Improved neural network modeling of inverse lens distortion

Jason P. de Villiers[a,b], Jaco Cronje[a] and Fred Nicolls[b]

[a]Council for Scientific and Industrial Research, Pretoria, South Africa;
[b]University of Cape Town, Cape Town, South Africa

ABSTRACT

Inverse lens distortion modelling allows one to find the pixel in a distorted image which corresponds to a known point in object space, such as may be produced by RADAR. This paper extends recent work using neural networks as a compromise between processing complexity, memory usage and accuracy. The already encouraging results are further enhanced by considering different neuron activation functions, architectures, scaling methodologies and training techniques.

Keywords: Neural network, lens distortion, inverse distortion distortion corrections

1. INTRODUCTION

1.1 Lens Distortion

Lens distortion is the non-linear refraction of light rays by a lens such that it deviates from the ideal pin-hole camera projection model. Complex lenses, those made of several glass elements, typically exhibit monotonically increasing contraction of the light ray bundle towards the optical axis for wide angle lenses. This is known as barrel distortion. Pin cushion distortion, found in some tele-photo lenses, is the expansion of the light ray bundle as a function of angle from the optical axis. Lenses incorporating aspherical elements can be designed to have significantly less distortion, often by several orders of magnitude, but the remaining residual distortion is no longer monotonic and is known as moustache distortion.

Distortion in images affects any measurements made from those images. Straight lines may appear curved, and measurements such as angular separation of two image points or triangulation of a point visible to multiple cameras will be adversely affected.

The definitive lens distortion model is that of Brown[1,2], which is a polynomial expansion of Conrady's model[3]. Brown's model, given by 1, allows one to calculate with the desired accuracy, the corresponding undistorted point (i.e. that which would have been produced by a pinhole camera) for any point in the distorted image. The accuracy of the model is a function of the number of radial and tangential parameters that are determined, and whether an optimal distortion centre is found. In order to create a distortion-free image, however, the inverse operation is required: for each (integer located) pixel in the undistorted image, the (probably non-integer located) pixel in the distorted image is required. This allows one, after interpolation in the distorted image, to generate the distortion-free image without gaps or holes. This process, known variously as undistortion, inverse distortion and distortion correction, additionally allows one to determine the image coordinates of a reference determined by external means such as RADAR or geo-location, after the provided vector is projected into the focal plane and scaled by the pixel dimensions.

Further author information: (Send correspondence to J.d.V.)
J.d.V.: E-mail: jdvilliers@csir.co.za

Visual Information Processing XX, edited by Zia-ur Rahman, Stephen E. Reichenbach, Mark A. Neifeld,
Proc. of SPIE Vol. 8056, 80560L · © 2011 SPIE · CCC code: 0277-786X/11/$18 · doi: 10.1117/12.884065

$$
\begin{aligned}
x_u =& x_d + (x_d - x_c)(K_1 r^2 + K_2 r^4 + \ldots) + \\
& \left(P_1(r^2 + 2(x_d - x_c)^2) + 2P_2(x_d - x_c)(y_d - y_c)\right) \times \\
& (1 + P_3 r^2 + \ldots) \\
y_u =& y_d + (y_d - y_c)(K_1 r^2 + K_2 r^4 + \ldots) + \\
& \left(2P_1(x_d - x_c)(y_d - y_c) + P_2(r^2 + 2(y_d - y_c)^2)\right) \times \\
& (1 + P_3 r^2 + \ldots)
\end{aligned} \tag{1}
$$

where:

(x_u, y_u) =undistorted image point,

(x_d, y_d) =distorted image point,

(x_c, y_c) =centre of distortion,

K_n =N$^{\text{th}}$ radial distortion coefficient,

P_n =N$^{\text{th}}$ tangential distortion coefficient,

r =$\sqrt{(x_d - x_c)^2 + (y_d - y_c)^2}$, and

$\ldots$ =an infinite series.

As Brown's model contains no precise inverse for non-trivial parameter sets, various methods of approximating it have been investigated. Candocia[4] presented an alternate scale preserving distortion function, which had an analytical inverse requiring the solving of two fifth-order polynomials. Mallon and Whelan[5] used a Taylor series expansion of a Brown model containing only a few coefficients. De Villiers *et al*[6] fitted a high-order Brown model to map the inverse distortion by using the pseudo-undistorted points created during distortion characterisation.

1.2 Artificial Neural Networks

Artificial Neural Networks (ANN) are a mathematical approximation to a biological brain. The synaptic connections to other neurons are represented by input weights. The action potential on the soma is represented by the output of a smooth output bounded function of the difference between the sum of the weighted inputs and an internal threshold. Standard texts (e.g.[7]) can be consulted for more details.

Feed-forward ANNs are a standard architecture[7] wherein the neurons are arranged in layers. The input neurons each receive one of the inputs and distribute it to the next layer. Each of the neurons in the subsequent layers accept as inputs the outputs of all the neurons in the preceding layer. This is called a fully-connected network. The original method to train a network consisting of only one neuron (sometime called a perceptron) was developed by Rosenblatt in 1958[8] and was extended to networks with multiple neurons in multiple layers by Bryson and Ho in 1969[9]. This is the ubiquitous backwards propagation model, which is analogous to steepest descent numerical optimisation.

Much work has been done on applying ANNs to computer vision problems, Egmont-Peterson *et al.*[10] provide a review and classification taxonomy for their application. Camera calibration falls within their pixel-level preprocessing category. Memon and Khan[11] trained an ANN to produce the 3-dimensional position of a point given the matched image coordinates of the points as observed by a stereo camera pair. Do[12] used neural nets both to model the entire system as well as to model only the deviation from the pinhole camera model. Ahmed *et al.*[13] used an ANN to determine the intrinsic and extrinsic calibration parameters of a camera, excluding lens distortion effects which were assumed either insignificant or accounted for upstream.

1.3 Contribution

This work is an extension of previous work published by the authors.[14] Earlier work on modelling inverse distortion did not consider neural networks. Previous work using ANNs for camera calibration either did not consider the effect of lens distortion or implicitly modelled distortion only in the direction of distorted to undistorted

domains. This work investigates the applicability of modelling inverse distortion using ANNs and the residual pseudo-undistorted points created during normal lens distortion procedures.

The rest of this paper is divided as follows: First, the equipment and methods used to capture the data are described in section 2. Then the architectures of the neural networks that were evaluated are discussed in section 3. Section 4 describes the algorithms used to train the neural networks. Section 5 provides the results of the training of the neural networks. Section 6 discusses and places the results in context. Finally, section 7 provides some concluding remarks.

2. DATA CAPTURE METHOD

The data used for this work is the same as the authors' earlier work.[14] This section is repeated for convenience. A 46" high definition liquid crystal display (LCD) television was used to create checker patterns to provide data for the distortion characterisation. This allowed many thousands of checkers to be generated and captured, whilst their exact positions were accurately known. A Prosilica GE1600 machine vision camera with a Schneider 4.8mm Cinegon lens was used to observe the LCD screen. The LCD and camera were arranged such that the LCD was as far away as possible, but still subtended the camera's entire field of view (FOV). The Schneider lens exhibited classical monotonic barrel distortion with a horizontal FOV of approximately 82°. The GE1600 had a resolution of 1600 by 1200 and 2/3 inch CCD. The ambient lighting was carefully controlled and simple background elimination performed by subtracting intermittently refreshed reference frames with a blank LCD. This ensured that erroneous reflections in the LCD did not adversely affect the data. Figure 1 provides a raw uncorrected picture captured with the GE1600 and 4.8mm Cinegon lens. Note how the building on the left, the column on the right and the fence at the bottom all appear to be curved.

Figure 1. Original distorted image

The checkers were found in the camera's image with sub-pixel accuracy using Lucchese and Mira's[15] method of finding the saddle point of the 6 coefficient second-order surface of intensity versus X and Y. This surface was fitted in a least squares sense in a window centered around the initial roughly found checker position. This refined position was further refined, using the same method and centering the window around the previous refined position to ensure that the final window was indeed centered (to the nearest pixel) around the refined checker position for maximum accuracy.

The distortion characterisation, and corresponding calculation of the pseudo-undistorted points, was performed as described by de Villiers *et al.*[6] using Brown's model with five radial coefficients, three tangential coefficients and an optimal distortion center.

3. NEURAL NETWORK ARCHITECTURES

All the ANNs considered in this work were of the fully connected feed forward type. A variety of sigmoidal activation functions were evaluated. The logistic function given by Eq. 2 has an output $\in (0, 1)$. The arctan output (Eq 3) is bounded by $(-\pi/2, \pi/2)$. The third function considered, the hyperbolic tangent (Eq 4), is $\in (: -1, 1)$. Figure 2 plots the output of the functions on a common axis.

$$Output = \frac{1}{1 + e^{th - \sum_{j=1}^{n}(w_j i_j)}} \tag{2}$$

$$Output = tan^{-1}\Big(\sum_{j=1}^{n}(w_j i_j) - th\Big) \tag{3}$$

$$Output = tanh\Big(\sum_{j=1}^{n}(w_j i_j) - th\Big) \tag{4}$$

where:

$n =$ the number of inputs to the neuron,
$w_j =$ the weight associated with input j,
$i_j =$ the jth input, and
$th =$ the neuron's threshold.

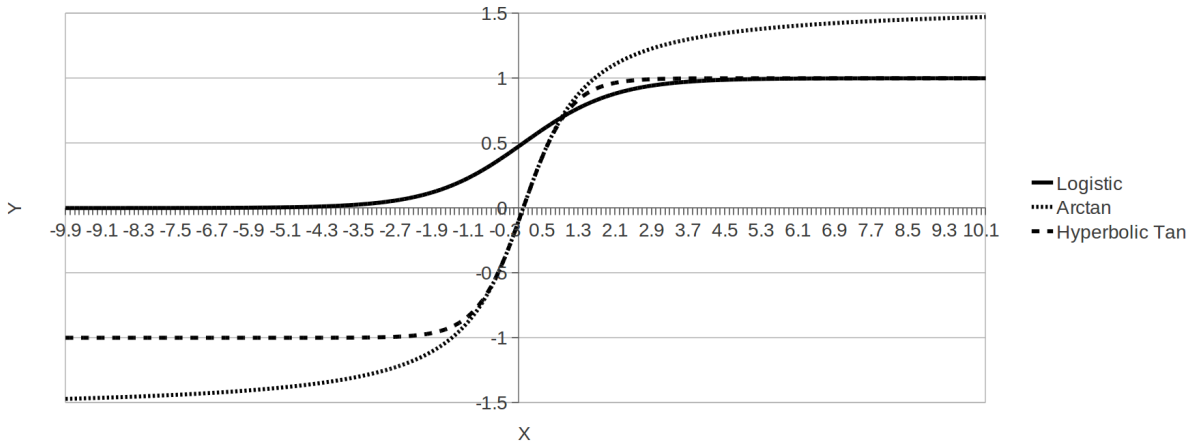


Figure 2. Plot of the sigmoidal functions considered.

The output of the ANN's were scaled such that only the middle 80% of the neuron activtion function range corresponded to pixel on the imager, i.e. 10% corresponds to pixel 0 and 90% corresponds to pixel 1600 (1200) for the horizontal (vertical) correction. Pairs of ANNs were evaluated. each with single outputs correcting either the horizontal or vertical corrections as well as single dual-output ANN. Following the results of earlier work[14], ANNs with 3 hidden layers (HL) were not evaluated, only 1 and 2 HL ANNs were. Single HL ANNs had either 5, 10 or 15 neurons in the HL. Dual HL ANNs had 5, 6 or 8 neurons in each HL.

4. ALGORITHM DESCRIPTIONS

For the first two sets of evaluations described in Section 3, a multi-start strategy was implemented. Twenty networks were initialised with random weights and thresholds, and were then trained using the standard backwards propagation technique.[9] An adaptive learning rate (initially set to 0.1) was employed whereby the learning rate was increased by a multiplicative factor of 1.01 (up to a maximum of 1.0) if two of the past three epochs resulted

in an improvement. The learning rate was decreased by the same multiplicative factor to a minimum of 0.05 if two of the past three epochs showed a worsening. The ANNs were trained using two thirds of the data, and evaluated on the last third. As the data was captured sequentially from top left to bottom right of the camera's FOV, the training and evaluation data were interleaved. Specifically the ANNs were trained on samples $3n + 0$ and $3n + 1$ and evaluated on $3n + 2$. All the image coordinates were scaled by the imager's resolution to be in the normalized range $(0, 1)$. This implies that the distorted image coordinates were $\in (0, 1)$ but undistorted ones corresponding to distorted points on the periphery were not.

The backwards propagation algorithm iterates sequentially through each input-output set in the epoch and makes a small adjustment. For the third series of evaluations, it was decided to determine what adjustments should be made to the neuron's parameters to make a global improvement for the entire epoch. To do this, ANNs of the specified architecture were trained and discarded using the procedure just described, until one achieved an error of 0.002 or better. Thereafter, the Leapfrog local optimiser[16] was used (with a centered difference gradient estimation) to refine the weights. This implies 2 evaluations of the epoch per weight and threshold in the ANN in order to determine what step to take to improve the network. A total of 102 epochs are thus required for a network with two inputs, two hidden layers of 5 neurons and a single output. The metric that was minimized was the Root of the Mean Square (RMS) error of the difference between the actual distorted image coordinate and the one that the ANN produced based on the pseudo-undistorted image coordinate.

As can be seen in Figure 2 the activation functions have different output ranges. It had previously been noted[14] that using only a subset, such as the inner 80%, of the range should be used to represent the undistorted pixel space. This allows points which are too far off the optical axis to fall on the CCD to not be mapped erroneously onto the image.

5. RESULTS

All timings in Tables 1, 2 and 3 were performed on the same computer. It had an Intel i7-960 CPU, and an NVidia GTX480 GPU. The CPU based parallelisation was implemented using openMP* and the GPU code was written in CUDA†.

Table 1 provides the results of the parallelisation of the neural network training and refinement algorithms. The effect of the precision of the floating point weights on the final results is displayed. The networks trained have two hidden layers, each with 5 neurons and using the logistic activation function (Eq. 2). The results are thus comparable to the authors' previous work,[14] except the suggested output scaling has been implemented. Each evaluation consisted of training 20 ANNs to correct only the horizontal distortion correction. The RMS error in pixels for each ANN was averaged and is presented along with the RMS error of the best ANN. The timing is the average time taken to train an ANN of that type.

Table 1. CPU vs. GPU and single vs. double precision comparison.

Processor	Precision	Backwards Propagation		Optimised		Time per ANN
		Average	Best	Average	Best	
		Pix (RMS)	Pix (RMS)	Pix (RMS)	Pix (RMS)	sec
CPU, single thread	double	10.14	1.50	2.83	0.83	1892
CPU, single thread	single	11.19	1.87	8.02	1.44	1754
CPU, multi-threaded	double	10.21	1.04	2.95	0.75	279
GPU	double	8.81	1.14	3.25	0.68	152

Table 2 provides the results of the comparison of the different activation functions. It contains both pairs of single-output ANNs (used in combination to perform a total correction) as well as dual-output ANNs, all of which were refined on the GPU. As it was anticipated that more neurons would be required for one ANN to

*openMP (www.openMP.org) is API for multi-platform shared-memory parallel programming in C/C++ and Fortran.
†CUDA (www.nvidia.com/object/cuda_home.html) is a parallel programming architecture developed by NVidia.

effectively encapsulate the horizontal and vertical corrections, ANNs with 6 and 8 neurons per HL were tested in addition to the prior ANNs with 5 neurons per HL. Each ANN architecture/activation function combination had 20 ANN's trained. The RMS error in pixels for each ANN was averaged and is presented along with the RMS error of the best ANN. As before, the timing is the average time taken to train an ANN of the that type. The combined column is the total error produced by two single output ANN's and its time is the sum of the times for the two ANNs. The best results have been highlighted, for ease of reference.

Table 2. Activation function and ANN architecture comparison.

Hidden Layers	Activation Function	Single output ANNs							Dual output ANN		
		Horizontal		Vertical		Combined					
		Ave Pix	Best Pix	Ave Pix	Best Pix	Ave Pix	Best Pix	Time sec	Ave Pix	Best Pix	Time sec
5x5	logistic	3.25	0.68	1.47	0.64	3.47	0.93	411	1.91	0.91	304
	arctan	1.56	0.60	0.89	0.31	1.80	0.68	328	1.01	0.59	303
	tanh	0.76	0.37	0.57	0.25	0.95	**0.45**	394	0.76	0.56	329
6x6	logistic	1.61	0.46	1.30	0.70	2.07	0.84	543	1.37	0.81	342
	arctan	0.74	0.59	0.93	0.32	1.19	0.67	423	0.96	0.46	275
	tanh	0.67	0.39	0.49	0.31	0.83	0.50	497	0.64	0.46	394
8x8	logistic	0.64	0.34	1.22	0.50	1.38	0.60	929	1.03	0.51	513
	arctan	1.26	0.54	0.56	0.23	1.39	0.59	580	0.62	**0.31**	526
	tanh	0.68	0.51	0.43	0.30	**0.80**	0.59	664	**0.57**	0.44	647

Table 3 provides equivalent data to Table 2 but uses ANNs that have only one HL. Only the total pixel error for two single output ANNs is presented. Each ANN architecture/activation function combination had 20 ANNs trained. The RMS error in pixels for each ANN was averaged and is presented along with the RMS error of the best ANN.

Table 3. Results using a single HL ANN

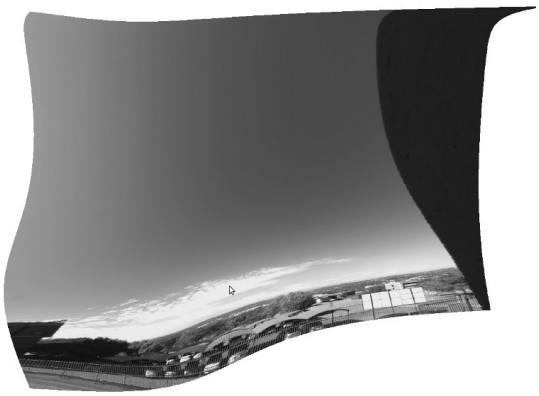
ANN architecture	Neuron activation function	5 Neurons			10 Neurons			15 Neurons		
		Ave Pix	Best Pix	Time Sec	Ave Pix	Best Pix	Time Sec	Ave Pix	Best Pix	Time Sec
Two single-output ANNs	logistic	8.46	2.17	245	2.03	1.30	463	1.98	1.07	731
	arctan	24.2	21.3	142	29.9	27.4	291	29.9	28.3	478
	tanh	10.6	10.4	91	10.4	10.4	266	10.0	2.83	236
One Dual-output ANN	logistic	4.28	1.21	159	1.67	1.16	286	1.38	0.73	421
	arctan	15.4	13.1	161	17.6	16.0	290	17.7	3.70	399
	tanh	2.26	2.16	61	0.89	0.57	160	2.38	2.02	219

6. DISCUSSION OF RESULTS

Table 1 provides the results of the attempts at speeding up the process of training and refining the neural networks. This was done to facilitate more experiments and increase the rigour with which they could be performed. It can be seen that switching from double to single precision resulted in only a marginal performance gain, outweighed by the decreased accuracy of the ANNs. Both the CPU and GPU multi-threaded applications (only the metric used for numeric refinement was parallelised, as pure Backwards propagation is not parallelisable) yielded significant performance gains in training times. The achieved accuracy of the ANNs was not degraded, with the results having similar distributions. All further evaluations were then done using double precision and GPU based refinement (which requires a GPU supporting CUDA compute capability 1.3 or above).

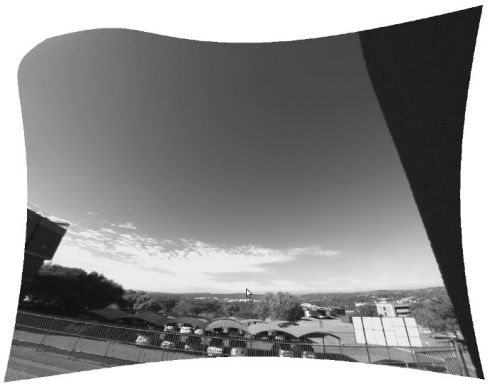
(a) Image undistorted as per de Villiers *et al*.[6]

(b) 10 pixel error ANN corrected image.



(c) 5 pixel error ANN corrected image.

(d) 2 pixel error ANN corrected image.



(e) Best single HL ANN (0.57 pixel error)

(f) Best dual HL ANN (0.31 pixel error)

Figure 3. Distortion corrected images.

From Table 2 it can be seen that, in almost all cases, a single dual-output ANN outperformed two single-output ANNs. This is true both for the average accuracy of the ANNs and for the best performing ANNs of each type. Dual-output ANNs are also quicker to train than the associated total time for a pair of single-output ANNs.

In terms of activation functions, the logistic function (Eq. 2) is in all cases the worst performing of the three functions. The hyperbolic tangent (Eq. 4) in all cases had the best average accuracy of the three activation functions. The arctan (Eq. 3) activation function had best results that were similar to or better than the hyperbolic tangent. However it was observed that several of the ANNs would converge to particularly poor results, and so it seems the increased range of arctan makes these ANNs more sensitive.

Figure 3 helps place these numerical results in perspective. Figure 3(a) is the reference distortion corrected image, and portrays the effect we are trying to achieve. Figures 3(b), 3(c) and 3(d) provide the results of images corrected to within 10, 5 and 2 pixels RMS over the entire image. Figures 3(e) and 3(f) provide the images obtained with the best single HL and dual HL ANNs obtained in this work. Note how the building on the left, the column on the right and the fence at the bottom are now all projected onto straight lines.

The single HL ANNs, whose results are given in Table 3, provide contrasting results. While a single network performing both horizontal and vertical corrections continues to provide a better fit (with fewer computations), the most accurate activation function is now the sigmoidal function (with the exception of a 10 neuron hyperbolic tan single HL network's which had the best results on average and overall). ANNs using the arctangent function performed particularly poorly, with most ANNs being trained by backwards propagation into a local minima that the leapfrog refinement could not significantly improve.

Table 4 places the results obtained in context with established and recently published papers. The first six rows correspond to the best results achieved, regardless of the number or neurons or hidden layers, for ANNs using different activation functions and numbers of outputs. The second section of the table gives the results achieved in the authors' previous work and from using high-order brown models[6] to create Figure 3(a). The final section provides, for comparison, results published in literature. Mallon and Whelan's camera resolution was inferred from their paper and the imager size assumed to be 2/3 inch.

It can be seen the relative to the author' prior work on neural modelling of inverse distortion[14] a further decrease in the error 63.5% has been obtained. This makes the work directly comparable to some of the most accurate algorithms published.

Table 4. Distortion results comparison

Correction Type	Resolution		Pixel Size (μ)	Pixel Error	Micron Error
	Horiz	Vert			
2 single-output logistic ANNs	1600	1200	5.5	0.60	3.30
2 single-output arctan ANNs	1600	1200	5.5	0.59	3.23
2 single-output tanh ANNs	1600	1200	5.5	0.45	2.48
1 dual-output logistic ANN	1600	1200	5.5	0.51	2.81
1 dual-output arctan ANN	1600	1200	5.5	0.31	1.71
1 dual-output tanh ANN	1600	1200	5.5	0.44	2.42
de Villiers[14]	1600	1200	5.5	0.85	4.68
de Villiers[6]	1600	1200	5.5	0.30	1.65
Mallon[5]	1024	768	8.6	0.42	3.61
de Villiers[6]	667	502	13.15	0.013	0.17

7. CONCLUSIONS

By parallelising the local refinement of the ANN weights, a total speedup of over 12 times was yielded without decreasing the accuracy of the resulting networks.

If only one or a few ANNs can be trained, a dual output 2 hidden layer network with between 6 and 8 neurons in each layer and a hyperbolic tangent activation function should be used, as these have the lowest average results.

If one can afford to train many ANNs the arctangent activation function will yield results better or similar to the hyperbolic tangent ANNs. If one is constrained to a single layer, the sigmoidal activation function yields the most predictable and generally best results.

It has been shown that artificial neural networks can effectively undistort an image by modelling the intractable inverse distortion. The results achieved are only 3% worse than the most accurate algorithms available when they are performed on the same apparatus.

REFERENCES

[1] Brown, D., "Decentering distortion of lenses," *Photogrammetric Engineering* **7**, 444–462 (1966).

[2] Brown, D., "Close range camera calibration," *Photogrammetric Engineering* **8**, 855–855 (1971).

[3] Conrady, A., "Decentered lens systems," *Monthly Notices of the Royal Astronomical Society* **79**, 384–390 (1919).

[4] Candocia, F., "A scale-preserving lens distortion model and its application to image registration," in [*Proceedings of the 2006 Florida Conference in Recent Advances in Robotics*], *FCRAR 2006* **1**, 1–6 (2006).

[5] Mallon, J. and Whelan, P., "Precise radial un-distortion of images," in [*Proceedings of the 17th International Conference on Pattern Recognition*], *ICPR 2004* **1**, 18–21 (2004).

[6] de Villiers, J., Leuschner, F., and Geldenhuys, R., "Centi-pixel accurate real-time inverse distortion correction," in [*Proceedings of the 2008 International Symposium on Optomechatronic Technologies*], *ISOT2008* **7266**, 1–8 (2008).

[7] Negnevitsky, M., [*Artificial Intelligence, A guide to intelligent systems*], Pearson Education, Harlow, England (2002).

[8] Rosenblatt, F., "Perceptron simulation experiments," in [*Proceedings of the institute of radio engineers*], **48**, 301–309 (1958).

[9] Bryson, A. and Ho, Y., [*Applied optimal control*], Blaisdell, New York, USA (1969).

[10] Egmont-Peterson, M., de Ridder, D., and H.Handels, "Image processing with neural networks - a review," *Pattern Recognition* **35**, 2279–2301 (2002).

[11] Memon, Q. and Khan, S., "Camera calibration and three-dimensional world reconstruction of stereo-vision using neural networks," *International Journal of Systems Science* **32**(9), 1155–1159 (2001).

[12] Do, Y., "Application of neural networks for stereo-camera calibration," in [*Neural Networks, 1999. IJCNN '99. International Joint Conference on*], **4**, 2719 –2722 vol.4 (1999).

[13] Ahmed, M., Hemayed, E., and Farag, A., "Neurocalibration: A neural network that can tell camera calibration parameters," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **79**, 384–390 (1999).

[14] de Villiers, J. and Nicolls, F., "Application of neural networks to inverse lens distortion modelling," in [*Proceedings of the 21st Annual Symposium of the Pattern Recognition Society of South Africa*], *PRASA2010* **1**, 63–68 (2010).

[15] Lucchese, L. and Mira, S., "Using saddle points for subpixel feature detection in camera calibration targets," in [*Proceedings of the Asia-Pacific Conference on Circuits and Systems*], **2**, 191–195 (2002).

[16] Snyman, J., "An improved version of the original leap-frog dynamic method for unconstrained minimization: Lfop1(b)," *Applied Mathematics and Modelling* **7**, 216–218 (1983).