

Boletim Técnico da Escola Politécnica da USP

Departamento de Engenharia Eletrônica

BT/PEE/94-03

**Transformada de Walsh e
Haar Aplicadas no
Processamento de Voz**

**Alexandre Augusto Ottati Nogueira
Thiago Antonio Grandi de Tolosa
Euvaldo F. Cabral Júnior**

São Paulo - 1994

Nogueira, Alexandre Augusto Ottati

Transformada de Walsh e Haar aplicadas no processamento de voz / A.A.O. Nogueira, T.A.G. Tolosa, E. F. Cabral Junior. -- São Paulo : EPUSP, 1994.

22p. -- (Boletim Técnico da Escola Politécnica da USP, Departamento de Engenharia Eletrônica, BT/PEE/94-03)

1. Transformada de Walsh 2. Transformada de Haar
3. Vozes - Processamento I. Tolosa, Thiago Antonio
Grandi de II. Cabral Júnior, Euvaldo F III. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Eletrônica IV. Título V. Série

CDU 517.51

517.51

CDD 621.399

Transformadas de Walsh e Haar Aplicadas no Processamento de Voz

Alexandre Augusto Ottati Nogueira
Thiago Antonio Grandi de Tolosa
Orientador : Euvaldo F. Cabral Júnior
Escola Politécnica da Universidade de São Paulo

29 de Abril de 1994

Resumo do Trabalho

O trabalho realizado utiliza as transformadas de Walsh, Haar e Cosseno para uma aplicação em processamento de voz.

Essa aplicação é dirigida para o reconhecimento do locutor de um sinal de voz e discute-se qual das transformadas mais se adapta ao seguinte processo :

Aquisição do sinal em um arquivo de voz com extensão .WAV :

$$\text{Size} \begin{cases} 70,373 \rightarrow \text{sample-frames} \\ 0.146 \rightarrow \text{minutes} \\ 68.8 \rightarrow \text{kbytes} \end{cases}$$

*Format{ 1 channel 8 bits int
Sampling rate 8000 Khz*

No processo, faz-se um janelamento de 800 amostras com espaço de 400 amostras entre as janelas, aplicando-se em cada uma, simultaneamente, pré-ênfase com $\alpha = 0.95$, o janelamento de Hanning, a Transformada Rápida de Fourier para 256 pontos, a deconvolução do sinal, a Transformada Rápida Cosseno e a Transformada Rápida de Walsh-Hadamard.

Obtém-se os 9 primeiros coeficientes de cada janela das 6400 amostras para os cálculos das distâncias inter-clusters e intra-clusters para cada transformada.

Sumário

| | | |
|---|----------------------------|----|
| 1 | Introdução | 2 |
| 2 | Desenvolvimento | 5 |
| 3 | Parte Experimental | 12 |
| 4 | Conclusão | 20 |
| 5 | Bibliografia e Referências | 21 |

1 - Introdução

1.1.1 - Série de Walsh

As funções de Walsh formam um conjunto ordenado de sinais retangulares, tomando apenas dois valores de amplitude: +1 e -1, definidas em um intervalo de tempo limitado. Dois argumentos são necessários para a definição completa; um período de tempo t e um número de ordem n , relacionado à frequência. A função pode ser expressa como:

$$WAL(n, t) \quad (1.1)$$

Exemplo de um conjunto de funções de Walsh:

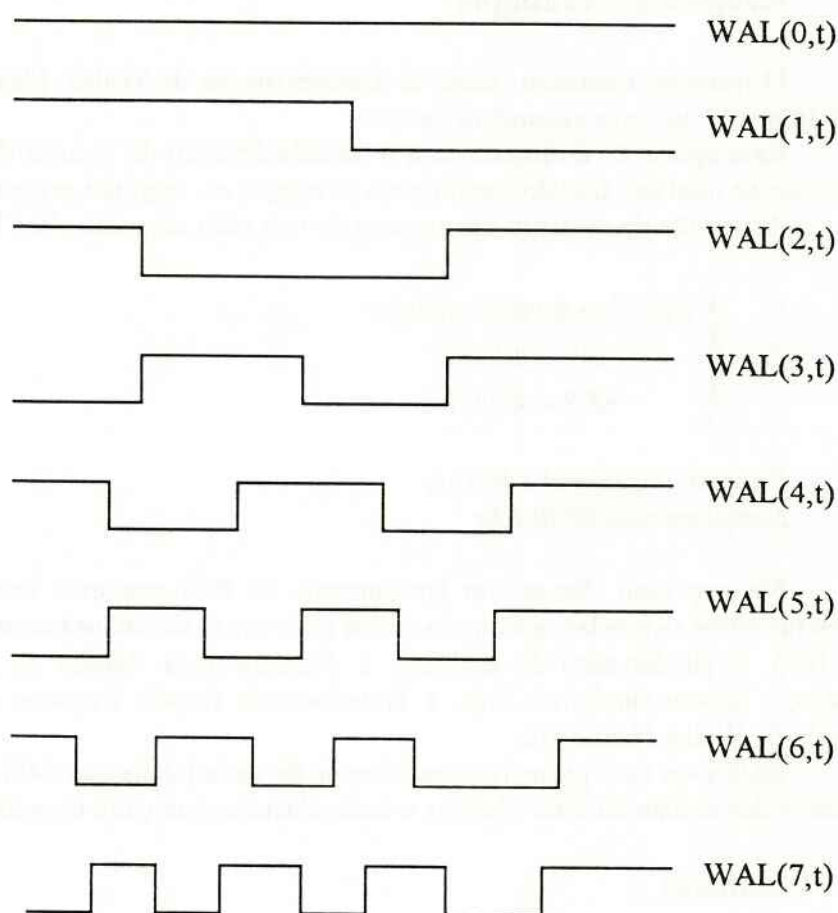


Fig. 1.1 - Conjunto de funções de Walsh.

Uma outra notação utilizada para classificar as funções de Walsh em termos de sinais pares e ímpares pode ser:

$$WAL(2k, t) = CAL(k, t) \quad (1.2)$$

$$WAL(2k-1, t) = SAL(k, t) \quad (1.3)$$

onde $k = 1, 2, \dots, (N/2-1)$.

Isso define duas séries de Walsh, com similaridades em relação as séries cosseno e seno.

O número de ordem n , define a razão de repetição periódica do sinal e é chamado de "*sequency*" para a função.

A série discreta de Walsh correspondente à série de Fourier, na análise de Fourier é:

$$f(t) = a_0 WAL(0, t) + \sum_{i=1}^{N/2-1} \sum_{j=1}^{N/2-1} [a_i SAL(i, t) + b_j CAL(j, t)] \quad (1.4)$$

onde a_i e b_j são coeficientes expressando a série em termos de sequência. A derivada dessa série de coeficientes é classificada como a decomposição de $f(t)$ em suas componentes espectrais. É importante notar que é possível combinar componentes senoidais e cossenoidais em uma variável complexa simples, expressando a mesma frequência na série de Fourier, mas não com a função de Walsh, devido a ausência de um teorema (Shift Theorem) similar.

A ortogonalidade da série de Walsh pode ser mostrada através de duas funções, $WAL(m, t)$ e $WAL(n, t)$:

$$\sum_{t=0}^{N-1} WAL(m, t) WAL(n, t) = \begin{cases} N & \text{para } m=n \\ 0 & \text{para } m \neq n \end{cases} \quad (1.5)$$

Assim as funções de Walsh formam um conjunto ortogonal, que através da divisão das equações acima por N , se transformam num sistema ortonormal.

Com as séries de Walsh, qualquer forma de onda complexa pode ser representada pela superposição de um número de elementos de forma análoga a análise de Fourier, mas poucas vezes, estas funções são utilizadas para a representação de formas de onda descontínuas.

1.1.2 - Série de Haar

A série de Haar, mostrada no exemplo abaixo, também é formada por um conjunto de ondas quadradas periódicas.

Exemplo de um conjunto de funções de Haar:

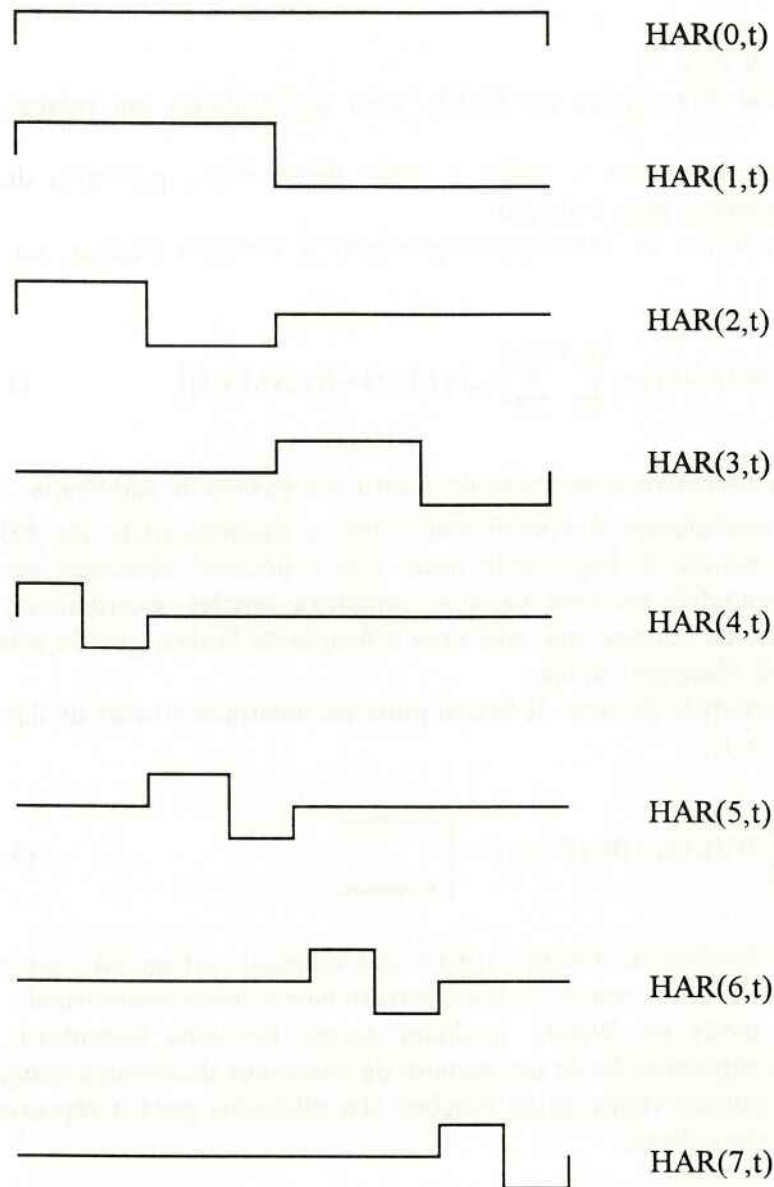


Fig. 1.2 - Conjunto de funções de Haar.

Os valores de amplitude desses sinais, não têm valores uniformes, como nas funções de Walsh, mas assumem um conjunto de valores limitados, $0, \pm 1, \pm\sqrt{2}, \pm 2, \pm 2\sqrt{2}, \pm 4$, etc. Uma forma de expressar essas funções vem a seguir :

$$HAR(n,t) \quad (1.6)$$

Se for considerada a base de tempo $0 < t < 1$, então seguindo uma definição simplificada sugerida por Kremer[1], segue :

$$HAR(0,t) = 1 \quad \text{para } 0 \leq t \leq 1$$

$$HAR(1,t) = 1 \quad \text{para } 0 \leq t < 1/2$$

$$HAR(1,t) = -1 \quad \text{para } 1/2 \leq t < 1/2$$

$$HAR(2,t) = \sqrt{2} \quad \text{para } 0 \leq t < 1/4$$

$$HAR(2,t) = -\sqrt{2} \quad \text{para } 1/4 < t \leq 1/2$$

$$HAR(2,t) = 0 \quad \text{para } 1/2 < t \leq 1$$

$$HAR(3,t) = 0 \quad \text{para } 0 < t \leq 1/2$$

$$HAR(3,t) = \sqrt{2} \quad \text{para } 1/2 \leq t < 3/4$$

$$HAR(3,t) = -\sqrt{2} \quad \text{para } 3/4 < t \leq 1$$

$$\dots$$

$$HAR(2^p + nt) = \sqrt{2^p} \quad \text{para } n/2^p \leq t < (n+1/2)/2^p$$

$$HAR(2^p + nt) = -\sqrt{2^p} \quad \text{para } (n+1/2)/2^p \leq t < (n+1)/2^p$$

$$HAR(2^p + nt) = 0 \quad \text{para qualquer outro valor} \quad (1.7)$$

onde $(p = 1, 2, \dots, \log_2 N)$, $(n = 0, 1, \dots, 2^p - 1)$ e $N = 2^p$. Analisando a *figura 1.2*, nota-se que as duas primeiras funções de Haar são idênticas a $WAL(0,t)$ e $WAL(1,t)$. $HAR(2,t)$ é simplesmente $HAR(1,t)$ deslocada para a metade esquerda da base de tempo com amplitude $+\sqrt{2}$. A próxima função, $HAR(3,t)$ é deslocada para o lado direito da base de tempo.

Como visto acima, as funções de Haar também são ortogonais:

$$\sum_{t=0}^{N-1} HAR(m,t) HAR(n,t) = \begin{cases} N \rightarrow m=n \\ 0 \rightarrow m \neq n \end{cases} \quad (1.8)$$

2 - Desenvolvimento

2.1 - Transformação das séries de Walsh e Haar

A Transformada Discreta de Walsh, pode ser estabelecida para uma série de tempo x_i composta de N amostras da seguinte forma:

$$X_n = 1/N \sum_{i=0}^{N-1} x_i WAL(n,i) \quad n = 0, 1, 2, \dots, N-1 \quad (2.1)$$

e sua inversa:

$$x_i = \sum_{n=0}^{N-1} X_n WAL(n, i) \quad i = 0, 1, 2, \dots, N-1 \quad (2.2)$$

A transformada e sua inversa podem ser obtidas pela multiplicação de matrizes. Se as matrizes forem simétricas, a transformada e sua inversa são idênticas, exceto por um fator $1/N$. Comparando a equação (2.1) com a correspondente DFT, verifica-se que enquanto $WAL(n, i)$ é real e limitado à ± 1 a outra é complexa e assume N diferentes valores. Como consequência disso, a Transformada de Walsh se torna mais fácil e mais rápida de ser calculada.

A Transformada Discreta de Haar (DHT) e sua inversa:

$$X_n = 1/N \sum_{i=0}^{N-1} x_i HAR(n, i/N) \quad (2.3)$$

$$x_i = \sum_{n=0}^{N-1} X_n HAR(n, i/N) \quad (2.4)$$

onde ($i, n = 0, 1, 2, \dots, N-1$). Diferentemente da transformada de Walsh, a matriz formada não é simétrica, portanto, o cálculo da transformada e da sua inversa são realizados separadamente.

Há muita diferença de tempo entre o cálculo da FHT e da FWT. Enquanto a transformada de Haar é mais rápida e seu formato provê melhores resultados em aplicações envolvendo transições abruptas de nível do sinal, tal como, detecção de bordas em processamento de imagens[2], a transformada de Walsh é mais propícia para o trabalho no domínio da frequência.

2.2 - Transformação Bi-Dimensional

Transformações ortogonais discretas em duas dimensões são necessárias para um grande número de aplicações, tal como processamento de imagens[2]. Uma imagem pode ser expressa como séries de sub-imagens ou "pixels" $x_{i,k}$, onde $i = 0, 1, 2, \dots, N-1$ e $k = 0, 1, 2, \dots, M-1$. Uma DFT bi-dimensional avaliada para valores discretos de frequência:

$$w_n = n(2\pi/N) \text{ e } w_m = m(2\pi/M) \quad (2.5)$$

é dada por:

$$X_{n,m} = 1/MN \sum_{i=0}^{N-1} \sum_{k=0}^{M-1} x_{i,k} \exp(-j(w_n i + w_m k)) \quad (2.6)$$

e a transformada inversa por:

$$x_{i,k} = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} X_{n,m} \exp(j(w_n i + w_m k)) \quad (2.7)$$

Embora $x_{i,k}$ seja uma função real e positiva, sua transformada é geralmente complexa. Assim enquanto a transformada inversa possui NM componentes, a transformada possui $2NM$ (parte real e parte imaginária de cada frequência espacial). $X_{n,m}$ possui a propriedade da simetria conjugada:

$$X_{n,m} = -X_{-n,-m} \quad (2.8)$$

Conseqüentemente:

$$X_{n,m} = X_{-n,-m}^* \quad (2.9)$$

Devido a essa propriedade da transformada de Fourier, será necessário considerar apenas as amostras de metade do plano de transformação.

Como no caso uni-dimensional, verifica-se que a representação para ser válida, o campo deve ser periódico. Deve-se considerar ainda, a região da imagem periodicamente repetida nas direções horizontal e vertical e será considerado no processamento sub-seqüente da imagem.

Para o cálculo da DFT bi-dimensional, não é necessário avaliar diretamente a somatória dupla da equação (2.7). Se a matriz de dados $x_{i,k}$ for submetida a duas operações separadas de DFT, sendo uma delas em relação às linhas da matriz de dados e a outra em relação às colunas resultantes, o problema se reduz a um múltiplo do uni-dimensional.

Enquanto a transformada de Fourier bi-dimensional processa muitas propriedades analíticas importantes, há dois problemas, cálculos com números complexos e razão de convergência baixa, que traz desvantagem no tratamento de imagens. As transformadas de Walsh, Haar e Coseno não possuem esse problema.

2.3.1 - Transformada Coseno

A representação de Fourier de uma função real e simétrica possui somente coeficientes reais, correspondente aos termos em coseno da série. Essa condição pode ser implementada em um campo de imagem. Define-se então a Transformada Discreta Coseno (DCT):

$$X_n = 1/N \sum_{i=0}^{N-1} x_i \cos[(n\pi(2i+1))/2N] \quad (2.10)$$

e sua inversa:

$$x_i = 2 \sum_{n=0}^{N-1} X_n \cos[(n\pi(2i+1))/2N] - X_0 \quad (2.11)$$

onde X_0 é o valor médio e $i, n = 0, 1, 2, \dots, N-1$.

O valor dessa transformada em processamento de imagens e compressão de dados deve-se a sua distribuição de variância e baixa taxa de distorção da função. Isso resulta em eficiente compactação de energia, como na transformada ótima de Karhunen - Loève.

2.3.2 - Transformadas de Walsh e Haar

Para valores amostrados e simétricos $x_{i,k}$, as transformadas bi-dimensionais de Walsh e Haar são aplicadas através de uma transformação uni-dimensional sobre as linhas e colunas da matriz. Assim a transformada de Walsh bi-dimensional fica:

$$X_{n,m} = 1/N^2 \sum_{i=0}^{N-1} \sum_{k=0}^{N-1} x_{i,k} WAL(ni) WAL(mk) \quad (2.12)$$

e a sua inversa:

$$x_{i,k} = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} X_{n,m} WAL(n,i) WAL(m,k) \quad (2.13)$$

De forma similar, a transformada de Haar bi-dimensional é definida:

$$X_{n,m} = 1/N^2 \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} x_{i,j} HAR(n,i/N) HAR(m,j/N) \quad (2.14)$$

2.3 - Transformada Rápida de Walsh

Os métodos de implementação da "Fast Walsh Transform" (FWT) incluem sínteses algébricas através de relações recursivas, multiplicação de um conjunto de funções onda-quadradas (Rademacher) através de lógica booleana e através de matrizes de Hadamard. Esse último método será considerado.

A matriz da função discreta de Walsh W_N para $N = 8$ e referindo-se a *figura(1.1)*, é dada por :

$$W_8 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix} = \begin{bmatrix} WAL(0,t) \\ WAL(1,t) \\ WAL(2,t) \\ WAL(3,t) \\ WAL(4,t) \\ WAL(5,t) \\ WAL(6,t) \\ WAL(7,t) \end{bmatrix} \quad (2.15)$$

Agora considerando-se a matriz de Hadamard H_8 como :

$$H_8 = H_4 \cdot H_2 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix} = \begin{bmatrix} HAD(0,t) \\ HAD(1,t) \\ HAD(2,t) \\ HAD(3,t) \\ HAD(4,t) \\ HAD(5,t) \\ HAD(6,t) \\ HAD(7,t) \end{bmatrix} \quad (2.16)$$

pode-se obter um conjunto de N funções discretas de Walsh. Representando as seqüências de N valores das linhas dadas em (2.16) como uma série de Hadamard, $HAD(n,t)$, $n = 0,1,2,\dots,7$. Vem :

$$HAD(n,t) = WAL(b(u),t) \quad (2.17)$$

onde u representa um "bit-reversal" para n e $b(u)$ é uma conversão "Gray-code-to-binary"[3].

Essa derivação traz um fato importante; que a série de Walsh pode ser representada em diferentes ordenações de seqüências, sendo todas ortogonais e válidas para análise. Há três convenções de ordem de uso comum :

"Sequency order" - como mostrado na *figura(1.1)* como $WAL(n,t)$;

"Natural order" - que resulta do método da matriz de Hadamard como $HAD(n,t)$;

"Dyadic order" - é a ordem obtida quando a série de funções for obtida por produtos de funções de Rademacher.

Transformações referindo-se às três convenções para os coeficientes transformados são utilizadas. A mais econômica em termos de simplicidade de programação e rapidez de cálculo é a baseada na matriz de Hadamard na "natural order" dada por (2.16), chamada de *Fast Hadamard Transform (FHT)*.

A transformada discreta de Walsh correspondente à equação (2.1) pode ser expressa em termos de matriz :

$$X_n = H_n \cdot x_i \quad (2.18)$$

onde x_i é um vetor coluna de ordem N representando os valores amostrados de um sinal de entrada. Esse procedimento levará $N(N - 1)$ adições e subtrações.

2.3.1 - "Fast transformation"

O algoritmo de transformação rápida será obtido pelo processo de fatorização de matrizes. Assim para a matriz de Hadamard para H_8 resulta em :

$$H_8 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}^3 \quad (2.19)$$

Com a presença de uma matriz com muitos valores nulos, há uma considerável redução no número de cálculos da Fast Walsh Transform (FWT) de $N(N - 1)$ para $N \log_2 N$ adições e subtrações.

Um diagrama de fluxo para essa derivação é mostrado em[3]. Isso leva a uma saída na "natural-order" e é conhecido como estrutura de *geometria constante*.

A estrutura recursiva desse algoritmo pode ser expressa como segue. Referindo-se ao diagrama de fluxo da estrutura de geometria constante, onde $N = 2^p$ e designando os valores de coeficientes à cada nó como $y_i(n)$, onde $i = 0, 1, 2, 3$ e com n como a posição de cada coluna de nós $n = 1, 2, \dots, 8$ então os valores de entrada, intermediários e transformados são respectivamente :

$$y_0(n) = x(n)$$

$$y_{i+1}(n) = y_i(2n-1) + y_i(2n) \quad \text{para } 0 \leq n \leq N/2$$

$$= y_i(2n - N - 1) - y_i(2n - N) \quad \text{para } N/2 < n \leq N \quad (2.20)$$

Esses resultados podem facilmente ser estendidos a maiores dimensões. Os algoritmos de geometria constante não permitem cálculos "in-place" para os coeficientes intermediários. Como na Fast Fourier Transform (FFT) de "radix-2", a FWT é bastante similar com os algoritmos C-T e S-T[3]. Portanto é possível programar a FWT com algumas modificações de algum algoritmo C-T, com redução dos valores trigonométricos utilizados, à unidade e removendo-se a parte complexa da operação. Um exemplo, é mostrado em[3] de um diagrama de fluxo, utilizando a transformada rápida de Walsh-Hadamard para $N = 8$.

Uma dificuldade com a Walsh-Hadamard Transform (WHT) é obter uma "sequency-order" saída, a qual é desejável em muitas aplicações. Alguns métodos se preocupam com isso, como o sugerido por Manz[7]. Um número de algoritmos alternativos para FWT vem se desenvolvendo, sendo que alguns se preocupam com dados de entrada ordenados normalmente com saída em "sequency-order", outros com saída de forma particular.

Dois exemplos são a transformada de Rademacher-Walsh usada em "logic design"[8] e o CAL-SAL usado em processamento de sinais[9].

2.4 - Transformada rápida de Haar

Como mostrado na *figura(1.2)*, com conjunto de funções de Haar para $N = 8$, obtém-se a matriz de Haar Ha_8 como :

$$Ha_8 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} \\ 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 \end{bmatrix} \quad (2.21)$$

Em forma de equação matricial :

$$X_n = Ha_N \cdot x_i \quad (2.22)$$

Como outras matrizes unitárias a matriz acima pode ser fatorada para obter-se o algoritmo de Fast Haar Transform (FHT). Como a matriz não é simétrica, um algoritmo separado é necessário para a transformada direta e inversa.

Algoritmos para essas transformadas foram desenvolvidos por Andrews[3] e seus diagramas de fluxo são mostrados em[3] para $N = 8$. Multiplicação dos valores de soma/diferença por 1 ou 2 são indicados no diagrama.

O total de somas/subtrações desse algoritmo é :

$$N + N/2 + N/4 + \dots + 2 = 2(N-1) \quad (2.23)$$

3 - Parte experimental

A implementação do trabalho segue os seguintes passos :

1. Aquisição do sinal de voz :

$$\text{Size} \begin{cases} 70,373 \rightarrow \text{sample - frames} \\ 8.8 \rightarrow \text{sec onds} \\ 68.8 \rightarrow \text{kbytes} \end{cases}$$

Format{ 1 channel 8 bits int

Sampling rate 8000 Khz

2. O programa de aquisição dos dados do arquivo de som de extensão .WAV e conversão para o formato MAT :

```
/*
 * Wave_Cabe.h
 *
 * Header do programa Wave_Cab.c
 *
 */

typedef int WORD;
typedef long int DWORD;

typedef struct {
    char nome [4];
    DWORD tam;
}
RIFF_BLOCK;

typedef struct {
    WORD wFormatTag,
        n_Channels;
    DWORD nSamplesPerSec,
        nAvgBytesPerSec;
    WORD nBlockAlign;
}
WAVE_FMT;
```



```

typedef struct {
    long type; /* type */
    long mrows; /* row dimension */
    long ncols; /* column dimension */
    long imagf; /* flag indicating imag part */
    long namlen; /* name length (including NULL) */
} Fmatrix;

extern void savemat(FILE *, int, char *, int, int, int, double *, double *);

/*
 * Wave_Cab.c
 *
 * Programa para exibir o cabeçalho de arquivos no formato .WAV
 *
 * Autores: Alexandre Augusto Ottati Nogueira,
 *          Thiago Antonio Grandi de Tolosa.
 *
 * Data: 30.05.94
 *
 * Baseado no arquivo extraído da revista PC TECH de 28.03.94
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>

#include "wave_cab.h"

#define NUM_MAX_AMOSTRAS 6400 /* 0.8 segundos */

/* variaveis */
static const char nome_vetor_saida[] = "voz",
                 nome_arquivo [] = "voz.mat";

FILE *arq,
     *vetor_saida;

static RIFF_BLOCK riff_block;
static WAVE_FMT wave_fmt;

static int d;
static long num_dados;
static char buffer[255],
           dados [NUM_MAX_AMOSTRAS];
static double voz [NUM_MAX_AMOSTRAS];

void main(int argc, char *argv[])
{

```

```

/* ler parametro de entrada - nome do arquivo .WAV */
if (argc != 2)
{
    printf("Wave_Cab <nome_arquivo_wav> !\n\n");
    exit(1);
}

/* abrir arquivo */
if ((arq = fopen(argv[1], "rb")) == NULL)
{
    printf("Arquivo de som nao encontrado !\n\n");
    exit(1);
}

/* ler cabecalho do arquivo (denominacao "RIFF" + tamanho) */
fread(buffer, 8, 1, arq);
memcpy(&riff_block, buffer, 8);
if (strcmp(riff_block.nome, "RIFF", 4) != 0)
{
    printf("O formato do arquivo nao e RIFF !\n\n");
    exit(1);
}

/* ler cabecalho WAVE */
fread(buffer, 4, 1, arq);
if (strcmp(buffer, "WAVE", 4) != 0)
{
    printf("O tipo do arquivo nao e WAVE !\n\n");
    exit(1);
}

/* ler os blocos de informacao sonora ate o fim do arquivo */
do {
    fread(buffer, 8, 1, arq);
    memcpy(&riff_block, buffer, 8);

    /* se for referente ao formato ("fmt ") do arquivo */
    if (strcmp(riff_block.nome, "fmt ", 4) == 0)
    {
        /* copia a informacao para o buffer e posiciona proximo bloco */
        fread(buffer, riff_block.tam, 1, arq);

        /* copia o formato para a estrutura WAVE_FMT */
        memcpy(&wave_fmt, buffer, sizeof(WAVE_FMT));
    }

    /* se for referente aos dados ("data") do arquivo */
    else if (strcmp(riff_block.nome, "data", 4) == 0)
    {
        /* definir o numero de dados a serem lidos */
        num_dados = (riff_block.tam > (long) NUM_MAX_AMOSTRAS) ?

```

```

        (long) NUM_MAX_AMOSTRAS : riff_block.tam;

fseek(arq, 20000, SEEK_CUR);

/* ler a informacao */
fread(&dados, num_dados, 1, arq);

/* break apos ter lido os dados necessarios */
break;
}

/* se for referente a descricao ("DISP") do arquivo */
else if (strncmp(riff_block.nome, "DISP", 4) == 0)
{
    /* posiciona proximo bloco */
    fseek(arq, riff_block.tam, SEEK_CUR);
}
}
while (! feof(arq));

/* fechar arquivo de entrada */
fclose(arq);

/* transformar os dados em double */
for (d = 0; d < num_dados; d++)
    voz[d] = (double) dados[d];

/* gravar vetor de saida no formato MAT-File */
if (num_dados)
{
    if ((vetor_saida = fopen(nome_arquivo, "wb")) == NULL)
    {
        printf("Erro na abertura do vetor de saida !\n\n");
        exit(1);
    }
    else
    {
        savemat(vetor_saida, 0, nome_vetor_saida, 1, num_dados, 0, voz,
            (double *) 0);

        /* fechar arquivo do MAT-Lab */
        fclose(vetor_saida);
    }
}
}

/*
* savemat - Rotina em linguagem C para salvar uma matriz texto em um MAT-file.
*
* Autores : Alexandre Augusto Ottati Nogueira

```

```

*           Thiago Antonio Grandi de Tolosa
*
*   FILE *fp;
*   double xyz[1000], ar[1000], ai[1000];
*   fp = fopen("foo.mat", "wb");
*   savemat(fp, 2000, "xyz", 2, 3, 0, xyz, (double *)0);
*   savemat(fp, 2000, "a", 5, 5, 1, ar, ai);
*   fclose(fp);
*
*   Baseado em : J.N. Little                               Data : 11-3-86
*/

#include <stdio.h>
#include <string.h>

#include "wave_cab.h"

/*****
fp   : File pointer
type : Type flag: Normally 0 for PC,
        1000 for Sun, Mac, and Apollo,
        2000 for VAX D-float,
        3000 for VAX G-float
        Add 1 for text variables.
        See LOAD in reference section of guide for more info.
pname : pointer to matrix name
mrows : row dimension
ncols : column dimension
imagf : imaginary flag
preal : pointer to real data
pimag : pointer to imag data

*****/

void savemat(FILE *fp, int type, char *pname, int mrows, int ncols, int imagf,
             double *preal, double *pimag)
{
    Fmatrix x;
    int mn;

    x.type = type;
    x.mrows = mrows;
    x.ncols = ncols;
    x.imagf = imagf;
    x.namlen = strlen(pname) + 1;
    mn = x.mrows * x.ncols;

    fwrite(&x, sizeof(Fmatrix), 1, fp);
    fwrite(pname, sizeof(char), (int)x.namlen, fp);
    fwrite(preal, sizeof(double), mn, fp);
    if (imagf)

```

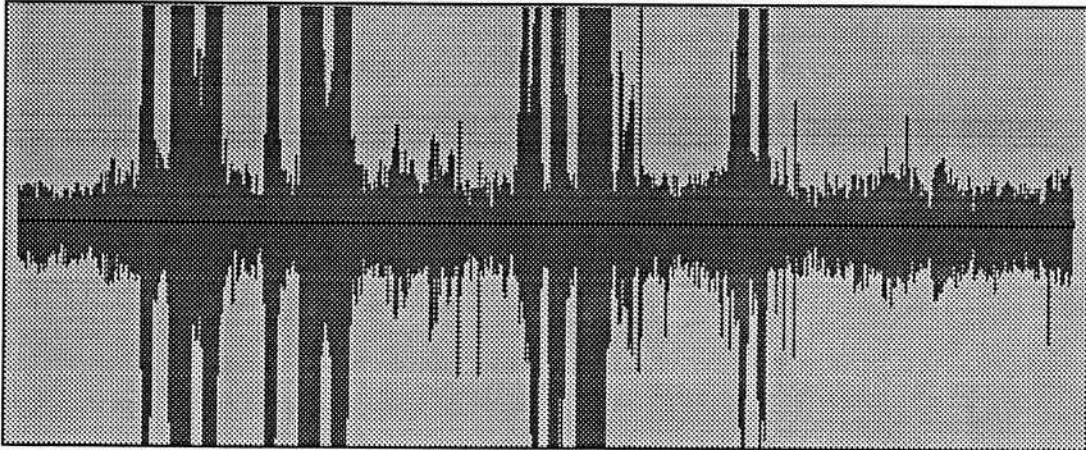
```

    {
    fwrite(pimag, sizeof(double), mn, fp);
    }
}

```

3. Sinal de voz utilizado no processamento :

"Alô, Alô, testando!"



4. O procedimento FCC.m desenvolvido para aplicação no Matlab :

```

% Trabalho Especial - FCC (Frequency Cepstral Coefficients)
%
% Disciplina: PEE-810 Processamento Digital de Sinais I
%
% Professor: Euvaldo F. Cabral Junior
%
% Componentes: Alexandre Augusto Ottati Nogueira
%              Thiago Antonio Grandi de Tolosa
%
% Data:                02/06/94

% carregar o vetor com a informacao sonora
load voz.mat

% janela para 100 milisegundos do sinal amostrado (fa = 8000)
extensao_janela = 800;

% deslocamento da janela para cada 50 milisegundo do sinal amostrado
delta_janela = 400;

% determinar vetor de componentes da janela de hanning
j_hanning = 0 .* (1 : extensao_janela);

```



```

for j = 1 : extensao_janela,
    j_hanning(j) = 0.5 * (1-cos(2*pi*(j)/(extensao_janela+1)));
end;

% carregar matriz de hadamard (matriz_h = hadamard(8), ou seja, para 256 pontos)
load matriz_h.mat

% ultima janela de voz
ultima_amostra = length(voz) - extensao_janela + 1;

% inicializar vetores
dct = 0 .* (1 : 9);
wht = 0 .* (1 : 9);
n_ite = 0;

% para cada janela do sinal, deslocando a janela em 'delta_janela' posicoes
for amostra = 1 : delta_janela : ultima_amostra,

    disp(amostra);

    % monta a janela de extensao 'extensao_janela'
    largura = amostra + extensao_janela - 1;
    x = voz(amostra : largura);

    % aplicar pre-ênfase na janela para alfa = 0.95
    x_aux = x;
    for j = 2 : extensao_janela,
        x_aux(j) = x(j) - (0.95 * x(j - 1));
    end;
    x = x_aux;

    % aplicar janela de Hanning na janela corrente
    x_aux = x .* j_hanning;
    x = x_aux;

    % aplicar Fast Fourier Transform (FFT) na janela para 256 pontos
    f = fft(x, 256);

    % aplicar logaritmo no vetor transformado e filtrado (256 pontos)
    % --- deconvolucao no tempo ----
    f = log(f);

    % aplicar Discrete Cossine Transform (DCT) para os 9 primeiros coeficientes
    dct_c = 0 .* (1 : 9);
    for j = 1 : 9,
        soma = 0;
        soma_p = 0;
        for i = 1 : 256,
            soma_p = (f(i) * cos((j*pi*(2*i+1))/512));
            soma = soma + soma_p;
        end;

```



```

dct_c(j) = soma / 256;

% acumula o vetor resultado para dct
dct(j) = dct(j) + dct_c(j);
end;

% aplicar Walsh-Hadamard Transform (WHT) (256 pontos)
wht_c = 0 .* (1 : 9);
for j = 1 : 9,
    soma = 0;
    soma_p = 0;
    for i = 1 : 256,
        soma_p = (f(i) * matriz_h(j,i));
        soma = soma + soma_p;
    end;
    wht_c(j) = soma / 256;

% acumula o vetor resultado para wht
wht(j) = wht(j) + wht_c(j);
end;

% contabiliza o numero de iteracoes
n_ite = n_ite + 1;

end;

% calcular o valor medio normalizado dos 9 primeiros coeficientes
dct = dct / n_ite;
wht = wht / n_ite;

end.

```

- Janelamento de 800 amostras com espaço de 400 amostras entre as janelas;
- Pré-ênfase com $\alpha = 0.95$;
- Janela de Hanning;
- FFT;
- Aplicação do LOG (deconvolução do sinal);
- Implementação e comparação entre as transformadas DCT e a WHT;
- Obtenção dos 9 primeiros coeficientes de cada janela das 6400 amostras.

4 - Conclusão

Com os resultados obtidos, conclui-se que a Transformada de Walsh-Hadamard, mais se adapta ao trabalho proposto.

Isto deve-se ao fato de que a média da parte normalizada para os 9 coeficientes ser praticamente nula, propiciando uma melhor observação de "features" na transformação.

O processo também foi muito importante para a determinação dos valores dos coeficientes constituintes do sinal :

| | C_1 | C_2 | C_3 | C_4 | C_5 | C_6 | C_7 | C_8 | C_9 |
|-----|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| DCT | 0.0618 | 0.2157 | 0.0474 | 0.0654 | 0.0781 | 0.0268 | 0.0670 | 0.0382 | 0.0459 |
| WHT | 5.7788 | 0.0247 | 0.0185 | 0.0305 | 0.0707 | 0.0208 | 0.0193 | 0.0204 | 0.0200 |

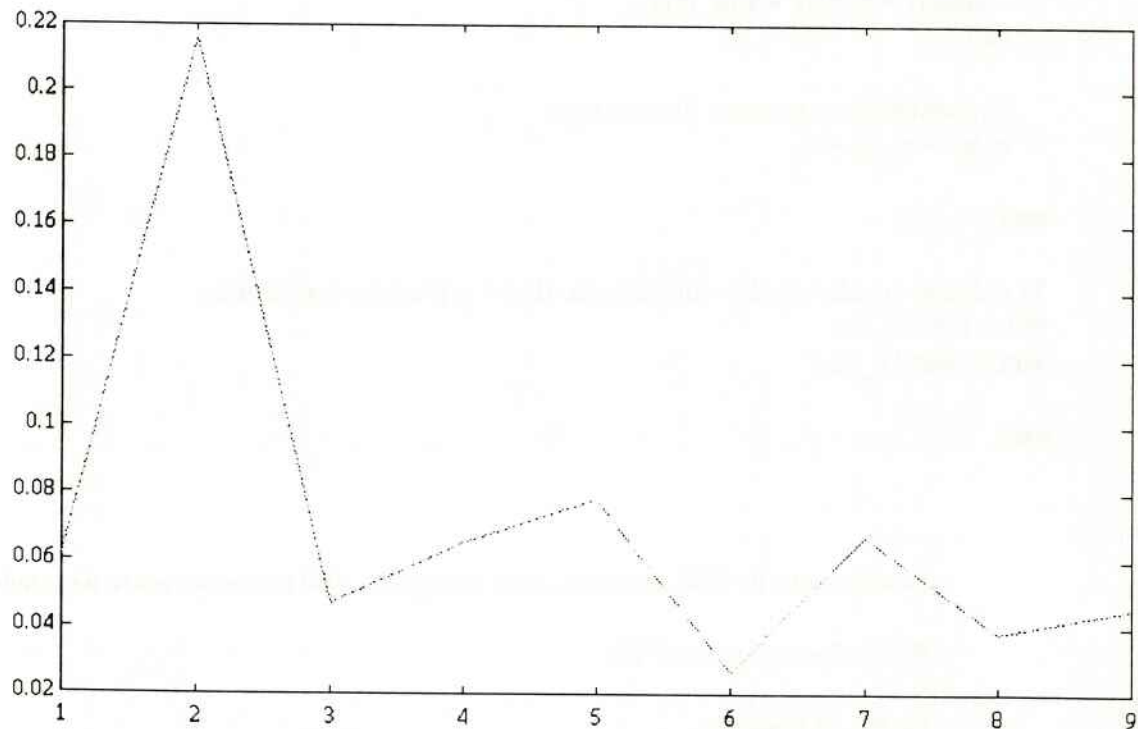


Fig.4.1 - Gráfico Coeficiente X Média, que mostra a média da parte normalizada dos 9 primeiros coeficientes das 6400 amostras utilizando a Transformada Discreta Cosseno no processamento.

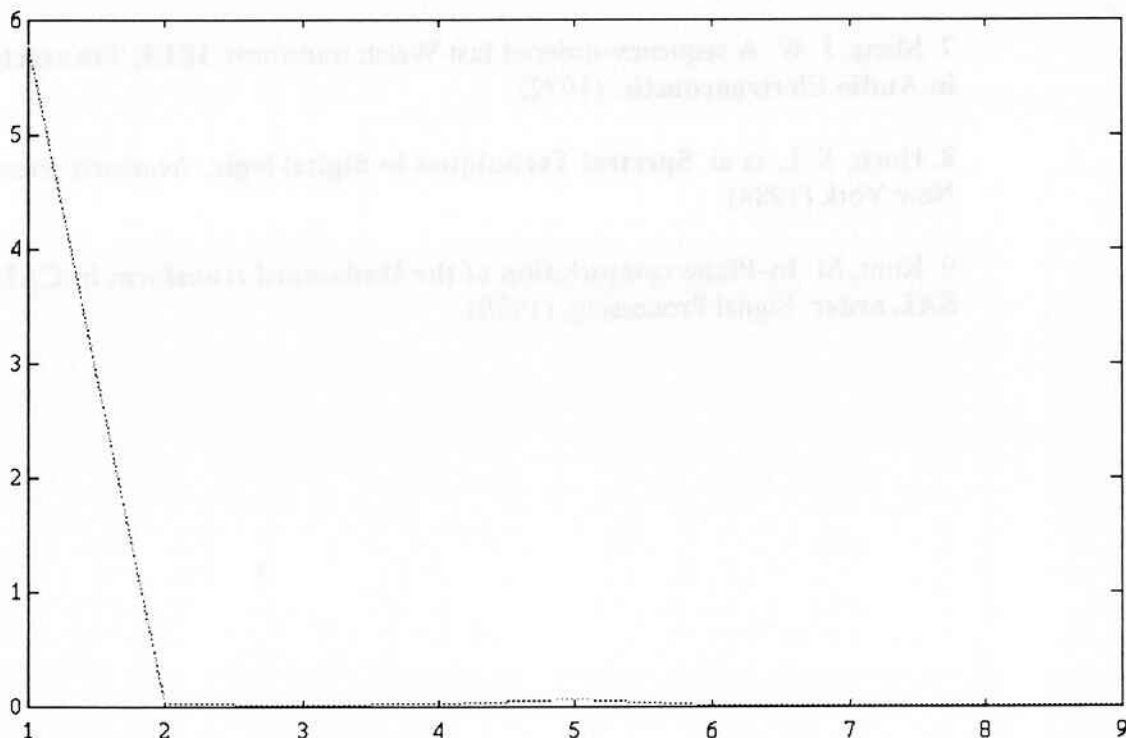


Fig.4.2 - Gráfico Coeficiente X Valor, que mostra a média da parte normalizada dos 9 primeiros coeficientes das 6400 amostras utilizando a Transformada de Walsh-Hadamard no processamento.

5 - Bibliografia/Referências

1. Kremer, H. Algorithms for the Haar function and the fast Haar transform. **Proceedings Theory and applications of Walsh functions**. Hatfield Polytechnic, Hatfield, UK (1971).
2. Andrews, H. C. **Computer techniques in image processing**. Academic Press, New York (1970).
3. Beauchamp, K. G. Transforms for engineers. **A guide to signal processing**. Oxford Science Publications (1987), p.12 - 53.
4. Hlawatsch, F. and Boudreaux G. F. Linear and Quadratic Time-Frequency Signal Representations, **IEEE SP Magazine** (1992).
5. Ahmed N., Natajara T. and Rao K. R. Discrete Cosine Transform, **IEEE Transactions on Computers** (1974).

6. Candy, J. V. **Signal Processing The Modern Approach**, McGraw-Hill (1988), p. 117 - 118.
7. Manz, J. W. A sequency-ordered fast Walsh transform. **IEEE Transactions in Audio Electroacoustic**. (1972).
8. Hurst, S. L. et al. **Spectral Techniques in digital logic**. Academic Press, New York (1984).
9. Kunt, M. **In-Plane computation of the Hadamard transform in CAL-SAL order**. Signal Processing, (1979).

BOLETINS TÉCNICOS - TEXTOS PUBLICADOS

- BT/PEE/93-01 - Oscilador a HEMT - 10 GHz - FÁTIMA S. CORRERA, EDMAR CAMARGO
- BT/PEE/93-02 - Representação Senoidal da Voz através dos Polos do Filtro Preditor - MARCELO B. JOAQUIM, NORMONDS ALENS
- BT/PEE/93-03 - Blindagens por Grades Condutoras: Cálculo do Campo Próximo - LUIZ CEZAR TRINTINALIA, ANTONIO ROBERTO PANICALI
- BT/PEE/93-04 - Sistema de Otimização e Controle de Produção em Minas de Pequeno e Médio Porte - TSEN CHUNG KANG, VITOR MARQUES PINTO LEITE
- BT/PEE/94-01 - Determinação das Frases de Aplicação Forense para o projeto NESPER e Tese de Mestrado IME/94, com Base em Estudos Fonéticos - MARCONI DOS REIS BEZERRA, EUVALDO F. CABRAL JUNIOR
- BT/PEE/94-02 - Implementação e Teste de uma Rede Neural Artificial do Tipo KSON (Kohonen Self-Organizing Network) com Entradas Bidimensionais - MARCELO YASSUNORI MATUDA, EUVALDO F. CABRAL JR.

