A Learning Mechanism for Parts Recognition in an Intelligent Assembly System

Paul S. Y. Wu* and Xiong Ying'en†

*Department of Manufacturing Engineering, City University of Hong Kong, Kowloon, Hong Kong; and †Electronics Department, Zhongshan University of China

In this paper, a learning mechanism (LM) for parts recognition in an intelligent assembly system is presented. It differs from the mechanism used in the standard back propagation (SBP) neural network in two ways. First, the searching direction is changed from the negative gradient direction to the variable metric direction. Secondly, the constant learning rate is changed to a variable optimal learning rate. The combination of these two improvements leads to the training process being greatly speeded up, and convergence is assured. Several application examples are presented. Results indicate that the proposed LM is superior to the SBP in learning speed, convergence and stability.

Keywords: Intelligent assembly system; Learning mechanism; Neural networks; Variable metric direction; Variable optimal learning rate

1. Introduction

In an intelligent assembly environment, a vision system for parts identification and recognition is required to be fast and robust [1]. The system must perform accurate and unambiguous parts recognition in a normal noise environment and meet the requirements of continuous operation at high speed.

In an intelligent assembly system, the parts recognition process requires the reconstruction of a 3D object model from 2D images [2]. Conventional approaches require an elaborate system calibration to determine the relationships between the two cameras and the global coordinate system [3,4]. To minimise the system error, an iterating process must be employed for the parts recognition. This is a time-consuming process and not suitable for intelligent assembly operations.

As in other fields, a neural network can provide intelligent 3D parts recognition. Its learning and parallel processing abilities [5] provide fast recognition speed and continuous robust operation. However, a neural network has its problems [6–8] also. During the training process, it may take considerable time or even exponential time for learning, and the convergence/stability cannot always be guaranteed. Lack of robustness is another problem. To take full advantage of a neural network in an intelligent assembly system, a more effective learning method must be developed.

In this paper, a learning mechanism (LM) for parts recognition is proposed. It is based on the general philosophy of the SBP approach. The error between the desired and actual outputs is taken as the objective function. By adjusting the weighting values of neurones in the neural network, the objective function is iteratively minimised. The proposed LM improves the learning speed and convergence by using an optimal learning rate (instead of a constant learning rate) in a variable metric direction (instead of the negative gradient direction) in each iteration to minimise the objective function.

Some application examples are used to illustrate the reliability of the new learning approach. The first example is to learn a nonlinear system in which the mathematical model, structure and other characteristics are unknown except for its input and desired outputs. The aim of this example is to train the neural network to learn the characteristics of the nonlinear system by using training data. A higher accuracy with less training time is obtained by using the LM than by the SBP. The second example is related to the 3D recognition process. A four-layer back-propagation neural network (3-20-20-1) is constructed to estimate the epipolar lines between the left- and right-hand images of a stereo image pair. It is used to learn the geometric relationship between the left- and right-hand images by using training data to estimate epipolar lines. The proposed LM provides better accuracy than the SBP for the same amount of training time and provides a much shorter training time for the same accuracy. In the last example, the proposed LM is used for the 2D to 3D points conversion process. A four-layer back propagation neural network (4-50-50-3) is constructed to learn the relationship between the two cameras and the global coordinate system, and to convert the 2D points of stereo image pairs into 3D points. The same conclusion is obtained that the LM can obtain better result with less training time than the SBP.

Correspondence and offprint requests to: Dr P. S. Y. Wu, Department of Manufacturing Engineering, City University of Hong Kong, 83 Tat Chee Avenue, Kowloon, Hong Kong.
2. The Basic Learning Algorithm

For a general three-layer neural network \((n-m-N)\) as shown in Fig. 1, the inputs are \(x_i (k = 1, 2, \ldots, n)\), the actual outputs are \(y_j (j = 1, 2, \ldots, N)\), and the desired outputs are \(d_j (j = 1, 2, \ldots, N)\). The error \(E\) between the desired \(d_j\) and actual \(y_j\) outputs is defined as:

\[
E = \frac{1}{2} \sum_{j=1}^{N} (y_j - d_j)^2
\]  
(1)

Equation (1) is the objective function where \(y_j\) is a function of weights \((W_{ij}, W_{ik})\). By adjusting the weights in the designed neural network, the error function \(E\) is expected to be minimised.

It is reasonable to assume that the input and output of a neuron are sigmoidal functions,

\[
y = f(x) = \frac{1}{1 + \exp(-x)}
\]
(2)

the inputs to each neuron in the output layer and the hidden layer of an ideal SBP are

\[
A_i(j) = \sum_{k=1}^{m} W_{ij}x_k; \quad A_i(i) = \sum_{k=1}^{n} W_{ik}x_k
\]
(3)

where \(W_{ij}(i = 1, 2, \ldots, N; j = 1, 2, \ldots, m)\) and \(W_{ik}(i = 1, 2, \ldots, m; k = 1, 2, \ldots, n)\) are weighting values between output and hidden layers, and between hidden and input layers, respectively. The gradients between the output and hidden layers, and the hidden and input layers are \(\frac{\partial E}{\partial W_{ij}}\) and \(\frac{\partial E}{\partial W_{ik}}\), respectively. They represent the negative gradient directions of minimisation as follows:

\[
[S_{ij}] = -\left[ \frac{\partial E}{\partial W_{ij}} \right]; \quad [S_{ik}] = -\left[ \frac{\partial E}{\partial W_{ik}} \right]
\]
(4)

\(i = 1, 2, \ldots, m; \quad j = 1, 2, \ldots, N\); \(i = 1, 2, \ldots, m; \quad k = 1, 2, \ldots, n\).

These two gradients are related in the expression as

\[
\frac{\partial E}{\partial W_{ik}} = \frac{\partial E}{\partial W_{ij}} W_{ij} (1 - y_j) y_k \quad (y_k = x_k, \quad k = 1, 2, \ldots, n)
\]
(5)

The above weighting vectors and direction vectors can be re-defined as

\[
\begin{bmatrix}
W_{ij} \\
S_{ij}
\end{bmatrix} = \begin{bmatrix}
W_{ik} \\
S_{ik}
\end{bmatrix}
\]
(6)

and combined into a new weighting vector of the network system and a new directional vector which updates the weight vector as

\[
W = \begin{bmatrix}
W_{ij} \\
W_{ik}
\end{bmatrix}; \quad S = \begin{bmatrix}
S_{ij} \\
S_{ik}
\end{bmatrix}
\]
(7)

\((i = 1, 2, \ldots, N; \quad i = 1, 2, \ldots, m); \quad (i = 1, 2, \ldots, m; \quad j = 1, 2, \ldots, N)\).

The process of updating can be expressed as

\[
W^{l+1} = W^l + \lambda S^l
\]
(8)

that is

\[
\begin{bmatrix}
W_{ij}^{l+1} \\
W_{ik}^{l+1}
\end{bmatrix} = \begin{bmatrix}
W_{ij}^l \\
W_{ik}^l
\end{bmatrix} + \lambda \begin{bmatrix}
S_{ij}^l \\
S_{ik}^l
\end{bmatrix}
\]
\((i = 1, 2, \ldots, m; \quad j = 1, 2, \ldots, N)\)

where \(\lambda\) is a constant learning rate and \(l\) is the number of iterations.

3. Techniques for New Optimal Direction and Step Size

The equations developed above for the SBP use the steepest descent method to minimise the error function \(E\). The use of a negative gradient direction and a constant step size (learning rate) are the main reasons why the objective function converges very slowly and the convergence/stability is not always guaranteed. To overcome these well-known problems associated with the SBP methodology, a method of finding the optimal direction and the optimal step size for each searching iteration is proposed.

3.1 A New Direction of Minimisation

Instead of using a negative gradient as expressed by equation (4) and a new directional vector by equation (7), we construct a variable metric direction by denoting the gradients as

\[
\nabla E_f = \begin{bmatrix}
\frac{\partial E_f}{\partial W_{ij}} \\
\frac{\partial E_f}{\partial W_{ik}}
\end{bmatrix}; \quad \nabla E_f = \begin{bmatrix}
\frac{\partial E_f}{\partial W_{ik}} \\
\frac{\partial E_f}{\partial W_{ik}}
\end{bmatrix}
\]
(9)

\((i = 1, 2, \ldots, m; \quad k = 1, 2, \ldots, n; \quad j = 1, 2, \ldots, N)\)

and then define the direction as

\[
S^l = \begin{bmatrix}
S_{ij}^l \\
S_{ik}^l
\end{bmatrix} = -H E_f; \quad \nabla E_f = \begin{bmatrix}
\nabla E_f^l \\
\nabla E_f^l
\end{bmatrix}
\]
(10)

where \(l\) is the iteration number, \(S^l\) is the directional vector, and \(H^l\) is the variable metric matrix. Once the matrix \(H^l\) is
calculated, the process of weighting vector updating can be carried out in the same way as
\[
W^{s+1} = W^s + \alpha S^s; \quad \begin{bmatrix} W_{f+1}^s \\ W_{l+1}^s \end{bmatrix} = \begin{bmatrix} W_f \\ W_l \end{bmatrix} + \alpha \begin{bmatrix} S_f \\ S_l \end{bmatrix}
\]
(11)
where \( \alpha \) is the optimal learning rate. The development of \( H \) is shown in Appendix A.

### 3.2 Techniques for Learning Rate

The learning rate influences the learning speed of arriving at optimal weighting values for the neural network. An optimal learning rate \( \alpha \) will minimise the objective function \( E \) and obtain the optimal weighting values faster. The objective function for the system is

\[
E^{s+1} = \frac{1}{2} \sum_{j=1}^{N} E_{j}^{s+1} (\alpha') = \frac{1}{2} \sum_{j=1}^{N} (y_{j}^{s+1} (\alpha') - d_{j})^2
\]
(12)

In order to find optimal step length \( \alpha \), we minimise the objective function \( E^{s+1} (\alpha') \). That is

\[
\min \frac{1}{2} \sum_{j=1}^{N} \left( \frac{1}{1 + \exp \left( -\sum_{i=1}^{m} \frac{W_{f_i} + \alpha'S_{f_i}}{1 + \exp \left( -\sum_{k=1}^{m} (W_{l_k} + \alpha'S_{l_k})x_k \right)} \right)} - d_j \right)^2
\]
(13)

The development of equations (12) and (13) is shown in Appendix B. It is difficult to minimise this complicated objective function analytically. A numerical method is used to find the optimal step length \( \alpha \). Since this is a one-dimensional minimisation problem, the direct search method called the Golden Section method [9] can be used to solve the problem.

### 4. Procedure of the Learning Algorithm

We can state the learning algorithm as follows:

Step 1. Input the inputs \( x_k \) (\( k = 1, 2, \ldots, n \)) and the desired outputs \( d_j \) (\( j = 1, 2, \ldots, N \)).

Step 2. Input the initial weighting values \( W^0 \).

Step 3. Input a positive definite symmetric matrix \( H^0 \). Usually, \( H^0 \) is taken as the identity matrix \( I \). Set the iteration number as \( l = 1 \).

Step 4. Calculate
\[
\frac{\partial E}{\partial W_{f_i}}; \quad \frac{\partial E}{\partial W_{l_k}}; \quad \nabla E^f = \begin{bmatrix} \frac{\partial E}{\partial W_{f_i}} \\ \frac{\partial E}{\partial W_{l_k}} \end{bmatrix}, \quad \nabla E_l = \begin{bmatrix} \frac{\partial E}{\partial W_{l_k}} \end{bmatrix}
\]

Step 5. Set \( S^l = -H^l \nabla E^l \), \( S^l = \begin{bmatrix} S_f \\ S_l \end{bmatrix} \)

Step 6. Solve the minimisation problem: \( \min E^{s+1} (\alpha') \) and find the optimal step length \( \alpha \) and direction \( S^l \).

Step 7. Update the weighting values: \( W^{s+1} = W^s + \alpha S^l \).

Step 8. Test the new points \( W_{f_i}^{s+1}, W_{l_k}^{s+1} \) for optimality. If \( W_{f_i}^{s+1}, W_{l_k}^{s+1} \) are optimal, terminate the iterative process, otherwise, go to step 9.

Step 9. Update the \( H \) matrix: \( H^{s+1} = H^s + M^s + N^s \), where
\[
M^s = \sum_{i=1}^{m} \frac{S_{f_i} S_{f_i}^T}{\vert S \vert^2}; \quad N^s = -\frac{H^s Q^s Q^s^T}{\vert Q^s \vert^2}; \quad Q^s = \begin{bmatrix} Q^f \\ Q^l \end{bmatrix}
\]

\[
\begin{bmatrix} Q^f_{i} \\ Q^l_{k} \end{bmatrix} = \begin{bmatrix} \nabla E(W_{f_i}) - \nabla E(W_{f_i}) \nabla E(W_{l_k}) - \nabla E(W_{l_k}) \end{bmatrix}
\]

\( (i = 1, 2, \ldots, m; \quad k = 1, 2, \ldots, n) \); \( (i = 1, 2, \ldots, m; \quad k = 1, 2, \ldots, n; \quad j = 1, 2, \ldots, N) \)

Step 10. Set the new iteration number \( l = l + 1 \), and go to step 4.

### 5. Application Examples

Several application examples of the new learning approach are presented. The results are compared with the SBP approach.

#### 5.1 Learning a Nonlinear System

This is a nonlinear system in which the mathematical model, structure and other characteristics are unknown, and may be considered as a "black box". The only known quantities are its inputs and desired output:

\[
\begin{align*}
\begin{cases}
x_1(s) = 1 \\
x_2(s) = 1.05s \\
y_d(s) = 1 + e^{-0.05s}
\end{cases}
\end{align*}
\]
(14)

For this nonlinear system, we construct a four-layer back-propagation neural network. The input layer has 2 neurones for two inputs: \( x_1, x_2 \). Each of the two hidden layers has 10 neurones and the output layer has a single neurone for one output. For \( s = 1, 2, \ldots, 100 \), we obtain 100 patterns as training vectors from equation (14). We use both the SBP learning algorithm and the proposed LM learning algorithm to train this neural network for comparison. The result after five minutes of training is given in Fig. 2. Where \( y_d \) indicates the desired output of the nonlinear system, \( y_i \) and \( y_d \) are outputs of the neural network trained by using LM and SBP, respectively.

The result clearly shows that \( y_i \) follows \( y_d \) very closely (indistinguishable in Fig. 2) except at the very beginning. The
errors versus the training time of the two methods are shown in Fig. 3.

As anticipated, the errors from both methods will decrease when the training time is increased. Nevertheless, the LM is superior to the SBP both in the speed of learning and accuracy. For example, as the training time is increased to 90 minutes, the corresponding errors for LM and SBP are 0.022893 and 0.063101, respectively.

5.2 Learning for Epipolar Line Correspondence

In the 3D parts recognition process, it is important to reconstruct a 3D parts model from a 2D stereo image pair. The stereo matching by using epipolar lines is by the conventional method. If cameras are precisely aligned, a point projected onto an epipolar line of the left-hand image will be projected along the corresponding epipolar line of the right-hand image. Although the process is very complicated, the epipolar lines of images can be calculated based on the camera geometry.

However, in the practical situation, such as in the parts assembly environment, cameras are not precisely aligned, the epipolar line is not a direct line for line mapping between two images in a stereo pair as shown in Fig. 4. We use the learning ability of the neural network to set up the relationship between the left-hand and the right-hand images of the stereo image pair. From this relationship, epipolar lines of the right (left) hand image can be estimated when epipolar lines of the left (right) hand image are given. In order to perform this process, we constructed a four-layer back-propagation neural network. The input layer has 3 neurones for three inputs: the x- and y-coordinates of the left-hand image and the x-coordinate of the right-hand image. Each of the hidden layers has 20 neurones for storing the camera geometric relationship learned through the training process. The output layer has one neurone for one output: the y-coordinate of the right-hand image.

We use 150 training vectors in the training process. The training time versus the errors are shown in Fig. 5. Again, the LM provides better accuracy than the SBP for the same training time and a much shorter training time for the same error.

5.3 Learning for 2D to 3D Point Conversion

As mentioned above, in the 3D parts recognition process, a 3D parts model must be reconstructed from 2D images. From the 3D model, the parts recognition process can be performed. In the conventional 2D to 3D points conversion process, an elaborate system calibration is required to determine the relationships between the two cameras and the global coordinate system. The usual approach is by using known 3D points in 3D space and the corresponding 2D points on images to create the relationship between the cameras and global coordinate system. Then each 3D point is found by feeding the two matching points into equations governing the transformations. Usually the collinear condition equations are used [10].

\[
\begin{align*}
x + f &\begin{pmatrix} a_1(X - X_l) + b_1(Y - Y_l) + c_1(Z - Z_l) \\
& a_2(X - X_l) + b_2(Y - Y_l) + c_2(Z - Z_l) \\
& a_3(X - X_l) + b_3(Y - Y_l) + c_3(Z - Z_l) \end{pmatrix} = 0 \\
y + f &\begin{pmatrix} a_1(X - X_l) + b_1(Y - Y_l) + c_1(Z - Z_l) \\
& a_2(X - X_l) + b_2(Y - Y_l) + c_2(Z - Z_l) \\
& a_3(X - X_l) + b_3(Y - Y_l) + c_3(Z - Z_l) \end{pmatrix} = 0
\end{align*}
\] (15)

where \( X, Y, Z \) are global coordinates of a point on an object, \( X_l, Y_l, Z_l \) are the global coordinates of the camera, and \( a_i, b_i, c_i \) \( (i = 1, 2, 3) \) are the extrinsic elements of the camera. The \( f \) is the focal length of the camera and \( x, y \) are the coordinates of the point on the image. The relationships in equation (15) are very complicated. If system error must be considered, an iteration process has to be used and the calculation is very time consuming.

We construct a four-layer back-propagation neural network to perform the operation. The input layer has four neurones for the \( x \)- and \( y \)-values of matched points from the left-hand and right-hand images of a stereo pair. The network has two hidden layers. Each contains 50 neurones for storing the complicated relationship between the 2D image and 3D space. The output layer has 3 neurones for 3D points. One hundred and seventy-five vectors are used for the network training. The results of SBP and LM are shown in Fig. 6. Again, the LM outperforms the SBP in accuracy and learning time.

6. Conclusion and Discussion

A learning mechanism for parts recognition in an intelligent assembly system is presented. It is based on the general philosophy of the standard back-propagation approach. A new optimal searching direction and a new variable optimal learning rate are suggested as the learning mechanism. Compared with the SBP which uses a constant step size in the negative gradient direction for error minimisation, the proposed learning mechanism offers the advantages of higher accuracy (smaller error) with shorter training time and superior converging reliability. The question of what is the acceptable error level, depends on the system requirement and the cost/performance objective. From the examples, the LM consistently provides the same performance with lower cost (time), or higher performance at the same cost.

The proposed LM is not suitable in all situations. As the number of neurones of the neural network increases, the size of the matrix \( H \) increases very quickly. It will be difficult to realise this approach by using conventional computing machines.

Acknowledgement

The authors would like to acknowledge the support provided by the Croucher Foundation under project number 9050044.
Fig. 4. Epipolar lines in a stereo image pair. (a) Left image. (b) Right image.

Fig. 5. The errors versus the training times using LM and SBP for epipolar line estimation.

Fig. 6. The errors versus the training times using SBP for 2D to 3D points conversion.

References


10. Y. Xiong, X. Qiao and D. Wang, “The measurement and three dimensional reconstruction for aircraft using image processing techniques”, Transactions of Nanjing University of Aeronautics and Astronautics, 10(1), December 1993.

Appendix A (development of $H$)

The variable metric direction is a quasi-Newton direction. Quasi-Newton methods usually converge faster and are more robust than gradient directions. The weakness is the requirement for much larger storage, therefore it is not suitable for very large problems. Following the Newton method, the direction is defined as

$$ S_{n} = -G^{-1}V E' $$

where $G$ is the Hessian matrix. It is a very complicated process to find the inverse Hessian matrix. For most problems, the quasi-Newton direction is suggested.
the outputs of the output layer neurones are obtained as

\[ y_{j}^{(i+1)} = f(A_{j}^{(i+1)}) = \frac{1}{1 + \exp(-A_{j}^{(i+1)})} \quad (j = 1, 2, \ldots, N) \]  

(4b)

By substitution,

\[ y_{j}^{(i+1)} = \frac{1}{1 + \exp\left(-\sum_{k=1}^{m} (W_{jk}^{(i)} + \alpha S_{jk}^{(i)}) y_{k}^{(i)}\right)} \quad j = 1, 2, \ldots, N \]  

(5b)

In the same way, we can update the weighted values between input and hidden layers.

\[ W_{jk}^{(i+1)} = W_{jk}^{(i)} + \alpha S_{jk}^{(i)} \quad (i = 1, 2, \ldots, m; \: k = 1, 2, \ldots, n) \]  

(6b)

Outputs of neurones in the hidden layer are as follows:

\[ y_{j}^{(i+1)} = \frac{1}{1 + \exp\left(-\sum_{k=1}^{n} (W_{jk}^{(i)} + \alpha S_{jk}^{(i)}) x_{k}\right)} \quad (k = 1, 2, \ldots, n; \: i = 1, 2, \ldots, m) \]  

(7b)

From equations (5b) and (7b), we obtain the following equation:

\[ y_{j}^{(i+1)} (\alpha') = \frac{1}{1 + \exp\left(-\sum_{i=1}^{m} \frac{W_{ji}^{(i)} + \alpha S_{ji}^{(i)}}{1 + \exp\left(-\sum_{k=1}^{n} (W_{jk}^{(i)} + \alpha S_{jk}^{(i)}) x_{k}\right)} \right)} \]  

(8b)

The error function of the number j neurone in the output layer is given as follows:

\[ E^{(i)}_{j} (\alpha') = (y_{j}^{(i)} (\alpha') - d_{j})^{2} = \frac{1}{1 + \exp\left(-\sum_{i=1}^{m} \frac{W_{ji}^{(i)} + \alpha S_{ji}^{(i)}}{1 + \exp\left(-\sum_{k=1}^{n} (W_{jk}^{(i)} + \alpha S_{jk}^{(i)}) x_{k}\right)} \right)} - d_{j}^{2} \]  

(9b)

So the error function for the system is

\[ E^{(i+1)} = \frac{1}{N} \sum_{j=1}^{N} E^{(i)}_{j} (\alpha') = \frac{1}{N} \sum_{j=1}^{N} (y_{j}^{(i+1)} (\alpha') - d_{j})^{2} \]  

(10b)

In order to find optimal step length \( \alpha \), we minimise the objective function \( E^{(i)} (\alpha') \). That is

\[ \min \frac{1}{N} \sum_{j=1}^{N} \left( \frac{1}{1 + \exp\left(-\sum_{i=1}^{m} \frac{W_{ji}^{(i)} + \alpha S_{ji}^{(i)}}{1 + \exp\left(-\sum_{k=1}^{n} (W_{jk}^{(i)} + \alpha S_{jk}^{(i)}) x_{k}\right)} \right)} - d_{j}^{2} \right) \quad (j = 1, 2, \ldots, N) \]  

(11b)