click which was only detected on one other channel, as this only provides one time delay which would locate the animal on a hyperbolic cone of infinite size which is not useful localisation information for this application.

Next select the **General** tab and ensure that **Display only lowest χ^2 value** is selected Figure 55. This means that only the localisation result with the lowest chi squared value will be saved for those clicks which have multiple possible delay combinations.

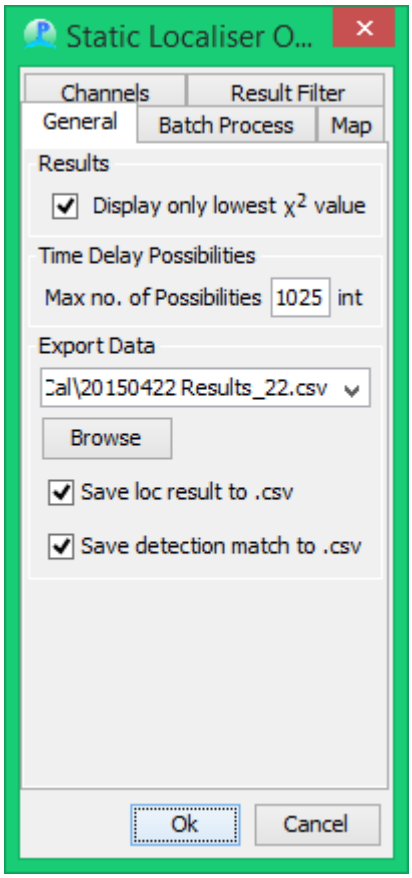


Figure 55. Large Aperture Localiser settings dialog

In the **General** tab there is the choice of exporting data to a spread sheet. Data are automatically saved to a database; however it can be convenient to quickly open data in Microsoft Excel or Open Office. Tick **Save to.csv** and specify a path using **Browse**. Click **OK**.

Now click **Start** and wait until the data processing is complete.

5.2.6 View results.

When the batch processing has finished a spreadsheet will have been created which contains all localisation results. These will also have been saved to the database. Using Microsoft Excel or import in MATLAB and select the **chi2** column and filter out any values above 3 (these are unlikely to provide reliable locations). Now selected the **z** column and plot a graph of results with time. Fragments of porpoise tracks should be visible. Some examples of data processed using different localisation algorithms are shown in Figure 56 and Figure 57.

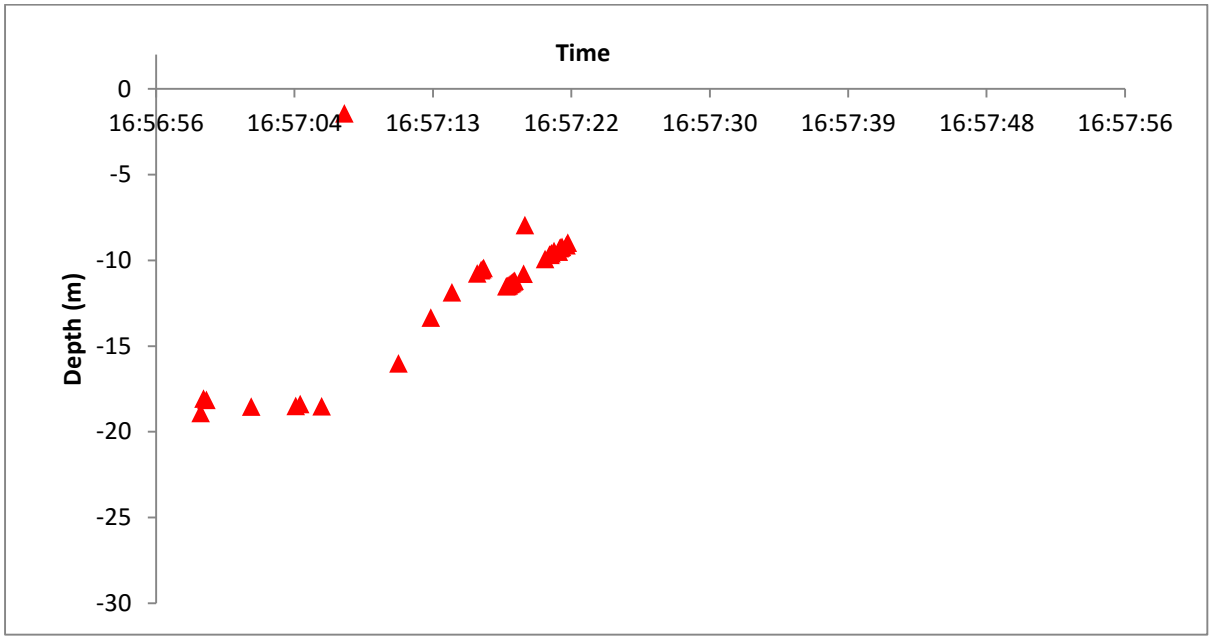


Figure 56. Your results with the Simplex algorithm should look something like this graph.

This concludes the process of localising the depth and range of animals form raw .wav files. Note that MCMC produces slightly better results. MCMC, although computationally demanding, fleshes out an accurate probability distribution for each localisation. It also appears to be less prone to 'running away' to infinity, which often happens with the simplex algorithm. This means MCMC produced both more results and results which have accurate error calculations.

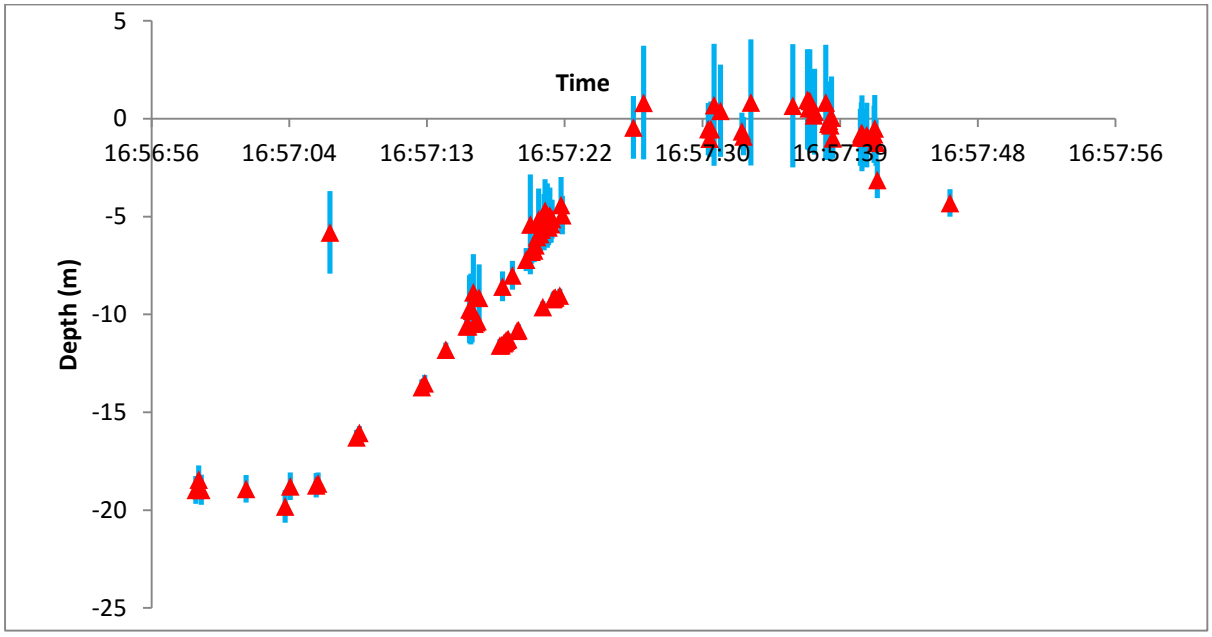


Figure 57. MCMC captures more of the dive.

This concludes the process of localising the depth and range of animals from raw .wav files.

5.2.7 Importing Hydrophone Locations

In many situations you may want to model how the PLABuoy array moves underwater and use a time series of hydrophone locations in the calculation rather than the assuming the vertical array and tetrahedral array remain perfectly vertical and static.

Hydrophone locations can be readily imported into PAMGUARD.

Choose **Settings->Hydrophone Array->Import Hydrophone Data…**This will open up the import dialog (Figure 58). Click the Browse button and select a .csv file of hydrophone locations. This could either be an output file from the PLABuoyHydrophone software detailed in section 4.1.2 or could be a file created using a custom program or script.

The format of a .csv file for import into PAMGuard is as follows: (x,y,z=(0,0,0) is referenced to (0,0,0) in the PAMGaurd array manager)

Time in excel date number	Channel 0 x (m)	Channel 0 y (m)	Channel 0 z (m)	Channel 0 x error (m)	Channel 0 y error (m)	Channel 0 z error (m)	Channel 1 x (m)	Channel 1 y (m)	… *etc.*
42116.55	-0.71047	-9.66806	3.186644	0.01	0.01	0.01	3.477375	-8.51075	…
42116.55	-0.644	-9.86795	2.908774	0.01	0.01	0.01	3.565267	-8.79141	…
…	…	…	…	…	…	…	…	…	…

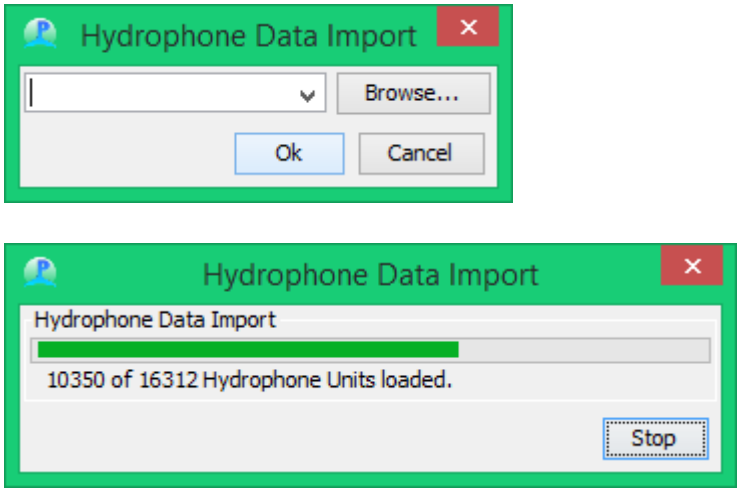


Figure 58. Importing a time series of hydrophone positions.

5.2.8 Geo referencing 3D localisations

In order to geo-reference localisation the geographical location of the array must be known. For this PAMGuard requires a time series of latitude and longitude values.

To geo reference locations in the localiser the GPS module must added to PAMGuard.

Go to **File-> Add Modules-> Maps and Mapping** and add **NMEA Data Collection**, **GPS Processing** and **Map**. A Map tab should appear in PAMGuard (Figure 59).

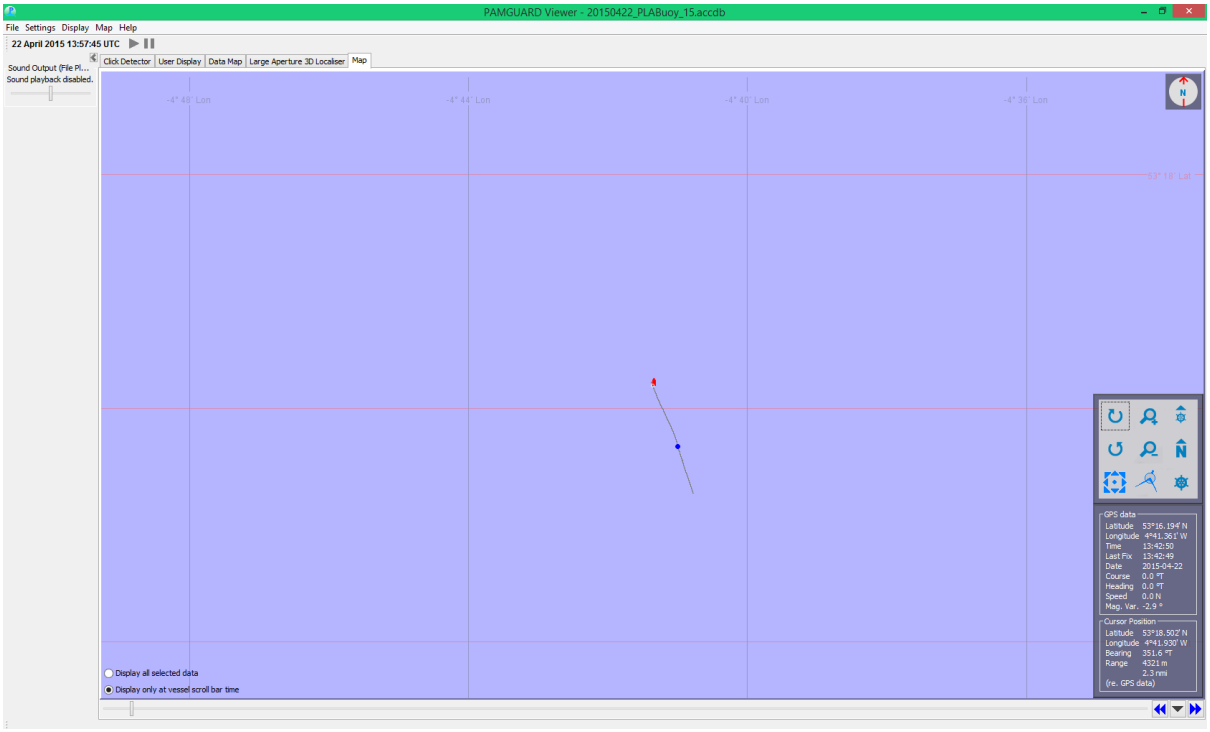


Figure 59. The Map module in PAMGuard allows users to view recorded GPS data.

To add **GPS data go to Detection->GPS->Import GPS Data….**

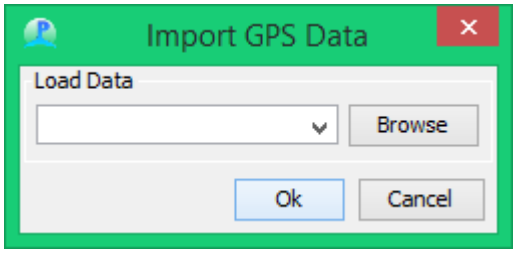


Figure 60. How to select a GPS file in PAMGuard.

In the dialog box open the NMEA file in the Exercise 4 folder (The PLABuoy saves these files as .txt files). A dialog box will appear asking you to select a date.

Ignore this, tick **Use only GPRMC strings** and click **OK** (Figure 61).

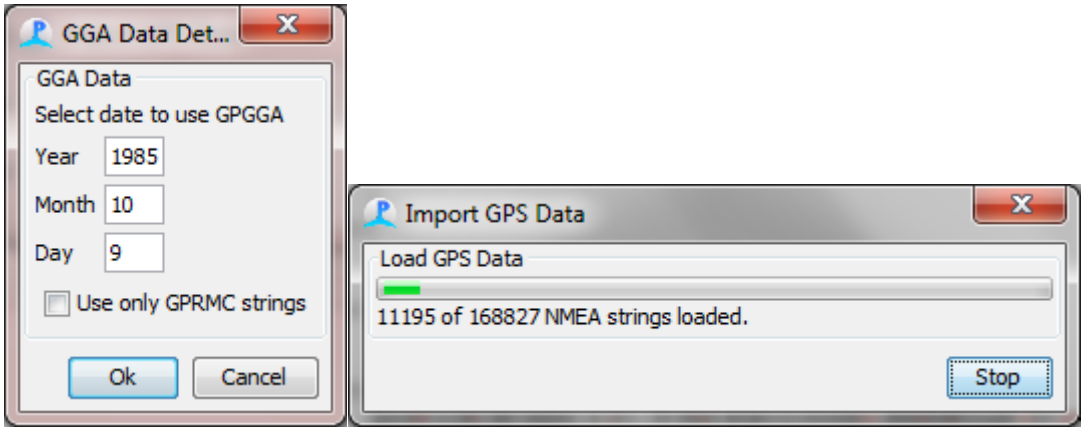


Figure 61. Loading GPS data into PAMGuard.

Wait until the GPS data has loaded

Select the map. A “GPS track” should be visible (Figure 62). Use the slider to change time which will also move the position of the buoy. It may be necessary to use the scroll bars to load more hours of data into memory.

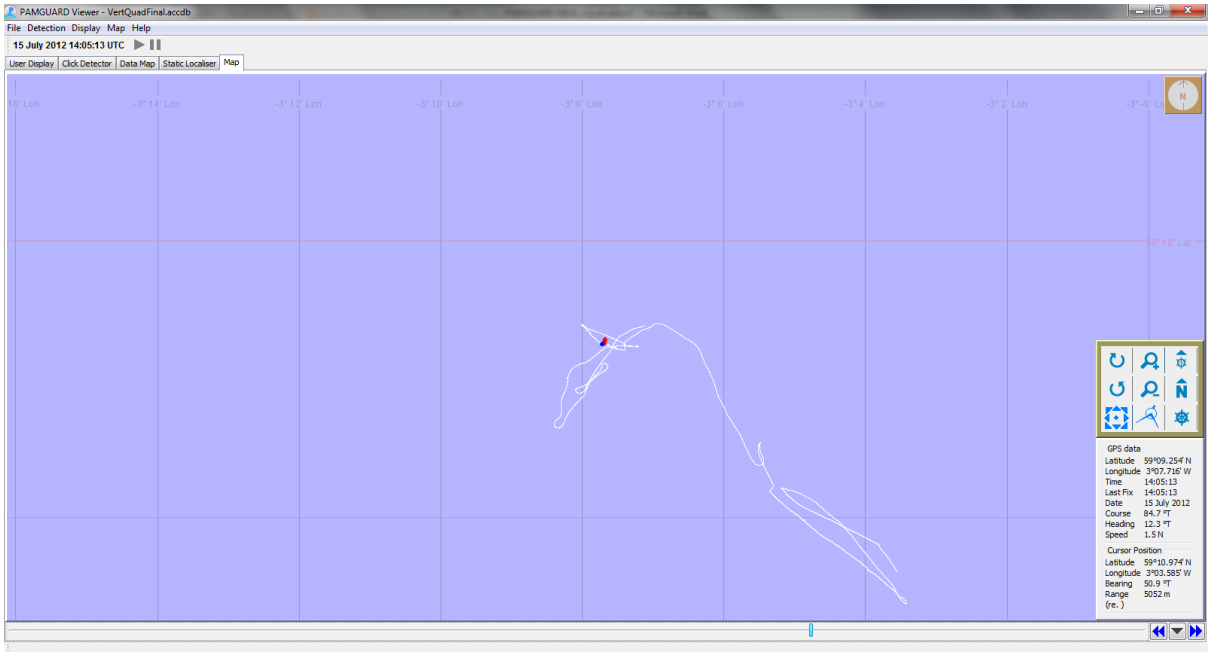


Figure 62. After successfully importing GPS data it is possible to view the PLABuoy tracks and heading in the PAMGuard map module.

Now select the Large Aperture Localiser module and find a suitable porpoise detection. In order to calculated a 3D position data from the tetrahedral array on the PLABuoy must also be used.

Go to **settings** and select all channels then localise using one of the algorithms. A realistic location should appear in the results panel (Figure 63). When batch processing latitude and longitude and a reference latitude and longitude will be recorded in the database and .csv file. The latitude and

longitude are the calculated locations of the sound source (vocalising animal) and the reference latitude and longitude is the position of the array at that time.

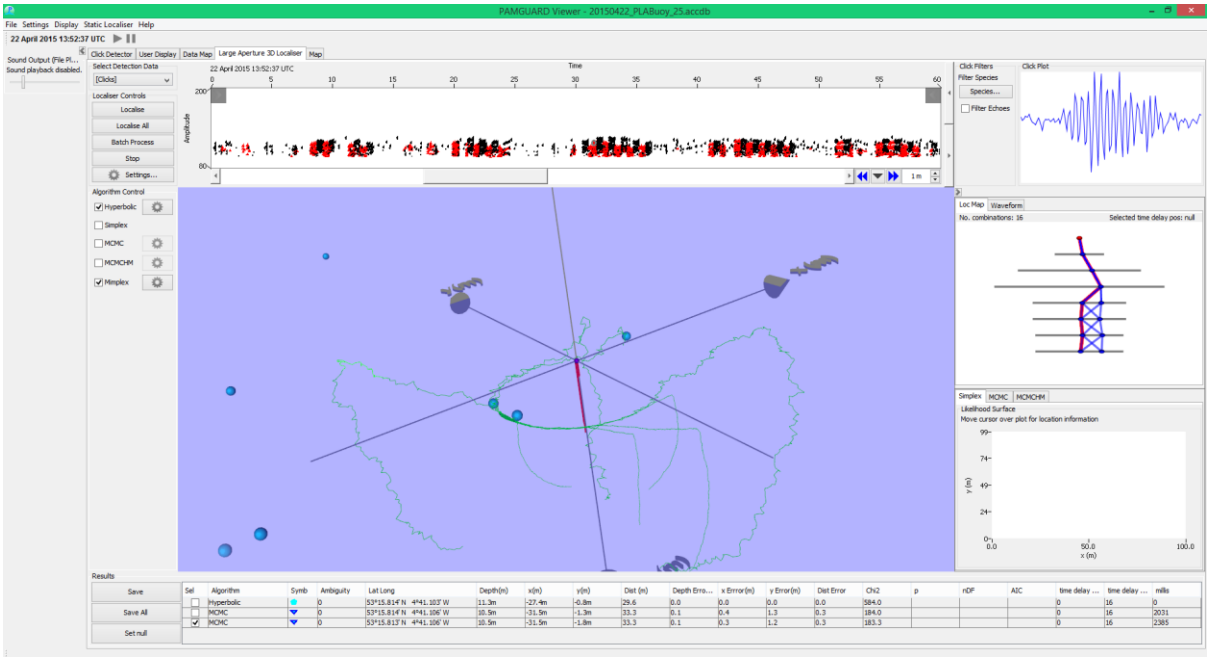


Figure 63. The Large aperture localiser module will automatically import GPS data and convert localisation results to a real latitude/longitude.

5.3 PLABuoy analysis

Sections 5.1 and 5.2 are aimed at familiarising users with the Large Aperture Localisation module rather than being a step by step guide on analysing PLABuoy data. To analyse PLABuoy data the following steps are required.

1) Extract clicks from binary files as described by in section 5.1
2) Set up the localisation module as described in section 5.2.1.
3) Create a time series of hydrophone positions from sensors located on the PLABuoy using the PLABuoyHydrophone software. Import the hydrophone positions into PAMGuard as detailed in section 5.2.7.
4) Import GPS data as detailed in section 5.2.8.
5) Check a few localisations manually to as shown in sections 5.2.2 and 5.2.3 to check all settings are correct.
6) If everything appears to be working, batch process the dataset as described in 5.2.5.