# Computer Vision and Machine Learning for Enhancing Pedestrian Safety

Tarak Gandhi and Mohan Manubhai Trivedi

Laboratory for Safe and Intelligent Vehicles (LISA), University of California San Diego, La Jolla, CA 92093, USA, tgandhi@ucsd.edu, mtrivedi@ucsd.edu

**Summary.** Accidents involving pedestrians is one of the leading causes of death and injury around the world. Intelligent driver support systems hold a promise to minimize accidents and save many lives. Such a system would detect the pedestrian, predict the possibility of collision, and then warn the driver or engage automatic braking or other safety devices. This chapter describes the framework and issues involved in developing a pedestrian protection system. It is emphasized that the knowledge of the state of the environment, vehicle, and driver are important for enhancing safety. Classification, clustering, and machine learning techniques for effectively detecting pedestrians are discussed, including the application of algorithms such as SVM, Neural Networks, and AdaBoost for the purpose of distinguishing pedestrians from background. Pedestrians unlike vehicles are capable of sharp turns and speed changes, therefore their future paths are difficult to predict. In order to estimate the possibility of collision, a probabilistic framework for pedestrian path prediction is described along with related research. It is noted that sensors in vehicle are not always sufficient to detect all the pedestrians and other obstacles. Interaction with infrastructure based systems as well as systems from other vehicles can provide a wide area situational awareness of the scene. Furthermore, in infrastructure based systems, clustering and learning techniques can be applied to identify typical vehicle and pedestrian paths and to detect anomalies and potentially dangerous situations. In order to effectively integrate information from infrastructure and vehicle sources, the importance of developing and standardizing vehicle-vehicle and vehicle-infrastructure communication systems is also emphasized.

## 1 Introduction

Intelligent Transportation Systems (ITS) show promise of making road travel safer and comfortable. Automobile companies have recently taken considerable interest in developing Intelligent Driver Support Systems (IDSS) for high-end vehicles. These include active cruise control, lane departure warning, blind spot monitoring, and pedestrian detection systems based on sensors such as visible light and thermal infrared cameras, RADARs, or LASER scanners. However, for an effective driver support system, it is desirable to take a holistic approach, using all available data from the environment, vehicle dynamics, and the driver that can be obtained using various sensors incorporated in vehicle and infrastructure [35]. Infrastructure based sensors can complement the vehicle sensors by filling gaps and providing more complete information about the surroundings. Looking in the vehicle at driver's state is as important as looking out in surroundings in order to convey warnings to the driver in the most effective and least distracting manner. Furthermore, due to the highly competitive nature of automobile manufacturing, it is necessary to make such systems cost effective. This makes multi-functional sensors that are used by several of these systems highly desirable.

Accidents involving pedestrians and other vulnerable road users such as bicyclists are one of the leading causes of death and injury around the world. In order to reduce these accidents, pedestrian protection systems need to detect pedestrians, track them over time, and predict the possibility of collision based on the paths that the pedestrian and the vehicle are likely to take. The system should relay the information to the driver in efficient and non-distracting manner or to the control system of the vehicle in order to take preventive actions. Considerable efforts have been made on enhancing pedestrian safety by programs in United states [3, 4], Europe [2, 5] and Japan [1]. Conferences such as Intelligent Vehicles Symposium [6]

T. Gandhi and M.M. Trivedi: *Computer Vision and Machine Learning for Enhancing Pedestrian Safety*, Studies in Computational Intelligence (SCI) **132**, 59–77 (2008)
www.springerlink.com
© Springer-Verlag Berlin Heidelberg 2008

and Intelligent Transportation Systems Conference [7] have a number of publications related to pedestrian detection every year. The recent survey on pedestrian protection [19] has covered the current research on pedestrian detection, tracking, and collision prediction. It is observed that detecting pedestrians in cluttered scenes from a moving vehicle is a challenging problem that involves a number of computational intelligence techniques spanning image processing, computer vision, pattern recognition, and machine learning. This paper focuses on specific computational intelligence techniques used in stages of sensor based pedestrian protection system for detecting and classifying pedestrians, and predicting their trajectories to assess the possibility of collision.

## 2 Framework for Pedestrian Protection System

Figure 1 shows the components of a general pedestrian protection system. The data from one or more types of sensors can be processed using computer vision algorithms to detect pedestrians and determine their trajectories. The trajectories can then be sent to collision prediction module that would predict the probability of collision between the host vehicle and pedestrians. In the case of high probability of collision, the driver is given appropriate warning that enables corrective action. If the collision is imminent, the automatic safety systems could also be triggered to decelerate the vehicle and reduce the impact of collision. In the following sections we illustrate these components using examples focusing on approaches used for these tasks.

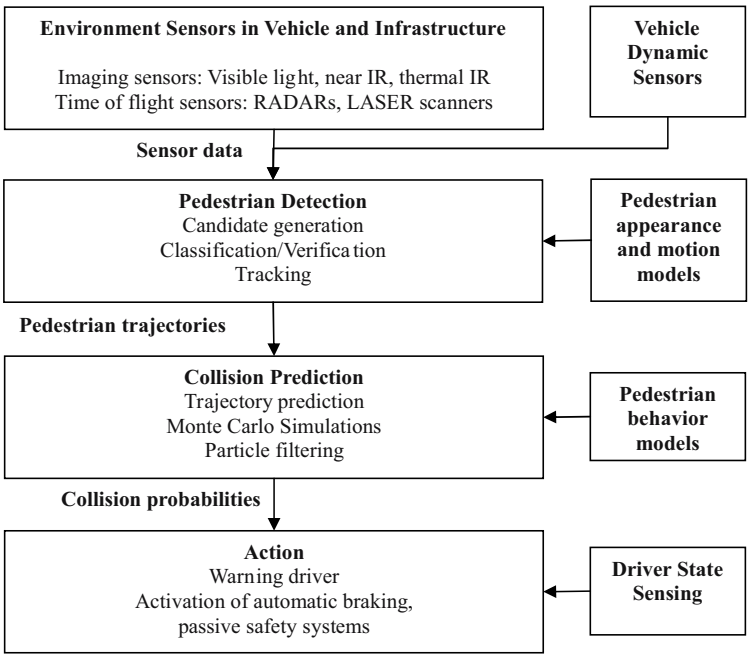



**Fig. 1.** Data flow diagram for pedestrian protection systems

## 3 Techniques in Pedestrian Detection

Pedestrian detection is usually divided into two stages. The candidate generation stage processes raw data using simple cues and fast algorithms to identify potential pedestrian candidates. The classification and verification stage then applies more complex algorithms to the candidates from the attention focusing stage in order to separate genuine pedestrians from false alarms. However, the line between these stages is often blurred and some approaches combine the stages into one. Table 1 shows the approaches used by researchers for stages in pedestrian detection.

### 3.1 Candidate Generation

Cues such as shape, appearance, motion, and distance can be used to generate potential pedestrian candidates. Here, we describe selected techniques used for generating pedestrian candidates using these cues.

#### Chamfer Matching

Chamfer matching is a generic technique to recognize objects based on their shape and appearance using hierarchical matching with a set of object templates from training images. The original image is converted to a binary image using an edge detector. A distance transform is then applied to the edge image. Every pixel $r = (x, y)$ in the distance transformed image has a value $d_I(t)$ equal to the distance to the nearest edge pixel:

$$d_I(r) = \min_{r' \in edges(I)} \|r' - r\| \tag{1}$$

The distance transformed image is matched to the binary templates generated from examples. For this purpose, the template is slided over the image and at every displacement, the chamfer distance between the image and template is obtained by taking the mean of all the pixels in the distance transform image that have an 'on' pixel in template image:

$$D(T, I) = \frac{1}{|T|} \sum_{r \in T} d_I(r) \tag{2}$$

Positions in the image where the chamfer distance is less than a threshold are considered as successful matches.

**Table 1.** Approaches used in stages of pedestrian protection

| Publication | Candidate generation | Feature extraction | Classification |
|---|---|---|---|
| Gavrila ECCV00 [20], IJCV07 [21], Munder PAMI06 [27] | Chamfer matching | Image ROI pixels | LRF Neural Network |
| Gandhi MM04 [15], MVA05 [16] | Omni camera based planar motion estimation | | |
| Gandhi ICIP05 [17], ITS06 [18] | Stereo based U disparity analysis | | |
| Krotosky IV06 [25] | Stereo based U and V disparity analysis | Histogram of oriented gradients | Support Vector Machine |
| Papageorgiou IJCV00 [28] | | Haar wavelets | Support Vector Machine |
| Dalal CVPR05 [13] | | Histogram of oriented gradients | Support Vector Machine |
| Viola IJCV05 [37] | | Haar-like features in spatial and temporal domain | AdaBoost |
| Park ISI07 [29] | Background subtraction, homography projection | Shape and size of object | |

In order to account for the variations between individual objects, the image is matched with number of templates. For efficient matching, a template hierarchy is generated using bottom up clustering. All the training templates are grouped into $K$ clusters, each represented by a prototype template $p_k$ and the set of templates $S_k$ in the cluster. Clustering is performed using using an iterative optimization algorithm that minimizes an objective function:

$$E = \sum_{k=1}^{K} \max_{t_i \in S_k} D_{min}(t_i, p_k) \tag{3}$$

where $D_{min}(t_i, p_k)$ denotes the minimum chamfer distance between the template $t_i$ and the prototype $p_k$ for all relative displacements between them. The process is repeated recursively by treating the prototypes as templates and re-clustering them to form a tree as shown in Fig. 2.

For recognition, the given image is recursively matched with the nodes of the template tree, starting from the root. At any level, the branches where the minimum chamfer distance is greater than a threshold are pruned to reduce the search time. For remaining nodes, matching is repeated for all children nodes. Candidates are generated at image positions where the chamfer distance with any of the template nodes is below threshold.

**Motion-Based Detection**

Motion is an important cue in detecting pedestrians. In the case of moving platforms, the background undergoes ego-motion that depends on camera motion as well as the scene structure, which needs to be accounted for. In [15, 16], a parametric planar motion model is used to describe the ego-motion of the ground in an omnidirectional camera. The perspective coordinates of a point $P$ on the ground in two consecutive camera positions is governed by a homography matrix $H$ as:

$$\begin{pmatrix} X_b \\ Y_b \\ Z_b \end{pmatrix} = \lambda \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{32} & h_{33} \\ h_{31} & h_{32} & 1 \end{pmatrix} \begin{pmatrix} X_a \\ Y_a \\ Z_a \end{pmatrix} \tag{4}$$

The perspective camera coordinates can be mapped to pixel coordinates $(u_a, v_a)$ and $(u_b, v_b)$ using the internal calibration of the camera:

$$\begin{aligned} (u_a, v_a) &= F_{int}([X_a, Y_a, Z_a]^T), \\ (u_b, v_b) &= F_{int}([X_b, Y_b, Z_b]^T) = F_{int}(HF_{int}^{-1}(u_a, v_a)) \end{aligned} \tag{5}$$

The image motion of the point satisfies the optical flow constraint:

$$g_u(u_b - u_a) + g_v(v_b - v_a) = -g_t + \nu \tag{6}$$

where $g_u$, $g_v$, and $g_t$ are the spatial and temporal image gradients and $\nu$ is the noise term.

Based on these relations, the parameters of the homography matrix can be estimated using the spatio-temporal image gradients at every point $(u_a, v_a)$ in the first image using non-linear least squares. Based on these parameters, every point $(u_a, v_a)$ on the ground plane in first frame corresponds to a point $(u_b, v_b)$ in the second frame. Using this transformation, the second image can be transformed to first frame, compensating the image motion of the ground plane. The objects that have independent motion or height above ground do not obey the motion model and their motion is not completely compensated. Taking the motion compensated frame difference between adjacent video frames highlights these areas that are likely to contain pedestrians, vehicles, or other obstacles. These regions of interest can then be classified using the classification stage. Figure 3 shows detection of a pedestrian and vehicle from a moving platform. The details of the approach are described in [15].

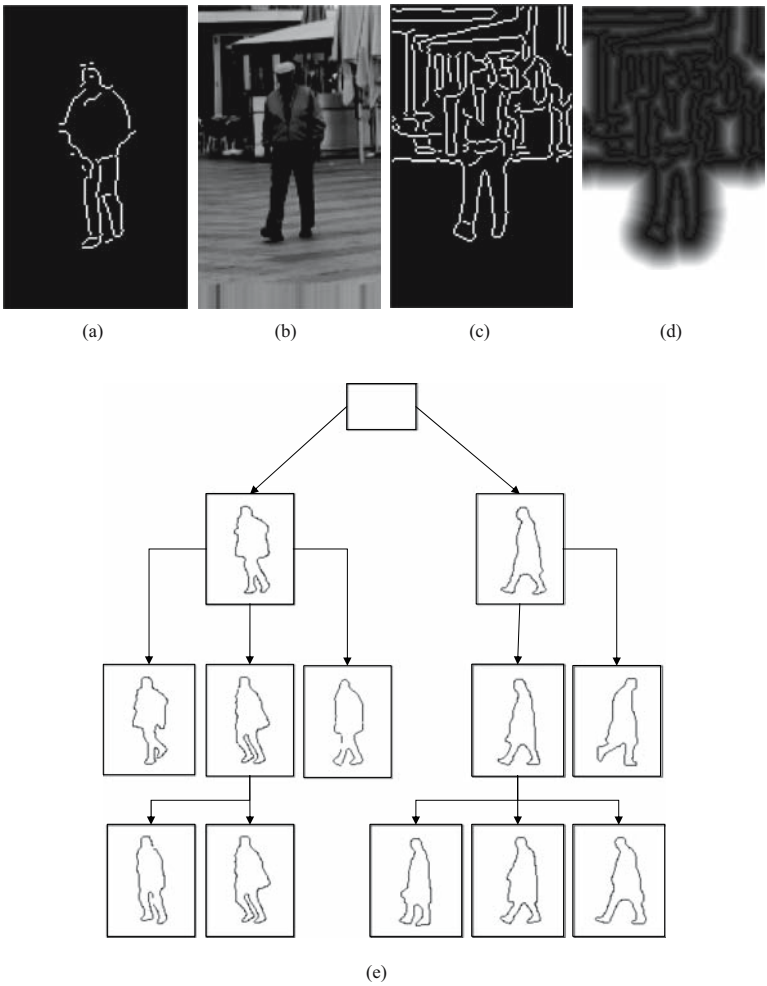



**Fig. 2.** Chamfer matching illustration: (**a**) template of pedestrian (**b**) test image (**c**) edge image (**d**) distance transform image (**e**) template hierarchy (partial) used for matching with distance transform (figure based on [27])

### Depth Segmentation Using Binocular Stereo

Binocular stereo imaging can provide useful information about the depth of the objects from the cameras. This information has been used for disambiguating pedestrians and other objects from features on ground plane [18, 25, 26], segmenting images based on layers with different depths [14], handling occlusion between pedestrians [25], and using the size-depth relation to eliminate extraneous objects [32]. For a pair of stereo cameras with focal length $f$ and baseline distance of $B$ between the cameras situated at the height of $H$

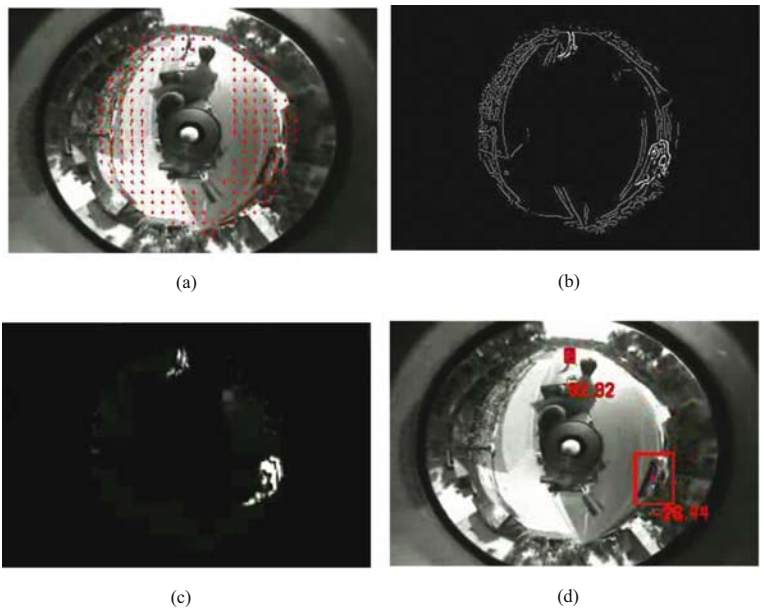



**Fig. 3.** Detection of people and vehicles using an omnidirectional camera on a moving platform [15] (**a**) estimated motion based on parametric motion of ground plane (**b**) image pixels used for estimation. Gray pixels are inliers, and white pixels are outliers. (**c**) Motion-compensated image difference that captures independently moving objects (**d**) detected vehicle and person

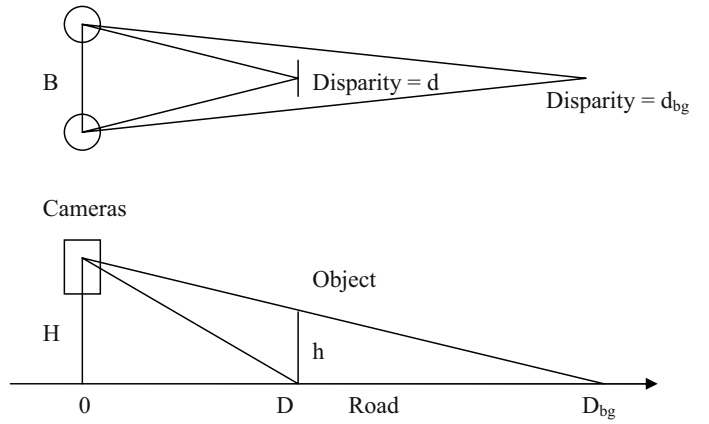
above the road as shown in Fig. 4a. An object at distance $D$ will have disparity of $d = Bf/D$ between the two cameras, that is inversely proportional to object distance. The ground in the same line of sight is farther away and has a smaller disparity of $d_{bg} = Bf/D_{bg}$. Based on this difference, objects having height above the ground can be separated.

Stereo disparity computation can be performed using software packages such as SRI Stereo Engine [24]. Such software produces a disparity map that gives disparities of individual pixels in the image. In [17], the concept of U-disparity proposed by Labayrade et al. [26] is used to identify potential obstacles in the scene using images from a stereo pair of omnidirectional cameras as shown in Fig. 4b. The disparity image $disp(u, v)$ generated by a stereo engine separates the pedestrian in a layer of nearly constant depth. U-disparity $udisp(u, d)$ image counts occurrences of every disparity $d$ for each column $u$ in the image. In order to suppress the ground plane pixels, only the pixels with disparity significantly greater than ground plane disparity are used.

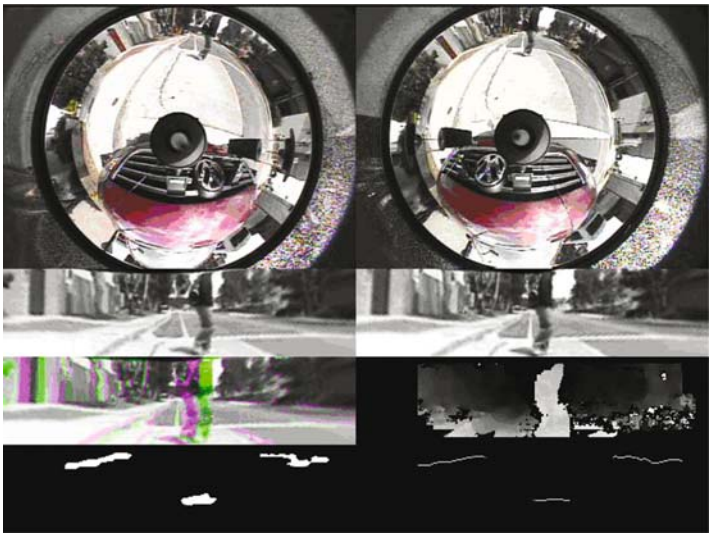
$$udisp(u, d) = \#\{v | disp(u, v) = d,\ d > d_{bg}(u, v) + d_{thresh}\} \tag{7}$$

where # stands for number of elements in the set.

Pixels in an object at a particular distance would have nearly same disparity and therefore form a horizontal ridge in the disparity histogram image. Even if disparities of individual object pixels are inaccurate, the histogram image clusters the disparities and makes it easier to isolate the objects. Based on the position of the line segments, the regions containing obstacles can be identified. The nearest (lowest) region with largest disparity corresponds to the pedestrian. The parts of the virtual view image corresponding to the U-disparity segments can then be sent to classifier for distinguishing between pedestrians and other objects.




(a)



(b)

**Fig. 4.** (**a**) Stereo geometry (top and side views). The disparity between image positions in the two cameras decrease with object distance. (**b**) Stereo based pedestrian candidate generation [17]: *Row 1*: Color images from a stereo pair of omni camera containing pedestrian. *Row 2*: Virtual front view images generated from omni images. *Row 3*: Superimposed front view images and disparity image with lighter shades showing nearer objects with larger disparity. *Row 4*: U-disparity image taking histogram of disparities for each column. The lower middle segment in U-disparity image corresponds to the pedestrian. Other segments corresponds to more distant structures above the ground (figure based on [17])

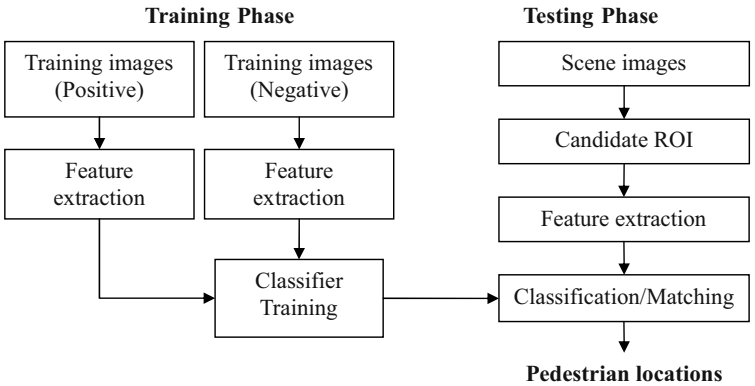



**Fig. 5.** Validation stage for pedestrian detection. Training phase uses positive and negative images to extract features and train a classifier. Testing phase applies feature extractor and classifier to candidate regions of interest in the images

### 3.2 Candidate Validation

The candidate generation stage generates regions of interest (ROI) that are likely to contain a pedestrian. Characteristic features are extracted from these ROIs and a trained classifier is used to separate pedestrian from the background and other objects. The input to the classifier is a vector of raw pixel values or characteristic features extracted from them, and the output is the decision showing whether a pedestrian is detected or not. In many cases, the probability or a confidence value of the match is also returned. Figure 5 shows the flow diagram of validation stage.

#### Feature Extraction

The features used for classification should be insensitive to noise and individual variations in appearance and at the same time able to discriminate pedestrians from other objects and background clutter. For pedestrian detection features such as Haar wavelets [28], histogram of oriented gradients [13], and Gabor filter outputs [12], are used.

*Haar Wavelets*

An object detection system needs to have a representation that has high inter-class variability and low intra-class variability [28]. For this purpose, features must be identified at resolutions where there will be some consistency throughout the object class, while at the same time ignoring noise. Haar wavelets extract local intensity gradient features at multiple resolution scales in horizontal, vertical, and diagonal directions and are particularly useful in efficiently representing the discriminative structure of the object. This is achieved by sliding the wavelet functions in Fig. 6 over the image and taking inner products as:

$$w_k(m,n) = \sum_{m=0}^{2^k-1} \sum_{n=0}^{2^k-1} \psi_k(m',n') f(2^{k-j}m + m', 2^{k-j}n + n') \tag{8}$$

where $f$ is the original image, $\psi_k$ is any of the wavelet functions at scale $k$ with support of length $2^k$, and $2^j$ is the over-sampling rate. In the case of standard wavelet transforms, $k = 0$ and the wavelet is translated at each sample by the length of the support as shown in Fig. 6. However, in over-complete representations, $k > 0$ and the wavelet function is translated only by a fraction of the length of support. In [28] the over-complete representation with quarter length sampling is used in order to robustly capture image features.

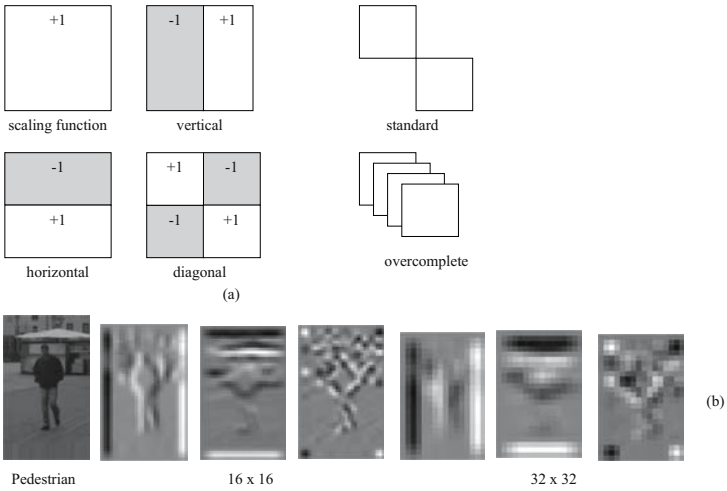



**Fig. 6.** Haar wavelet transform framework. *Left*: Scaling and wavelet functions at a particular scale. *Right*: Standard and overcomplete wavelet transforms (figure based on [28])

The wavelet transform can be concatenated to form a feature vector that is sent to a classifier. However, it is observed that some components of the transform have more discriminative information than others. Hence, it is possible to select such components to form a truncated feature vector as in [28] to reduce complexity and speed up computations.

*Histograms of Oriented Gradients*

Histograms of oriented gradients (HOG) have been proposed by Dalal and Triggs [13] to classify objects such as people and vehicles. For computing HOG, the region of interest is subdivided into rectangular blocks and histogram of gradient orientations is computed in each block. For this purpose, sub-images corresponding to the regions suspected to contain pedestrian are extracted from the original image. The gradients of the sub-image are computed using Sobel operator [22]. The gradient orientations are quantized into K bins each spanning an interval of $2\pi/K$ radians, and the sub-image is divided into $M \times N$ blocks. For each block $(m, n)$ in the subimage, the histogram of gradient orientations is computed by counting the number of pixels in the block having the gradient direction of each bin $k$. This way, an $M \times N \times K$ array consisting of $M \times N$ local histograms is formed. The histogram is smoothed by convolving with averaging kernels in position and orientation directions to reduce sensitivity to discretization. Normalization is performed in order to reduce sensitivity to illumination changes and spurious edges. The resulting array is then stacked into a $B = MNK$ dimensional feature vector $\mathbf{x}$. Figure 7 shows examples with pedestrian snapshots along with the HOG representation shown by red lines. The value of a histogram bin for a particular position and orientation is proportional to the length of the respective line.

## Classification

The classifiers employed to distinguish pedestrians from non-pedestrian objects are usually trained using feature vectors extracted from a number of positive and negative examples to determine the decision boundary

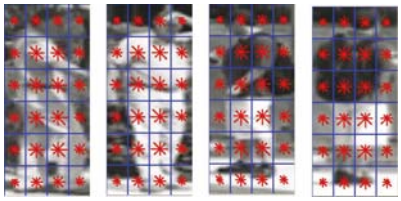


**Fig. 7.** Pedestrian subimages with computed Histograms of Oriented Gradients (HOG). The image is divided into blocks and the histogram of gradient orientations is individually computed for each block. The lengths of the *red lines* correspond to the frequencies of image gradients in the respective directions

between them. After training, the classifier processes unknown samples and decides the presence or absence of the object based on which side of the decision boundary the feature vector lies. The classifiers used for pedestrian detection include Support Vector Machines (SVM), Neural Networks, and AdaBoost, which are described here.

*Support Vector Machines*

The Support Vector Machine (SVM) forms a decision boundary between two classes by maximizing the "margin," i.e., the separation between nearest examples on either side of the boundary [11]. SVM in conjunction with various image features are widely used for pedestrian recognition. For example, Papageorgiou and Poggio [28] have designed a general object detection system that they have applied to detect pedestrians for a driver assistance. The system uses SVM classifier on Haar wavelet representation of images. A support vector machine is trained using a large number of positive and negative examples from which the image features are extracted. Let $\mathbf{x}_i$ denote the feature vector of sample $i$ and $y_i$ denote one of the two class labels in $\{0, 1\}$. The feature vector $\mathbf{x}_i$ is projected into a higher dimensional kernel space using a mapping function $\Phi$ which allows complex non-linear decision boundaries. The classification can be formulated as an optimization problem to find a hyperplane boundary in the kernel space:

$$\mathbf{w}^T \Phi(\mathbf{x}_i) + b = 0 \tag{9}$$

using

$$\min_{\mathbf{w},b,\boldsymbol{\xi},\rho} \mathbf{w}^T\mathbf{w} - \nu\rho + \frac{1}{L}\sum_{i=1}^{L} \xi_i \tag{10}$$

subject to

$$\mathbf{w}^T \Phi(\mathbf{x}_i) + b \geq \rho - \xi_i \ , \ \xi_i \geq 0, i = 1 \ldots L, \rho \geq 0$$

where $\nu$ is the parameter to accommodate training errors and $\boldsymbol{\xi}$ is used to account for some samples that are not separated by the boundary. Figure 8 illustrates the principle of SVM for classification of samples. The problem is converted into the dual form which is solved using quadratic programming [11]:

$$\min_{\boldsymbol{\alpha}} \sum_{i=1}^{L}\sum_{j=1}^{L} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_j) y_j \alpha_j \tag{11}$$

subject to

$$0 \leq \alpha_i \leq 1/L, \sum_{i=1}^{L} \alpha_i \geq \nu, \sum_{i=1}^{L} \alpha_i y_i = 0 \tag{12}$$

where $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_j)$ is the kernel function derived from the mapping function $\Phi$, and represents the distance in the high-dimensional space. It should be noted that the kernel function is usually much easier to compute than the mapping function $\Phi$. The classification is then given by the decision function:

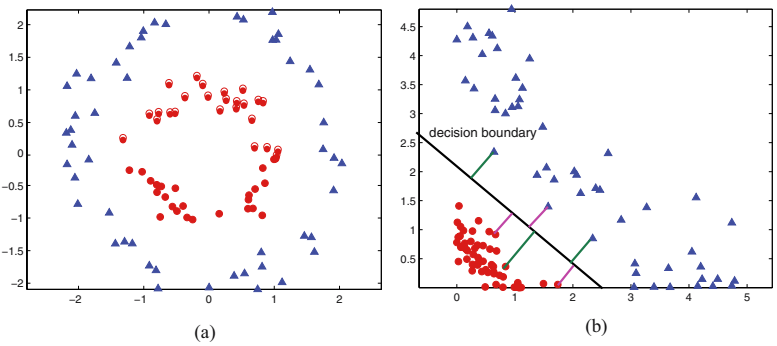



**Fig. 8.** Illustration of Support Vector Machine principle. (**a**) Two classes that cannot be separated by a single straight line. (**b**) Mapping into Kernel space. SVM finds a line separating two classes to minimize the "margin," i.e., the distance to the closest samples called 'Support Vectors'

$$D(\mathbf{x}) = \sum_{i=1}^{L} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b \tag{13}$$

*Neural Networks*

Neural networks have been used to address problems in vehicle diagnostics and control [31]. They are particularly useful when the phenomenon to be modeled is highly complex but one has large amount of training data to enable learning of patterns from them. Neural networks can obtain highly non-linear boundaries between classes based on the training samples, and therefore can account for large shape variations. Zhao and Thorpe [41] have applied neural networks on gradient images of regions of interest to identify pedestrians. However, unconstrained neural networks require training of a large number of parameters necessitating very large training sets. In [21, 27], Gavrila and Munder use Local receptive fields (LRF) proposed by Wöhler and Anlauf [39] (Fig. 9) to reduce the number of weights by connecting each hidden layer neuron only to a local region of input image. Furthermore, the hidden layer is divided into a number of branches, each encoding a local feature, with all neurons within a branch sharing the same set of weights. Each hidden layer can be represented by the equation:

$$G_k(r) = f\left[\sum_i W_{ki} F(T(r) + \Delta r_i)\right] \tag{14}$$

where $F(p)$ denotes the input image as a function of pixel coordinates $p = (x, y)$, $G_k(r)$ denotes the output of the neuron with coordinate $r = (r_x, r_y)$ in the branch $k$ of the hidden layer, $W_{ki}$ are the shared weights for branch $k$, and $f(\cdot)$ is the activation function of the neuron. Each neuron with coordinates of $r$ is associated with a region in the image around the transformed pixel $t = T(r)$, and $\Delta r_i$ denote the displacements for pixels in the region. The output layer is a standard fully connected layer given by:

$$H_m = f\left[\sum_i w_{mk}(x, y) G_k(x, y)\right] \tag{15}$$

where $H_m$ is the output of neuron $m$ in output layer, $w_{mk}$ is the weight for connection between output neuron $m$ and hidden layer neuron in branch $k$ with coordinate $(x, y)$.

LeCun et al. [40] describe similar weight-shared and grouped networks for application in document analysis.

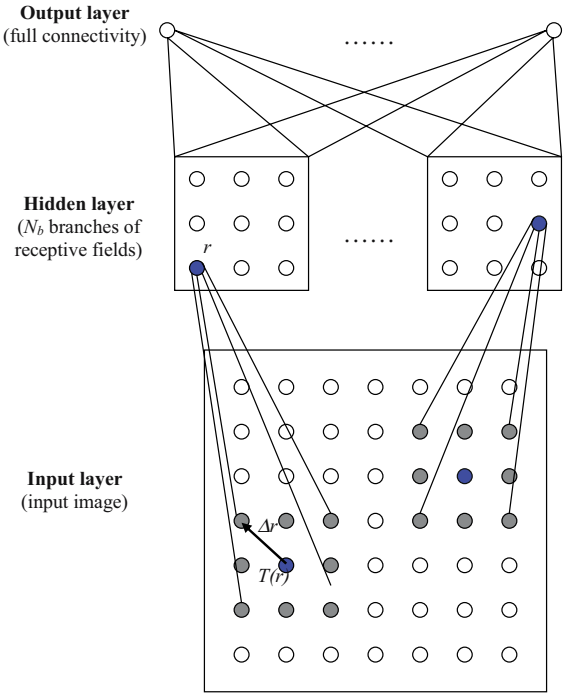



**Fig. 9.** Neural network architecture with Local Receptive Fields (figure based on [27])

*Adaboost Classifier*

Adaboost is a scheme for forming a strong classifier using a linear combination of a number of weak classifiers based on individual features [36, 37]. Every weak classifier is individually trained on a single feature. For boosting the weak classifier, the training examples are iteratively re-weighted so that the samples which are incorrectly classified by the weak classifier are assigned larger weights. The final strong classifier is a weighted combination of weak classifiers followed by a thresholding step. The boosting algorithm is described as follows [8, 36]:

- Let $\mathbf{x}_i$ denote the feature vector and $y_i$ denote one of the two class labels in $\{0, 1\}$ for negative and positive examples, respectively
- Initialize weights $w_i$ to $1/2M$ for each of the $M$ negative samples and $1/2L$ for each of the $L$ positive samples
- Iterate for $t = 1 \ldots T$
  - Normalize weights: $w_{t,i} \leftarrow w_{t,i} / \sum_k w_{t,k}$
  - For each feature $j$, train classifier $h_j$ that uses only that feature. Evaluate weighted error for all samples as: $\epsilon_j = \sum_i w_{t,i} |h_j(\mathbf{x}_i) - y_i|$
  - Choose classifier $h_t$ with lowest error $\epsilon_t$
  - Update weights: $w_{t+1,i} \leftarrow w_{t,i} \left( \frac{\epsilon_t}{1-\epsilon_t} \right)^{1-|h_j(\mathbf{x}_i)-y_i|}$

– The final strong classifier decision is given by the linear combination of weak classifiers and thresholding the result: $\sum_t \alpha_t h_t(\mathbf{x}) \geq \sum_t \alpha_t/2$ where $\alpha_t = \log\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$

## 4 Infrastructure Based Systems

Sensors mounted on vehicles are very useful for detecting pedestrians and other vehicles around the host vehicle. However, these sensors often cannot see objects that are occluded by other vehicles or stationary structures. For example, in the case of the intersection shown in Fig. 10, the host vehicle X cannot see the pedestrian P occluded by a vehicle Y as well as the vehicle Z occluded by buildings. Sensor C mounted on infrastructure would be able to see all these objects and help to fill the 'holes' in the fields of view of the vehicles. Furthermore, if vehicles can communicate with each other and the infrastructure, they can exchange information about objects that are seen by one but not seen by others. In the future, infrastructure based scene analysis as well as infrastructure-vehicle and vehicle-vehicle communication will contribute towards robust and effective working of Intelligent Transportation Systems.

Cameras mounted in infrastructure have been extensively applied to video surveillance as well as traffic analysis [34]. Detection and tracking of objects from these cameras is easier and more reliable due to absence of camera motion. Background subtraction which is one of the standard methods to extract moving objects from stationary background is often employed, followed by classification of objects and activities.

### 4.1 Background Subtraction and Shadow Suppression

In order to separate moving objects from background, a model of the background is generated from multiple frames. The pixels not satisfying the background model are identified and grouped to form regions of interest that can contain moving objects. A simple approach for modeling the background is to obtain the statistics of each pixel described by color vector $\mathbf{x} = (R, G, B)$ over time in terms of mean and variance. The mean and variance are updated at every time frame using:

$$\begin{aligned} \boldsymbol{\mu} &\leftarrow (1-\alpha)\boldsymbol{\mu} + \alpha\mathbf{x} \\ \sigma^2 &\leftarrow (1-\alpha)\sigma^2 + \alpha(\mathbf{x}-\boldsymbol{\mu})^T(\mathbf{x}-\boldsymbol{\mu}) \end{aligned} \tag{16}$$

If for a pixel at any given time, $\|\mathbf{x} - \boldsymbol{\mu}\|/\sigma$ is greater than a threshold (typically 2.5), the pixel is classified as foreground. Schemes have been designed that adjust the background update according to the pixel

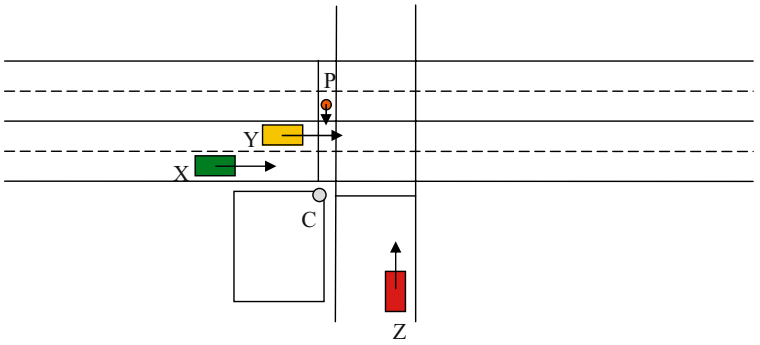



**Fig. 10.** Contribution of sensors mounted in infrastructure. Vehicle X cannot see pedestrian P or vehicle Z, but the infrastructure mounted camera C can see all of them

currently being in foreground or background. More elaborate models such as Gaussian Mixture Models [33] and codebook model [23] are used to provide robustness against fluctuating motion such as tree branches, shadows, and highlights.

An important problem in object-background segmentation is the presence of shadows and highlights of the moving objects, which need to be suppressed in order to get meaningful object boundaries. Prati et al. [30] have conducted a survey of approaches used for shadow suppression. An important cue for distinguishing shadows from background is that the shadow reduces the luminance value of a background pixel, with little effect on the chrominance. Highlights similarly increase the value of luminance. On the other hand, objects are more likely to have different color from the background and brighter than the shadows. Based on these cues, bright objects can often be separated from shadows and highlights.

### 4.2 Robust Multi-Camera Detection and Tracking

Multiple cameras offer superior scene coverage from all sides, provide rich 3D information, and enable robust handling of occlusions and background clutter. In particular, they can help to obtain the representation of the object that is independent of viewing direction. In [29], multiple cameras with overlapping fields of view are used to track persons and vehicles. Points on the ground plane can be projected from one view to another using a planar homography mapping. If $(u_1, v_1)$ and $(u_2, v_2)$ are image coordinates of a point on ground plane in two views, they are related by the following equations:

$$u_2 = \frac{h_{11}u_1 + h_{12}v_1 + h_{13}}{h_{31}u_1 + h_{32}v_1 + h_{33}},\ v_2 = \frac{h_{21}u_1 + h_{22}v_1 + h_{23}}{h_{31}u_1 + h_{32}v_1 + h_{33}} \tag{17}$$

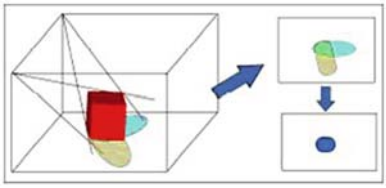
The matrix $H$ formed from elements $h_{ij}$ is the Homography matrix. Multiple views of the same object are transformed by planar homography which assumes that pixels lie on ground plane. Pixels that violate this assumption result in mapping to a skewed location. Hence, the common footage region of the object on ground can be obtained by intersecting multiple projections of the same object on the ground plane. The footage area on the ground plane gives an estimate of the size and the trajectory of the object, independent of the viewing directions of the cameras. Figure 11 depicts the process of estimating the footage area using homography. The locations of the footage areas are then tracked using Kalman filter in order to obtain object trajectories.

### 4.3 Analysis of Object Actions and Interactions

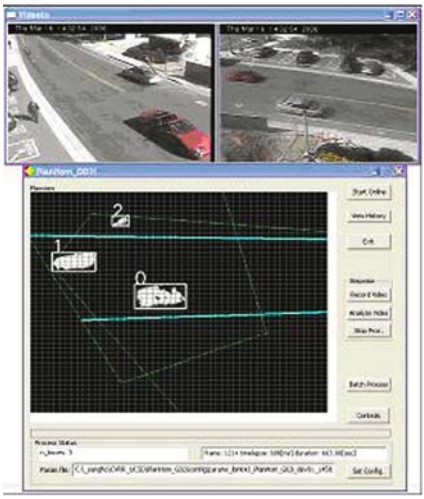
The objects are classified into persons and vehicles based on their footage area. The interaction among persons and vehicles can then be analyzed at semantic level as described in [29]. Each object is associated with spatio-temporal interaction potential that probabilistically describes the region in which the object can be subsequent time. The shape of the potential region depends on the type of object (vehicle/pedestrian) and speed (larger region for higher speed), and is modeled as a circular region around the current position. The intersection of interaction potentials of two objects represents the possibility of interaction between them as shown in Fig. 12a. They are categorized as safe or unsafe depending on the site context such as walkway or driveway, as well as motion context in terms of trajectories. For example, as shown in Fig. 12b, a person standing on walkway is normal scenario, whereas the person standing on driveway or road represents a potentially dangerous situation. Also, when two objects are moving fast, the possibility of collision is higher than when they are traveling slowly. This domain knowledge can be fed into the system in order to predict the severity of the situation.

## 5 Pedestrian Path Prediction

In addition to detection of pedestrians and vehicles, it is important to predict what path they are likely to take in order to estimate the possibility of collision. Pedestrians are capable of making sudden maneuvers in terms of the speed and direction of motion. Hence, probabilistic methods are most suitable for predicting



(a)



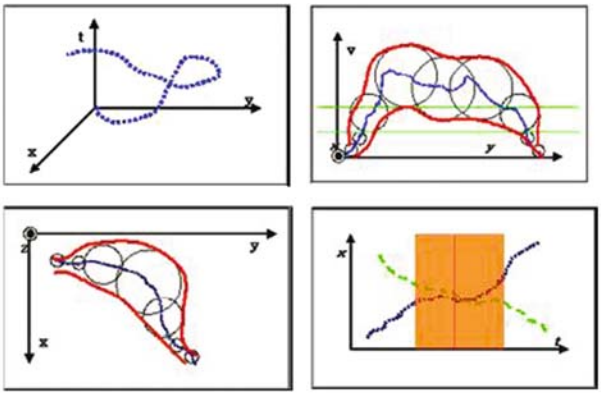
(b)

**Fig. 11.** (**a**) Homography projection from two camera views to virtual top views. The footage region is obtained by the intersection of the projections on ground plane. (**b**) Detection and mapping of vehicles and a person in virtual top view showing correct sizes of objects [29]

the pedestrian's future path and potential collisions with vehicles. In fact, even for vehicles whose paths are easier to predict due to simpler dynamics, predictions beyond 1 or 2 seconds is still very challenging, making probabilistic methods valuable even for vehicles.

For probabilistic prediction, Monte-Carlo simulations can be used to generate a number of possible trajectories based on the dynamic model. The collision probability is then predicted based on the fraction of trajectories that eventually collide with the vehicle. Particle filtering [10] gives a unified framework for integrating the detection and tracking of objects with risk assessment as in [8]. Such a framework is shown in Fig. 13a with following steps:

1. Every tracked object can be modeled using a state vector consisting of properties such as 3-D position, velocity, dimensions, shape, orientation, and other appropriate attributes. The probability distribution of the state can then be modeled using a number of weighted samples randomly chosen according to the probability distribution.
2. The samples from the current state are projected to the sensor fields of view. The detection module would then produce hypotheses about the presence of vehicles. The hypotheses can then be associated with the samples to produce likelihood values used to update the sample weights.




(a)

| Site \ Person | Stay | Walk | Run |
|---|---|---|---|
| Walkway | ○ | ○ | × |
| Driveway | × | ○ | ○ |
| ROI | ○,× | ○ | × |

(b)

| Object-1 \ Object-2 | Stay | Slow | Fast |
|---|---|---|---|
| Stay | ○ | ○ | △ |
| Slow | ○ | △ | × |
| Fast | △ | × | × |

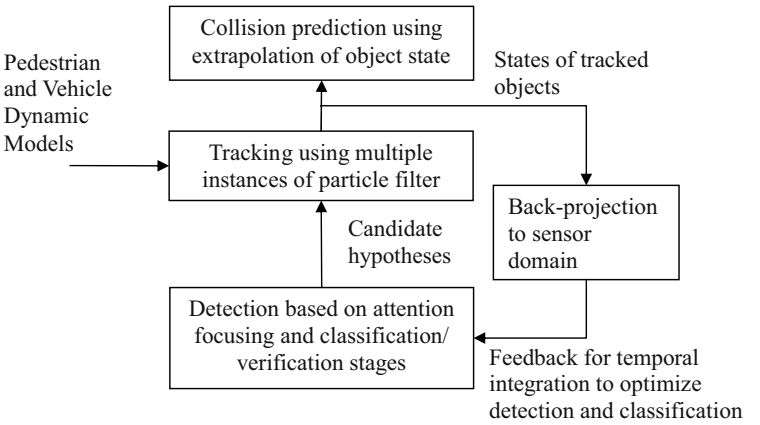
(c)

**Fig. 12.** (**a**) Schematic diagrams for trajectory analysis in spatio-temporal space. *Circles* represent interaction potential boundaries at a given space/time. *Red curves* represent the envelopes of the interaction boundary along tracks. (**b**) Spatial context dependency of human activity (**c**) Temporal context dependency of interactivity between two objects. Track patterns are classified into normal (*open circle*), cautious (*open triangle*) and abnormal (*times*) [29]

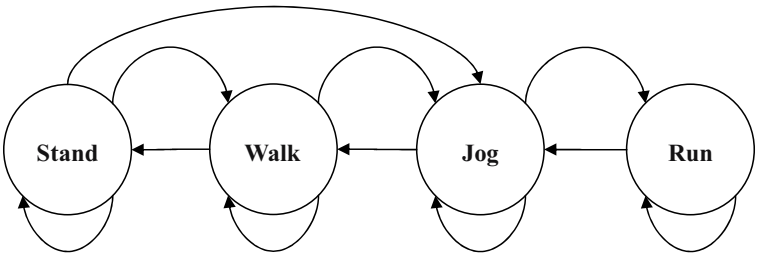
3. The object state samples can be updated at every time instance using the dynamic models of pedestrians and vehicles. These models put constraints on how the pedestrian and vehicle can move over short and long term.
4. In order to predict collision probability, the object state samples are extrapolated over a longer period of time. The number of samples that are on collision course divided by the total number of samples gives the probability of collision.

Various dynamic models can be used for predicting the positions of the pedestrians at subsequent time. For example, in [38], Wakim et al. model the pedestrian dynamics using Hidden Markov Model with four states corresponding to standing still, walking, jogging, and running as shown in Fig. 13b. For each state, the probability distributions of absolute speed as well as the change of direction is modeled by truncated Gaussians. Monte Carlo simulations are then used to generate a number of feasible trajectories and the ratio of the trajectories on collision course to total number of trajectories give the collision probability. The European project CAMELLIA [5] has conducted research in pedestrian detection and impact prediction based in part on [8, 38]. Similar to [38], they use a model for pedestrian dynamics using HMM. They use the position of pedestrian (sidewalk or road) to determine the transition probabilities between different gaits and orientations. Also, the change in orientation is modeled according to the side of the road that the pedestrian is walking.

In [9], Antonini et al. another approach called "Discrete Choice Model" which a pedestrian makes a choice at every step about the speed and direction of the next step. Discrete choice models associate a utility




(a)




(b)

**Fig. 13.** (**a**) Integration of detection, tracking, and risk assessment of pedestrians and other objects based on particle filter [10] framework. (**b**) Transition diagram between states of pedestrians in [38]. The *arrows* between two states are associated with non-zero probabilities of transition from one state to another. *Arrows* on the same state corresponds to the pedestrian remaining in the same state in the next time step

value to every such choice and select the alternative with the highest utility. The utility of each alternative is a latent variable depending on the attributes of the alternative and the characteristics of the decision-maker. This model is integrated with person detection and tracking from static cameras in order to improve performance. Instead of making hard decisions about target presence on every frame, it integrates evidence from a number of frames before making a decision.

## 6 Conclusion and Future Directions

Pedestrian detection, tracking, and analysis of behavior and interactions between pedestrians and vehicles are active research areas having important application in protection of pedestrians on road. Pattern classification approaches are particularly useful in detecting pedestrians and separating them from background. It is

seen that pedestrian detection can be performed using sensors on vehicle itself or in the infrastructure. Vehicle based sensing gives continuous awareness of the scene around the vehicle and systems are being designed to detect pedestrians from vehicles. However, it is also seen that relying on vehicle sensors is not always sufficient to give full situational awareness of the scene. Infrastructure based sensors can play a complementary role of providing wide area scene coverage. For seamless integration of information from vehicle and infrastructure, efficient and reliable communication is needed. Communication can be performed at image level or object level. However, transmitting full images over the network is likely to be very expensive in terms of bandwidth. Hence, it would be desirable to perform initial detection locally and transmit candidate positions and trajectories along with sub-images of candidate bounding boxes as needed. Future research will be directed towards developing and standardizing these communications between vehicle and infrastructure to efficiently convey all the information needed to get complete situational awareness of the scene.

**Acknowledgment**

The authors thank the UC Discovery Grant with several automobile companies, Department of Defense Technical Support Working Group, National Science Foundation for the sponsorship of the research. The authors also thank the members in the Computer Vision and Robotics Research Laboratory including Dr. Stephen Krotosky, Brendan Morris, Erik Murphy-Chutorian, and Dr. Sangho Park for their contributions.

## References

1. Advanced Highway Systems Program, Japanese Ministry of Land, Infrastructure and Transport, Road Bureau. http://www.mlit.go.jp/road/ITS/index.html.
2. http://prevent.ertico.webhouse.net/en/home.htm, http://prevent.ertico.webhouse.net/en/prevent_subprojects/vulnerable_road_users_collision_mitigation/apalaci/.
3. http://www.path.berkeley.edu/.
4. http://www.walkinginfo.org/pedsmart.
5. Deliverable 3.3b report on initial algorithms 2. Technical Report IST-2001-34410, CAMELLIA: Core for Ambient and Mobile Intelligent Imaging Applications, December 2003.
6. *IEEE Intelligent Vehicle Symposium*, Istanbul, Turkey, June 2007.
7. *IEEE International Transportation Systems Conference*, Seattle, WA, September 2007.
8. Y. Abramson and B. Steux. Hardware-friendly pedestrian detection and impact prediction. In *IEEE Intelligent Vehicle Symposium*, pp. 590–595, June 2004.
9. G. Antonini, S. Venegas, J.P. Thiran, and M. Bierlaire. A discrete choice pedestrian behavior model for pedestrian detection in visual tracking systems. In *Proceedings of Advanced Concepts for Intelligent Vision Systems*, September 2004.
10. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.
11. C.-C. Chang and C.-J. Lin. *LIBSVM: A Library for Support Vector Machines*, Last updated June 2007.
12. H. Cheng, N. Zheng, and J. Qin. Pedestrian detection using sparse gabor filters and support vector machine. In *IEEE Intelligent Vehicle Symposium*, pp. 583–587, June 2005.
13. N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, June 2005.
14. U. Franke. Real-time stereo vision for urban traffic scene understanding. In *IEEE Intelligent Vehicle Symposium*, pp. 273–278, 2000.
15. T. Gandhi and M.M. Trivedi. Motion analysis for event detection and tracking with a mobile omni-directional camera. *Multimedia Systems Journal, Special Issue on Video Surveillance*, 10(2):131–143, 2004.
16. T. Gandhi and M.M. Trivedi. Parametric ego-motion estimation for vehicle surround analysis using an omnidirectional camera. *Machine Vision and Applications*, 16(2):85–95, 2005.
17. T. Gandhi and M.M. Trivedi. Vehicle mounted wide FOV stereo for traffic and pedestrian detection. In *Proceedings of International Conference on Image Processing*, pp. 2:121–124, 2005.
18. T. Gandhi and M.M. Trivedi. Vehicle surround capture: Survey of techniques and a novel omni video based approach for dynamic panoramic surround maps. *IEEE Transactions on Intelligent Transportation Systems*, 7(3):293–308, 2006.

19. T. Gandhi and M.M. Trivedi. Pedestrian protection systems: Issues, survey, and challenges. *IEEE Transactions on Intelligent Transportation Systems*, 8(3), 2007.
20. D.M. Gavrila. Pedestrian detection from a moving vehicle. In *Proceedings of European Conference on Computer Vision*, pp. 37–49, 2000.
21. D.M. Gavrila and S. Munder. Multi-cue pedestrian detection and tracking from a moving vehicle. *International Journal of Computer Vision*, 73(1):41–59, 2007.
22. R.C. Gonzalez and R.E. Woods. *Digital Image Processing*. Prentice Hall, Upper Saddle River, NJ, 3rd edition, 2008.
23. K. Kim, T.H. Chalidabhongse, D. Harwood, and L.S. Davis. Real-time foreground-background segmentation using codebook model. *Real-Time Imaging*, 11(3):172–185, 2005.
24. K. Konolige. Small vision system: Hardware and implementation. In *Eighth International Symposium on Robotics Research*, pp. 111–116, 1997. http://www.ai.sri.com/~konolige/papers.
25. S.J. Krotosky and M.M. Trivedi. A comparison of color and infrared stereo approaches to pedestrian detection. In *IEEE Intelligent Vehicles Symposium*, June 2007.
26. R. Labayrade, D. Aubert, and J.-P. Tarel. Real time obstacle detection in stereovision on non flat road geometry through V-disparity representation. In *IEEE Intelligent Vehicles Symposium*, volume II, pp. 646–651, 2002.
27. S. Munder and D.M. Gavrila. An experimental study on pedestrian classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1863–1868, 2006.
28. C. Papageorgiou and T. Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38(1):15–33, 2000.
29. S. Park and M.M. Trivedi. Video Analysis of Vehicles and Persons for Surveillance. *Intelligent and Security Informatics: Techniques and Applications*, Springer, Berlin Heidelberg New York, 2007.
30. A. Prati, I. Mikic, M.M. Trivedi, and R. Cucchiara. Detecting moving shadows: Algorithms and evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 918–923, July 2003.
31. D.V. Prokhorov. Neural Networks in Automotive Applications. *Computational Intelligence in Automotive Applications*, Studies in Computational Intelligence, Springer, Berlin Heidelberg New York, 2008.
32. M. Soga, T. Kato, M. Ohta, and Y. Ninomiya. Pedestrian detection with stereo vision. In *International Conference on Data Engineering*, April 2005.
33. C. Stauffer and W.E.L. Grimson. Adaptive background mixture model for real-time tracking. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 246–252, 1999.
34. M.M. Trivedi, T. Gandhi, and K.S. Huang. Distributed interactive video arrays for event capture and enhanced situational awareness. *IEEE Intelligent Systems, Special Issue on AI in Homeland Security*, 20(5):58–66, September–October 2005.
35. M.M. Trivedi, T. Gandhi, and J. McCall. Looking-in and looking-out of a vehicle: Computer vision based enhanced vehicle safety. *IEEE Transactions on Intelligent Transportation Systems*, 8(1):108–120, March 2007.
36. P. Viola and M.J. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. I:511–518, June 2001.
37. P. Viola, M.J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. *International Journal of Computer Vision*, 63(2):153–161, 2005.
38. C. Wakim, S. Capperon, and J. Oksman. A markovian model of pedestrian behavior. In *Proceedings of IEEE Intelligent Conference on Systems, Man, and Cybernetics*, pp. 4028–4033, October 2004.
39. C. Wöhler and J. Anlauf. An adaptable time-delay neural-network algorithm for image sequence analysis. *IEEE Transactions on Neural Networks*, 10(6):1531–1536, 1999.
40. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998.
41. L. Zhao and C. Thorpe. Stereo and neural network-based pedestrian detection. *IEEE Transactions Intelligent Transportation*, 1(3):148–154, September 2000.