Neurocalibration: A Neural Network That Can Tell Camera Calibration Parameters*

Moumen T. Ahmed, ElSayed E. Hemayed and Aly A. Farag
CVIP Lab. University of Louisville
Louisville, KY 40292, USA
E-mail:(moumen,sayed,farag)@cairo.spd.louisville.edu

Abstract

Camera calibration is a primary crucial step in many computer vision tasks. In this paper we present a new neural approach for camera calibration. Unlike some existing neural approaches, our calibrating network can tell the perspective-projection-transformation matrix between the world 3D points and the corresponding 2D image pixels. Starting from random initial weights, the net can specify the camera model parameters satisfying the orthogonality constraints on the rotational transformation. The neurocalibration technique is shown to solve four different types of calibration problems that are found in computer vision applications. Moreover, it can be extended to the more difficult problem of calibrating cameras with automated active lenses. The validity and performance of our technique are tested with both synthetic data under different noise conditions and with real images. Experiments have shown the accuracy and the efficiency of our neurocalibration technique.

1 Introduction

Artificial neural networks have been used to solve some computer vision problems [11] such as static stereo, lateral motion stereo, longitudinal motion stereo, computation of optical flow, and image restoration. In this paper, we investigate applying a neural approach to another problem of computer vision, camera calibration. Basically, there are two aspects associated with the problem of camera calibration: calibration of the internal parameters of a camera (*intrinsic* parameters) and pose estimation of a camera system relative to a 3D world reference system (*extrinsic* parameters). The existing techniques to solve this problem can be broadly classified into three main categories: linear, nonlinear (or iterative), and two-step methods. The interested reader is referred to [3], [2] for a complete review.

We present how to map the camera calibration problem into a learning problem of a multi-layer feed-forward neural network (MLFN). Inspired by the previous success of neural nets in solving some vision problems [11], our motivations for doing so are to investigate if such a net would:

- relax the requirement to start with a good initial guess which is required by other non-linear camera calibration techniques [1],[2],[8],
- allow using a neural-based framework for the more difficult problem of calibrating cameras with automated active lenses [10].

Some neural net-based calibration techniques have been reported in the literature [7],[5],[4]. These techniques used neural networks either to learn the mapping from 3D world to 2D images without specifying the camera models [5], or as an additional stage to improve the performance of other existing techniques [4],[7]. Since knowing the camera model parameters feeds very important information to following vision tasks (e.g., stereo-reconstruction), our technique goes beyond the existing methods by having the neural network specify the camera intrinsic and extrinsic parameters. We also demonstrate the ability of our calibrating network to solve four different types of calibration problems that are found in computer vision applications [6]:

- *Type 1*: to estimate all the camera parameters simultaneously.
- *Type 2*: to estimate all the other camera parameters given the image center.
- *Type 3*: to estimate the extrinsic parameters given the intrinsic parameters.
- *Type 4*: to estimate the intrinsic parameters given the extrinsic parameters.

*This research was partially supported by grants from NSF (ECS 9505674) and The Department of Defense (USNV N00014-97-11076).

We present a camera model and state the calibration problem in the next section. In Section 3, we introduce our calibrating neural network. Section 4 demonstrates how to solve the four types of calibration problems and to calibrate a camera with an active lens. The performance of our method is evaluated by the results of simulations and experiments with real camera systems in Section 5. Finally, Section 6 gives our concluding remarks.

2 Camera Model

The camera model that we consider is the perspective projection model based on the pinhole model [1]. If M has world coordinates (X, Y, Z) and projects onto a point m that has pixel coordinates (u, v), the operation can be described, in homogeneous coordinates, by the equation:

$$S \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \mathbf{P} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \quad (1)$$

where S is a scaling factor and the matrix $\mathbf{P}$ is in the format

$$\mathbf{P} = \begin{pmatrix} \mathbf{p}_1^\top & p_{14} \\ \mathbf{p}_2^\top & p_{24} \\ \mathbf{p}_3^\top & p_{34} \end{pmatrix}. \quad (2)$$

The 3×4 matrix $\mathbf{P}$ is commonly referred to as *perspective projection matrix* and decomposed into two matrices: $\mathbf{P} = \mathbf{A}\ \mathbf{D}$ where

$$\mathbf{D} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_3^\top & 1 \end{pmatrix} \quad \mathbf{A} = \begin{pmatrix} \alpha_u & -\alpha_u cot\theta & u_0 & 0 \\ 0 & \alpha_v / sin\theta & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

The 4×4 matrix $\mathbf{D}$ represents the mapping from world coordinates to camera coordinates and accounts for six extrinsic parameters of the camera: three for the rotation $\mathbf{R}$ which is normally specified by three rotation (*Euler*) angles: R_x, R_y and R_z and three for the translation $\mathbf{t} = (t_x, t_y, t_z)^\top$. $\mathbf{0}_3$ represents the null vector $(0, 0, 0)^\top$. The 3×4 matrix $\mathbf{A}$ represents the intrinsic parameters of the camera: the scale factors α_u and α_v, the coordinates u_0 and v_0 of the principal point, and the angle θ between the image axes.

This camera model ignores lens distortion which is often accounted for in the camera model by adding some distortion coefficients [3],[2]. This can be overcome by estimating the distortion coefficients then correcting for the lens distortion in the captured images even before calibration proceeds [9]. The benefit from this would be twofold. The calibration accuracy will not only be increased, but this will also allow us to maintain the simple relation in (1) thus making following vision tasks easier.

With this camera model, the calibration problem can be stated as follows: Given a sufficient number N of reference points whose world coordinates $M_i(X_i, Y_i, Z_i)$ are known with a high precision, as well as their corresponding observed pixel positions $m_i(u_i, v_i)$, estimate the 11 camera parameters that project the 3D points to these 2D image points. Accordingly, from (1) after eliminating the scale factor S which is different for each point, the error criterion E to be minimized is

$$E = \sum_{i=1}^{N} \left(\frac{\mathbf{p}_1^\top M_i + p_{14}}{\mathbf{p}_3^\top M_i + p_{34}} - u_i\right)^2 + \left(\frac{\mathbf{p}_2^\top M_i + p_{24}}{\mathbf{p}_3^\top M_i + p_{34}} - v_i\right)^2 \quad (3)$$

3 Neurocalibration

Our interest is to employ a neural network not only to learn the mapping from 3D points to 2D pixel points which minimizes the error in (3), but also to extract the projection matrix and camera parameters. Therefore, the network structure is laid out accordingly. The net is a two-layer feedforward neural network. The input has three neurons plus one augmented fixed at 1. These three correspond to the three coordinates X, Y, Z of a 3D point. The number of output units is three, and the hidden layer consists of four neurons (three plus one dummy). The hidden and output neurons have unity activation functions. The weight matrix of the hidden layer is denoted by $\mathbf{V}$, and it is assumed to correspond to the extrinsic parameters matrix $\mathbf{D}$. The weight matrix of the output layer is denoted $\mathbf{W}$ and corresponds to the intrinsic parameters matrix $\mathbf{A}$. For any input pattern $i, N \geq i \geq 1$, the input vector is formed as $\mathbf{Z}_i = (X_i\ Y_i\ Z_i\ 1)^\top$, and the outputs are $o_{ik}, k = 1, 2, 3$. With a literal interpretation of the error criterion in (3), the error measure here would be

$$E = \sum_{i=1}^{N} \left(\frac{o_{i1}}{o_{i3}} - u_i\right)^2 + \left(\frac{o_{i2}}{o_{i3}} - v_i\right)^2 \quad (4)$$

This last equation is not in a handy form than can be used by a gradient descent error-driven network due to the presence of the two ratios in terms of the network outputs. To tackle this problem, one might be tempted to train the net such that the desired value of o_{i3} (which yet to be figured out) to 1 thus getting rid of the unwanted ratios. Unfortunately, this will do us little good because, from (3), this means that $\mathbf{p}_3^\top M_i + p_{34} = 1$ for all i. Clearly, no values of $\mathbf{p}_3$ and p_{34} will hold this true for all M_i resulting in no convergence of the network. A good solution is to train the net letting $\gamma_i o_{i3}$ approach 1, so we can write E in (4) as

$$E = \sum_{i=1}^{N} (\gamma_i o_{i1} - u_i)^2 + (\gamma_i o_{i2} - v_i)^2 + (\gamma_i o_{i3} - 1)^2 \quad (5)$$

Each input pattern will have its own factor γ_i. The question now is what this value should be. This factor γ_i is initially set to 1 for all i then allowed to update according to gradient descent rule to minimize the error measure in (5). Based on this, the desired vector for each pattern, $\mathbf{d}_i$, is set to $(u_i\ v_i\ 1)^\top$. Letting $\mathbf{O}_i = \gamma_i \mathbf{o}_i$, the dynamics of the network follows equations (6, 7, 8) as below. Given input pattern Z_i, hidden neuron j produces output

$$Y_j = \sum_{l=1}^{4} V_{jl} Z_{il} \quad (6)$$

While an output neuron k produces

$$O_{ik} = \sum_{j=1}^{4} \gamma_i W_{kj} Y_j \quad (7)$$

And the error measure for the ith input point from (5), is now cast in the familiar form,

$$E_i = \frac{1}{2} \sum_{k=1}^{3} (d_{ik} - O_{ik})^2 \quad (8)$$

where a factor of 0.5 is introduced here for convention only and d_{ik} designates the k-th component of the desired vector $\mathbf{d}_i$. The weights of the network W_{kj} and V_{lj} are initialized at random values in the range $-1 : 1$. Since the fourth hidden neuron's output is fixed at 1, its corresponding weights are fixed, $V_{l4} = 0, l = 1, 2, 3, V_{44} = 1$. The network is trained with the error back-propagation algorithm updating the weights W_{kj}, V_{lj} and γ_i according to the gradient descent rule as follows:

$$\Delta W_{kj} = \alpha_1 (d_{ik} - O_{ik}) \gamma_i Y_j \quad (9)$$

$$\Delta V_{lj} = \alpha_1 \gamma_i \sum_{k=1}^{3} (d_{ik} - O_{ik}) W_{kj} Z_{il} \quad (10)$$

$$\Delta \gamma_i = \alpha_2 \sum_{k=1}^{3} (d_{ik} - O_{ik}) \sum_{j=1}^{4} W_{kj} Y_j \quad (11)$$

where α_1 and α_2 are positive learning constants. For ease of network learning, it is advised that the input and desired patterns of the network be normalized. Therefore the two sets, 2D pixel coordinates and 3D point coordinates, are normalized by s_1 and s_2, respectively. After training the network, the projection matrix $\mathbf{P}$ can be shown equal to

$$\mathbf{P} = \mathbf{S}_1\ \mathbf{W}\ \mathbf{V}\ \mathbf{S}_2 \quad (12)$$

where
$\mathbf{S}_1 = \mathbf{diag}(s_1, s_1, 1)$ and $\mathbf{S}_2 = \mathbf{diag}(s_2^{-1}, s_2^{-1}, s_2^{-1}, 1)$.

4 More Aspects of Neurocalibration

This section maps the network weights to camera parameters, then shows how to use the neurocalibration technique for calibrating cameras with automated active lenses.

4.1 Four Calibration Problems

Taking into account the scaling factors s_1 and s_2, the matrix $\mathbf{D}$ will be given by $\mathbf{V}\ \mathbf{S}_2$ while matrix $\mathbf{A}$ is equal to $\mathbf{S}_1\ \mathbf{W}$. This would be completely correct if the orthogonality constraints on $\mathbf{R}$ are met. $\mathbf{R}$ as a rotation matrix should satisfy

$$\mathbf{r}_i^\top \mathbf{r}_j = \delta_{ij};\ i, j = 1, 2, 3,\ i \leq j \quad (13)$$

where $\mathbf{r}_j$ is the jth row of $\mathbf{R}$ and δ_{ij} is the usual Kronecker function. The relations in (13) form a set of six independent quadratic constraints that are to be satisfied by the elements of the orthogonal matrix $\mathbf{R}$. The neurocalibration approach is flexible enough to accommodate these constraints. The constraints are represented as additional terms added to the error criterion to be minimized. The new error measure per input will be

$$E_{tot} = E_{2D} + \beta E_{orth} \quad (14)$$

where E_{2D} is the same in (8) and

$$\begin{aligned} E_{orth} = & \sum_{l=1}^{3} (V_{l1}^2 + V_{l2}^2 + V_{l3}^2 - a)^2 \\ & + (V_{11}V_{21} + V_{12}V_{22} + V_{13}V_{23})^2 \\ & + (V_{11}V_{31} + V_{12}V_{32} + V_{13}V_{33})^2 \\ & + (V_{21}V_{31} + V_{22}V_{32} + V_{23}V_{33})^2 \end{aligned}$$

In (14), β is a small positive weighting factor while constant a (originally should be 1) accounts for any scaling in the $\mathbf{V}$ weights. The updating rules, equations (9), (10) and (11), for the various network weights are the same, except for the weights $V_{lj}, 3 \geq$

$l, j \geq 1$ which are determined by the gradient descent rule as

$$\begin{aligned} \Delta V_{lj} = \alpha_1 \gamma_i \sum_{k=1}^{3} (d_{ik} - O_{ik}) W_{jk} Z_{il} \\ -4\alpha_1 \beta \ (V_{l1}^2 + V_{l2}^2 + V_{l3}^2 - a) V_{lj} \\ -2\alpha_1 \beta \ \delta v_{lj} \end{aligned} \quad (15)$$

where

$$\delta v_{lj} = \begin{cases} (V_{2j} + V_{3j})\{(V_{11}V_{21} + V_{12}V_{22} + V_{13}V_{23}) \\ \quad +(V_{11}V_{31} + V_{12}V_{32} + V_{13}V_{33})\}, l = 1 \\ (V_{1j} + V_{3j})\{(V_{11}V_{21} + V_{12}V_{22} + V_{13}V_{23}) \\ \quad +(V_{21}V_{31} + V_{22}V_{32} + V_{23}V_{33})\}, l = 2 \\ (V_{1j} + V_{2j})\{(V_{11}V_{31} + V_{12}V_{32} + V_{13}V_{33}) \\ \quad +(V_{21}V_{31} + V_{22}V_{32} + V_{23}V_{33})\}, l = 3 \end{cases}$$

Moreover, some elements of the $\mathbf{A}$ matrix have values of zero or one. Therefore the corresponding weights of $\mathbf{W}$ are fixed accordingly. Now, one can use this approach to solve four different types of calibration problems [6]. Since all the camera parameters in *Type 1* are to be estimated, this is exactly what our network in Section 3 accomplishes. The projection matrix $\mathbf{P}$ is calculated from (12) and can be decomposed into the camera parameters [1]. In *Type 4* calibration problems, the extrinsic parameters are assumed to be known, therefore the $\mathbf{V}$ matrix weights are initialized to $\mathbf{D}\mathbf{S}_2^{-1}$ with appropriate scaling if necessary. Only the five $\mathbf{W}$ weights (corresponding to the 5 intrinsic parameters) are updated according to the gradient descent rule in (9).

On calibrating a camera, the image center can be possibly estimated first using another independent calibration process such as the autocollimated laser technique [10], then the remaining camera parameters are calibrated. This occurs in *Type 2* problems, whereas problems of *Type 3* appear when the camera is moved thus changing the extrinsic parameters while keeping the intrinsic parameters unchanged. Therefore the extrinsic parameters only need to be re-calibrated. To solve *Type 2* problems, W_{13} and W_{23} are initialized to $s_1^{-1}u_0$ and $s_1^{-1}v_0$, respectively, while the other network weights are randomly initialized. On the other hand, the $\mathbf{W}$ weight matrix, in *Type 3*, is initially taken $\mathbf{W} = \mathbf{S}_1^{-1}\mathbf{A}$. In either problem type, only the network weights corresponding to the ungiven parameters are updated according to the proper rules with the orthogonality constraints taken care of by (15).

4.2 Calibrating Cameras with Active Lenses

Cameras with automated active lenses become more and more important in vision systems due to their flexibility and controllability [8],[10]. However, calibrating such cameras raises several challenges [10],[8]. The calibration problem becomes how to characterize how the parameters of the fixed camera model vary with lens settings [10]. The calibration approach, generally, involves first calibrating a conventional static camera model at a number of lens settings spanning the lens' control space. To model how the terms of the static camera model vary with lens setting, partial lookup tables and interpolations [8] or fitting multi-variable polynomials [10] can be used.

Our calibrating neural net, with each camera model parameter mapped to a network weight, allows us to present an all-neural approach for this problem. This approach can capture complex variations in camera parameters. While the details of such an approach will not be described here due to lack of space, the overall approach has two main steps. First, some model parameters, namely $\alpha_u, \alpha_v, u_0, v_0$, and t_z, are approximated by multi-variable functions using 5 MLFNs with the proper network topology selected for each parameter. The other parameters are modeled by zero-order terms. With the calibrating net being the basis of the second step of global optimization, these 5 MLFNs serve to provide values for the corresponding network weights. The main calibrating net is trained minimizing the error in (14) over all the collected calibration data. After one set of data at the same optical setting updates the main network weights, the 5 MLFNs will have one training iteration to learn the updated network weights (which hence become the desired values for them) and so on. As the six neural networks learn concurrently and cooperatively, the variations of the camera model are captured across the calibrated ranges of lens parameters. This step takes into account the interaction and the correlation [6] between the model parameters (which is absent during fitting a function to each parameter model alone). One key feature of this approach is that all of the parameters are fitted to the calibration data at the same time, while in the polynomial-based approach [10] one parameter is fitted at a time and the final level of error generally depends on the order in which the models are fit to the data [10]. The all-neural approach has been used successfully to calibrate Hitachi KP-M1 CCD cameras with H10x11E Fujinon active lenses in our lab.

5 Experimental Results

The estimation of camera model parameters presented in the previous sections has been tested with synthetic data and with real images. Due to lack of space, we present here the results of *Type 1* calibration problems only, while more extensive results are

Parameter	ground truth	calibrated
t_x(mm)	-27.0	-26.98
t_y(mm)	-28.0	-28.00
t_z(mm)	701.0	701.00
R_x(rad)	0.09	0.09000
R_y(rad)	0.80	0.799973
R_z(rad)	-0.03	-0.03001
α_u(pix)	556.0	556.001165
α_v(pix)	549.0	549.000605
u_0(pix)	172.0	171.982447
v_0(pix)	121.0	120.998733
θ(rad)	1.5708	1.5708

Table 1: Ground truth versus noise-free calibrated parameters.

provided in a manuscript under review [12].

5.1 Simulations with Synthetic Data

After choosing specific external and internal camera parameters, which are referred to later as ground truth, the simulation proceeds as follows: given a set of 3D points ($N = 440$) and the pre-specified camera parameters, we compute the projected 2D image points. Then, some pseudo-noise v with Gaussian distribution $g(0, \sigma^2)$ is added to the 2D image coordinates to represent the uncertainty in detecting these 2D points. Using these 2D points to calibrate the camera parameters, we have performed a series of experiments using different σ, each repeated 50 times. After training the net, the projection matrix, $\mathbf{P}$, is computed from (12) and decomposed into the 11 parameters as described in [1]. The actual ground truth values and the resultant noise-free parameters after neurocalibration are shown in Table 1. As shown in this table, the calibrating network could tell the right values for the camera parameters. For comparison's sake, the camera parameters for the above experiments are estimated using the linear calibration approach [1] and a nonlinear technique by minimizing the error in (3) using the well-known Levenberg-Marquardt algorithm. This algorithm starts with the output of the linear approach as an initial estimate of the parameters. The *root mean square error* of some parameters computed over the 50 trials are plotted in Fig. 1 as a function of σ using the three techniques. As expected, the linear approach is the least robust against noise while the other two techniques have fairly similar curve trends. This is logical since both techniques optimize a squared error metric so they have nearly similar robustness to noise. However, in the above experiments the neurocalibration approach starts with a random initial state while the Levenberg-Marquardt algorithm would fail without a good starting point.

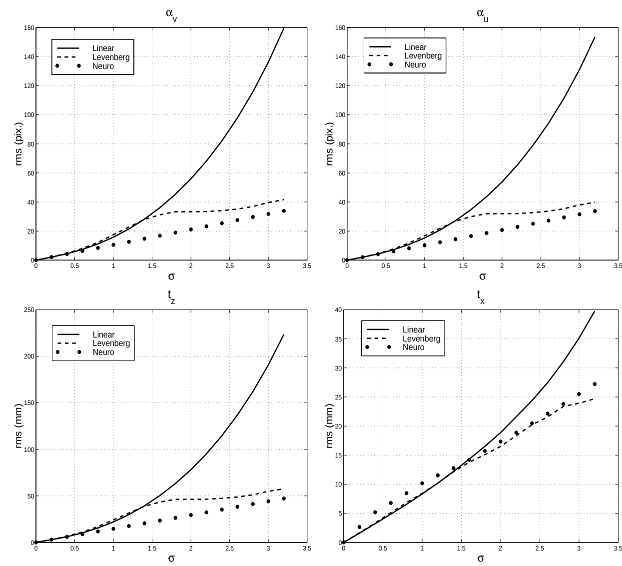

Figure 1: How the *rms* of some calibrated parameters (α_v, α_u, t_z and t_x) varies with the pixel noise σ for three approaches: linear, nonlinear using Levenberg-Marquardt algorithm and neurocalibration.

5.2 Experiments with Real Images

With real images, the accuracy of the calibration is measured in terms of the accuracy in reconstructing 3D points through triangulation [2],[3]. Therefore, calibration is done for two cameras. Two images were captured for a chessboard-like calibration pattern by a pair of CCD cameras (6.1mm HA6300 TeleCamera). The 3D points used for calibration are the mid-points between the vertices of the board squares. Knowing the spatial positions (X_i Y_i Z_i) of these points, the corresponding image-point locations in the two images are estimated with sub-pixel accuracy using a technique based on edge-detection and fitting [2],[3]. The two sets, 3D points and corresponding 2D pixels are used for calibrating the two cameras as *Type 1* problem.

To assess the calibration accuracy, two error measures have been used. The first is the usual *root mean squared calibration error* (RMSE), while the other is suggested by Weng *et. al.* [2] and called *normalized stereo camera error* (NSCE). Weng claims that through this error measure, the performance of different calibration approaches can be quantitively evaluated and compared. The two above error measures have been computed using the neurocalibration approach and by Tsai's algorithm. The reason to compare our results to Tsai's is that the latter is a very popular calibration technique used in many computer vision labs around the world. Clearly, the two measures shown in Table 2 are in favor of the neurocalibration technique.

Measure	Neuro	Tsai's
RMSE	0.092211	1.20885
NSCE	0.022978	1.671617

Table 2: Calibration error measures for our experiment.

More experiments have been conducted to test the sensitivity of the network learning to network parameters, α_1 and α_2. Under the same predefined set of these parameters, the net was tested with 70 data sets collected for calibrating the CardEye, a trinocuclar active-vision system developed at our lab. The data had great variations in the spatial positions of the 3D reference points and in camera parameters (due to changing the focus and zoom settings of the active lenses). For all the data sets, the network started with random weights and could converge yielding the correct camera parameters.

6 Summary and Conclusions

We have presented a new neural network approach to the problem of camera calibration. We demonstrated how four different types of calibration problems can be addressed and described in brief how to use this approach for calibration of cameras with active lenses. Our experimental results have demonstrated the high performance of our approach with both synthetic and real data. We believe that this approach has the following features:

- unlike some other techniques [7],[5],[4] found in the literature that addresses the same problem using a neural network, ours is able to specify the calibration parameters of the camera model.
- it relaxes the requirement of a good initial guess to proceed; in all the experiments conducted, the network has converged starting from random initial weights.
- the orthogonality constraints on the rotation matrix are satisfied in the obtained parameters, while some other techniques use a separate step to impose those [2].
- most importantly, this approach allows employing an all-neural approach for calibrating cameras with active lenses, in which a number of MLFNs learn concurrently the variations of model parameters across optical lens settings.

References

[1] Olivier Faugeras, "Three-dimensional computer vision: a geometric viewpoint," *MIT Press*, 1993.

[2] J. Weng, Paul Cohen and M. Herniou, "Camera calibration with distortion models and accuracy evaluation," *IEEE Trans. Patt. Anal. Machine Intell.*, Vol. 14, No. 10, Oct 1992.

[3] Roger Tsai, "A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses," *IEEE Journal of Robotics and Automation*, Vol. RA-3, No. 4, Aug 1987.

[4] J. Wen and G. Schweitzer, "Hybrid calibration of CCD cameras using artificial neural nets," *Proc. Int Joint Conf. Neural Networks (New York)*, Vol. 1, Nov. 1991.

[5] M. Lynch and C. Dagli, "Backpropagation neural network for stereoscopic vision calibration," *Proc. SPIE. Int. Soc. Opt. Eng.*, Vol. 1615, Nov. 1991.

[6] S. Shih, Y. Hung and W. Lin, "Accuracy analysis on the estimation of camera parameters for active vision systems," *Proc. Int. Conf. Pattern Recognition (Vienna, Austria)*, Vol. 1, Aug 1996.

[7] D. Choi, S. Oh, H. Chang and K. Kim, "Nonlinear camera calibration using neural networks," *Neural, parallel and scientific computations*, Vol. 2, No. 1, March 1994.

[8] Mengxiang Li and Jean-Marc Lavest, "Some aspects of zoom lens camera calibration," *IEEE Trans. Patt. Anal. and Machine Intell.*, Vol. 18, Nov. 1996.

[9] T. Pajdla, T. Werner and V. Hlavac, "Correcting radial lens distortion without knowledge of 3D structure," *Technical Report, Machine Perception Center, Czech Technical University*, Dec. 1997.

[10] R. G. Willson, "Modeling and calibration of automated zoom lenses", *PhD dissertation, Dept. Elect. Comp. Eng., Carnegie Mellon Univ.*, 1994.

[11] Y. Zhou and R. Chellappa, "Artificial neural networks for computer vision", *Springer-Verlag*, 1992.

[12] Moumen Ahmed, ElSayed Hemayed, Aly Farag and Jacek Zurada, "Neurocalibration: a neural approach for the problem of camera calibration", *submitted to IEEE Trans. Patt. Anal. and Machine Intell.*, Dec. 98.

0-7695-0164-8/99 $10.00 (c) 1999 IEEE