Proceedings of the Institute of Acoustics

DOLPHIN WHISTLE CLASSIFICATION WITH THE 'DOLPHIN' SOFTWARE

C. Sturtivant & S. Datta

Underwater Acoustics Group, Electronic & Electrical Engineering, Loughborough University

1. INTRODUCTION

Identification and re-identification is often an important part of field studies of dolphins. This task has traditionally been accomplished with photographic identification techniques, although it is also possible to use acoustic methods to obtain the same information. The hypothesis that dolphins have a 'signature' whistle was proposed over thirty years ago [1], and suggested that the majority of dolphin whistles carried the identity of the vocalising animal. Although recently questioned [2], the more current proposal still maintains that whistles carry identity information, but in the wider context of groups of animals. Such identifying whistles have been found for several dolphin species as well as for the killer whale [3,4,5,6,7], and so this method of identification potentially has applications to a wide number of toothed cetaceans.

The Underwater Acoustics Group at Loughborough University has developed software, named 'Dolphin', for quantitative comparison of dolphin whistles [8,9,10]. The software can identify parts of recordings containing whistle-like sounds, extract their frequency-time-intensity contours, and then apply automatic pattern recognition techniques to classify them against previous whistles. These techniques have the benefits of being both objective and quantitative, and provide a probability that any candidate whistle belongs to each existing class, or to some new class.

The mathematical background to the software routines has been presented elsewhere in these proceedings [11]. This paper will describe and explain the use of the 'Dolphin' program, with an example of how a typical whistle contour is extracted from background noise, encoded, and classified.

2. HARDWARE REQUIREMENTS

The software was written to execute on a standard IBM PC, with no additional custom hardware. The platform that is currently used is a 133 MHz PCI Pentium running Windows 95, with a Matrox Mystique graphics card and a Sound Blaster 16 card. The development system for the software was a 486 DX 50 with Sound Blaster 16 card, although the software is designed to run on a platform containing an Intel 386 processor or later.

The development environment was the Gnu GCC C++ compiler, ported to the PC as DJGPP, and the GRX graphics extensions. The compiler produces 32-bit code, and was chosen since this runs considerably faster than 16-bit code required for pre-386 processors. Many of the calculations made in the program are computationally intensive, and it is doubtful that processors slower than a 486 would be suitable hosts for the software.

The choice of the Sound Blaster 16 as the sound card for the software was due to its wide commercial availability, and its ability to sample at sufficiently high sample rates (44 kHz) for dolphin whistles and at a suitable resolution (16 bits). Although better sound cards are readily available today, this seemed the best choice at the beginning of the project. The software is written modularly, so that should support for another sound card be required it could easily be incorporated into the program.

3. SOFTWARE FRAMEWORK

The whistle analysis process fell into several inter-related parts, mostly as a sequential process, but requiring some transfer of data between different modules in a more unstructured way. The programming language C++ was chosen, which allows a modular approach to be taken with programming whilst still allowing fast execution of the resulting program. A graphical user interface was required so that information from different parts of the program could be displayed simultaneously on the screen. At the start of development, no general windowing package was available for this compiler, and no links were available to enable it to use systems such as Microsoft Windows. For these reasons it was decided to write a simple windowing system that could be used for other software packages written with the compiler.

The windows system was designed around just simple areas of the screen, with additional elements being built on top of that in a hierarchical structure. Objects could contain other objects, and the tools that were made available included variable text boxes, framed boxes, push buttons, toggle buttons, user input boxes (text and/or numeric), pull down menus, and composite windows. Thus, the spectrogram of a whistle could be displayed inside a window on the screen and other information displayed concurrently.

4. SOFTWARE IMPLEMENTATION

4.1 Sound File Data Formats

Several different data formats were needed for storing the whistles at different stages of processing. Initially, the raw signal data was read from a file. This information was stored as a stream of 16-bit numbers, which corresponds to the 'raw' format for many sound file manipulation utilities. Although the RIFF, or 'wav' format is certainly more widely used, these files can easily be converted to 'raw' files.

After loading, the sound signal is converted to a spectrogram using a fast Fourier transform (FFT) routine. Since this format is the one most often used in the analysis routines, a file format was developed for saving the spectrogram information to disk as a '.fft' file.

Once the whistle's time-frequency contour has been extracted from the surrounding spectrogram, it can be encoded as a string of frequency-time-intensity tuples instead of the entire spectrum. This format normally reduces the file size by a factor of 100 (e.g. a 'typical' whistle of 1.2 seconds was stored in an '.fft' file of 185 kb, but a contour encoded '.ctr' file of 1.5 kb).

The next stage of contour processing is to encode it as a series of 'segments' according to general shape. The contour in each segment is modelled by a quadratic equation of the form:

$$y(x) = a_0 + a_1 x + a_2 x^2 \quad \text{..........(1)}$$

Since this form very much reduces the data requirements for whistle representation, and also forms the representation used immediately prior to the classification process, a file format for an encoded whistle contour '.ewc' was produced. This format contains the most compact representation of the whistle's contour (e.g. the previous 1.5 kb '.ctr' file is encoded in 344 bytes).

4.2 Viewing the Whistle Signal

The raw whistle signal is read in as a series of amplitudes from the file. This unprocessed information can be viewed in a window on the screen in the standard waveform representation (Figure1). Two slide bars are available for repositioning the centre of the window to any point along the signal, and both the amplitude and time scales can be modified in pre-set increments or to any specific value. The side of the signal window shows scales for both the time (in seconds or sample number) and amplitude (between −1 and +1).

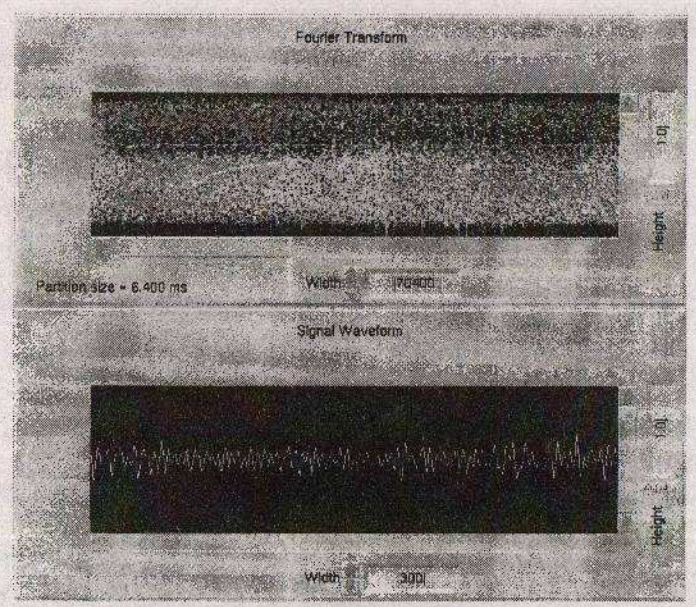

Figure 1: Waveform and FFT windows

Since it is generally quite difficult to separate simultaneous signals from the raw data, the time-amplitude data was passed through a discrete fast Fourier transform routine so that it could be represented in a time-frequency-intensity manner. Tonal signals become distinct from more broadband sounds when this representation is used. A window is used to display the transformed signal that is similar to the one used for the raw signal. The scales in this case are time on the x-axis, and frequency on the y-axis. The intensity can be represented in three different ways using a black to white, a colour-cycling, or simple two-colour palette.

No information on the sample rate at which the file was recorded is stored in the raw signal file. Since this affects both the frequency scale in the FFT window, and the time scales in both FFT and signal windows, a default of 44.1 kHz is used unless the user specifies a different value. Similarly, the time partition for the FFT defaults to 256 sample, giving 128 equally spaced frequency bins displayable on the screen. This can be reduced to increase the time resolution or increased to provide better frequency resolution (each at the expense of the other parameter, of course).

It is possible to save the FFT data once it has been derived from the raw signal. Similarly, one can load the FFT data and recalculate the initial signal from it. The user can supply the format in which he wants to save to the data file from a drop down menu.

4.3 Background Noise Filtering

It is possible for the human eye to determine which parts of the spectrogram consist of background noise and which are the required whistle signals, but any computer algorithm for this task would need to search forward and backward through the spectrogram with quite complex calculations to determine which part was noise. Far easier is to develop an algorithm that reduces background noise, and then to have a simpler whistle contour following algorithm.

The algorithm of choice for noise reduction leaves a low intensity 'trough' around any tonal components in the spectrogram (Figure 2). The equations employed to achieve this filtering are detailed elsewhere in these proceedings [11].

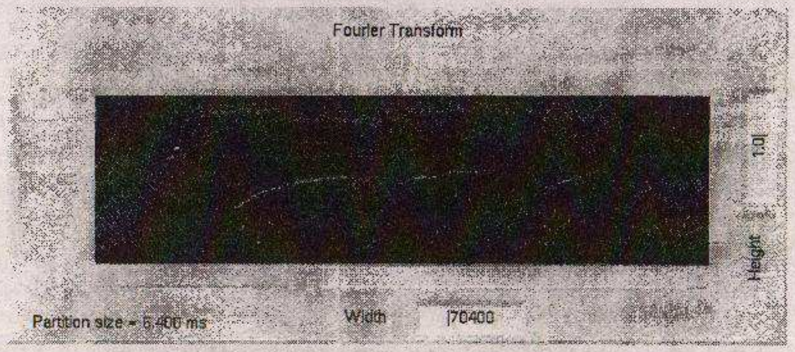

Figure 2: Whistle spectrogram after background noise filtering

4.4 Contour Extraction and Encoding

If we visually trace a whistle's contour from a spectrogram, we take account of which way the contour was heading previously when judging where it goes next. This contextual information improves contour following performance considerably over a straight 'ridge-following' technique, and it was included in the contour extraction mechanism used in the software. A 'notional point' was used to trace through the contour, the motion of which was controlled by a number of parameters (Figure 3).

Our notional point moves through the spectrogram with the aim of following the track of highest intensity. Each point on the spectrogram immediately in front of the point exerts a pull proportional to its intensity towards itself. Thus, when no account is made of any previous direction of travel, the notional point becomes the point that produced the strongest pull.

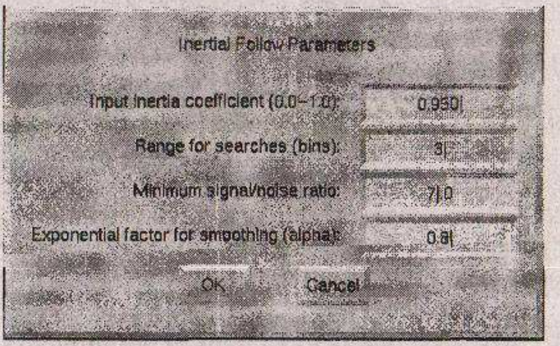

Figure 3: Parameters and typical values for the inertial following algorithm

The 'inertial coefficient' shown in Figure 3 determines how much weight is applied to the previous direction of travel when calculating the new direction the notional point should take. Equation 2 shows the way that the direction of travel is calculated.

$$V'_{xy}(k,t+1)=\alpha_{inertia}V_{xy}(k,t)+(1-\alpha_{inertia})\left(\frac{\Delta}{|\Delta|}\left|f'(k,t+1)\right|^2\right) \qquad (2)$$

If the inertial coefficient is set to 1.0, the direction of travel V stays the same as it was for the previous time partition. If it is 0.0, the direction of travel becomes wholly dependent on the direction to the new point (represented by the Δ component) and the intensity at that point. By varying the inertial coefficient between 0.0 and 1.0, it is possible to specify the sensitivity of the tracing algorithm to changes in direction of the contour. The next point on the contour is the point that results in the largest magnitude for the direction of travel vector, V.

The algorithm only searches within a set number of frequency bins of the previous position when searching for further points on the contour. This reduces the time taken to trace the contour, since dolphin whistles very rarely have sudden changes in frequency. However, some whistles may contain rapidly changing frequencies, so the user can alter this search range.

The minimum signal to noise ratio sets the point at which the algorithm is to finish tracing the contour. The same variable is used to determine when a contour begins or restarts after a gap. However, the background noise in a spectrogram sometimes masks the signal for one or two time partitions. In order to overcome this, an exponential smoothing factor was introduced which takes a running average over the previous partition (Equation 3). S in the equation indicates the current spectrum. Values for α in the equation were between 0.0 and 1.0, similarly to the inertial coefficient used in Equation 2. Values near 1.0 produce a long time average, and those near 0.0 a short one. Spectrum smoothing is used for determining both the start and the end of the contour.

$$S'_{ave} = \alpha S_{ave} + (1-\alpha)S \qquad (3)$$

Once the contour has been extracted from the spectrogram, it is encoded in a series of segments. Each of these segments contains areas where the contour is generally rising, falling, flat, or temporarily absent. A segment's contour is then modelled by a quadratic curve (Equation 1), with the origin reset for each segment, allowing them to be compared easily. An extracted whistle contour and the same contour after encoding can be seen in Figure 4. Encoding is carried out automatically in the program with no user intervention required.

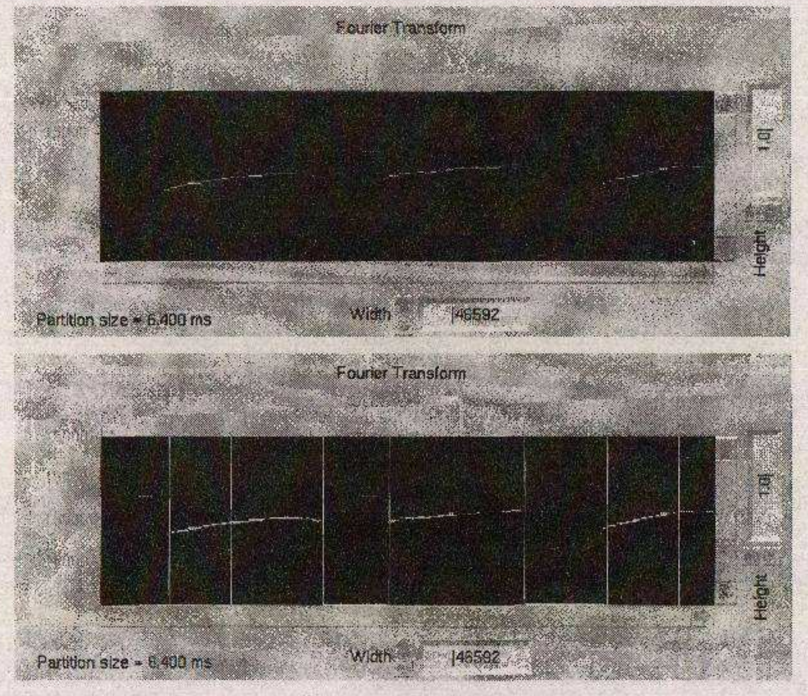

Figure 4: Isolated whistle time-frequency contour before and after encoding

4.5 Classification

Classification is also a predominantly automatic activity from the user's point of view. The software can provide listings of the classes already constructed, and their component whistles, or can find the most probably class for a whistle. When an encoded whistle is to be added to a class, the software calculates the top five most probable classes for it, including the probability it belongs to none of them. The user then has the final decision as to which class to assign the whistle. This intervention can be important if, for example, two classes contain quite similar whistles giving membership probabilities of 27% and 33%, and the probability of a new class is 40%. In this situation, the user might wish to assign the whistle to the class with the 33% probability rather than assigning it to a 'new' class, since the combined probabilities of the two existing classes is 60%, outweighing the 40% for the 'new' class assignment. An example of a class assignment window for a whistle is shown in Figure 5.

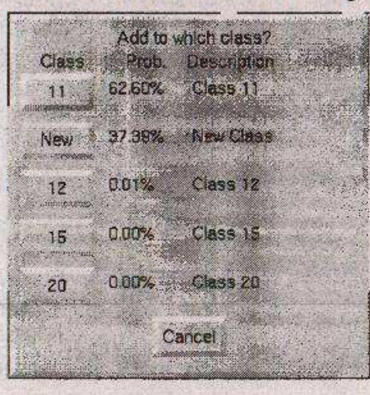

Figure 5: Class assignment prompt, showing top five classes

5. FURTHER ANALYSIS

Many recordings contain whistles that do not seem to be associated with group or individual identities. In these cases one cannot examine the probability of one whistle belong to an existing set of classes and determine if that dolphin's group is the same as that previously encountered. Instead information from a number of whistles must be classified for each group, and the similarity in class descriptions used to determine how similar the two sets of whistles are.

	A Classes	B Classes
Group A	5.72	1.90
Group B	2.46	9.56

$\chi^2 = 3.784$, 1 d.f., $p = 1.75\%$

Table 1: Example table of expected whistle distribution into sets of classes.

An example might be for two groups of dolphins to have been recorded: one in the morning and the other in the afternoon. The distinctiveness of the two groups must be assessed. This question can be answered by constructing separate sets of classes for the morning and afternoon groups. The similarity between the groups can be measured by assessing the distribution of the whistles throughout the classes. If the groups are actually the same, then the whistles from the morning group when sorted into classes for the afternoon group whistles should fall into a similar distribution, and vice versa. The software provides the probabilities of a whistle's membership in an existing class or some as yet undiscovered class. So the probabilities can be summed to form a table showing expected number of whistles from the two groups falling into each of the two sets of classes.

In the example shown in Table 1, the sum of membership probabilities for classes formed by whistles in group A was 5.72 for whistles from group A, and 2.46 for whistles in group B. The sum of probabilities for classes from group B whistles was 1.90 for whistles from group A, and 9.56 for whistles in group B. If these two groups consisted of the same dolphins, they should have the same distribution of whistles between the set of classes for A and B. A chi-squared analysis indicates that this is unlikely, and could happen by chance with a probability of 1.75%. Thus, on a comparison of whistle types, there are significant differences at the 5% level between the two groups.

6. CONCLUSIONS

An analysis software tool has been developed that to a large extent can automate the task of pattern recognition of dolphin whistles. Several species of dolphins use identifying whistles, giving this tool a wide applicability to studies of reoccurrence of individual groups. In addition to classifying whistles into individual classes, the results from the software can be used to provide quantitative evidence for the similarity or dissimilarity of two groups.

The analysis techniques presented here still require some human intervention, but a fully automated system that can extract whistle signals from live data and assign it into a class based on its structure requires more research, but is a realistic short-term goal.

7. ACKNOWLEDGEMENTS

The authors gratefully acknowledge the help of the CETASEL project members for providing the data for this research, and the aid of Kristin Kaschner, David Goodson, and Professor Bryan Woodward. Funding for this project was provided by the U.K. Department of the Environment under contract number CR 0129, and the Ministry of Agriculture, Fisheries, and Foods under contract CSA 2270. Much of the preliminary work on this project would not have been possible without the help of several trainers and oceanariums, especially Mr Peter Bloom of Flamingo Land, U.K., and Mats Amundin and Susanne Hultman of Kolmårdens Djurpark, Sweden.

8. REFERENCES

[1] M C CALDWELL & D K CALDWELL, 'Individualized whistle contours in bottlenosed dolphins (*Tursiops truncatus*)', *Nature* (London) **207**, 434–435 (1965).

[2] B MCCOWAN & D REISS, 'Quantitative comparison of whistle repertoires from captive adult bottlenose dolphins (Delphinidae, *Tursiops truncatus*): a re-evaluation of the signature whistle hypothesis', *Ethology* **100**, 194–209 (1995).

[3] M C CALDWELL & D K CALDWELL, 'Statistical evidence for individual signature whistles in Pacific whitesided dolphins, *Lagenorhynchus obliquidens*', *Cetology* **3**, 9 pp. (1971).

[4] M C CALDWELL, D K CALDWELL, & J F MILLER, 'Statistical evidence for individual signature whistles in the spotted dolphin, *Stenella plagiodon*', *Cetology* **16** (1973).

[5] J K FORD & H D FISHER, 'Killer whale (*Orcinus orca*) dialects as an indicator of stocks in British Columbia', *International Whaling Commission Scientific Committee Report* SC/JN81/KW8 (1981).

[6] M E DAHLHEIM, 'Signature information in killer whale calls', *Whalewatcher* **15(1)**, 12–13 (1981).

[7] J K FORD & H D FISHER, 'Group specific dialects of killer whales (*Orcinus orca*) in British Columbia', In *Communication and behavior of whales* (R. Payne, ed.), AAAS selected Symposia Series, Westview, Boulder, Colorado, pp. 129–161, (1983).

[8] C R STURTIVANT & S DATTA, 'Techniques to isolate dolphin whistles and other tonal sounds from background noise', *Acoustics Letters* **18(10)**, 189–193 (1995).

[9] C R STURTIVANT & S DATTA, 'The isolation from background noise and characterisation of bottlenose dolphin (*Tursiops truncatus*) whistles', *Journal of the Acoustical Society of India* **23(4)**, 199–205 (1995).

[10] C R STURTIVANT & S DATTA, 'An acoustic aid for population estimates', *European Research on Cetaceans — 11* (1997).

[11] C R STURTIVANT & S DATTA, 'Automatic dolphin whistle detection, extraction, encoding, and classification,' (This volume).