and, for most cases, approximately 50–150 ms were spent for the corner detection and matching and 20–500 ms (but mostly under 150 ms) were spent for the optimization.

V. Conclusion and Future Work

We presented a moving-object detection algorithm for railroad-crossing safety. We proposed a fast triangulation method and a robust optimization algorithm to meet our application's need. We presented preliminary experiments that showed that the proposed approach is promising. A more thorough evaluation will be performed when a large volume of video data is collected. We will generate a learning data set by showing video clips to human and compare the detection result with our algorithm.

Our algorithm works at pseudoreal time (about three to four frames/s on a Pentium III personal computer). We expect that, in the near future (with the development of computer hardware), it can be applied to real-time applications such as an in-vehicle collision warning system.

Appendix
Proof of (9)

We define the object function

$$\mathrm{Err}(x', y', x, y, \hat{Z}) = \{(\hat{x'}(x, y, \hat{Z}) - x')^2 + (\hat{y'}(x, y, \hat{Z}) - y')^2\}$$

to be minimized with regard to $\hat{Z}$. From (3) and (4)

$$\hat{x'}(x, y, \hat{Z}) = \frac{\hat{X}'(x, y, \hat{Z})}{\hat{Z}'(x, y, \hat{Z})} = \frac{\hat{Z} r_x + \hat{T}_X}{\hat{Z} r_z + \hat{T}_Z}$$

and

$$\hat{y'}(x, y, \hat{Z}) = \frac{\hat{Y}'(x, y, \hat{Z})}{\hat{Z}'(x, y, \hat{Z})} = \frac{\hat{Z} r_y + \hat{T}_Y}{\hat{Z} r_z + \hat{T}_Z}$$

where $r_x = \hat{R}_{1,1}x + \hat{R}_{1,2}y + \hat{R}_{1,3}$, $r_y = \hat{R}_{2,1}x + \hat{R}_{2,2}y + \hat{R}_{2,3}$, and $r_z = \hat{R}_{3,1}x + \hat{R}_{3,2}y + \hat{R}_{3,3}$.

We get $\hat{Z}$ that minimizes $\mathrm{Err}()$ by taking its partial derivative with regard to $\hat{Z}$

$$\frac{\partial \mathrm{Err}}{\partial \hat{Z}} = \frac{-2}{(\hat{Z} r_z + \hat{T}_Z)^3}[(r_x \hat{T}_Z - \hat{T}_X)\{(x' - r_x)\hat{Z} + (x'\hat{T}_Z - \hat{T}_X)\} + (r_y \hat{T}_Z - \hat{T}_Y)\{(y' - r_y)\hat{Z} + (y'\hat{T}_Z - \hat{T}_Y)\}] = 0.$$

$$\hat{Z} = \frac{(r_x \hat{T}_Z - \hat{T}_X)(x'\hat{T}_Z - \hat{T}_X) + (r_y \hat{T}_Z - \hat{T}_Y)(y'\hat{T}_Z - \hat{T}_Y)}{(r_x \hat{T}_Z - \hat{T}_X)(r_x - x') + (r_y \hat{T}_Z - \hat{T}_Y)(r_y - y')}.$$

This is the only extremum of $\mathrm{Err}()$ with regard to $\hat{Z}$, which is the global minimum.

Acknowledgment

The authors would like to thank R. Vidal for his helpful comments on the optimization methods.

References

[1] Railroad safety statistics annual report. Office Safety Anal., Fed. Railroad Admin., U.S. Dept. Transport. [Online]. Available: http://safetydata.fra.dot.gov/officeofsafety/

[2] O. Shakernia, Y. Ma, T. J. Koo, and S. Sastry, "Landing an unmanned air vehicle: Vision based motion estimation and nonlinear control," *Asian J. Control*, vol. 1, no. 3, pp. 128–145, 1999.

[3] J. Oliensis, "A multi-frame structure-from-motion algorithm under perspective projection," *Int. J. Comp. Vision*, vol. 34, no. 2, pp. 163–192, 1999.

[4] M. Han and T. Kanade, "Creating 3-D models with uncalibrated cameras," in *Proc. 5th IEEE Workshop Applications of Computer Vision*, 2000, pp. 178–185.

[5] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2000.

[6] B. K. Horn. (1999) Projective geometry considered harmful [Online]. Available: http://www.ai.mit.edu/people/bkph/

[7] W. Forstner and E. Gulch, "A fast operator for detection and precise location of distinct points, corners, and centers of circular features," in *Proc. Int. Conf. Fast Processing of Photogrammetric Data*, 1987, pp. 281–305.

[8] R. Vidal and S. Sastry, "Optimal segmentation of dynamic scenes from two perspective views," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, 2003, pp. 281–286.

[9] K. Daniilidis and H.-H. Nagel, "Analytical results on error sensitivity of motion estimation from two views," *Image Vision Comput.*, vol. 8, pp. 297–303, 1990.

[10] R. I. Hartley and P. Sturm, "Triangulation," in *Proc. Conf. Computer Analysis of Images and Patterns*, 1995, pp. 190–197.

[11] D. Nistér, "Preemptive ransac for live structure and motion estimation," in *Proc. IEEE Int. Conf. Computer Vision*, vol. 1, 2003, pp. 199–206.

Stochastic Car Tracking With Line- and Color-Based Features

Tao Xiong and Christian Debrunner

***Abstract*—Color- and edge-based trackers can often be "distracted," causing them to track the wrong object. Many researchers have dealt with this problem by using multiple features, as it is unlikely that all will be distracted at the same time. It is also important for the tracker to maintain multiple hypotheses for the state; sequential Monte Carlo filters have been shown to be a convenient and straightforward means of maintaining multiple hypotheses. In this paper, we improve the accuracy and robustness of real-time tracking by combining a color histogram feature with an edge-gradient-based shape feature under a sequential Monte Carlo framework.**

***Index Terms*—Color-based tracking, condensation, edge-based tracking, feature integration, Monte Carlo filter, multiple hypotheses, particle filter.**

I. Introduction

In recent years, researchers have developed the concept of driver-assistance systems as a means of reducing the number of traffic accidents and increasing driver comfort. Different types of sensors, such as radar, laser, and acoustic, have been considered for sensing in this application. Thanks to the increasingly powerful computer systems and the less-expensive high-performance video cameras that have become available in the past few years, the use of computer vision technology as a sensor in driver-assistance systems has become more common and has led to increased performance. Vision sensors can provide rich information about the vehicle's surroundings and also have the advantage over active sensors (such as laser rangers or radars) of not causing intervehicle interference.

Manuscript received December 1, 2003; revised July 15, 2004 and August 1, 2004. The Associate Editor for this paper was F.-Y. Wang.

T. Xiong is with the Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: txiong@ece.umn.edu).

C. Debrunner is with the Engineering Division, Colorado School of Mines, Golden, CO 80401 USA (e-mail: cdebrunn@mines.edu).

Digital Object Identifier 10.1109/TITS.2004.838192

In this paper, we present a vision algorithm that is able to track other vehicles from a moving vehicle. This system has been tested on several hours worth of real traffic data and is found to perform robustly and stably in a relatively large range of road and weather conditions. Also, the system runs easily in near real time, qualifying it for deployment in many current applications. We will first review previous work and then provide a detailed algorithm description and experimental results in subsequent sections.

II. Previous Work

Tracking algorithms have a long history in computer vision research. In particular, in the intelligent transportation systems (ITS) area, vehicle tracking, one of the specialized tracking paradigms, has been extensively investigated. Beymer *et al.* [1], Smith [2], and Stauffer and Grimson [3] develop a vehicle-tracking system by tracking features such as points and lines. Vehicle subfeatures are then grouped together using the constraints of common motion. This method has the advantage of dealing with partial occlusions, but the feature-grouping process can be problematic and it can be difficult to determine the features accurately and quickly. Koller [4] and Gardner [5] use three-dimensional (3-D) models of vehicle shapes. Line edge segments extracted from the image are matched to the two-dimensional (2-D) model edge segments that are obtained by projecting a 3-D polyhedral model of the vehicle into the image plane. The weakness in this approach lies in the fact that it is unrealistic to expect it to be able to have detailed models for all vehicles that could be found on the roadway. Active contour models or snakes [6] have also been used by several authors to extract contours of vehicles for tracking purpose [4], [7]. These, however, require accurate initialization and sometimes do not converge reliably in the presence of distracting features.

While these methods are effective in less crowded situations, most cannot reliability track vehicles in situations that are complicated by the occlusion and clutter commonly seen in realistic driving scenarios. In addition, most vehicle-tracking algorithms use Kalman filters, which are often distracted by background clutter because they cannot accurately represent multimodal state distributions generated by ambiguous observations.

Many computer vision researchers have explored object tracking in domains other than vehicle tracking by using a variety of image features (or measures), including shape and color histogram (e.g., [8]–[11]). To improve robustness over single-measure-based methods, some of these researchers have built trackers based on the integration of multiple measures, so tracking could be maintained even if one or more of the measures is unreliable [8], [9], [12]–[14]. These methods have shown the advantage of integrating multiple sources of information, but the best means of integrating them remains an open question. Methods that have been explored include combining measurements as the product of probabilities [15], additively combining edge and color measures [8], *democratic integration* [16], [17] that forms a linear combination of cues and adapts the weights of the linear combination, and sequential Monte Carlo filters using a weighted sum of the cues [16]. Another multiple cue integration method is *importance sampling* [14]. This method injects new state samples into the standard prediction and diffusion process of a Monte Carlo filter. Wu and Huang [13] use a similar approach that they refer to as co-inference, in which they maintain three weights for each state sample, one based on each of two measures and one based on the combination of the measures. We also maintain a set of weights based on each measure, as well as the combination of the measures, but use them differently than as in [13]. We draw from the state distributions based on individual measures to ensure that the correct samples are included, even if only one measure provides correct information.

In this paper, we present a novel car-tracking method based on the integration of a shape and color measure using a sequential Monte Carlo filter. The integration is performed by independently tracking the state for each measure and injecting samples from these multiple distributions to ensure reliable tracking even if only a single measure is reliable. Our shape measure is based on the ideas of [18] and [19]. We track the boundary of the vehicle, but instead of using an active contour to model the shape of vehicles, we track only the rectangular 2-D bounding box. This is justified by the fact that there are generally strong edges on all sides. This simplification leads to considerable information compression and a corresponding reduction in runtime without a reduction in the performance of the tracker. The color measure is based on matching the histogram of a reference image region to the current image region using the method described in [10]. To better deal with partial occlusion and background distractions in cluttered traffic scenes, the shape and color measures are incorporated under a sequential Monte Carlo sampling framework [20]. The resulting tracker is tested extensively on several hours of real image sequences and is shown to be robust to partial occlusion, illumination change, significant clutter, target scale variations, rotations in depth, similar background colors, and ego-motion of the observing platform.

III. Algorithm Description

In sequential Monte Carlo filters, as in Kalman filters, the state is updated in two steps: a prediction step that predicts the next state in terms of the current state and a measurement update step that updates the state based on measurements from the image. Denoting by X_t and Z_t, respectively, the state at time t and the measurements at time t and earlier, the sequential Monte Carlo tracking algorithm maintains an estimate of the posterior distribution $p(X_t \mid Z_t)$ as a set of M weighted samples $\{x_t^m\}_{m=1,\ldots,M}$. In the prediction step, the samples are propagated to new states at the next time step and, in the measurement update step, the weights associated with the samples are updated based on the measurements. The following section describes the integration of the shape and color measures in the sequential Monte Carlo filter; subsequent sections define the color and shape similarity measures.

A. Sequential Monte Carlo Filter

The sequential Monte Carlo filter we use in our tracking method is defined in terms of its state, dynamic model, and measurement update. The steps of the filter algorithm are shown in Fig. 1, where steps 1 and 2 implement the dynamic model and step 3 implements the measurement update. As in [10], the state in our approach consists of the image position and the scale of the target at times t and $t-1$. For each state sample x_t^m at time t, we maintain three weights denoted by $\omega_t^{s,(m)}$, $\omega_t^{c,(m)}$ and $\omega_t^{(m)}$, which are based on the shape measurement, color measurement, and a combination of both measurements, respectively. The combined weight is computed as

$$\omega_t^{(n)} = p_t^s \omega_t^{s,(n)} + (1 - p_t^s)\, \omega_t^{c,(n)} \tag{1}$$

where $p_t^s \in (0,1)$ represents our confidence of the shape cue relative to the color cue at time t.

Importance sampling is a technique that was developed to improve the efficiency of factored sampling. It applies when auxiliary knowledge is available in the form of an importance function $g(X_t)$ describing which areas of state–space contain most information about the posterior [14]. Importance sampling concentrates samples in those areas of state–space by generating sample positions x_t^m from $g(X_t)$ rather than sampling from the prior $p(X_t)$. The weights of these samples are then adjusted to insure that the samples represent the original

Generate $\{(x_t^m, \omega_t^{(m)}, \omega_t^{s,(m)}, \omega_t^{c,(m)})\}$ from
$\{(x_{t-1}^m, \omega_{t-1}^{(m)}, \omega_{t-1}^{s,(m)}, \omega_{t-1}^{c,(m)})\}$, $m = 1, \ldots, M$.

Construct the m^{th} of M new samples as follows:

1. Choose the sampling method by generating a uniformly distributed random number $\alpha \in [0,1)$.
2. Sample from the prediction density as follows:
   (a) If $\alpha < q$, choose x_t^m by sampling from $g_{s,(t-1)}$ and set $\lambda_t^m = f_t(x_t^m) / g_{s,(t-1)}(x_t^m)$ where $f_t(x_t^m) = \sum_{k=1}^{M} \omega_{t-1}^{(j)} p(X_t = x_t^m \mid X_{t-1} = x_{t-1}^k)$.
   (b) If $q \le \alpha < q + r$, choose x_t^m by sampling from $g_{c,(t-1)}$ and set $\lambda_t^m = f_t(x_t^m) / g_{c,(t-1)}(x_t^m)$, $f_t(x_t^m)$ is the same as above.
   (c) If $\alpha \ge q + r$, choose a base sample x_{t-1}^i with probability $\omega_{t-1}^{(i)}$, then choose x_t^m by sampling from $p(X_t \mid X_{t-1} = x_{t-1}^i)$ and set $\lambda_t^m = 1$
3. Measurement update process
   $\omega_t^{s,(m)} = \lambda_t^m * p(Z_{s,t} \mid X_t = x_t^m)$
   $\omega_t^{c,(m)} = \lambda_t^m * p(Z_{c,t} \mid X_t = x_t^m)$
   Normalize $\omega_t^{c,(m)}$ and $\omega_t^{s,(m)}$ such that $\sum_{m=1}^{M} \omega_t^{c,(m)} = \sum_{m=1}^{M} \omega_t^{s,(m)} = 1$
   $\omega_t^{(m)} = p_t^s \omega_t^{s,(m)} + (1 - p_t^s) \omega_t^{c,(m)}$

Fig. 1. Tracking algorithm with importance sampling. See the text for details.

posterior. Thus, to approximate a posterior $p(X_t \mid Z_t)$, instead of sampling directly from the prior $p(X_t \mid Z_{t-1})$, the samples x_t^m could be drawn from the distribution $g_t(X_t)$ and the weight of each sample is chosen to be

$$\omega_t^{(m)} = \frac{f_t(x_t^m)}{g_t(x_t^m)} p(Z_t \mid X_t = x_t^m) \tag{2}$$

where $f_t(x_t^m) = p(X_t = x_t^m \mid Z_{t-1})$ and Z_t is the measurement at time t.

In step 2 of Fig. 1, we use two importance sampling functions, one from shape $g_{s,t}(X_t) \propto (x_t^m, \omega_t^{s,(m)})$ in step 2(a) and one from color $g_{c,t}(X_t) \propto (x_t^m, \omega_t^{c,(m)})$ in step 2(b). The basic idea behind these steps is that we would like to enable the samples with higher color or shape measurements, or both, to have better chances to propagate to the next time step. Step 3 of the figure then implements the weighting of (2).

Step 2(c) of Fig. 1 implements sampling from the dynamic model, as in the standard sequential Monte Carlo filter. The dynamic model assumes constant velocity and scale change rate from frame to frame, so to predict x_t^m from x_{t-1}^m we simply change its position and scale by the same amount that they changed between times $t-1$ and $t-2$ and add Gaussian-distributed random noise to account for uncertainty in the dynamic model.

Weighting in the linear combination of (1) is determined manually based on the specific tracking environment. This approach allows us to incorporate our prior knowledge of the reliability of each module. For example, the shape measure is more reliable if internal edges are included, but in some real-time applications (for example, the vehicle-tracking problem) it is not feasible to include internal structure in the shape model due to the additional computation that is required for tracking and for the initialization of the reference model. In such applications, we only track the outer contour of the object and reduce the shape cue weight, since the outer contour is generally a less reliable edge than many of the internal edges.

The method used for the initial detection of regions to be tracked will vary with the application. For instance, in a face-tracking application, one might create initial state samples based on the detection of skin-colored regions [13], [14]. In automobile-tracking applications, one can create initial state samples based on motion or trained appearance models. Alternatively, initialization of the state and histogram could be based on vehicle detections in range sensor data. In this paper, our focus is on the tracking component of the problem; therefore, we initialize our state samples and color histogram based on hand-selected regions in the first frame of the sequence.

B. Color Similarity

Our color-similarity measure is based on the similarity between the color histogram of a reference region in the first image and that of the image region in frame t represented by a sample x_t^m. In order to estimate the proper weight for this sample during the measurement update step (step 3 in Fig. 1), we need the conditional distribution $p(Z_{c,t} \mid X_t = x_t^m)$ of the color measure $Z_{c,t}$ at time t. This distribution could be estimated from training data, but for simplicity we follow Pérez [10] and define

$$p(Z_{c,t} \mid X_t = x_t^m) \propto \exp\{-\gamma_c D^2 [q^*, q_t(x_t^m)]\} \tag{3}$$

where γ_c is an experimentally determined constant and q^* and $q_t(x_t^m)$ are the N-element color histograms of the reference region and the region defined by x_t^m, respectively. The distance measure D is derived from the Bhattacharyya similarity coefficient and is defined as

$$D[q^*, q_t(x_t^m)] = \left[1 - \sum_{n=1}^{N} \sqrt{q^*(n) q_t(n; x_t^m)}\right]^{1/2}.$$

C. Shape Similarity

We use a rectangle of fixed aspect ratio (the bounding box) to model the shape contour of the vehicle. The vehicle candidates are those rectangular contours with strong edges on all sides. We define an objective function $G(I, X_t)$ that measures the strength of the edges in image I on all sides of the bounding box defined by state X_t and encodes our domain knowledge about what constitutes a likely bounding box. If we let $P(X_t)$ be the perimeter of the bounding box specified by state X_t and define $I_G(a)$ as the image gradient at location a in the image, we define $G(I, X_t)$ as the integral around the bounding box of the gradient component perpendicular to the contour on all four sides as

$$G(I, X_t) = \frac{1}{L I_{G,\max}} \int_{a \in P(X_t)} \left| I_G(a) \cdot O_{P(X_t)} \right| da$$

where $L = \int_{a \in P(X_t)} da$ is the length of the perimeter of the hypothesized boundary, $I_{G,\max} = \max_{a \in P(X_t)} |I_G(a)|$ is the maximum gradient magnitude along the boundary, and $O_{P(X_t)}(a)$ is the unit normal vector to the perimeter $P(X_t)$ at location a.

We can relate this objective function to a Bayesian likelihood function by using the Gibbs/Boltzmann distribution, as in [19]. This formula has its origin in statistical mechanics and reflects the intuition that lower energy states are more probable than higher energy states. As in [19], we define $E_s(X_t) = -G(I, X_t)$ as the energy term that we are trying to minimize. Then, the likelihood that we observe the

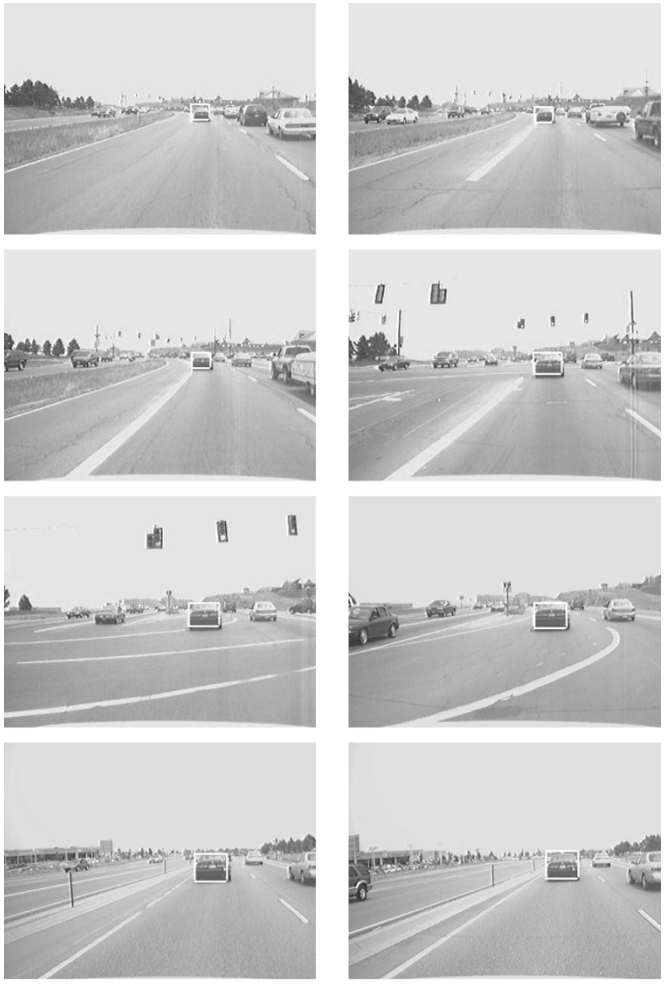

Fig. 2. Car-tracking result on a highway with little traffic.

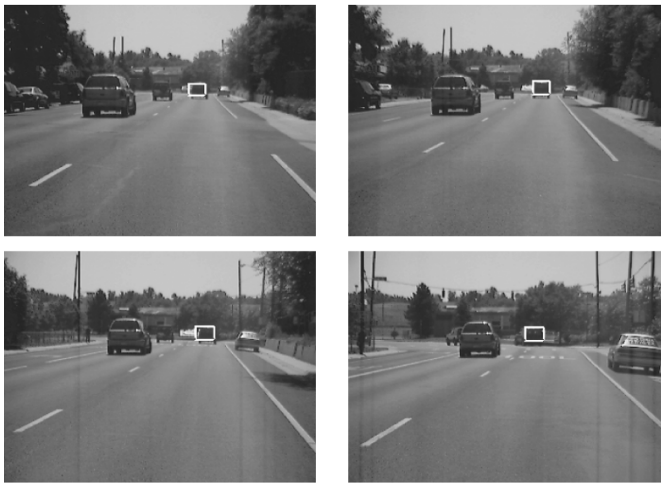

Fig. 3. Tracking result in low light.

measurement $Z_{s,t} = G(I, X_t)$ given that the state $X_t = x_t^m$ can be expressed as

$$P\left(Z_{s,t}\,|X_t = x_t^m\right) \propto \exp\left(-\gamma_s E_s(X_t)\right) \tag{4}$$

where γ_s is experimentally determined.

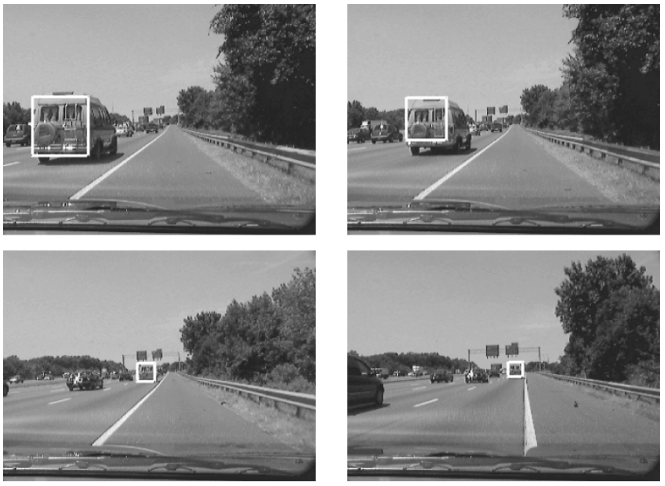

Fig. 4. Tracking results for a vehicle of a color similar to background (the road).

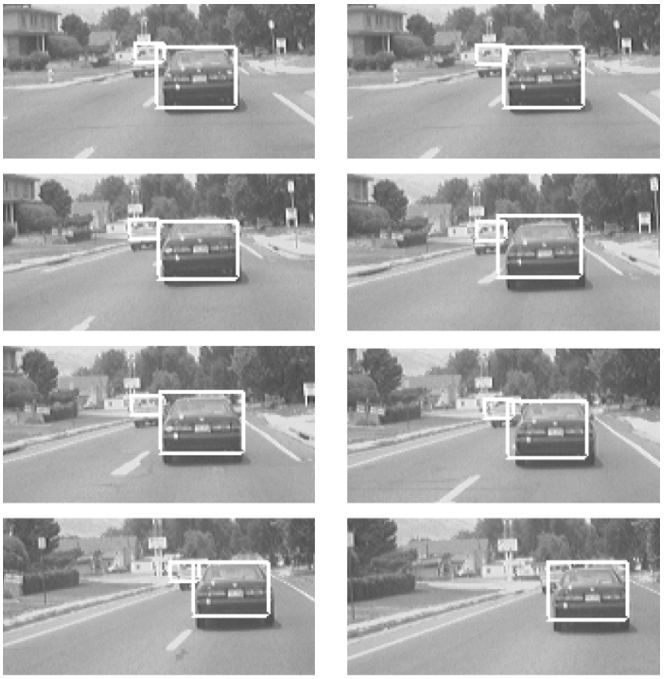

Fig. 5. Tracking results with occlusion. A magnified subimage of the original frame is shown for clarity. The tracking of the pickup on the left is terminated before the last frame.

IV. Experimental Results

We tested our tracking algorithm on several hours of traffic data collected from a camera on a moving vehicle. We set γ_c of (3) and γ_s of (4) to 10 in all the experiments. p_t^s of (1) is set to 0.5, which means that we trust the shape and color measurement equally well. Some sample tracking results are shown in Figs. 2–5.

In Fig. 2, car-tracking results in a highway scene are presented. The leading vehicle is accurately tracked through 9000 frames. Fig. 3 shows that our tracking algorithm also works in low light. In Fig. 4, a vehicle with a color that is similar to the background (the road) is successfully tracked because of the combined color and shape measurement. In Fig. 5, we show the capability of our algorithm to track a pickup truck even though it is partially occluded by the green car. Moreover, the problem is made more challenging by the similarity between the color of the background and the pickup. In tests of the algorithm using

only the color feature (using the same number of particles as in the two-feature experiments), tracking errors were larger than when both features were used.

In our experiments, very simple shape models were used. The generic shape models used appear to be adequate for reliable tracking, although it might be useful to also include strong internal edges taken from a reference image. In real-time applications, the increase in robustness from using additional edge features must be traded off against additional processing delays.

V. Conclusion

We have demonstrated a sequential Monte Carlo filter-based method for tracking objects in video sequences. This method uses a color histogram similarity feature and an edge-based shape feature to locate the object in each new frame and maintains three estimates of the state distribution based on the color measure, shape measure, and a combination of both measures. New samples are drawn from the shape and color distributions in an importance sampling approach and from the combined feature using a dynamic model of the tracking process. This method insures that some samples of the state space always capture the information from each of the measures, which captures the state distribution more efficiently and integrates the results of the measures.

The method was tested on an automobile-tracking application. The approach can reliably track (in almost 100% of the frames) the objects in scenes with little traffic. Also, it rarely gets lost in difficult sequences with dramatic color changes, poor lighting, large amounts of occlusion, and many background clutter edges.

Two recent papers describe methods that could further improve the reliability of our tracker. Li and Chellappa [21] add verification hypotheses to a particle filter, which could be used to maintain the identities of several vehicles during tracking, and Li and Chua [11] present a color-based tracker that allows changes in the color template with lighting changes.

References

[1] D. Beymer, P. Mclauchlan, B. Coifman, and J. Malik, "A real-time computer vision system for measuring traffic parameters," in *Computer Vision and Pattern Recognition Conf.*, June 1997, pp. 495–501.

[2] S. M. Smith and J. M. Brady, "ASSET-2: Real-time motion segmentation and shape tracking," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, pp. 814–820, Aug. 1995.

[3] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proc. Computer Vision and Pattern Recognition Conf.*, vol. II, June 1999, pp. 246–252.

[4] D. Koller, K. Daniilidis, and H.-H. Nagel, "Model-based object tracking in monocular image sequences of road traffic scenes," *Int. J. Comp. Vision*, vol. 10, pp. 257–281, 1993.

[5] W. F. Gardner and D. T. Lawton, "Interactive model-based vehicle tracking," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 1115–1121, Nov. 1996.

[6] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *Int. J. Comp. Vision*, vol. 1, pp. 321–331, 1988.

[7] N. Peterfreund, "Robust tracking of position and velocity with Kalman snakes," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 21, pp. 564–569, June 1999.

[8] S. Birchfield, "Elliptical head tracking using intensity gradients and color histograms," in *Proc. Computer Vision and Pattern Recognition Conf.*, Santa Barbara, CA, June 1998, pp. 232–237.

[9] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of nonrigid objects using mean shift," in *Computer Vision and Pattern Recognition Conf.*, vol. 2, Hilton Head Island, SC, June 2000, pp. 142–151.

[10] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," in *Eur. Conf. Computer Vision*, Copenhagen, Denmark, May 2002, pp. 661–675.

[11] J. Li and C.-S. Chua, "Transductive inference for color-based particle filter tracking," in *Int. Conf. Image Processing*, vol. 2, Sept. 2003, pp. 949–952.

[12] C. R. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland, "Pfinder: Real-time tracking of the human body," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 780–785, July 1997.

[13] Y. Wu and T. S. Huang, "A co-inference approach to robust visual tracking," in *Int. Conf. Computer Vision*, vol. 2, Vancouver, BC, Canada, July 2001, pp. 26–33.

[14] M. Isard and A. Blake, "Icondensation: Unifying low-level and high-level tracking in a stochastic framework," in *Eur. Conf. Computer Vision*, vol. 1, Freiburg, Germany, June 1998, pp. 893–908.

[15] M. Isard and J. MacCormick, "BraMBLe: A Bayesian multiple-blob tracker," in *Int. Conf. Computer Vision*, vol. 2, Vancouver, BC, Canada, July 2001, pp. 34–41.

[16] M. Spengler and B. Schiele, "Toward robust multi-cue integration for visual tracking," in *Proc. Int. Workshop Computer Vision Systems*, Vancouver, BC, Canada, July 2001, pp. 94–107.

[17] J. Triesch and C. V. D. Malsburg, "Democratic integration: Self-organized integration of adaptive cues," *Neural Comput.*, vol. 13, pp. 2049–2074, 2001.

[18] S. Gil, R. Milanese, and T. Pun, "Combining multiple motion estimates for vehicle tracking," in *Proc. Eur. Conf. Computer Vision*, vol. 2, Apr. 1996, pp. 307–320.

[19] F. Dellaert and C. Thorpe, "Robust car tracking using Kalman filtering and Bayesian templates," *SPIE: Intell. Transpor. Syst.*, 1997, to be published.

[20] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Processing*, vol. 50, pp. 174–188, Jan. 2002.

[21] B. Li and R. Chellappa, "A generic approach to simultaneous tracking and verification in video," *IEEE Trans. Image Processing*, vol. 11, pp. 530–544, May 2002.