Proceedings of the Institute of Acoustics

AN IMPROVED COMPUTER MODEL OF AFFERENT NEURAL PROCESSING FROM THE COCHLEA TO DORSAL ACOUSTIC STRIA

Michael J. Pont[1,2] (1) & Seyed J. Mashari (2)

(1) Department of Engineering, University of Leicester, Leicester, UK.
(2) Department of Computer Science, University of Sheffield, Sheffield, UK

1. INTRODUCTION

In recent years, computer models of the mammalian auditory nervous system have come to be widely accepted as a tool for investigating the relationship between physiology and overt behaviour (e.g. Damper *et al.*, in press). We have previously described a comprehensive computer model of afferent neural processing from the cochlea to the dorsal acoustic stria (Pont and Damper, 1989; 1991; 1992). Originally used to explore the representation of initial stop-consonants (e.g. Pont and Damper, 1992; Mashari and Pont, this meeting) in the dorsal cochlear nucleus (DCN) and auditory nerve (AN), this model has also been sucessfully used as a "front end" for speech recognition (Mashari and Pont, in preparation).

We are currently embarked upon a major programme of work aimed at expanding and improving the original simulations: in the present paper we describe our progress to date. We begin by briefly reviewing the organisation and operation of the original model.

2. THE ORIGINAL COMPUTATIONAL MODEL (V1.0)

2.1 The system to be modelled

The AN is the only route by which incoming auditory information can reach the cortical structures known to be involved in later stages of language processing. All afferent (that is, centrally directed) nerve fibres within the AN terminate in the cochlear nucleus [CN] (Palmer, 1987), the first auditory relay and processing station. The CN itself consists of two major sub-divisions, dorsal (DCN) and ventral (VCN). Neurons within the DCN have, in comparison with those of the AN and VCN, particularly complex properties (Young, 1984), and the focus of our modelling efforts to date has been on this division of the nucleus (but see Section 4 below).

The response properties of DCN cells (in the cat) are generally divided into two main classes, the first made up of Type II/III responses, and the second of response Type IV (see Young and Voigt, 1981 for descriptions of class membership criteria). Type II/III units are thought to correspond to small interneurons within the DCN, and Type IV units to the larger fusiform cells, whose axons make up the dorsal acoustic stria (DAS). The DAS is thus the (main) DCN "output" pathway, and it projects principally to the contralateral inferior colliculus.

2.2 The Original Model: v1.0

The primary aim of our modelling efforts has been to reproduce the known responses of the DCN Type IV units to simple tonal and noise stimuli, using a network of simple cell models, interconnected in accordance with published anatomical data.

Our original computational model (v1.0) is detailed fully by Pont and Damper (1991), and will only be described very briefly here. It is shown schematically in Figure 1. The model is coded in Pascal and simulates afferent neural processing up to the level of DAS. The model is "fed" from a 128-channel cochlear filterbank. Cochlear transduction, rectification, logarithmic compression, and two-tone suppression functions are performed at the first stage of the simulation. The 512 artificial neurons employed are simple "point" models, simulating the nerve cells at the level of transmembrane potential.

2.3 Known problems

Our original model has proved to be a useful tool for the investigation of the neural mechanisms underlying certain speech perception behaviours (e.g. see Mashari and Pont, this meeting) and has also been applied successfully as a "front end" to a hybrid speech recognition system (Mashari and Pont, in preparation).

However, as knowledge of auditory system morphology and function accumulates, any model must be subject to continual revision if it is to remain accurate. Although Version 1.0 of our model is able to reproduce accurately the findings from a number of important physiological studies, it is far from being perfect in this regard. Furthermore, it runs quite slowly and is rather cumbersome to use. We are therefore currently involved in long-term project aimed at both extending and enhancing the original model.

As a first stage of that process we describe here the conversion of the conversion of the original model code from Pascal to "C", and some initial enhancements to make the model easier to use.

3. MODEL VERSION 1.1

Here we will refer to the improved version of the model as model v1.1, the original simulation (as described above, and in Pont and Damper, 1989; 1991) being version 1.0.

3.1 Main Changes cf. v1.0

To date, for model Version 1.1, we have re-coded the original simulation in "C" and improved the user interface.

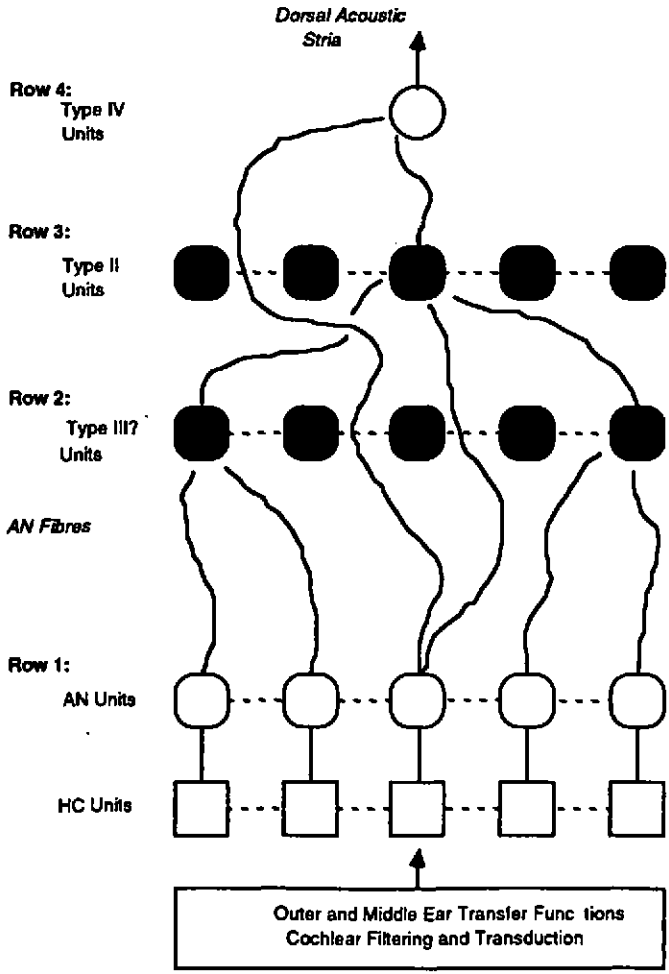

FIGURE 1.
The original (v1.0) model.
(Redrawn from Pont and Damper, 1991, Figure 1.)

Our main reasons for opting to use "C" for our new model were that (a) the language is widely available - it is even distributed "free" with most Unix workstations - and, (b) the language is very flexible.

The conversion process itself, we knew, was likely to be a source of error in its own right, and - despite the inherent similarities between Pascal and "C" - likely to be long and rather tedious. Fortunately, we were able to obtain[3] a copy of the program "P2C", produced by the Free Software Foundation. This Freeware program, available on a number of Bulletin Boards, allows "automatic" conversion of Pascal code into a "C" equivalent.

We ran the P2C program on our original program with considerable success. By way of illustration, a fragment of the Pascal code, and the "C" equivalent are shown in Figures 2 and 3. The first thing to note from the listings is that variable names and comments are maintained in the conversion. The second thing to note is that the structure of the program too remains virtually identical in the "new" version. The result is that - while the "C" code produced may (possibly) be a little inefficient, it is easy to read, and easy to optimise if necessary.

The only area which we had to hand edit substantially was the file access routines, which did not convert very well at all.

3.2 Associated changes

A number of other changes have also been made at this time, the two main ones being:

1. Removal of first eight "DC" channels.

In our original simulation, the first eight (of 128) filters were tuned down to DC levels, and were ignored in the simulations. In this version of the model, we have tidied this response by reducing the frequency range from 100 Hz (cf 50) to 5000 Hz, and the number of channels from 128 to 120. The spacing of the filters remains as before.

The result has been to reduce the number of columns in the model from 128 to 120. Of course, it is still possible to simulate only a small portion of the auditory nervous system; for example, humans have some 3500 inner hair cells, and the model simulates only 120 in the current implementation. In a crude effort to simulate a larger array of cells, we conduct the probabilistic simulations of the cell responses several (usually twenty) times.

2. Allow the user to specify "signal level".

V1.0 of the model has a facility for calculating and reporting the level of any applied stimulus (in "dB SPL" - see Pont and Damper, 1991, for details of this process). Thus if (as is often the case) the user wishes to examine the simulated response to a sound at a specific level, he or she must generally apply any stimulus twice - the first time to measure the level, and the

AN IMPROVED COMPUTER MODEL

```
PROGRAM DCN_NETWORK (INPUT, OUTPUT, hair_cell_data_file,
                         outrow2, outrow3, outrow4, outrow5);

CONST
   n_runs          = 20; { Number of runs }
   dB_required     = 65; { Required signal level }
   sample_rate     = 10; { kHz, sample rate }
   n_cols          = 120;      { No. of columns in cell array }
   n_rows          = 5;        { No. of rows in cell array }

...

{-------------------------------------------------------}

nt[c] := random_uni(100);
    hc_agc_gain[c] := 1.0
    end;

{ DCN data }
FOR R:= 3 TO n_rows DO
    FOR c := 1 TO n_cols DO
        BEGIN

        { Initialise relevant cell_outs to FALSE }
        FOR time := 1 TO syn_delay DO
            cell_outs[r,c,time]:=FALSE

        END;

...

PROCEDURE initialise_network;

VAR
   r, c, t_r, t_c, time : INTEGER;

BEGIN { initialise_network }

reset_random_number_generator := TRUE;

{Initialisation}

FOR c := 1 to n_cols DO
    begin
    samples_since_TypeIV_inhib[c] := TypeIV_inhib_period;
    samples_since_hc_out[c] := 1;
    hc_agc_cou

...
```

FIGURE 2.
Fragment of code for the original model (v1.0) in Pascal.

```
/* Output from p2c, the Pascal-to-C translator */
/* From input file "dcn_network.p" */

#include "/home/hamish/mjp/Pascal-to-C/src/p2c.h"
#include <math.h>
#include <time.h>

#define n_runs                  20    /* Number of runs */
#define dB_required             65    /* Required signal level */
#define sample_rate             10    /* kHz, sample rate */
#define n_cols                  120   /* No. of columns in cell array */
#define n_rows                  5     /* No. of rows in cell array */

...

/*-------------------------------------------------------*/

Static Void initialise_network()
{
  long r, c, t_r, t_c, time;

  reset_random_number_generator = true;

  /*Initialisation*/

  for (c = 0; c < n_cols; c++) {
    samples_since_TypeIV_inhib[c] = TypeIV_inhib_period;
    samples_since_hc_out[c] = 1;
    hc_agc_count[c] = random_uni(100L);
    hc_agc_gain[c] = 1.0;
  }

  /* DCN data */
  for (r = 2; r < n_rows; r++) {
    for (c = 0; c < n_cols; c++) {
      /* Initialise relevant cell_outs to FALSE */
      for (time = 1; time <= syn_delay; time++) {
      cell_outs[r][c][time] = false;

      }
    }
  }

...
```

FIGURE 3.
Fragment of code for the modified model (v1.1) in "C".

second after amplifying or attenuating the signal as necessary. For example, having recorded a sample of speech and wishing to examine the simulated DCN response to this sound at a normal listening level (say 70 dB SPL), the user of v1.0 has first to run the simulation once in order to determine the equivalent signal level (say "24 dB SPL"). Using this information, the user then has to manually alter the applied stimulus (in this case, amplifying it by 46 dB), then re-run the model to obtain the required response. Clearly, this process is cumbersome and time-consuming.

With v1.1, the user can now specify the required signal level, and the model will adjust itself as necessary to generate the desired response. This facility is of particular value when using the model in batch mode, with a number of different sounds.

4. FUTURE WORK

Version 1.1 of our model is intended only as a stepping-stone to the next version (v2.0), which is currently under development.

It is intended that future simulations will take advantage of the massive increase in affordable computer power available even since the original model was first developed three years ago. Our aim is to utilise this power as a means to allow both an extension of the present simulations (adding new areas of the nervous system), and in the resolution of the modelling (improving both the individual cell models, and increasing their number). In particular, no account is taken in the present simulations of neural processing within the ventral division of the cochlear nucleus: this is an area we are currently investigating.

In addition, we have conducted a pilot study[4] investigating techniques of "parallelising" the "C" code described here, for use on an array of Transputers. This study has proved successful, and will continue.

As with the previous version (Pont and Damper, 1989) full details of the revised model, including source-code listings, will shortly be made available in the form of a technical report.

5. CONCLUSIONS

We have described here some modifications to an existing computer model of the mammalian auditory nervous system, and briefly described our plans for developing a new version of the model.

6. NOTES

1 Some of the work described here was carried out while MJP was in the Departments of Computer Science and Psychology, University of Sheffield.

2 Please address correspondence to: Dr. M.J. Pont, Department of Engineering, University of Leicester, University Road, Leicester, LE4 7RH.

3 We are grateful to Mr Malcolm Crawford, University of Sheffield, for his help in obtaining this software.

4 This work was carried out with Dr Peter Croll, University of Sheffield.

7. REFERENCES

Damper, R.I., Pont, M.J. and Elenius, K, (in press) "Representation of Initial Stop Consonants in a Computational Model of the Dorsal Cochlear Nucleus," in , Ainsworth, W.A. (ed.) Advances in Speech, Hearing and Language Processing, Volume 3, JAI Press Ltd, London.

Palmer, A.R. (1987) "Physiology of the cochlear nerve and cochlear nucleus," British Medical Bulletin, 43: 838-835.

Pont, M.J. and Damper, R.I. (1989) "Software for a computational model of afferent neural processing from the cochlea to dorsal acoustic stria," VSSP Technical Report 89/TR2, University of Southampton, UK.

Pont, M.J. and Damper, R.I. (1991), "A computational model of afferent neural processing from the cochlea to the dorsalk acoustic stria", Journal of the Acoustical Society of America, 89 (3): 1213-1228.

Pont, M.J. and Damper, R.I. (1992), "Exploring the rôle of the dorsal cochlear nucleus in the perception of voice-onset time," in , Ainsworth, W.A. (ed.) Advances in Speech, Hearing and Language Processing, Volume 2, JAI Press Ltd, London, pp. 135-166.

Young, E.D. (1984) "Response characteristics of neurons of the cochlear nuclei," in, Hearing Science: Recent Advances, Ed. C.I. Berlin (College-Hill, San Diego, CA), pp.423-459.

Young, E.D. and Voigt, H.F. (1981) "The internal organization of the dorsal cochlear nucleus," in, Neuronal Mechanisms of Hearing, Ed. J.Syka and H.F. Voigt (Plenum, New York), pp.127-133.