UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE MATEMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM MATEMÁTICA APLICADA

Um estudo sobre a Transformada Rápida de Fourier e seu uso em processamento de imagens

por

Louis Augusto Gonçalves

Dissertação submetida como requisito parcial
para a obtenção do grau de

Mestre em Matemática Aplicada

Prof. Dr. Rudnei Dias da Cunha
Orientador

Porto Alegre, junho de 2004.

CIP - CATALOGAÇÃO NA PUBLICAÇÃO

Gonçalves, Louis Augusto

Um estudo sobre a Transformada Rápida de Fourier e seu uso em processamento de imagens / Louis Augusto Gonçalves.—Porto Alegre: PPGMAp da UFRGS, 2004.

61 p.: il.

Dissertação (mestrado) —Universidade Federal do Rio Grande do Sul, Programa de Pós-Graduação em Matemática Aplicada, Porto Alegre, 2004.
Orientador: Cunha, Rudnei Dias da

Dissertação: Matemática Aplicada
FFT, Computação Gráfica.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS

FFT	Fast Fourier Transform
DCT	Discrete Fourier Transform
BMP	Arquivo de imagem Windows Bitmap
DC	Direct Current
PSF	Point Spread Function

RESUMO

Este trabalho é uma síntese da transformada de Fourier na forma discreta e uma de suas principais aplicações à computação gráfica, a restauração de imagens corrompidas por movimento, seja do aparelho óptico ou da própria imagem.

ABSTRACT

This work ...

1 INTRODUÇÃO

A transformada rápida de Fourier foi um dos grandes passos que a humanidade conquistou. Uma vez que descobre-se um novo algoritmo que economiza custo computacional e há vários pesquisadores com implementações cujo tempo de processamento é inviável, ocorre uma corrida para tentar usar de alguma forma este algoritmo na implementação de seus projetos. Este é o avanço que a Transformada Rápida de Fourier possibilita, diminuir o tempo de processamento em aplicações que variam desde resolução de equações diferenciais, equações integrais, problemas inversos até incluir problemas de teoria dos números.

A motivação inicial para a criação da transformada rápida de Fourier foi de criar um algoritmo que permitisse acelerar o processamento da interpolação de uma grande quantidade de dados por polinômios trigonométricos, que é um método usado em Óptica, Mecânica Quântica e inúmeros problemas de simulação. A interpolação de $2m$ pontos de dados pelo cálculo direto requer aproximadamente $4m^2$ multiplicações e o mesmo número de adições. Em áreas que requerem interpolação trigonométrica costumam-se ser necessários cálculos com milhares de pontos, o que torna necessário uma avaliação de milhões de operações onde o erro de arredondamento associado costuma dominar a aproximação. Em 1965, J. W. Cooley e J. W. Tukey publicaram na revista *Mathematics of Computation* um método que requeria em torno de $m.log_2 m$ multiplicações e adições, desde que fosse escolhido uma quantidade apropriada de termos, o que possibilita a diminuição de número de cálculos de milhões para milhares, por isso recebendo o nome de "*Algoritmo da Transformada Rápida de Fourier*".

Neste trabalho a implementação do algoritmo FFT (Fast Fourier Transform) foi realizado na versão Radix 2 com decrescimento em freqüência, por ser esta a forma mais adequada para trabalhar com tratamento de sinais. Como a transformada rápida de Fourier possui bom rendimento em sua versão paralela foram realizados testes do algoritmo FFT em cluster de 4 pcs e na estação de trabalho

SUN 8600 utilizando 2, 4, 8 e 16 processadores, utilizando a linguagem Fortran 90, juntamente com a biblioteca de troca de mensagens MPI (Message Passage Information).

No capítulo 2 descrevem-se as particularidades da Transformada de Fourier na sua forma discreta e a comparação com o modo contínuo, listando as propriedades, as características e aplicações simples da Transformada Discreta de Fourier (DFT) e sua evolução para Transformada Rápida de Fourier.

No capítulo 3 discute-se o algoritmo FFT em duas versões seqüenciais implementadas e a respectiva adaptação ao ambiente paralelo MPI, incluindo resultados sobre escalabilidade e eficiência dos algoritmos.

No capítulo 4 aplica-se o Algoritmo FFT em tratamento de imagens-exemplo e para restauração de imagens com efeitos indesejados, como por exemplos distorções causadas por câmera em movimento e imagens corroídas por insetos.

No capítulo 5 estuda-se uma aplicação envolvendo a equação de Poisson, utilizando a parte imaginária da FFT como método de diagonalização da matriz associada à discretização das derivadas parciais envolvidas.

2 TRANSFORMADA DE FOURIER

A Transformada de Fourier é, em essência, uma ferramenta matemática que realiza a transição entre as variáveis tempo e freqüência de sinais. Este capítulo tem como objetivo principal resumir as principais propriedades da Transformada de Fourier e apresentar técnicas computacionais para sua determinação na forma discreta, chamada DFT (do inglês "Discrete Fourier Transform"), e alguns de seus algoritmos rápidos, chamados coletivamente de FFT (do inglês "Fast Fourier Transform").

Existe uma vasta literatura que trata das características da transformada de Fourier. Na forma contínua há inúmeras obras, onde se encontram textos resumidos e baseados em aplicações como em [16] e [14] e outras com um extenso rigor matemático, como em [12]. Na forma discreta as principais obras de referência são [1] e [8], que formam uma base sólida para o entendimento e aplicação da Transformada Discreta de Fourier. A seguir, apresentaremos alguns destes conceitos.

2.1 Condições suficientes

Seja f uma função que possua as características do teorema integral de Fourier, ou seja, $f(x)$ e $f'(x)$ são seccionalmente contínuas num intervalo finito e $\int_{-\infty}^{\infty} |f(x)|\, dx$ converge.

O teorema integral de Fourier estabelece que:

$$f(x) = \int_0^{\infty} A(\alpha)cos(\alpha x) + B(\alpha)sen(\alpha x)dx \tag{2.1}$$

onde:

$$\left.\begin{aligned} A(\alpha) &= \frac{1}{\pi}\int_{-\infty}^{\infty} f(x)\cos\alpha x dx \\ B(\alpha) &= \frac{1}{\pi}\int_{-\infty}^{\infty} f(x)\sin\alpha x dx \end{aligned}\right\} \tag{2.2}$$

A equação (2.1) é válida se não houver pontos de descontinuidade; caso contrário, substitui-se $f(x)$ por $\frac{f(x+0)+f(x-0)}{2}$. Deve-se salientar que estas condições sao suficientes, porém não necessárias.

2.1.1 Formas equivalentes do Teorema Integral de Fourier

Podemos reescrever o teorema nas formas:

$$f(x) = \frac{1}{\pi}\int_{\alpha=0}^{\infty}\int_{u=-\infty}^{\infty} f(u)\cos\alpha(x-u)dud\alpha \tag{2.3}$$

$$f(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(u)e^{i\alpha(x-u)}dud\alpha \tag{2.4}$$

De forma análoga, para $f(x)$ descontínua, substitui-se $f(x)$ por $\frac{f(x+0)+f(x-0)}{2}$.

2.2 Transformada Contínua de Fourier

Da equação (2.4) temos que[1]:

$$F(\alpha) = \int_{-\infty}^{\infty} f(u)e^{-i\alpha u}du \tag{2.5}$$

e, na forma inversa,

$$f(x) = \frac{1}{2\pi} = \int_{-\infty}^{\infty} F(\alpha)e^{i\alpha x}d\alpha. \tag{2.6}$$

A função $F(\alpha)$ em (2.5) é chamada transformada de Fourier de $f(x)$, e costuma-se representar por $F(\alpha) = \mathcal{F}\{f(x)\}$. Analogamente, a função $f(x)$ em (2.6) é a transformada inversa de $F(\alpha)$ e representamo-la por $f(x) = \mathcal{F}^{-1}\{F(\alpha)\}$. Oppenhein, em [9] utiliza, entretanto, a definição do par da transformada contínua de Fourier para uma seqüência $x(n)$:

$$\begin{aligned} X(e^{j\omega}) &= \sum_{n=-\infty}^{\infty} x(n)e^{-i\omega n}; \\ x(n) &= \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{i\omega n})d\omega \end{aligned} \tag{2.7}$$

[1]As constantes 1 e $\frac{1}{2\pi}$ nas equações (2.5) e (2.6) podem ser alteradas para quaisquer outras desde que o produto entre as duas seja igual a $\frac{1}{2\pi}$.

O símbolo ω usado em (2.7) tem o significado físico de *freqüência angular*, dado em *radianos por segundo*, que obedece à simples lei $\omega = 2\pi f$.

É conveniente utilizar (2.7) quando se trabalha com sinais digitais, ou seja, sinais cuja amplitude e cujo tempo são discretos. Em oposição, é comum tratar sinais analógicos, isto é, sinais com continuidade na variável tempo e na amplitude, com (2.5).

A convergência da transformada de Fourier é sujeita a diferentes definições e interpretações. Se $x(n)$ for absolutamente aditiva, em outras palavras tiver a propriedade $\sum_{n=-\infty}^{\infty} |x(n)| < \infty$, então a série é dita absolutamente convergente e converge uniformemente para uma função contínua de ω. Segundo [4], tal condição é, em quase todos os casos, satisfeita na prática, e nesse caso a seqüência possui energia finita, o que matematicamente significa $\sum_{n=-\infty}^{\infty} |x(n)|^2 < \infty$. A recíproca, porém, não é verdadeira, pois a seqüência $x(n) = \frac{\operatorname{sen}(\omega_0 n)}{\pi n}$ possui energia finita e não é absolutamente aditiva.

A transformada de Fourier pode ser interpretada como um limite da série de Fourier. Seja f uma função L^1 em $[-L, L]$, periódica e de período $2L$. A série de Fourier de f é dada por:

$$f(x) \sim \frac{1}{2}a_0 + \sum_{n=1}^{\infty}(a_n \cos(\frac{n\pi x}{L}) + b_n \operatorname{sen}(\frac{n\pi x}{L})) \tag{2.8}$$

Usando a fórmula de Euler, $e^{i\theta} = \cos(\theta) + i\operatorname{sen}(\theta)$ vem,

$$\begin{aligned} \cos\theta &= \frac{e^{i\theta} + e^{-i\theta}}{2} \\ \operatorname{sen}\theta &= \frac{e^{i\theta} - e^{-i\theta}}{2i} \end{aligned} \tag{2.9}$$

o que possibilita escrever:

$$\left(a_n cos\left(\frac{n\pi x}{L}\right) + b_n sin\left(\frac{n\pi x}{L}\right)\right) = \left(\frac{a_n}{2} + \frac{b_n}{2i}\right) e^{in\pi x/L} + \left(\frac{a_n}{2} - \frac{b_n}{2i}\right) e^{-in\pi x/L}. \tag{2.10}$$

Temos portanto o coeficiente c_n de $e^{in\pi x/L}$ como:

$$c_n = \frac{1}{2}(a_n - ib_n) = \frac{1}{2L}\int_{-L}^{L} f(x)\left(cos\frac{n\pi x}{L} - isen\frac{n\pi x}{L}\right) \tag{2.11}$$

Definindo $c_0 = \frac{a_0}{2} = \frac{1}{2L}\int_{L}^{L} f(x)dx$ teremos:

$$c_n = \frac{1}{2L}\int_{-L}^{L} f(x)e^{-i\pi x/L}dx; \quad para \quad n = 0, \pm 1, \pm 2, \ldots \tag{2.12}$$

Concluindo, pode-se escrever a série de Fourier na forma:

$$f(x) \sim \sum_{n=-\infty}^{\infty} c_n e^{in\pi x/L} \tag{2.13}$$

Introduzindo a notação $\xi_n = \dfrac{n\pi}{L}$, para $n = 0, \pm 1, \pm 2 \ldots$ podemos escrever:

$$c_n = \frac{1}{2L}\int_{-L}^{L} f(x)e^{-i\xi_n x}dx, \quad n = 0, \pm 1, \pm 2, \ldots \tag{2.14}$$

Definindo $h = \dfrac{\pi}{L}$ teremos

$$f(x) \sim \frac{1}{2\pi}\sum_{n=-\infty}^{\infty} hc_n e^{in\pi x/L} \tag{2.15}$$

Quando $L \to \infty$ temos $h \to 0$, e o somatório passa a ter o aspecto de uma soma de Riemann para a integral de $c_n e^{i\xi_n x}$. Temos portanto que as expressões:

$$c(\xi) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} f(x)e^{-i\xi x}dx \tag{2.16}$$

$$f(x) \sim \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} c(\xi)e^{i\xi x}dx \tag{2.17}$$

formam a transformada de Fourier e a transformada inversa de Fourier, respectivamente.

Boa parte das aplicações da Transformada de Fourier utilizam funções de variável real; todavia, a transformada de uma função real é, via de regra, complexa, o que nos leva a considerar $F(u) = R(u) + iI(u)$, onde $R(u)$ e $I(u)$ são as

partes real e imaginária de $F(u)$. Podemos expressar a função de valores complexos na forma polar, ou

$$F(u) = |F(u)|e^{i\varphi(u)}, \tag{2.18}$$

onde

$$|F(u)| = [R^2(u) + I^2(u)]^{\frac{1}{2}}, \tag{2.19}$$

e

$$\varphi(u) = tan^{-1}\left[\frac{I(u)}{R(u)}\right]. \tag{2.20}$$

O termo $|F(u)|$ é chamado espectro de Fourier da função $f(x)$ e $\varphi(u)$ é chamado ângulo da fase; $|F(u)|^2 = R^2 + I^2$ é denominado densidade espectral de $f(x)$ e possui aplicações em imagens. A variável u que aparece na transformada é freqüentemente chamada de variável freqüência, cujo nome é originário da exponencial complexa $e^{-i2\pi ux}$, que pela fórmula de Eüler $e^{-i2\pi ux} = cos(2\pi ux) - i.sen(2\pi ux)$ deixa evidente que (2.5) é composta por uma soma infinita de senos e cossenos; e mais ainda, cada valor de u determina a freqüência de seu correspondente par seno-cosseno.

2.3 Transformada Discreta de Fourier (DFT)

Seja uma seqüência x_m que represente N amostras consecutivas de um sinal contíguo $x(t)$. A transformada discreta de Fourier (DFT) de N termos é definida por[2]:

$$\begin{aligned} &\overline{X}_k = \sum_{m=0}^{N-1} x_m W^{mk}, \quad k = 0, 1, ..., N-1 \\ &W = e^{-i\frac{2\pi}{N}}, i = \sqrt{-1} \end{aligned} \tag{2.21}$$

[2]Encontra-se na literatura tanto as letras 'i' como 'j' para designar a unidade imaginária. Livros sobre Engenharia Elétrica costumam utilizar a letra 'j' enquanto livros de Matemática ou Computação a letra 'i'.

Uma propriedade importante da DFT é a unicidade do par x_m e $\overline{X}_k$ utilizando a DFT como operador, com a transformada direta definida em (2.21) e a transformada inversa definida por:

$$y_l = \frac{1}{N}\sum_{k=0}^{N-1}\overline{X}_k W^{-lk}, \quad l = 0, 1, ..., N-1. \tag{2.22}$$

Deve-se observar que W nada mais é do que a enésima raiz primitiva da unidade. O tratamento completo das raízes primitivas é bem conhecido e pode-se encontrar em [17]. Cada valor da potência ($W^j, j \in \mathbb{Z}$) é chamado fator de rotação, pois cada potência divide o ciclo unitário no plano complexo por argumentos de mesma abertura.

A Transformada Discreta de Fourier relaciona a n-upla g_n entre os números complexos $g = [g_0, g_1, ..., g_{N-1}]^T$ e o vetor de valores complexos $\widehat{g}$, de um espaço vetorial com a mesma dimensão N dada por (2.21), que juntamente com (2.22) formam o núcleo da DFT. Ela nada mais é do que o produto interno dos vetores g com o conjunto de M vetores ortonormais descritos por

$$b_v = [1, W_v^N, W_{2v}^N, ..., W_{(N-1)v}^N]^T, \tag{2.23}$$

onde b_v é a base de vetores ortonormais.

Usando a definição de produto interno de vetores com valores complexos, podemos escrever

$$< g, h >= \sum_{n=0}^{N-1} g_n h_n^*, \tag{2.24}$$

e a transformada discreta de Fourier se reduz a $\widehat{g} =< g, b_v >$, o que justifica uma forma matricial para DFT. Isto significa que o coeficiente $\widehat{g}_v$ no espaço de Fourier é obtido pela projeção do vetor g sobre a base b_v. As N bases vetoriais b_v são ortogonais uma a outra, ou seja:

$$< b_v, b_v' >= \delta_{v-v'} = \begin{cases} 1 & \text{se } v = v', \\ 0 & \text{caso contrário.} \end{cases} \tag{2.25}$$

Por conseqüência, o conjunto b_v forma uma base ortonormal para o espaço vetorial e, assim, cada vetor do espaço pode ser expresso como combinação linear dos vetores da base do espaço de Fourier. A DFT calcula as projeções do vetor g em todos os vetores da base diretamente, ou seja, as componentes de g nas direções dos vetores da base.

Sob esta óptica a DFT é um função que transforma coordenadas num espaço vetorial com N dimensões, com a peculiaridade de trabalhar no corpo dos complexos, e não dos reais como em transformações mais familiares como rotação em espaço tridimensional, e de ter um grande número de dimensões.

Ao contrário do caso contínuo, não há sentido em discutir a existência da Transformada Discreta de Fourier, uma vez que (2.21) sempre existe.

Usando a relação de ortogonalidade, temos que

$$\sum_{x=0}^{N-1} e^{2\pi i \frac{rx}{N}} e^{-2\pi i \frac{ux}{N}} = \begin{cases} N & \text{se } r = u' \\ 0 & \text{caso contrário.} \end{cases} \tag{2.26}$$

que é de fácil verificação, uma vez que

$$\sum_{x=0}^{N-1} e^{\frac{2\pi i}{N} x} = \frac{e^{\frac{2\pi i}{N}(N-1)} e^{\frac{2\pi i}{N}} - 1}{e^{\frac{2\pi i}{N}} - 1} = 1. \tag{2.27}$$

Aplicando a equação (2.22) a $F(u)$, vem

$$\begin{aligned} F(u) &= \frac{1}{N} \sum_{x=0}^{N-1} \sum_{r=0}^{N-1} F(r) e^{\frac{2\pi i}{N} rx} e^{\frac{-2\pi i}{N} ux} = \\ &= \frac{1}{N} \sum_{x=0}^{N-1} F(r) \sum_{r=0}^{N-1} e^{\frac{2\pi i}{N} rx} e^{\frac{-2\pi i}{N} ux} = F(u) \end{aligned} \tag{2.28}$$

2.3.1 Forma matricial da DFT

É possível tratar a DFT na forma matricial

$$\overline{X}_k = [\mathbb{F}][x_m] \tag{2.29}$$

onde $[\mathbb{F}]$ é a chamada matriz de Fourier a qual é uma matriz quadrada, de ordem N, definida por:

$$[\mathbb{F}_{(k,l)}] = W^{(k-1)(l-1)}. \tag{2.30}$$

É importante observar que o termo $\overline{X}_0$ é igual à soma de todos os valores de x_m de (2.21) e, por esta razão, $\overline{X}_0$ é chamada de componente constante ou componente DC da transformada de Fourier. O termo DC vem do inglês "direct current", que é um termo utilizado por engenheiros elétricos para se referir a uma fonte de corrente-voltagem constante, em contraste a fontes cuja voltagem varia de forma senoidal.

A matriz $[\mathbb{F}]$ possui inversa, dada por

$$[\mathbb{F}_{i,j}\mathbb{F}^{-1}_{i,j}] = \frac{1}{N}\sum_{k=1}^{N} W^{(i-1)k}W^{-k(j-1)} = \frac{1}{N}\sum_{k=1}^{N} W^{(i-j)k}. \tag{2.31}$$

A título de exemplo, abaixo mostra-se a matriz de Fourier e sua inversa para $N = 4$.

$$\mathbb{F}_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & (-i) & (-i)^2 & (-i)^3 \\ 1 & (-i)^2 & (-i)^4 & (-i)^6 \\ 1 & (-i)^3 & (-i)^6 & (-i)^9 \end{bmatrix} \tag{2.32}$$

$$\mathbb{F}_4^{-1} = \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & i^2 & i^3 \\ 1 & i^2 & i^4 & i^6 \\ 1 & i^3 & i^6 & i^9 \end{bmatrix} \tag{2.33}$$

e, obviamente, $I = \mathbb{F}\mathbb{F}^{-1}$, onde I é a matriz identidade. Este é um resultado particular do teorema abaixo:

Teorema 2.1. *$[\mathbb{F}_{\mathbb{N}}]$ e $[\mathbb{F}_N^{-1}]$ são matrizes inversas.*

Demonstração. Por hipótese, temos: $[\mathbb{F}_{i,j}] = \frac{1}{N}W^{(i-1)(j-1)}$ e $[\mathbb{F}_{i,j}^{-1}] = W^{-(i-1)(j-1)}$. $[\mathbb{F}_{i,j}\mathbb{F}_{i,j}^{-1}] = \frac{1}{N}\sum_{k=1}^{N} W^{(i-1)k}W^{-k(j-1)} = \dfrac{1}{N}\displaystyle\sum_{k=1}^{N} W^{(i-j)k}$.

$$\text{Para } i = j \ [\mathbb{F}_{i,j}\mathbb{F}_{i,j}^{-1}] = \frac{1}{N}\sum_{k=1}^{N} W^0 = 1$$

$$\text{Para } i \neq j \sum_{k=1}^{N} W^{k(i-j)} = \frac{W^{N(i-j)}W^{i-j} - W^{i-j}}{W^{(i-j)} - 1} = \frac{W^{(i-j)}(W^{N(i-j)} - 1)}{W^{(i-j)} - 1}$$

Como W significa a primeira raiz n-ésima da unidade, então $W^N \equiv 1 mod(N) \Longrightarrow W^{N(i-j)} \equiv W^N \equiv 1 mod(N) \Rightarrow W^{N(i-j)} - 1 = 0$.

Como a matriz de Fourier é quadrada, de dimensão N, então $i - j < N$, logo $W^{i-j} \neq 1$. Assim $\dfrac{1}{N}\displaystyle\sum_{k=1}^{N} W^{k(i-j)} = 0$. □

Como cada elemento de $\overline{X}_k$ é obtido pelo produto interno de uma linha de $[\mathbb{F}]$ pelo vetor de entrada $x(n)$, o que necessita que se efetuem N multiplicações, teremos ao todo N^2 operações de multiplicação, o que computacionalmente é uma operação cara. O algoritmo FFT é de suma importância pelo fato de se conseguir diminuir significativamente a complexidade da DFT, e será tratado mais adiante ainda neste capítulo.

2.3.2 Interpretação física da Transformada de Fourier

Um processo físico pode ser descrito em seu domínio temporal, tomando valores de uma determinada quantidade h em função do tempo, neste caso $h(t)$; ou em domínio freqüencial, cujo processo é especificado pela amplitude H (via de regra um número complexo que também informa a fase) como uma função da freqüência, f denotado por $H(f)$. Deve-se entretanto relacionar h e H como duas representações

da mesma função, utilizando como operador a transformada de Fourier. Neste caso as equações (2.5) e (2.6) podem ser escritas como

$$H(f) = \int_{-\infty}^{\infty} h(t)e^{-2\pi ift}dt \tag{2.34}$$

$$h(t) = \int_{-\infty}^{\infty} H(f)e^{2\pi ift}df \tag{2.35}$$

Caso t seja medido em segundos teríamos f, na equação, (2.34) medidos em ciclos por segundo, se h é função da posição x (supostamente em metros), então H seria uma função de ciclos por metro (inverso do comprimento de onda).

2.4 Comparação entre a Transformada Contínua e a Discreta de Fourier

Freqüentemente, sinais discretizados no tempo são provenientes de sinais contínuos obtidos por uma amostragem periódica. Por exemplo, seja a função definida por:

$$f(x) = \begin{cases} 1 & \text{se } |x| < 3, \\ 0 & \text{se } |x| > 3. \end{cases} \tag{2.36}$$

A transformada de Fourier desta função é dada por:

$$F(\alpha) = \int_{-\infty}^{\infty} f(u)e^{-i\alpha u} = \int_{-\alpha}^{\alpha} e^{-i\alpha u} = \frac{e^{i\alpha a} - e^{-i\alpha a}}{i\alpha} = 2\frac{\sin(a\alpha)}{\alpha}, \quad \alpha \neq 0. \tag{2.37}$$

Figura 2.1: Transformada de Fourier da função (2.36)

Poderíamos desta função (2.36) ter discretizado o sinal, obtendo o gráfico mostrado na Figura 2.2

Figura 2.2: Valores discretos da função (2.36)

Figura 2.3: Transformada Discreta da função anterior.

Consideremos, de forma geral, um sinal analógico $x_\alpha(t)$ cuja representação de Fourier é:

$$x_a(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X_a(i\Omega)e^{i\Omega t}d\Omega \tag{2.38}$$

$$X_a(i\Omega) = \int_{-\infty}^{\infty} x_a(t)e^{-i\Omega t}dt \tag{2.39}$$

A seqüência $x(n)$ com valores $x(n) = x_a(nT)$ é chamada originária de $x_a(t)$ por amostragem periódica, e T é dito período da amostra. Para determinar a maneira como $x(n)$ representa o sinal original $x_a(n)$, é oportuno relacionar $X_a(i\Omega)$, a transformada contínua no tempo de Fourier de $x_a(t)$, com $X(e^{i\omega})$, a transformada discreta de Fourier de $x(n)$.

Usando (2.38), podemos escrever

$$x(n) = x_a(nT) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X_a(i\Omega)e^{i\Omega nT}d\Omega \tag{2.40}$$

Utilizando (2.7), ou seja, a transformada em domínio de tempo discreto, temos:

$$x(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{i\omega})e^{i\omega n}d\omega \tag{2.41}$$

Para relacionar as duas equações anteriores, é cômodo expressar (2.40) como a soma de integrais sobre intervalos de período $\dfrac{2\pi}{T}$, assim:

$$x(n) = \frac{1}{2\pi}\sum_{r=-\infty}^{\infty}\int_{\frac{(2r-1)\pi}{T}}^{\frac{(2r+1)\pi}{T}} X_a(i\Omega)e^{i\Omega nT}d\Omega \tag{2.42}$$

Alterando a variável Ω para $\Omega + \dfrac{2\pi r}{T}$, obtemos uma redução nos limites de integração:

$$x(n) = \frac{1}{2\pi}\sum_{r=-\infty}^{\infty}\int_{-\frac{\pi}{T}}^{\frac{\pi}{T}} X_a\left(i\Omega + i\frac{2\pi r}{T}\right)e^{i\Omega nT}e^{i2\pi rn}d\Omega \tag{2.43}$$

Observando que $e^{2\pi n} = 1$, e uma vez que r e n são inteiros, vem

$$x(n) = \frac{1}{2\pi}\sum_{r=-\infty}^{\infty}\int_{-\frac{\pi}{T}}^{\frac{\pi}{T}} X_a\left(i\Omega + i\frac{2\pi r}{T}\right)e^{i\Omega nT}d\Omega \tag{2.44}$$

Se, agora, trocarmos a ordem dos símbolos de integração e de somatório, e fizermos a troca de variáveis $\Omega = \frac{\omega}{T}$, teremos:

$$x(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left[\frac{1}{T} \sum_{r=-\infty}^{\infty} X_a \left(i\frac{\omega}{T} + i\frac{2\pi r}{T} \right) \right] e^{i\omega n} d\omega \tag{2.45}$$

Observando que a equação anterior é idêntica a (2.41), teremos:

$$X(e^{i\omega}) = \frac{1}{T} \sum_{r=-\infty}^{\infty} X_a \left(i\frac{\omega}{T} + i\frac{2\pi r}{T} \right) \tag{2.46}$$

De forma alternativa, poderíamos expressar (2.46) pela variável freqüência Ω.

$$X(e^{i\Omega T}) = \frac{1}{T} \sum_{r=-\infty}^{\infty} X_a \left(i\Omega T + i\frac{2\pi r}{T} \right) \tag{2.47}$$

Utilizando o fato de que a equação acima é invariante para translações no tempo, então (2.46) e (2.47) formam uma relação entre a transformada contínua no tempo e uma transformada de uma seqüência obtida por amostragem. Podemos afirmar, finalmente, que a seqüência $x(n)$ deve ser vista como a representação de N amostras consecutivas $x_a(nT)$ de um sinal contínuo $x(t)$, de forma que a DFT é a aproximação da transformada contínua de Fourier de uma função.

2.5 FFT - Algoritmos rápidos para DFT

2.5.1 FFT na forma seqüencial

Considere uma DFT $\overline{X}_k$ conforme definido em (2.21), de dimensão N, onde N é um número composto, dessa forma teremos $N = N_1 N_2$. Sob este ponto de vista é possível redefinir os índices m e k por:

$$m = N_1 m_2 + m_1, \qquad m_1, k_1 = 0, ..., N_1 - 1. \tag{2.48}$$

$$k = N_2 k_1 + k_2, \qquad m_2, k_2 = 0, ..., N_2 - 1. \tag{2.49}$$

Substituindo (2.48) e (2.49) em (2.21) obtém-se:

$$\overline{X}_{N_2k_1+k_2} = \sum_{m_1=0}^{N_1-1} W^{N_2m_1k_1} W^{m_1k_2} \sum_{m_2=0}^{N_2-1} x_{N_1m_2+m_1} W_2^{m_2k_2}. \tag{2.50}$$

Isto mostra que a DFT de dimensão N pode ser considerada como uma DFT de tamanho N_1N_2, a menos da introdução dos fatores de rotação $W^{m_1k_2}$. O cálculo de $\overline{X}_k$ por (2.50) é efetivado em três passos, inicialmente correspondendo ao cálculo de N_1 DFTs $\left(\overline{Y}_{m_1k_2}\right)$, o que corresponde a N_1 valores distintos de m_1

$$\overline{Y}_{m_1k_2} = \sum_{m_1=0}^{N_1-1} x_{N_1m_2+m_1} W_2^{m_2k_2}, \tag{2.51}$$

$\overline{Y}_{m_1k_2}$ é então multiplicado pelos fatores de rotação $W^{m_1k_2}$ e temos $\overline{X}_k$ calculado por N_2 DFTs de N_1, de N_2 elementos, usando $\overline{Y}_{m_1k_2}W^{m_1k2}$, usando

$$\overline{X}_{N_2k_1+k_2} = \sum_{m_1=0}^{N_1-1} \overline{Y}_{m_1k_2} W^{m_1k_2} W_1^{m_1k_1}, \tag{2.52}$$

É importante salientar que o procedimento de cálculo poderia ter sido organizado na ordem contrária, com a multiplicação dos fatores de rotação realizados antes do cálculo das primeiras DFTs. Neste caso, encontrar-se-ia:

$$\overline{X}_{N_2k_1+k_2} = \sum_{m_2=0}^{N_2-1} W_2^{m_2k_2} \sum_{m_1=0}^{N_1-1} \left(x_{N_1m_2+m_1} W^{m_1k_2}\right) W_1^{m_1k_1} \tag{2.53}$$

O algoritmo abaixo é descrito em [**?**] e [**?**].

Existem duas formas distintas do algoritmo FFT pelo procedimento acima mencionado, ambas todavia com o mesmo custo computacional. Nos dois processos ocorre permutação das posições do vetor de entrada, um fato previsível uma vez que a seqüência de entrada é formada de N_1 polinômios de N_2 termos e a saída por N_2 polinômios de N_1 termos e implica a adição de uma permutação ao final para completar o algoritmo FFT.

O algoritmo FFT possui eficiência por substituir uma DFT longa por várias DFTs menores, o que depende da fatoração do número N. Uma vez que o

número de operações necessárias para o cálculo direto de uma DFT de N pontos é proporcional a N^2, o número de operações diminui rapidamente, de acordo com a partição da DFT original em várias DFTs menores. No caso mais trivial, uma DFT de tamanho N_1N_2 requer $N_1^2N_2^2$ multiplicações. Em oposição, utilizando as equações (2.51) e (2.52) teremos a separação de N_1 DFTs de N_2 termos, N_2 DFTs de N_1 termos mais N_1N_2 multiplicações pelos fatores de rotação. Conclui-se que o total de multiplicações que devem ser levadas a cabo para calcular a DFT de N_1N_2 pontos se reduz a $N_1N_2^2 + N_1^2N_2 + N_1N_2 = N_1N_2(N_1 + N_2 + 1)$, o que obviamente é menor do que $N_1^2N_2^2$. Na prática, o algoritmo FFT é um recurso poderoso quando N é decomponível em muitos termos, preferencialmente uma potência de um inteiro, sendo indicada a potência de 2. Assim é possível que cada DFT de tamanho N_1 e N_2 seja calculada recursivamente pelo algoritmo FFT até que se obtenha um vetor com número primo de elementos sobre o qual será calculada uma DFT simples, daí a preferência por N igual a uma potência de 2. Utilizando-se deste método em cada estágio, obtém-se uma redução de custo, o que o torna favorável quando N não for potência de 2 o completamento do vetor com zeros até a potência de 2 mais próxima, ação feita na implementação do algoritmo para tratamento de imagens presente nesta dissertação.

2.5.2 O algoritmo FFT de base 2

Seja o cálculo de uma DFT $\overline{X}_k$ de uma seqüência $x(m)$ de N pontos, onde $N = 2^t$. Neste caso, a primeira fase da FFT será definida escolhendo-se $N_1 = 2$ e $N_2 = 2^{t-1}$. Isto equivale a dividir o vetor de entrada com N pontos em dois outros vetores, x_{2m} e x_{2m+1} , de $\frac{N}{2}$ elementos cada, correspondendo aos elementos de posição par e ímpar do vetor original. Sob estas condições, $\overline{X}_k$ torna-se:

$$\overline{X}_k = \sum_{m=0}^{N/2-1} x_{2m}W^{2mk} + W^k \sum_{m=0}^{N/2-1} x_{2m+1}W^{2mk}; \tag{2.54}$$

e dado que $W^{\frac{N}{2}=-1}$:

$$\overline{X}_{k+\frac{N}{2}} = \sum_{m=0}^{N/2-1} x_{2m}W^{2mk} - W^k \sum_{m=0}^{N/2-1} x_{2m+1}W^{2mk}; \qquad k = 0, 1, ..., \frac{N}{2} - 1. \tag{2.55}$$

As equações (2.54) e (2.55) formam a operação conhecida como *'butterfly'* de dois pontos pela aparência dada ao diagrama abaixo:

Figura 2.4: Operação butterfly de dois pontos.

Esta versão do algoritmo FFT é chamada decrescimento temporal. O cálculo da DFT de n pontos é substituído por duas DFTs de comprimento $\frac{N}{2}$ mais N adições e $\frac{N}{2}$ multiplicações por W^k. O procedimento é repetido para trocar duas DFTs de comprimento $\frac{N}{2}$ por 4 DFTs de comprimento $\frac{N}{4}$ ao custo de N adições e $\frac{N}{2}$ multiplicações. O FFT completo calcula a DFT de comprimento 2^t em $t = log_2 N$ passos, pois o algoritmo opera sobre o vetor de entrada como se percorrendo uma árvore binária, como mostrado na figura (2.5.2), e cada etapa converte uma DFT de dimensão 2^{t-i} em 2^{i+1} DFTs de dimensão 2^{t-i-1} com o custo de N adições e $\frac{N}{2}$ multiplicações.

Figura 2.5: Abertura da árvore binária do FFT.

Observando que se trata de uma expansão em árvore binária do vetor inicial, usando como lei de separação a paridade das posições, teremos $log_2 N$ passos. Assim, como a cada passo o algoritmo FFT altera as posições dos elementos do vetor de entrada, a última operação deve ser a reorganização dos elementos. As posições do vetor de entrada são invertidos na forma binária, que é facilmente perceptível no diagrama de fluxo da FFT, bastando escrever as posições de entrada e de saída em base 2 para verificar.

Pode-se, portanto, concluir que o número de multiplicações complexas e adições requeridas para o cálculo de uma DFT de N dados de entrada pelo algoritmo FFT de base 2 será dado respectivamente por $\frac{Nlog_2 N}{2}$ e $Nlog_2 N$.

Esta versão foi implementada sequencialmente, e em paralelo, cujos detalhes são discutidos no capítulo 3.

2.5.3 Forma matricial do algoritmo FFT

O algoritmo matricial da FFT é, conforme o seqüencial, dividido em três etapas; a utilização da matriz DFT do passo anterior, a operação butterfly e a reordenação dos termos. Nesse sentido, o algoritmo tem natureza recursiva e inicia com a matriz DFT de menor dimensão, $\mathbb{F}_2$. Este algoritmo, apesar de possuir interesse teórico é desastroso computacionalmente pois utiliza uma enorme quantidade de memória desnecessária e sua natureza recursiva faz com que matrizes grandes sejam reutilizadas a cada passo, o que provoca rápido estouro de memória, mesmo com vetores de entrada razoavelmente pequenos.

A vantagem do algoritmo está na sua paralelização pois não é necessário comunicação entre os processos. A desvantagem é que, mesmo sem período de latência, o tempo gasto na reconstrução da matriz FFT obtida da DFT anterior

gera uma perda exponencial para o tempo de sua construção, visto que as matrizes sempre dobram de tamanho a cada passo.

O algoritmo recursivo é por conseguinte composto pela multiplicação de três matrizes, uma matriz 'butterfly', a matriz DFT do passo anterior e a matriz de reagrupamento dos elementos. O primeiro passo do algoritmo é o feito abaixo:

$$\mathbb{F}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{2.56}$$

Definimos $\mathbb{F}_2^*$ a matriz de Fourier aumentada de segunda ordem por:

$$\mathbb{F}_2^* = \begin{bmatrix} 1 & 1 & & \\ 1 & -1 & & \\ & & 1 & 1 \\ & & 1 & -1 \end{bmatrix} \tag{2.57}$$

$\mathbb{F}_2^*$ é formada por duas cópias de $\mathbb{F}_2$.

A matriz butterfly é expressa por:

$$\mathbb{B}_2 = \begin{bmatrix} 1 & & 1 & \\ & 1 & & -i \\ 1 & & -1 & \\ & 1 & & i \end{bmatrix} \tag{2.58}$$

E, por último, a matriz de permutação será composta por:

$$\mathbb{P}_2 = \begin{bmatrix} 1 & & & \\ & & 1 & \\ & 1 & & \\ & & & 1 \end{bmatrix} \tag{2.59}$$

Obtém-se a matriz $\mathbb{F}_4$ a partir do produto matricial $[\mathbb{F}_4] = [\mathbb{B}_2][\mathbb{F}_2^*][\mathbb{P}_2]$.

$$[\mathbb{F}_4] = \begin{bmatrix} 1 & & 1 & \\ & 1 & & -i \\ 1 & & -1 & \\ & 1 & & i \end{bmatrix} \begin{bmatrix} 1 & 1 & & \\ 1 & -1 & & \\ & & 1 & 1 \\ & & 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & & 1 & \\ & 1 & & \\ & & & 1 \end{bmatrix} \tag{2.60}$$

A matriz $[\mathbb{F}_4^*]$ terá duas cópias de $\mathbb{F}_4$ nos quadrantes ímpares da matriz e assim sucessivamente, os detalhes de implementação seqüencial e paralela serão discutidos no capítulo 3.

3 IMPLEMENTAÇÃO DOS ALGORITMOS FFT

3.1 FFT de base 2

3.1.1 Versão seqüencial

O algoritmo FFT de base 2 é um algoritmo *'in place'*, ou seja, utiliza o próprio vetor de entrada para efetuar os cálculos, sem a necessidade de trabalhar com vetores auxiliares de dados. Esta é uma das grandes vantagens do algoritmo, pois propicia aproveitamento das posições de memória em todo o seu decorrer. A segunda vantagem é que não é necessário a implementação recursiva, já que é conhecido o número de passos do algoritmo, (o que corresponde à altura da árvore).

O diagrama de fluxo do FFT de base 2 é mostrado abaixo.

Figura 3.1: Diagrama de Fluxo FFT de base 2 (array de entrada com 8 dados).

O fator de rotação $W = e^{\frac{2\pi i}{N}}$ é uma constante complexa calculada no início do programa e é utilizada em todos os passos. O algoritmo FFT pode ser descrito de modo sumário na seguinte forma (considerando m o número de passos, r o passo da FFT, as componentes em ordem seqüencial do vetor de entrada $\mathbf{f}$ dadas por $\mathbf{f}_0(k), k = 0, 1, ..., N-1$ e o resultado após p passos do vetor de entrada dado por $\mathbf{f}_r(k), r = 0, ..., m-1$):

Para r = 1,2,...,m faça:

1. Calcule o expoente p segundo o procedimento (*exp-p*).
2. $\mathbf{f}_r(k) = f_{r-1}(k) + w^p f_{r-1}(k + \frac{N}{2^r})$;
3. $\mathbf{f}_r(k + \frac{N}{2^r}) = f_{r-1}(k) - w^p f_{r-1}(k + \frac{N}{2^r})$;

4. $F(n) = f_m(n), n = 0, 1, ..., N-1$

Fim

Procedimento *(exp-p)*:

1. O índice k é escrito na forma binária com m bits $k_{(2)} = (a_1, a_2, ..., a_m)$;
2. $k_{(2)}$ sofrerá uma permutação cíclica de $(m - r)$ posições para a direita e os bits à esquerda são substituídos por zeros.
3. O número obtido será escrito na forma com os dígitos na ordem contrária, e este será o expoente p.

Dado **f** ordenado seqüencialmente, a utilização deste algoritmo gera o vetor F com cada posição invertida na forma binária, restando portanto uma última tarefa a ser executada, como pode ser observado em (3.1.1). Os códigos em Fortran 90 encontram-se no Anexo 1.

3.1.2 Versão paralela

As operações de adição, subtração e multiplicação da versão seqüencial podem ser facilmente adaptados para o caso paralelo usando-se o vetor de entrada **f**, supondo-o com n valores, sendo este potência de 2. O primeiro processo da paralelização é o de dividir entre os processadores e isto é feito, na implementação apresentada, de forma contígua. Supondo um vetor de entrada com n valores e p processadores, o processador $myid = 0$ guarda as primeiras $\frac{n}{p}$ posições de **f**, $myid = 1$ guarda as próximas $\frac{n}{p}$ posições e assim sucessivamente. Desta forma teremos, usando um comunicador com número de processos igual a uma potência de 2, $log_2(p)$ operações 'butterfly' com comunicação entre os processadores, e as demais calculadas internamente. Na figura (3.1.2) destaca-se o fluxo de dados no processador $myid = 1$.

Figura 3.2: Fluxo de informações no processador myid=1

Para evitar problemas de 'deadlock' na etapa paralela do programa utiliza-se, na operação butterfly, uma chamada à função SENDRECV, uma vez que é necessário uma comunicação recíproca entre dois processadores. Apesar de a função SENDRECV possibilitar a utilização de mesma origem e destino, como em [15], preferiu-se o uso de uma rotina seqüencial para executar a parte não paralela do algoritmo. Por fim, cada processador possuirá um bloco da FFT do vetor de entrada armazenado com a posição invertida na forma binária e, se for necessário, uma redução é realizada para a obtenção do vetor FFT completo em um processador.

3.1.2.1 Resultados

Os testes da tabela (3.1.2.1) foram feitos em um cluster de 4 PCs Pentium IV com clock de 1.7GHz. Os percentuais de ganho de tempo estão relacionados abaixo.

Tabela 3.1: Escalabilidade do cluster

cluster	2^{12} dados	2^{15} dados	2^{17} dados	2^{19} dados	2^{21} dados	2^{22} dados
1 processador	150000	1770000	3900000	19040000	90450000	195840000
2 processadores	900000	1010000	2200000	10590000	50350000	108250000
variação percentual	60%	57,1%	56,4%	56,1%	55,66%	55,27%
4 processadores	70000	750000	1670000	7560000	34450000	73370000
variação percentual	46,6%	42,4%	42,8%	39,7%	38,1%	37,5%

A tabela nos mostra que para valores de entrada maiores o aproveitamento aumenta. Para 2 processadores o valor mínimo do tempo de execução seria de 50% do tempo utilizado por 1 processador. Iniciando com 2^{12} dados e terminando com 2^{22} dados passamos de 60% para 55,7% do tempo de execução. Isto se explica pelo motivo de o número de comunicacoes ser de $log_2 p$.

Os testes a seguir foram realizados usando a estação de trabalho *Sun Fire 6800*, composta de 20 processadores SPARC III de 750MHz, localizada no

Laboratório Nacional de Computação Científica (LNCC - MCT), em Petrópolis, RJ.

Observamos pelo gráfico (3.1.2.1) que o crescimento do número de processadores possui um valor ótimo onde a partir deste o período de latência supera o ganho de tempo promovido pelo acréscimo dos processadores. Verifica-se claramente este fato quando usamos se utiliza um vetor de entrada razoavelmente pequeno (2^{10} dados).

Figura 3.3: Escalabilidade do algoritmo FFT de base 2.

Tabela 3.2: Escalabilidade Sun Fire 6800

cluster	2^{12} dados	2^{15} dados	2^{17} dados	2^{19} dados	2^{21} dados	2^{22} dados
1 processador	000000	0	3870000		00000000	000000000
2 processadores	000000	0	6170000	0000000	00000000	000000000
variação percentual	00%	0%	56,4%	0%	00000%	00000%
4 processadores	00000	0	4700000	0000000	00000000	00000000
variação percentual	0000%	0%	42,8%	0%	0000%	0%
8 processadores	00000	000000	2400000	0000000	00000000	00000000
variação percentual	0000%	42,4%	42,8%	0%	0000%	0000%
16 processadores	00000	000000	1350000	5950000	00000000	00000000
variação percentual	0000%	0000%	42,8%	0%	0000%	00000%

4 APLICAÇÃO DO ALGORITMO FFT EM IMAGENS

4.1 A qualidade das imagens

Os computadores não são capazes de trabalhar com imagens contínuas, assim se faz necessário transformar a imagem em uma matriz tridimensional de números que identificam a intensidade média de cor em cada região. Um ponto na matriz é chamado pixel ou pel, abreviações do inglês *picture element*, cuja posição é dada pela notação normal das matrizes, ou seja, o primeiro índice indica a linha e o segundo índice a coluna. Dessa forma uma imagem que contenha $M \times N$ pixels é representada por uma matriz $M \times N$.

Não há um método para tratamento de imagens computacionais. Quando uma imagem é processada o usuário é o último julgador da qualidade final da imagem, portanto a avaliação é subjetiva. Tornar uma imagem aceitável depende de uma escolha adequada de qual algoritmo utilizar. Os melhores resultados são costumeiramente obtidos por uma combinação de experiência e intuição. Umbaugh, em [18] define o trabalho de restauração de imagens digitais como uma arte e não como uma ciência.

A série de fotografias abaixo representam a mesma imagem com diferentes quantidades de pixels. Quantidades maiores causam a impressão de continuidade das formas. Com pixels de tamanho maior a resolução empobrece e detalhes não são reconhecidos. A quantidade de pixels necessária para cada imagem depende da necessidade do usuário, pois na figura (4.1) a imagem pode servir para identificar a criança, para obter o modelo da motocicleta ou num caso extremo determinar a marca do relógio da pessoa que segura o bebê.

4.2 A representação das imagens por sinais

Analisa-se luminosidade como uma grandeza positiva. Um ser humano normal percebe uma faixa média de 2.000.000 a 16.000.000 de cores, e como as cores visíveis pelo olho humano são combinações de vermelho, verde e azul (RGB - red, green, blue) é razoável trabalhar com inteiros sem sinal de 8 bits para cada cor fundamental, ou de forma mais clara, valores que variam de 0 a 255. Assim dispõe-se de um total de 16.777.216 cores possíveis. O formato Windows BMP versão 3, sem compressão, utiliza exatamente esta descrição, possui apenas um cabeçalho com as informações básicas da imagem e a seguir os dados de cor para cada pixel. Mais detalhes sobre este formato são apresentados .

/**********/

Trabalha-se com imagens digitais como um conjunto, onde cada um tem sua posição definida univocamente. Desta forma qualquer imagem pode ser composta de imagens fundamentais onde apenas um pixel possui valor unitário e todos os demais são nulos. Denota-se esta imagem essencial, valor 1 na linha m e coluna n, por:

$$ {}^{m,n}P : \quad {}^{m,n}P_{m',n'} = \begin{cases} 1, & \text{se } m = m' \text{ e } n = n' \\ 0, & \text{em todos os outros casos.} \end{cases} \tag{4.1} $$

As imagens fundamentais ${}^{m,n}P$ formam uma base ortonormal, cujo produto interno é definido em (4.1).

Uma imagem qualquer pode ser composta de imagens fundamentais de (4.1) usando o somatório:

$$ C = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} C_{m,n} \left({}^{m,n}P \right); \tag{4.2} $$

onde $C_{m,n}$ representa cada valor das cores fundamentais (RGB). Definindo o produto vetorial entre duas imagens G e H por:

$$< G, H >= \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} G_{m,n} H_{m,n} \tag{4.3}$$

Da igualdade (4.3) podemos obter a relação de ortogonalidade para as imagens fundamentais (${}^{m,n}P$):

$$C = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \left({}^{m',n'}P_{m,n} \right) \left({}^{m'',n''}P_{m,n} \right) = \delta_{m'm''} \delta_{n'n''}; \tag{4.4}$$

onde δ_{mn} é o símbolo de Kronecker, definido como 0 se $m \neq n$ e 1, se $m = n$.

Concluindo, o produto interno entre duas imagens fundamentais é nulo se forem diferentes e unitário caso contrário, portanto os MN pixels geram um espaço vetorial de dimensão $M \times N$ sobre o conjunto dos reais e conseqüentemente uma imagem $M \times N$ representa um ponto no espaço vetorial $M \times N$.

Se porventura trocarmos o sistema de coordenadas a imagem deve continuar a mesma, porém com coordenadas alteradas. Assim, os dados da imagem não se modificam, mas devem ser avaliados por outro ponto de vista. De forma simples pode-se afirmar que ambas as representações devem ser equivalentes e reproduzir a imagem. Uma transformação de coordenadas apropriada nos possibilita a passagem de um sistema para outro com nenhuma ou pouca perda - ou modificação - de informação.

4.3 Transformada de Fourier bidimensional

Dentre as várias possíveis representações, a transformada de Fourier tem recebido grande importância. Nesta transformada as imagens fundamentais possuem comportamento periódico.

As figuras (4.2) demonstram a mesma imagem composta por pixels individuais e por pixels expostos em períodos.

Para utilização da transformada de fourier em imagens devemos nos valer da sua forma bidimensional, que nada mais é do que uma extensão do caso unidimensional.

4.3.1 Extensão da Transformada de Fourier para duas dimensões (2D-DFT)

O par de Transformada de Fourier para duas variáveis é dado por:

$$\begin{aligned} F(u,v) &= \frac{1}{N_1 N_2} \sum_{u=0}^{N_1-1} \sum_{v=0}^{N_2-1} f(x,y) e^{-2\pi i(\frac{ux}{N_1}+\frac{vy}{N_2})}; \\ u &= 0,1,\ldots,N_1-1; \\ v &= 0,1,\ldots,N_2-1; \end{aligned} \tag{4.5}$$

e a respectiva inversa por:

$$\begin{aligned} f(x,y) &= \sum_{u=0}^{N_1-1} \sum_{v=0}^{N_2-1} F(u,v) e^{2\pi i(\frac{ux}{N_1}+\frac{vy}{N_2})}; \\ x &= 0,1,\ldots,N_1-1; \\ y &= 0,1,\ldots,N_2-1; \end{aligned} \tag{4.6}$$

4.3.2 Transformada Rápida de Fourier para duas dimensões

Considerando a equação (4.5) escrita com fatores de rotação separados, de dimensão $N_1 \times N_2$, temos:

$$\begin{aligned} \overline{X}_{k_1 k_2} &= \sum_{m_1=0}^{N_1-1} \sum_{m_2=0}^{N_2-1} x_{m_1 m_2} W_1^{m_1 k_1} W_2^{m_2 k_2}, \quad para \\ k_1 &= 0,1,...,N_1-1; \\ k_2 &= 0,1,...,N_2-1; \quad onde \\ W_1 &= e^{-i\frac{2\pi}{N_1}}, \\ W_2 &= e^{-i\frac{2\pi}{N_2}}; \quad e \quad i=\sqrt{-1}. \end{aligned} \tag{4.7}$$

A fim de resolver (4.7) podemos separar os índices de somatório, obtendo:

$$\overline{X}_{k_1 k_2} = \sum_{m_1=0}^{N_1-1} W_1^{m_1 k_1} \sum_{m_2=0}^{N_2-1} x_{m_1 m_2} W_2^{m_2 k_2} \tag{4.8}$$

Definindo $\overline{Y}_{m_1k_2} = \sum_{m_2=0}^{N_2-1} x_{m_1m_2} W_2^{m_2k_2}$, teremos $\overline{X}_{k_1k_2} = \sum_{m_1=0}^{N_1-1} W_1^{m_1k_1} \overline{Y}_{m_1k_2}$.
Sob este ponto de vista temos o cálculo de N_2 DFTs de N_1 dados nas N_2 seqüências $\overline{Y}_{m_1k_2}$, correspondendo a N_2 valores distintos de k_2, $\overline{X}_{k_1k_2} = \sum_{m_1=0}^{N_1-1} \overline{Y}_{m_1k_2} W_1^{m_1k_1}$.

Este método é chamado linha-coluna, e computa inicialmente as DFTs das colunas e a seguir das linhas. O algoritmo foi implementado em Fortran 90 sendo utilizado para o tratamento das imagens.

Como colocado por [6], a estrutura algébrica da DFT bidimensional é de produto externo entre os vetores linha e coluna denotado pelo símbolo $\otimes$ que formam a base de vetores da DFT de uma dimensão, em outras palavras o núcleo da 2D-DFT é separável, ou seja:

$$B_{u,v} = \begin{bmatrix} 1 \\ W_M^u \\ W_M^u \\ \vdots \\ W_M^{(M-1)u} \end{bmatrix} \begin{bmatrix} 1, W_N^v, W_N^{2v}, \ldots, W_N^{(N-1)v} \end{bmatrix} = b_u \otimes b_v \tag{4.9}$$

Como no caso direto, a 2D-FFT inversa é uma simples extensão do caso unidimensional, e portanto é dada por:

$$G_{mn} = \frac{1}{N_1 N_2} \sum_{m_1=0}^{N_1-1} \sum_{m_2=0}^{N_2-1} \overline{x}_{m_1m_2} W_1^{m_1k_1} W_2^{m_2k_2} \tag{4.10}$$

A ilustração da transformada de Fourier por imagens requer obviamente que os valores utilizados sejam reais, daí a importância da densidade espectral. Em casos gerais, a densidade espectral de uma função discretizada possui uma diferença entre os valores máximo e mínimo muito maior do que a variação suportada pelos algoritmos de geração de imagem digitais, de forma que a reprodução da densidade na forma pura não é fidedigna com a informação contida no espectro, e mesmo o olho humano é incapaz de perceber flutuações pequenas de intensidade luminosa. A

forma encontrada descrita em ([4]) é a de gerar uma imagem a partir da função

$$D(u,v) = c.log[1 + |F(u,v)|] \tag{4.11}$$

ao invés de $|F(u,v)|$, onde c é uma constante de proporcionalidade utilizada para cada formato de imagem sobre a qual se está trabalhando. No caso utilizado neste trabalho utilizou-se $c = \frac{255}{DC}$, uma vez que as imagens utilizadas são do padrão Windows BMP de 24 bits, conforme descrita em anexo.

4.3.3 Propriedades da Transformada de Fourier de duas dimensões

4.3.3.1 Translação

As propriedades de translação da transformada de Fourier são dadas por:

$$f(x,y)e^{\frac{2\pi i}{N}(u_0x+v_0y)} \rightleftharpoons F(u-u_0, v-v_0) \tag{4.12}$$

e por

$$f(u-u_0, v-v_0) \rightleftharpoons F(u,v)e^{\frac{-2\pi i}{N}(ux_0+vy_0)} \tag{4.13}$$

onde as setas duplas indicam que as transformadas são auto-recíprocas, isto é, existe correspondência entre a função e sua transformada de Fourier na ordem direta e inversa. Em outras palavras multiplicar uma função pela exponencial explicitada e calcular a transformada deste produto significa trasladar o centro do gráfico $u \times v$ para (u_0, v_0) no plano freqüencial. Analogamente multiplicar $F(u,v)$ pelo termo exponencial altera o centro do plano espacial para o ponto (x_0, y_0).

É importante salientar que a densidade espectral não é alterada pela alteração aplicada a $f(x,y)$, ou seja:

$$|F(u,v)e^{\frac{-2\pi i}{N}(ux_0+vy_0)}| = |F(u,v)|$$

este é um fato importante para a visualização da transformada de Fourier, uma vez que a imagem não sofrerá variação se a função for trasladada.

4.3.3.2 Periodicidade

Supondo uma imagem disposta numa matriz quadrada de dimensão N, a transformada discreta de Fourier é periódica e tem período N, logo:

$$F(u,v) = F(u+N,v) = F(u,v+N) = F(u+N,v+N) \tag{4.14}$$

o que é facilmente verificado substituindo na equação (4.5), bastando para isso observar que

$$F(u+N,v) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)e^{-2\pi i(\frac{ux}{N})}e^{-2\pi ix}e^{-2\pi i\frac{vy}{N}}. \tag{4.15}$$

O termo $e^{-2\pi ix}, x = 1, 2, \ldots, N$ é sempre igual a unidade, o que explica a periodicidade em questão.

A equação (4.14) indica que $F(u,v)$ se repete indefinidamente, todavia N valores contíguos são suficientes para recompor completamente a função $f(x,y)$, isto justifica o procedimento de localizar o DC da função no centro da matriz, o que traz benefícios visuais, uma vez que os maiores valores da transformada passam a se encontrar no centro da imagem, e nao espalhados nas fronteiras.

4.3.3.3 Convolução

Define-se a convolução entre duas funções $f(x)$ e $g(x)$, denotada por $f(x) * g(x)$, pela integral:

$$f(x) * g(x) = \int_{-\infty}^{\infty} f(\alpha)g(x-\alpha)d\alpha \tag{4.16}$$

A grande importância da convolução no domínio freqüencial é que $f(x)*g(x)$ e $F(u)G(u)$ constituem um par de transformadas de Fourier. Desta forma, tem-se:

$$\begin{aligned} f(x) * g(x) &\rightleftharpoons F(u)G(u) \\ F(u) * G(u) &\rightleftharpoons f(x)g(x) \end{aligned} \tag{4.17}$$

O par de equações (4.22) é conhecido por *teorema da convolução*. Em outras palavras o teorema indica que a convolução no domínio temporal pode ser obtido pela transformada inversa do produto $F(u)G(u)$ no domínio freqüencial e, de forma semelhante, a convolução no domínio freqüencial se reduz à multiplicação no domínio temporal.

Suponha duas funções $f(x)$ e $g(x)$ discretizadas em dois vetores de comprimento A e B, respectivamente, isto é: $[f(0), f(1), \ldots, f(A-1)]$ e $[g(0), g(1), \ldots, g(B-1)]$. Conforme a seção (4.3.3.2), a transformada discreta de Fourier e sua inversa são funções periódicas. Considerando f e g com período A e B pelo teorema da convolução é de se esperar que a convolução discreta de f e g sejam também periódicas, com algum período M. Segundo mostrado em [2], o valor de M que deve ser usado é $M = A + B - 1$, do contrário ocorre o fenômeno conhecido por *"aliasing"*, que é formado pela perturbação causada pela contribuição de dados de períodos adjascentes.

Completando com zeros as funções obtemos as seqüências extendidas:

$$f_e(x) = \begin{cases} f(x) & \text{se } 0 \leq x \leq A-1, \\ 0 & \text{se } A \leq x \leq M-1. \end{cases} \tag{4.18}$$

$$g_e(x) = \begin{cases} g(x) & \text{se } 0 \leq x \leq B-1, \\ 0 & \text{se } B \leq x \leq M-1. \end{cases} \tag{4.19}$$

Podemos, agora, definir a convolução discreta de $f_e(x)$ e $g_e(x)$ pela expressão:

$$\begin{aligned} f_e(x) * g_e(x) &= \frac{1}{M} \sum_{m=0}^{M-1} f_e(m) g_e(x-m), \\ x &= 0, 1, 2, \ldots, M-1 \end{aligned} \tag{4.20}$$

Desta forma, temos a convolução como uma função discreta, periódica e de período M, sendo que os valores de x descrevem um período completo de $f_e * g_e$.

A convolução em duas dimensões é uma extensão da equação (4.16), assim:

$$f(x,y) * g(x,y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(\alpha, \beta) g(x-\alpha, y-\beta) d\alpha d\beta \tag{4.21}$$

O teorema da convolução toma a forma:

$$\begin{aligned} f(x,y) * g(x,y) &\rightleftharpoons F(u,v)G(u,v) \\ f(x,y)g(x,y) &\rightleftharpoons F(u,v) * G(u,v) \end{aligned} \tag{4.22}$$

A convolução discreta em duas dimensões é feita tomando $f(x,y)$ e $g(x,y)$ como matrizes de dimensão $A \times B$ e $C \times D$. Da mesma forma que no caso unidimensional deve-se assumir que f e g sejam periódicas com mesmo período, M e N, nas direções x e y. O surgimento indesejado de "aliasing" é prevenida escolhendo

$$\begin{aligned} M &\geq A + C - 1 \\ N &\geq B + D - 1 \end{aligned} \tag{4.23}$$

As seqüências periódicas são formadas extendendo $f_e(x,y)$ e $g_e(x,y)$ como segue:

$$f_e(x,y) = \begin{cases} f(x,y) & \text{se } 0 \leq x \leq A-1 \quad \text{e} \quad 0 \leq y \leq B-1, \\ 0 & \text{se } A \leq x \leq M-1 \quad \text{ou} \quad B \leq y \leq N-1. \end{cases} \tag{4.24}$$

$$g_e(x,y) = \begin{cases} g(x,y) & \text{se } 0 \leq x \leq C-1 \quad \text{e} \quad 0 \leq y \leq D-1, \\ 0 & \text{se } C \leq x \leq M-1 \quad \text{ou} \quad D \leq y \leq N-1. \end{cases} \tag{4.25}$$

Os cálculos envolvendo convolução bidimensional devem ser executados com as funções estendidas. A forma discreta da convolução é dada por:

$$\begin{aligned} f_e(x,y) * g_e(x,y) &= \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f_e(m,n) g_e(x-m, y-n), \\ x &= 0, 1, 2, \ldots, M-1 \\ y &= 0, 1, 2, \ldots, M-1 \end{aligned} \tag{4.26}$$

Utilizando o domínio freqüencial, o algoritmo FFT se torna uma ferramenta poderosa para o cálculo da convolução, diminuindo o número de operações executadas. Esta é o grande motivo da sua utilização e os filtros descritos adiante fazem uso desta propriedade. Inicialmente, as duas funções no domínio freqüencial são multiplicadas e então a transformada inversa é calculada, obtendo-se a convolução das duas funções no domínio temporal.

4.3.3.4 Correlação

A correlação entre duas funções $f(x)$ e $g(x)$, $f \circ g$ é definida por:

$$f(x) \circ g(x) = \int_{\infty}^{-\infty} \overline{f}(\alpha) g(x+\alpha) d\alpha \tag{4.27}$$

ou na forma discreta:

$$\begin{aligned} f(x) \circ g(x) &= \frac{1}{M} \sum_{m=0}^{M-1} \overline{f_e}(m) g(x+m) \\ x &= 0, 1, 2, \ldots, M-1 \end{aligned} \tag{4.28}$$

onde $\overline{f}$ significa o conjugado complexo de f, e M é calculado da mesma forma que a convolução, para evitar "aliasing". Uma definição importante é que se $f(x)$ e $g(x)$ forem a mesma função a equação é chamada autocorrelação.

Analogamente à convolução temos, para a forma contínua e discreta em duas dimensões:

$$f(x,y) \circ g(x,y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \overline{f}(\alpha, \beta) g(x+\alpha, y+\beta) d\alpha d\beta \tag{4.29}$$

$$\begin{aligned} f_e(x) \circ g_e(x) &= \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \overline{\overline{f}_e}(m,n) g(x+m, y+n) \\ x &= 0, 1, 2, \ldots, M-1; \\ y &= 0, 1, 2, \ldots, N-1 \end{aligned} \tag{4.30}$$

O teorema da correlação a seguir se aplica aos casos contínuo e discreto, assumindo neste último que todas as funções extendidas são periódicas:

$$\begin{aligned} f(x,y) \circ g(x,y) &\rightleftharpoons \overline{F}(u,v) G(u,v) \\ \overline{f}(x,y) g(x,y) &\rightleftharpoons F(u,v) \circ G(u,v) \end{aligned} \tag{4.31}$$

O teorema da correlação possui como corolário

Teorema 4.1. *A transformada de Fourier da autocorrelação de uma função $f(x)$ é a densidade espectral $|F(u)|^2$.*

A demonstração deste teorema, proposto em [4] é de demonstração extremamente simples, $f(x,y) \circ f(x,y) \rightleftharpoons \overline{F}(u,v)F(u,v) = |F(u,v)|^2$, e possui aplicações em reconhecimento e interpretações de imagens, que não são tratadas no presente trabalho, .

4.4 Aplicações da transformada de Fourier em domínios freqüenciais

As técnicas básicas para utilização da Transformada de Fourier em imagens se baseia no teorema da convolução. Seja $g(x,y)$ uma imagem com algum tipo de degradação e $f(x,y)$ a imagem ideal, sem o efeito indesejado de $g(x,y)$.

Considera-se então que $g(x,y)$ é obtida pela convolução de f pelo operador $h(x,y)$, linear e invariante por translação:

$$g(x,y) = h(x,y) * f(x,y) \tag{4.32}$$

De forma análoga, [11] relaciona f e g pela fórmula:

$$g(x,y) = \int\int h\left(x,y,x',y'\right) f\left(x',y'\right) dx'dy' + \eta(x,y) \tag{4.33}$$

onde $h(x,y,x',y')$ é a função de degradação da imagem e $\eta(x,y)$ algum tipo de ruído aditivo contido na imagem. A função h por conseguinte depende da posição (α,β) do ponto na imagem ideal f. Na ausência de ruído externo a imagem degradada passa a ser descrita por $f(x',y') = \delta(x'-\alpha, y'-\beta)$, ou por $h(x,y,\alpha,\beta)$.

Se, excetuando a propriedade de translação, a degradação da imagem num ponto for independente da posição do ponto a PSF toma a forma $h(x,x',y-y')$, e (4.33) torna-se:

$$g(x,y) = \int\int h(x-x', y-y') f(x',y') dx'dy' \tag{4.34}$$

Usando transformada de Fourier de ambos os lados de (4.34), como em (4.44) obtém-se:

$$G(u,v) = H(u,v). \times F(u,v) \tag{4.35}$$

definindo $G(u,v)$, $F(u,v)$ e $H(u,v)$ as transformadas de Fourier de $g(x,y)$, $f(x,y)$ e $h(x,y)$. A notação $.\times$ aproveita a convenção usada pelo software Matlab, isto é, não é o produto entre as matrizes H e F na forma tradicional, ou seja, cada elemento de G como o produto interno das linhas de H com as colunas de F, e sim o produto entre os termos de mesma posição entre as duas matrizes, conforme o exemplo (4.36):

$$\begin{bmatrix} 50 & 10 & 12 \\ 40 & -25 & 12 \\ 1 & 0 & -1 \end{bmatrix} .\times \begin{bmatrix} -2 & 1 & -1 \\ 1 & 0 & 4 \\ 34 & -1 & 1 \end{bmatrix} = \begin{bmatrix} -100 & 10 & -12 \\ 40 & 0 & 48 \\ 34 & 0 & -1 \end{bmatrix} \tag{4.36}$$

A transformada $H(u,v)$ é chamada função de transferência ou função de degradação (sua magnitude é chamada modulação) e a função $h(x,y)$ é chamada resposta impulso do sistema ou PSF, do inglês *"Point Spread Function"*, que nada mais é do que o equivalente em duas dimensões da resposta impulso, e descreve o grau de espalhamento sofrido por um ponto luminoso ao percorrer um sistema óptico.

Uma imagem é descrita completamente por uma função $f(x,y)$, onde x e y são as posições ortogonais dos pixels e f dá a intensidade média de luminosidade no pixel. Conforme a equação (4.33), a adição de algum tipo de ruído $\eta(x,y)$ na imagem pode ser descrito matematicamente por: $g(x,y) = f(x,y) + \eta(x,y)$.

4.4.1 O modelo para a degradação das imagens

Como descrito da seção (4.4), a modelagem do processo de degradação é feito de forma teórica pelo operador linear H, que juntamente com a função de ruído age sobre a imagem obtida $f(x,y)$ para produzir a imagem degradada $g(x,y)$. O processo de restauração é realizado buscando uma aproximaçâo de $f(x,y)$ dado $g(x,y)$ e o operador H. Assim:

$$g(x,y) = H[f(x,y)] + \eta(x,y) \tag{4.37}$$

É útil que o operador H seja invariante por translaçôes, em outras palavras

$$H[f(x-\alpha, y-\beta)] = g(x-\alpha, y-\beta) \tag{4.38}$$

para todo $f(x,y)$ e quaisquer que sejam α e β.

Aplicando-se a funçâo impulso sobre a imagem $f(x,y)$ obtemos:

$$f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha,\beta)\delta(x-\alpha, y-\beta)d\alpha d\beta \tag{4.39}$$

Desprezando ruído na imagem, ou seja $\eta(x,y) = 0$, temos:

$$g(x,y) = H[f(x,y)] = H\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha,\beta)\delta(x-\alpha, y-\beta)d\alpha d\beta\right] \tag{4.40}$$

Dado que H é um operador linear, extendendo a propriedade da aditividade para integrais teremos:

$$g(x,y) = H[f(x,y)] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} H[f(\alpha,\beta)\delta(x-\alpha, y-\beta)]d\alpha d\beta \tag{4.41}$$

Utilizando a homogeneidade do operador H e o fato de que $f(\alpha,\beta)$ não depende de x e y:

$$g(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha,\beta)H\left[\delta(x-\alpha, y-\beta)\right] d\alpha d\beta \tag{4.42}$$

Neste caso o termo $h(x,\alpha,y,\beta) = H\left[\delta(x-\alpha, y-\beta)\right]$ é a PSF de H.

4.4.2 Imagens deterioradas por movimento uniforme

O procedimento a seguir para obtenção de uma função de restauração de imagens distorcidas por movimento uniforme é descrito em [4] e em [11].

Por hipótese seja uma imagem $f(x,y)$ que se move em linha reta e que $x_0(t)$ e $y_0(t)$ sejam as equações de variação no espaço em relação ao tempo nas direções x e y. Seja T o intervalo em que o diafragma da câmera fotográfica permaneça aberto, e que este seja o tempo de exposição luminosa sobre o filme. Temos, supondo movimento contínuo, que;

$$g(x,y) = \int_{-\frac{T}{2}}^{\frac{T}{2}} f[x-\alpha(t), y-\beta(t)]dt \tag{4.43}$$

e neste caso temos $g(x,y)$ a imagem deteriorada e $f(x,y)$ a imagem teórica (sem movimento). Aplicando a transformada de Fourier obtém-se:

$$G(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)exp[-j2\pi(ux+vy)]dxdy \tag{4.44}$$

Substituindo (4.43) em (4.44) teremos

$$G(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}[\int_{-\frac{T}{2}}^{\frac{T}{2}} f[x-\alpha(t), y-\beta(t)]dt]exp[-j2\pi(ux+vy)]dxdy \tag{4.45}$$

Invertendo a ordem de integração obtém-se:

$$G(u,v) = \int_{-\frac{T}{2}}^{\frac{T}{2}}\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f[x-\alpha(t), y-\beta(t)]exp[-j2\pi(ux+vy)]dxdy\right]dt \tag{4.46}$$

O termo dentro dos colchetes corresponde à transformada de Fourier da função deslocada, isto é,

$$f[x-\alpha(t), y-\beta(t)] \tag{4.47}$$

Utilizando a propriedade da translação e de termos $F(u,v)$ independente de t temos que:

$$G(u,v) = F(u,v)\int_{-\frac{T}{2}}^{\frac{T}{2}} [exp(-j2\pi(u\alpha(t)+v\beta(t))]\,dt \tag{4.48}$$

Para alcançar a forma

$$G(u,v) = H(u,v)F(u,v) \tag{4.49}$$

definimos

$$H(u,v) = \int_{-\frac{T}{2}}^{\frac{T}{2}} [exp(-j2\pi(u\alpha(t)+v\beta(t))]\,dt \tag{4.50}$$

Supondo que o movimento ocorra na horizontal com velocidade V teremos:

$$\begin{aligned} \alpha(t) &= Vt \\ \beta(t) &= 0 \end{aligned} \tag{4.51}$$

Substituindo em (4.50) teremos, desprezando a parte imaginária (que não forma a imagem):

$$H(u,v) = \frac{sin(\pi uVT)}{\pi uV} \tag{4.52}$$

De forma que o PSF $h(x,y)$ pode ser obtido pela transformada inversa de Fourier de (4.52).

No exemplo (4.4.2) temos a imagem original e em (4.4.2) o arrastamento horizontal de 32 pixels sobre a imagem.

A simulação é importante no sentido em que há um controle sobre o ruído adicionado, ou seja, não há a contaminação da imagem por agentes de captação, como a própria câmera ou scanner.

Chama-se máscara de PSF à função de degradação que gera a deterioração pretendida. No caso do movimento uniforme máscara é obtida criando um vetor com comprimento igual à quantidade de pixels deslocados e com valor igual ao inverso da quantidade destes pixels. No caso da imagem (4.4.2) utilizou-se a PSF abaixo, de comprimento 32:

$$PSF = \left[\frac{1}{32}, \frac{1}{32}, \ldots, \frac{1}{32}\right] \tag{4.53}$$

Deve-se observar que as equações (2.36) e (2.37) refletem o mesmo resultado das equações (4.53) e (4.52), desde que em (2.37) sejam desprezados os valores negativos de α. Sumariamente, se n for o total de pixels deslocados, então a função de degradação tem a sua PSF equivalente por um vetor com o número de pontos igual a n, e o valor de cada posição do vetor igual a $\frac{1}{n}$.

Distorção com n pixels horizontais:

$$PSF = \left[\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n}\right]; \quad n \text{ posições.} \tag{4.54}$$

Ou seja, (4.53) representa o PSF para imagens degradadas com movimento linear uniforme, com a escolha adequada da função contínua.

Deve-se completar este vetor com zeros formando uma matriz com a mesma dimensão da imagem, então aplicar FFT bidimensional com o cuidado de deixar o valor DC no centro da matriz. Esta operação é realizada dividindo a matriz em quatro quadrantes e invertendo o primeiro com o quarto e o segundo com o terceiro quadrantes. A convolução entre as duas matrizes gera a imagem distorcida. Neste caso teremos:

$$G(u,v) = H(u,v). \times F(u,v) \tag{4.55}$$

É importante salientar que a multiplicação usada entre a imagem e a transformada da função de degradação é uma multiplicação ponto a ponto - símbolo ($.\times$) - e não uma multiplicação usual de matrizes, como descrito em (4.36)- símbolo($\times$).

Em [11] há o modelo para construção de uma PSF para turbulência atmosférica

4.4.3 Restauração de imagens deterioradas por movimento uniforme

O método utilizado para restaurar imagens é a aplicação da equação:

$$\widetilde{f}(i,j) = \mathbb{F}^{-1}[F(u,v)] = \mathbb{F}^{-1}[R_{filtro}(u,v). * G(u,v)] \tag{4.56}$$

onde:

$\mathbb{F}^{-1}[\quad]$ é a transformada inversa de Fourier.

$\widetilde{f}(i,j)$ é a imagem restaurada, uma aproximação da imagem original.

R_{filtro} é o tipo de filtro utilizado na restauração.

4.4.4 Filtro Inverso

Assumindo que uma dada imagem não apresenta ruído podemos concluir que da equação (4.37) no domínio espectral temos que:

$$G(u,v) = H. \times F(u,v) + 0 \tag{4.57}$$

e a transformada de Fourier da imagem original pode ser calculada usando a equação:

$$F(u,v) = (1./H) \times G(u,v) \tag{4.58}$$

Assim a restauração da imagem original torna-se:

$$f(i,j) = (1./H) \times G(u,v) = \mathbb{F}^{-1}\left[(1./H) \times G(u,v)\right] \tag{4.59}$$

De forma simplificada o filtro inverso obtém a aproximação da imagem multiplicando ponto a ponto a forma degradada $G(u,v)$ por $\dfrac{1}{H(u,v)}$, sendo que a $\dfrac{1}{H(u,v)}$ represente a inversão ponto a ponto de cada termo da matriz e não a inversão da matriz.

Na prática é comum que vários pontos de $H(u,v)$ são nulos e isto provoca divisão por zero na aplicação do filtro inverso. Outro problema comum é a transformada $G(u,v)$ também possuir zeros no plano uv, o que pode gerar indeterminações na equação (4.58), de forma que mesmo na ausência de ruído a

reconstrução torna-se impossível ou gera imagens com extrema perda de nitidez, uma vez que os valores tornam-se muito altos pelo fator da divisão por zero ou por valores na vizinhança de zero.

No caso da restauração de imagens deterioradas por movimento uniforme a função $H(u, v)$ é dada por (4.52). Esta função, que depende apenas de u é mostrada na figura (4.4.4):

Em [11] há uma aproximação da função (4.52) que possibilita o uso do filtro inverso para restauração de degradação causada por movimento linear, que todavia é pobre na qualidade da imagem corrigida, e por isso este procedimento não foi implementado.

4.4.5 Filtro de Wiener

O filtro de Wiener serve para diminuir os problemas de divisão por zero obtidos no filtro inverso. O filtro é definido por:

$$R_w = \frac{H^*(u,v)}{|H(u,v)|^2 + \left[\dfrac{S_n(u,v)}{S_f(u,v)}\right]} \tag{4.60}$$

Onde são dados:
$H^*(u, v)$ = complexo conjugado da matriz $H(u, v)$.
$S_n(u, v)$ = densidade espectral do ruído $\eta(x, y)$.
$S_f(u, v)$ = densidade espectral da imagem original.

Em aplicações práticas a imagem original freqüentemente é desconhecida, então o valor da densidade espectral da imagem é substituída por um parâmetro K, cujo valor ótimo é deteminado experimentalmente.

Desta forma tem-se:

$$R_w = \frac{H^*(u,v)}{|H(u,v)|^2 + K} \tag{4.61}$$

Concorde [13] é interessante manter K como um parâmetro dependente dos valores (u, v) do domínio freqüencial, uma vez que os valores de ruído costumam ser dominantes em freqüências altas. Assim conforme a freqüência aumente é interessante que K aumente.

Existem outros efeitos indesejáveis em imagens onde o filtro de Wiener pode ser aplicado obtendo uma boa restauração. Fotografias antigas costumam sofrer o ataque de traças, e isto gera uma deterioração conhecida na literatura como *'salt and pepper'*

Figura 4.1: Imagem superior esquerda com 328 por 496 pixels, superior direita com 66 por 99 pixels.
Imagem inferior esquerda com 33 por 50 pixels e inferior direita com 16 por 25 pixels.

Figura 4.2: Imagem simples (30×30 pixels) e a densidade espectral da imagem com valores na forma $log[1 + |F(u, v)|]$ (256×256 pixels), onde o array de entrada foi completado com zeros até ter o tamanho de 2^8.

Figura 4.3: Imagem Original - Densidade Espectral

Figura 4.4: Imagem com arrasto de 32 pixels na horizontal - Densidade Espectral

Figura 4.5: Gráfico da função (4.52)

Figura 4.6: Imagem Restaurada usando filtro de Wiener - Densidade Espectral

Figura 4.7: Imagem superior com aproximadamente 10 pixels deslocados.
Imagem inferior restaurada usando a equação (4.61), com K constante e PSF dado por (4.54).

5 RESOLUÇÃO DA EQUAÇÃO DE POISSON VIA FFT

5.1 Introdução

Existem vários métodos para resolver numericamente a equação de Poisson, e alguns destes são chamados 'Fast Poisson Solvers', ou (FPS), por serem mais eficazes do que o método da eliminação gaussiana.

Dentre todos os FPS o de mais fácil implementação é o via FFT. Neste caso a transformada rápida de Fourier é utilizada para aprimorar o custo computacional em sistemas de equações geradas por diferenças finitas ou elementos finitos a partir da discretização da equação de Poisson. Este capítulo trata da resoluçâo da equação num domínio retangular com condições de contorno de Dirichlet.

A grande vantagem da FFT é preservar a precisão dos cálculos e baixar o tempo de processamento para a complexidade de $O(n^2 logn)$. A título de comparação, o método da eliminação gaussiana opera com a ordem de $O(n^4)$, com o mesmo erro de truncamento. Sob este ponto de vista a FFT pode ser comparada com um dos melhores métodos para obter a solução numérica para a equação de Poisson.

5.2 Discretização da equação de Poisson

A equação de Poisson sujeita às condições de contorno de Dirichlet:

$$\begin{aligned} &\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x,y), \quad para(x,y) \in \Omega; \\ &u(x,y) = \phi(x,y), \quad para(x,y) \in \partial\Omega; \end{aligned} \tag{5.1}$$

Onde $\Omega = (0,1) \times (0,1)$ é um quadrado unitário e $\partial\Omega$ sua respectiva fronteira.

À fim de discretizar a equação diferencial o domínio Ω é coberto com uma malha com passo $h = 1/(N+1)$, sendo que cada ponto (x_i, y_i) possui a representação:

$$x_i = ih \quad \text{e} \quad y_j = jh \quad \text{para} \quad i, j = 0, 1, \ldots, N+1. \tag{5.2}$$

Os pontos acima descritos estão dispostos de forma que se i ou j forem iguais a 0 ou $N+1$ então estarão representando pontos na fronteira e todos os demais representam pontos interiores. A resolução do problema resume-se em encontrar uma aproximação dos valores de $u(x_i y_i)$ para cada um dos pontos interiores.

Aplicando o método das diferenças finitas de segunda ordem em aproximações centrais, conforme descrito em [10], temos os pontos interiores tendendo a:

$$\frac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{h^2} + \frac{u_{i,j-1} - 2u_{i,j} + u_{i,j+1}}{h^2} = f(x_j, y_i) \tag{5.3}$$

Na equação (5.3) por conveniência considera-se que os pontos da fronteira Ω estão incluídos. Podemos descrever o sistema matricialmente por:

$$T_N U + U T_N = h^2 \widetilde{F} \tag{5.4}$$

onde T_N representa a matriz tridiagonal $(-1, 2, -1)$ e define-se $\widetilde{F}$ como o conjunto obtido pelos pontos $F_{i,j}$, interiores de Ω, e pelos pontos $\phi(x_i, y_i)$ da fronteira $\partial\Omega$.

5.3 Resolução do sistema linear

Encontra-se a solução do autosistema da matriz T_N supondo que esta é diagonalizável. Sendo assim chamando V a matriz dos autovetores e L a matriz diagonal dos autovalores teremos:

$$T_N = V L V^{-1} \tag{5.5}$$

de forma que o sistema torna-se:

$$V L V^{-1} U + U V L V^{-1} = h^2 \widetilde{F} \tag{5.6}$$

Multiplicando (5.6) por V^{-1} e por V, respectivamente, teremos a equação:

$$LV^{-1}UV + V^{-1}UVL = h^2V^{-1}\widetilde{F}V \tag{5.7}$$

A fim de obter mais simplicidade em (5.7) definimos:

$$\overline{U} = V^{-1}UV \tag{5.8}$$

$$\overline{F} = h^2V^{-1}\widetilde{F}V \tag{5.9}$$

Uma vez que:

$$L(j,j)\overline{U}(j,k) + \overline{U}(j,k)L(k,k) = \overline{F}(j,k) \tag{5.10}$$

teremos:

$$\overline{U}(j,k) = \frac{\overline{F}}{L(j,j) + L(k,k)} \tag{5.11}$$

Este problema resolvido por métodos matriciais tem ordem de complexidade $O(N^4)$, o que representa um custo computacional elevado. A transformada rápida de Fourier, conforme mencionado em (5.1) pode então ser aplicada ao problema, preferencialmente, quando a malha é dividida em número de células igual a uma potência de dois.

5.3.1 Aplicação da Transformada Rápida de Fourier

A transformada rápida de Fourier se torna aplicável na resolução da equação de Poisson uma vez que a matriz T_N em (5.3) é da forma TST (Toeplitz, simétrica e tridiagonal), que possui autovalores conhecidos pela conjectura de Fischer-Hartwig, como citado em [3] e [5]. Aqui, de forma simplificada mostra-se uma forma de se obter os autovalores e autovetores da matriz TST obtida da discretização sugerida:

Da equação 5.3 define-se:

$$T_m = \begin{bmatrix} -2 & 1 & 0 & \dots & \dots & \dots & 0 \\ 1 & -2 & 1 & 0 & \dots & \dots & 0 \\ 0 & & & \ddots & & & \vdots \\ \vdots & & & & 1 & -2 & 1 \\ 0 & 0 & & \dots & & 1 & -2 \end{bmatrix} \tag{5.12}$$

Temos as identidades trigonométricas conhecidas:

$$sin((k+1)\theta) = cos(\theta).sin(k\theta) + sin(\theta).cos(k\theta) \tag{5.13}$$

$$sin((k-1)\theta) = cos(\theta).sin(k\theta) - sin(\theta).cos(k\theta) \tag{5.14}$$

Somando as equações (5.13) e (5.14) teremos:

$$sin((k-1)\theta) - 2.cos(\theta).sin(k\theta) + sin((k+1)\theta) = 0 \tag{5.15}$$

A título de simplificação faz-se $sin(K\theta) = S_K$ e $cos(K\theta) = C_K$, para $0 \leq \theta \leq \frac{\pi}{2}$. Assim temos a propriedade:

$$T_m \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_k \\ \vdots \\ S_m \end{bmatrix} = \begin{bmatrix} S_0 - 2S_1 + S_2 \\ S_1 - 2S_2 + S_3 \\ \vdots \\ S_{k-1} - 2S_k + S_{k+1} \\ \vdots \\ S_{m-1} - 2S_m \end{bmatrix} \tag{5.16}$$

Da equação (5.13) conclui-se que:

$$S_{k-1} - 2.S_k + S_{k+1} = 2S_k(1 + C_1) \tag{5.17}$$

Na segundo termo da igualdade (**??**) o primeiro item da matriz foi completado com S_0 uma vez que sen(0) = 0, porém a última posição do vetor está incompleto, assim usando a equação (5.17) obtemos:

$$T_m \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_k \\ \vdots \\ S_m \end{bmatrix} = 2.(C_1 - 1) \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_k \\ \vdots \\ S_m \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ (S_{m-1} - 2.S_m) - 2(C_1).S_m \end{bmatrix} \tag{5.18}$$

A última linha do segundo vetor de (5.18) é equivalente a

$$S_{m-1} - 2.S_m - 2.C_1.S_m = S_{m-1} - 2C_1.S_m \tag{5.19}$$

e usando a equação (5.13) tem-se:

$$S_{m-1} - 2.C_1.S_m = -S_{m+1} \tag{5.20}$$

o que nos leva à matriz:

$$T_m \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_k \\ \vdots \\ S_m \end{bmatrix} = 2.(C_1 - 1) \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_k \\ \vdots \\ S_m \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ S_{m+1} \end{bmatrix} \tag{5.21}$$

Tornando $n = m - 1$ temos:

$$T_{n-1} \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_k \\ \vdots \\ S_{n-1} \end{bmatrix} = 2.(C_1 - 1) \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_k \\ \vdots \\ S_{n-1} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ S_n \end{bmatrix} \tag{5.22}$$

Fazendo a substituição $\theta_j = \frac{j\pi}{n}$ teremos $sin(n\theta_j) = sin(\pi) = 0$, e portanto:

$$T_{n-1}\begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_k \\ \vdots \\ S_{n-1} \end{bmatrix} = 2.(C_1 - 1)\begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_k \\ \vdots \\ S_{n-1} \end{bmatrix} \tag{5.23}$$

De uma forma mais clara escreve-se:

$$T_{n-1}\begin{bmatrix} sin(\theta_j) \\ sin(2\theta_j) \\ \vdots \\ sin((n-2)\theta_j) \\ sin((n-1)\theta_j) \end{bmatrix} = 2.(cos(\frac{j\pi}{2n}) - 1)\begin{bmatrix} sin(\theta_j) \\ sin(2\theta_j) \\ \vdots \\ sin((n-2)\theta_j) \\ sin((n-1)\theta_j) \end{bmatrix} \tag{5.24}$$

Definindo a matriz $V_{(n\times 1, n\times 1)}$ por:

$$\left[V_{(1:n-1,1:n-1)}\right]_{kj} = sin(\frac{kj\pi}{n}), \quad 0 \leq k, j \leq 2n-1 \tag{5.25}$$

teremos $V(1:n-1, j)$ um autovetor da equação (5.24), ou seja:

$$T_{n-1}V(1:n-1, j) = \lambda_j V(1:n-1, j) \tag{5.26}$$

onde define-se $\lambda_j = 2(cos\frac{jk\pi}{n} - 1)$. Desta forma tem-se para $j = 1 : n-1$:

$$V^{-1}T_{n-1}V = \begin{bmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_{n-1} \end{bmatrix} = L \tag{5.27}$$

Os autovalores e autovetores tornam-se:

$$L(j,j) = 2(cos(\frac{j\pi}{n}) - 1) \quad \text{para} \quad 1 \leq j \leq n \tag{5.28}$$

e

$$V(j,k) = sin(\frac{j\pi k}{n}) \quad \text{para} \quad 1 \leq j,k \leq n \tag{5.29}$$

Obtém-se desta forma um método rápido para calcular a inversa de T_{n-1}:

$$\begin{gathered} V^{-1}T_{n-1}V = L \Leftrightarrow L^{-1}V^{-1}T_{n-1} = V^{-1} \Leftrightarrow \\ L^{-1}V^{-1} = V^{-1}T_{n-1}^{-1} \Leftrightarrow T_{n-1}^{-1} = VL^{-1}V^{-1} \end{gathered} \tag{5.30}$$

Dado um vetor x, com n valores, define-se o vetor extendido xx por:

$$xx = (0, x_1, \ldots, x_n, \ldots, 0) \tag{5.31}$$

tal que xx possua $2n$ valores.

Da matriz de Fourier definida em (2.30) é fato que

$$\begin{gathered} \mathbb{F}_{(j,k)} = [f_{jk}], \quad \text{e} \quad f_{jk} = \omega_n^{jk}; \quad 0 \leq j,k \leq m-1. \\ \omega_n = cos(\tfrac{2\pi}{n}) - isin(\tfrac{2\pi}{n}); \quad i = \sqrt{-1} \end{gathered} \tag{5.32}$$

Escrevendo o vetor extendido xx com comprimento $2n$ temos:

$$\mathbb{F}(xx) = cos(\frac{2\pi jk}{2n}) - isin(\frac{2\pi jk}{2n}) = cos(\frac{\pi jk}{n}) - isin(\frac{\pi jk}{n}) \tag{5.33}$$

Comparando as equações (5.29) e (5.33) verifica-se a parte imaginária da matriz de Fourier gera os autovetores $V(j,k)$ de T_N. Conseqüentemente dispõe-se da igualdade:

$$DST_x = Im(\mathbb{F}_{xx}(1:n)) \tag{5.34}$$

onde $\mathbb{F}_{xx}$ é a DFT do vetor extendido (xx).

Uma vez que a parte imaginária da DFT é chamada DST (Discrete Sine Transform), e conforme verificado no capítulo 2, a inversa da DST (IDST) tem a seguinte equação:

$$IDST = \frac{2}{n}DST \tag{5.35}$$

Por conseguinte a questão de encontrar os autovetores da matrix T_N resume-se em calcular a DST (via FFT) da matriz de Fourier de dimensão $2n$.

Utilizando este modelo temos uma forma eficiente de calcular a solução da equação de Poisson, porém para que o método seja satisfatório devemos considerar que a seqüência de entrada tenha valores igualmente espaçados e que seja de comprimento igual a uma potência de 2, conforme discutido anteriormente.

5.3.2 Algoritmo

Devemos resolver a equação (5.11). Os passos devem ser:

5.3.2.1 Obtenção de (5.9)

Devemos encontrar $\overline{F} = V^{-1}\widetilde{F}V$. De [7] temos:

$$T_{n-1}^{-1} = \frac{2}{n}L^{-1} \tag{5.36}$$

Aplicando (5.36):

$$\overline{F} = \frac{2}{n}V\widetilde{F}VT_{n-1}^{-1} = \frac{2}{n}L^{-1} \tag{5.37}$$

De forma análoga:

$$U = V\overline{U}V^{-1} = V(\frac{2}{n}\overline{U})V^{-1} \tag{5.38}$$

A equação (5.11) nos dá como resposta:

$$U(j,k) = \frac{2}{n}V\left(\frac{\overline{F}(j,k)}{L(j,j)+L(k,k)}\right)V \tag{5.39}$$

5.3.2.2 Pseudo-código para aquisição de U

(1) Aplicar FFT bidimensional aos valores de $F(i,j)$.

(2) Dividir cada transformada $F(i,j)$ por $L(j,j)+L(k,k)$.

(3) Aplicar FFT bidimensional inversa aos valores de $F(j,k)$.

Tendo em vista que a aplicação da FFT bidimensional em uma matriz equivale a aplicar FFT sobre as linhas e consecutivamente sobre as colunas descreve-

se o algoritmo:

função x =FFT-Poisson(F,h)

(n,m) = dimensão de F

Para i variando de 1 até n faça // (Cômputo dos autovalores)

$L(i,i) = 2(1 - cos(\frac{i\pi}{2x}))$

$L(i,j) = 0$, *parai* $\neq j$

pare

// (FFT bidimensional em F)

$x =$ DST(F') (F' é a transposta de F)

$x =$ DST(x')

$x = \frac{2}{n} * x$

Para i variando de 1 até n faça

Para j variando de 1 até n faça

$x(i,j) = \frac{x(i,j)}{\frac{1}{h^2}(L(j) + L(i))}$

pare

pare

//(Transformada inversa)

$x = DST(x')$

$x = \frac{2}{n} * x'$

$x = DST(x)$

6 CONCLUSÃO

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Blahut, R. E. *Fast algorithms for digital signal processing.* Addison-Wesley Publishing Company, 1985.

[2] Brigham, E. *The Fast Fourier Transform.* Prentice Hall, New Jersey, 1974.

[3] Estelle L. Basor, K. E. M. The fisher-hartwig conjecture and toeplitz eigenvalues. Tech. rep., California Polytechnic State University, 1992.

[4] Gonzalez, R. C., and Woods, R. E. *Digital Image Processing*, 2nd ed. Addison-Wesley Publishing Company, Massachussets, 1993.

[5] Iserles, A. Numerical analysis lecture 17. Tech. rep., Cambridge Numerical Analysis Group, 2004.

[6] Jähne, B. *Digital Image Processing*, 4th ed. Springer Verlag Berlim, Heilderberg, 1997.

[7] Loan, C. Computational frameworks for the fast fourier transform. *SIAM, Philadelphia* (1992), 229–258.

[8] Nussbaumer, H. J. *Fast Fourier Transform and Convolution Algorithms.* Springer-Verlag, Berlin, 1981.

[9] Oppenhein, A. V., and Schafer, R. W. *Digital Signal Processing.* Prentice Hall Inc., Englewood Cliffs, New Jersey, 1974.

[10] Richard L. Burden, J. F. *Análise Numérica.* Thompson Publishers.

[11] Rosenfeld, A., and Kak, A. C. *Digital Picture Processing*, vol. 1 of . Academic Press, San Francisco, 1982.

[12] Rudin, W. *Real and Complex Analysis*, terceira edição ed. Mc Graw Hill International Editions, 1987.

[13] Sezan, M. I., and Lagendijk, R. L. *Motion Analysis and Image Sequence Processing*. Kluwer Academic Publishers, Boston, 1993.

[14] Sneddon, I. N. *Fourier Transforms*. Dover Publications, New York, 1995.

[15] Snir, M., and Otto, S. W. *MPI The Complete Reference*. MIT Press, Massachussetts, USA, 1996.

[16] Spiegel, M. R. *Fourier Analysis with applications to boundary value problems*. Mc Graw Hill, New York, 1974.

[17] Strang, G. *Linear Algegra and its applications*. Harcourt Brace Jovanovich Publishers, San Diego, 1988.

[18] Umbaugh, S. E. *Computer Vision and Image Processing*. Prentice Hall, New Jersey, 1998.