Robust Video-Based Surveillance by Integrating Target Detection with Tracking

Junxian Wang[1], George Bebis[1] and Ronald Miller[2]
[1] Computer Vision Laboratory, University of Nevada, Reno, NV
[2]Vehicle Design R&A Department, Ford Motor Company, Dearborn, MI
(junxian,bebis)@cse.unr.edu, rmille47@ford.com

***Abstract*— Target detection and tracking represent two fundamental steps in automatic video-based surveillance systems where the goal is to provide intelligent recognition capabilities by analyzing target behavior. This paper presents a framework for video-based surveillance where detection and tracking are addressed simultaneously in a unified framework (i.e., detection results trigger tracking, and tracking re-enforces detections)to improve detection results. In contrast to methods that apply target detection and tracking sequentially and independently from each other (i.e., *"detect-then-track"*), we feed the results of tracking back to the detection stage to adaptively optimize the threshold used in the detection stage and improve system robustness (i.e., *"detect-and-track"*). Specifically, the initial locations and representations of the targets are extracted by background subtraction. To model the background, we employ Support Vector Regression (SVR) along with an on-line learning scheme to update it efficiently over time. Target detection is performed by thresholding the outputs of the SVR model. Tracking uses shape projection histograms to iteratively localize the targets and achieve a high shape matching confidence level. Feeding back the results of tracking to the detection stage restricts the range of threshold values, suppress false alarms due to noise, and allows to continuously detect small targets as well as targets undergoing projection distortions. We have validated the proposed framework by detecting vehicles and pedestrians in traffic scenes using both visible and thermal video sequences. Experimental results and comparisons with frame-based detection and kernel-based tracking methods illustrate the robustness of our approach.**

I. Introduction

Target behavior analysis depends heavily on the reliability of target detection and tracking which can provide important information about the location of targets and their temporal correspondences over time. Both target detection and tracking have been investigated widely over the last two decades with the majority of approaches employing detection alone or frame-based detection, tracking alone, or hybrid schemes such *"detect-then-track"* schemes where detection and tracking work sequentially and independent from each other (i.e., detect the target in the first frame and turn it over to the tracker in the subsequent frames) [1].

In detection alone schemes, various detection algorithms have been proposed based on background subtraction, frame differencing, and optical flow. Methods based on background subtraction are common in video-based surveillance systems when cameras are fixed. In these systems, accurate and robust background modeling is a prerequisite step; however, due to significant intensity variations in images, it is difficult to parameterize the scene analytically. Recently, statistical learning approaches have been exploited to provide more accurate background models.

Table I presents an overview of background subtraction methods for target detection. For each method, we report the model used, the level of information extracted (i.e., pixel-based vs region-based), the type of information extracted (e.g., spatial vs temporal), decision rule, and updating scheme (i.e., for dealing with light changes). It should be noted that, many of the existing approaches utilize both spatial and temporal information to represent complex background scenes containing stationary and non-stationary information.

Tracking methods can be divided into two main categories. In the first category, the state sequence of a target is iteratively predicted and updated using prior information from past measurements and likelihood information from current measurements, respectively. Various filters have been used to predict the state sequence of a target including Kalman filters [3] and extended Kalman filters for linear predictions, and unscented Kalman filters [3] for non-linear predictions. The most general class of filters, however, is represented by particle filters [4][5], also called bootstrap filters [6], which are based on Monte Carlo integration methods. Methods belonging in the second category use various target characteristics, such as color or gray-level information, shape, and motion information and perform tracking by predicting changes in the appearance of the target from frame to frame [7].

In tracking alone methods, the initial locations of the targets is usually specified manually. On the other hand, the majority of methods employing detection along with tracking use a *detect-then-track* approach where the target is detected in the first frame and then turned over to the tracker in subsequent frames. The main problem with these methods is that they aim to resolve detection and tracking sequentially and independently from each other.

An important issue considered in this work is enhancing the results of target detection by feeding back to the detection stage temporal information from tracking. In this context, we propose a *detect-and-track* scheme where detection and tracking are addressed simultaneously in a unified framework (i.e., detection results trigger tracking, and tracking re-enforces detections). One approach to deal with this problem is by using a Bayesian decision framework which combines prior probability information provided by tracking with likelihood

TABLE I

OVERVIEW OF TARGET DETECTION APPROACHES

Statistical parametric						
Method	Model	Spectral	Spatial	Temporal	Decision	Updating scheme
W4[10]	Minima and maxima	Pixel-gray	——	Motion support map	Threshold	Parameters
Pfind[11]	Single Gaussian	Pixel-color	——	——	MAP	Parameters of Gaussian
MoG[12]	Mixture of Gaussian	Pixel-gray	——	——	Threshold	Parameters of Gaussian
Non-statistical parametric						
Non-parametric[13]	Probability density of pixel density	Pixel	Neighboring pixels	——	Threshold	Probability density function
Olvier[14]	Eigen background	Region-color	——	——	Threshold	Threshold
Monnet[15]	PCA of region	Region-color	——	Auto-regressive model	Threshold	Basis vectors of eigenspace
Wallflower [16]	Self-regression model	P-R-F	4-connected regions	——	Threshold	Prediction coefficients
Liyuan [18]	Principal feature of pixels	Pixel-color	Gradient	Color co-occurrence	Bayesian decision	Linear model

information provided by frame-based detection [8]. However, the performance of target detection depends heavily on the threshold used to distinguish between foreground and background. Another approach is propagating the probabilities of detection parameters (e.g. at several scales and poses) over time using a condensation filter and factored sampling [9].

In this paper, we propose a framework for integrating target detection with tracking to improve detection results. In this framework, we employ SVR [20] to model the background and an on-line learning scheme [21] to update it efficiently over time. The initial locations and representations of the targets are extracted by thresholding the outputs of the SVR model where the threshold is adaptively optimized using feedback from tracking. Tracking uses shape projection histograms to iteratively localize the targets and achieve a high shape matching confidence level between successive frames. Using feedback from tracking restricts the range of threshold values during detection, suppresses false alarms due to noise, and allows to continuously detect small targets and targets undergoing projection distortions. Besides improving detection, integrating detection with tracking can help to initialize tracking automatically. We have validated the proposed framework by detecting vehicles and pedestrians in traffic scenes using both visible and thermal video sequences.

II. TARGET LOCATION INITIALIZATION

In order to effectively detect the precise location of targets in a scene but also not to miss small targets, an accurate background model is required. Moreover, many practical applications require an effective way to incorporate background changes by updating the background model fast and effectively. In this study, we propose using SVR to build a model of the background. SVR is a statistical learning technique for estimating a function from a set of training data [20]. To update the background model, we use an on-line SVR learning algorithm [21].

A. Background modeling using SVR

Given a set of training data, SVR fits a function by specifying an upper bound on a fraction of training data allowed to lie outside of a distance ε from the regression estimate. This type of SVR is usually referred to as ε-insensitive SVR [20]. For each pixel belonging to the background, we employ a separate SVR model to model it as a function of the intensity. To classify a given pixel as background or not, we feed its intensity value to the SVR model associated with that pixel and we threshold the output of the SVR.

Specifically, let us assume a set of training data for some pixel p obtained from a number of frames, $\{(x_1, y_1), ..., (x_l, y_l)\}$, where x_i corresponds to the intensity value of pixel p at frame i, and y_i corresponds to the confidence of pixel p belonging to the background. Once the SVR has been trained, the confidence of pixel p in a new frame k, $f(x_k)$, is computed as follows:

$$f(x_k) = \sum_{j=1}^{l} (a_j - a_j^*) k(x_k, x_j) + b \tag{1}$$

where $k(x_i, x_j)$ is a kernel function and a, a^* are Lagrange multipliers. In this work, we used a Gaussian kernel. The solution of the ε-insensitive SVR corresponds to finding values for the Lagrange multipliers a, a^* minimizing the following quadratic objective function:

$$\begin{aligned} W &= \frac{1}{2} \sum_{ij} (a_i - a_i^*) k(x_i, x_j) (a_j - a_j^*) - \sum_i y_i (a_i - a_i^*) \\ &\quad + \varepsilon \sum_i (a_i + a_i^*) \end{aligned} \tag{2}$$

where $0 \leq a_i, a_i^* \leq C$ and $\sum_i (a_i - a_i^*) = 0$

To illustrate the SVR-based background modeling approach, we use a video sequence captured at a traffic intersection assuming a fixed camera (see Fig. 1(a)). To collect the training data for the SVR, first we built B "clean" background images

(i.e., without containing moving vehicles or pedestrians). This was done by taking F successive frames and finding the median intensity value at each pixel location (see Fig. 1(b)). Here, we used a total of 90 frames to build B=30 clean background images using F=30 frames each time. It should be noted that, although all the images were captured using a fixed camera, there were still fluctuations in the intensity values in the "clean" background images due to light changes caused by outdoor environmental conditions. To train the SVR model assigned to a particular pixel location, we used the intensity values at this location from all clean images (i.e., x_i) and assigned a high confidence value to this pixel (i.e., y_i=1).

Fig. 1. Captured traffic scenes and the computed "clean" background scene using median filtering.

Fig. 2 shows the results of background modeling at a fixed pixel location using Mixtures of Gaussians (MoG) and SVR. In each graph, the x-axis corresponds to the intensity of the pixel over the 30 "clean" background images (i.e., shown as red circles), while the y-axis corresponds to confidence of that pixel belonging to the background (i.e., set to 1). Fig. 2 (a) shows the SVR-based model while Figs. 2 (b) and (c) show the MoG-based models using four and two Gaussians correspondingly. As it can be observed, SVR can find a better solution than MoG. Fig. 2 (d) shows an SVR-based solution corresponding to a different pixel location.

B. Extracting initial target locations

Given the SVR-based background model, the intensity of each pixel in a new frame forms an input to the SVR. The output of the SVR is represents the confidence that the given pixel belongs to the background. Eventually, a pixel is labelled as background if its confidence is higher than a threshold S. Specifically, a binary foreground detection map $M_{x_i}^t$ is formed at frame t as follows:

$$M_{x_i}^t = \begin{cases} 1, & \text{foreground}, \quad f(x_i) > S \\ 0, & \text{background}, \quad \text{otherwise.} \end{cases} \tag{3}$$

where $f(x_i)$ is the SVR output and S is an initial threshold. For each region in the binary map, we fit an ellipse. The initial location of each target is represented by the center, long axis, and short axis of the ellipse as shown in Fig. 3.

C. On-line SVR learning

To update the background model over time, we need an efficient method that avoids expensive re-training. Here, we

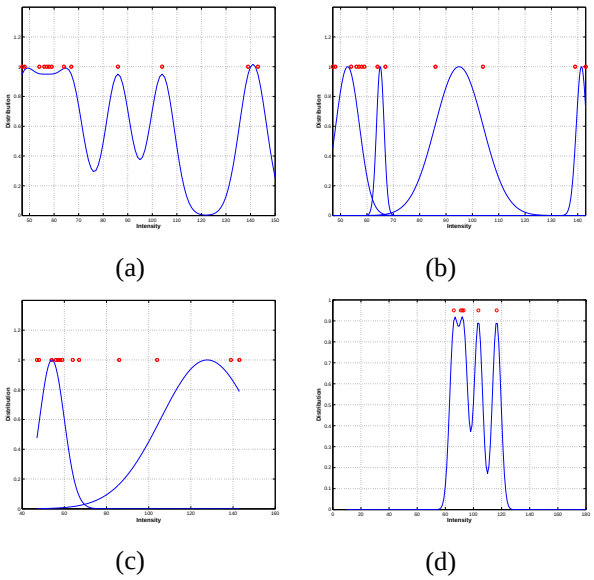

Fig. 2. Different solutions for background modeling. (a) SVR solution,(b) MoG solution using 4 Gaussians. (c) MoG solution using 2 Gaussians (d) SVR solution at a different pixel location.

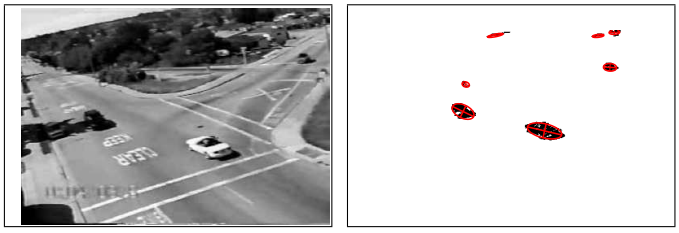

Fig. 3. Initial locations of targets represented by the best-ellipse fitting.

use an efficient on-line SVR learning algorithm which updates the SVR function whenever new training data becomes available [21]. The main idea is changing the Lagrange multipliers a, a^* in a finite number of steps until the Karush-Kuhn-Tucker (KKT) conditions are satisfied [21].

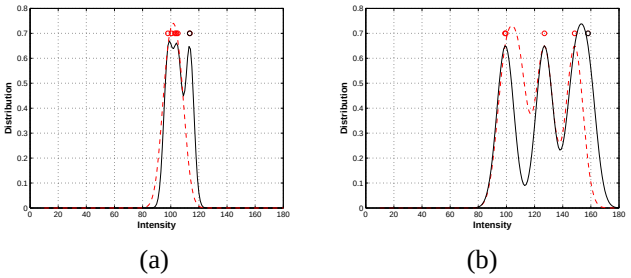

Fig. 4. Illustration of on-line SVR learning.

Figs. 4 (a) and (b) illustrate the on-line SVR learning procedure where the training data is shown by red circles. After training, the regression function in Fig. 4 (a) is estimated by a single peak (i.e., red dashed line). When new data comes along, shown as black circles, the regression function is

updated using on-line learning. In this example, the regression function becomes bimodal (i.e., black line). Figure 4 (b) shows another case where the regression function contains multi-peaks. In this case, the number of peaks before (i.e., red dashed line) and after (i.e., black line) the addition of new examples do not change, however, they do shift to the right.

III. Integrating Target Detection with Tracking

In this section, we describe the framework for integrating target detection with tracking in order to improve detection results. First, we discuss our target representation scheme which is based on normalized shape projection histograms. Then, we describe the algorithm used to predict the location of targets in subsequent frames. Finally, we present the feedback mechanism for optimizing the threshold used during detection.

A. Target representation

Our target representation scheme is based on shape information rather than on color or texture. The main reason is that shape is more robust to light changes in outdoor environments. In order to make our target representation scheme robust to perspective projection, scale, and rotation transformations, we employ normalized shape projection histograms.

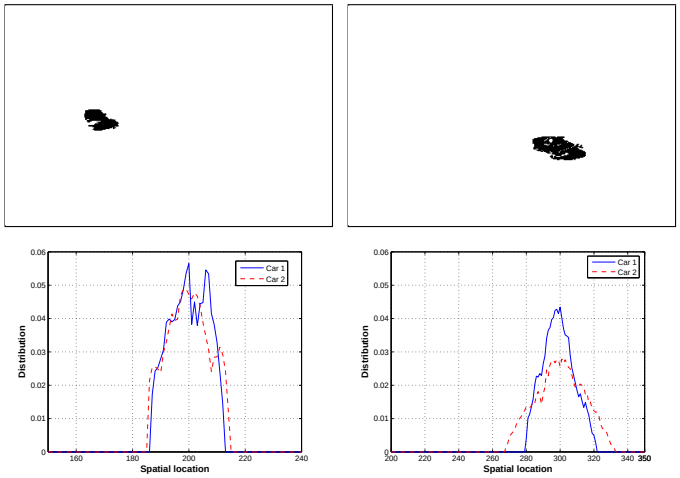

Fig. 5. Left: vertical shape projection histograms; Right: horizontal shape projection histograms.

1) Normalized shape projection histograms: We denote the location of a target by (x_i, y_i) which corresponds to the location of the best-fitting ellipse. To compute the projection histograms, we project the target horizontally and vertically by counting the number of pixels in each row and in each column correspondingly. To make the projection histograms invariant to target orientation, first we transform the target to a default coordinate system. This is done in two steps. First, we find the best-fitting ellipse of the the target. Second, we align its major and minor axes with the x- and y-axis of the default coordinate system. The main assumption here is that the targets are approximately 2-D; this is a valid assumption in our application since the depth of the targets is much smaller compared to their distance from the camera.

Since projection histograms are sensitive to the location and size of the targets, we normalize them by shifting the middle bins of the histogram to the geometric center of the target and resizing the number of bins to a fixed size. Specifically, the normalized horizontal (i.e., $\bar{H}_x$) and vertical (i.e., $\bar{H}_y$) shape projection histograms are defined as follows:

$$\begin{aligned} \bar{H}_x(m) &= \{(x_i, y) | (x_i, y) \in R\}, \\ \bar{H}_y(n) &= \{(x, y_j) | (x, y_j) \in R\}, \\ m &= x_i - x + M/2, \\ n &= y_j - y + N/2. \end{aligned} \tag{4}$$

where (x, y) is the geometric center of the target (i.e., the center of the best-fitting ellipse), m and n are indices, and M and N are the number of bins in the horizontal and vertical projection histograms.

2) Weighted shape projection histograms: In order to reduce the effects of background noise and image outliers, we introduce weights to improve the robustness of the normalized shape projection histograms. This is done by employing an isotropic kernel function $k(\cdot)$ in a similar way as in [7]. The role of the kernel function is to assign smaller weights to pixels farther away from the center bin of the project histogram. Then, the weighted target model histograms, denoted as H_x^T and H_y^T, are defined as follows:

$$H_x^T(m) = \frac{\bar{H}_x(m) + k(\cdot)}{\sum_{m=1}^{M} \bar{H}_x(m) + k(\cdot)} \qquad H_y^T(n) = \frac{\bar{H}_y(n) + k(\cdot)}{\sum_{n=1}^{N} \bar{H}_y(n) + k(\cdot)} \tag{5}$$

where $k(x_i, y_j) = c - [(x_i - x)^2 + (y_j - y)^2]$, and $c = (w/2+1)^2 + (h/2+1)^2$ (i.e., computed from the size $w \times h$ of the target).

To find the targets in subsequent frames, we search a window of size $W \times H$. Then, candidate targets are identified in this window by thresholding the outputs of the SVR models. The weighted target candidate projection histograms, denoted as H_x^C and H_y^C, are defined as follows:

$$H_x^C(m) = \frac{\bar{H}_x(m) + g(\cdot)}{\sum_{m=1}^{M} \bar{H}_x(m) + g(\cdot)} \qquad H_y^C(n) = \frac{\bar{H}_y(n) + g(\cdot)}{\sum_{n=1}^{N} \bar{H}_y(n) + g(\cdot)} \tag{6}$$

where $g(x_i, y_j) = c - \{[(x_i - x)/h]^2 + [(y_j - y)/h]^2\}$, and c is calculated from the size $h = W \times H$ of the search window. Fig. 5 shows an example of shape projection histograms.

B. Predicting target location

To find the location of a target in subsequent frames, we need to define a similarity measure between the target model, computed in previous frames, and the target candidates, detected in the current frame. Here, we use a similarity measure based on the Manhattan distance between the weighted shape projection histograms of the target model and the candidates:

$$
\begin{aligned}
D_x &= \sum_{m=1}^{M} [H_x^C(m) - H_x^T(m)] \\
D_y &= \sum_{n=1}^{N} [H_y^C(n) - H_y^T(n)]
\end{aligned} \tag{7}
$$

To accurately localize a target in a search window, we need to minimize the objective function defined in Eq. (9). Below, we show the derivation of the objective function for the case of horizontal shape projection histograms; similar derivations apply for the case of vertical shape projection histograms:

$$
\begin{aligned}
\Phi &= \min_k \sum_{m=1}^{M} [H_{x^k}^{C_k^S}(m) - H_x^T(m)] \\
&= \sum_k w_k \sum_{m=1}^{M} [H_{x^k}^{C_k^S}(m) - H_x^T(m)] \\
&= \sum_k w_k \sum_{x_i \in R} [H_{x^k}^{C_k^S}(x_i - x + M/2) \\
&\quad - H_x^T(x_i - x + M/2)] \\
&\quad \longrightarrow \min \text{over } S \text{ and } x^k
\end{aligned} \tag{8}
$$

where S is the threshold used to find the target candidates and w_k restricts the spatial position x^k of the target candidate around the geometric center x of target model. $H_{x^k}^{C_k^S}(m)$ is the weighted shape projection histogram of the k-th target candidate detected using threshold S.

To perform the above minimization, we employ a multi-scale iterative scheme by gradually decreasing the value of the threshold S used for target detection and changing the spatial center position of the search window:

$$
\begin{aligned}
\Phi(l) &= \sum_k w_k \sum_{m=1}^{M} [H_{x^k(l)}^{C_k^{S(l)}}(m) - H_x^T(m)] \\
&= \sum_k w_k \sum_{x_i \in R(l)} [H_{x^k(l)}^{C_k^{S(l)}}(x_i - x^k(l) + M/2) \\
&\quad - H_x^T(x_i - x + M/2)]
\end{aligned} \tag{9}
$$

C. Confidence coefficient

A key issue in implementing the above idea is how to choose appropriate functions for decreasing S and changing the geometric center (x, y) of the candidate targets at each iteration l. For this, we use the ratio between the weighted shape projection histogram of the target model and the candidates. We refer to this ratio as the *confidence coefficient* of shape matching and it is defined below:

$$
\xi_x(l) = \sum_{x_i \in R(l)} \sqrt{\frac{H_{x^k(l)}^{S(l)}[x_i - x^k(l) + M/2]}{H_x^C[x^k(l) - x + M/2]}} \tag{10}
$$

$$
\xi_y(l) = \sum_{y_i \in R(l)} \sqrt{\frac{H_{y^k(l)}^{S(l)}[y_i - y^k(l) + N/2]}{H_y^C[y_i - y + N/2]}} \tag{11}
$$

The confidence coefficient provides the weights in the iterative procedure used to change the spatial location of the targets and select the threshold range. Using the confidence coefficient, the new center of the search window is updated as follows:

$$
\begin{aligned}
x^k(l) &= x^k(l-1) \times \xi_x(l-1) \\
y^k(l) &= y^k(l-1) \times \xi_y(l-1);
\end{aligned} \tag{12}
$$

D. Adaptive threshold optimization

The confidence coefficient is also used to update the threshold S used in the target detection stage. Specifically, let us denote the threshold at the $l-1$ iteration as $S(l-1)$, then the threshold at the l iteration $S(l)$ is defined as follows:

$$
S(l) = S(l-1) - \left[1 - \sqrt{\xi_x^2(l-1) + \xi_y^2(l-1)}\right]. \tag{13}
$$

The above iterative procedure decreases D_x and D_y while moving the spatial center of the search window iteratively closer to the geometric center of the target. The iterative procedure terminates when the distance between the weighted shape projection histogram of target model and the target candidates is small than a given value. However, when the confidence coefficient is too low, we increase the threshold of target detection to avoid under-segmentation which could cause differences in the shape of the targets in successive frames.

IV. Experimental results

The proposed framework has been evaluated by detecting vehicles and pedestrians using both visible and thermal video sequences. The visible video sequence was captured at a traffic intersection and contains a total of two hours video with a sampling rate 4 frames/second. The thermal video data was captured at a university campus walkway intersection over several days (morning and afternoon) using a Raytheon 300D thermal sensor core with 75mm lens mounted on an 8-story building [23].

A. Results using detection alone

First, we demonstrate the performance of a system employing frame-based detection and SVR-based background modeling, without feedback from tracking for threshold optimization. Fig. 6 shows the locations of the targets found using this approach.

Fig. 7 presents comparison results between SVR-based background modeling and AdaBoost [23]. Our first observation is that SVR-based detection produces more accurate detections (i.e., the window enclosing the targets is much narrower). Moreover, on the left part of Fig. 7, we can observe a pedestrian who was detected as two separate entities by

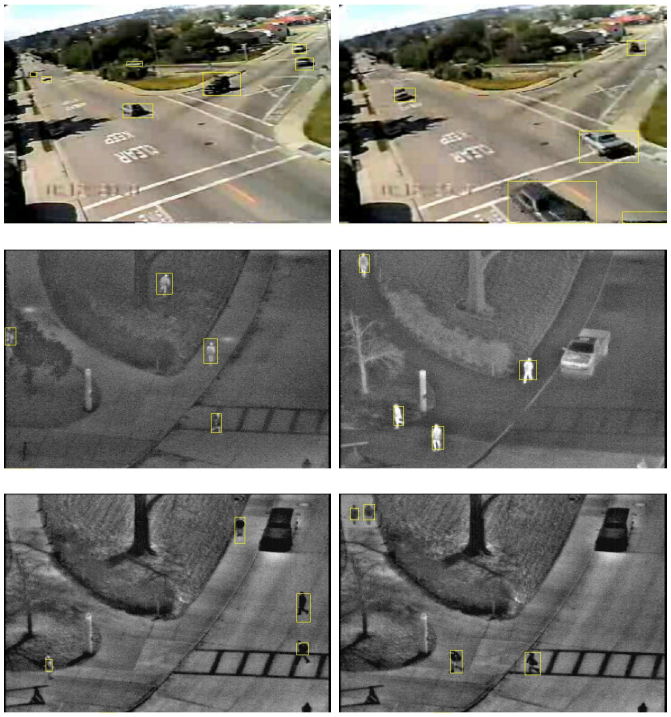

Fig. 6. Detection results using SVR-based background modeling, without feedback from tracking.

Adaboost. On the right part of Fig. 7, however, the same pedestrian was detected as a single entity using SVR. Nevertheless, the performance of detection without employing some kind of feedback from tracking depends heavily on the choice of the threshold. If the threshold is not chosen properly, we might end up with many false alarms as shown in Fig. 3.

Fig. 7. Comparison results between AdaBoost [23] (left) and SVR (right).

B. Results using integration

Figs. 8 and 9 present comparison results between frame-based detection without feedback from tracking and the proposed method which integrates detection with tracking. Each target is tracked and labelled with rectangles having different colors. The 1^{st} and 2^{nd} rows of Fig. 8, show tracking results and detection maps using the proposed method. The last row presents detection results using frame-difference and no threshold optimization. Among the results shown, it is interesting to note that the small target, labelled by a green

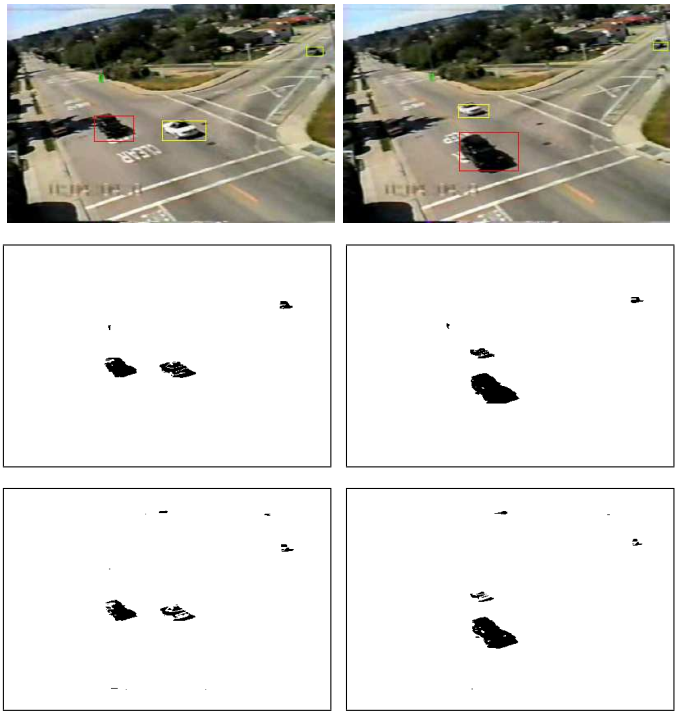

Fig. 8. Comparison results between the proposed method and frame-based detection using visible video. 1^{st} and 2^{nd} rows: Tracking results and corresponding detections using the proposed method. 3^{rd} row: Detection results using frame-based detection.

rectangle in the 1^{st} row of Fig. 8, is very difficult detect using frame-based detection and non-optimized thresholds as shown in the 3^{rd} row of Fig. 8. On the other hand, the proposed method shows more accurate detection results by optimizing the threshold. Moreover, the proposed method shows has suppressed the false alarms that appear in frame-based detection as shown in the 2^{nd} row of Fig. 9. Table II shows quantitative comparisons in terms of true positives and false alarms for frame-based detection and the proposed approach. Obviously, the proposed approach has lower false alarm and higher true positive rates than frame-based detection.

TABLE II
QUANTITATIVE COMPARISONS IN TERMS OF TRUE POSITIVES (TP), FALSE ALARMS (FA), AND GROUND TRUTH (GT)

Data sets	Methods	Ground truth	True Positive	False Alarm
Visible video	Frame-based detection	346	296	30
	Integrating detection with tracking	346	340	5
Thermal video	Frame-based detection	371	371	35
	Integrating detection with tracking	371	371	0

Figs. 10 (a) and (b) present another quantitative comparison by counting the number of pixels in two different segmented regions between frame-based detection and the proposed

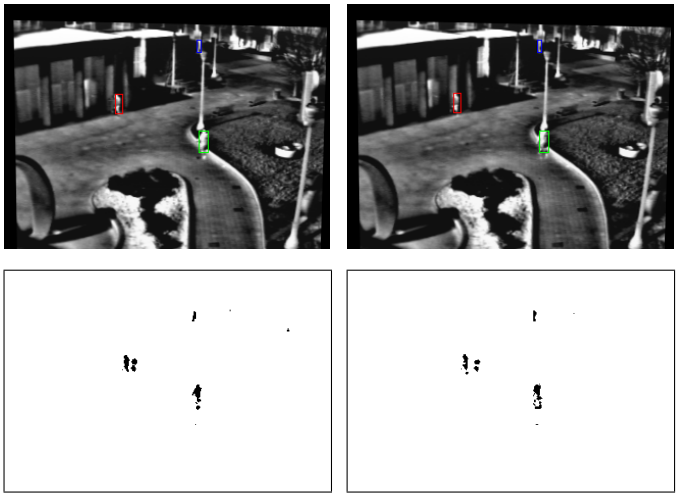

Fig. 9. Comparison results between the proposed method (1^{st} row) and frame-based detection (2^{nd}-row) using thermal video. Frame-based detection yields many false positives.

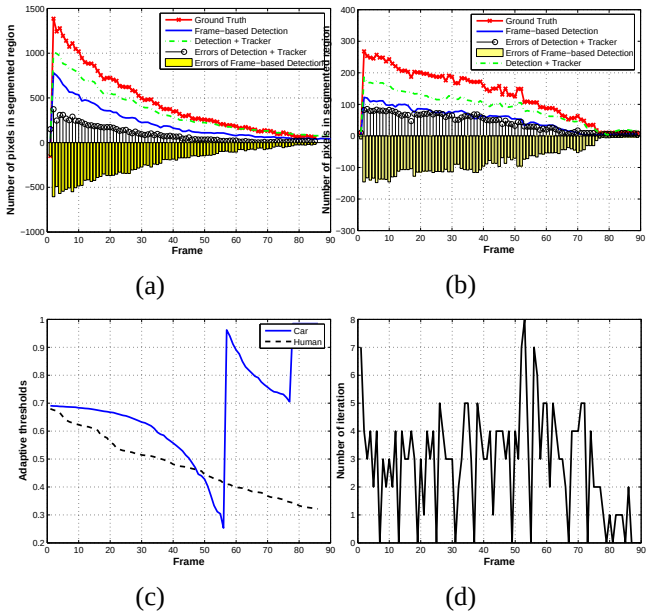

Fig. 10. (a),(b): Comparison results between frame-based detection and the proposed approach by counting the number of pixels in two different segmented regions. The red curve indicates ground truth information while the green and blue curves indicate the performance of the proposed method and frame-based detection, respectively; (c): Adaptive threshold values over time for two different targets; (d): Average number of iterations.

method. The red curve indicates ground truth information (i.e., the true number of pixels in the segmented region). The reason that the number of pixels decrease over time is because the target becomes smaller and smaller over time due to moving away from the camera. The green and blue curves show the performance of the proposed method and frame-based detection respectively. In Figs. 10 (a) and (b), the green curves are closer to the red curves, indicating that the proposed method makes less errors compared to frame-based detection.

Fig. 10 (c) shows the adaptive threshold values over time for two targets with different motion characteristics (i.e., a car and a pedestrian). As it can be observed, the thresholds were iteratively decreased based on the confidence coefficient computed from the shape projection histogram matching process. To avoid under-segmentation, the threshold was re-set to a higher value when the confidence coefficient fell below a certain value. Fig. 10 (d) demonstrates the average number of iterations at each frame. As it can be observed, the time complexity of this step is not high.

C. Comparison with kernel-based tracking

In this section, we present comparison results between kernel-based tracking [7] and the proposed approach.

Fig. 11 shows tracking results for frames 4, 12, 22, 26, 39 of a test sequence using the proposed method (1^{st}-row) and kernel-based tracking (2^{nd}-row). The 3^{rd} row of Fig. 11 shows the detected targets using the proposed method. In order to make the comparison fair, kernel-based tracking was initialized using the initial target locations found by our approach, shown in the first column of Fig 11. As it can be observed, kernel-based tracking has difficulties with tracking small targets (e.g., a small human walking along the road) and targets with perspective projection distortions. On the other hand, the proposed approach can handle these cases due to integrating tracking with the detection and the use of adaptive thresholding in the detection stage.

V. Conclusions

We have proposed a framework for improving video-based surveillance by integrating target detection with tracking. The proposed framework was evaluated by detecting and tracking pedestrians and vehicles in both visible and thermal video sequences. On-line SVR was used to model the background and to accurately detect the initial locations of the targets. Moreover, weighted shape projection histograms were exploited to predict the location of targets in successive frames. At the same time, a confidence coefficient for shape matching was computed to suppress false alarms. Using weights derived from the confidence coefficient of shape matching, we were able to optimize the threshold used in the target detection stage. Our experimental results show good performance, especially when dealing with small targets and targets undergoing perspective projection distortions. Moreover, they show good suppression of false alarms due to noise. For future work, we plan to improve the speed of the method. Although we were able to achieve good speed in our experiments by sub-sampling the captured images, further improvements are necessary for true real-time performance. One way to improve speed is by using region-based instead of pixel-based SVR models to represent the background.

Acknowledgments

This research was supported by Ford Motor Company under grant No. 2001332R, and the University of Nevada, Reno (UNR) under an Applied Research Initiative (ARI) grant.

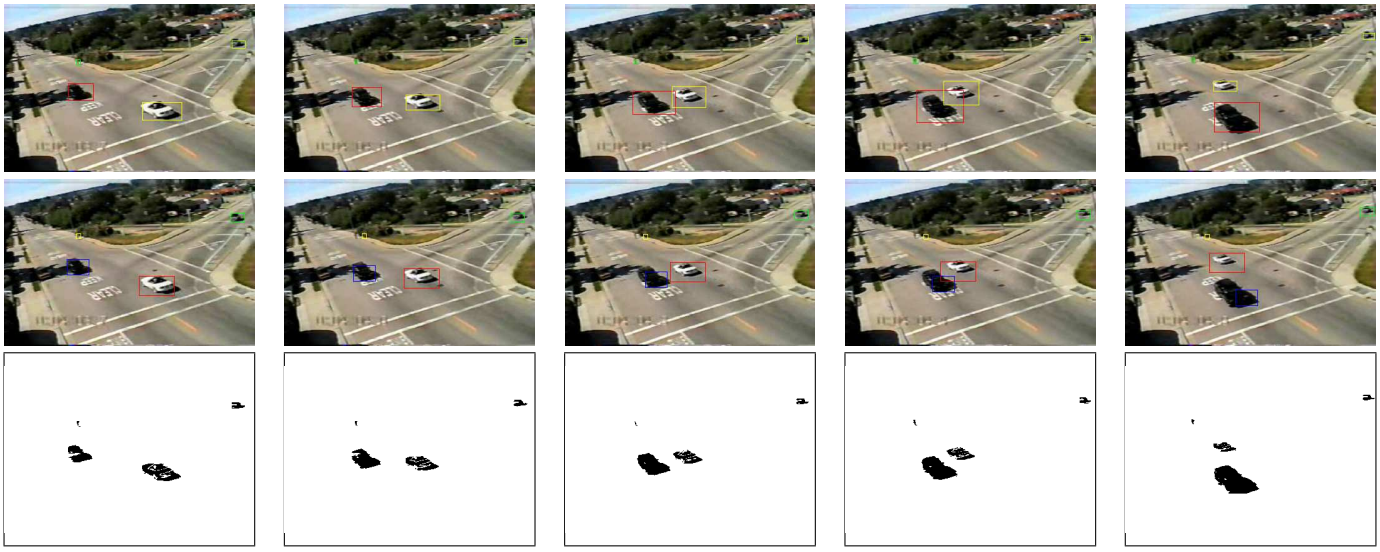

Fig. 11. Comparison results between kernel-based tracking and the proposed approach when tracking small targets and targets with projection distortion. Tracking results are shown in frames 4, 12, 22, 26, 39 of the test sequence using rectangles of different colors. 1^{st} row: Tracking results using the proposed method. 2^{nd} row: Tracking results using kernel-based tracking where, the initial target locations were chosen to be the same to those found by our approach (i.e., first row); kernel-based tracking has difficulties with tracking small targets (e.g., small human walking along the road) and targets with perspective projection distortions. 3^{rd} row: Detection results using the proposed method.

REFERENCES

[1] Z. Sun and G. Bebis and R. Miller, "On-road Vehicle Detection: A Review," *IEEE Trans. Pattern Anaysis and Machine Intelligence*, vol. 28. no. 5 pp. 694-711, May, 2006.

[2] S. Avidan, "Support vector tracking ," *IEEE Trans. Pattern Anaysis and Machine Intelligence*, vol. 26. no.8 pp. 1064-1072, 2004.

[3] Y. Bar-shalom and T. Fortmann,*Tracking and Data Association.* Academic Press. 1988.

[4] G. Kitagawa,"Non-Gaussian state-space modeling of nonstationary time series ," *Journal Am. Statistical Assoc.*, vol. 82, pp. 1032-1063, 1987.

[5] N. Gordon, S. Maskell and T. Kirubarajan, "Efficient particle filters for joint tracking and classification," *Proc. SPIE Conf. on Signal and Data Processing of Small Targets*, April. 2002.

[6] N. Gordon, D. Salmond and A. Smith,"A novel approach to nonlinear and non-Gaussian Bayesian state estimation," *IEE Proc. -F*, vol. 140, pp. 107-113, 1993.

[7] D. Comaniciu, V. Ramesh and P. Meer, "Kernel-based object tracking," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, May 2005.

[8] J Wang, H. Eng, A. Kam, W. Yau, "A framework for foreground detection in complex environments," *European Conference on Computer Vision, Workshop on Statistical Modeling for Video Processing*,pp.129-140,2004.

[9] R. Verma, C. Schmid and K. Mikolajczyk , "Face detection and trackign in a video by propagating detection probabilities," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1215-1227, 2003.

[10] I. Haritaoglu, D. Harwood and L. Davis, "W^4: Real-time surveillance of people and their activities," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 809-830, 2000.

[11] W. Wren, A. Azarbaygaui, T. Darrell and A. Pentland,"Pfinder: real-time tracking of the human body," *IEEE Trans. Pattern Analysis and Machine Intelligence* , vol. 19, no. 7, pp. 780-785, 1997.

[12] C.Stauffer and W. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 747-757, 2000.

[13] A. Elgammal, D. Harwood and L. Davis,"Non-parametric model for background adaptation," *European Conference on Computer Vision*, 2000.

[14] O. Javed, K. Shafique and M. Shah,"A hierarchical apporach to robust background subtraction using color and gradient," *IEEE Workshop on Motion and Video Computing*, pp. 22-27, 2002.

[15] A. Monnet, A. Mittal, N. Paragios and V. Ramesh,"Background modeling and subtraction of dynamic scenes ," *IEEE Internallational Conference on Computer Vision*, 2003.

[16] K. Toyama, J. Krumm, B. Brumitt and B. Meyers,"Wallflower: principles and practice of background maintenance," *International Conference on Computer Vision*, pp. 255-261, 1999.

[17] M. Seki, T. Wada, H. Fujiwara and K. Sumi, "Background Subtraction based on Co-occurrence of Image Variations," *International Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 65, 2003.

[18] L. Li and M. Leung,"Integrating intensity and texture differences for robust change detection," *IEEE Trans. Image Processing*, vol. 11, no. 2, pp. 105-112, 2002.

[19] M. Hearst, "Trends and controversies - support vector machines,"*IEEE Intelligent Systems*, vol.13. No.4 pp. 18-28, 1998.

[20] A. Smola and B. Scholkopf, "A tutorial on support vector regression," *NeuroCOLTS technical report Series NC2-TR-1998-030*, Oct. 1998.

[21] J. Ma and J. Theiler, "Accurate on-line support vector regression," *Neural Computation*, vol. 15, pp. 2683-2703, 2003.

[22] J. Davis and V. Sharma, "Robust background-subtraction for person detection in thermal imagery," *IEEE Int. Conf. on Computer Vision and Pattern Recognition, Workshop on Object Tracking Classification Beyond Visible Spectrum* , 2004.

[23] J. Davis and M. Keck, "A two-stage template approach to person detection in thermal imagery," *IEEE Int. Conf. on Computer Vision and Pattern Recognition, Workshop on Object Tracking Classification Beyond Visible Spectrum* , 2005.