Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

Image and Vision Computing 22 (2004) 563–570

www.elsevier.com/locate/imavis

A real-time system for monitoring of cyclists and pedestrians

Janne Heikkilä*, Olli Silvén

Machine Vision Group, Infotech Oulu and Department of Electrical and Information Engineering, PO Box 4500, FIN-90014 University of Oulu, Finland

Abstract

Camera based systems are routinely used for monitoring highway traffic, supplementing inductive loops and microwave sensors employed for counting purposes. These techniques achieve very good counting accuracy and are capable of discriminating trucks and cars. However, pedestrians and cyclists are mostly counted manually. In this paper, we describe a new camera based automatic system that utilizes Kalman filtering in tracking and Learning Vector Quantization for classifying the observations to pedestrians and cyclists. Both the requirements for such systems and the algorithms used are described. The tests performed show that the system achieves around 80–90% accuracy in counting and classification.

Keywords: Human tracking; Traffic counting; Target classification

1. Introduction

The analysis of traffic flows has traditionally been based on inductive and microwave sensors. These technologies are not, however, capable of reliable monitoring of pedestrians and cyclists for two reasons. First, the reliability of inductive sensors depend on the materials in the vehicle, and second, the structure of pedestrian traffic is weak and does not focus on easily recognizable lanes. It is necessary to find an alternative technique for pedestrian traffic analysis due to the problems of the traditional approaches.

Traffic flow statistics are mainly used by road administration which tries to resolve the needs for new pavements, cycle paths and tunnels, for instance. Currently, such surveys are performed manually. However, manual counting is tedious and typically continues only for a short period of time which makes the results quite unreliable.

A natural choice for monitoring pedestrians and cyclists automatically is to use visual perception. However, there are still quite few camera based systems developed for people detection and tracking. A well known undertaking in this area was the ESPRIT project PASSWORDS [2]. Another visual surveillance application is W4 [6], which is a real time system for determining interactions between people. Most of the systems presented are based on an assumption of a stationary camera which greatly simplifies the problem of human detection, although solutions for pedestrian detection from moving platforms have also been suggested, for example, in Refs. [5] and [10]. Furthermore, a model based approach for pedestrian detection was proposed in Ref. [11], and a statistical framework for learning and recognizing human movements in Ref. [3].

In human tracking, there are many visual clues that can be utilized. These include color histograms [4], coherent connected regions or 'blobs' [12], and object contours [1]. Kalman filtering is probably the most commonly used algorithm for implementing the tracker, although recently Condensation algorithm [7] and mean shift algorithm [4] have shown to provide certain advantages especially in the presence of significant background clutter. Despite of the methodological advancements, many of the solutions proposed are computationally expensive and they make limiting assumptions about the target appearance and dynamics. Moreover, there is often a long way to go from a controlled laboratory environment to unpredictable outdoor scenes.

When the project for developing an automatic traffic monitoring system was started in 1994, there were no commercial systems available for that purpose. According to our knowledge there still exist no other systems capable of fulfilling even partly the following requirements that were specified by our user group:

* Corresponding author.
E-mail addresses: jth@ee.oulu.fi (J. Heikkilä), olli@ee.oulu.fi (O. Silvén).

0262-8856/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/j.imavis.2003.09.010

- counting of the volume of the traffic
- classification of the traffic into separate traffic classes (pedestrians, cyclists,…)
- route tracing
- counting accuracy better than 90%
- classification accuracy better than 85%
- portability and easy installation
- one week stand alone operation time
- transfer of data into a portable field terminal through a telephone network or via a memory card

In addition, the weather conditions set extra requirements on the hardware. The device should work in various temperatures (from -20 to $+30$ °C), in rain, snowfall, and wind. Also, adaptation to a wide illumination range with sudden changes is typically very difficult for camera based systems.

One of the main problems in developing a stand alone system is the power consumption. The limited capacity of the batteries and long operation time allows us to use only economic hardware solutions at the cost of processing speed. This also forced us to divide the monitoring problem into two parts: real-time tracking and off-line analysis.

The tracking unit is installed beside the road on a pole. It performs several image processing tasks to detect and track moving objects in the scene. The shape and the trajectory of the objects are stored in a removable memory card to wait for further processing. After several days of stand alone operation, the memory card is fetched and the data is transferred to a separate desktop computer equipped with analysis software. The objects are then classified to pedestrians and cyclists based on the features stored during the tracking phase. The architecture of the monitoring system is depicted in Fig. 1.

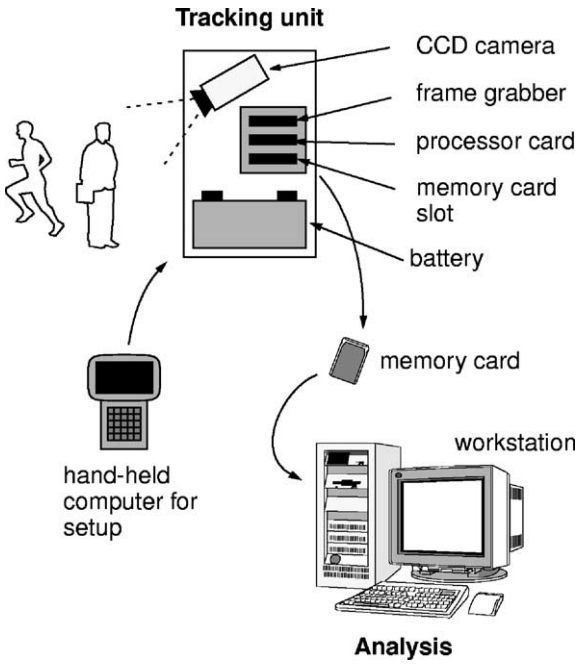


Fig. 1. System architecture.

In Section 2, we will describe the basic techniques used in the tracking unit including the image processing steps for feature extraction and the Kalman filter based tracking scheme. Section 3 introduces the Learning Vector Quantization (LVQ) based analysis for producing the final counting and classification result. The preliminary test results are given in Section 4.

2. The tracking unit

The main modules in the tracking unit are the CCD camera, frame grabber and Ampro 486/50 MHz processor card. The power consumption of the unit was optimized to around 7 W in full operation mode and 1 W in sleep mode. The sleep mode is used at night time when the overall lighting is not adequate for reliable observations.

The weight limit of the tracking device was 5 kg, including the used 20 Ah 6 V sealed lead acid battery. The selection of the battery technology was dictated by the temperature conditions ranging from -20 to $+30$ °C. The mean power consumption of the tracking unit is approximately 2.5–3 W of which roughly 1.5 W is taken by the camera that is always on, except during the sleep mode. The technology solutions were selected from among several alternatives, mostly based on power control and consumption characteristics.

The purpose of the tracking unit is to detect moving people from the video stream and collect appropriate data of their routes and shapes. Each event is saved and compressed with a time stamp in a removable memory card.

The images read from the frame grabber are decimated to 160 by 120 pixels with only luminance information preserved in order to reduce the computational load and to guarantee adequate frame rate (around 10 fps) for tracking. Each incoming frame goes through four successive image processing stages where the raw intensity data is reduced to a compact set of features which can be used as an input of the classifier. These four stages, motion detection, filtering, feature extraction and tracking, are described next.

2.1. Motion detection

Motion detection is started by computing a pixel based absolute difference between each incoming frame and an adaptive background frame $B(k)$. The pixels are assumed to contain motion if the absolution difference exceeds a predefined threshold level. As a result, a binary image is formed where active pixels are labeled with ‘1’ and non-active ones with ‘0’.

It is necessary to update the background image frequently in order to guarantee reliable motion detection. The basic idea in background adaptation is to integrate

the new incoming information into the current background image using the following first order recursive filter:

$$B(k+1) = (1-\alpha)B(k) + \alpha I(k) \quad (1)$$

where α is an adaptation coefficient. The larger it is, the faster new changes in the scene are updated to the background frame. However, α cannot be too large because it may cause artificial 'tails' to be formed behind the moving objects.

In our approach, α is kept small and the update process based on Eq. (1) is only indented for adapting to slow changes in overall lighting. Reactions to sudden changes are different. The activity of each pixel is monitored during several consecutive frames. The intensity values of those pixels that are active most of the time are directly copied from the latest $I(k)$ to $B(k)$. In this way we can adapt reasonably fast to new static objects appearing on the scene, like stationary cars, or to sudden changes in the illumination level.

In outdoor scenes, swinging branches of trees cause unwanted motion in the images. This kind of motion can be determined and eliminated based on its randomness. A simple but efficient method is to monitor the changes in activity during a few consecutive frames. Those pixels which tend to change their activity frequently are masked out from the binary image representing the motion detection result.

2.2. Filtering

The figures directly extracted from the resulting binary image are typically fragmented to multiple segments. In order to avoid this, a morphological closing operation with a 3-by-3 square kernel is applied to the image. As a result, small gaps between the isolated segments are erased and the regions are merged.

Morphological operations are often computationally expensive. However, binary closing (and opening) can be implemented at the register level as a combination of the shift and the logical AND and OR operations. With this technique, the closing operation can be applied to the whole image in real-time, also with moderate hardware.

After closing, the image may contain regions which are composed of background noise pixels. Due to the noise suppression described in Section 2.1, these regions are typically smaller than the actual motion regions that we are interested in. Therefore, we use in this stage a connected component analysis followed by region area detection. The regions with a smaller area than the predefined threshold are now discarded.

2.3. Feature extraction

In the feature extraction step, a set of parameters is computed for each remaining region. The centroids of the regions are used in tracking and the other structural features are utilized in classification of the figures. These structural features include:

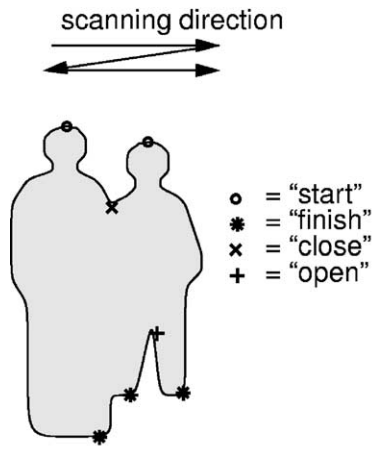
- the ratio between the width and the height of the bounding box (W/H)
- the area of the region (A)
- the relative location of the centroid inside the bounding box (X_0, Y_0)
- the standard deviation of the active pixel coordinates in the region (S_x, S_y)
- the correlation coefficient between horizontal and vertical coordinates of the active pixels (R_{xy})
- the topographic features (QTC)

Topographic features are useful in counting the number of the individual persons by recognizing their heads and upper bodies from the connected regions. This is performed by detecting peaks and valleys of the silhouettes. Fitting a deformable curve over the region would be computationally too complex. Therefore, we decided to use the Quasi-Topological Codes (QTC), which are more familiar features in the area of optical character recognition [9].

The original idea behind the QTCs is to find a sequence of different topological events (or codes) in raster scanned images. The approach adopted in our system uses four distinct codes: 'start', 'finish', 'open', and 'close'. The meaning of these codes is illustrated in Fig. 2a. Two consecutive rows of the binary image are scanned at the same time. For each location, the pair of pixels (upper and lower) is considered as an input pattern to a code generator. Thus, there are four possible patterns that can be detected: (0, 0), (0, 1), (1, 0), and (1, 1). The code generator is actually a state machine that uses the input patterns to trigger the state transitions and to output the right codes. The code generator table used in our system is presented in Fig. 2b. There are six possible states and four input patterns. The table gives the next state and output code for each combination of the input patterns and current states. As we can see, the codes are generated without any arithmetical operations, and therefore, scanning of the entire image is extremely fast.

In Fig. 2c some example objects with the corresponding QTCs are shown. If we concentrate on the upper half of the blobs we notice that the number of the 'start' and 'close' codes correlates with the number of persons in the blob. For each head there should be one start code, and the gap between the persons forms a close code. By recording the numbers of the start and the close codes above the center line, we have a simple means of estimating the number of persons in the region.

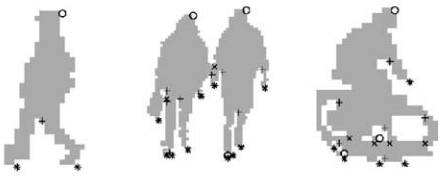
The lower parts of the objects are often so incoherent that they do not provide any extra information for counting, but in classification between pedestrians and cyclists they can be useful. As we see, the tyres of the bicycle in Fig. 2c form extra open and close codes which help us to distinguish the cyclist from the single pedestrian on the left. Therefore, we



(a)

Inputs		State/Code					
upper	lower	0	1	2	3	4	5
0	0	0	0/o	0/✳	0	0	0
0	1	1	1	1/✳	4	4	1
1	0	2	2/o	2	5	2	5
1	1	3	3	3	3	3/x	3/+

(b)



(c)

Fig. 2. Quasi-Topological Codes in feature extraction.

also record the numbers of the open and close codes below the center line to the feature vector for further processing.

In a sequence of images there are typically several observations of each moving object. Some of the observations may be badly misleading due to, for example, fragmentation of the blobs. Thus, estimating the feature value by averaging over all samples is unreliable. A more precise estimate is obtained by calculating the median of the sample values which is more robust against outliers in the data.

2.4. Tracking

Tracking is needed for determining the object correspondences between frames. In our approach, the tracked feature is the object centroid which is a quite steady fixation point regardless of the small fragmentation of the blobs from time to time. Tracking enhances the structural information perceived from the moving objects, which improves the classification results. We can also estimate the apparent speed of the objects which is often a useful feature in classification between pedestrians and cyclists. The feature vector is therefore augmented with the median of the apparent speed of the tracked object.

Tracking is initiated every time a new moving object is determined in the scene and the features found fulfil the predefined criteria. This object is compared with the motion regions appearing in the next few frames within a search window. If the structural features are alike, a Kalman filter based tracker is started. A simple tracking scheme is used where the horizontal x_i and the vertical coordinates y_i of the object motion vector are assumed to be independent of each other, which makes it possible to decompose the motion into two separate single coordinate models:

$$\begin{bmatrix} x_{i+1} \\ \dot{x}_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_i \\ \dot{x}_i \end{bmatrix} + \begin{bmatrix} T^2/2 \\ T \end{bmatrix} v_{x,i} \tag{2}$$

$$\begin{bmatrix} y_{i+1} \\ \dot{y}_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} y_i \\ \dot{y}_i \end{bmatrix} + \begin{bmatrix} T^2/2 \\ T \end{bmatrix} v_{y,i} \tag{3}$$

where T is the time step between frames, and $\dot{x}_i$ and $\dot{y}_i$ are the components of the velocity vector. It is assumed that the target is moving with constant velocity, but is subject to zero-mean white Gaussian acceleration errors $v_{x,i}$ and $v_{y,i}$. Measurements $(X_i,\ Y_i)$ are made of the position of the target:

$$X_i = [\,1 \quad 0\,] \begin{bmatrix} x_i \\ \dot{x}_i \end{bmatrix} + \eta_{x,i} \tag{4}$$

$$Y_i = [\,1 \quad 0\,] \begin{bmatrix} y_i \\ \dot{y}_i \end{bmatrix} + \eta_{y,i} \tag{5}$$

where $\eta_{x,i}$ and $\eta_{y,i}$ are white Gaussian noise sequences representing the errors in the feature extraction process.

The tracking model used leads to a computationally very efficient steady-state Kalman filter formulation. It means that the error covariance matrices as well as the Kalman gain attain constant values after a few frames. By recording these values in advance, and using them in filtering, it becomes possible to restrain the computation required to only state vector updating. First, the state of the tracker is predicted using the dynamic model of Eqs. (2), (3). The new observations are selected based on their geometric distances from the predicted coordinates, and also based on their structural similarities with respect to previous observations. The best match is used as a new measurement of the object location. This new information is then integrated by the Kalman filter to the state variables.

All tracking techniques are very sensitive to the quality of the observations. If the features tracked are unreliably found the tracking may diverge or produce erroneous results. Due to occasional errors in feature extraction, or occlusions caused by foreground objects, the measurements are not always available for each frame. In those cases tracking should not be terminated, because the discontinuity in the path may cause false increment of the people counter. This problem is solved, at least in most cases, by keeping the tracker alive for a while without new observations. In practice, this means that the new estimates of the object locations are obtained by prediction. The normal operation is resumed as soon as there are valid observations available. If the object is not observed during a certain period of time, the trajectory, the feature vector and a 32 by 32 pixel sized binary silhouette of the tracked object is recorded in the memory card for further analysis.

3. Analysis

The data collected by the tracking unit is transferred to a desktop computer for analysis. The data contains raw information about the number and the category of the moving objects. But it also includes the spatio-temporal coordinates of the targets, which can be utilized to reproduce the main directions of the traffic flows. The availability of the spatio-temporal coordinates may also extend the use of the system, for example, to the analysis of the traffic accidents.

In the first stage of the analysis, very short trajectories are eiher discarded or sustained depending on their context. If long trajectories exist within a short distance in the spatio-temporal space, there is strong evidence that the person associated with the trajectory is occasionally deviating from a larger group of people. If these persons are counted separately, it would probably result in an overestimated number of people. On the other hand, if the trajectories are alone, they should be treated similarly to the longer ones.

It was noticed during the development of the monitoring system that a general classifier could not be constructed for an arbitrary installation of the tracking unit. The shapes of the objects depend much on the direction and the distance between the camera and the road. For this reason, the analysis system was designed to be trained before the actual use. Training would only be necessary if the tracking device is installed in a new location where earlier training data is not available.

The classification of the objects is based on the assumption that the feature vectors are clustered so that

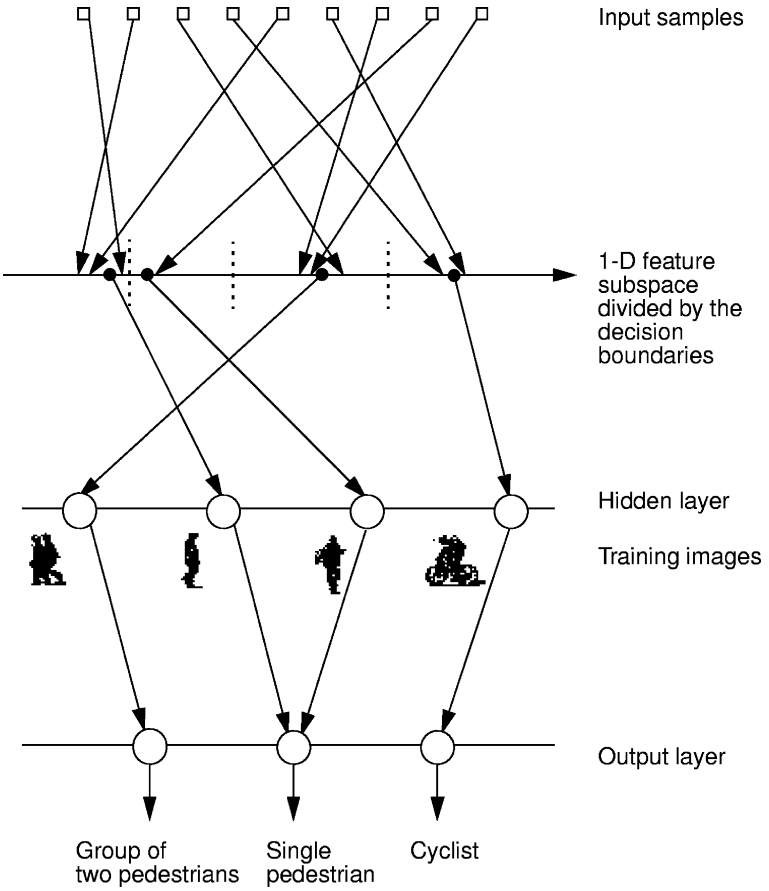


Fig. 3. A simplified example of the LVQ based classifier.

pedestrians and cyclists form separate groups. Some of the tracked objects may consist of several persons, which is taken into account by forming classes also for groups of two and three pedestrians. Furthermore, the data may contain some unwanted observations, which are classified into a 'trash can' category. There can also be several clusters for a single class. For example, at a crossroads the shape of a cyclist can vary quite much depending on the heading. In practise, linear discriminant functions cannot be utilized due to incoherent allocation of the feature space.

Our solution to the classification problem is LVQ [8], which utilizes nonlinear decision boundaries (K-nearest neighbor classification) between the categories. The basic principle of the classifier is shown in Fig. 3 with a simplified example. LVQ is based on a supervised learning principle, where a training set with known class labels is used to first organize a neural network, and then apply the network in a recall mode for classification of the feature patterns.

Topologically, the network contains an input layer, a single hidden layer of neurons and an output layer. The number of the hidden layer neurons should be equal to or greater than the number of the clusters in the feature space. Just in case, extra neurons should be included in the network, because the trash can category typically allocates several neurons.

In the training mode, this supervised network uses a hidden layer such that the distance of the training vector to the weight vector of each neuron is computed and the nearest neuron is declared the winner. If the winning neuron is in the expected class of the training vector, it is reinforced toward the training vector. If the winning neuron is not in the class of the training vector, the connection weights entering the neuron are moved away from the training vector. During this training process, neurons assigned to a particular class migrate to the region associated with their specific class.

In practise, the training procedure is performed by selecting randomly a subset of the objects from the whole input data set and showing the user the small binary images of the objects recorded, along with the features and the trajectories. Based on these images, the user

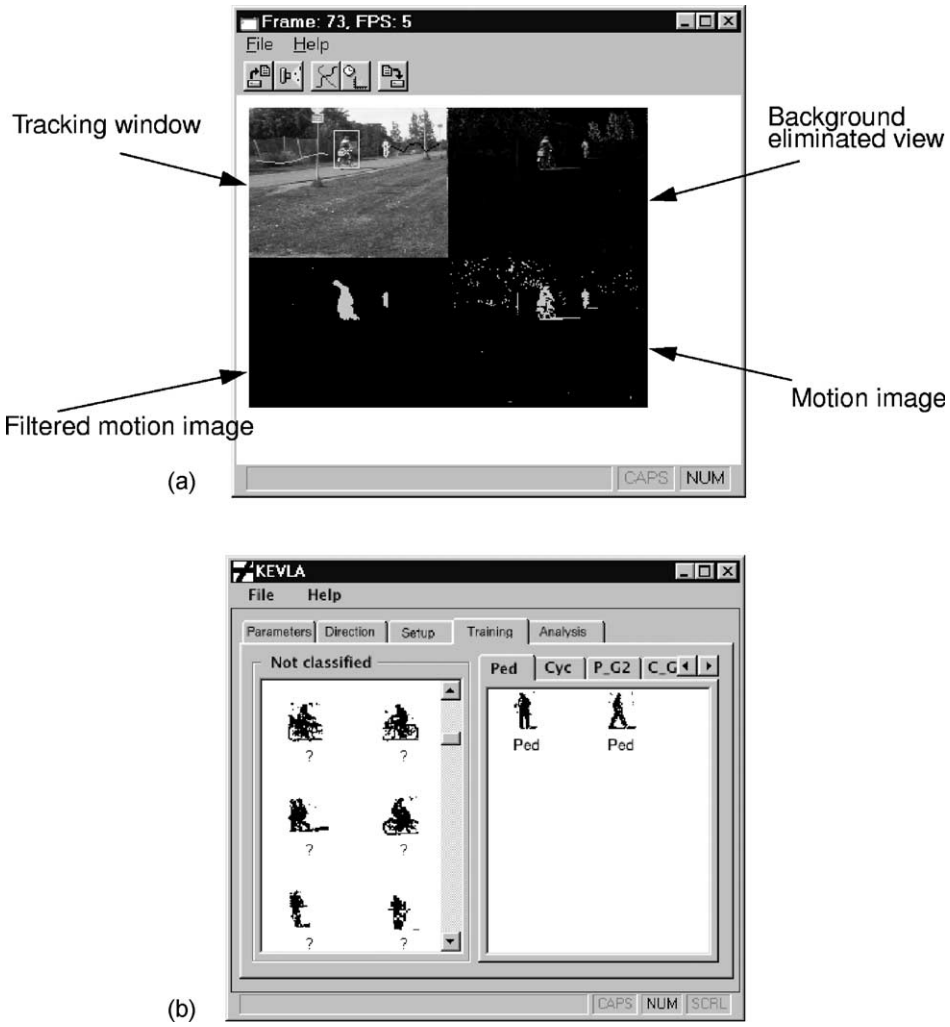


Fig. 4. View of (a) the tracking unit development software, (b) the analysis software.

selects the correct classes for all the samples which are then used in training of the network. Typically, 20–100 samples are needed for training. The reliability of the training result is verified from the relative proportion of the mis-classified samples. After training, the whole data set is processed with the classifier. The number and the category of the objects are finally solved based on the classification result.

4. Experiments

A development version of the tracking system, including a graphical user interface was built on a laptop computer. Fig. 4a shows a snapshot of the user interface as the tracking is going on. The actual long-term tests were performed with an embedded system specially designed for traffic monitoring purposes. In the tests, the tracking device was installed on a pole at 3 m height from the ground and about 10 m from the road. In the current implementation, the total counting time was only 2–3 days, but with advanced power management and more efficient batteries the operation time could be extended to one week.

The analysis part of the monitoring system is separate software running in a PC workstation. A snapshot of the program is shown in Fig. 4b. The user can select either a pre-trained network or execute the training procedure first. The unknown objects on the left are simply picked and moved to the correct category.

A set of tests was made in various weather conditions. Table 1 shows the counting and classification results of five independent experiments, where also simultaneous manual counting was performed as a reference. In all cases, the tracking device was installed beside roads where only pedestrians and cyclists are allowed. The first row of Table 1 shows the duration of the experiment in hours. The second row gives the counting result, and the corresponding reference measurement is in the third row of the table. The proportional error of the counting result is given in the fourth row. The number of cyclists produced by the system, the ground truth, and the corresponding error rate are presented in the last three rows of the table.

Table 1
Counting and classifying results when only pedestrians and cyclists are present

Exp. #	1	2	3	4	5
Duration (h)	3	2	2	1	0.5
Count	97	35	89	51	23
Ref. count	107	23	91	51	23
Error %	−9	52	−2	0	0
Cyclists	45	23	24	7	15
Ref. cycl.	52	11	21	7	16
Error %	−2	18	4	0	−4

Table 2
Counting and classifying results when mixed traffic is present

Exp. #	1	2	3	4	5
Duration (h)	2	2	2	2	2
Count	166	101	131	243	169
Ref. count	148	121	88	241	210
Error %	12	−17	49	1	−20
Cyclists	105	58	69	14	72
Ref. cycl.	112	85	50	10	124
Error %	−12	−13	−4	2	−16

The sample sizes are rather small, but they still give a reasonable estimate of the accuracy which can be achieved with the current implementation. We notice that the counting error remains under 10% in all cases except one (experiment 2) where strong wind caused camera motion resulting as increased rate of false observations. In the same case, also the classification error was slightly worse than expected, but in the other experiments the limit of a 15% misclassification rate was not exceeded.

In the other test set, the tracking device was installed beside roads where also motor vehicles are present. The results of these experiments with manual reference data are given in Table 2. We observe that the error rates both in the count and the class increased and the target level was not achieved. The cars seem to cause mainly increment in the counting results. However, in some cases the counter produces less observations than the reference count. This is mainly due to the difficulties in placing the counter in such a manner that the people on each side of the road are properly observed. There is also strong variation in the counting results, which is primarily a sign of the different rates of the car traffic in the scenes. Nevertheless, the errors are still at a tolerable level for most tasks where the system was intended to be used.

5. Discussion

In this paper, we presented a new approach to count and classify cyclists and pedestrians automatically. The test results are promising, but there is still room for improvement. Especially, object recognition and classification could be enhanced by using more descriptive shape features. The QTC codes are fast to extract, but they are very sensitive to noise. Using correlation masks or higher order statistics could provide a better solution, but they also increase the amount of computation needed.

Another problem is the shadows that occasionally remain connected to the silhouette after motion detection. They can cause the tracker to lose the target resulting in disjointed trajectories. They also cause the feature vectors to be

scattered within the same category, which makes the classification task more difficult.

Dividing the system into real-time tracking unit and off-line analysis seems to be a good solution. We believe that in the future this architecture will be a basis for similar monitoring systems.

Acknowledgements

We would like to thank Arto Pesonen for the system development and implementation of the software. Elektrobit Ltd. and the Finnish National Road Administration are gratefully acknowledged for their financial support and also their efforts in the development and testing. This work was partially funded by the Academy of Finland and the Technology Development Centre (TEKES). Their support is acknowledged with thanks.

References

[1] A. Baumberg, D. Hogg, An efficient method for contour tracking using active shape models, in IEEE Workshop on motion of Non-Rigid and Articulated Objects, Austin, 1994, pp. 194–199.

[2] M. Bogaert, N. Chleq, P. Cornez, C.S. Regazzoni, A. Teschioni, M. Thonnat, The PASSWORDS Project, In International Conference on Image Processing (ICIP'96), Lausanne, Switzerland 3 (1996) 675–678.

[3] C. Bregler, Learning and recognizing human dynamics in video sequences, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR'97), San Juan, Puerto Rico, 1997, pp. 568–574.

[4] D. Comaniciu, V. Ramesh, P. Meer, Real-time tracking of non-rigid objects using mean shift, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR'00), Hilton Head Island, SC, pp. 142–149.

[5] D.M. Gavrila, V. Philomin, Real-time object detection for smart vehicles, in International Conference on Computer Vision (ICCV'99), Corfu, Greece, 1999, pp. 87–93.

[6] I. Haritaoglu, D. Harwood, L. Davis, W4: Who, When, Where, What: A Real Time System for Detecting and Tracking People, in Third Face and Gesture Recognition Conference, 1998, pp. 222–227.

[7] M. Isard, A. Blake, Contour Tracking by Stochastic Propagation of Conditional Density, in European Conference on Computer Vision (ECCV'96), Cambridge, UK, 1996, pp. 343–356.

[8] T. Kohonen, Self-organizing Maps, Springer, Berlin, 1995.

[9] M. Nadler, E.P. Smith, Pattern Recognition Engineering, Wiley, New York, 1993.

[10] C. Papageorgiou, T. Poggio, A. pattern, A pattern classification approach to dynamic object detection, in Int. Conference on Computer Vision (ICCV'99), Corfu, Greece, 1999, pp. 1223–1228.

[11] K. Rohr, Incremental recognition of pedestrians from image sequences, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR'93), New York, 1993, pp. 8–13.

[12] C. Wren, A. Azarbayejani, T. Darrell, A. Pentland, Pfinder: Real-time tracking of the human body, IEEE Trans. Pattern Anal. Mach. Intell. 19 (7) (1997) 780–785.