International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

IJIEMR Transactions, online available on 06th Dec 2022. Link :http://www.ijiemr.org/downloads.php?vol=Volume-11&issue=ISSUE-12

DOI: 10.48047/IJIEMR/V11/ISSUE 12/50

Title COUNTING NUMBER OF HUMAN IN THE IMAGE VIDEO USING COMPUTER VISION TECHNIQUES

Volume 11, Issue 12, Pages: 375-387

Paper Authors

Dr. PAMBALA NAGESWARARAO , K GANESH SAI SUDARSAN , A SRIKANTH

COUNTING NUMBER OF HUMAN IN THE IMAGE VIDEO USING COMPUTER VISION TECHNIQUES

Dr. PAMBALA NAGESWARARAO [1], K GANESH SAI SUDARSAN [2], A SRIKANTH [3]

Associate professor [1], Assistant Professor[2], Assistant Professor[3]

CSE Department, Sri *Mittapalli College* of *Engineering*, *Guntur*, *Andhra Pradesh*-522233

ABSTRACT-

The paper is devoted to the problem of estimating the number of people visible in a camera. It uses as features a portion of foreground pixels in each cell of a rectangular grid. Using the above features and data mining techniques allowed reaching accuracy up to 85% for exact match and up to 95% for plus-minus one estimate for an indoor surveillance environment. The architecture of a real-time people-counting estimator is suggested. The results of the analysis of experimental data are provided and discussed.

Keywords : Multi-camera surveillance, video surveillance, counting people.

1. INTRODUCTION

The proliferation of video cameras in public places such as airports, train stations, streets, parking lots, hospitals, governmental buildings, hotels, shopping malls, etc. has created the infrastructure that allows the development of security and business applications. Surveillance for threat detection, monitoring sensitive areas to detect unusual events, tracking customers in retail stores, controlling and monitoring movements of assets, and monitoring elderly and sick people at home are just some of the applications that require the ability to automatically detect, recognize and track people and other objects by analyzing multiple streams of often unreliable and poorly synchronized sensory data. Counting people is an important task in automatic surveillance systems. It can serve as a subtask in multiple stage processing or can be of primary interest. The robust estimate of people count could improve low level procedures, such as blob extraction or it can provide answers to questions such as, how many people entered a room between times t1 and t2 or how many people are inside the room at time t3? The related problem is to estimate the density of people in a crowd in places such as a subway platform or a street. Usually the output is a "fill rate" of the space expressed as a percentage. To solve this problem, different types of location sensors could be used. We will not attempt to exhaustively enumerate all the types of location sensors (see [1] for a survey). However, we can mention some special sensors that are used mostly in indoor environments and for which there are many commercial solutions already in place: electro-optical, thermic and passive-optics directional sensors are just a few. These sensors are placed in the entrances of buildings, rooms or vehicles. They can detect the passage and direction of a person and report with high accuracy the count of people that entered or left the facility, but they come at the cost of having to install proprietary hardware (the sensors and the data collection devices) and software to analyze the data. In contrast to this, video-based sensors – or video cameras – have increasingly become a commercial off-the-shelf product for surveillance purposes. Additionally, existing video cameras used for surveillance can be leveraged to perform more tasks such as automatic tracking, behavior recognition and crowd estimation, among others. This takes us back to our original goal: "estimating the count of people in an image". In the past years there has been a bulk of research work in the area of image processing with the objective of obtaining more accurate and reliable people-count estimations. Imagine a camera viewing a concourse in an airport or a platform at a railway station; a webcam with a view to a side walk in a busy street or a set of cameras covering different areas in a shopping mall. All these cameras provide a large amount of images that can be used to estimate the count of people at different times, but the conditions of the environment in which the cameras are embedded (sudden luminosity changes, camera location and angle of view, etc.) challenge the accuracy of the estimations. Moreover, the approach taken to estimate the count of people in a given image varies depending on: a) the spatial characteristics of the area under surveillance (confined vs. open area); b) features extracted from the image; c) expected response time (real time requirements vs. offline processing); and d) the maximum size of the crowd. An intuitive solution to the problem of estimating the size of a crowd in an image will be, literally, to obtain a head count. While this would be a tedious, but yet feasible, task for a human it certainly is a difficult problem for an automatic system. That is exactly the problem tackled in [2], where wavelets are used to extract head-shaped features from the image. Further processing uses a support vector machine (SVM) to correctly classify the feature as a "head" or "something else" and ultimately apply a perspective transform to account for the distance to the camera.

A similar idea is used in [3], where a face detection program is used to determine the person count. Unfortunately, as pointed out by its authors, this method is affected by the angle of view at which the faces are exposed to the camera. Additionally, images where a person's back is only visible will result in a poor estimation as well. Another approach has been suggested in [4], it aims to obtain an estimation of the crowd density, not the exact number of people. It requires a reference image – where no people are present – in order to determine the foreground pixels in a new image. A single layer neural network (NN) is fed with the features extracted from the new image (edge count and densities of the background and crowd objects) and the hybrid global learning (HGL) algorithm is used to obtain a refined estimation of the crowd density. This paper compares different classification algorithms for estimating the number of people in an image obtained from a video surveillance camera. Our approach differs from previous works in that we do not attempt to obtain and count specific features from the images (head-shaped objects in [2] or faces in [3]). We just exploit the correlation between the percentage of foreground pixels and the number of people in an image [5].

2. DATASETS

In order to determine how different classification algorithms would perform on both an outdoor environment with high traffic and an indoor environment with moderate and low traffic, we selected the following two sources:

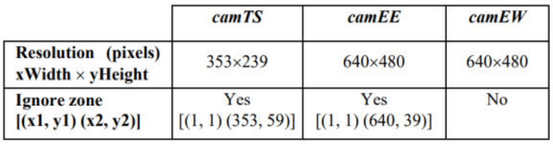
1) A publicly available webcam at Times Square.

2) Two webcams in the premises of the Accenture Technology Labs in Chicago.

The Times Square webcam, denoted herein as camTS, is located at the intersection of 46th Street and Broadway in New York City, NY and it streams video images through a publicly available URL [6]. In contrast, the remaining two webcams can only be accessed from within the Accenture intranet and are part of a larger set of webcams of an experimental surveillance system already in place. The two cameras chosen for the experiments have opposite views of an elevator area on the 36th floor. This area is the gateway to the offices of the Accenture Technology Labs located on the same floor, and the cameras that will be referred as camEE and camEW cover the elevators from the East and West respectively. All cameras capture snapshot images in the JPEG format. Table 1 summarizes the characteristics of these images for each camera. Here the “ignore zone” was not considered by the image preprocessing algorithms. It represents an area of the image that is not worth using and from which no features are extracted. It is expressed as a rectangle with two pairs of coordinates where the first coordinate is the upper-leftmost corner and the second one is the bottom-rightmost corner. The axis origin of the images is (1, 1) and is located in the upper-leftmost corner. For camTS, Table 1 indicates that all the pixels above y = 60 are ignored and this is also illustrated in Figure 1a. To create the datasets that were fed into the classification algorithms we followed a three-step process:

1) Capture the images and manually annotate them indicating the number of persons present in them.
2) Pre-process the original images to obtain the foreground pixels. This step creates binary images (black and white) where the white areas reflect the density of foreground pixels.
3) Finally, divide the binary images into grids of different sizes and obtain the percentage of foreground (white) pixels in each cell of the grid.

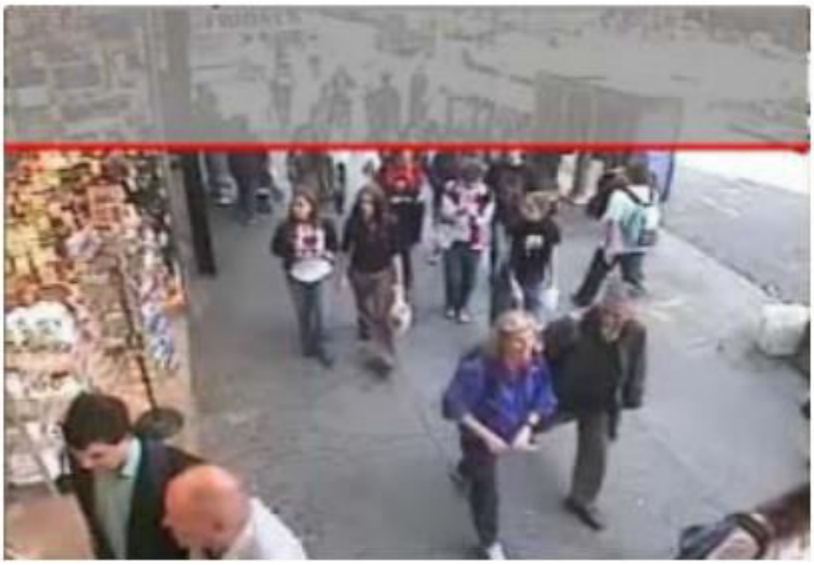
Table 1. Summary of characteristics for the three webcams analyzed (camTS = Times Square; camEE = Elevators, East view; camEW = Elevators, West view)



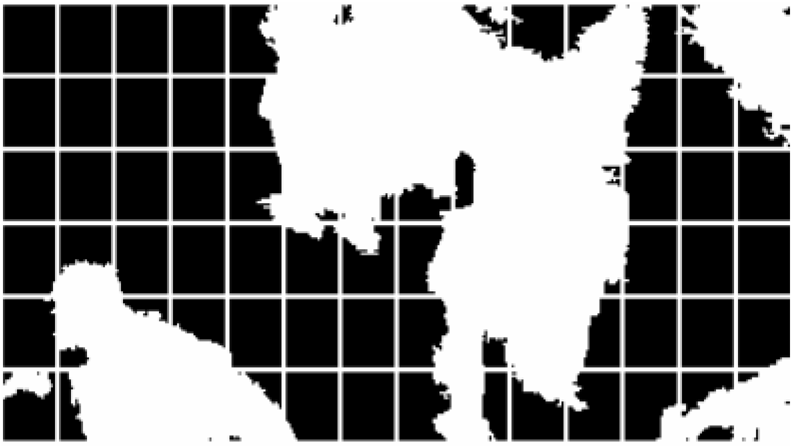
	camTS	*camEE*	*camEW*
Resolution (pixels) xWidth × yHeight	353×239	640×480	640×480
Ignore zone [(x1, y1) (x2, y2)]	Yes [(1, 1) (353, 59)]	Yes [(1, 1) (640, 39)]	No

IMAGE PRE-PROCESSING: The features we extracted are based on the density of foreground pixels. To extract the foreground pixels we used the well known median filter background modeling technique [7]. According to this technique each background pixel is modeled as median of a pool of images accumulated over some period of time and periodically updated by adding the current image and discarding the oldest one. This technique works well when each background pixel is occluded in less than 50% of images of the pool. To get the foreground pixels the background model is subtracted from the current image pixel by pixel, the absolute values of differences are summed and compared to a threshold. All pixels that have difference above the threshold are marked as foreground. Then morphological operations are applied to smooth the result. Figures 1b, 1d and 1f show the results of foreground pixel extraction. You can see the noise in the top right corner of the image in Figure 1b, which is caused by vehicles moving along the street.

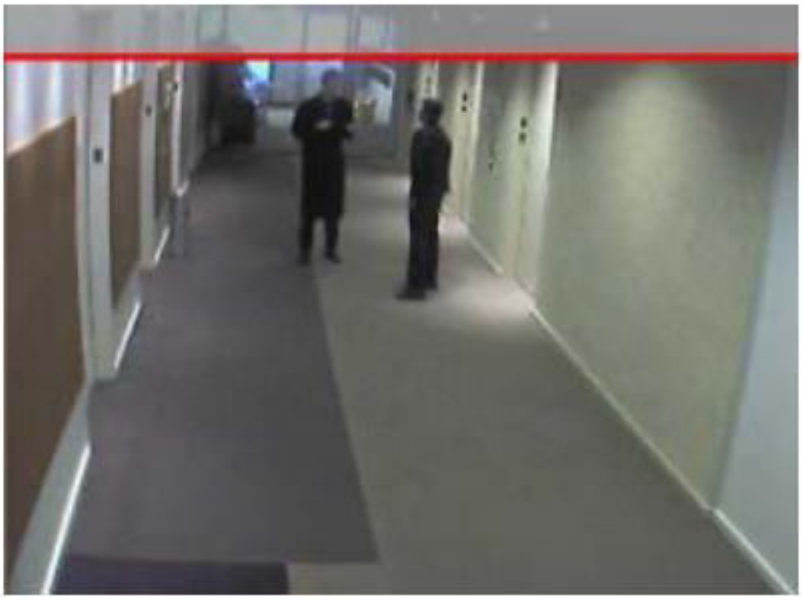
FEATURE EXTRACTION: For each camera, we created three grids of different size and applied them to the binary images. We obtained the ratio of foreground pixels in each cell of the grid by counting foreground pixels and then dividing the count by the area of the cell. Thus for each camera we created three different datasets. A record of a dataset has the image ID and N real values in the range from 0 to 1, which correspond to the proportion of foreground pixels in each cell. Figures 1b, 1d and 1f illustrate how the images were divided into a grid. Additionally, Table 2 shows the different grids implemented for the three cameras. It is important to note that, regardless of the grid applied to a camera, the datasets generated by this process will have the exact same number of records (rows). For a given camera, the difference between the datasets will be in the number of columns. The size of the cells is arbitrary and for our experiments we considered grid sizes that differ by a factor of two, double or half the size of the original. We wanted to learn how the sizes of the cells influence the accuracy of the classification algorithms. Figure 1 shows the layouts for the three cameras that have been used in the experiments.



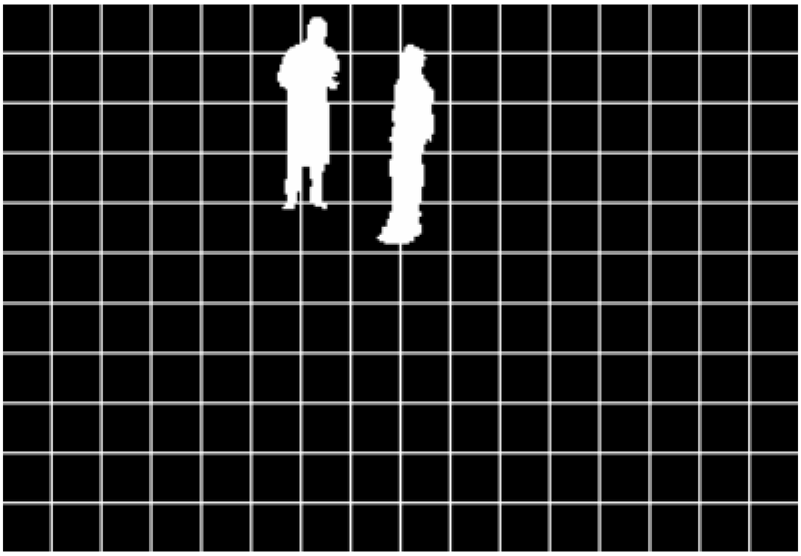
a) camTS captured image; ignore zone above y = 60



b) camTS binary image with foreground pixels; original grid of 83 cells; size of each cell is 25×30 pixels



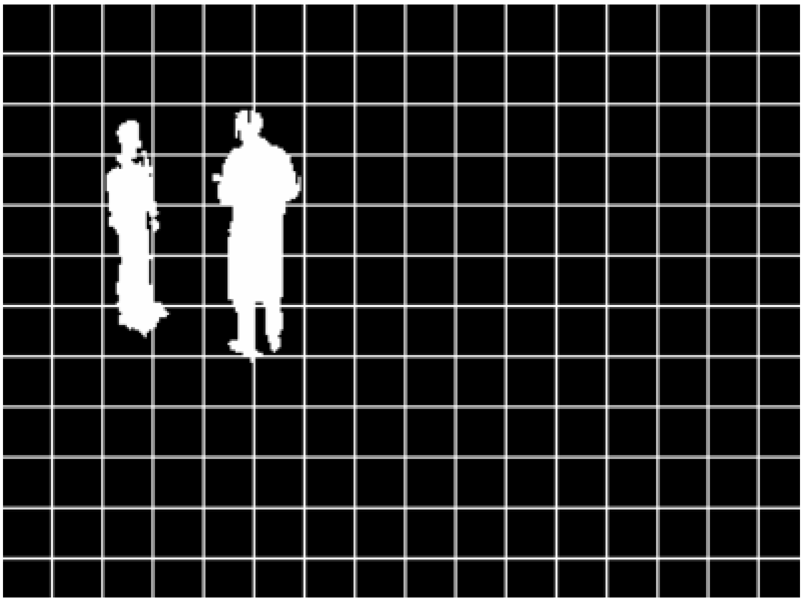
c) camEE captured image; Ignore zone above y = 40



d) camEE binary image with foreground pixels; grid of 176 cells of 40×40 pixels

e) camEW captured image (opposite view of camEE).

f)



g) camEW binary image with foreground pixels; grid of 192 cells of40×40 pixels

3. CLASSIFICATION ALGORITHMS

This section describes the characteristics of the algorithms used in our experiments and their parameters. For the sake of clarity, we would like to briefly explain how the algorithms were tested and what data they used. This digression will help understand how the datasets were actually used. In general, a dataset will have the layout that is presented in Table 3. For a given classification method, some records from the dataset are used to train it. The rest of the records are used to test it and the labels are compared to the values predicted by the algorithm. Since the label is the person count, we can measure the accuracy of the algorithm to estimate the count of persons in unseen images. By unseen we mean images that belong to the test set.

1) The input consists of N-element vectors (where N is the number of cells in the grid).

2) A radial basis layer that computes the distance of a new vector to the training input vectors. This generates a vector of probabilities of the element belonging to each class. This layer has a neuron for each vector in the training set.

3) A competitive layer that finds the maximum of these probabilities and generates a vector with a 1 for the chosen class and zeros for the other classes (e.g.: if there are 5 possible classes {1, 2, 3, 4, 5} and the new vector is classified as belonging to class = 2, this layer will output [0, 1, 0, 0, 0]). The number of neurons in this layer will be equivalent to the length of the output vector, i.e., the maximum count of persons in the dataset.

Support Vector Machines (SVM) classifies: objects of two different classes by finding a hyperplane that divides both sets and maximizes the distance between the plane and the closest data points from each class. Because of its search for a hyperplane, SVM is considered a linear classifier, but in a more rich feature space. In our experiments, we implemented an SVM for the zero-person detection task (binary classification of “zero” or “one or more” persons). As it will become clear later, some of the classification methods implemented were used for different tasks. The next section provides details on when these methods were used and what were the results of their application.

4. RESULTS

The methodology implemented to compare the above mentioned classifiers follows the traditional sequence of steps used in creating and evaluating supervised learning algorithms. First, the data collection was performed to acquire images for the three cameras. The images were then manually annotated to indicate the number of people visible in each image creating of the ground truth data for each dataset. Each classifier was then trained and tested with different datasets and, unless otherwise noted, all the experiments detailed in this section were repeated one hundred times to attain statistical significance. At each iteration, 70% of the records of the dataset were randomly selected to train the classifier and the remaining 30% was used to test it. Then the labels in the test set were compared to the output generated by the classifier and the accuracy of the classifier was estimated. The records in the training and test set were randomly selected using a uniform distribution. Before an experiment started, all the records in the dataset were shuffled to avoid any side effects due to the chronological order of the images. The experiments were run using Matlab® 7.1 on a Pentium 4, 2.8GHz CPU with 512 Mb of RAM. Two third party toolboxes were used for SVM [11] and k-nearest neighbor [12].

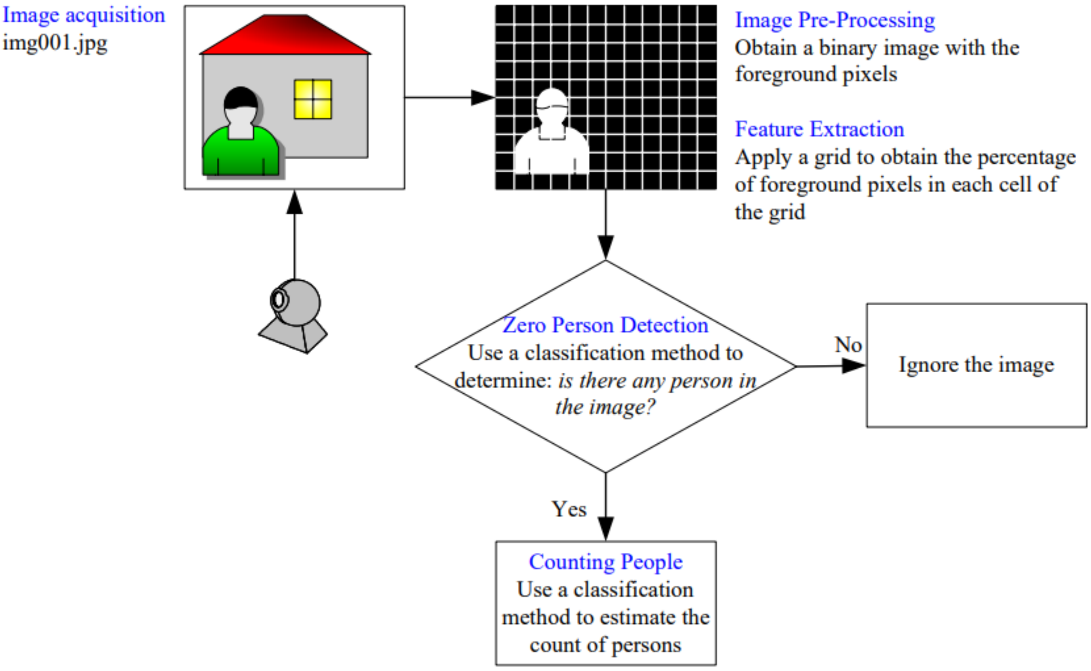


Image acquisition
img001.jpg
Image Pre-Processing
Obtain a binary image with the foreground pixels
Feature Extraction
Apply a grid to obtain the percentage of foreground pixels in each cell of the grid
Zero Person Detection
Use a classification method to determine: *is there any person in the image?*
No
Ignore the image
Yes
Counting People
Use a classification method to estimate the count of persons

Figure is Architecture of a real-time person counter

5. SUMMARY AND FUTURE WORK

We described an approach to estimating the number of people in an image using rather simple features – a portion of foreground pixels in rectangular areas that cover the image. It turns out that these simple features fed into modern machine learning algorithms can produce very impressive results and can be used for creating a real-time people counting estimator for surveillance applications. Our count estimator was divided into two tasks: 1) zero person detection and 2) counting people in an image with one or more persons present in it. For each task we compared different classifiers based on their accuracy and execution times. The next step will be to create a prototype to count persons in real-time with an architecture similar to the one described in Figure 3. Additionally, there are several directions of future research. First, all our recognizers do not take into account the interdependencies between consecutive frames. They assume that features are independent in consecutive frames. To conform to these requirements we shuffled the dataset before applying the classifiers. But in an actual real-time implementation, our person counter will receive frames in chronological order and the system should be able to exploit this. One possibility could be to use recurrent neural networks. For example, in [13] Generalized Locally Recurrent Probabilistic Neural Networks (GLRPNNs) were used in the area of speaker verification process to take into consideration the correlations between speech frames. GLRPNNs make use of the past values of the outputs to determine the probability, of a current element, of belonging to a specific class. In our case, the spatial correlation between frames can be used by a GLRPNN to improve the accuracy of the classification and this will be an interesting direction to pursue. Another approach that we can take for further evaluation is the creation of adaptable grids. Figures 7a and 7b show the elevator area and a grayscale image that presents the frequencies of foreground pixels accumulated from all images of the dataset. A preliminary analysis of the results shows that people tend to follow certain frequent paths when exiting or entering an elevator. This can be used to create cell grids of different sizes that will avoid the near-far effect caused when a person close to a camera occupies more area than the same person far from it [5]. Additionally, the areas of the image that can be sources of noise can be excluded from the consideration. To finalize, we can list some limitations of our current work. Since the features we extract from the images are simply the foreground pixels, our estimator is very sensitive to the set of images used during the training phase. For example, let's assume a person pushing a food cart walks in front of a camera and this situation has never occurred before. Most likely, the estimator will report the number of persons as being 2 or 3 instead of just 1 (because there are more foreground pixels in the image than when just 1 person walks without a food cart). On the other hand, if these images were added to the training set the estimator will improve its accuracy. It will still remain oblivious to the fact that the object in question is a food cart but it will simply remember that certain distribution of foreground pixels was labeled as "1 person". Later, when an image with a person pushing a similar food cart arrives, it will be correctly classified.

REFERENCES

[1] Jeffrey Hightower, Gaetano Borriello, Location Systems for Ubiquitous Computing, Computer, August 2001 Vol. 34, No. 8, pp. 57-66.

[2] Sheng-Fuu Lin, Jaw-Yeh Chen, Hung-Xin Chao, Estimation of Number of People in Crowded Scenes Using Perspective Transformation, IEEE Transactions on Systems, Man and Cybernetics, November 2001, Part A, Vol. 31, Issue 6, pp. 645-654. [3] L. Sweeney and R. Gross. Mining Images in PubliclyAvailable Cameras for Homeland Security. AAAI Spring Symposium on AI Technologies for Homeland Security. Palo Alto, 2005.

[4] Siu-Yeung Cho, T.W.S. Chow, Chi-Tat Leung, A neuralbased crowd estimation by hybrid global learning algorithm, IEEE Transactions on Systems, Man and Cybernetics, August 1999Part B, Vol. 29, Issue 4, pp. 535- 541.

[5] A. C. Davies, J. H. Yin, and S. A. Velastin, Crowd monitoring using image processing, Electronics & Communication Engineering Journal, February 1995, Vol. 7, Issue 1, pp. 37-47.

[6] Times Square webcam URL: http://www.earthcam.com/usa/newyork/timessquare/

[7] Sen-ching S.Cheung, Chandrika Kamath., Robust techniques for background subtraction in urban traffic video. Proc. of SPIE, Visual Communications and Image Processing 2004, S. Panchanathan, B. Vasudev (Eds), January 2004, Vol. 5308, pp. 881-892.

[8] M. Boukadoum, A. Bensaoula, and D. Starikov, A Neural Network-based System for Live Bacteria Detection, Proc. (403) Artificial Intelligence and Applications, 2003.

[9] R. Bates, M. Sun, M.L. Scheuer, R. Sclabassi, Seizure Detection by Recurrent Backpropagation Neural Network Analysis, 4th International Symposium on Uncertainty Modeling and Analysis, (2003), p. 312-320.

[10] Donald F. Specht, Probabilistic neural networks. Neural Networks, January 1990, Vol. 3, Issue 1, pp. 109-118.

[11] J.A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle, Least Squares Support Vector Machines, World Scientific, Singapore, 2002 (ISBN 981- 238-151-1).

[12] David G. Stork, Elad Yom-Tov, Computer Manual in MATLAB to Accompany Pattern Classification, 2nd Edition, April 2004, (ISBN: 0-471-42977-5).

[13] Ganchev, T. Fakotakis, N. Tasoulis, D.K. Vrahatis, M.N., Generalized locally recurrent probabilistic neural networks for text-independent speaker verification, IEEE International Conference on Acoustics, Speech, and Signal Processing, 2004.