

# VEHICLE CLASSIFICATION USING EVOLUTIONARY FORESTS

Murray Evans, Jonathan N. Boyle and James Ferryman

*Computational Vision Group, School of Systems Engineering, University of Reading, Reading RG6 6AY, U.K.*

Keywords: Vehicle classification, Evolutionary forests.

Abstract: Forests of decision trees are a popular tool for classification applications. This paper presents an approach to evolving the forest classifier, reducing the time spent designing the optimal tree depth and forest size. This is applied to the task of vehicle classification for purposes of verification against databases at security checkpoints, or accumulation of road usage statistics. The evolutionary approach to building the forest classifier is shown to out-perform a more typically grown forest and a baseline neural-network classifier for the vehicle classification task.

## 1 INTRODUCTION

Vehicle classification is a potentially very useful tool. In the context of border security checkpoints, automatic classification of vehicles could be used in conjunction with automatic number-plate recognition to provide verification that the type of car registered against the plate number matches the car observed at the checkpoint. Mismatches can indicate stolen or unlicensed vehicles which should be more thoroughly checked. Alternatively, a camera observing a road or gate could be used to determine statistics of vehicle types that pass through, providing information that can be of use in future road planning activities.

At a border checkpoint, passenger vehicles (busses, cars) and light goods vehicles (vans) are often split from heavy goods vehicles. This paper presents the results of research developing a classifier capable of distinguishing between different sub-classes of passenger vehicles: hatchbacks, saloons, estate cars, sports-utility vehicles, sports-cars etc. At this level, the information provided can be useful for verification against the registered details of a vehicle. For instance, vehicle registration information will often store details of the make and model of a car against its registration details. In the UK, a Ford Focus can be either an estate or a hatchback, so an SUV travelling with registration plates befitting a Ford Focus could indicate foul play, and should arouse suspicion. Although previously published works have considered multiple vehicle classes, few have considered the task of vehicle classification beyond the car/bus/truck level.

To achieve the classification, it was decided to explore the potential of forest classifiers. Forests are ensembles of decision trees, and have seen much interest in the vision community after the publication of (Lepetit and Fua, 2006). Of particular interest have been the so called randomised forests. Forest classifiers are appealing in part because of their simplicity, but equally from their fast run-time performance, inherent ability to deal with multiple classes, and the simplicity with which wildly different primitive classifiers can form the building blocks of the forest.

This paper proposes constructing the forest classifier using an evolutionary approach, resulting in a classifier that is generally smaller and of superior performance to a forest grown in the original manner.

The remainder of this paper is structured as follows: Section 2 will discuss previous works on vehicle classification. Section 3 will then detail the data set created for the experiments. The developed classifier will be introduced in Section 4 and then evaluated in Section 5, before Section 6 concludes and considers future work.

## 2 RELATED WORK

Previous research into the area of vehicle classification has shown some success. Classification is performed in (Gupte et al., 2002) to split vehicles in a highway scene into Truck and non-truck classes by estimating the size of the vehicle using calibration information, an approach similarly taken by (Shi et al., 2007). The calibration requirement is avoided in (Av-

ery et al., 2004) by determining the length of motion blobs and determining the statistics for the current view to split into long and short vehicles. The calibration step is also avoided by (Hsieh et al., 2006), which automatically determines the lanes of the road and normalises the size of the detected vehicles by the width of the lane at the part of the image. Vehicles are classified based on their size and a feature termed “linearity” into cars, mini-vans, trucks and “van-trucks”. (Huang and Liao, 2004) also make use of vehicle size from a side profile view of a highway, but also incorporate further measurements of the aspect ratio and compact ratio of the motion silhouettes to get a finer grained classification considering seven classes.

Earlier work using a model fitting approach can be seen in (Sullivan et al., 1996) which showed promising results classifying cars versus vans, again in a highway scenario. A related approach was later taken by (Buch et al., 2008) using the possible classifications: Bus/Lorry, Van, Car/Taxi, Motor-bike/bicycle. The advantage of such model based approaches should be a reduction in view-point dependence, though building the models could be expensive and prone to error.

Other approaches include the use of Gabor filters and a minimum-distance classifier (Ji et al., 2007), using normalised side-profile images of the vehicles. (Zhang et al., 2006) perform the classification using “Eigen-vehicles” (a reference to Eigenfaces), along with PCA and a support vector machine. (Zhang et al., 2008) use a transformation-ring-projection with wavelet fractal signatures to describe side-profile vehicle segmentations for classification. Classification and tracking are considered together in (Morris and Trivedi, 2006), and (Negri et al., 2006) use frontal views to go as far as classifying individual models of cars.

### 3 DATA PREPARATION

In common UK parlance, cars can be grouped into one of the following types: hatchbacks, saloons, estates, off-roaders or Sports Utility Vehicles (SUVs), sports-cars, convertibles and people-carriers. Add to this, there are vans, busses, trucks, pickups and lorries. To collect a suitable data-set, a camera was erected at one of the main entrance/exit gates at the University of Reading, looking perpendicular across the entrance road. This recorded the morning rush-hour, accumulating a total of approximately four hours of footage. The orientation of the camera was chosen to capture as close to a side-profile of the pass-

ing cars as was possible, as the side-profile was considered to be the most discriminating between the different classes of vehicles.

A motion detector (Zivkovic, 2004) was used to detect frames in the video where the quantity of motion in a pre-defined section towards the centre of the image passed a threshold, indicating the presence of a vehicle. A wheel detector was used to determine the location of the wheels of the vehicle, and then scaling and cropping was applied to both the RGB and motion-mask images, such that the wheels occupied a pre-defined location in the resulting normalised images. The normalised images were sized at  $200 \times 85$  pixels, with the front and back wheels centred at (50,80) and (150,80) respectively. The resulting images were then manually parsed to determine their correct classification, and to verify robust normalisation.

Examples of the normalised images for several of the observed vehicle classes are shown in Figure 1, along with example motions masks in Figure 2. The resulting image database contained example images in the classes and quantities listed in Table 1. It is quite clear that a strong bias exists for vehicles fitting into the hatchback category.



Figure 1: Examples of the normalised vehicle images. From top to bottom the images depict samples from the classes: hatchback, estate, saloon, people-carrier, SUV and van.

### 4 FOREST CLASSIFIER

To classify the vehicle images, an “evolutionary forest” classifier has been developed. This fuses the traditional forest classifier (an ensemble of decision trees) (Breiman, 2001), with techniques from genetic algorithms to evolve a forest optimised for the classification task.

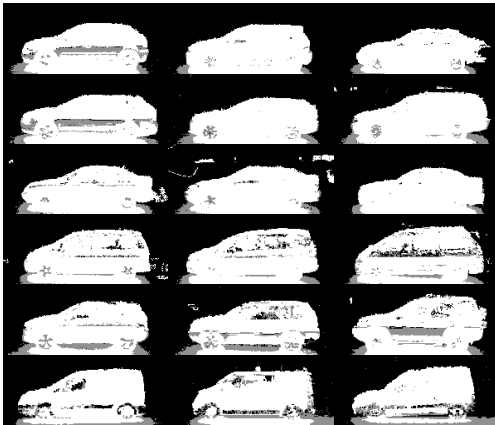


Figure 2: Examples of the normalised mask images accompanying the normalised RGB images depicted in Figure 1.

Table 1: Classes and number of example images acquired, as well as example training/testing splits.

class	id	total	training	testing
hatchback	1	7027	351	6676
estate	2	1163	58	1105
saloon	3	1640	82	1558
convertible	4	342	50	292
sports-car	5	173	50	123
people-carrier	6	680	50	630
SUV	7	543	50	493
van	8	1246	62	1184

Recently, randomised decision forests have been of great interest. These grow a set of decision trees where each decision node of the tree is configured to exploit a randomly chosen classifier from a set of available classifiers, often with randomly initialised parameters that are optimised to produce the best split of the training examples. Typically, only binary decision trees are used, where each node has two children. These randomised decision forests can be effective in many scenarios, perhaps most famously in the Microsoft Kinect player tracking system (Shotton et al., 2011), however it can be unclear as to how large the forest should be, and how deep each tree should be permitted to become. Furthermore, while each tree is optimised to be the best classifier it can be, the trees are not optimised to work together to produce the best forest that can be produced.

This work considers an evolutionary approach to growing the forest ensuring that every tree that is a part of the forest is there to optimise the forest's overall performance.

## 4.1 Decision Tree Nodes

Each decision tree in the forest is a binary tree where each node is a primitive binary classifier. Given an input, each node can either send the input to its left or right child. For the purposes of the vehicle classification task four very simple low-level image based classifiers have been implemented.

### 4.1.1 Pixel Difference Node

This first node has five parameters, namely the  $(x,y)$  image coordinates of two pixels in the image, and a threshold value  $t_c$ . The colour  $\mathbf{p}_1$  and  $\mathbf{p}_2$  of each pixel is extracted and represented in CIELAB space, then the Euclidean difference is computed between the two colours, and compared to a threshold. If the result is larger than the threshold, the second child is activated, otherwise, the first child is activated. If  $c$  represents the child, then the node's function can be represented by the equation:

$$c = \begin{cases} 1 & \text{if } t_c < \sqrt{\mathbf{p}_1 \cdot \mathbf{p}_2} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

### 4.1.2 Edge Orientation Node

The input image  $I$  can be transformed to the two channel image  $E$  representing the magnitude and orientation of gradients in the image. A decision node can be created that checks if, for a given pixel  $(x,y)$ , the magnitude is above a threshold  $t_m$  and the orientation is between the values  $t_{om} \rightarrow t_{oM}$ . This node therefore has five parameters (the pixel co-ordinates and three thresholds). Let  $m$  and  $o$  be the gradient magnitude and orientation at  $(x,y)$ , then the appropriate child is selected from:

$$c = \begin{cases} 1 & \text{if } (m > t_m) \wedge (t_{om} < o < t_{oM}) \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

### 4.1.3 Chamfer Node

The edge magnitude image can be thresholded to produce a binary edge image. This in turn can be used to produce a distance transform image, or chamfer image  $C$  where every pixel is valued by its distance from the nearest edgel. A very simple node can then be created that, given two pixels as input, selects the appropriate child based on whether the first or second pixel has a larger value in  $C$ .

$$c = \begin{cases} 1 & \text{if } C(x_0, y_0) > C(x_1, y_1) \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

#### 4.1.4 Mask Node

The final node takes in the binary motion mask  $M$  that was created at the same time as the RGB image  $I$ . For a given pixel, this node selects a child based only on whether the pixel is active in the mask image, or not.

$$c = \begin{cases} 1 & \text{if } M(x,y) > 0 \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

## 4.2 The Evolutionary Forest

To create a randomised forest, each tree is grown one by one. First, a classifier is selected at random for the root node, and the parameters which create the optimal split of the input data determined (see section 4.2.2). The split data are then sent to the child nodes, and so on, until a maximum tree depth is achieved, or a split of the data contains only one class type.

When an unseen testing example is presented to the forest, it will pass through each tree and reach a leaf node, providing information on what the likely class of the unknown image is. For instance, if the leaf node recorded 200 instances of a hatchback, and only one instance of SUV, the probability would be that the unknown image also represents a hatchback. The vector of class counts for all leaf nodes can be accumulated to determine a belief as to the most appropriate class for the input image.

In this approach, each tree is an independently optimised classifier, and the forest result a majority belief. There exists some prior work in using evolutionary approaches to create decision trees (Papagelis and Kalles, 2001), however in this work the aim is not to produce a population of independently optimal decision trees, but rather to evolve a set of trees that work together in an optimal forest.

### 4.2.1 Constructing the Evolutionary Forest

To begin creating the forest, the first decision tree must be created. An initial set of short decision trees is grown. This will form the population from which the first tree of the final forest classifier is evolved. Each of the initial trees is grown using a subset of the whole set of training images  $T$ . The trees are then evaluated against their ability to classify the whole of  $T$ . The worst  $w\%$  of trees are then exposed to replacement by one of the following genetic operators:

1. **Cross-over:** The poor performing tree  $t_d$  is removed from the set of trees, and replaced by a new tree that is the product of a cross-over “breeding” of two trees with better performance. This

involves selecting a random node on each of the better trees, and swapping them, children and all. This results in two trees, the shorter of which is selected as the replacement tree.

2. **Regeneration:** The poor performing tree  $t_d$  is removed from the set of trees and replaced by a completely new tree.
3. **Mutation:** A node is selected in the poorly performing tree  $t_d$ . The node is replaced by a different node with randomly selected parameters.

After a number of iterations (this can be a fixed number, or a number based on the current best performance of the population), the genetic process is stopped. The best performing tree is taken from the set of trees, and becomes the first tree of the final forest classifier.

Now, the process repeats, adding trees one at a time to the final forest. However, after the first optimisation, the definition of the “best” tree is slightly altered. No longer is it desirable to produce an optimal, stand-alone decision tree. Rather, what is sought is the tree that best augments the current final forest. The evaluation criteria and the algorithm to optimise the individual trees will now be discussed in the remaining parts of this section. The training algorithm is summarised in Figure 3.

```

Let  $T$  be the set of training images
Create an empty forest  $F'$ 
while no. of trees < maximum:
    Create  $V$ , a random subset of  $T$ 
    Create a set of decision trees,  $F_i$ 
    for  $c = 0$  to max no. iterations:
        Train the trees in  $F_i$ 
        Evaluate the trees in  $F_i$ 
        Replace the worst  $w\%$  of trees in  $F_i$ 
        Add the best tree in  $F_i$  to  $F'$ 
Perform a final training of  $F'$ 

```

Figure 3: Pseudocode of evolutionary forest algorithm.

### 4.2.2 Optimising Tree Nodes

When growing the initial trees, it is useful to optimise the parameters of a node to produce the “best” possible split of the data. This is achieved by using a greedy search of the parameters to maximise the information gain between the input data to the node, and the two output sub-sets, identical to the approach used in (Shotton et al., 2011).

### 4.2.3 Evaluating Tree Performance

On the initial iteration that produces the first tree for the final forest, the evaluation function for the trees is set such that the evolutionary process is attempting to produce the best possible stand-alone decision tree. This performance is captured by the following equation:

$$\phi(f_i) = 0.3\bar{z} + 0.7 \min_j Z(i, j) \quad (5)$$

Here,  $Z(i, j)$  is the number of images that the tree  $f_i$  correctly classified of class  $h_j$ , normalised by the number of examples of class  $h_j$  that were shown to the tree. Meanwhile,  $\bar{z}$  is used to indicate the median of  $Z(i, j)$  over the possible classes  $h_j$ . This equation is structured to favour a tree that performs reasonably well across all of the possible classes, rather than a tree that excels at any one class but is wholly wrong on some others, and the parameters 0.3 and 0.7 balance the equation to encourage a jack-of-all trades classifier which has generally provided the best starting point for the evolutionary process.

Once the first tree has been selected, the optimisation function used to design subsequent trees alters. Rather than trying to make a tree that is an optimal classifier of the data, the aim is to produce a tree that best augments the existing forest. This is a more subtle equation to produce.

Let  $F'$  be the final forest that the algorithm is trying to produce, and  $F_i$  be the population of trees that are being optimised to produce the  $i$ 'th tree for  $F'$  (meaning that  $F'$  contains  $i - 1$  trees).

For any given image  $I$  with known class  $h$ ,  $F'$  can be used to produce a classification  $h'$ . In doing so, it will also produce the belief vector  $\mathbf{h}'$ , which is the accumulated beliefs of all the  $i - 1$  leaf nodes reached during the classification process. In an ideal classifier, all but the belief in the correct class will be 0. More realistically, the desire will be that the belief in the correct class is larger than the belief in each of the other classes. The larger this difference, the more confident the classification.

As such, for a given image, it is possible to calculate  $\mathbf{h}'$  as the belief of the forest  $F'$ , as well as  $\mathbf{h}_i$  as the belief of the forest  $F'_i$ , which is the forest  $F'$  augmented by a tree in  $F_i$ . The effectiveness of the tree to improve or complement the performance of  $F'$  can then be written as:

$$\sigma(f_n \in F_i) = \sum_I ((c_i - m_i) - (c' - m')) \quad (6)$$

where  $c_i$  and  $c'$  are the values of  $\mathbf{h}_i$  and  $\mathbf{h}'$  for the correct classification of the image  $I$ , while  $m_i$  and  $m'$  is

the largest values in  $\mathbf{h}_i$  and  $\mathbf{h}'$ . If the forest is correct, the term  $c - m$  will be positive, and the more confident the classification, the larger the positive value. If the augmented forest performs better than  $\sigma(f_n)$  will be positive, and the better it performs, the more positive it becomes. In this way,  $F_i$  is breeding trees designed to improve the performance of the final forest  $F'$ , rather than trees that operate as expert independent classifiers.

## 5 RESULTS

To evaluate the performance of the classifier, the image dataset was randomly split such that for each class  $h_j$  with  $n_j$  training examples there were at least  $\max(50, 0.05 \times n_j)$  images removed from the training set to the testing set. Table 1 shows an example split of the data, and also gives each class an identifying number corresponding to the numbers in the confusion matrices in Table 2.

The classification performance of the genetically optimised classifier is compared to a traditionally grown randomised forest, as well as “baseline” results from a multi-layer perceptron neural network. The classification results of each classifier are displayed as confusion matrices. Given the set of images for a specific class (e.g. hatchback, row 1), the columns specify the percentage of images classified, or misclassified, as each of the possible classifications. A perfect classifier will produce 100% along the diagonal, and 0% everywhere else. The confusion matrices in Table 2 show average values over twenty different training/testing splits.

The genetic forests typically reach maximum performance against the testing set by about 30 to 45 trees, as such, all were grown to a size of 50 trees. Individual trees in the forests reached a maximum depth of 16 levels, however, the median depth for any one tree is only 4.

The randomised forests were each grown to a maximum depth of 11 nodes, with again 50 trees per forest, a size empirically suggested as near-optimal for the classification task in hand. Note that this is a far larger forest (due to the increased depth) than the evolved forest.

For comparison with other traditional classification approaches, a multi-layer perceptron was also trained on the data, using gradient images as the input. The results of this approach are the weakest of the tested methods, showing that the forest, and evolutionary forest, are an excellent choice for this classification task.

Table 2: Classification results for (from top to bottom), the evolutionary forest, randomised forest, and neural network.

	0	1	2	3	4	5	6	7
1	97.2	0.4	0.6	0.0	0.5	0.8	0.1	0.4
2	5.1	81.6	7.2	0.6	0.4	4.1	0.6	0.4
3	0.5	2.7	90.1	2.5	4.1	0.0	0.0	0.1
4	8.7	1.1	10.7	76.2	3.2	0.0	0.2	0.0
5	2.7	0.0	6.6	1.3	89.3	0.0	0.0	0.0
6	13.5	3.7	0.0	0.0	0.0	75.7	6.6	0.5
7	4.2	2.5	0.1	0.6	0.0	5.5	85.3	1.8
8	2.5	0.7	0.0	0.0	0.0	2.6	1.9	92.4
	0	1	2	3	4	5	6	7
1	95.6	0.4	0.2	0.1	0.8	2.8	0.0	0.1
2	6.5	63.8	12.3	1.6	0.8	13.2	1.8	0.1
3	0.9	3.1	86.9	5.0	3.8	0.1	0.1	0.0
4	15.8	0.6	18.1	63.2	2.2	0.1	0.1	0.0
5	6.7	0.0	15.0	2.9	75.5	0.0	0.0	0.0
6	18.3	0.9	0.0	0.0	0.0	71.0	9.7	0.0
7	6.9	1.4	0.1	0.8	0.0	18.5	71.9	0.4
8	5.0	0.6	0.0	0.0	0.0	9.0	4.6	80.6
	0	1	2	3	4	5	6	7
1	92.9	1.6	2.1	0.2	2.3	0.4	0.4	0.1
2	18.3	58.5	6.4	11.1	1.0	3.8	0.3	0.8
3	10.3	2.2	77.0	6.9	1.2	0.3	1.8	0.4
4	11.4	3.1	20.6	56.7	2.7	0.5	2.5	2.4
5	16.1	0.2	4.2	0.1	72.7	1.2	4.6	0.9
6	10.0	13.4	0.7	1.2	9.8	54.9	3.0	7.0
7	3.5	1.0	5.3	2.5	4.4	2.5	69.9	11.0
8	0.8	1.7	0.9	3.9	2.0	5.3	6.5	79.0

## 6 CONCLUSIONS AND FUTURE WORK

This paper has considered the application of forest classifiers to the task of vehicle classification, proposing in the process a method of growing the forest by use of an evolutionary approach. Compared to the typical randomised forest, the genetic forest showed superior performance, and also performed better than a baseline neural network.

Future work will aim to extend the current implementation to classify vehicles into make and model categories, alternative image features to exploit at each low-level classifier node, as well as determining the efficacy of the evolutionary forest approach in other contexts.

## ACKNOWLEDGEMENTS

This work was partially funded by the EU FP7 project EFFISEC with grant No. 217991.<sup>1</sup>

<sup>1</sup>However, this paper does not necessarily represent the opinion of the European Community, and the European

## REFERENCES

- Avery, R., Wang, Y., and Scott Rutherford, G. (2004). Length-based vehicle classification using images from uncalibrated video cameras. In *Intelligent Transportation Systems, 2004. Proceedings. The 7th International IEEE Conference on*, pages 737 – 742.
- Breiman, L. (2001). Random forests. *Machine Learning*, 45:5–32. 10.1023/A:1010933404324.
- Buch, N., Orwell, J., and Velastin, S. (2008). Detection and classification of vehicles for urban traffic scenes. In *Visual Information Engineering, 2008. VIE 2008. 5th International Conference on*, pages 182 –187.
- Gupte, S., Masoud, O., Martin, R., and Papanikolopoulos, N. (2002). Detection and classification of vehicles. *Intelligent Transportation Systems, IEEE Transactions on*, 3(1):37 –47.
- Hsieh, J.-W., Yu, S.-H., Chen, Y.-S., and Hu, W.-F. (2006). Automatic traffic surveillance system for vehicle tracking and classification. *Intelligent Transportation Systems, IEEE Transactions on*, pages 175 – 187.
- Huang, C.-L. and Liao, W.-C. (2004). A vision-based vehicle identification system. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 4, pages 364 – 367 Vol.4.
- Ji, P., Jin, L., and Li, X. (2007). Vision-based vehicle type classification using partial gabor filter bank. In *Automation and Logistics, 2007 IEEE International Conference on*, pages 1037 –1040.
- Lepetit, V. and Fua, P. (2006). Keypoint recognition using randomized trees. *Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1465–1479.
- Morris, B. and Trivedi, M. (2006). Improved vehicle classification in long traffic video by cooperating tracker and classifier modules. In *IEEE International Conference on Advanced Video and Signal based Surveillance*.
- Negri, P., Clady, X., Milgram, M., and Poulénard, R. (2006). An oriented-contour point based voting algorithm for vehicle type classification. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, pages 574 –577.
- Papagelis, A. and Kalles, D. (2001). Breeding decision trees using genetic algorithms. In *International Conference on Machine Learning, Proceedings of*.
- Shi, S., Qin, Z., and Xu, J. (2007). Robust algorithm of vehicle classification. In *Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2007. SNPD 2007. Eighth ACIS International Conference on*, pages 269 –272.
- Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., and Blake, A. (2011). Real-time human pose recognition in parts from a single depth image. In *Computer Vision and Pattern Recognition, IEEE Conference on*.

Community is not responsible for any use which may be made of its contents.

- Sullivan, G. D., Baker, K. D., Worrall, A. D., Attwood, C. I., and Remagnino, P. R. (1996). Model-based vehicle detection and classification using orthographic approximations. In *7th British Machine Vision Conference*.
- Zhang, C., Chen, X., and bang Chen, W. (2006). A pca-based vehicle classification framework. In *Data Engineering Workshops, 2006. Proceedings. 22nd International Conference on*, page 17.
- Zhang, D., Qu, S., and Liu, Z. (2008). Robust classification of vehicle based on fusion of tsrp and wavelet fractal signature. In *Networking, Sensing and Control, 2008. ICNSC 2008. IEEE International Conference on*, pages 1788–1793.
- Zivkovic, Z. (2004). Improved adaptive gaussian mixture model for background subtraction. In *Pattern Recognition, 2004. 17th International Conference on*, pages 28–31.



SciTeP Press  
Science and Technology Publications