IDENTIFICAÇÃO DE PONTOS DE APOIO PRÉ-SINALIZADOS COM O USO DE REDES NEURAIS ARTIFICIAIS E CORRELAÇÃO

Identifying Pre-signalized Points by the Use of Artificial Networks and Correlation

ROMUALDO WANDRESEN[1]
EDSON APARECIDO MITISHITA[2]
JOSÉ BITTENCOURT DE ANDRADE[3]

[1,2,3] Universidade Federal do Paraná
Departamento de Geomática
Setor de Ciências da Terra
Curso de Pós-Graduação em Ciências Geodésicas, Curitiba-PR.
Caixa Postal 19011 – Centro Politécnico – Jardim das Américas

[1]Pontifícia Universidade Católica do Paraná
Setor de Ciências Exatas e de Tecnologia
Rua Imaculada Conceição , 1155 – Prado Velho – Curitiba PR
[1] wandrese@rla01.pucpr.br; [2] mitishit@geoc.ufpr.br; [3] jbittencourt@sulbbs.com

RESUMO

Este trabalho apresenta uma introdução às redes neurais artificiais como uma alternativa para a identificação de pontos de apoio pré-sinalizados. Inicialmente é feita uma rápida revisão bibliográfica sobre redes neurais artificiais, onde são apresentados os modelos de um neurônio, o perceptron, as redes feedforward e retropropagação, bem como um resumo de seu algoritmo. Em seguida são realizadas as etapas que compõem as redes neurais artificiais: coleta de dados, treinamento e simulação dos dados. São analisados dois tipos de pontos de apoio pré-sinalizados: em forma de cruz e em forma circular. São apresentados, examinados e comparados os resultados das identificações dos pontos pré-sinalizados, nas duas formas em cruz e circulares usando códigos fonte em MATLAB e em Visual C++, tanto para a fase de treinamento dos dados, quanto para a fase de testes dos mesmos, usando em conjunto a técnica de correlação estatística, com o objetivo de evitar ambigüidades.

ABSTRACT

This work presents an introduction to artificial neural networks as an alternative for the signalized point recognition. At first a fast bibliographical revision on artificial

neural networks is done, where the models of a neuron, perceptron, the networks are presented feedforward and backpropagation, as well as a summary of its algorithm. After that the stages are carried through that compose artificial the neural networks: it collects of data, training and simulation of the data. Two types of signalized point are analyzed: in cross form and circular form. They are presented, examined and compared the results of the identifications of the signalized point are compared, in the two form using codes source in MATLAB and Visual C++ , for the phase of training and for the phase of simulation of the data, using as well the technique as of correlation statistic for avoiding ambiguities.

1 INTRODUÇÃO

A identificação e medição das coordenadas fotogramétricas dos pontos participantes de uma fototriangulação podem ser realizadas manualmente em imagens digitais, no entanto oferecem o potencial de permitir automatização de inúmeras tarefas (ANDRADE, 1998, p.121). No tratamento digital de imagens, a correlação, que se constitui na comparação de imagens para a identificação de pontos homólogos que melhor se adaptam pela comparação de uma imagem de referência com outra imagem de busca é uma ferramenta possível de ser usada para a identificação de pontos de apoio nas imagens digitais de fotos aéreas. Entretanto, a área da matriz de busca deve ser restrita àquela em que se saiba, com certeza, que contém o ponto homólogo (ANDRADE, 1998, p.124). Isto pode se constituir em uma limitação se for utilizado métodos de correlação para identificar pontos de apoio. Neste sentido, técnicas aplicando redes neurais artificiais podem ser uma importante alternativa quando utilizadas em conjunto com a técnica de correlação estatística.

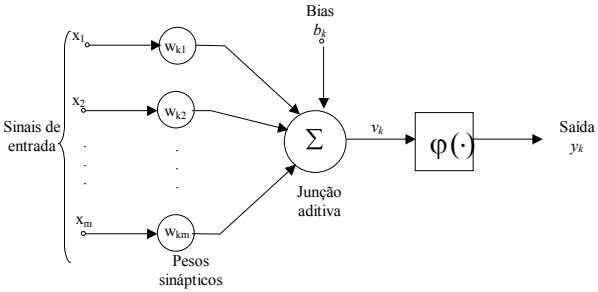
2 CONCEITOS BÁSICOS SOBRE REDES NEURAIS

As redes neurais artificiais são sistemas computacionais que imitam as habilidades computacionais do sistema nervoso biológico, usando um grande número de simples neurônios artificiais.

2.1 MODELO DE UM NEURÔNIO

Um neurônio é uma unidade de processamento de informação fundamental para as operações em uma rede neural. A figura 1 mostra o modelo de um neurônio que forma a base para o projeto de redes neurais artificiais. Identificam-se os seguintes elementos básicos em um modelo de neurônio artificial (HAYKIN, 2001, p.36):

Figura 1 - Modelo não linear de um Neurônio



a) Um conjunto de *sinapses* ou elos de conexão, cada uma caracterizada por um peso. O sinal x_j na entrada da sinapse j, conectada ao neurônio *k* é multiplicado pelo peso sináptico w_{kj}. O primeiro subscrito (*k*) se refere ao neurônio *k* e o segundo (*j*) ao sinal de entrada relativo ao respectivo peso (w_{kj}).

b) Uma *somatória*, caracterizada por uma combinação linear entre os sinais de entrada e os seus respectivos pesos.

c) Uma *função de ativação*, cuja finalidade é restringir a amplitude de saída (y_k) do neurônio.

d) Uma *perturbação* (bias), aplicada externamente, representada por bk (figura1). A perturbação tem o efeito de aumentar ou diminuir a entrada líquida da função de ativação, dependendo de ela ser positiva ou negativa, respectivamente.

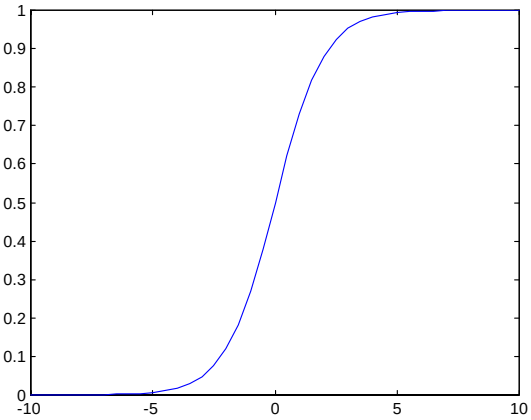
Do anteriormente exposto, pode-se então expressar um neurônio *k* pelas equações (1) e (2)

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{1}$$

$$y_k = \varphi\left(u_k + b_k\right) \tag{2}$$

A forma mais comum de função de ativação utilizada na elaboração de redes neurais artificiais é a denominada função logística *sigmóide* e apresenta forma de ***S***, como mostrado na Figura 2.

Figura 2 – Função logística sigmóide



A função logística sigmóide que dá origem ao gráfico da Figura 2, obtido com o uso do MATLAB é

$$f(x) = \frac{1}{1+e^{-x}} \tag{3}$$

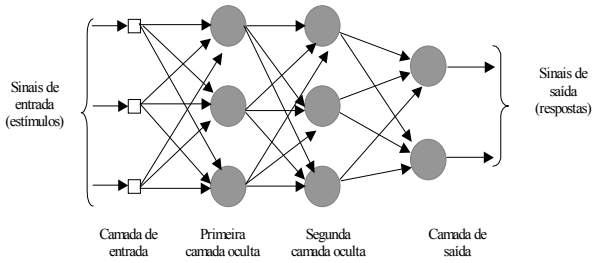
e cuja derivada é dada por:

$$\frac{df(x)}{dx} = f(x)[1 - f(x)] \tag{4}$$

2.2 REDES FEEDFORWARD E RETROPROPAGAÇÃO

Uma rede Feedforward (alimentadas para frente) ou Perceptron de múltiplas camadas consiste de um conjunto de unidades sensoriais formando a *camada de entrada,* uma ou mais *camadas ocultas* e uma *camada de saída* de nós computacionais. Este tipo de rede é uma generalização do perceptron simples (HAYKIN, 2001, p.183). Na Figura 3 está representada a estrutura de um Perceptron de múltiplas camadas.

Figura 3 - Perceptron de múltiplas camadas



O termo *retropropagação* (backpropagation), refere-se a um método de treinamento empregado na arquitetura do Perceptron de múltiplas camadas, no qual os pesos das conexões são ajustados pela retropropagação de erro. Esta retropropagação consiste de dois passos através das diferentes camadas da rede: um passo para frente, a *propagação*, e um passo para trás, a *retropropagação*. No passo de propagação, os pesos sinápticos são invariantes. No *passo para trás*, os pesos sinápticos são *ajustados* por uma regra de correção de erro que consiste na diferença entre a resposta real da rede e a resposta desejada. Isto produz um sinal de erro, que é propagado para trás através da rede em direção contrária às conexões sinápticas.

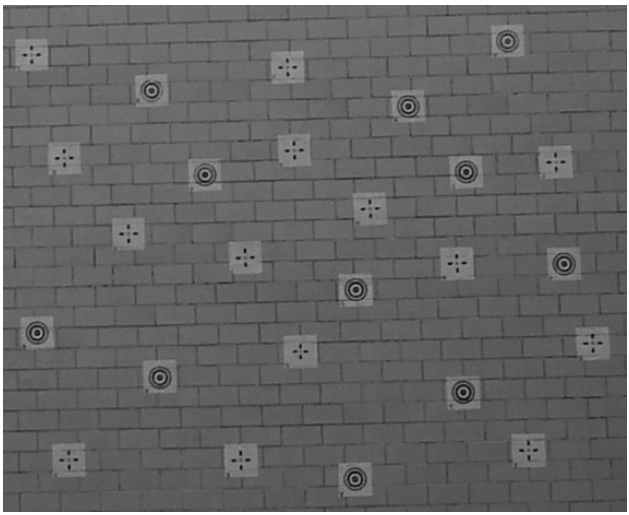
3 IDENTIFICFAÇÃO DOS PONTOS

Na aplicação de redes neurais, como em qualquer pesquisa de ordem prática, deve-se estabelecer etapas lógicas de execução. De acordo com (FILHO, p.1), numa etapa inicial, deve ser feita a *coleta de dados* do problema em questão. Com os dados coletados, a etapa seguinte é estabelecer um conjunto de treinamento. Em seguida é elaborado o treinamento da rede com o uso do conjunto de dados coletados. Após o treinamento vem a fase de testes (simulações) da rede com os dados usados no conjunto de treinamento, e principalmente com os dados do problema que não foram envolvidos no conjunto de treinamento.

3.1 COLETA DE DADOS

Nesta etapa inicial, ocorre a decisão de que tipos de pontos pré-sinalizados são convenientes para um início de identificação, utilizando inteligência artificial, através de redes neurais artificiais. Numa das paredes externas do Centro Politécnico da Universidade Federal do Paraná foram fotografados, com uma câmara digital, pontos em forma de cruz e também pontos em forma de círculo, como está mostrado na Figura 4, após transformar a imagem obtida pela foto com o auxílio do software Jasc Paint Shop Pro, para tons de cinza.

Figura 4 - Imagem 527x464 pixels em tons de cinza



3.2 TREINAMENTO

Após a coleta de dados, ocorre a fase de treinamento. Esta é também conhecida como fase de aprendizagem. Nesta fase é preciso fornecer à rede todas as informações necessárias, isto é, informações que sejam capazes de abranger o problema em estudo. Como o problema em foco é a identificação de pontos pré-sinalizados procedeu-se diversas tentativas, trabalhando-se com informações da imagem representada na Figura 5.

A imagem para o treinamento que forneceu melhores resultados foi aquela contendo os valores dos tons de cinza dos pixels de dois pontos distintos pré-sinalizados escolhidos aleatoriamente em forma de círculo e dois pontos pré-sinalizados em forma de cruz, bem como partes da imagem do fundo da parede. A codificação do programa fonte para o treinamento com a finalidade de identificar o ponto de apoio pré-sinalizado foi elaborada no MATLAB. O treinamento foi executado por feed forward-Backpropagation. Os primeiros treinamentos foram executados para que o ponto pré-sinalizado fosse identificado na posição os quais foram fotografados, como aparecem na Figura 5. Estes primeiros treinamentos foram elaborados para duas situações. Na primeira foi considerado o ponto de apoio pré-sinalizado como sendo aquele em forma de cruz. Na segunda situação, considerou-se o ponto de apoio pré-sinalizado como sendo aquele em forma de círculos concêntricos. No treinamento usando o Visual C++, foi implementado o

algoritmo RPROP. RPROP é um processo que tem, por finalidade tornar mais eficiente o algoritmo de retro-propagação "backpropagation" e significa as iniciais em inglês "Resilien Propagation". Conforme (RIEDMILLER,1993) a idéia básica do RPROP é eliminar a influência do tamanho das derivadas parciais nos cálculos dos pesos. Como conseqüência é considerado somente o sinal da derivada parcial na atualização dos pesos, de uma época (iteração) para outra.

3.2.1 Treinamento do alvo na forma de cruz

A Figura 5 apresenta o conjunto de treinamento utilizado para o treinamento no sentido de identificar o alvo na forma de cruz. Este conjunto é uma imagem de 183x33 pixels. O ponto pré-sinalizado cruz foi treinado em sua posição original, e também em várias outras posições, desde zero grau até 90 graus, com intervalo de variação de 11º25'(KËPUSKA e MASON ,1995,p.921)

Figura 5 – Conjunto de treinamento 183x33 pixels

Para que a rede neural reconheça a posição exata do alvo cruz, considera-se esta cruz contida, como no presente caso em uma matriz quadrada de 20x20 "pixels" que se denomina matriz de treinamento do ponto considerado, cujo canto esquerdo superior tem as coordenadas Y=5;X=5 e a partir daí é feito um cálculo para saber se o pixel está ou não contido nesta matriz. Este procedimento é elaborado novamente para a posição do canto esquerdo superior da segunda posição da matriz de treinamento com as coordenadas Y=5; X=105, isto é, na linha 5 e coluna 105. A rede neural está configurada com três camadas , sendo uma de entrada, uma escondida e uma de saída. A primeira, com o número de neurônios igual ao número de "pixels" do quadrado que contém o alvo, isto é, 400.A camada escondida apresenta o mesmo número de neurônios da camada de entrada, porém aplicando em cada um deles a função de transferência logística sigmóide. A camada de saída, que apresenta um único neurônio também com a aplicação da função logística sigmóide. Para que o treinamento encerre, estabelece-se o número máximo de 600 iterações (épocas) no processo de retro-propagação, bem como o mínimo valor para o erro médio quadrático (MSE) , isto é, o desvio padrão (GEMAEL,1994,P.88) de 10^{-20}. Finalmente, o treinamento é levado a efeito, onde a saída para o mesmo é o alvo quando o neurônio de saída indicar um valor tendendo para a unidade e não alvo quando o neurônio de saída indicar um valor tendendo para zero, conforme a função logística indicada na figura 2.

Tabela 1 – Treinamento com alvo cruz com o código em MatLab

Ângulo	Épocas	Desvio Padrão MSE Erro	Tempo (min)
0º	157	$2,7374x10^{-12}$	10
11º15	131	$1,2149x10^{-11}$	09
22º30	184	$6,5746x10^{-12}$	13
33º45	69	$1,1432x10^{-11}$	06
45º00	95	$2,6364x10^{-12}$	08
56º15	296	$4,8847x10^{-12}$	19
67º30	314	$1,2425x10^{-11}$	23
78º45	343	$1,3605x10^{-11}$	25

Tabela 2 – Treinamento com alvo cruz com o código em Visual C++

Ang.	Nivel	Épocas	Desvio Padrão MSE Erro	Tempo (seg)
0º	2	222	$4,6074x10^{-37}$	13
11º15	2	55	$1,3273x10^{-36}$	4
22,50	2	333	$9,7021x10^{-32}$	19
33,75	2	154	$6,4244x10^{-31}$	9
45	2	149	$1,2464x10^{-86}$	9
56,25	2	93	$2,9068x10^{-31}$	5
67,50	2	307	$1,2845x10^{-62}$	18
78,75	2	317	$9,8547x10^{-31}$	19
90,00	2	327	$3,9928x10^{-31}$	19
0º	1	75	$6,4162x10^{-31}$	21
11º15	1	53	$1,0286x10^{-41}$	15
22º30	1	48	$8,7274x10^{-34}$	13
33º45	1	104	$5,1050x10^{-31}$	18
45º00	1	62	$3,6331x10^{-31}$	17
56º15	1	200	$2,7450x10^{-49}$	55
67º30	1	56	$2,9651x10^{-38}$	16
78º45	1	67	$6,4159x10^{-31}$	19
90º	1	51	$2,0069x10^{-63}$	14

Na tabela 1, para cada ângulo foi realizado um treinamento, usando o código fonte em MatLab. Na primeira coluna estão indicados os ângulos de treinamento com variação de 11º15' (KËPUSKA; MASON,1995). Na segunda coluna estão

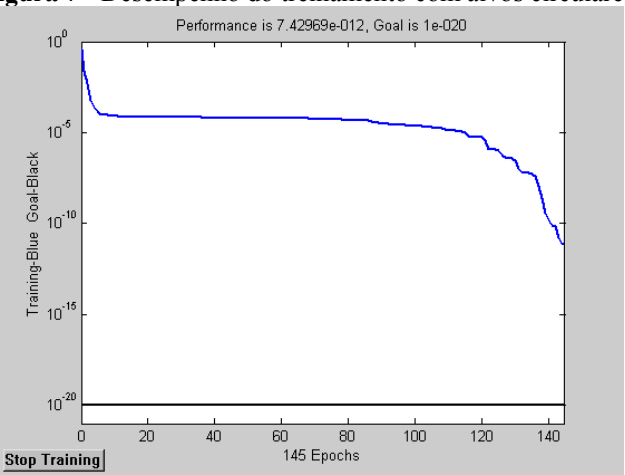
indicados os valores das iterações (épocas) em cada treinamento. Nas terceira e quarta colunas estão indicados respectivamente os valores dos erros e dos tempos de treinamento, em minutos. Na tabela 2 além das colunas indicadas na tabela 1 ocorre a coluna adicional indicando o nível onde o treinamento foi realizado. Aqui foi utilizado método de pirâmide de imagens (COELHO, 2001). A imagem no presente trabalho obtida no nível 1 consiste em substituir cada 4 "pixels" da imagem original pela sua média aritmética e a imagem no nível 2 em substituir cada 4 "pixels" da imagem do nível 1 pela sua média aritmética.

3.2.2 Treinamento de alvo na forma de círculos

Figura 6 – Conjunto de treinamento 162x36 "pixels"

Nesta hipótese, na qual o ponto pré-sinalizado é considerado a imagem dos círculos concêntricos, o código fonte é praticamente o mesmo para identificar cruzes. As mudanças neste caso são: o tamanho da imagem de treinamento que é de 162x36 pixels, mostrada na Figura 6; as coordenadas do canto esquerdo superior da primeira matriz de "pixels" que contém a imagem de círculos concêntricos são Y=8 e X=8. Neste caso também o treinamento ficou simplificado: bastou treinar o ponto pré-sinalizado em forma de círculos para a posição de zero grau, para que a rede reconheça na fase de simulação (testes) outras posições com ângulos diferentes de zero grau. A curva que representa o desempenho deste treinamento em MatLab está representada na Figura 7.

Figura 7 - Desempenho do treinamento com alvos circulares



No gráfico da Figura 7 , o eixo das abscissas representa o número de iterações do treinamento e o eixo das ordenadas o desvio padrão correspondente em determinada iteração. Os resultados do treinamento apresentam-se na Tabela 3. Na Tabela 4 está apresentado os resultados do treinamento com o código fonte em Visual C++.

Tabela 3 – Treinamento com alvos circulares com o código fonte em MatLab

Ângulo	Épocas	Desvio Padrão MSE	Tempo(minutos)
0	145	$7{,}42969 \times 10^{-12}$	09

Tabela 4 - Treinamento com alvos circulares com o código fonte em Visual C++

Ângulo	Nível	Épocas	Desvio Padrão MSE	Tempo (seg.)
0	2	216	$4{,}88152 \times 10^{-31}$	10
0	1	41	$1{,}02948 \times 10^{-32}$	07

Neste caso não há a necessidade de efetuar o treinamento para giros diferentes de zero grau, por ser o alvo circular.

3.3 TESTE DA REDE (SIMULAÇÃO)

Após o treinamento sucede a fase de teste (simulação) da rede, isto é, a fase de testes de validação. Os testes devem ser feitos não apenas para o conjunto de treinamento, mas também para todo o conjunto universo dos dados de onde o conjunto de treinamento foi retirado.

3.3.1 Testes para os alvos

O código fonte para os testes dos alvos elaborou-se em separado do código fonte de treinamento. Inicialmente é levada a efeito a leitura da imagem na qual se pretende identificar o alvo. Os tamanhos das matrizes para os testes devem ser aqueles do treinamento, isto é, 20x20 pixels. Se o teste for para identificar alvos do tipo cruz, então devem ser considerados todos os giros dos alvos candidatos do tipo cruz , desde zero grau até $78^{o}45^{'}$. Se for identificado um alvo, então estará indicada a sua posição central na matriz de "pixels" 20x20.

3.3.1.1 Resultados obtidos na identificação automática dos alvos em "cruz"

O primeiro teste elaborado foi na imagem I183x33.raw representada na Figura 5, que é a própria imagem de treinamento. Os resultados estão indicados nas Tabelas 5 e 6.

Tabela 5 - Simulação da imagem de treinamento com o código em MatLab

Ângulo	Linha	Coluna	Simulação
0	14	15	0,999979509
0	15	14	0,999991486
0	**15**	**15**	**0,999995577**
0	15	16	0,999995480
0	15	114	0,999990318
0	**15**	**115**	**0,999995520**
0	16	115	0,999994400

Tabela 6 – Simulação da imagem de treinamento com o código em Visual C++

Ângulo	Linha	Coluna	Simulação	Correlação estatística
0	15	15	1	1
0	15	115	1	0,977025

Analisando a Figura 5, nota-se duas imagens que representam o símbolo de cruz. Os resultados dos centros dessas cruzes estão indicados em negrito na Tabela 5, isto é, o centro da matriz de pixels da primeira cruz ocorre na posição Linha (L = 15) e Coluna (C =15), onde o valor da simulação **0,999995577** é maior do que para as outras três posições indicadas e o centro da ocorrência da segunda cruz está na posição Linha 15 e Coluna 115, com o valor de simulação **0,999995520,** que é maior do que as outras duas posições indicadas. A Tabele 6 apresenta o resultado da simulação com o uso do código fonte em Visual C++. Neste código, para evitar as ambigüidades ocorridas como no caso da Tabela 5 , aí foi incluído juntamente com redes neurais o método de correlação estatística (ANDRADE, 1998, p.123). No entorno do centro dos pontos com maiores simulações, como ocorreu na Tabela 5, correlaciona-se as matrizes com mesmo tamanho da matriz de amostra que foi usada no conjunto de treinamento. Aquela que apresentar a maior correlação é o alvo procurado. Na Tabela 5, a correlação estatística para o primeiro círculo da Figura 5 é 1, porque esta é própria matriz de amostra, cujo centro está na posição linha L = 15 e coluna L = 15. A segunda posição do alvo cruz , cujo centro na Tabela 6 é linha L=15 e coluna C=115 apresentou correlação estatística de 0,977025. Ao se comparar a Tabelas 5 com a Tabela 6 constata-se que em ambas o alvo cruz foi

identificado na mesma posição. No entanto, a vantagem do no uso da correlação estatística está no fato de que as ambigüidades foram eliminadas.

Um outro teste foi feito com a imagem representada na Figura 8:

Figura 8 - Imagem I137x67.raw

O centro do ponto pré-sinalizado cruz foi encontrado na posição L=18 e C=16, com o respectivo valor de simulação 0,9999955754, no código em MatLab. A Figura 9 foi obtida a partir da Figura 8, pela inclusão de alguns símbolos de cruz com diferentes rotações

Figura 9 - Imagem I137x67a.raw

Os resultados do teste estão apresentados nas Tabela 7 e 8. A cruz do meio da parte superior da Figura 9 foi girada em 30^o, no sentido horário, e o seu reconhecimento se deu com o ângulo treinado de $33^o45'$. Fatos similares ocorreram para as posições seguintes das cruzes.

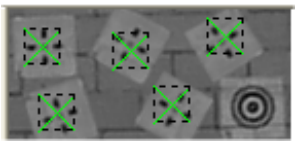
Tabela 7 - Simulação na imagem I137x67a.raw com o código em MatLab

Giro	L	C	Simulação
0	18	16	0,999995574
30^o ($33^o45'$)	15	57	0,999992790
300^O($33^o45'$)	47	79	0,999992536
150^o($56^o15'$)	14	103	0,999994812
75^o($78^o45'$)	50	25	0,999989393

Tabela 8 – Simulação na imagem I137x67a.raw com o código em Visual C++

Giro	L	C	Simulação	Correlação
0 (90^o)	18	16	1	0,982108
30^o ($67^o30'$)	20	59	1	0.857140
300^O($33^o45'$)	46	79	1	0,817270
150^o($45^{o'}$)	13	105	1	0,832333
75^o($67^o30''$)	50	23	1	0,851231

Figura 10 – Resultado da Simulação na imagem I137x67a.raw com o código em Visual C++



Na Figura 10, o "X" sobre o alvo indica a posição do centro do mesmo, como resultado da aplicação do código em Visual C++

Em seguida apresentam-se, nas Tabela 9 e 10 , os resultados obtidos pelo teste feito na Figura 4 .

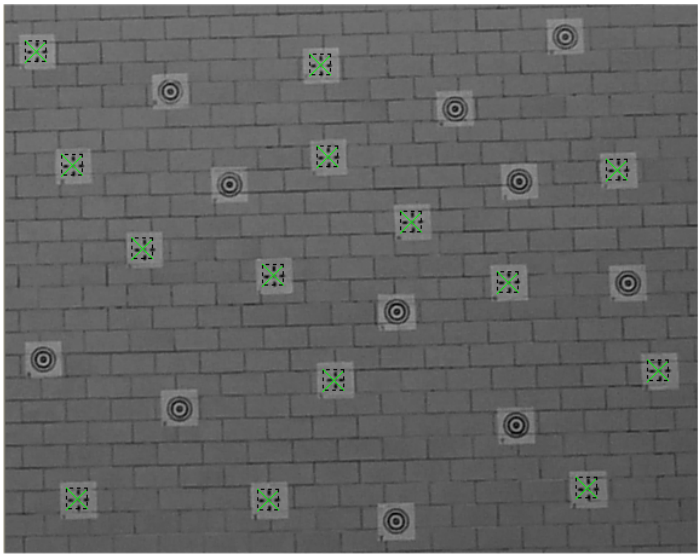
Tabela 9 – Simulação na imagem da Figura 4 com o código fonte em MatLab

L	C	Simulação
45	25	0,999993913
56	260	0,999993001
132	266	0,999993898
139	55	0,999993892
143	506	0,999993891
186	335	0,999993898
209	114	0,999993909
231	221	0,999993901
236	415	0,999993912
309	539	0,999993907
407	481	0,999993908
416	59	0,999993900
416	217	0,999993913

Tabela 10 – Simulação na imagem da Figura 4 com o código fonte em Visual C++

L	C	Simulação	Correlação estatística
45	25	1	1
56	260	1	0,977025
132	266	1	0,981162
139	55	1	0,982287
143	505	1	0,979135
186	335	1	0,985409
208	113	1	0,941564
230	221	1	0,964593
236	415	1	0,985852
309	539	1	0,981650
406	480	1	0,940443
415	59	1	0,967371
416	217	1	0,987316

Figura 11 – Resultado da simulação na imagem da Figura 4 com o código fonte em Visual C++



3.3.1.2 Resultados obtidos na identificação automática dos alvos "circulares"

Considere-se agora o teste nas imagens levando em conta o treinamento da situação na qual os pontos pré-sinalizados são os círculos concêntricos. Inicialmente considera-se a própria imagem de treinamento, definida na Figura 11. Os resultados deste teste indicam-se na tabela 11

Tabela 11 - Simulação da imagem de treinamento

Ângulo	Linha	Coluna	Simulação valor de y
0	18	18	0,999993008
0	18	98	0,999992573

A seguir apresenta-se na Tabela 12 o teste na imagem representada na Figura 12.

Figura 12 - Imagem I192x202.raw

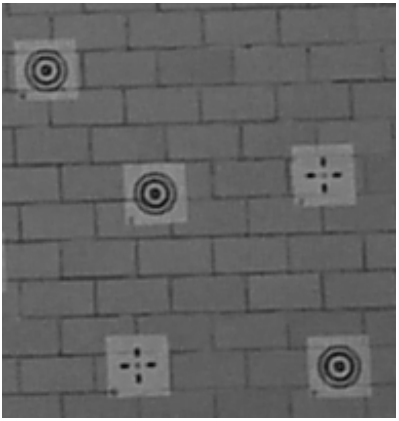


Tabela 12 - Teste na imagem I192x202.raw usando o código fonte em MatLab

Ângulo	Linha	Coluna	Simulação
0	32	21	0,999992949
0	92	74	0,999993005
0	176	164	0,999993001

Na Tabela 14 apresentam-se os resultados da simulação na imagem I192x202a.raw, da Figura 13, onde os pontos pré-sinalizados são os círculos concêntricos, em diversas posições angulares.

Figura 13 - Imagem I192x202a.raw

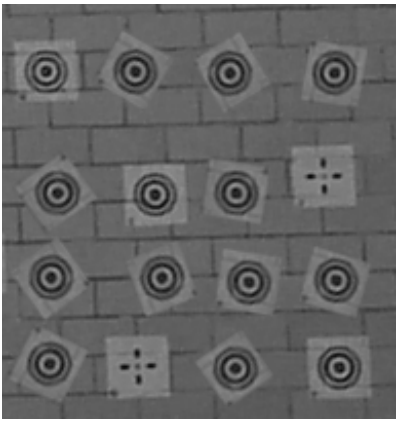


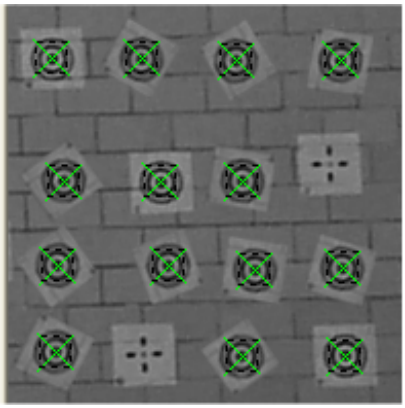
Tabela 14 - Simulação na imagem I192x202a.raw usando o código fonte em MatLab

L	C	Simulação
32	21	0,999992949
33	65	0,999992958
33	111	0,999992999
33	162	0,999993000
92	27	0,999993000
92	74	0,999993000
92	114	0,999993000
133	24	0,999993002
133	78	0,999993003
135	120	0,999993006
134	163	0,999992997
175	23	0,999993002
177	114	0,999992976
177	164	0,999993008

Tabela 15 - Simulação na imagem I192x202a.raw usando o código fonte em Visual C++

L	C	simulação	Correlação estatística
32	21	1	0,923710
32	65	1	0,930932
33	111	1	0,960768
33	162	1	0,973340
92	27	1	0,947917
92	74	1	0,927460
92	114	1	0,955171
133	24	1	0,954503
133	78	1	0,963829
135	120	1	0,980794
134	163	1	0,957435
175	23	1	0,965874
177	114	1	0,942007
177	164	1	0,936388

Figura 14 – Resultado da Simulação na imagem I192x202a.raw usando o código fonte em MatLab



As duas primeiras colunas da Tabela 14 representam as coordenadas do centro dos círculos, que são identificados como pontos pré-sinalizados. Na terceira coluna estão representados os valores da simulação, isto é, os valores aplicados à função de transferência logística sigmóide. Não foi necessário indicar o giro em cada posição

dos pontos pré-sinalizados, porque todos foram identificados pela rede neural, como posição de zero grau, isto é, sem giro.

Nas Tabelas 16 e 17 estão apresentados os resultados obtidos para o teste de identificação dos centros dos alvos círculos concêntricos usando respectivamente os códigos fonte MatLab e Visual C++

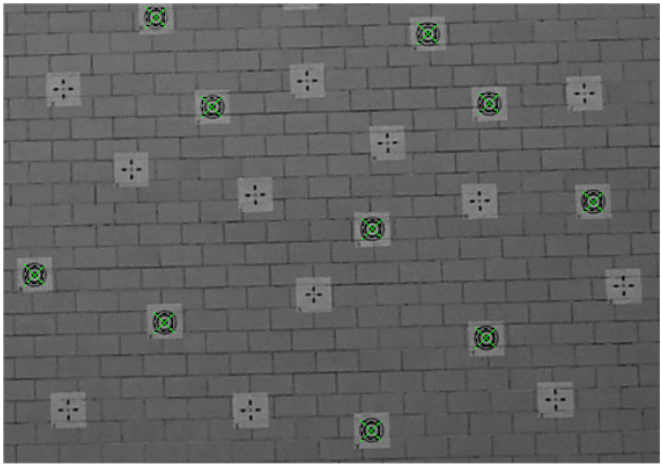
Tabela 16 – Simulação na imagem da Figura 4 com o código fonte MatLab

L	C	Simulação
33	462	0,999992573
78	136	0,999993008
92	371	0,999992949
152	424	0,999993005
155	185	0,999993000
236	514	0,999993001
260	326	0,999993008
300	31	0,999993008
341	144	0,999992982
422	354	0,999993008
434	323	0,999993005

Tabela 17 – Simulação na imagem da Figura 4 com o código fonte em Visual C++

L	C	Simulação	Correlação Estatística
33	462	1	0,952747
78	136	1	1
92	371	1	0,923710
152	424	1	0,927460
155	185	1	0,833280
237	514	1	0,936388
260	323	1	0,952935
300	31	1	0,959468
341	144	1	0,895522
421	354	1	0,833531
434	323	1	0,875737

Figura 15 – Resultado da Simulação na imagem da Figura 4 com o código fonte em Visual C++



4 CONCLUSÕES

Ao se utilizar o ponto pré-sinalizado "cruz" foi necessário rotação conforme se indica na Tabela 1, para que a rede neural, assim treinada fosse de identificar este ponto em qualquer outro giro diferente daquele indicado nesta tabela. Por outro lado, quando se considerou o ponto sinalizado "círculo" foi suficiente realizar o treinamento apenas na posição "zero grau". Desta forma se for feita comparação entre os treinamentos dos pontos pré-sinalizados "círculo" e "cruz" , o "círculo" leva enorme vantagem em relação à "cruz". Esta vantagem também se verifica quando da simulação (testes de validação) da rede neural. Nas tabelas 3, 4 e 5 estão representadas simulações para a identificação de "cruzes" e nas tabelas 6,7 e 8 simulação para a identificação de "círculos". Tanto na fase de treinamento, quanto na fase de simulação o esforço computacional para o treinamento e posterior identificação do ponto pré-sinalizado "círculo" foi bem inferior ao do ponto pré-sinalizado "cruz".

Por outro lado, quando se comparam os esforços computacionais nos códigos em MatLab e Visual C++ , enquanto o tempo para o treinamento está na ordem de minutos para o caso do MatLab, no caso de Visual C++, o tempo baixa drasticamente para a ordem de segundos. Isto é sem dúvida uma enorme vantagem quando se quiser trabalhar com imagens maiores de treinamento ou de simulação, como pode ocorrer, por exemplo em aplicações na aerofotogrametria se for considerado o problema de encontrar pontos de apoio homólogos pré-sinalizados ou não para o apoio no problema de aerotriangulação.

REFERÊNCIAS BIBLIOGRÁFICAS

ANDRADE, J.B. **Fotogrametria**. Curitiba: SBEE, 1998, 258p.

COELHO, A. H . et al. **Utilização do método de pirâmide de imagens para a extração de modelos digitais de terreno de imagens geradas por dados de laser scanner.** Universidade de Karlsruhe, Instituto de Fotogrametria e Sensoriamento Remoto, Karlsruhe, Alemanha, 2001.

DEPENAU, J. **Automated Design of Neural Network Architecture for Classification**. Aarhus, 1995, 187p. Ph.D. Thesis. Daimi, Computer Science Department, Aarhus University.

FAUSETT, L. **Fundamentals of Neural Networks: Architectures, Algorithms, and Aplications**. New Jersey: Florida Institut of Technology, 1991. 461p.

FILHO, Elson Felix Mendes (2002). **Uma Introdução a Redes Neurais Artificiais** <http://sites.uol.com.br/elson_mendes/rn/rn_dap.html>. Acesso: 27 fevereiro 2002.

GEMAEL,C. Introdução ao Ajustamento de Observações. Aplicações Geodésicas. Curitiba: Editora UFPR, 1994, 319p.

HAYKIN, S. **Neural Networks. A Comprehensive Foundation.** New York: IEEE Computer Society Press, 1994, 696p.

_____. **Redes Neurais Princípios e Prática.** Tradução de Paulo Martins Engel. Porto Alegre: Bookman, 2001. 900p. Título original: **Neural Networks. A Comprehensive Foundation.**

KËPUSKA, V. Z.; MASON, S. O. A Hierarchical Neural Network System for Signalized Point Recognition in Aerial. Photographs. **Photogrammetric Engineering & Remote Sensing**, vol. 61, Nº7 , pp. 917-925, jul.1995.

_____. Automatic Signalized Point Recognition With Feed-Forward Neural Network. In: IEE International , Conference on Artificial Neural Networks, 2., 1991, Bournemouth, United Kindom, p. 359-363.

KOVÁCS, L. Z. **Redes Neurais Artificiais: Fundamentos e Aplicações.** 2ª ed. São Paulo: Collegium Cognitio, 1996,173p.

LOESCH, C.; SARI, S.T. , **Redes Neurais Artificiais. Fundamentos e Modelos.** Blumenau: Ed. Da FURB, 1996, 166p.

RIEDMILLER, M.A Direct Adaptative Method for Faster Backpropagation Learnein: The RPROP Algorithtm. In: Proceedings of the IEEE. **International Conference on Neural Network (ICN)** , San Francisco – USA, 1993, p.586-591

(Recebido em abril/03. Aceito em novembro/03)