CONVOLUCIONADOR DE IMAGENS

Maria Fernanda Eppinghaus Belford Roxo
UFRJ-NCE
Caixa Postal 2324 - CEP 20001 - Rio de Janeiro - RJ

RESUMO - Este trabalho apresenta um acelerador para o processamento da convolução de imagens. É proposto um circuito a ser ligado a um microcomputador compatível com o IBM-PC XT[1] que, aliado a uma rotina em Turbo Pascal 4.0[2], multiplica a velocidade deste processo em relação aos métodos convencionais.

1. INTRODUÇÃO

Ao se trabalhar com processamentos envolvendo imagens, frequentemente depara-se com problemas de tempo. Estes problemas se devem, principalmente, a grande massa de dados envolvida. Para se ter uma idéia, para uma imagem de boa resolução (512 linhas por 512 colunas), cada ciclo de máquina economizado numa operação, representa mais de 260 mil ciclos no processamento como um todo. Por isto, os programas que visam o tratamento de imagens tem como preocupação fundamental a otimização dos algorítmos. Na prática, porém, comprova-se que mesmo utilizando-se algorítmos eficientes, o tempo relativo a alguns processamentos ainda é muito alto, tornando imprescindível uma solução por *hardware*.

Este projeto faz parte do sistema VIR (Visão e Inspeção Robótica), atualmente sendo desenvolvido no Núcleo de Computação Eletrônica (NCE) da UFRJ. Ele visa agilizar um processo muito utilizado no tratamento de imagens, que se mostrou extremamente lento durante o desenvolvimento do sistema: a convolução.

Inicialmente, cabe apresentar algumas características do sistema,

---

[1] IBM é marca registrada da International Business Machines Corporation.

[2] Turbo Pascal é marca registrada da Borland International.

relevantes para o perfeito entendimento das soluções propostas por este trabalho. É utilizado um microcomputador compatível com o IBM-PC XT, configurado com: relógio de 4,77MHz, 704 Kbytes de memória RAM, microprocessador 8088 e co-processador 8087. Os programas são escritos em Turbo Pascal 4.0 e as imagens são armazenadas com: 8 bits por célula (256 tons de cinza) e dimensões horizontal e vertical variáveis (desde que a imagem não ultrapasse a memória RAM disponível).

2. CONVOLUÇÃO

A convolução é um procedimento que minimiza os efeitos produzidos por cada célula individualmente, levando em consideração a influência das células vizinhas. Ela consiste em varrer toda a imagem e, para cada célula, encontrar um novo valor, levando em consideração a própria célula e suas vizinhas, podendo atribuir diferentes pesos a cada uma delas.

Para isto é criada uma matriz de covolução que define o peso da própria celula e o de cada vizinha. A escolha do tamanho e dos valores desta matriz vai depender do objetivo da convolução.

Além da matriz de convolução, mais dois fatores podem ser utilizados no cálculo da nova imagem: o divisor e o patamar. O resultado do somatório das multiplicações de cada célula por seu peso é dividido pelo divisor e depois somado ao patamar. O divisor normalmente equivale à soma dos termos da matriz de convolução, embora isto não seja uma regra. Estes dois fatores tem por objetivo manter a maioria dos resultados dentro dos limites de armazenamento do sistema, isto é, entre zero e o maior nível de cinza.

Ao final dos cálculos, os resultados obtidos são normalizados, isto é, os resultados negativos são igualados a zero e os que ultrapassarem o maior nível de cinza são igualados ao mesmo.

A vasta utilização da convolução se deve a sua versatilidade. Ela comporta uma série de aplicações, escolhidas através da definição dos seus parâmetros.

3. TEMPO DE PROCESSAMENTO

O excessivo número de multiplicações necessárias para a execução de uma convolução é o principal limitador de velocidade do processo. Isto porque

o microprocessador utilizado (8088) é extremamente lento na execução deste tipo de operação.

Uma forma de contornar este problema é a criação de uma tabela, onde ficam armazenados os resultados da multiplicação de qualquer célula (valores de 0 a 255 para o sistema em questão) por qualquer fator da matriz de convolução. O tempo gasto na preparação desta tabela é perfeitamente compensado pelo número de multiplicações executadas durante o processamento. Com a tabela, o tempo de cada multiplicação cai sensivelmente, mas ela ainda não pode ser considerada uma instrução rápida, pois são necessários alguns cálculos para a indexação da tabela. o que torna o seu endereçamento um pouco lento.

A divisão, embora seja executada menos vezes, também é um importante fator de limitação da velocidade. Neste caso, a criação de uma tabela não é viável, pois ela ocuparia praticamente toda a memória RAM do microcomputador.

Por último, o fato de uma imagem ser um vetor bidimensional, faz com que o seu endereçamento seja mais lento, contribuindo para o aumento do tempo de processamento.

4. SOLUÇÕES PROPOSTAS

A solução proposta consiste de um circuito ligado a um *slot* do microcomputador e uma rotina em Turbo Pascal 4.0. Esta solução limita alguns parâmetros da convolução, visando facilitar a sua implementação. Os limtes impostos às variáveis do sistema são:

- 256 tons de cinza, isto é, cada célula da imagem é composta por 8 bits.
- máximo de 1024 células para cada linha da imagem.
- matriz de convolução com 3 linhas e 3 colunas.
- elementos da matriz de convolução e patamar limitados ao intervalo de -128 a +127.
- divisor limitado ao intervalo de 1 a 127.

A única limitação grave é quanto às dimensões da matriz de convolução. É possível processar a convolução por matrizes maiores através de recursos de programação, mas o processamento perde um pouco da sua rapidez e precisão.

É utilizada uma tabela de multiplicação nos mesmos moldes da tabela apresentada no ítem anterior. Porém, esta tabela não pertence à memória do microcomputador. Ela fica armazenada numa memória RAM localizada no circuito externo ao microcomputador. A rotina é encarregada de calcular e enviar para o circuito os valores da tabela. Além de economizar memória no microcomputador, esta memória tem um tempo de acesso mais rápido e uma indexação imediata. Na saída da tabela é colocado um acumulador, que executa a soma necessária após cada multiplicação. Desta forma, cada multiplicação/soma leva apenas um ciclo de máquina para ser executada.

Para otimizar os acessos à memória, a imagem é transmitida para o circuito apenas uma vez, linha por linha, por DMA (acesso direto à memória). Da mesma forma, a imagem resultante é transferida do circuito para o microprocessador, linha por linha, por DMA. Isto soluciona os problemas de número de leituras e forma de acesso à cada célula da imagem. Agora, cada célula é lida da memória do microcomputador uma única vez e de forma sequencial, o que facilita o seu endereçamento. O circuito possui uma memória RAM denominada MAT que armazena as duas últimas linhas da imagem enviadas pelo microcomputador. Esta memória é repartida logicamente em MAT [0] e MAT [1], separando assim as duas linhas aí armazenandas.

Para que o processamento seja feito linha a linha, é necessário o armazenamento de resultados parciais, já que a obtenção de um resultado final só é possível após o cálculo sobre três linhas. Para isto, o circuito proposto possui uma memória RAM denominada MAT_TEMP.

Há também uma memória RAM denominada NEW_MAT que armazena uma linha do resultado final da convolução.

Na convolução de uma imagem **f** por uma matriz de convolução **k**, o elemento da coluna **h**, da linha **v**, da imagem resultante **g** é dado por:

```
g[v,h] = ( k [0,0] * f [v-1,h-1] + k [0,1] * f [v-1,h] +
           k [0,2] * f [v-1,h+1] + k [1,0] * f [v,h-1] +
           k [1,1] * f [v  ,h]   + k [1,2] * f [v,h+1] +
           k [2,0] * f [v+1,h-1] + k [2,1] * f [v+1,h] +
           k [2,2] * f [v+1,h+1] ) / divisor + patamar
```

Para esta mesma convolução, é proposto um cálculo por etapas da seguinte forma:

```
ETAPA 1 → MAT_TEMP [h-1] = patamar * divisor +
                           k [0,0] * f [v-1,h-1] +
                           k [0,1] * f [v-1,h]   +
                           k [0,2] * f [v-1,h+1]
ETAPA 2 → MAT_TEMP [h-1] = MAT_TEMP [h-1] +
                           k [1,0] * f [v, h-1] +
                           k [1,1] * f [v, h]   +
                           k [1,2] * f [v, h+1] +
ETAPA 3 → SHIFTER_A =      MAT_TEMP [h-1] +
                           k [2,0] * f [v+1,h-1] +
                           k [2,1] * f [v+1,h]   +
                           k [2,2] * f [v+1,h+1]
ETAPA 4 → NEW_MAT [h-1] =  SHIFTER_A / divisor
```

Obs: SHIFTER_A é a denominação dada a entrada do circuito divisor.

Os resultados armazenados em NEW_MAT ficam, assim, prontos para serem transferidos para o microcomputador.

Outro artifício utilizado para a aceleração da convolução é o da concorrência de processos, como pode ser visto na tabela 1 que mostra as diversas etapas do cálculo da convolução.

Observa-se que, inicialmente, é necessário enviar três linhas da imagem original para se obter uma linha da nova imagem. A partir daí, para cada linha da imagem original enviada, obtem-se uma linha da nova imagem, pois as duas linhas anteriores já estão armazenadas em MAT. Isto diminui o número de transferências entre o microcomputador e o circuito do convolucionador.

A maior aceleração, no entanto, é decorrente da concorrência entre os processos de: transferência de dados, cálculo (multiplicação/acumulação) sobre uma linha e divisão/armazenamento de um novo resultado. Este tipo de concorrência é de grande valia para a maior velocidade do processo pois, enquanto uma célula está sendo dividida, a próxima já está sendo calculada, ou enquanto uma linha da nova imagem está sendo transferida, o cálculo da próxima linha já está sendo executado.

Por último, o mesmo artifício utilizado no armazenamento das linhas é também utilizado para as colunas. Sendo assim, no cálculo sobre uma linha, cada coluna só precisa ser lida uma única vez. Para isto, ao invés de utilizar

as memórias MAT[0] e MAT[1], são utilizados *latches* denominados LAT [0] e LAT [1], de acesso ainda mais rápido do que uma memória RAM. Da mesma forma, para o armazenamento dos resultados parciais, no lugar da memória MAT_TEMP, é utilizado um registrador denominado AC (acumulador) para o armazenamento dos resultados parciais.

transferência de → para	cálculo sobre linha	divisor	resultado armazenado em
lin0 → MAT[0]	lin0	inativo	MAT_TEMP
lin1 → MAT[1]	lin1	inativo	MAT_TEMP
lin2 → MAT[0]	lin2	ativo	NEW_MAT
NEW_MAT → n_lin1	MAT[1] (lin1)	inativo	MAT_TEMP
	MAT[0] (lin2)	inativo	MAT_TEMP
lin3 → MAT[1]	lin3	ativo	NEW_MAT
NEW_MAT → n_lin2	MAT[0] (lin2)	inativo	MAT_TEMP
	MAT[1] (lin3)	inativo	MAT_TEMP
lin4 → MAT[0]	lin4	ativo	NEW_MAT
NEW_MAT → n_lin3	MAT[1] (lin3)	inativo	MAT_TEMP
...	...	...	...

Obs: lin x → linha x da imagem original
n_lin x → linha x da imagem resultante

TABELA 1 - Etapas de uma convolução mostrando a concorrência entre processos.

A figura 1 apresenta um diagrama de blocos com a estrutura básica do circuito acima proposto. Com esta estrutura, a rotina passa a executar uma função de gerenciamento, tendo as seguintes atribuições:

- igualar a zero as bordas da imagem resultante.
- enviar os parâmetros da convolução para o circuito (número de colunas da imagem, patamar x divisor e -divisor).
- montar e enviar para o circuito a tabela de multiplicação.
- programar a pastilha controladora de DMA (8237-5) para a transfer ncia das imagens original e resultante.
- enviar comandos de controle para ativar os processos no circuito.
- controlar o fim dos processos no circuito.

5. CONCLUSÕES

Após a fase de projeto, foram executadas algumas simulações, visando

provar a viabilidade do convolucionador e estimar a velocidade do mesmo. Os resultados das simulações mostram que o sistema é bastante veloz em relação aos métodos até então utilizados, cumprindo assim o principal objetivo do projeto.

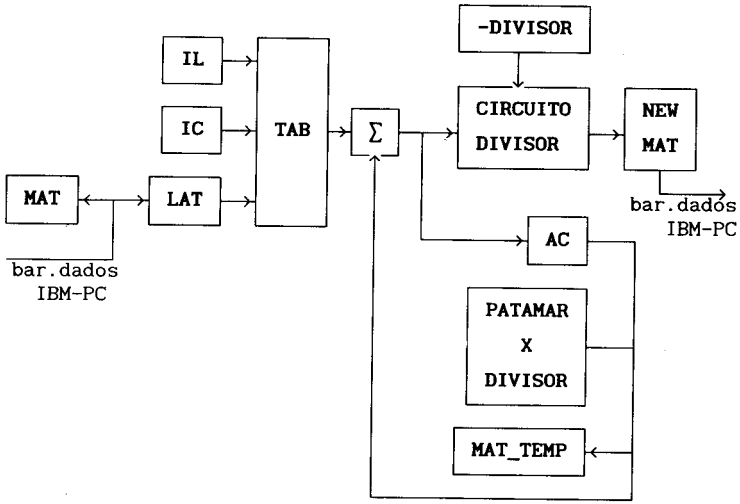

FIGURA 1 - Diagrama de blocos do convolucionador.

Dimensões da imagem		Tempos de Processamento da Convolução		
hor	ver	método convencional	convolucionador (simulação)	fator de ganho
64	64	25,38 s	0,25 s	101
128	64	51,41 s	0,33 s	154
128	128	104,52 s	0,51 s	206
256	128	210,37 s	0,94 s	225
256	256	424,08 s	1,23 s	344
512	256	851,07 s	1,90 s	447

TABELA 2 - Comparação entre os tempos de processamento da convolução.

A Tabela 2 mostra alguns resultados da comparação entre os tempos de processamento da convolução pelos métodos usuais e pela simulação do projeto. Observa-se facilmente que, quanto maiores as dimensões da imagem, mais vantajosa se torna a utilização deste circuito.

Este projeto abre um caminho para as soluções específicas de baixo custo visando o tratamento de imagens. Para uma solução mais genérica, entretanto, seria interessante a utilização de uma arquitetura paralela do tipo SIMD (*Single Instruction Multiple Data*), baseada em microprocessadores com acessos à memória eficientes e processamentos aritméticos rápidos.

BIBLIOGRAFIA

[1] PAVLIDIS, Theo.
Algorithms for Graphics and Image Processing
Computer Science Press - 1982

[2] PAZ, E. P.; CUNHA, T. N.
Iniciação ao Processamento Digital de Imagens
NCE/UFRJ - 1988

[3] EGGEBRECHT, L. C.
Interfacing to the IBM Personal Computer
Howard W. Sams & Co. - 1983