Correspondence

A Neural-Based Crowd Estimation by Hybrid Global Learning Algorithm

Siu-Yeung Cho, Tommy W. S. Chow, and Chi-Tat Leung

***Abstract*—A neural-based crowd estimation system for surveillance in complex scenes at underground station platform is presented. Estimation is carried out by extracting a set of significant features from sequences of images. Those feature indexes are modeled by a neural network to estimate the crowd density. The learning phase is based on our proposed hybrid of the least-squares and global search algorithms which are capable of providing the global search characteristic and fast convergence speed. Promising experimental results are obtained in terms of accuracy and real-time response capability to alert operators automatically.**

I. Introduction

Recent efforts in crowd estimation at underground stations, airports or stadiums are currently addressed in the research field of automatic surveillance systems [1]–[5]. Most current visual surveillance systems use a set of cameras to provide human operators with visual information. Making decision is rather difficult for the operators in a control room simply by monitoring a sequential images obtained from different cameras and different areas under surveillance control. The area under surveillance is also relatively large and semi-confined. The crowd behavior in such semi-confined space is more complex and will be influenced by psychological, physiological and environmental factors, social relationship to neighboring pedestrians, purpose of walk, area topography, ... and so on. As a result, conventional techniques are not able to fulfill a fundamental requirement of reliability in the estimation of crowd density. Conventional methods for an automatic crowd detection are based on counting of people in the station. Jukkala *et al.* [1] used a radiometer for counting of people. The method is only reliable for light traffic. Mudaly [2] proposed an infrared detection system to counting people in a long and straight corridor. Obviously, accurate head counting is an excessively complicated or impossible task to accomplish. In addition, an accurate head counting figure is largely unnecessary in most practical operations.

Recently, two real-time crowd density estimation systems in London [3] and Genova [4], [5] have been proposed based on existing installed closed circuit television (CCTV). In order to comply with the criterion of being a real-time system, the two systems basically employ a number of simple image processing techniques for feature extraction over the image frames from CCTV. The image processing techniques are summarized as follows. Firstly, background removal is an idea to measure the area occupied by the crowd from those of the background. Secondly, edge detection is an alternative idea to measure the total perimeter of all the regions occupied by people. Meanwhile, an analytical model correlating each feature value extracted from the aforementioned processing techniques is required. The model optimization is currently based upon a distributed extended Kalman filter (DEKF) system using both spatial and temporal information [5], [6]. Crowd estimation was performed, according to a distributed scheme, by a hierarchical network of nodes. Each node is implemented as a Kalman filter of which the number of people is the local status variable to be estimated and predicted. The maximum number of people counted by the above system is only about 30 and the error may be large under overcrowded conditions. The computation is complex and multi-nodes Kalman filter type algorithm is of slow convergence.

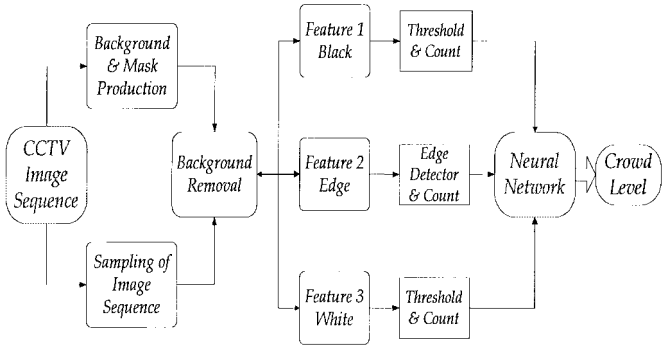


Fig. 1. The basic block diagram of the neural based crowd monitoring system.

In this paper, a neural based crowd estimation system is applied at the platforms of the mass transit railway (MTR) stations in Hong Kong. The main objectives of the crowd estimations are to provide a system to detect overcrowded situations on the underground stations and a statistical temporal evaluation of the number of people for planning traffic activities. The system is based upon the use of a CCTV camera which provides visual information for a MTR station. The image pre-processing techniques are required to provide a feature extraction module to map the visual information coming from the images into a two-dimension feature space. The feature coefficients, from the extraction module, are then fed into the neural networks to classify the level of crowd. The subsequent crowd classification is performed by the hybrid global learning (HGL) algorithm [7]–[9] which combines the least-squares method together with different global optimization methods, such as random search (RS) [10], simulated annealing (SA) [11], and genetic algorithm (GA) [12]. Based on the above global search methods, the limitation of conventional backpropagation algorithms suffered from the problem of local minima can be overcome. The objective of the global search method is to design a methodology which is capable of providing a chance to converge to the global minimum in the error surface. The HGL algorithm used in this paper is based on the concept that the weights between the output and the hidden layers are determined by the least-squares algorithm, while the weights at the lower layers are determined by the global search algorithm. The dimensionality of weight space for global search algorithm can then be reduced and hence the convergence can be significantly speeded up. Benefiting from the HGL algorithm, the performance of the neural based crowd estimation is significantly enhanced in terms of efficiency, speed and accuracy. The classification results from the neural network is fed into the decision module to interpret the crowd density. The block diagram of our proposed neural based crowd monitoring system is shown as Fig. 1.

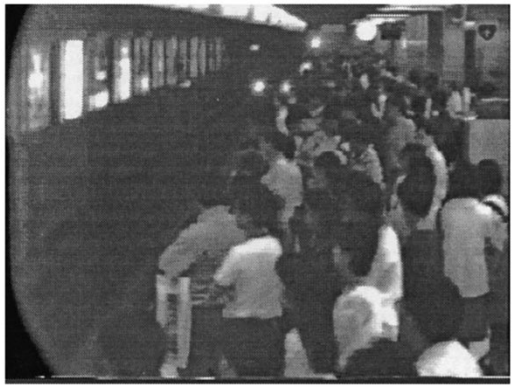
This paper is organized as follows. In Section II, the feature extraction techniques for crowd estimation is described. The methodologies

Manuscript received October 23, 1997; revised December 24, 1998. This paper was recommended by Associate Editor T. H. Lee.

The authors are with the Department of Electronic Engineering, City University of Hong Kong, Kowloon Tong, Hong Kong.

Publisher Item Identifier S 1083-4419(99)07277-5.

(a)



(b)

Fig. 2. The typical crowded images capturing from the platform of MTR station. The crowd densities estimated by human scene are approximately (a) 90% and (b) 92%.

of edge detection, crowd objects extraction and background removal are included. The neural model and hybrid global learning algorithm are derived in Section III for this study. Three different types of hybrid algorithms are also summarized in this section. In Section IV, the experimental results for the crowd estimation are presented and the conclusion is drawn in Section V.

II. Feature Extraction for Crowd Estimation

The objective of the feature extraction is to extract, from the grey-scale video images, for low-level visual information relative to the crowd level in the scene. In principle, the visual features should be able to discriminate well among crowd classes. The first assumption is that the crowd level present in the scene is correlated to the "fragmentation" of the image regions in which a significant motion occurs. In our study, the results of feature extraction contribute three indexes to represent the crowd density of the corresponding image. The three indexes are represented by the length of edges of the crowd objects, the crowd objects density and the background objects density. Fig. 2(a) and (b) depict the typical original images of the platform of the MTR stations. The corresponding crowd densities of these images was estimated by human scene.

A. Edge Detection

The length of edges of the crowd objects is evaluated by an edge detector to count the number of pixels of the passengers' "edges." The edge detection is a standard low-level image processing function which can be used to derive outlines of individuals and groups from a video image. It has an advantage for the classification in a low density crowds. The classical edge detection for the image processing technique is employed by convolution by mean of a Sobel filter evaluation. The main drawback of Sobel filtering is that a limited 3 $\times$ 3 filter mask can only give edge on a single direction and hence at least two times of filtering are necessary to obtain satisfactory results. The computation is time-consuming and cannot serve the purpose of a real-time operation. Consequently, a fast edge detection scheme is proposed for the practical crowd estimation. For a single image, edge can be searched more easily and precisely by an appropriate binary thresholding of the source image which results a substantial reduction on the computational complexity. The block diagram of the proposed fast edge detector is shown in Fig. 3. Fig. 4(a) and (b) shows the resulting images extracting from the Sobel filter type and the proposed fast edge detector, respectively.

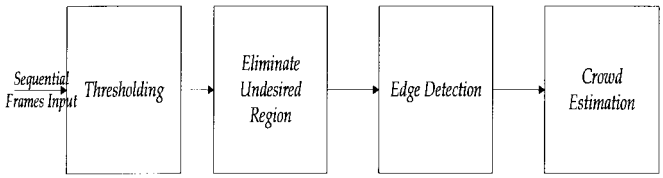
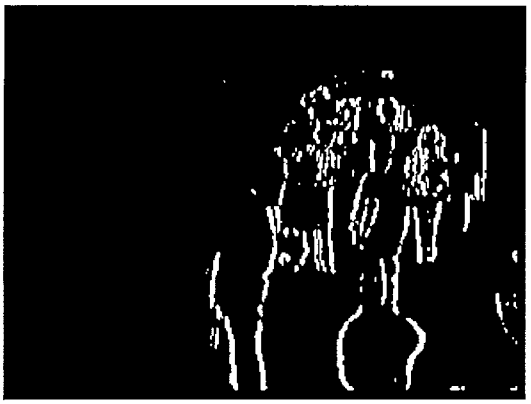
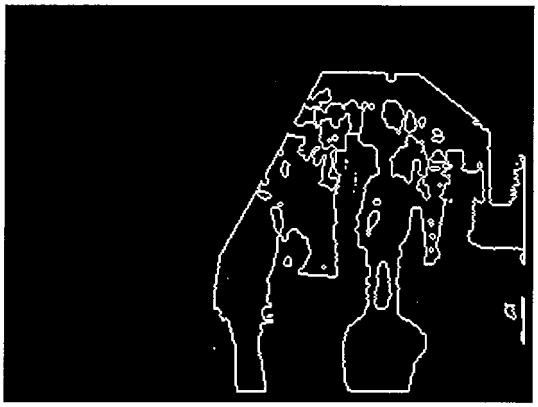


Fig. 3. The block diagram of the proposed fast edge detector.



(a)



(b)

Fig. 4. The images of the passengers "edges" have been extracted by the different types of the edge detectors (a) Sobel Filter and (b) proposed fast edge detector.

B. Crowd Objects Extraction and Background Removal

It is clear that a human observer has absolutely no problem in distinguishing a very dense crowd from the background. It is believed that human brain is well trained and would be likely to use the ratio

Fig. 5. A typical masking image as a "reference" for either extraction of crowd objects or background removal.

of "crowd area" to "background area" as an estimate for the crowd density. This idea could be applied quantitatively to computer-based density estimation if the image-pixels corresponding to the crowd could be separated from those of the background. In our study, crowd density can be represented by the number of image pixels corresponding to the crowd which must be separated from those of the background at the station platform. The separation is based on the processes of the masking and filtering the background information, especially for the track in the station, by using a "reference image" of the scene. A typical masking image is shown in Fig. 5 as a reference for either extraction of crowd objects or background removal. In accordance with the aforementioned methodology, an efficient crowd objects (people) extraction is applied to an original image so that the binary level $C_{i,j}$ at pixel (i,j) is approximated by

$$C_{i,j} = \begin{cases} M_b & \text{if } R_{i,j} = M_b \text{ or } I_{i,j} < T_c \\ M_w & \text{otherwise} \end{cases} \tag{1}$$

where M_b and M_w are the binary value of the black and white color respectively, i.e., $M_b = 1$ and $M_w = 0$. $I_{i,j}$ and $R_{i,j}$ are the intensity of the original current image and the binary reference image for masking at pixel (i,j), respectively. T_c is a threshold level for the crowd objects. The selection of T_c is dependent on the lighting layout and grey-scale color of the platform background. In this study, T_c is set to 104. Fig. 6 shows a typical result in which the original image is processed by the image masking and filtering procedures. On the other hand, the determination of the background density is evaluated by the similar thresholding technique of the crowd object extraction. Fig. 7 shows a processed image for the background density evaluation.

III. Neural Model and Hybrid Global Learning Algorithm

In this paper, a single hidden layer network is considered for the modeling of the crowd density monitoring. A neural network with n input units, m hidden units and q output units is used and the activation function is in a form of $f_H(x) = 1/(1+\exp(-x))$. The following notation are used. There are p patterns extracted from the feature extractor in the training set. For pattern $k = 1, 2, \cdots, p$, let $t_k = (t_{1k}, t_{2k}, \cdots, t_{qk})^T$ denote the desired output vector of the network, $o_k = (o_{1k}, o_{2k}, \cdots, o_{qk})^T$ denote the true output vector of the network, $v_k = (1, v_{1k}, v_{2k}, \cdots, v_{mk})^T$ denote the vector of outputs of the hidden layer, and $x_k = (1, x_{1k}, x_{2k}, \cdots, x_{nk})^T$ denote the vector of inputs of the network. Let a_{ji} denote the weight of the ith input to the jth neuron of the hidden layer and a_{j0} denote the bias of the jth neuron of the hidden layer; let b_{lj} denote the weight from the jth hidden neuron to the lth neuron of the output layer and b_{l0} denote the bias of the lth neuron of the output layer for $1 \le l \le q$, $1 \le j \le m$, $0 \le i \le n$. The network acts as an approximating function $F(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{x}_k)$ with two sets of fixed weights

Fig. 6. The resulting image evaluated by the crowd object extraction.

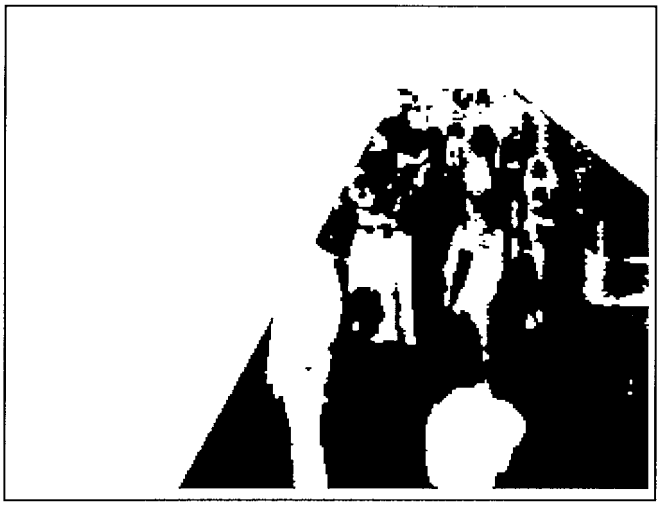


Fig. 7. The resulting image evaluated by background density evaluation.

$$\boldsymbol{A} = \begin{pmatrix} a_{10} & a_{11} & \cdots & a_{1n} \\ a_{20} & a_{21} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{m0} & a_{m1} & \cdots & a_{mn} \end{pmatrix}, \quad \text{and}$$

$$\boldsymbol{B} = \begin{pmatrix} b_{10} & b_{11} & \cdots & b_{1m} \\ b_{20} & b_{21} & \cdots & b_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ b_{q0} & b_{q1} & \cdots & b_{qm} \end{pmatrix}.$$

The output of the lth neuron of the output layer is given by

$$o_{lk} = f_H\left(\sum_{j=0}^{m} b_{lj} f_H\left(\sum_{i=0}^{n} a_{ji} x_{ik}\right)\right), \qquad 1 \le k \le p. \tag{2}$$

The (2) can be rewritten in matrix form, i.e.

$$\begin{aligned} \boldsymbol{v}_k &= \begin{pmatrix} 1 \\ F_H(\boldsymbol{A}\boldsymbol{x}_k) \end{pmatrix} \\ \boldsymbol{o}_k &= F(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{x}_k) = F_H(\boldsymbol{B}\boldsymbol{v}_k) \end{aligned} \tag{3}$$

where

$$F_H\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \end{pmatrix} = \begin{pmatrix} f_H(x_1) \\ f_H(x_2) \\ \vdots \\ f_H(x_k) \end{pmatrix}.$$

A Sum squared-error function, which is given by

$$E(\boldsymbol{A},\boldsymbol{B}) = \sum_{k=1}^{p} (\boldsymbol{t}_k - \boldsymbol{o}_k)^T (\boldsymbol{t}_k - \boldsymbol{o}_k) \tag{4}$$

is selected to be a cost function. Learning algorithm strives to minimize learning error defined by the cost function. In other words, learning algorithm is to find suitable weight matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ such that the learning error is minimum. The learning error $E(\boldsymbol{A},\boldsymbol{B})$ can be in a form of

$$\begin{aligned} E(\boldsymbol{A},\boldsymbol{B}) &= E_A(\boldsymbol{B}) \\ &= \sum_{k=1}^{p} (\boldsymbol{t}_k - F_H(\boldsymbol{B}\boldsymbol{v}_k))^T (\boldsymbol{t}_k - F_H(\boldsymbol{B}\boldsymbol{v}_k)). \end{aligned} \tag{5}$$

The error surface defined by the learning error $E_A(\boldsymbol{B})$ have one and only one minimum because the function F_H is invertable such that the optimal weight matrix $\boldsymbol{B}$ can exactly be computed by

$$\hat{\boldsymbol{B}} = F_H^{-1}(\boldsymbol{t})\boldsymbol{v}^T(\boldsymbol{v}\boldsymbol{v}^T)^* \tag{6}$$

where

$$\boldsymbol{t} = (\boldsymbol{t}_1, \boldsymbol{t}_2, \cdots, \boldsymbol{t}_p), \quad \boldsymbol{v} = (\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_p)$$

F_H^{-1} is the inverse function of F_H and $(\boldsymbol{v}\boldsymbol{v}^T)^*$ is a pseudo-inverse of $\boldsymbol{v}\boldsymbol{v}^T$. Thus, the learning error $E(\boldsymbol{A},\boldsymbol{B})$ can be reformulated in a form of

$$E(\boldsymbol{A},\boldsymbol{B}) = E(A,\boldsymbol{B}(\boldsymbol{A})) = E'(\boldsymbol{A}) \tag{7}$$

because the weight matrix $\boldsymbol{B}$ can be expressed as a matrix function of matrix $\boldsymbol{A}$. There exists a new weight space $\boldsymbol{W}'$ with lower dimension mapping into the original weight space $\boldsymbol{W}$. In other words, the minimization of the new cost function $E(\boldsymbol{A})$ is equivalent to minimizing the learning error over the whole weight space $\boldsymbol{W}$. Based on the HGL algorithm, the least-squares method is used to compute the corresponding matrix $\boldsymbol{B}$ for any given weight matrix $\boldsymbol{A}$ according to the (6). This results in speeding up the convergence rate substantially. The global search method is used to iteratively estimate the optimal matrix $\boldsymbol{A}$ via minimizing the new cost function $E'(\boldsymbol{A})$. Global optimization [13], [14] are, recently, employed to provide a training method for neural networks to avoid the convergence being trapped in undesired local minima. For instance RS [10] and SA [11] are stochastic type global minimization methods which allow any downhill and uphill movements for escaping from local minima and hence the search converges to a global minimum at the end. GA [12], based on the computational model of evolution, i.e., the survival of the fittest strategy, is another mechanism of global optimization to determine connectivity in neural networks. In our study, we employ three hybrid approach algorithms, i.e., hybrid of LS/RS methods, hybrid of LS/SA methods, and hybrid of LS/GA methods, for learning neural based crowd monitoring. Those HGL algorithms are summarized as follows.

1) Algorithm Initialization.
   a) The weights matrix Ai of the network are randomly initialized with the values between -0.5 and 0.5.
   b) For RS method: Set σ of the Gaussian noise matrix $N(0,\sigma)$ by the preset value.
   c) For SA method:
      - Set σ of the Gaussian noise matrix $N(0,\sigma)$ by the preset value.
      - Set r_T: temperature reduction factor, T_o: initial temperature, N_ϵ: # times ϵ tolerance is achieved before termination, N_S: # times through function before noise level adjustment, N_T: # times through N_S loop before T reduction.
   d) For GA method: Set LB_{ji}: lower bound for element j and i, UB_{ji}: upper bound for element j and i, $G_{\max}$: maximum number of generations, α: a shape parameter, N: number of population.
2) Evaluate matrix $\boldsymbol{B}$ in (6).
3) If learning error $E(\boldsymbol{A},\boldsymbol{B})$ in (4) is smaller than the specified value, the training is completed.
4) For RS method:
   a) perform $\boldsymbol{A} = \boldsymbol{A}_i + N(0,\sigma)$.
   b) evaluate matrix $\boldsymbol{B}$ in (6) and learning error $E'(\boldsymbol{A},\boldsymbol{B})$ in (4).
   c) perform $\sigma = \sigma(1 - \mathrm{dec})$.
   d) If $E(\boldsymbol{A},\boldsymbol{B}) > E'(\boldsymbol{A},\boldsymbol{B})$, then $E(\boldsymbol{A},\boldsymbol{B}) = E'(\boldsymbol{A},\boldsymbol{B})$, $\sigma = \sigma_{\mathrm{bound}}$ and $\boldsymbol{A}_i = \boldsymbol{A}$.

   For SA method:
   a) perform $\boldsymbol{A} = \boldsymbol{A}_i + N(0,\sigma)$.
   b) evaluate matrix $\boldsymbol{B}$ in (6) and learning error $E(\boldsymbol{A},\boldsymbol{B})$ in (4).
   c) If $E'(\boldsymbol{A},\boldsymbol{B}) \geq E(\boldsymbol{A},\boldsymbol{B})$ then apply Metropolis criteria: $p = \exp(E'(\boldsymbol{A},\boldsymbol{B}) - E(\boldsymbol{A},\boldsymbol{B})/T)$. If p is greater than a uniformly distributed random number, then $\boldsymbol{A}_i = \boldsymbol{A}$ and $E(\boldsymbol{A},\boldsymbol{B}) = E'(\boldsymbol{A},\boldsymbol{B})$, otherwise reject $\boldsymbol{A}$.
   d) If $E(\boldsymbol{A},\boldsymbol{B}) > E'(\boldsymbol{A},\boldsymbol{B})$, then $E(\boldsymbol{A},\boldsymbol{B}) = E'(\boldsymbol{A},\boldsymbol{B})$, $\sigma = \sigma_{\mathrm{bound}}$ and $\boldsymbol{A}_i = \boldsymbol{A}$.
   e) repeat step 3 until N_S times through function before noise level adjustment.
   f) perform $\boldsymbol{A} = \boldsymbol{A}_i + N(0,\sigma)$.
   g) repeat step 3 until N_T times through N_S loops before T reduction.
   h) Set $\boldsymbol{A}_i = \boldsymbol{A}$ and $T = r_T \cdot T$.

   For GA method:
   a) Employ the selection scheme by the roulette wheel selection method that the probability, P_i, for each element is defined by:

$$P[\text{element } i \text{ be chosen}] = \frac{E_i}{\sum_{j=1}^{\mathrm{PopSize}} E_j},$$

   where E_i equals the fitness function.
   b) Create nonuniform mutation operator as follow:

$$m_i' = \begin{cases} m_i + (UB_i - m_i) f(G) & \text{if } r_1 < 0.5, \\ m_i - (m_i + LB_i) f(G) & \text{if } r_1 \geq 0.5,, \\ m_i, & \text{otherwise} \end{cases}$$

   where

$$f(G) = \left(r_2 \left(1 - \frac{G}{G_{\max}} \right) \right)^{\alpha},$$

   r_1, r_2 are a uniform random number between (0,1) and G is the current generation.
   c) Set $\boldsymbol{A} = \boldsymbol{M}'$ and repeat item (i) until N population.

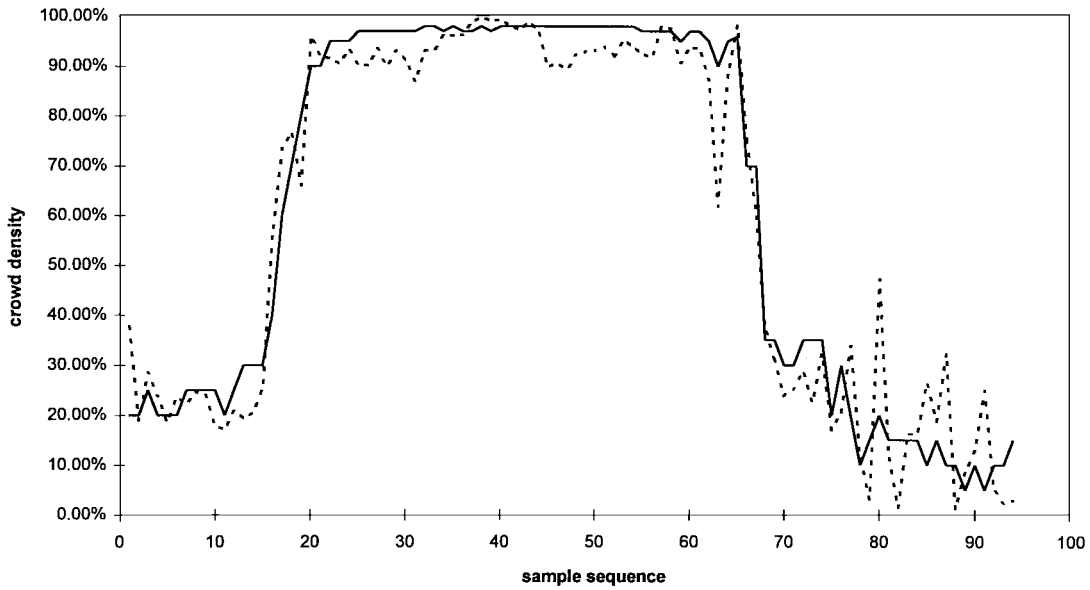


Fig. 8. Estimated results of the neural network trained by the hybrid of least-squares and random search algorithm (dashed line) versus real crowd densities (solid line) over 95 samples sequence.

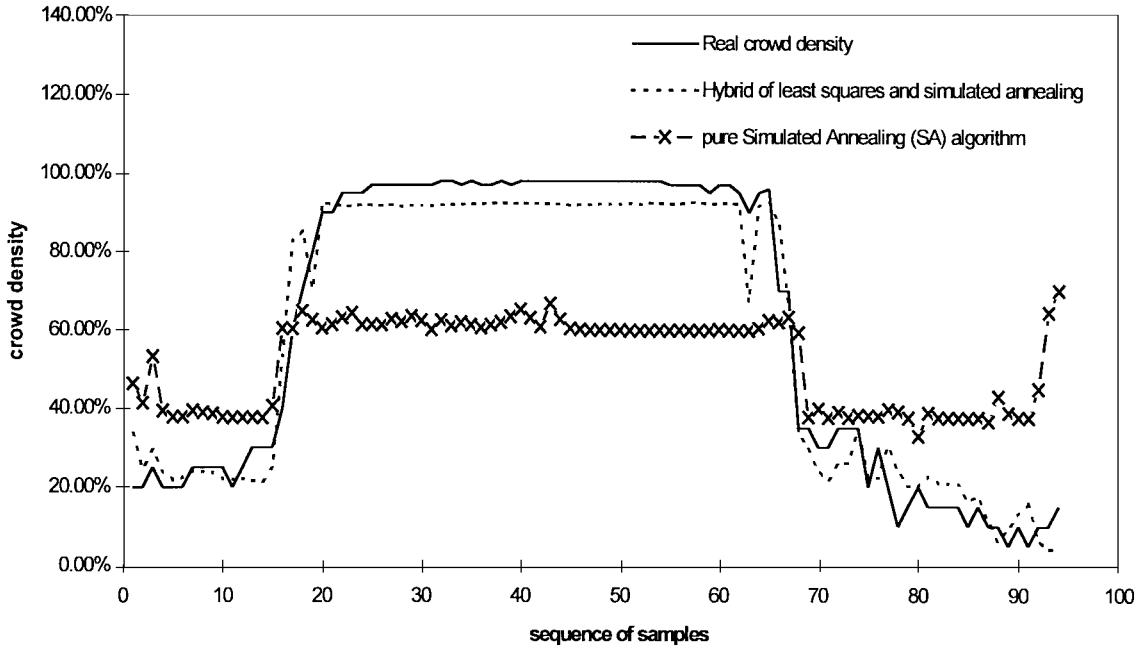


Fig. 9. Estimated results of the neural network trained by the pure SA and the hybrid of least-squares and simulated annealing algorithm versus real crowd densities over 95 samples sequence.

d) evaluate matrix $\boldsymbol{B}$ in (6) and learning error $E'(\boldsymbol{A}, \boldsymbol{B})$ in (4).
e) If $E'(\boldsymbol{A}, \boldsymbol{B}) < E(\boldsymbol{A}, \boldsymbol{B})$ then $\boldsymbol{A}_i = \boldsymbol{A}$ and $E(\boldsymbol{A}, \boldsymbol{B}) = E'(\boldsymbol{A}, \boldsymbol{B})$.

5) Go to step 3.

IV. Experimental Results and Discussion on Crowd Estimation

In this section, experimental results of the neural based crowd estimation by hybrid global learning (HGL) algorithm are presented. The system validations were performed on estimating of sequences of CCTV images acquired in a MTR station of Hong Kong. The performance is investigated and compared for three different types of hybrid algorithms. The results refer to data coming from a single CCTV, installed in a platform area of the underground station. In order to collect wide and representative sets of CCTV images at different scenario, acquisition of CCTV images were performed at different times of days; rush hours and nonrush hours at different weekdays and weekends. Sequence of images capturing from the CCTV were sampled at a frequency of about 1 Hz. In our approach, three indexes were obtained by the feature extraction procedure on each image. Afterward, these indexes are modeled by the neural networks for estimating the crowd density. The neural network topology is chosen as three input, 15 hidden, and one output neurons and the training processes were terminated at 1000 epochs. The performance of those three hybrid algorithms are compared and summarized as in Table I. It can be concluded that the learning rate

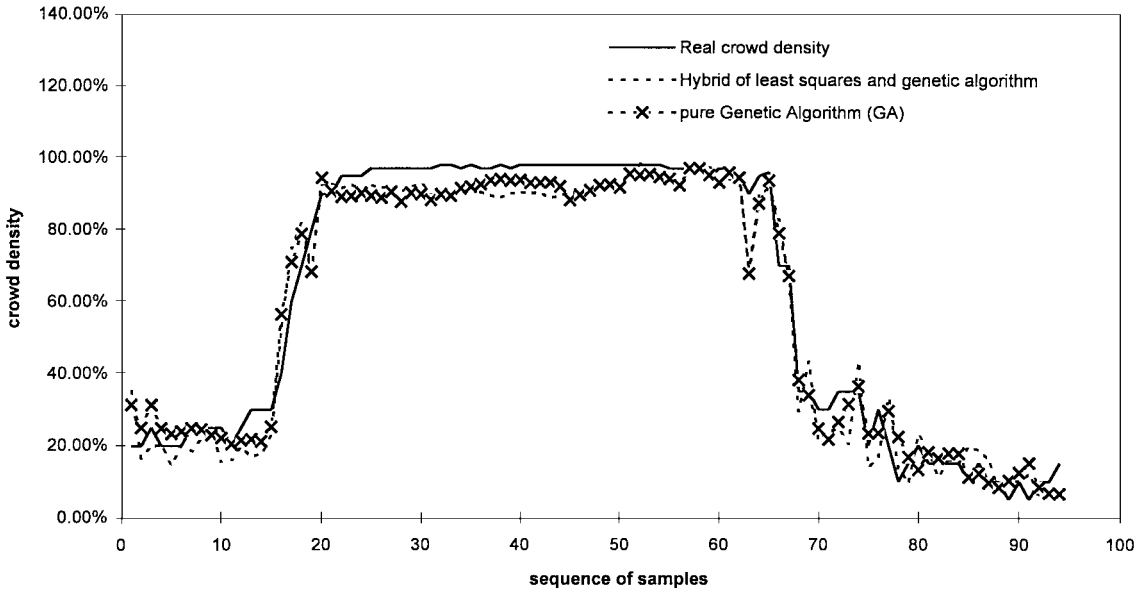


Fig. 10. Estimated results of the neural network trained by the pure GA and the hybrid of least-squares and genetic algorithm versus real crowd densities over 95 samples sequence.

TABLE I
STATISTICAL RESULTS OF THE THREE HYBRID ALGORITHMS FOR THE APPLICATION OF CROWD ESTIMATION

Algorithm	CPU running time for learning (mins)	Training Root Mean Squares Error	Validation Root Mean Squares Error	Estimation accuracy in percentages
Hybrid of least squares and random search	2.02	0.11792	0.088	90.72%
Hybrid of least squares and SA	197.5	0.1119	0.0668	94.36%
Hybrid of least squares and GA	75.3	0.11097	0.0731	93.89%

required for the hybrid of LS/RS algorithm is the fastest attributed to the less computation steps for the RS method. But lower estimation accuracy compared with LS/SA and LS/GA algorithms is obtained. Also, from Table I, it shows that the hybrid of LS/SA and LS/GA algorithms provide more robust and reliable performance than that of the RS method. Fig. 8 compares the performance obtained by the hybrid of LS and RS algorithm under 95 test sequences at rush hour. Besides, Figs. 9 and 10 also present the results yielded by the hybrid of LS method with SA as well as GA and compared with the pure simulated annealing and genetic algorithms. We conclude that the results yielded by the HGL algorithms almost coincide with the real density. On average, over 90% accuracy of amongst these three algorithms are obtained. These results are significantly promising and also meet the end-user's accuracy and robustness requirements.

V. CONCLUSION

This paper presents a neural based crowd estimation system for MTR station in Hong Kong based on HGL algorithm, which include the hybrid of the least-squares method and different types of global optimization, such as, random search, simulated annealing and genetic algorithm. The major novel contribution of the HGL algorithms are its capability of escaping from local minima by the global search characteristic and exhibiting a fast convergence speed by the least-squares computation. Prior to neural computation, the application of image processing technique to the original images captured from CCTV of the platform was essential for extracting the corresponding features. These features indexes are modeled and are used to estimate the crowd density of the platform via a neural network at the corresponding time step. Promising results of over 90% estimation accuracy are obtained. The developed system is able to facilitate the crowd monitoring at the station platform. As a result, the system output can be used to alert the operators automatically.

ACKNOWLEDGMENT

The authors would like to thank Mass Transit Railway Corporation, Hong Kong, for granting access to their underground stations and resources. The authors would also like to thank the anonymous reviewers for their constructive comments and suggestions.

REFERENCES

[1] P. Jukkala, J. Ylinan, and A. Raisanen, "Use of a millimeter wave radiometer for detecting pedestrian and bicycle traffic," in *Proc. 17th European Microwave Conf.*, Rome, Italy, Sept. 1987, pp. 585–589.
[2] S. S. Mudally, "Novel computer-based infrared pedestrian data-acquistion system," *Electron. Lett.*, vol. 15, pp. 371–371, June 1979.
[3] A. C. Davies, J. H. Yin, and S. A. Velastin, "Crowd monitoring using image processing," *Electron. Commun. Eng. J.*, pp. 37–47, Feb. 1995.
[4] C. S. Regazzoni, A. Tesei, and V. Murino, "A real-time vision system for crowding monitoring," in *Proc. IECON'93*, 1993, pp. 1860–1864.
[5] ——, "Distributed data fusion for real-time crowding estimation," *Signal Process.*, vol. 53, pp. 47–63, 1996.
[6] F. Cravino, M. Dellucca, and A. Tesei, "DEKF system for crowding estimation by a multiple-model approach," *Electron. Lett.*, vol. 30, pp. 390–391, Mar. 1994.
[7] C. T. Leung and T. W. S. Chow, "A hybrid global learning algorithm based on global search and least-squares techniques for backpropagation networks," in *Proc. IEEE ICNN'97*, vol. 3, pp. 1890–1893, 1997.
[8] S. Y. Cho and T. W. S. Chow, "A fast heuristic global learning algorithm for multilayer neural networks," *Neural Process. Lett.*, vol. 9, pp. 177–187, 1998.
[9] S. Y. Cho and T. W. S. Chow, "Training multilayer neural networks using fast global learning algorithm—Least squares and penalized optimization methods," *Neurocomputing*, vol. 25, pp. 115–131, 1999.
[10] N. Baba, "A new approach for finding the global minimum of error function of neural networks," *Neural Networks*, vol. 2, pp. 367–373, 1989.

[11] W. L. Goffe, G. D. Ferrier, and J. Rogers, "Global optimization of statistical functions with simulated annealing," *J. Econometrics*, vol. 60, pp. 65–99, 1994.
[12] C. R. Houck, J. A. Joines, and M. G. Kay, "A genetic algorithm for function optimization: A Matlab implementation," submitted for publication.
[13] A. Torn and A. Zillinskas, *Global Optimization*. Berlin, Germany: Springer-Verlag, 1987.
[14] R. Horst and P. M. Pardalos, *Handbooks of Global Optimization*. Amsterdam, The Netherlands: Kluwer, 1995.

Temporal Knowledge Representation and Reasoning Techniques Using Time Petri Nets

Woei-Tzy Jong, Yuh-Shin Shiau, Yih-Jen Horng, Hsin-Horng Chen, and Shyi-Ming Chen

***Abstract*—In this paper, we present temporal knowledge representation and reasoning techniques using time Petri nets. A method is also proposed to check the consistency of the temporal knowledge. The proposed method can overcome the drawback of the one presented in [16]. It provides a useful way to check the consistency of the temporal knowledge.**

***Index Terms*—Knowledge representation, rule-based system, temporal knowledge, time Petri nets.**

I. Introduction

The concept of time plays a very important role in our lives. In order to solve the temporal knowledge representation and reasoning problem, developing a system that can store and manipulate the knowledge about time is necessary. In [1], Allen described 13 kinds of relations of time, where each of the 13 relations represents the order of two time intervals. In [16], Yao pointed out that there are mainly two kinds of representation and reasoning schemes for temporal information, i.e., Dechter's linear inequalities [6] to encode metric relations between time points and Allen's temporal calculus [1]. Each scheme has its advantages and disadvantages. In [12], Kautz *et al*. introduced a model to integrate two schemes for temporal reasoning in order to benefit from the advantages of each scheme. In [8], Dutta presented an event-based fuzzy temporal logic. It can determine effectively the various temporal relations between uncertain events or their combinations. In [7], Deng *et al*. presented a G-Net for knowledge representation and reasoning. In [5], we presented a fuzzy Petri net model (FPN) to represent the fuzzy production rules of rule-based systems and presented a fuzzy reasoning algorithm to deal with fuzzy reasoning in rule-based systems. However, the models presented in [5] and [7] cannot be used for temporal knowledge representation. In [16], Yao presented a model based on time Petri nets for handling both qualitative and quantitative temporal information. In [4], we pointed out that the method presented in [16] has a drawback in checking the consistency of temporal knowledge.

Manuscript received February 6, 1998; revised January 30, 1999. This work was supported in part by the National Science Council, R.O.C., under Grant NSC 86-2213-E-009-018.

W.-T. Jong, Y.-S. Shiau, Y.-J. Horng, and H.-H Chen are with the Department of Computer and Information Science, National Chiao Tung University, Hsinchu, Taiwan, R.O.C.

S.-M. Chen is with the Department of Electronic Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan, R.O.C.

Publisher Item Identifier S 1083-4419(99)05278-4.

In this paper, we present a method to describe the relationships between states and events using time Petri nets for temporal knowledge representation and reasoning. We also present an algorithm to check the consistency of temporal knowledge. The proposed method can overcome the drawback of the one presented in [16].

The rest of the paper is organized as follows. In Section II, we introduce the basic concepts and definitions of time Petri nets. The temporal knowledge representation techniques using time Petri nets are also presented in Section II. In Section III, we present some operations between time intervals and between paths in a time Petri net. In Section IV, we present an algorithm to check the consistency of temporal knowledge. The conclusions are provided in Section V.

II. Time Petri Nets

In this section, we introduce the basic concepts of time Petri nets. A time Petri net is a bipartite directed graph which contains two types of nodes, i.e., places and transitions, where circles represent places and bars represent transitions. There are several definitions of time Petri nets [11], [16]. A time Petri net is a ten-tuple $(S, E, P, T, B, F, M_0, \alpha, \beta,$ SIM), where

S	finite set of states, $S = \{S_1, S_2, \cdots, S_n\}$;
E	finite set of events, $E = \{E_1, E_2, \cdots, E_m\}$, where each event is associated with a transition;
P	finite set of places, $P = \{P_1, P_2, \cdots, P_n\}$, where each place is associated with a state;
T	finite set of transitions, $T = \{t_1, t_2, \cdots, t_m\}$, where each transition is associated with a time interval;
B	backward incidence function, B: $T \times P \rightarrow N$, where N is the set of nonnegative integers;
F	forward incidence function, F: $T \times P \rightarrow N$;
M_0	initial marking function M_0: $P \rightarrow N$;
α	mapping function from places to states, α: $P \rightarrow S$;
β	mapping function from events to transitions, β: $E \rightarrow T$;
SIM	mapping function called static interval mapping function, SIM: $T \rightarrow Q^*$, where Q^* is a time interval.

In a time Petri net, each transition is associated with a time interval $[a, b]$, where a is called the static Earliest Firing Time, b is called the static Latest Firing Time, and $a \leq b$, where

1) a $(0 \leq a)$ is the minimal time that must elapse, starting from the time at which the transition is enabled until the transition can fire;
2) b $(0 \leq b \leq \infty)$ represents the maximum time during which the transition is enabled without being fired.

The values of a and b are relative to the moment the transition is enabled. If the transition is enabled at time λ, then the transition cannot be fired before time $\lambda + a$, and the transition must be fired before time $\lambda + b$.

In a time Petri net, a place may contain tokens. A time Petri net with some places containing tokens is called a marked time Petri net. For example, Fig. 1 shows a marked time Petri net, where events E_1, E_2, E_3, and E_4 are associated with time intervals, $[t_{11}, t_{12}], [t_{21}, t_{22}], [t_{31}, t_{32}]$, and $[t_{41}, t_{42}]$, respectively. An arc from a place to a transition defines the place to be the input (backward incidence) place of the transition. An arc from a transition to a place defines the place to be the output (forward incidence) place of the transition. A transition is enabled if and only if each of its input places has a token. When a transition is enabled, it may be fired. When a transition fires, all tokens are removed from its input places,