PROJETO DE UM MÓDULO DE HARDWARE PARA OTIMIZAÇÃO DA REPRESENTAÇÃO E DESCRIÇÃO DE IMAGENS SEGMENTADAS ATRAVÉS DA MORFOLOGIA MATEMÁTICA EM SISTEMAS EMBARCADOS

Luiz Antonio de Oliveira Júnior[1]; Edna Natividade da Silva Barros[2]

[1]Estudante do Curso de Engenharia da Computação - CIn – UFPE; E-mail: laoj2@cin.ufpe.br,
[2]Docente/pesquisador do Centro de Informática – CIn – UFPE. E-mail: ensb@cin.ufpe.br .

Sumário: A maioria dos algoritmos de morfologia matemática pode ser resolvida em processadores de propósito geral, tipicamente usando o *Matlab* (MATLAB, 2010) ou o *OpenCV* (OPENCV, 2014). Porém, quando os sistemas possuem restrições de tempo, de memória de eficiência energética, essa abordagem nem sempre é eficiente. Este projeto, portanto, objetiva otimizar a performance dos algoritmos tradicionais de morfologia matemática, fornecendo alta vazão de pixels e pequena ocupação de área da *FPGA*, através de uma implementação em *System Verilog* que possa ser validada e aplicada em contextos reais e práticos em sistemas de tempo real embarcados. A solução utilizando a biblioteca *OpenCV* apresentou até 301% ciclos de *clock* a mais que a solução proposta em FPGA. Adicionalmente, os módulos implementados ocupam apenas 245 elementos lógicos, o que corresponde a menos que 1% da área da *FPGA Cyclone IV*.

Palavras–chave: FPGA; processamento de imagens; sistemas embarcados; morfologia matemática; visão computacional

INTRODUÇÃO

A crescente necessidade de inspeção e análise minuciosas das informações visuais nas mais diversas áreas de conhecimento alavancou o desenvolvimento de sistemas que utilizam processamento de imagens, já que a velocidade com que essas informações chegam ao olho humano e o grande volume de dados são incompatíveis para a capacidade de processamento do cérebro humano.

Nesse contexto, surgiu uma grande quantidade de algoritmos de processamento de imagens. Mas, por causa da constante manipulação de matrizes, esses algoritmos tendem a ter um custo computacional não-linear, como, por exemplo, os algoritmos de morfologia matemática.

A morfologia matemática, desenvolvida em 1960 por Georges Matheron e Jean Serra (SERRA, 1982), descreve um conjunto de técnicas de processamento de imagem baseado na extração de características que são úteis para representar e descrever a forma - geometria e topologia - das entidades desconhecidas de uma imagem a partir da teoria dos conjuntos.

A maioria dos algoritmos de morfologia matemática pode ser resolvida em processadores de propósito geral, tipicamente usando o *Matlab* ou o *OpenCV*. Porém, quando os sistemas possuem restrições de tempo, de memória de eficiência energética, essa abordagem nem sempre é eficiente. Adicionalmente, o *Matlab* e o *OpenCV* não são portáveis de forma simples para as diferentes arquiteturas encontradas em sistemas embarcados.

Este projeto, portanto, objetiva otimizar a performance dos algoritmos tradicionais de morfologia matemática, fornecendo alta vazão de pixels, típica exigência de sistemas de tempo real, através de uma implementação em *SystemVerilog* que possa ser validada e aplicada em contextos reais e práticos.

Existem alguns trabalhos na literatura que realizaram a implementação de operações morfológicas em sistemas embarcados.

O sistema desenvolvido em (TICKLE, et al., 2013) implementa as operações de morfologia matemática a partir de blocos *DSP Builds* da ferramenta de desenvolvimento *Simulink*, do *Matlab*. Apesar da simplicidade de implementação, as operações só podem ser executadas na *FPGA* no processador *Nios*. A análise temporal da solução é feita somente no *Simulink*.

Na técnica apresentada em (HENTATI et al., 2014), as operações morfológicas de erosão e dilatação foram desenvolvidas utilizando a técnica *DPR (Dynamic and Partial Reconfiguration)*. Porém, a *Xilinx* era, até a publicação deste resumo, a única fornecedora de *FPGA* que oferecia suporte a *DPR*. Em (GENOVESE, et. al, 2013), as unidades de morfologia matemática implementadas em *FPGA* apresentam alto valor de frames por segundo. Em contrapartida, há uma grande ocupação de área da *FPGA*. Na proposta apresentada em (BAUMANN, et. al, 2005), foi implementada uma biblioteca em *VHDL* de operações de morfologia matemática. Porém, as imagens de teste têm dimensões 160x120 e os módulos ocupam boa parte da área da *FPGA*.

MATERIAIS E MÉTODOS

Em *FPGA*, imagens geralmente são um conjunto de sinais seriais e discretos que são obtidos a partir de uma câmera digital. Portanto, na abordagem em *FPGA*, há a movimentação da imagem (e não do elemento estruturante, como na abordagem em software) de modo que, pela própria natureza da fonte de dados (uma câmera), o elemento estruturante estará centrado em um novo pixel a cada ciclo de *clock* com uma estrutura aproximadamente linear.

Nessa perspectiva, seja $\mathcal{S}$ um elemento estruturante de dimensões NxN e $\mathcal{A}$ uma imagem binária de dimensões PxQ. Para que a abordagem em *FPGA* funcione corretamente, é necessário armazenar $ceiling\left(\frac{N}{2}\right) * Q$ pixels da imagem de entrada antes de realizar qualquer operação, onde $ceiling(x)$ é a função teto, para que o elemento estruturante esteja posicionado corretamente (com o seu pixel central sobreposto ao primeiro pixel da imagem) na imagem. Portanto, a vazão de pixels na saída terá uma latência inicial de $ceiling\left(\frac{N}{2}\right) * Q$ pixels. Em seguida, os pixels de saída serão fornecidos um a um sem interrupções.

Adicionalmente, como as operações morfológicas envolvem todos os pixel na fronteira delimitada pela forma do elemento estruturante, é necessário que o sistema também tenha acesso a linhas anteriores e posteriores da imagem analisada que façam parte da fronteira.

Portanto, neste trabalho foram desenvolvidos três módulos principais: um buffer de linha (baseado em registradores de deslocamento), um módulo de erosão e um módulo de dilatação (os módulos de abertura e fechamento foram implementados a partir da associação em cascata dos módulos supracitados). A arquitetura do módulo de dilatação é ilustrada na figura 1.

Todos os módulos supracitados foram desenvolvidos na linguagem *System Verilog*, foram sintetizados com o software *Quartus II Web Edition*, versão 13.1 e foram testados na placa de prototipação *DE2i-150*, da fabricante *Terasic*, que possui uma *FPGA Cyclone IV GX* e um processador *Intel Atom* (N2600). A câmera digital *TRDB-D5M*, da fabricante *Terasic*, foi acoplada à placa de prototipação Os módulos de dilatação e erosão possuem 280 linhas de código, cada. A fim de validar os módulos desenvolvidos em hardware, foram desenvolvidos e executados dois *testbenches* com o programa *ModelSim Student Edition*.

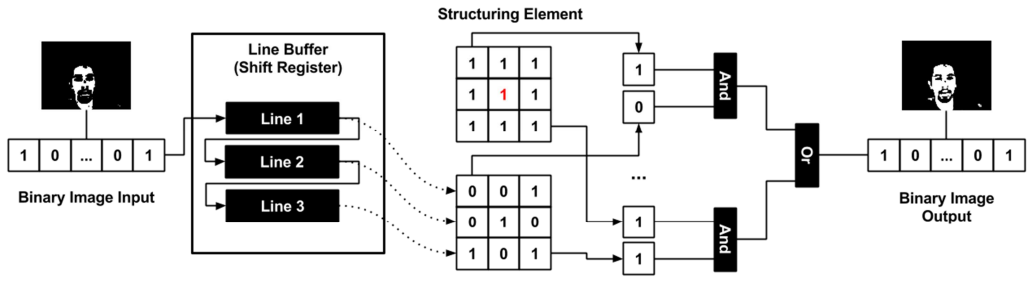

Figura 1 – Arquitetura do módulo da operação morfológica Dilatação

A medição das soluções em software foi realizada utilizando as funções *dilate* e *erode* do *OpenCV* versão 2.4, no processador *Intel Atom (N2600)*, no sistema operacional *Windows* 7, a partir das funções da biblioteca *time.h*, que é padrão de C. Para realizar a medição de tempo de execução das soluções em *FPGA*, foi utilizada a fórmula descrita a seguir. Seja $\mathcal{A}$ uma imagem de dimensões PxQ, seja $\mathcal{S}$ um elemento estruturante de dimensões NxN, seja $L = ceiling\left(\frac{N}{2}\right) * Q$ a latência inicial em pulsos de *clock* e seja f a frequência de operação da *FPGA*, o tempo total para execução das operações é dado por: $T_{simples} = (L + N * M) * f$.

RESULTADOS

As tabelas 1 e 2 mostram os resultados dos testes de desempenho realizados em ambas abordagens, software e hardware, considerando um cenário com 30 imagens de 8MP.

Tabela 1 – Resultados das simulações, em ciclos de *clock*, para a erosão e o fechamento.

Arquitetura	Erosão (Média)	Erosão (Desvio Padrão)	Fechamento (Média)	Fechamento (Desvio Padrão)
Atom (OpenCV)	75.200.000	1.118.400	153.883.000	2.447.120
FPGA	24.883.200	0	49.766.400	0

Tabela 2 – Resultados das simulações, em tempo de execução, para a fechamento e erosão.

Arquitetura	Erosão (Média)	Erosão (Desvio Padrão)	Fechamento (Média)	Fechamento (Desvio Padrão)
Atom(OpenCV)	0.047s	0.000699s	0.096176s	0.001529s
FPGA (50 MHz)	0.497664s	0s	0.497664s	0s
FPGA (1.5GHz)	0.016589s	0s	0.016589s	0s

DISCUSSÃO

Os resultados indicam que as funções do *OpenCV* utilizam, aproximadamente, 301% a mais de ciclos para realizar as operações de morfologia matemática, do que a solução desenvolvida em *FPGA*.

Os resultados da análise temporal indicam que a vantagem da solução, em tempo de execução, em relação à aplicação desenvolvida em *OpenCV* está em função da frequência de *clock*. Portanto, caso a aplicação seja executada em uma *FPGA* que atinja uma frequência próxima à frequência do processador *Atom*, como a *FPGA SPD60*, da fabricante *Achronix* (ACHRONIX, 2015), que consegue atingir até 1.5GHz, a melhoria obtida em relação a solução em *OpenCV* seria de até 291%.

Adicionalmente, cada módulo, individualmente, utiliza 245 elementos lógicos, o que corresponde a menos que 1% dos elementos disponibilizados pela *Cyclone IV*.

CONCLUSÕES

Operações de morfologia matemática são amplamente utilizadas em aplicações de visão computacional e reconhecimento de padrões, áreas que estão em constante crescimento e que, ultimamente, estão sendo cada vez mais requeridas em aplicações com sistemas embarcados. Então, a utilização da solução desenvolvida nesse projeto, que é capaz de prover uma melhora no desempenho de até 291% em relação a implementações clássicas em software, que possui baixo custo, ocupa pequena área e consuma menos energia, características de sistemas que utilizam *FPGA*, é altamente viável.

A solução desenvolvida conseguiu ser bastante eficiente, com uma vazão de um pixel por pulso de *clock* (mesma vazão da câmera *TRDB-D5M* utilizada), o que permite sua aplicação, inclusive, em sistemas de tempo real, que possuem restrições de tempo e memória.

AGRADECIMENTOS

Gostaria de agradecer a Deus, à minha orientadora Edna Barros pela oportunidade de crescimento profissional e intelectual, ao PIBIC/CNPq e aos meus familiares.

REFERÊNCIAS

ACHRONIX, Speedster FPGA Family. Disponível em <http://www.inst.eecs.berkeley.edu/~cs29459/fa10/resources/achronix/Speedster_Datasheet_DS001.pdf>. Acesso em 20 de julho de 2015.

BAUMANN, Damien; TINEMBART, Jacques. Designing Mathematical Morphology Algorithms on FPGAs: An Application to Image Processing. In: Computer Analysis of Images and Patterns. Springer Berlin Heidelberg, 2005. p. 562-569.

GENOVESE, Mariangela; NAPOLI, Ettore. FPGA-based architecture for real time segmentation and Denoising of HD video. Journal of Real-Time Image Processing, v. 8, n. 4, p. 389-401, 2013.

HENTATI, R.; HENTATI, M.; AOUDNI, Y.; ABID, M., "The implementation of basic morphological operations on FPGA using partial reconfiguration," in *Image Processing, Applications and Systems Conference (IPAS), 2014*, vol., no., pp.1-5, 5-7 Nov. 2014 doi: 10.1109/IPAS.2014.7043260

MATLAB version 7.10.0. Natick, Massachusetts: The MathWorks Inc., 2010.

OPENCV, OpenCV Documentation. Disponível em <http://docs.opencv.org/>, Acesso em 18 de dezembro de 2014.

SERRA, J., Image Analysis and Mathematical Morphology, Academic Press, 1982.

TICKLE, Andrew J. et al. Development and simulation of soft morphological operators for a field programmable gate array. Journal of Electronic Imaging, v. 22, n. 2, p. 023034-023034, 2013.