REPORT DOCUMENTATION PAGE

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
May 1990	Conference paper	

4. TITLE AND SUBTITLE

See report.

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

See report.

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

See report.

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

See report.

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Distribution Statement A - Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

Presented at the IEEE 1990 National Aerospace and Electronics Conference (NAECON 1990) held in Dayton, Ohio, on 21-25 May 1990.

14. ABSTRACT

See report.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**	UU		**19b. TELEPHONE NUMBER** *(include area code)*
Unclassified	Unclassified	Unclassified			

NEURAL NETWORK SYSTEM FOR MANUFACTURING ASSEMBLY LINE INSPECTION

Alastair D. McAulay, PhD
Wright State University
Department of Computer Science and Engineering
Dayton, OH

Paul Danset
Wright State University
Department of Computer Science and Engineering
Dayton, OH

Devert Wicker
Wright Research and Development Center (WRDC/AART)
Wright-Patterson Air Force Base, OH

Abstract

Assembly line inspection is currently performed for General Motor's clutch drivers by means of a vision system. When the part is changed, the system must be reprogrammed, which takes time and is expensive. A new system has been developed and demonstrated in the Computer Science and Engineering Department at Wright State University that permits an operator to teach the system what is to be considered good and bad without any need for computer reprogramming. The machine is shown good parts and flawed parts. In the latter case, the type of flaw is entered in the computer. Preprocessing is used to provide position and rotation invariance. A feedforward network is then trained to provide the correct output. The system is shown to perform reliably and has been modified to cope with more difficult inspection systems in which back lighting may not be used.

1 BACKGROUND

A vision system to inspect clutch drivers for missing rivets and springs at the Harrison Radiator Plant of General Motors (GM) works only on parts without covers (Figure 1) and is expensive. The system does not work when there are cover plates (Figure 2) that rule out backlight passing through the area of missing rivets and springs. Also, the system like all such systems must be reprogrammed at significant time and cost when the system needs to classify a different fault or a different part [1]. So the desired features of an inspection system are that it can be easily adapted to new parts as well as being fast and low-cost. A part inspection demonstration system now exists which demonstrates both ease of adaptation to new parts through a user-friendly interface and a learning part inspection algorithm. The hardware consists of a Sun workstation, Datacube, and camera. The classifier used by the inspection algorithm depends on a feed-forward neural network trained using back-propagation [2,3,4].

2 USER INTERFACE

The user interface is a mouse driven graphics software package (Figure 3) which allows the operator to easily work with the system to train on new parts. In learning a new part, the operator selects *LOAD* to load a previously stored image or selects *GRAB* to capture a new image from the camera. Each image pixel is sampled into 256 grey levels. For the first training part, the mouse is used to select the regions of interest (ROI's). The system is currently tailored for the clutch driver part of figures 1 and 2 in that the ROI is a circular ring. The clutch is round and all relevant information for determining a good or bad part is found in a circular band concentric to the center of the clutch. In a fully developed system, these ROI could be of different user selectable geometries such as a circle, rectangle, circular ring, or others. The operator enters names for the significance of the ROI's such as "missing rivet" or "missing spring" so that the system can give descriptive classifications of parts in the recognition phase. Selecting true (T) or false (F) for each of these ROI descriptions and then selecting *TEACH* gives the computer a training pattern set which will be learned later. This process is repeated for all the training parts.

Once the part inspection algorithm is trained, recognition is simply performed by capturing a part's image into the computer (select *GRAB*) and performing recognition with the inspection algorithm (select *RECOGNIZE*). The true and false boxes used previously to train the system will display the results. New training parts may be entered at any time. The interface also allows the operator to print or save the neural network information, select an auto center option, and display where the ROI pixels are on the part. The interface also allows the operator to save the system's state information (select *SAVE NET*) so that retraining is not required after a power outage. For parts which are not in a fixed position, an auto center option is available to the operator (select *AUTO CENTER*).
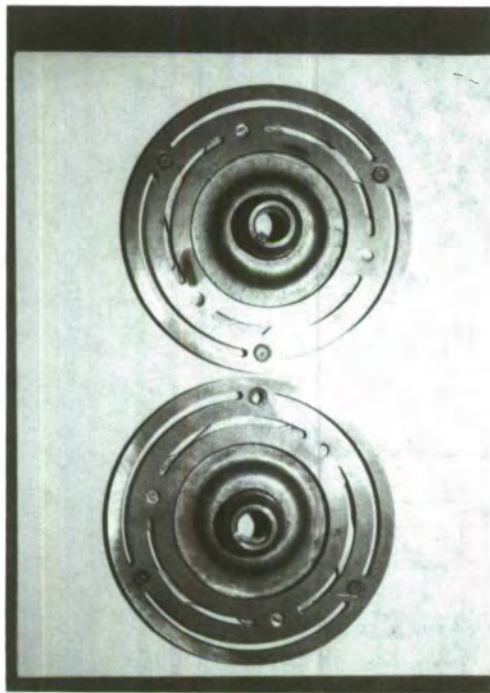
Figure 1: Example parts without a cover. The top part is good and the bottom part has a missing rivet. In this case, a rivet is missing at the top.
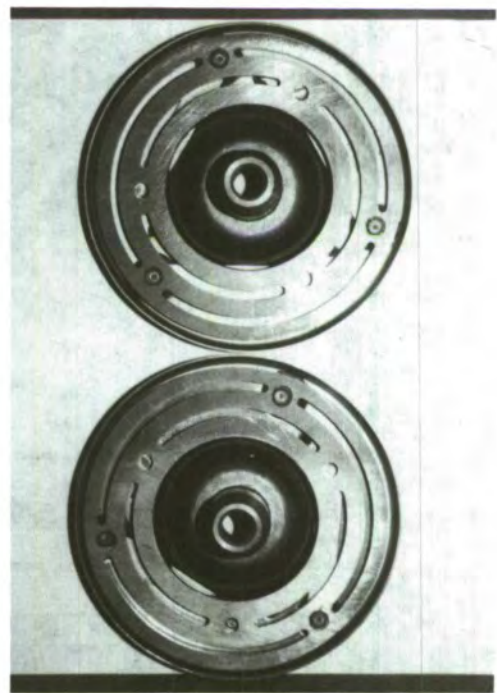

Figure 2: Example parts with a cover. The top part is good and the bottom part has a missing spring. The spring is a long flat rectangular piece which is mounted on an outer rivet and an inner rivet. In this case, a spring is missing at the top.
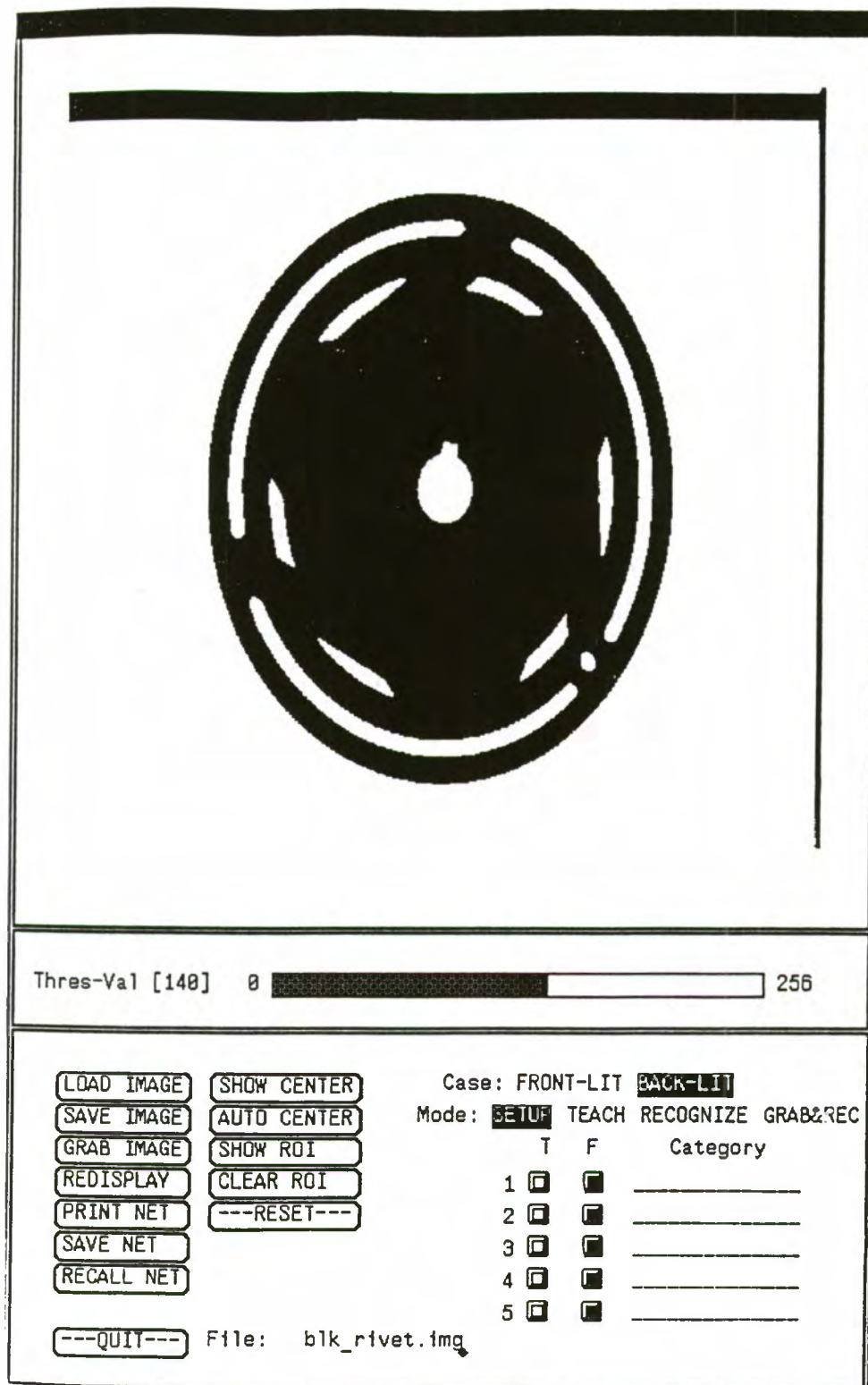
Figure 3: The demonstration inspection system's screen for a backlit part. Note, a missing rivet appears as a white hole at the bottom right of the part.
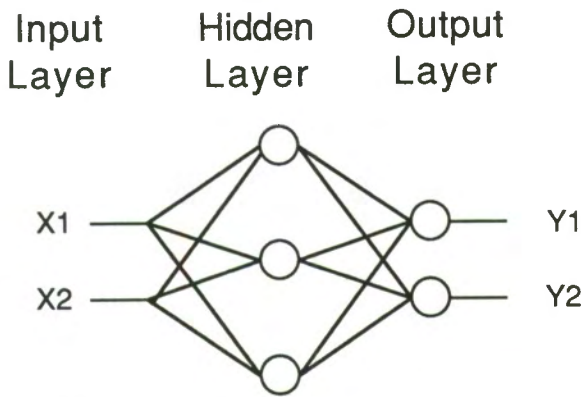
Input Layer   Hidden Layer   Output Layer

Figure 4: An example feed-forward neural network.

3 INSPECTION ALGORITHM

The inspection algorithm solves the classical recognition problem and thus must consist of a feature extractor and a classifier [5].

3.1 Feature Extraction

The demonstration system feature extraction steps consist of centering the part, extracting the ROI pixels, thresholding the pixels, calculating the frequency magnitude spectrum, zeroing out the DC component, and mapping the spectrum to numbers between 0 and 1. If selected, centering the part is automatically performed by using a simple edge detection algorithm. The ROI's circular band is sampled at equal angle increments into 1024 pixels for later application of a Fast Fourier Transform (FFT) [6]. The user selects the ROI for the first part. Using the ROI radius of the first part, the system can then automatically retrieve the ROI from all other parts as they are digitized.

The ROI data may then be processed in different ways based on which type of part is used. In the backlit case (part without cover), the image pixels are threshed into 0 or 1. In the frontlit case (part with cover), thresholding removes significant features such as edges of metal on metal from the pixel information so the grey levels must be used. The FFT magnitude spectrum is used to produce a shift invariant pattern and since the ROI is a circular band, the image is made rotation invariant. For the type of part considered only the first 100 spectral frequencies were needed, thus reducing learning time relative to including all spectral frequencies. Next, the DC component is removed and the FFT magnitude spectrum is normalized to a maximum of one. Without the last two steps, the learning algorithm did not converge.

3.2 The Classifier

The pattern recognition capability of neural networks is documented by many researchers [2,3,4,7]. Figure 4 shows a three layer network and an example node. The demonstration system has one neural network per ROI. Each network consists of three layers with 100 input, 10 hidden, and 2 output nodes. Weights are randomly selected to be from 0.0 to 1.0. Back-propagation uses an iterative gradient algorithm to adjust the network's weights so as to minimize the mean square error between the actual and desired output of all the training patterns [2,3,4].

The target output patterns are [0.1,0.9] for a good part and [0.9,0.1] for a bad part. Consider the network with J neurons in the output layer, I hidden neurons, and K neurons in the input layer. The outputs may be written for $j = 1$ to J.

$$y_j = g \left[\sum_{i=1}^{I} w_{ij} g \left(\sum_{k=1}^{K} w'_{ki} x_k \right) \right] \qquad (1)$$

The nonlinear neuron functions were selected as in references [2, 3,4]

$$g(t) = (1 + e^{-t})^{-1} \qquad (2)$$

The weights are updated sequentially for each layer, starting at the layer nearest the output. At the l th iteration, for a given layer, the update is performed with

$$w_{ij}(l + 1) = w_{ij}(l) - \mu(l) v_j(l) p_i(l) \qquad (3)$$

where $v_j(l)$ and $p_i(l)$ are computed as described in reference [8] and $\mu(l)$ permits overrelaxation. v_j depends on the output and error out of the jth node. p_i is the value of the input from the ith node. The outputs for each set of data are computed using the forward equations (1). For the output layer, the last term in equation (3), the weight update-increment, is computed for each training set. These weight update-increments are averaged across all sets of training data for application in equation (3). The hidden layer weights are then computed in a similar manner. The procedure is repeated iteratively until the weights stabilize. The network also has one bias node with a value of 1 which is connected through weighted links to all other nodes as in [4]. The bias node's link weights are trained similarly to other weights in the network. The bias node allows the training algorithm to individually shift each node's output curve. In the recognition stage, the previously trained network is simply propagated using the input pattern in question. The resulting output is used to classify the input pattern.

4 TESTS AND RESULTS

The system has been tested and found able to correctly recognize good and bad parts. In the backlit rivet case, the system was given two training patterns, a good part and a bad part, and found to correctly identify the training parts and one unknown good part at rotations of 6° increments. In the frontlit (no thresholding) spring case, the system was given 16 training patterns, 8 orientations each of a good and bad part, and found to correctly identify the training parts and one unknown good part at rotations of 6° increments. Without training the system with different orientations of the same part, the system incorrectly identified parts. Variations in the frequency spectrum were observed to be due to milling and ambient lighting variations and therefore additional rotations were required.

Timing runs were performed on the backlit case clutch driver part to see if the system requires less than the estimated assembly line requirement of 2 seconds per part. The backlit case is estimated to take $0.63 + 0.78 * N$ seconds to recognize a part's N ROI in a production line system using a SUN 3/160. The 0.63 seconds is required to sample the image (1/30 second) and center the part automatically (0.60 seconds) while the 0.78 seconds is due to the other feature extraction steps and the neural net propagation time (0.04 seconds per ROI) which are required for each ROI. So the system is fast enough to process a part with one ROI within 2 seconds. The time could be reduced by at least an order of magnitude if a signal processing chip or array processor were incorporated into the system.

5 DISCUSSION

The neural net approach makes the pattern recognition system flexible because the system uses learning to be able to classify good and bad parts. The existing demonstration system looks applicable for separating round parts in the class of problems which have all required information in a circular band concentric to the center of the part and which have features which are visually detectable. This is of considerable interest given the fairly simple approach presented above compared to more complex approaches of current day systems [1]. In the research performed so far, only a limited number of parts were used for training and recognition studies. For more difficult cases, extraction of different features other than a circular band is needed and/or the system can be trained with more and more parts until it narrows in on exactly which inputs in the pattern are significant.

6 CONCLUSION

The feasibility of using neural networks combined with a simple feature extraction algorithm to make visual inspection systems which learn has been demonstrated. The system seems to be a viable option for the factory line environment because the system is flexible and fast. As far as flexibility, the existing demonstration system can separate round parts in the class of problems which have all of the required information in a circular band concentric to the center of the part and which have features which are visually detectable. The user can easily train the system by showing it good and bad parts. The system can be readily adapted to parts which are easily centered.

7 Acknowledgements

References

[1] J. Wilder, "Industrial requirements for real-time image processing", SPIE '87, Vol 849, Automated Inspection and High Speed Vision Architectures.

[2] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning internal representations by error propagation", in *Parallel distributed processing: Explorations in the Microstructure of cognition*, The MIT Press, Cambridge, 1986.

[3] R.P. Lippmann, "An Introduction to Computing with Neural Nets", IEEE ASSP April 1987.

[4] W.P. Jones and J. Hoskins, "Back-Propagation: A generalized delta learning rule." Byte, October 1987.

[5] K.S. Fu, *Sequential Methods in Pattern Recognition and Machine Learning*. Academic Press 1968.

[6] A. Oppenheim and R. Schafer, *Digital Signal Processing*, Prentice-Hall Inc. 1989.

[7] M. Schmutz, M. Rueff, and U. Mussigmann, "Neural networks: a new approach to pattern recognition." SPIE '87, Vol 849, Automated Inspection and High Speed Vision Architectures.

[8] A.D. McAulay, "Engineering design neural networks using split inversion learning", IEEE First International Conference on Neural Networks, Vol. **4**, pp. 635-642, June 1987.