2011 14th International IEEE Conference on
Intelligent Transportation Systems
Washington, DC, USA. October 5-7, 2011

Real-Time GPU Implementation of an Improved Cars, Pedestrians and Bicycles Detection and Classification System

Alessandro Moro[1 2], Enzo Mumolo[2], Massimiliano Nolich[2] and Kazunori Umeda[1]
[1] Chuo University / CREST-JST, Japan
[2] Industrial and Information Engineering Department, University of Trieste, Italy

Abstract— **In this paper a real time system for cars, pedestrians and bicycle detection and classification is presented. The system aims at monitoring the traffic flow in urban zones and uses video data acquired with both mono and stereo cameras. All the algorithms have been developed in a pixel-wise fashion in order to be parallelized on a GPU device for real-time performances. We show that by a GPU implementation of the time-consuming parts of the proposed system, we perform detection and classification at about 25 frame per second to ensure prompt and effective reaction to the monitored events.**

I. Introduction

Systems capable of detecting and recognizing people and other moving objects from video data are getting more and more important due to the increasing complexity of the human environments. Applications include interactions among humans and devices for traffic management.The main difficulty in performing this task is that, in real environments, a large amount of objects categories are captured by a camera under different environmental conditions. In a street scenario, for example, pedestrians, cars, and bikes are the most common moving objects, but it is possible to detect unexpected objects such as barrows or animals. Moreover, these objects can interact in complicated ways.

The goal of this work is to detect and classify humans, cars and bicycles from acquired video streams. Our approach is based on the work described in [16] for the detection of Regions of Interest, and uses multi-class neural network pattern classifier for object labeling. The proposed system is based on a suite of improved algorithms specifically devised by us for performing background maintenance, shadow detection, ROI estimation and object classification. All the developed algorithms are pixel-wise and have been implemented on a GPU device for real-time performance. The main characteristics of the developed algorithms are the following. The background model, which is fundamental for moving object extraction, uses optimized thresholds for capturing the dynamic of the pixel intensities. The shadow removing algorithm removes from the detected moving object the pixels detected as shadow on the basis of several information extracted from video data, including the depth obtained with stereo vision, but also the classical between pixel and within pixel measurements, mixed together with a Dempster-Shafer data fusion framework. The importance of depth information in scene analysis and shadow detection has been remarked by several authors, for example by Torralba et al. [14], who state that there exists a strong relationship between structure of the scene and depth. As a matter of fact, thanks to the stereo information, the proposed shadow detection algorithm outperforms other classical approaches and helps increasing the classification accuracy. The detected moving pixels are grouped together using a Kalman tracking approach to reliably detect the moving objects. Finally, the objects, which are represented with HOG features, are classified with a multi-class pattern classifier.

The main contributions of this paper concern the preprocessing of stereo images for ROI detection based on improved background and shadow models. Moreover, another relevant contribution of this paper is that all the system algorithms have been implemented on a GPU for real-time performances.

The rest of this paper is organized as follows. In Section II we present and discuss the used methodologies. In Section III the proposed algorithms for the detection of objects, namely background model, shadow detection, pixel clustering and tracking, are described, and in Section IV the object classification is issued. In Section V the GPU implementation of the overall system is outlined. Experimental results and performances comparison with other state-of-the-art approaches are presented in Section VI. Finally, in Section VII some final remarks and conclusions are reported.

II. Methodology

Starting with a raw stereo video sequence, we compute the background model to extract the candidate moving pixels by background subtraction and we refine them by shadow detection and removing. Candidate moving pixels are grouped by a method proposed by Ubukata et al. [15] which exploits stereo information and refine results by a mean-shift clustering. The algorithm continues as follows. Detected regions are tracked by a Kalman filter method described in [4]. For each region, Histogram of Oriented Gradients (HOG) features [3] are estimated and fed into a multi-class Neural Network which estimates the probability that a region belongs to a certain group. It is important to note that cars and pedestrian are detected using an analysis window whose size is related to the depth data obtained from the stereo camera. Then, the labels of the moving regions are computed with the classification algorithm.

978-1-4577-2196-0/11/$26.00 ©2011 IEEE

III. Extraction of moving regions

A. Background model

In this work, to obtain a background model more precise as possible, we use an histogram-based pixel-wise algorithm. Background maintenance and other problematic aspects like shadow detection and noise reduction are faced separately.

First of all, the histogram value is increased by one if a difference is detected, and by two in case of no differences: we assign lower weights to pixels that frequently change their intensity value as they probably belong to moving objects rather than to the background. In the proposed technique, the color the background will assume corresponds to the first peak value of the histogram; this is repeated for all the histograms of the three color channels.

The difference between background and foreground is computed to establish which pixels of the image would be updated. The difference vector Δ is calculated as follows: $\Delta = \left[\left|I^R_{x,y} - B^R_{x,y}\right|, \left|I^G_{x,y} - B^G_{x,y}\right|, \left|I^B_{x,y} - B^B_{x,y}\right|\right]^T$ where (x, y) is the pixel position, I^c the intensity of the current image for the channel $c = (Red, Green, Blue)$, and B^c the intensity of the background image.

For each image I^c, at each frame t, the color distribution for each pixel (x, y) is calculated using histogram analysis:

$$H(t+1,a) = \begin{cases} H(t,a) + 2 \cdot \delta\left[I^c_{x,y} - a\right] & \text{if } \Delta \geq \tau \\ H(t,a) + \delta\left[I^c_{x,y} - a\right] & \text{otherwise} \end{cases} \tag{1}$$

where a is a color intensity, $\delta(\cdot)$ is the Dirac delta function defined as

$$\delta[p-q] = \begin{cases} 1 & \text{if } p = q \\ 0 & \text{if } p \neq q \end{cases}$$

and $\tau = [\tau^R, \tau^G, \tau^B]^T$ is a vector of thresholds used to detect changes in each channel.

At each frame t, for each pixel the numbers of Found Changes (FC) and Not Found Changes (NFC) are computed. FC and NFC are used to trigger the background updating phase, which is performed if the number of Changes Found for the pixel (x, y) is greater than an adaptively threshold, $\phi_{x,y}$, computed as

$$\phi_{x,y} = (\alpha_{x,y} - \beta_{x,y}) \cdot U \tag{2}$$

In equation (2), $\alpha_{x,y}$ and $\beta_{x,y}$ are weights describing the variability of the intensity of the pixel (x, y) and the number of changed pixels respectively and U is a parameter that have to be assigned in order to control the update rate of the background model.

More precisely, if we characterize with the binary matrix $M_{x,y}(t)$ the instantaneous change of pixel (x, y), i.e.

$$M_{x,y}(t) = \begin{cases} 1 & \text{if } \Delta \geq \tau \text{ at time t} \\ 0 & otherwise \end{cases} \tag{3}$$

The weights α and β are computed as

$$\alpha_{x,y} = \frac{1}{max\left(1, \sigma\left(x,y\right)\right)} \cdot \left(1 - \frac{1}{\gamma}\frac{\sum_{i=1}^{T} M_{x,y}(i)}{T})\right), \tag{4}$$

where the fraction $\frac{1}{\gamma}$ is typically around $\frac{1}{3}$, and

$$\beta_{x,y} = \frac{1}{\gamma} \cdot \left(\frac{\sum_{x,y} M_{x,y}}{\text{total pixels number}} + 1\right), \tag{5}$$

In conclusion, if $FC_{x,y} > \phi_{x,y}$ the pixel in the background is considered to be changed and hence its histogram model is updated. Moreover, if the model is changed, the background image should be reconstructed from the histogram model.

The threshold $\phi_{x,y}$ is computed independently for each pixel, thus leading to a better description of local dynamic changes in the image with respect to fixed threshold approaches as shown in the experimental section.

B. Shadow detection

In real environments, pixels detected by background subtraction may belong to a foreground object, or may represent light effects, or shadows. Changes in illumination can yield to false detection or can merge blobs. We use the beliefs, drawn from independent information sources, that the pixels of the moving region are a shadow. The beliefs are combined using the Dempster-Shafer theory of evidence.

1) The Dempster-Shafer Fusion: The goal of the Dempster-Shafer theory of evidence [12], is to represent uncertainty and lack of knowledge. The theory can combine different measures of evidence. At the base of the theory is a finite set of possible hypotheses, say $\theta = \{\theta_1, \ldots, \theta_K\}$.

In our case, a hypothesis set is defined for each texel in which is divided the image. Within each texel, the hypothesis concerns the possibility that the pixel (i, j) corresponds to an object or not. In other words, we have hypothesis for each pixel (i, j) of the moving region, namely θ = $\{\theta_1(i, j), \theta_2(i, j)\}$, where $\theta_1(i, j)$ is the belief that the pixel is a shadow and $\theta_2(i, j)$ is the belief that the pixel is not a shadow.

2) Combination of evidence: Consider two Basic Belief Assignments $m_1(.)$ and $m_2(.)$ and the corresponding belief functions $bel_1(.)$ and $bel_2(.)$. Let $\mathcal{A}_j$ and $\mathcal{B}_k$ be subsets of θ. Then $m_1(.)$ and $m_2(.)$ can be combined to obtain the belief mass assigned to $\mathcal{C} \subset \theta$ according to the following formula [12]:

$$m(\mathcal{C}) = m_1 \bigoplus m_2 = \frac{\sum_{j,k,\mathcal{A}_j \cap \mathcal{B}_k = \mathcal{C}} m_1(\mathcal{A}_j) m_2(\mathcal{B}_k)}{1 - \sum_{j,k,\mathcal{A}_j \cap \mathcal{B}_k = \varnothing} m_1(A_j) m_2(B_k)} \tag{6}$$

The denominator is a normalizing factor, which measures how much $m_1(.)$ and $m_2(.)$ are conflicting.

3) Belief functions combination: The combination rule can be easily extended to several belief functions by repeating the rule for new belief functions. Thus the sum of n belief functions $bel_1, bel_2, \ldots, bel_n$, can be formed as $((bel_1 \bigoplus bel_2) \bigoplus bel_3) \ldots bel_n = \bigoplus_{i=1}^{n} bel_i$. It is important to note that the basic beliefs combination formula given above assumes that the belief functions to be combined are independent.

4) Basic Belief Assignment for shadow detection: The basic beliefs are assigned with color consistency between pixels, color consistency within pixels, depth data and the level of reflected light.

The first belief is based on color consistency between pixels, defined as follows [7]:

$$\Psi(x,y) = \sum_{c\in R,G,B} \sum_{(i,j)\in\omega(x,y)} \left| d_c(i,j) - d_c^{'}(i,j) \right| \tag{7}$$

where $\omega(x,y)$ is a neighborhood of the pixel (x,y), $d_c(x,y)$ is the intensity ratio which minimizes the variation of intensity ΔI of the pixel (x,y): $d_c(x,y) = min\left(|\ln(\Delta I)|\right)$, and $d_c^{'}(x,y)$ is the same quantity evaluated on the background image.

The second belief is based on color consistency within pixels, defined as:

$$\Lambda(x,y) = |C_1(x,y) - C_1^{'}(x,y) + C_2(x,y) - C_2^{'}(x,y)| \tag{8}$$

where

$$\begin{cases} C_1(x,y) = & \arctan\left(\frac{I_{x,y}^{R}}{I_{x,y}^{B}}\right) \\ C_2(x,y) = & \arctan\left(\frac{I_{x,y}^{G}}{I_{x,y}^{B}}\right) \end{cases} \tag{9}$$

and $C_1^{'}$ and $C_2^{'}$ are the corresponding quantity for the background image.

The third belief that the pixel is a shadow is $\delta_t(x,y)$ which represents the difference of the distances of the pixel from the camera in the foreground and background images, computed from the stereo depth. Its meaning is the following: is the pixel (x,y) of the moving object a real object or is it a shadow? If it is a real object the difference should be high, otherwise should be low. Hence if $(1-\delta)$ is high the belief the pixel represents a shadow is also high.

The last belief, $\xi_t(x,y)$, is the quantity of reflected light represented by the pixel. Of course, mainly in outdoor environments, the more reflected light is detected the more likely the pixel represents a shadow. This can be computed as follows:

$$\xi_t(x,y) = |I_{x,y}^{R} - \mu_{x,y}^{R}| + |I_{x,y}^{G} - \mu_{x,y}^{G}| + |I_{x,y}^{B} - \mu_{x,y}^{B}| \tag{10}$$

where $I_{x,y}$ is the intensity of the pixel and $\mu_{x,y}$ its mean value, computed in an initial training phase.

C. Pixel clustering and tracking

Foreground pixels, segmented from the input image, are grouped together. To cluster the detected pixels, we use the method proposed in [15]. To track the pedestrians we use a slightly modified version of Kalman filter proposed in [4], which exploits the use of the stereo information.

IV. Image classification

In this section, we discuss the use of multi-class neural network for objects classification. In our specific case, human car and bicycle are the detected moving objects. The Regions of Interest (ROIs) may contain the mentioned objects, or be created by image noise. The regions of interest obtained from the previous steps are labelled based on the set *likelihood(q)*, where q is the region to classify.In our specific case ”Human”, ”Car” or ”Bicycle” are the detected objects. A forth class, ”Undefined”, is for objects which cannot be classified in the described categories. As multiple class are searched, a multiple class classifier, in our case neural network based, is well suited to our problem. It is necessary to define a proper input space. In general, considering the description of objects, both edges orientations and spatial information have high relevance. In this paper, as we consider humans and cars like objects,

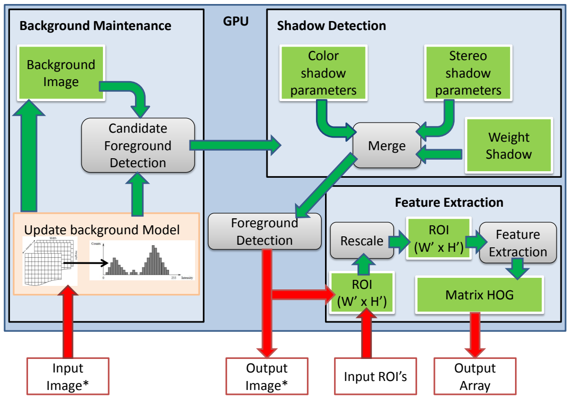


Fig. 1. GPU-based human and car detector.

A. HOG Features

The HOG feature has shown success in object detection [3] and they are accepted as one of the best features to capture gradient information. However, it is quite complex computationally. The collection of HOG for each image will compose the Matrix of HOG which is the observation vector O_t, at time t, used to classify the region. The Matrix of HOG is then uniformly divided in blocks with partial overlapping.

B. ROI classification

Each ROI estimated as described in Section III is then processed as a new single image. The image is represented with HOG features as previously described.

According to [10] the multiclass pattern recognition problem has been realized by K binary neural networks, trained separately, with a final decision module, which is used to select the final classification results based on the output of all the neural networks.

V. GPU Implementation

Since the algorithms are mostly pixel-wise, there are many processes that can be computed in parallel on a GPU. A single GPU board (NVIDIA GeForce 9800 GT with 512 MB) is used for general purpose computation and NVIDIA CUDA SDK is the software stack, C style. This card is able to run 512 threads at once.

We divide the GPU-based detector into three modules: background maintenance, shadow detection, and feature extraction. The background maintenance updates a model of the background to perform the difference between the background and the current images. The background maintenance downloads the input image into GPU and provides to estimate the difference between the background model and current image. The shadow detection collects the detection results and discards pixels labelled as shadow. Candidate pixels are then returned to CPU, and clustered to estimate current ROIs. The feature extraction module collects the detected ROIs and returns the feature array for the image, with one gradient histogram generated per fragment. Each ROI is scaled and it renders the feature array from the image, with one gradient histogram generated per fragment. The entire procedure is illustrated in Fig. 1.

For human and bicycle detection the best results have been achieved using 16x16 pixel blocks containing 2x2 cells of 8x8 pixels. The block stride is 8 pixels (one cell) so the histogram of a cell is normalized over 4 blocks. Each histogram has 9 bins. A big amount of memory is required because, for each pixel, inside the GPU, for each concurrent thread several data structures have to be stored, namely the three histograms Hc, M, FC and NFC. Each thread updates the model of a single pixel of the background. As the pixels are update by independent threads, this approach does not require inter-thread communication to synchronize the thread operations.

Shadow detection algorithm is made by different components. At instant t only the pixels detected as changed are analyzed, with the exception of the background. When the background image is updated, the background parameters and data are re-estimated.

The estimation of HOG features can be separated in three main phases: gradients estimation, block histograms, normalization. The estimation of the gradients, each thread computes gradients, magnitudes and orientations is computed in two steps. First step, we opted to assign a thread for each pixel in this phase and associate 64 threads for each blocks in order to compute the local values. To reduce the computation cost to estimate cells and blocks values, an integral image is calculated, which requires a temporary structure to memorize the intermediate results. To optimize the performances the number of threads associated depends on the number of orientations (in our case 9) and maximized the number of threads per blocks. The total number of operations are $log_2(rows) + log_2(cols)$ where in the case of rows the memory access are coalesced. In the histogram computation step a HOG pixel block is mapped to a CUDA thread block. The block has 4 cells and each cell 8 columns of 8 pixels. Moreover we associate a thread to each different orientation, so we will use 8x4x9=288 threads. Even if this model might not fully utilize the GPU hardware, it does have the advantage of scaling to different block/cell configurations. In our implementation each thread computes its own histogram and stores it in a shared memory. In the normalization step, each thread processes one pixel and the HOG pixel block structure is kept. During the process, each block is kept in memory and copy back to the global memory in the same grid once the process terminate.

VI. Experimental results

We have evaluated our system with the standard database PETS 20, PETS 2001 and PASCAL VOC2010. Moreover, a dataset of 2000 frames at 24 frames/s with a resolution of 320x240 pixels, obtained with the stereo camera Bumblebee2, where both humans and cars appear inside the video scenes, has been acquired. The performance of passive stereo camera are increasing in these days, and the main advantage offered by a stereo system is to obtain distance information from the objects. The stereo camera used in this experiment is reliable in a distance within 25 meters. In our tests the distances between camera and objects are among 15 meters. The scenarios contain both cars and humans in city areas.

A. Background modeling

Fig. 2 shows that the proposed algorithm gives better similarity results than the effisient histogram based versions, and provides the best results among the other considered algorithms. These results have been computed on one core of an Intel Core 2 Quad Q9550 CPU running at 2.83 GHz and will be used to evaluate the GPU speedup. In this figure, we indicate with HB, EHB, MoG and LBP the following algorithms: Histogram Based [6], Efficient Histogram Based [5], Mixture of Gaussians [13], Linear Binary Pattern [9], and the proposed algorithm.

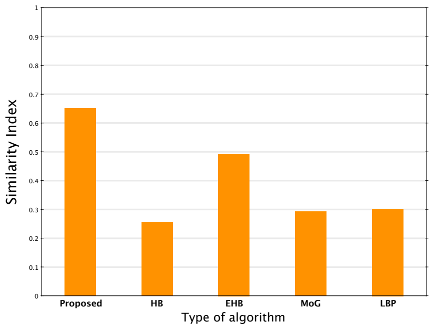


Fig. 2. Similarity measures computed on a single core of an Intel Core 2 Quad Q9550 CPU, on a set of 13000 images.

In Fig. 3, the Recall and Precision measures, obtained with the same set of images, are reported. It is worth noting that Precision and Recall are two widely used metrics for evaluating the correctness of a pattern recognition algorithm. They can be seen as extended versions of accuracy, a simple metric that computes the fraction of instances for which the correct result is returned. The proposed algorithm gives the highest values of both these measures.

B. Shadow detection

The background subtraction evaluation compares every ground-truth frame against the results of a specific background subtraction algorithm. Each comparison determines False Negatives (FN) and False Positives (FP). If a foreground moving object becomes stationary, we do not measure the performances for this region because of the ambiguity of

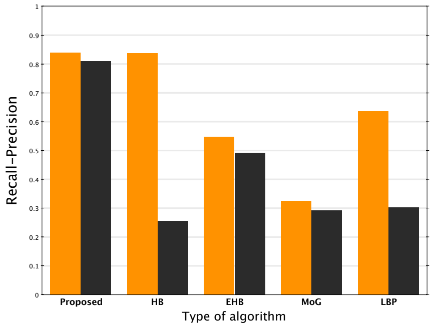


Fig. 3. Recall and Precision measures computed on the same conditions of Fig. 2. For each algorithm, Recall is represented by the left bar and Precision by the right bar.

Method	$\eta(\%)$	$\xi(\%)$
Proposed method without stereo	85.6%	83.6%
Proposed method	92.03%	92.83%

TABLE I

SHADOW DETECTION RATE OF THE PROPOSED METHOD. THE DETECTION RATE HAS BEEN COMPARED TO THE SAME METHODOLOGY WITHOUT USING THE STEREO INFORMATION.

the situation. The evaluation determines also the number of True Positives (TP) over all ground-truth frames.

For a quantitative evaluation, we calculate the accuracy of the cast shadow detection by using two metrics proposed in [11]. The *shadow detection rate* η measures the percentage of correctly labeled shadow pixels among all detected ones, while the *shadow discrimination rate* ξ measures the discriminative power between foregrounds and shadows.

$$\eta = \frac{TP_S}{TP_S + FN_S}, \quad \xi = \frac{\overline{TP_F}}{TP_F + FN_F} \tag{11}$$

where

TP_F: the foreground pixels correctly detected;

$\overline{TP_F}$: the ground-truth pixels which belongs to the foreground minus the shadow detected points which belongs to the foreground;

FN_F: the foreground pixels detected as shadow;

TP_S: the shadow pixels correctly detected;

FN_S: the shadow pixels detected as foreground.

In Tab. I we report the measures described in Eq. (11) computed with the proposed algorithm on the acquired dataset with and without the stereo information. We show that the use of depth raises both the measures by more than 7%. In Tab. II we report the performance of our shadow detection algorithm compared with other algorithms. The sequences are divided in Indoor, Outdoor No Shadow, Outdoor Low Intensity Shadow, and Outdoor Strong Intensity.

C. ROI classification

In Tab. III we report the classification results for human and cars (in terms of False Positive and False Negative) computed with the proposed algorithm and with the algorithms Mixture of Gaussians (MOG) and Saliency-Based (SAL) reported in [1]. All the data is obtained with the PETS01 dataset except the results reported in the last line which are obtained with our dataset.

Method	FP(%)	FN(%)
Proposed method (PETS)	.06%	8.2%
MOG 30 [1]	.48%	7.8%
MOG 100 [1]	.11%	20.2%
SAL 30 [1]	.11%	19.9%
SAL 100 [1]	.07%	27.1%
Proposed method (Stereo)	.03%	4.8%

TABLE III

CLASSIFICATION RESULTS OF HUMANS AND CARS FROM PETS01 DATASET OF THE PROPOSED METHOD COMPARED WITH MIXTURE OF GAUSSIANS (MOG) AND SALIENCY-BASED (SAL) ALGORITHMS [1].

Two sequences of the PETS dataset, compatible for camera and objects orientation, are used for training and testing the classifier. Due to different image resolutions and camera orientations, we use a different training set for the stereo camera. From the training set are randomly taken groups of images. Classification results are shown in Fig. 4. The left panel of the Fig. 4 shows that the recognition rate is over 90% using an adequate number of examples from the training. The right panel shows that classification of cars as a better performance than classification of humans, and it is optimal for low FP rate.

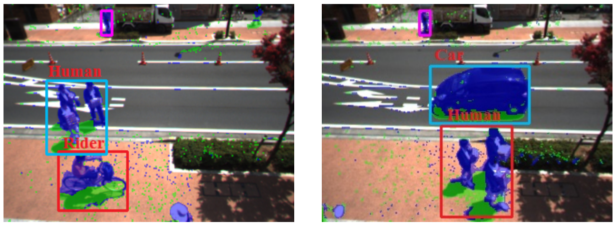
Finally, in Tab. IV we report the performances and GPU computing time of the classification of bicycles.

	True Pos.	False Pos.	Precision	Time [ms]
Proposed	78.5	3.2	96.1	58.3

TABLE IV

ACCURACY OF BICYCLE CLASSIFICATION.

In Fig. 5 we report finally two examples of ROI detection and classification of car, pedestrina and bicycle with the described system.



(a) Example: humans and riders are detected.

(b) Example: humans and cars are detected.

Fig. 5. Example of classification results. In this figure, the point coloured in blue are the ROI to be classified, the points in green are the pixels estimated as shadow.

VII. FINAL REMARKS AND CONCLUSION

In this paper, we described a fast human, car and bicycle detector suited for GPU implementation. The proposed approach uses the HOG features and multi-class neural network

	Sequence							
	$Indoor$		$OutdoorNS$		$OutdoorLIS$		$OutdoorSIS$	
$Method$	η	ξ	η	ξ	η	ξ	η	ξ
Moving Cast Shadow [17]	0.740	0.751	0.857	0.693	0.703	0.766	0.807	0.781
Proposed	0.818	0.886	0.905	0.896	0.787	0.849	0.884	0.897
Stereo [8]	/	/	0.916	0.721	0.856	0.704	0.707	0.642
Human Shadow Removal [2]	0.801	0.829	/	/	/	/	0.857	0.832

TABLE II

PERFORMANCE OF THE SHADOW DETECTING ALGORITHM IN DIFFERENT SEQUENCE AND COMPARED WITH OTHER CLASSICAL ALGORITHMS.

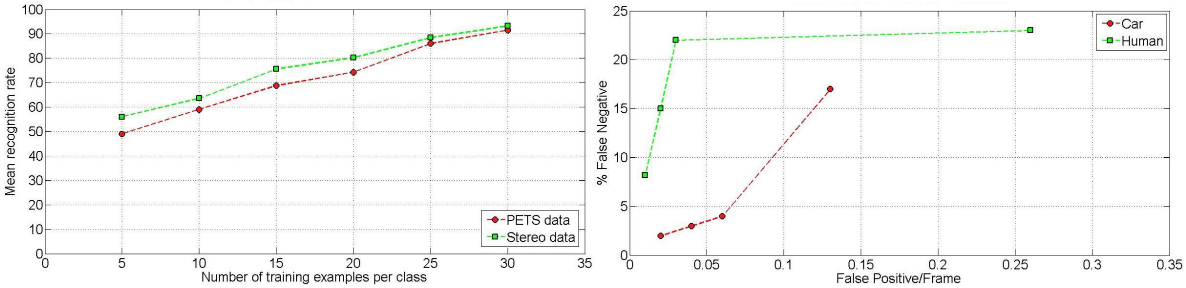


Fig. 4. Multiclass neural network classifier performances for cars and humans. Left panel: effect of the stereoinformation. Right curve: ROC on Pets data

classifier. It is flexible and allows to increase the number of classified objects. In Tab. V we report the computational time required to compute the proposed algorithm on a CPU and on a GPU. The last column shows the speedup of the GPU parts with respect to the corresponding CPU implementation. All the algorithms, if computed on a GPU, require about 40 ms and about 696 ms if implemented on a CPU. Thus, the GPU implementation of all the described system allow a real time operation at about 25 frames/s. The main contributions of this paper concern the pre-processing of stereo images for ROI detection based on background and shadow models that can be efficiently implemented in parallel on a GPU.

	CPU [ms]	GPU [ms]	Speedup
Grab Image	7.2	-	
Stereo Subtraction	20.0	0.8	25
Shadow detection	320.2	15.6	20.53
Background Maintenance	318.6	13.6	23.43
Segmentation/Labeling	5.1	0.3	17
HOG	21.4	2.2	9.73
NN Classification	3	0.06	50
Output image	0.3	-	
Total	695.8	40.06	17.37

TABLE V

AVERAGE COMPUTATIONAL TIME OF THE PROPOSED APPROACH FOR PROCESSING 320X240 IMAGES.

REFERENCES

[1] L. M. Brown, A. W. Senior, Y. li Tian, J. Connell, A. Hampapur, C. fe Shu, H. Merkl, and M. Lu. Performance evaluation of surveillance systems under varying conditions. In *In: Proceedings of IEEE PETS Workshop*, pages 1–8, 2005.

[2] C. Chen and J. Aggarwal. Human shadow removal with unknown light source. *ICPR2010*, pages 2407–2410, 2010.

[3] N. Dalal and B. Triggs. Histogram of oriented gradients for human detection. In *Int. Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 886–893, 2005.

[4] Y. Hoshikawa, et al. Human tracking using subtraction stereo and color information,. In *AWSVCI*, pages 5–8, 2009.

[5] C.-M. Kuo, W.-H. Chang, S.-B. Wang, and C.-S. Liu;. An efficient histogram-based method for background modeling. In *Innovative Computing, Information and Control (ICICIC), 2009 Fourth International Conference on*, pages 480 – 483, 2009.

[6] A.-N. Lai, H. Yoon, and G. Lee. Robust background extraction scheme using histogram-wise for real-time tracking in urban traffic video. In *Computer and Information Technology, 2008. CIT 2008. 8th IEEE International Conference on*, pages 845 – 850, 2008.

[7] K.-H. Lo and M.-T. Yang. Shadow detection by integrating multiple features. In *International Conference on Pattern Recognition (ICPR)*, volume 1, pages 743–746, 2006.

[8] C. Madsen, T. Moeslund, A. Pal, and S. Balasubramanian. Shadow detection in dynamic scenes using dense stereo information and an outdoor illumination model. *Proc. in DAGM*, pages 110–125, 2009.

[9] T. Ojala, M. Pietikainen, and D. Harwood. Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In *Proceedings of the 12th IAPR International Conference on Pattern Recognition (ICPR 1994)*, volume 1, pages 582–585, 1994.

[10] G. Ou, Y. Murphey, and L. Feldkamp. Multiclass pattern classification using neural network. In *ICPR*, 2004.

[11] A. Prati, I. Mikic, M. Trivedi, and R. Cucchiara. Detecting moving shadows: Algorithms and evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(7):918–923, 2003.

[12] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

[13] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2, pages 22 – 29, 1999.

[14] A. Torralba and A. Olivia. Depth estimation from image structure. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, volume 24, 2002.

[15] T. Ubukata, K. Terabayashi, A. Moro, and K. Umeda. Multi-object segmentation in a projection plane using subtraction stereo. In *International Conference on Pattern Recognition (ICPR)*, pages 3296–3299, 2010.

[16] K. Umeda and al. Subtraction stereo - a stereo camera system that focuses on moving regions. In *Proc. Of SPIE-IS&T Electronic Imaging, 7239 Three-Dimensional Imaging Metrology*, 2009.

[17] M. Yang, K. Lo, C. Chiang, and W. Tai. Moving cast shadow detection by exploiting multiple cues. *Image Processing.*, 2:95–104, 2008.