A Sound Localization and Recognition System using Pulsed Neural Networks on FPGA

Kaname Iwasa, Mauricio Kugler, Susumu Kuroyanagi and Akira Iwata

Abstract— Pulsed neurons are suitable for processing time series data, like sound signals, and can be easy implemented in hardware. In this paper, we propose an aural information processing system based on the human auditory system using a pulsed neuron model and a correspondent implemetation in an FPGA device. Experimental results show that a FPGA based implementation of the proposed system could successfully identify the results faster than a similar software implementation. Noise tolerance experimental results are also presented.

I. Introduction

By the information provided from the hearing system, the human being can identify any kind of sound (sound recognition) and where it comes from (sound localization)[1]. If this ability could be reproduced by artificial devices, many applications would emerge, from support devices for people with hearing loss to security devices.

Two important points to consider when developing such kind of device are the physical dimensions and response speed. A common approach for sound localization is the use of microphone arrays [2]. Although achieving good precision, this method requires the processing of several signal sources and presents large dimensions. Sound recognition applications usually make use of a host computer and a microphone, achieving good results [3]. However, when performing other concurrent tasks, as sound localization or image processing, a powerful host computer is required in order to perform all those processing tasks in real time. This makes the system unusable in daily life.

Neural networks are generic classification methods that can deal with virtually any kind of information. However, standard artificial neuron models (e.g. McCulloch-Pits model) are not appropriate for time series data processing, requiring input signals to be transformed in static vectors by "unnatural" and complex windowing processes, as, for example, the Time Delay Neural Network [4]. Another approach for processing temporal data is the use of Pulsed Neuron(PN) models [5]. This type of neuron deal with input signals on the form of pulse trains, using an internal membrane potential decay as a reference for generating pulses on its output. PN models can directly deal with temporal data, avoiding unnatural windowing processes. Moreover, a PN model can be efficiently implemented in hardware, due to its simple structure. Hence, the complete sound localization and recognition system can be implemented in a single and compact hardware. Futhermore, high processing speeds can be achieved, as PN model based methods are usually highly parallelizable.

Kaname Iwasa is with the Department of Information Engineering, Nagoya Institute of Technology, Gokiso-cho, Showa-ku, Nagoya, 466-8555, Japan (phone: 052-732-2111; fax: 052-744-1355; email: kaname@mars.elcom.nitech.ac.jp).

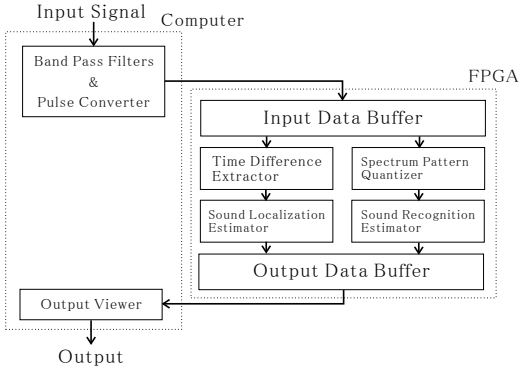

Fig. 1. Basic structure of the proposed aural information system

This paper proposes a hardware implementation of an aural information processing system able to simulataneously locate and identify the sound source. The implementation makes use of a Field Programmable Gate Array (FPGA) device. Due to the use of PN models and the use of similar blocks on the various stages, the final implementation is simple and efficient, as demonstrated by the experimental results.

II. Proposed system

The basic structure of proposed system is shown in Fig.1. This system consists of five main blocks, the frequency-pulse convereter, the time difference extractor, the sound localization ditector, the spectrum pattern quantizer and the sound recognition detector. The sound localization and recognition blocks are based on a Pulsed Neuron(PN) model [5], working in parallel.

The left and right signal's time difference informaiton is used to localize the sound source, while the spectrum pattern is used to recognize the type of the source.

A. Pulsed Neuron Model

When processing time series data (e.g. sound), it is important to consider the time relation and to have computationally inexpensive calculation procedures to enable real-time processing. For these reasons, a Pulsed Neuron (PN) model is used in this research.

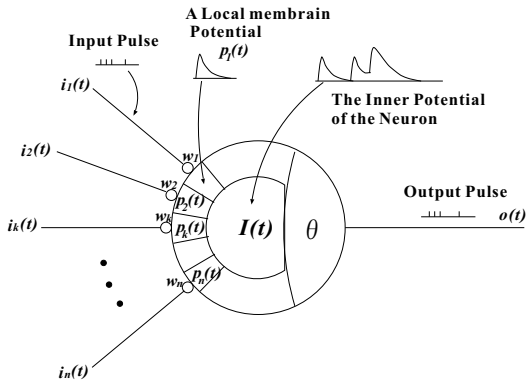

Fig. 2. Pulsed neuron model

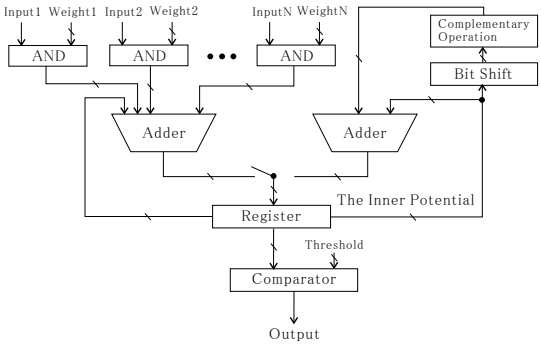

Fig. 3. FPGA implementation of the pulsed neuron model

Figure 2 shows the structure of the PN model. When an input pulse $i_k(t)$ reaches the k^{th} synapse, the local membrane potential $p_k(t)$ is increased by the value of the weight w_k. The local membrane potentials decay exponentially with a time constant τ_k across time. The neuron's output $o(t)$ is given by

$$o(t) = H(I(t) - \theta) \qquad I(t) = \sum_{k=1}^{n} p_k(t) \tag{1}$$

where n is the total number of inputs, $I(t)$ is the inner potential, θ is the threshold and $H(\cdot)$ is the unit step function.

A block diagram of the FPGA implementation of this PN model is shown in Fig.3. In a general neuron model, the analog input values require a multiplexer in order to obtain the input's weighted sum. On the other hand, in the case of pulsed neurons, the input signal is a pulse train, and this sum can be implemented with AND circuits. All AND circuits outputs are then added and the result compared with the threshold.

The time constant decaying is achieved by a bit shift and a complement operation, subtracting a fraction of the former inner potential. Ideally, all local membrane potential units should contain the decaying operator. However, on this research, for the sake of simplicity, only the inner potential unit contains that operator, as the time constant is the same for all local membrane potentials.

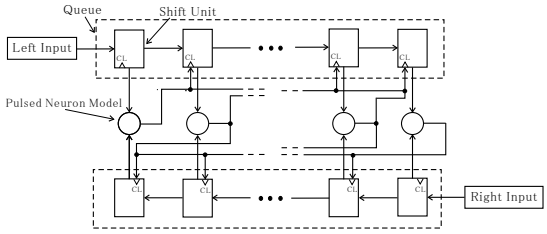

Fig. 4. Circuit diagram of the time difference extractor

The learning mechanism of the pulsed neurons is not implemented on FPGA. Therefore, the pulsed neurons' weights are determined in advance, converted to fixed-point representation and loaded in the FPGA at programming time.

B. Frequency-Pulse Converter

In order to enable pulsed neuron based modules to process the sound data, the analog input signal must be divided on its frequency components and converted to pulses. A bank of Band-Pass Filters (BPF) decomposes the signal, and each frequency channel is independently converted to a pulse train, which rate is proportional to the amplitude of the correspondent signal.

The filter's center frequencies were determined in order to divide the input range (100Hz to 7kHz) in 43 channels equally spaced in a logarithm scale. All the channels are used by the sound recognition and 15 channels (approximately one every three) are used for sound localization.

C. Time Difference Extractor

Each frequency channel's correspondent left and right signals are inputted on an independent time difference extactor. The structure of the extractor is based on Jeffress's model [7], in which the pulsed neurons and the shift operators are organized as shown in Fig.4. The left and right signals are inputted in opposed sides of the extractor, and the pulses are sequentially shifted at each clock cycle. When a neuron receives two simultaneous pulses, it fires. The positon of the firing neuron on the chain determines the time difference.

In the original Jeffress's model, the pulses are shifted until the last pulsed neuron. However, this leads to several false detections due to the matching of pulses of different cycles. This work proposes an improvement to Jeffress's model, which consists on deleting the two input pulses when a neuron fires. On the FPGA implementation, this is achieved by sending each neuron's output to the corresponding shift operator, clearing the input when the neuron fires.

D. Sound Localization Estimator

The sound localization estimator is based on the Competetive Learning Network using Pulsed Neurons

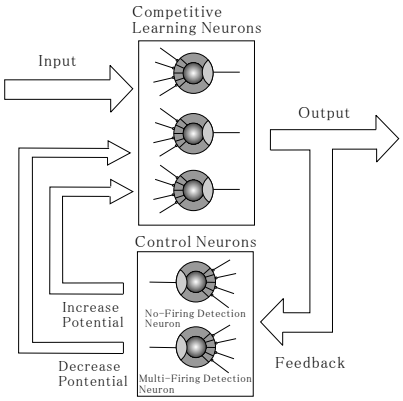

Fig. 5. Competetive Learning Network using Pulsed Neurons(CONP)

(CONP) proposed in [6]. The basic structure of CONP is shown in Fig.5.

In the learning process of CONP, the neuron with the most similar weights to the input (winner neuron) is chosen for learning in order to obtain a topological relation between inputs and outputs. For this, it is necessary to fire only one neuron at a time. However, in the case of two or more neurons firing, it is difficult to decide which one is the winner, as their outputs are only pulses, and not real values. In order to this, CONP has extra external units called control neurons. Based on the output of the Competitive Learning (CL) neurons, the control neurons increase or decrease the inner potential of all CL neurons, keeping the number of firing neurons equal to one. Controlling the inner potential is equivalent to controlling the threshold. Two types of control neurons are used in this work. The No-Firing Detection (NFD) neuron fires when no CL neuron fires, increasing their inner potential. Complementarily, the Multi-Firing Detection (MFD) neuron fires when two or more CL neurons fire at the same time, decreasing their inner potential.

The CL neurons are also controlled by another potential, named the input potential $p_{in}(t)$, and a gate threshold θ_{gate}. The input potential is calculated as the sum of the inputs (with unitary weights), representing the frequency of the input pulse train. When $p_{in}(t) < \theta_{gate}$, the CL neurons are not updated by the control neurons and become unable to fire, as the input train has a too small potential for being responsible for an output firing. Futhermore, the input potential of each CL neuron is decreased along time by a factor β, avoiding rapid changes on the inner potential and improving its adjustment.

Considering the all the described adjustments on the inner potential of CONP neurons, the output equation

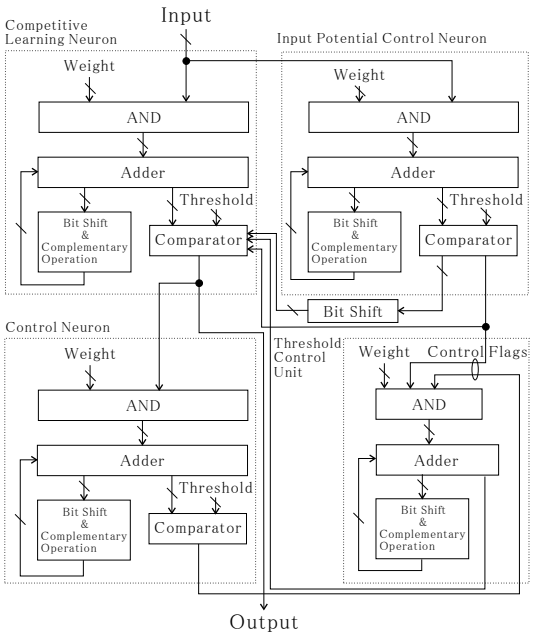

Fig. 6. The Competitive learning network implementation on FPGA

(1) of each CL neurons becomes:

$$o(t) = H\left(\sum_{k=1}^{n} p_k(t) - \theta + p_{nfd}(t) - p_{mfd}(t) - \beta \cdot p_{in}(t)\right) \quad (2)$$

where $p_{nfd}(t)$ and $p_{mfd}(t)$ corresponds respectively to the NFD and MFD neuron, $p_{in}(t)$ is the input potential and $\beta(0 \leq \beta \leq 1)$ is a parameter.

The FPGA implementation of CONP is shown in Fig.6. When calculating the input potential in the FPGA, a single circuit can be used for all CL neurons, as the time decays and weights are constant. Therefore, a single external neuron called Input Potential (IP) control neuron is used for this task. The IP neuron also calculates the threshold $\theta_{in}(t)$ to be decrease from the input potentials by the β rate.

In order to obtain single firing outputs, the control neuron do not act directly on each CL neuron's inner potential. Instead, the output of the control neurons, based on the CL neurons firing state, is sent to the Threshold Control (TC) units. The TC units, one for each control neuron NFD and MFD, calculate how the inner potential of all CL neurons will change. Finally, the TC units send $\theta_{nfd}(t)$ and $\theta_{mfd}(t)$ threshold values to the CL neurons. Equation (2) becomes the equivalent

form:

$$o(t) = H\left(\sum_{k=1}^{n} p_k(t) - \theta - \theta_{nfd}(t) + \theta_{mfd}(t) - \theta_{in}(t)\right) \quad (3)$$

E. Additional Supervised Learning for CONP

The original CONP performs an unsupervised learning, automatically getting the topological relation from data. In order to obtain more accurate angles of sound source, this system performs an additional supervised learning. In oreder to this, two new potentials are introduced, $p_{out}(t)$ and $p_{sup}(t)$, respectively the output potential and supervised pulse potential. When the winner CL neuron fires, if $p_{sup}(t) > p_{out}(t)$, its weights are increased, and if $p_{sup}(t) < p_{out}(t)$, they are decreased. Equation (4) resumes these conditions. After each updating, the weights of each CL neuron are normalized to present unitary absolute value. This prevents some neurons to present very larger weights that could result in wrong firings due to too large potentials.

$$w_k(t+1) = w_k(t) + \alpha \cdot p_{in}(t) \cdot \{p_{sup}(t) - p_{out}(t)\} \quad (4)$$

F. Spectrum Pattern Quantizer

In order to identify the sound source, the proposed system uses the pulsed spectrum pattern of the sound created by the frequency-pulse converter. However, the sound spectrum presents a complex variation along time. In order to reduce its complexity and improve the sound recongition performance, the pulsed spectrum is quantized before being send to the sound recognition estimator. The spectrum pattern quantizer consists of a CONP block, which weights are determined by unsupervised learning. The outputted quantized spectrum can be more easily processed by the sound recognition estimator. Only the left channel is used to recognize the sound source.

G. Sound Recognition Estimator

After the spectrum information is quantized, the sound source information can be retrieved. This is done by another CONP based module, with one CL neuron corresponding to each sound source to be recognized. Another extra neuron is added to indicate that the inputted sound cannot be identified (e.g. a sound source not used on the learning procedure). This block also uses the supervised learning CONP described in section II-E in order to control which neuron corresponds to what kind of sound.

III. Experimental Results

The experimental sounds were recoded in an anechoic room to eliminate the influence of reverberation and noise. A dummy head KU100 was used as the microphone. The sound source was moved circularly from 90° left to 90° right in relation to the dummy head, 1m away from it, as shown in Fig.7.

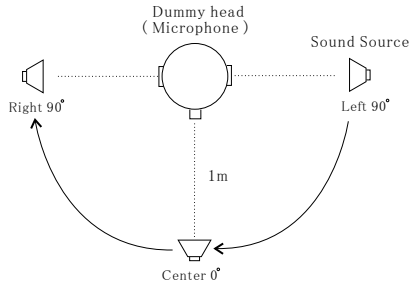

Fig. 7. Sound source movement on experiments

The sound localization system has a resolution of 30°, being able to retrive seven different directions. This resolution can be increased by using a larger number of output neurons. However, for the majority of the applications mentioned before (e.g. aid system for deaf people) a higher resolution is unnecessary.

Six different sound sources were used for the experiments: “alarm bell”, “interphone”, “kettle” (kettle’s neck sound when boiling water), “phone” (telephone ring), “voice” (one vowel) and “white noise”.

At first, all the CONP modules were trained by software using all sound data according to the parameters described in Table I. After that, the obtained weights and parameters were combined with the FPGA circuit code, compiled and loaded into the FPGA. Table II shows the discription of the computer and the FPGA used in this work. Table III shows the number of logic units used by each modules.

A. Localization and recognition

Table IV shows the results for the sound localization of the white noise data and the alarm bell data. The first left column corresponds to the true direction of the sound. The following columns show the obtained firing rate of the neurons corresponding to each of the possible directions. The highest firing rates (indicated by the underlines) always correspond to the correct direction, confirming the efficiency of the proposed FPGA implementation on locating the sound source.

On this experiment, only the white noise sound data was used for learning the sound localization estimator. Nevertheless, other kinds of sound could be correctly located. The reason for this is that each channel’s time difference extractor uses only one frequency band. Different sounds contain different frequency spectrums, but the time difference for each frequency is the same for a fix direction. As the white noise contains all frequency components, a sound localization estimator learned with this data can recognize directions for any other sound data.

The sound recognition experimental results are shown in Table V. Again, the first left vertical column corre-

TABLE I

Parameters of each module used on the experiments

Input Sound	
Sampling frequency	16[kHz]
Quantization bit	8[bit]
Frequency-Pulse Converter	
Number of frequency channels for sound localization	15[Channel]
Number of frequency channels for sound recognition	43[Channel]
Time Difference Extractor	
Total number of shift units	41[units]
Number of output neurons	21[units]
Threshold	1.0
Time constant	0.35[msec]
Competitive learning Neuron	
Number of CL neurons for sound localization estimator	7[units]
Number of CL neurons for spectrum pattern quantizer	10[units]
Number of CL neurons for sound recognition estimator	6[units]
Threshold θ	1.0×10^{-4}
Gating threshold θ_{gate}	150.0
Rate for input pulse frequency β	3.125×10^{-2}
Time constant τ_p	20[msec]
Learning coefficient α	2.0×10^{-7}
Learning iterations	1000
No-Firing Detection Neuron	
Time constant τ	0.5[msec]
Threshold θ	-1.0×10^{-3}
Connection weight to each CL neurons	-0.5
Multi-Firing Detection Neuron	
Time constant τ	1.0[msec]
Threshold θ	2.0
Connection weight to each CL neurons	0.5

TABLE II

Specification of the used instruments

Computer		FPGA	
CPU	Pentium4 3.8GHz	Board	Altera StratixII EPS60
memory	2 GB	Available logic	48,352[ALUTs]

sponds to the true sound sources. The following columns show the rate of firing of the neurons corresponding to each type of sound source over the duration of the input sound data. The underlined values correspond to the maximal values of each sound source.

The FPGA results also presented many firing of neurons corresponding to other sound sources than the true one. Even though the highest frequency firing neuron was the correct one, ideally, other neurons should not fire. One possible reason for this is the use of fixed-point numerical representation on the FPGA PN units' local membrane potentials and inner potentials. This increased the calculation errors when comparing with the software simulations, in which floating-point numerical representation was used. Nevertheless, when considering only the output neuron with the highest firing frequency, the results are comparable and can applied to pratical devices.

TABLE III

Size of each circuit on FPGA

Part	Circuit Size[ALUTs]
I/O circuit	91
Time Difference Extractor	14,025
Sound Localization Estimator	10,011
Frequency Pattern Quantizator	4,594
Sound Recognition Estimator	1,570
Total	30,291

TABLE IV

FPGA results of sound localization

Output Rate[%]	Input Location [white noise / alarm bell] left 90°	left 60°	left 30°	center 0°
left 90°	98.3 / 61.3	0.4 / 19.7	0.3 / 10.6	0.3 / 3.9
left 60°	0.4 / 15.5	98.1 / 47.6	0.3 / 4.0	0.2 / 2.6
left 30°	0.3 / 8.9	0.3 / 7.1	98.4 / 69.8	0.3 / 2.2
center 0°	0.3 / 3.5	0.3 / 2.0	0.3 / 1.7	98.6 / 80.9
right 30°	0.3 / 4.6	0.3 / 14.4	0.3 / 2.6	0.2 / 1.4
right 60°	0.2 / 3.0	0.3 / 5.2	0.2 / 8.0	0.2 / 3.8
right 90°	0.2 / 3.1	0.3 / 4.1	0.2 / 3.3	0.2 / 5.2
Output Rate[%]	Input Location [white Noise / alarm bell] right 30°	right 60°	right 90°	
left 90°	0.2 / 4.4	0.3 / 4.8	0.2 / 2.1	
left 60°	0.2 / 9.9	0.3 / 6.3	0.2 / 3.2	
left 30°	0.2 / 3.0	0.3 / 13.5	0.3 / 5.9	
center 0°	0.3 / 1.6	0.3 / 3.7	0.3 / 2.6	
right 30°	98.5 / 56.9	0.3 / 6.1	0.3 / 8.3	
right 60°	0.3 / 5.4	98.2 / 46.7	0.3 / 10.9	
right 90°	0.3 / 18.9	0.3 / 18.9	98.4 / 66.9	

B. Time Cost and Noise Tolerance

One of the advantages of implementing PN models in hardware is that the neurons can work independently and in parallel. Table VI shows the execution time cost comparision between the FPGA and the software simulation. These results correspond to the white noise data, which has a length of 10 seconds. Not considering the I/O times, the FPGA implementation is approximately 50 times faster than the software simulation.

When developing systems based on sound information,

TABLE V

FPGA results of sound recognition

		Input alarm bell	interphone	kettle
O	alarm bell	99.5	0.0	0.0
u	interphone	0.0	96.9	0.0
t	kettle	0,0	0.0	99.5
p	phone	0.4	0.3	0.5
u	voice	0.0	1.5	0.0
t	white noise	0.1	1.3	0.0
		Input phone	voice	white noise
O	alarm bell	0.0	0.0	0.2
u	interphone	0.0	0.7	0.2
t	kettle	0.0	0.0	0.2
p	phone	99.0	0.5	0.2
u	voice	0.0	98.8	0.2
t	white noise	1.0	0.0	99.0

TABLE VI

Time cost of each unit

Part	Computer[sec]	FPGA[sec]
The length of input signal	10.00	10.0
Time Difference Extracter	41.25	0.84
Sound Localization Estimator	8.61	0.88
Frequencial Pattern Quantizer	2.31	0.73
Sound Recognition Estimator	1.66	0.64
All Processes	53.82	1.10

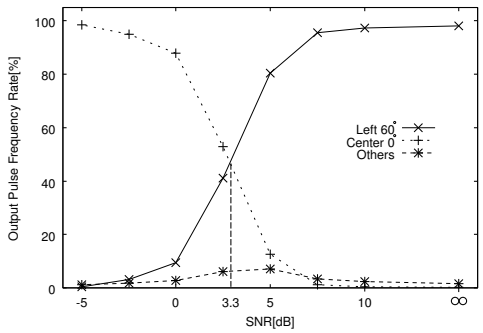

Fig. 8. Result of noise tolerancy in sound localization

an important characeristic is the noise tolerance. Hence, the proposed system's noise tolerance was verified by adding white noise with determined signal/noise rates (SNR) to the sound being located and recognized. The inserted noise has the same time amplitude for both left and right channels. The calculation of SNR of each sound is based on power values.

Figure 8 shows the results for the sound localization modules. The X-axis shows the SNR and the Y-axis corresponds to the output neurons firing rate. The input sound data corresponds to the 60° left direction. Figure 9 shows the results for the sound recognition modules. The alarm bell sound was used as the reference input data.

In both of cases, as the SNR decreases, the time difference and the spectrum pattern tends to the noise signal. The correct direction could be identified with a SNR up to 3.3dB and the correct sound source could be identified with a SNR up to 13.8dB. These results

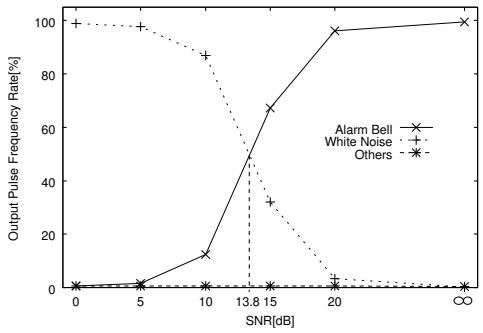

Fig. 9. Result of noise tolerancy in sound recognition

indicate that, if sound localization and recognition results are combined in order to determine an object sound, the performance of the proposed system can be improved.

IV. Conclusions

On this paper, we propose an FPGA implementation of a sound localization and recognition system based on PN model. The system is composed by a time difference extractor and three competitive learning networks for locating and recongizing the sound source. The experimental results show that the proposed system can efficiently identify and locate several sound sources. The use of PN models and the hardware implementation resulted in very small processing times, especially when comparing to software simulations.

For large levels of background noise, this system tends to fail on localizing the sound direction and recognizing the sound source, as no preprocessing is being performed on. Future works include a method for dividing two or more sound source in order to define an objective sound.

Acknowledgment

This research is supported in part by a grant from the Hori Information Science Promotion Foundation, the Grant-in-Aid for Scientific Research and the Knowledge Clusters (Gifu/Ogaki area), both from the Minister of Education, Culture, Sports, Science and Techonology, Government of Japan.

References

[1] Pickles J.O.: "An Introduction to the Physiology of Hearing" , ACADEMIC PRESS, 1988

[2] M.S. Brandstein, J.E. Adcock, and H.F. Silverman : "A closed-form location estimator for use with room environment micorphone arrays", IEEE Trans. Speech Audio Process., vol.5, no.1, pp.45-60, 1997.

[3] K.Kashino, K.Nakadai, T.Kinoshita, H.Tanaka : "Organization of Hierarchical Perceptual Sounds : Music Scene Analysis wit Autonomous Processing Modules and a Quantitative Information Integration Mechanism", In Proceeding of the 1995 IJCAI, 1995.

[4] A.Waibel, T.Hanazawa, G.Hinton, K.Shiano, and K.J.Lang, : "Phoneme recognition using time-delay neural networks", IEEE Trans. Acousti., Speech,& Signal Process., vol.37, no.3, pp.328-339, 1989.

[5] Maass W., and Bishop C.M., : "Pulsed Neural Networks", MIT Press , 1998

[6] Susumu K. , Akira I. : "A Competitive Learning Pulsed Neural Network for Temporal Signals", Proceedings of ICONIP 2002, pp.348-352, 2002.

[7] Jeffress, L.A,"A place theory of sound localization", J.Comp.Physiol.Psychol. , 41 , pp.35-39(1948).