PATTERN RECOGNITION FOR FORMANT TRAJECTORIES USING THE HOUGH TRANSFORM

M. A. TOWNSEND & M. B. SANDLER

Dept. Electronic & Electrical Engineering
King's College London
Strand
London
WC2R 2LS
ENGLAND

RÉSUMÉ

Ce document décrit une nouvelle application du Hough Transform, et la possibilité du mettage en oeuvre de ceci pour la détection des changements de fréquence dans les roulements de tambour. Les résultats des expériences sont présentés pour des données synthétiques aussi bien que réelles, et on peut ainsi voir que jusqu'à 40% des trajectoires peuvent être identifiées avec une grande précision.

ABSTRACT

This paper describes a new application of the Hough Transform, and its implementation for the detection of frequency changes in drum beats. Test results are presented for synthetic and real data, and it is shown that up to 40% of possible trajectories can be identified with a high degree of accuracy.

1. INTRODUCTION

This paper continues previous work, which described a method and implementation for the extraction of resonant frequencies (formants) from high order autoregressive (AR) models.[1] The algorithm itself was developed for the particular purpose of finding these formants in the acoustic waveforms of percussion instruments (drums).

The basic AR model is very similar to that used in speech analysis, although there are some significant differences between the two types of waveform, requiring different applications of Linear Predictive Coding (LPC) theory. First, only a single explosive input is required to re-synthesise each drum beat, as opposed to a train of pulses, or continuous noise in speech. Also, many more formants are needed for a high quality rendition of the synthesised waveform, and these formants change frequency slowly over time with controlled variation.

The procedure finds the factors of a polynomial using signal processing techniques, rather than numerical methods, as the latter become unreliable for high order models. First estimates for formant positions are made by applying well known Fourier Transform derived measures to an LPC model of a drum beat[2,3], then the magnitude surface of the z transform in a region around these estimates in the unit circle is evaluated to define precise positions. Previous work has shown well over 90% of possible formants can be located with reasonable accuracy.

As in LPC speech analysis, the original time signal must be analysed in short segments typically 30ms long. This provides a piece-wise linear model of a non-linear system. Unfortunately when very short time windows are used, together with high order models, large amounts of data are produced. It is therefore useful to find a way of describing these variations in frequency over time with a continuous function, rather than the piece-wise model. Traditional methods used for speech signals are easily able to track the movements of formant positions as there are only ever 4 or 5 frequencies to deal with. Several algorithms have been shown to successfully extract individual trajectories by assuming frequencies in any time frame to be near where they were in the previous frame[4,5]. The time/frequency plot acquired from a 200th order drum model in Fig. 1 suggests that identifying tracks in such dense data requires a more complex approach. Furthermore, the tracking algorithm must be able to deal with not just noisy, but missing data, as the extraction method may only find 80 - 90% of formants. This is basically a pattern recognition problem, and in image processing a standard solution is the Hough Transform.

2. HOUGH TRANSFORM

The Hough Transform[6] (HT) is now widely established as a simple technique for detecting complex patterns of points in binary images. This is achieved by determining values of parameters which are specific to these patterns. The most common use of the HT is to detect straight lines in an image, and this approach best illustrates the main concepts. Consider detecting a set of image points (x,y) which lie on a straight line. They may be defined by a function, f :

$$f((m,c)(x,y)) = y - m.x - c = 0 \quad (1)$$

where m and c are the two parameters, the gradient and intercept, which fully describe the line. Equation (1) gives, for any image point, a set of all the parameter combinations (m,c) which are valid for that point. i.e. all the straight lines which pass through (x,y). Thus, the mapping is one to many from the image space to the parameter space. This mapping is repeated for every (x,y) position, producing a parameter, or accumulator space for the whole image, where 'votes' for particular lines are logged. Points which lie on a common line will intersect at a common point in the parameter space, and the co-ordinates of the parameter point characterise the line connecting the image points. The determination of this intersection then becomes a simple local peak detection problem, rather than a complex global detection problem in the image space.

3. ALGORITHM

For the purposes of this application several significant modifications have been made to the standard HT described above. Firstly, the use of an 'image' is not obvious, in fact the x and y co-ordinates are actually time and frequency, to denote the changing formant frequencies. Secondly, in image processing problems, the HT normally uses a coarsely quantised image space (e.g. 256 x 256 pixels), and a parameter space of a comparable size. This automatically limits the number of possible lines that can pass though any one point. For our purposes the time co-ordinates are quantised to the length of the time windows used (e.g. 30ms). However, this is not true of the formant frequencies, which are to double floating point precision. The number of possible lines passing through any frequency is therefore far too large to apply standard techniques due to computational and accuracy constraints. To overcome this, a method has been devised where each formant in turn is paired with every other formant frequency in subsequent time frames. This approach is actually a variation of the Combinatorial Hough Transform[7], which has been successfully used to improve performance in the presence of noise and decrease computation time. For each formant pair, $(k_1.\Delta x, y_1)$ and $(k_2.\Delta x, y_2)$, where k_n is the position and Δx the length of each time frame, and y_n the frequency, the values of m and c for the straight line connecting the two points can be found by solving :

$$m = \frac{(y_2 - y_1)}{((k_2 - k_1)\Delta x)} \quad (2)$$

$$c = y_1 - (m.k_1.\Delta x) \quad (3)$$

The resulting floating point values of m and c are then quantised into the parameter space, which is of a typical size (e.g. 256 x 256 bins). This particular parameterisation is not preferred in image processing applications as the value of m can range from -∞ to ∞. However, formants are only paired with values in successive time frames, so vertical lines are never encountered, and the slow moving frequencies will not produce large rates of change.

Although the procedures described above refer to detecting straight lines, experimental measurements and other works have shown that the change in fundamental frequency of a drum beat is not linear, but of an exponential form[8]. To model this correctly the formant frequencies are simply analysed on a log scale, while time is left linear. The values of m and c then become octaves/second and log(radians) respectively.

Once the parameter space has been fully determined, the local maxima must be detected. Here, a threshold value may be used to ensure that only very common candidates are chosen. The choice of threshold can be critical as too low a value will mean 'noisy' peaks are identified as lines, and too high a value will lead to some real lines remaining undetected. A statistical measure for automatic selection of the threshold has been developed to counter variations in test data. The mean, μ, and standard deviation, σ, of the votes in the parameter space are calculated, and a threshold of $\mu + 2\sigma$ has been found suitable through empirical methods. This value is then used in a simple nearest neighbour peak detection routine, where each parameter space bin is compared with surrounding bins to detect local maxima. The selected formant tracks may then be recorded as a starting frequency and the rate of change of frequency.

4. IMPLEMENTATION

The algorithm has been implemented in the C programming language on a personal computer equipped with a floating point math co-processor. Graphics have be incorporated into the program to give plots of the original formant data, the accumulator space, the pdf of the accumulator space and the selected formant tracks. The graphics were mainly introduced for algorithmic debugging purposes, and may be disabled.

The main program options include the dimensions of the parameter space, a choice between log and linear frequency scales, threshold value (or automatic selection), and a choice between a 3x3 or 5x5 search grid for local maxima. The program also has the capability to restrict the parameters m and c to user-defined ranges, so lines outside these limits are ignored. Values of c should be restricted to 0 and π radians as a starting frequency must lie in that interval. Similarly, the 'gradient', m, can be limited to +ve or -ve slopes and ranges within those depending upon the input data. The parameter space will only contain bins for

parameter values in the specified limits, which can increase resolution.

5. EXPERIMENTS AND RESULTS

Two classes of experiments were performed, one using synthetic formant data, and one on data obtained from real sound sources.

Synthetic Data

The synthetic tests were used to assess the algorithm's accuracy in locating formant tracks, as its results can be compared to those used to generate the test data. A program was specially written to generate these trajectories. The user provides a starting frequency and a rate of change of frequency, and the formant data for the specified number of time frames is then easily calculated. Other options include a random number generator to select the parameters of tracks, with limits being imposed on both the starting frequency and gradient. Also 'noise' may be added by including a number of random formant positions in each time frame.

Four types of synthetic data were generated; a single track, a single track in 99% noise, 60 tracks, and 60 tracks in 40% noise. Table 1 shows the averages of results obtained from ten experiments on each data type. The limits on the values of m were varied between 0 to $-0.1\omega/s$ and 0 to $-0.5\omega/s$ in steps of $0.1\omega/s$, while c was restricted to 0 and π. A 100x100 and a 200x200 bin parameter space was used.

The algorithm's accuracy was deemed more than adequate for synthetic models, so experiments were initiated using real data.

Real Data

Before analysing data obtained from a drum beat, a sound source with an audible change in frequency was used. A recording of a 'swanee whistle' was made on a high quality DAT recorder. This was transferred to the personal computer, and an LPC model (order 100) was generated using the autocorrelation method. Formant extraction then took place with, on average, 80% of the possible resonances being identified. Simple spectral analysis had shown the fundamental frequency of the whistle to vary from 1.313kHz to 2.156kHz in an non-linear manner over a period of 700ms. The formant data was then analysed using limits of $0 < c < \pi$ and $0 < m < +1.0$ octave/sec to account for the approximate doubling in frequency. A 200x200 bin parameter space was used, with the previously described automatic threshold value for peak identification. The algorithm detected 12 unique formant trajectories, including the fundamental with an average error of 0.124% over the duration of the sound.

A recording of a 'tom-tom' was also made and transferred to the personal computer. Two 200^{th} order LPC models were generated using the autocorrelation and covariance methods, and formant extraction found, on average, 80% of possible resonances. Experiments were carried out with varying ranges of m. The results in Table 2 show that up to 40% of the formant trajectories can be identified uniquely from the data. Fig 2 is an example of just three calculated trajectories from the data in Fig. 1, together with the formant positions closest to them. All other data has been removed and the frequency scale enlarged for clarity.

6. CONCLUSIONS

A new application of the Hough Transform for pattern recognition has been described. Evidence of both the algorithm's accuracy and ability to detect patterns in real and synthetic data has been presented. The algorithm has been shown to identify formant trajectories of both a linear and exponential type. Future work will include detection of trajectory end points and the synthesis of sounds for perceptual analysis. Also, the results of this analysis may be used as a first approximation in an improved formant extraction technique in order to identify values that had been previously undetected.

7. REFERENCES

[1] M. Townsend & M. Sandler, Sept. 1991, IEE Proc. 6th DSP Conf., pp 246-250.

[2] N.S. Reddy & M.N.S. Swamy, Dec. 1984, IEEE Trans. ASSP Vol. 32, No. 6, pp 1136-1144.

[3] R.L. Christensen, W.J. Strong & E.P. Palmer, Feb. 1976, IEEE Trans. ASSP Vol. 24, No. 1, pp 8-14.

[4] J.D. Markel & A.H. Gray, 1976, "Linear Prediction of Speech", Springer-Verlag.

[5] S. McCandless, April 1974, IEEE Trans. ASSP Vol. 22, No. 2, pp135-141.

[6] J. Illingworth & J. Kittler, 1988, CVGIP Vol. 44, pp87-116.

[7] D. Ben-Tzvi & M. Sandler, March 1990, Patt. Recog. Letters 11, pp167-174.

[8] H. Fletcher & I. Bassett, Dec. 1978, J. Acoust. Soc. Am., Vol. 64, No. 6, pp 1570-1576.

	No. lines found	min % error	max % error	av. % error
1 track	1	0.132	0.981	0.423
1 track (99% noise)	1	0.221	1.513	0.762
60 tracks	51	0.147	1.883	0.621
60 tracks (40%noise)	57	0.213	2.172	0.789

Table 1. Results of synthetic data tests.

	Autocorrelation model		Covariance model	
Gradient range (octaves/sec)	No. lines found	average % error	No. lines found	average % error
-0.05 to 0	34	0.126	32	0.131
-0.1 to 0	27	0.130	28	0.134
-0.15 to 0	23	0.151	25	0.140
-0.2 to 0	23	0.158	24	0.149
-0.25 to 0	23	0.162	25	0.153
-0.1 to -0.05	32	0.131	29	0.128
- 0.15 to -0.05	27	0.143	26	0.136
-0.2 to -0.05	23	0.149	22	0.136
-0.25 to -0.05	25	0.152	24	0.128
-0.15 to -0.1	26	0.128	25	0.135
-0.2 to -0.1	25	0.136	25	0.137
-0.25 to -0.1	23	0.155	24	0.141
-0.2 to -0.15	25	0.120	23	0.151
-0.25 to -0.15	25	0.137	22	0.139
-0.25 to -0.2	25	0.134	23	0.140

Table 2. Results of real drum tests

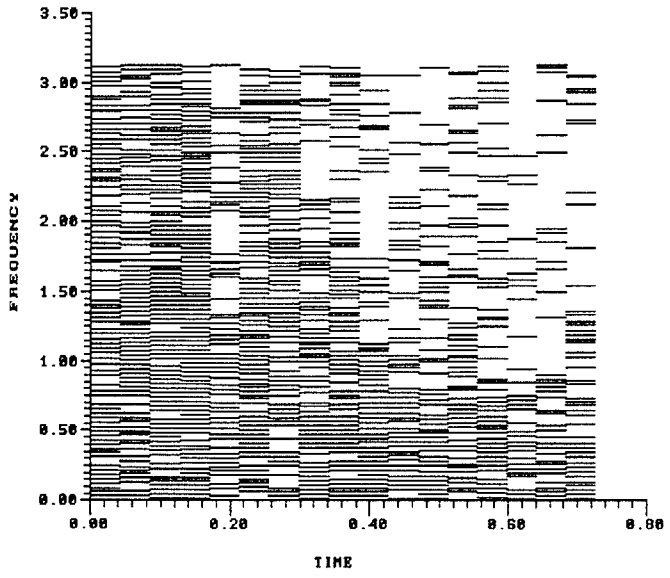

Fig 1. Raw formant data from a 200th order model of a single drum beat modelled with 42ms time frames.

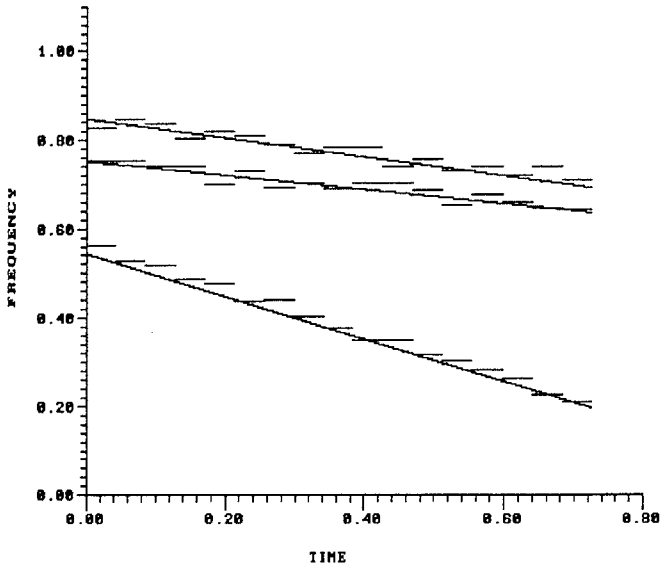

Fig 2. Three formant trajectories with original data.