Estudo da viabilidade de implementação de um sistema de localização e reconhecimento de objetos com uso de RNAs implementadas em FPGA

Rolf F. Molz[1], Paulo M. Engel[1], Fernando G. Moraes[2], Lionel Torres[3], Michel Robert[3]

[1]Inst. de Informática – UFRGS - Caixa Postal 15064 - POA - RS. CEP: 91501-970.
[2]Fac. de Informática – PUC/RS - Av. Ipiranga, 6681. CEP: 90619-900.
[3]LIRMM -Université Montpellier II - 161, rue Ada 34392 MONTPELLIER Cedex 5 France
E-mails: {rolf, engel}@inf.ufrgs.br , moraes@inf.pucrs.br , {torres, robert}@lirmm.fr

Abstract: Pattern localization and recognition are CPU time intensive, being normally implemented in software. Custom implementations in hardware allow real-time processing. In practice, in ASIC or FPGA implementations, the digitization process introduces errors that should be taken into account. This paper presents initially the state-of-the-art in this field, analyzing the performance and implementation of each work. After, we present our implementation in a prototype platform, exploring the inherent parallelism of the hardware implementation. We analyze our implementation in terms of required hardware for the ANN (artificial neural network) and the maximum frequency of operation. Also the same model is implemented in software, and analysis in terms of image quality is presented. A quantitative analysis is finally presented between the performance obtained by the FPGA and the possible one obtained by a DSP.

Keywords: Pattern localization and recognition, hardware implementations, Artificial Neural Networks.

1. Introdução

Redes Neurais Artificiais (RNAs) têm sido empregadas em vários campos da ciência. Estas são utilizadas para solucionar uma grande variedade de problemas que são de difícil solução por métodos tradicionais. As RNAs são especialmente úteis para problemas onde a característica de generalização se faz necessária. Exemplos de aplicações se estendem das áreas comercias às áreas científicas.

Embora RNAs sejam muito implementadas em software, versões implementadas em hardware estão ganhando importância. Versões implementadas em software possuem a vantagem de serem facilmente implementadas, mas não utilizam a característica de massivo paralelismo existente nas RNAs. As versões implementadas em hardware são geralmente mais difíceis e consomem mais tempo para as implementações. Uma relação não exaustiva de várias implementações de RNAs em hardware pode ser encontrada em [23].
O campo de processamento de imagens é um exemplo de área de utilização de RNAs e de implementações em hardware devido a quantidade de dados a serem processados em um curto intervalo de tempo e a possibilidade de processá-los de uma forma paralela. Para tanto, estamos propondo este artigo que demonstra a possibilidade de se implementar RNAs em FPGAs operando em paralelo.
Este artigo também apresenta um levantamento bibliográfico de trabalhos realizados na tarefa de localização e classificação de objetos visando embasar um projeto para a concepção de um sistema para a localização de retângulos e posterior classificação dos mesmos. Sistema este que deverá ser implementado em FPGA ou em FPGA/DSP. Para tanto, a Seção 2 apresenta alguns trabalhos nesta área. A Seção 3 apresenta uma implementação neural em FPGA para a solução de um problema na área de classificação de padrões bem como os seus resultados. Por fim, na Seção 4 apresentam-se as conclusões.

2. Estado da Arte

Nesta seção, apresentaremos alguns trabalhos realizados nos últimos anos com o esforço de realizar tarefas complexas na área de tratamento de imagens. Esta lista não é exaustiva, mas aborda bem a necessidade de um sistema compacto. Abaixo há 19 trabalhos com uma breve descrição para cada um deles.
O trabalho apresentado por [4] propõe o uso de redes neurais para a localização e classificação de marcas para a navegação em robôs móveis situados em ambientes fechados. A procura por marcas ocorre somente em regiões possíveis, as quais são selecionadas por um "screener". O ponto negativo é que as marcas devem possuir o mesmo ângulo e estarem em posições pré-estabelecidas da imagem. A implementação é realizada em software e possui uma velocidade de 2,5 quadros por segundo para imagens de 256x256 pixels. O trabalho também apresenta uma comparação de tempos de execução quando implementados em plataformas diferentes. As plataformas utilizadas foram: Sparc20, Pentium, Intel486, PAPRICA. Esta última realiza o processamento paralelo sendo a plataforma mais rápida.
Em [5] há somente a segmentação em hardware. Este trabalho apresenta uma variação na etapa de filtragem para se adequar as restrições de hardware. O processo foi implementado em um ASIC com tecnologia de 0,8µm, dois níveis de metal. O desempenho anunciado é de 30 quadros por segundo para imagens VGA.
O trabalho apresentado por [6] propõe uma estratégia para implementar uma convolução em um dispositivo reconfigurável (FPGA). O desempenho obtido é de 14 vezes superior ao desempenho de um software aplicado em um DSP C-40. O trabalho mostra um estudo de ocupação de FPGAs e a implementação é realizada em dois FPGAs Xilinx XC4013 (consumo total de 962 CLBs).

A proposta de [7] aborda a extração de pontos dominantes e contornos de objetos. Esta implementação compreende um ASIC (tecnologia 0,8μm) e um FPGA para o controle de dados e interface. A extração de pontos é por meio da utilização de filtros FIR (Finite Impulse Reponse).

O trabalho apresentado em [8] realiza a detecção de objetos após a supressão do background. Para tal, supõe-se ter o background e atualizá-lo caso a implementação seja out-door. Esta detecção é realizada com o uso da esqueletização estatística. A etapa de reconhecimento é através da comparação do objeto esqueletizado com os modelos previamente armazenados. Por fim, este trabalho também contempla a perseguição ao modelo localizado. A implementação foi idealizada completamente em software, onde a plataforma utilizada foi um Pentium II 300MHz. O desempenho obtido é em média 3 quadros por segundo. Este desempenho é para toda a tarefa, contudo, há um percentual de 75% deste tempo destinado para a operação de extração de característica (esqueletização) e para o reconhecimento.

Em [9] temos um sistema instalado dentro de um veículo para a detecção de sinais de trânsito ou de pedestres. O algoritmo de detecção é baseado no cálculo da "Distance Transform" e de um algoritmo de busca hierárquica. O reconhecimento é realizado por um pré treinamento de um clusterizador (tipo K-means). O trabalho contempla como uma última etapa uma rede neural do tipo RBF (Radial Basis Function) para verificação e conseqüente diminuição do número de respostas falsas positivas. As imagens utilizadas para teste são imagens em 256 tons de cinza com um tamanho de 360x288 pixels. Esta implementação foi idealizada em um dual-Pentium MMX 450MHz e possui um desempenho entre 10 a 15 quadros por segundo para o reconhecimento de placas. Para o reconhecimento de pedestres o desempenho cai para entre 1 a 5 quadros por segundo.

Em [10] há a proposição de um neurochip analógico (ASIC, tecnologia 1,2μm) para o processamento de imagem. O trabalho apresenta a construção de três diferentes tipos de redes neurais sem qualquer modificação estrutural do circuito. O autor apresenta como resultado a simulação do problema XOR. A taxa de saída do neurochip é de $12x10^6$ sinapses/segundo.

A proposta de [11] é uma metodologia para a localização de placas de perigo instaladas em containers no porto de Esbjerg, Dinamarca. A etapa de localização é realizada por meio da seleção de todos os objetos que possuem formas semelhantes às placas de perigo. A classificação é realizada por uma rede neural baseada em RAM (Random Acess Memory). A implementação foi realizada em software e testada sobre uma máquina 486 - 66 MHz. O tempo para localização e classificação é em torno de 10 segundos por imagem.

O trabalho [12] apresenta uma metodologia para a geração de ASICs para o processamento de imagens em tempo real. Apresenta, ao final do artigo, uma análise do resultado de algumas técnicas de processamento de imagens com o tempo de resposta do ASIC, bem como, a tecnologia de produção deste ASIC.

Em [13], temos o reconhecimento de objetos com um certo grau de oclusão. A detecção é realizada por meio de uma estrutura piramidal, onde cada estágio desta pirâmide é formado por transformações na imagem de entrada. Um primeiro nível é formado pela diferença entre gaussianas e outro nível é formado pela interpolação. A partir desta pirâmide calculam-se pontos máximos e mínimos (chamados de SIFT). O reconhecimento é realizado por meio da comparação entre pontos característicos (SIFT) entre a imagem de entrada e os objetos previamente armazenados. As imagens

testes são de 384x512 pixels. A implementação foi realizada a nível de software e foi executada em uma Sun Sparc 10. O tempo de todo o tratamento e reconhecimento é em torno de 1,5 segundos. O autor comenta que 0,9 segundos é destinado para a geração da estrutura piramidal e 0,6 segundos para a etapa de reconhecimento.

A proposta de [14] é realizar a extração de linhas e seus pontos extremos por meio da Transformada de Hough em uma placa contendo FPGAs em paralelo. O tempo obtido para extração dos parâmetros de linhas em imagens de 256x256 pixels é de 48ms.

Em [15] temos um sistema para tratamento de imagens em tempo real. É implementada uma placa para o tratamento de imagens, conectada a um barramento PCI. Como exemplo de operação, ele demonstra uma aplicação para o reconhecimento de placas de carro na entrada de estacionamentos. Os carros estão parados e todos possuem aproximadamente a mesma distância até a câmera, com isto não há problemas de escalonamento. A etapa de localização e detecção da placa é realizada por tratamentos na imagem e é realizada em hardware (FPGA - 10K100 Altera). O reconhecimento da placa é realizado em software (PC) por meio de uma rede neural feedforward de cinco camadas (600 x 241 x 42 x 100 x 34). O tempo para localização e reconhecimento da placa é de 30 quadros por segundo.

O trabalho de [16] apresenta a concepção de um projeto para a classificação de padrões. A técnica é implementada em um ASIC (tecnologia 1,2µm), e em FPGA. O desempenho obtido para a implementação ASIC é de um ponto classificado em menos de 100ns.

Em [17], o trabalho possui a finalidade de localização de objetos com um certo grau de oclusão. A localização e conseqüente reconhecimento se faz por meio da procura da menor distância de Hausdorff. A implementação é em software e foi testada sobre Sun Sparc Server 1000 - 8 processadores, onde, para o exemplo apresentado no artigo, o processo leva aproximadamente 7 minutos para a localização.

O trabalho de [18] apresenta a nova versão do chip PAPRICA, que é um ASIC para processamento geral. Este foi desenvolvido em tecnologia 0,8µm CMOS (um nível de polisilício e dois de metal). É apresentada uma aplicação deste chip onde são utilizados 4 circuitos integrados para realizar a tarefa de reconhecimento do montante expresso em cheques. A tarefa é por meio de uso de redes neuro-fuzzy, mas o trabalho não contempla a descrição desta parte, somente os resultados obtidos em comparação com PC e Sparc 10.

O trabalho de [19] também aborda a detecção e reconhecimento de objetos. Porém, este trabalho contempla o reconhecimento de objetos com algum grau de oclusão. O algoritmo contempla uma etapa de pré-processamento com: segmentação da imagem, traçado do contorno, suavizamento do contorno e por fim uma aproximação para polígonos. A etapa de reconhecimento é realizado por mais 6 sub-etapas, onde, em uma delas, há a utilização de "m" (onde m é o número de objetos diferentes na base de conhecimento) redes neurais do tipo MLP (Multi-Layer Perceptron) . Cada MLP possui uma camada de entrada com 2 neurônios, uma camada escondida com 30 neurônios, e uma camada de saída com X neurônios, onde X é o número de características a serem consideradas no objeto. As imagens utilizadas para teste são imagens em 256 tons de cinza com um tamanho de 640x512 pixels. O desempenho obtido é de uma imagem em menos de 3 minutos. Toda a implementação foi realizada em software e o maior custo de tempo no algoritmo é a etapa de pré-processamento.

O trabalho proposto por [20] explora a implementação de um algoritmo para esqueletização de uma imagem binária de 240x240 pixels. É realizada a comparação qualitativa entre a implementação em software e em hardware.
Em [21] é apresentado uma implementação de um filtro para a detecção de bordas pelo método de Canny-Deriche. A implementação é realizada em um ASIC com um processo de 1,0µm. O desempenho é de 30ns por pixel.
O trabalho de [22] aborda a detecção e reconhecimento de objetos com um certo grau de oclusão. A etapa de pré-processamento é responsável pela detecção de pontos principais. Após há a geração de modelos AR (Auto Regressive) ortogonais e o reconhecimento é realizado pela análise da distância Euclidiana entre o modelo gerado e os modelos AR pertencentes aos padrões pré-armazenados. Por fim, a Tabela 1 apresenta o resumo do estado-da-arte dissertado acima.

Tabela 1. Resumo do estado da arte

Ref.	**Função**	**Metodologia**	**Implementação**	**Desempenho**
[9]	Det. e rec. de sinais de trânsito ou pedestres	Det. baseada em Distance Transforms e árvore hierárquica. Rec. por RBF	Dual-Pentium II 450MHz MMX	10 a 15 frames por seg (360 x 288 pixels). Pedestres 1 a 5 frames por seg.
[19]	Det. e rec. de objetos com oclusão	Vários estágios de pré-processamento Clas. por MLP	Implementado em software	1 frame em menos de 3 minutos.
[22]	Rec. de objetos parcialmente oclusos	Det. de pontos característicos - Rec. por meio da distância Euclidiana	Implementado em software	
[8]	Det. e rec. de objetos	Det. por extração do background e por esqueletização Rec. por comparação em banco de dados.	Implementado em software. Pentium II - 300MHz	Média de 3 frames por segundo
[13]	Det. e rec. de objetos com oclusão	Filtros Gaussianos com pré-processamento. Geração de pontos característicos. Rec. por combinação de SIFTs	Sun Sparc 10	1 frame (384 x 512 pixels) em 2 seg.
[17]	Localização de objetos	Hausdorff Distance	8 processadores Sun SPARC Server 1000	Exemplo box = +-7min.
[11]	Det. e clas. de placas de perigo	Det. e clas. por pré-processamento e por uma rede neural do tipo RAM	Pentium 486-66MHz	10 segundos (det. e clas.)
[15]	Det. e rec. de placas de carro parado	Pré-processamento para det. e rec. por MLP	Pré-processamento em FLEX10K100 Rec. em PC	30 frames por segundo
[10]	Somente arquitetura neural.		ASIC - 1,2µm double-poly CMOS analog	Taxa de saída : $12x10^6$ sinapse/seg.
[4]	Det. e rec. de marcas	Uso de RNA Rec. em MLP¨.	Comparação: PC, Sparc20, PAPRICA.	400ms por imagem de 256x256 (2,5 frames /seg.)
[18]	Processa mento de imagens genérico		Uso de 4 chips PAPRICA - 3 (ASIC 0,8µm CMOS)	
[12]	ASICS para processamento de imagens		Diversos ASICs	Aborda vários tipos de algoritmos

[14]	Extração de linhas	Transformada de Hough	Placa com FPGAs	
[20]	Esqueletização de imagens	Comparação entre software e hardware		
[7]	Extração de pontos dominantes e contornos de objetos		ASIC (0,8µm) e FPGA	Não aborda o desempenho claramente
[5]	Segmentação em tempo-real		ASIC 0,8µm double-metal	30 frames/segundo
[6]	Convolução em FPGA		FPGA (2 x XC4013)	14 x mais rápido que o DSP C-40
[16]	Clas. de padrões		ASIC 1,2µm e também FPGA	ASIC = menos de 100ns por ponto
[21]	Det. de bordas		ASIC 1,0µm	30ns por pixel

Onde : ref. : referência
Impl. : forma de implementação
rec. : reconhecimento
det. : detecção

2.1 Análise

Com base neste estudo, pode-se salientar nove trabalhos que são próximos do nosso objetivo. São eles: [4], [8], [9], [11], [13], [15], [17], [19] e [22]. Dentre estes nove, cito os trabalhos [4], [9], [11] e [15] como sendo os trabalhos com mais afinidade ao nosso objetivo.
Em [4] há a necessidade de que as marcas estejam em lugares pré-definidos, além de ser processado em software. É feito uma análise com o uso do PAPRICA, mas este é conectado a um host. Em [15], o carro necessita estar a uma distância fixa e conhecida. Esta fixação quanto a distância permite aplicar uma regra de proporção que facilita uma rápida detecção dos números da placa. O sistema foi desenvolvido como sendo parte em software, instalado em PC, e parte em hardware, FPGA. Em [11] o sistema foi implementado totalmente em software. Por último, em [9] o sistema também foi implementado totalmente em software, portanto não se beneficia de qualquer grau de paralelismo que possa ser aplicado a todas as aplicações de processamento de imagens. Estes quatro trabalhos não comentam a realização de suas tarefas com algum grau de oclusão nos objetos a serem localizados e ou classificados.
O sistema proposto visa obter um sistema portátil, de baixo peso, de baixo consumo. Além disto, o sistema utilizará operações em paralelo, o que conduz a um aumento do desempenho do processamento. Observa-se que o sistema será capaz de realizar todas as operações necessárias para a localização de formas retangulares aprendidas previamente. Certamente num futuro próximo podemos ter este sistema como um system-on-chip, ou utilizar microprocessadores DSP com parte reconfigurável.
Este sistema por ser ligado a um circuito CIS (CMOS Image Sensor), permitindo com isto uma utilização mais compacta. Além disto, para se aumentar a capacidade de processamento pode-se implementar a fase de aprendizado da RNA em DSP de modo a garantir o desempenho e com isto permitir o treinamento on-line do circuito. Esta implementação garante também a compacticidade e baixo consumo.
Por fim, o hardware para a realização deste sistema pode vir a ser o esqueleto para uma placa de prototipação rápida onde pode ser desenvolvido diversos algoritmos para o

processamento de imagens em VHDL e estes servirem como uma base de biblioteca para futuros trabalhos em hardware ou hardware/software.

3. Implementação e Teste

Esta seção descreve a implementação de uma RNA em FPGA que será posteriormente validada. A RNA foi implementada na placa de prototipação rápida LIRMM [3].
Foi utilizada para esta implementação uma rede neural do tipo MLP com aprendizado por backpropagation padrão, realizado de modo off-line. A arquitetura da rede possui três camadas, sendo: 2 neurônios na camada de entrada, 3 neurônios na camada escondida e 1 neurônio na camada de saída.
Como comentado, o aprendizado foi realizado de modo off-line no DSP existente na placa LIRMM. O DSP também realiza a interface entre a placa de prototipação e o computador PC.
As codificações dos pesos sinápticos, dos padrões de entrada e da função de ativação são descritas em [2].
A Figura 1 apresenta o diagrama de blocos da RNA implementada em FPGA. Nesta figura, a camada escondida (neurônios conectados às entradas) e a camada de saída são mostradas. Foi adotado uma implementação síncrona, onde, em cada camada, há um shift-register como unidade de tempo para a computação do resultado de saída. O número de bits no shift-register foi obtido pelo atraso existente no cálculo do valor de saída do neurônio. Este valor de atraso foi obtido por uma ferramenta de análise de *timing*. Esta configuração garante uma rede estável, pois são garantidos os valores de saída de cada um dos neurônios.

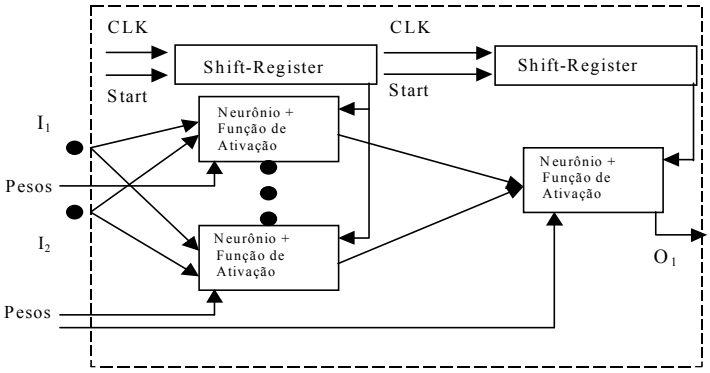
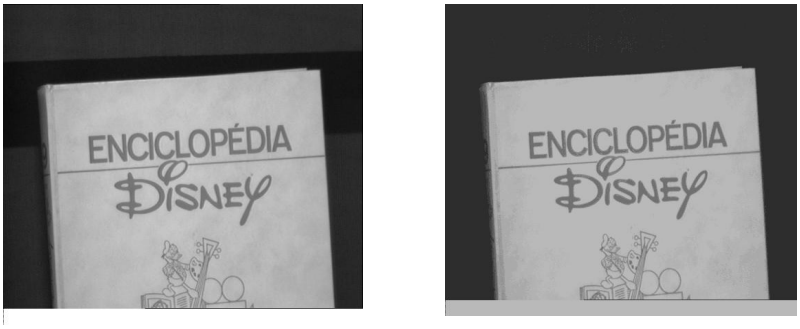


Figura 1. Diagrama de Blocos

3.1 Validação

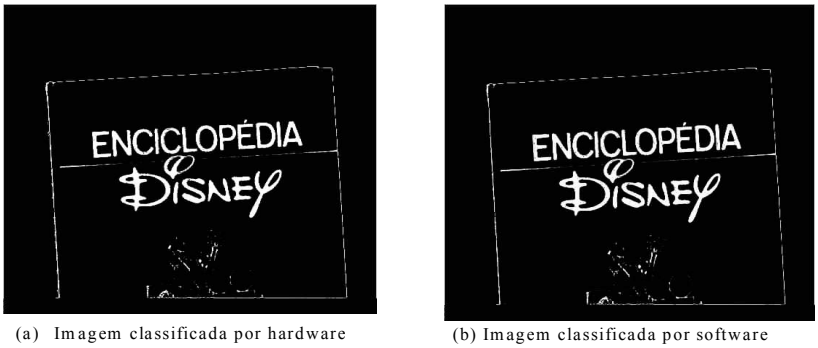
Uma aplicação típica na área de processamento de imagens foi utilizada como validação da arquitetura neural. A tarefa consiste em localizar letras de uma dada imagem. A Figura 2.a mostra a imagem original. Esta imagem foi obtida por uma câmera CCD com uma resolução de 512x512 pixels e 256 tons de cinza.

Os padrões de entrada são compostos pelo tom de cinza do pixel e pela média dos pixels vizinhos. Esta média é calculada sobre uma janela móvel de tamanho 3x3, centrada no pixel em questão.
Como explicado em [2], a codificação de entrada permite somente 16 valores diferentes (1 bit para sinal e 3 bits para o valor). Como a imagem de entrada possui 256 tons de cinza, esta necessita ser pré-processada para reduzir a quantidade de tons de cinza de 256 para 16. Diferentes métodos de clusterização podem ser utilizados para o pré-processamento, como exemplo, quantização uniforme, crescimento de regiões ou Mapas de Kohonen. Nós temos implementado os Mapas de Kohonen para realizar a clusterização da imagem mostrada na Figura 2.a. Na Figura 2.b, temos a imagem clusterizada em 16 tons de cinza.
A imagem clusterizada em 16 tons de cinza é a imagem a ser tratada pela RNA. A fase de aprendizado é realizada pelo DSP, ou seja, por *software*. Após a fase de aprendizado, os pesos sinápticos que foram calculados são adequados para a codificação do hardware e são enviados para o FPGA. A imagem é processada durante a fase de propagação, com todos os neurônios, em cada camada, operando em paralelo. O resultado é mostrado na Figura 3.a. Para comparar o resultado obtido, Figura 3.a, a fase de propagação também foi realizado no DSP, Figura 3.b.



(a) – Imagem original (b) – Imagem clusterizada

Figura 2. Imagem original e imagem clusterizada



(a) Imagem classificada por hardware (b) Imagem classificada por software

Figura 3. Imagens classificadas por hardware e por software

A Figura 4 apresenta o histograma resultante da diferença entre a imagem classificada por hardware e a imagem classificada por software (Figura 3.a e Figura 3.b). Como pode ser observado, a imagem processada por hardware é muito próxima da imagem processada por software, pois a diferença entre quase todos os pixels é zero.

Figura 4. Histograma comparando imagens classificadas

4. Conclusões

Este artigo apresentou um estudo bibliográfico sobre trabalhos realizados visando a solução de problemas na área de tratamento de imagens. Tal estudo é fundamental para embasar um próximo trabalho explorando a implementação de algoritmos para o tratamento de imagens em FPGA. Além disto, foi demonstrada a implementação de uma RNA em FPGA por meio da placa de prototipação rápida LIRMM. Uma RNA completamente paralela foi implementada, pois o nosso objetivo é explorar o paralelismo de redes neurais. Esta implementação utilizou 558 CLBs dos 576 CLBs disponíveis no FPGA XC4013. A frequência máxima de operação, estimada pela ferramenta de análise de timing, foi de 3.5Mhz.

A descrição da rede neural é modular, sendo por isto fácil de aumentar ou diminuir o número de neurônios. Dois shift-registers foram utilizados para a parte de controle da rede neural, tendo um atraso de 10 períodos de clock cada um. Como a freqüência do FPGA é de 20MHz, 1 μs é necessário para propagar os padrões de entrada. Considerando que a frequência do DSP igual a 50 MHz, 1 μs resultaria em 50 ciclos de clock. Nós não temos uma análise comparativa do desempenho entre as implementações em hardware e em software (DSP), mas estes dados indicam que o desempenho obtido em hardware é melhor do que o desempenho obtido por software, pois o DSP não é capaz de executar todas as tarefas da fase de propagação em somente 50 ciclos de clock.

5. Agradecimentos

O autor Rolf F.Molz agradece o suporte da CAPES Capes/Cofecub projeto 245/98-BR.
O autor Fernando G.Moraes agradece o suporte do CNPq projeto 522939/96-1-BR.

6. Referências Bibliográficas

[1] MOLZ, R. F; ENGEL, P.M. A New Proposal for Implementation of Competitive Neural Networks in Analog Hardware. Vth Brazilian Symp. on N.N., BH, MG, pp.186-191. Dec. 1998.

[2] MOLZ, R. F.; ENGEL, P.M.; MORAES, F.G. Uso de um ambiente codesign para a implementação de redes neurais. IV CBRN, p.13-18, São José dos Campos, Brasil, July 1999.

[3] TORRES,L.; PILLEMENT,S.; ROBERT,M. LIRMM: Prototyping Platform for Hardware/ Software Codesign. Journal of Current Issue in Electr. Modeling, vol. 9, p. 49-59, March 1997.

[4] ADORNI, G.; GORI, M.; MORDONINI, M. Just-in-time Landmarks Recognition. Real-Time Imaging, vol. 5, pp. 95-107, 1999.

[5] ALZAHRANI, F.M.; CHEN, T. A Real-Time Edge Detector: Algorithm and VLSI Architecture, Real-Time Imaging, vol. 3, pp. 363-378, 1997.

[6] BOSI, B.; BOIS, G.; SAVARIA, Y. Reconfigurable Pipelined 2-D Convolvers for Fast Digital Signal Processing. IEEE Trans. On VLSI Systems, vol. 7, n°. 3, Sept. 1999.

[7] DALLAIRE, S.; TREMBLAY, M.; POUSSART, D. VLSI Architectures for the Embedded Extraction of Dominant Points on Object Contours. Proceedings of the Computer Architectures for Machine Perception (CAMP-97), Como, Italy, October 20-22, 1997.

[8] FORESTI, G.L. Object Recognition and Tracking for Remote Video Surveillance. IEEE Trans. On Circuits and Systems for Video Technology, vol. 9, n° 7, pp. 1045-1062, Oct. 1999.

[9] GAVRILA, D.M.; PHILOMIN, V. Real-Time Object Detection for "Smart" Vehicules. International Conference on Computer Vision. Vol. 1. Corfu, Greece, 20-25 Sept. 1999.

[10] HAN, G.; SÁNCHEZ-Sinencio, Edgar. A Flexible and Expendable Neuroimage Processor Architecture. IEEE Trans. on Circuits and Systems-I, vol.46, n°9, pp.1055-1063, Sept. 1999.

[11] JORGENSEN, T.M.; CHRISTENSEN, S.S.; ANDERSEN, A.W. Detecting danger labels with RAM-based neural networks. Pattern Recognition Letters, vol. 17, pp. 399-412, 1996.

[12] KRALJIC, I.C.; QUENOT, G.M.; ZAVIDOVIQUE, B. Investigating real-time validation of Real-Time Image Processing ASICs. Proceedings of the Computer Architectures for Machine Perception, Como, Italy, October 20-22, 1997.

[13] LOWE, David G. Object Recognition from Local Scale-Invariant Features. Proc. Of the International Conference on Conputer Vision, Corfu, Greece, 20 - 25 September, 1999.

[14] MAHMOUD, M.; NAKANISHI, M.; OGURA, T. Hough transform implementation on a reconfigurable highly parallel architecture. Proceedings of the Computer Architectures for Machine Perception, Como, Italy, October 20-22, 1997.

[15] MINGO, Ferran L. Configurable Computing for Real-Time Vision. Tése de Doutorado, Universitat Autônoma de Barcelona - UAB, pp. 180, Bellaterra, september, 1998.

[16] ROBERT, M; GORRIA, P.; MITÉRAN, J.; TURGIS, S. Architectures for a Real Time Classification Processor. Conference on Custom Integrated Circuits, 1994.

[17] RUCKLIDGE, William. Efficiently Locating Objects using the Hausdorff Dsitance. International Journal of Computer Vision vol. 24, n° 3, pp. 251-270, 1997.

[18] SANSOÉ, C.; GREGORETTI, F. A Neuro-Fuzzy Real-Time Image Processing System. Proceedings of the 25th Euromicro Conference, Milan, Italy, 8 - 10 Sept. 1999.

[19] SIM, H.C.; NG. G.S. Recognition of Partially Ocluded Objects with Back-Propagation Neural Network. Inter. Journal of Pattern Rec. and Art. Intel., vol. 12, n°. 5, pp. 645-660, 1998.

[20] SUDHA, N.; NANDI, S. A Parallel Skeletonization Algorithm and its VLSI Architecture. Proc. of the V Inter. Conf. on High Performance Computing, Madras, India, 17-20 Dec. 1998.

[21] TORRES, L.; BOURENNANE, E.; ROBERT, M.; PAINDAVOINE, M. A Recursive Digital Filter Implementation for Noisy and Blurred Images. Real Time Imaging Journal, vol. 4, n° 3, pp. 181-191, June, 1998.

[22] TSANG, K.M.; TO, F.W. Recognition of Partially Occluded Objects using an Orthogonal Complex AR Model Approach. Inter. Journal of Pattern Recognition and Artificial Intelligence, vol. 13, n°. 1, pp. 85-107, 1999.

[23] LINDSEY Clark S., LINDBLAD Thomas. Review of hardware neural networks: a user's perspective 1999. http://msia02.msi.se/~lindsey/elba2html/elba2html.html.