Tracking Complex Objects using Graphical Object Models

Leonid Sigal[1], Ying Zhu[3], Dorin Comaniciu[2] and Michael Black[1]

[1] Department of Computer Science, Brown University, Providence, RI 02912
{ls, black}@cs.brown.edu
[2] Integrated Data Systems, Siemens Corporate Research, Princeton, NJ 08540
dorin.comaniciu@scr.siemens.com
[3] Real Time Vision & Modeling, Siemens Corporate Research, Princeton, NJ 08540
ying.zhu@scr.siemens.com

Abstract. We present a probabilistic framework for component-based automatic detection and tracking of objects in video. We represent objects as spatio-temporal two-layer graphical models, where each node corresponds to an object or component of an object at a given time, and the edges correspond to learned spatial and temporal constraints. Object detection and tracking is formulated as inference over a directed loopy graph, and is solved with non-parametric belief propagation. This type of object model allows object-detection to make use of temporal consistency (over an arbitrarily sized temporal window), and facilitates robust tracking of the object. The two layer structure of the graphical model allows inference over the entire object as well as individual components. AdaBoost detectors are used to define the likelihood and form proposal distributions for components. Proposal distributions provide 'bottom-up' information that is incorporated into the inference process, enabling automatic object detection and tracking. We illustrate our method by detecting and tracking two classes of objects, vehicles and pedestrians, in video sequences collected using a single grayscale uncalibrated car-mounted moving camera.

1 Introduction

The detection and tracking of complex objects in natural scenes requires rich models of object appearance that can cope with variability among instances of the object and across changing viewing and lighting conditions. Traditional optical flow methods are often ineffective for tracking objects because they are memoryless; that is, they lack any explicit model of object appearance. Here we seek a model of object appearance that is rich enough for both detection and tracking of objects such as people or vehicles in complex scenes. To that end we develop a probabilistic framework for automatic component-based detection and tracking. By combining object detection with tracking in a unified framework we can achieve a more robust solution for both problems. Tracking can make use of object detection for initialization and re-initialization during transient failures or occlusions, while object detection can be made more reliable by considering the consistency of the detection over time. Modeling objects by an arrangement of image-based (possibly overlapping) components, facilitates detection of complex articulated objects, as well as helps in handling partial object occlusions or local illumination changes.

Object detection and tracking is formulated as inference in a two-layer graphical model in which the coarse layer node represents the whole object and the fine layer nodes represent multiple component "parts" of the object. Directed edges between nodes represent learned spatial and temporal probabilistic constraints. Each node in the graphical model corresponds to a position and scale of the component or the object as a whole in an image at a given time instant. Each node also has an associated AdaBoost detector that is used to define the local image likelihood and a proposal process. In general the likelihoods and dependencies are not Gaussian. To infer the 2D position and scale at each node we exploit a form of non-parametric belief propagation (BP) that uses a variation of particle filtering and can be applied over a loopy graph [8, 15].

The problem of describing and recognizing categories of objects (e.g. faces, people, cars) is central to computer vision. It is common to represent objects as collections of features with distinctive appearance, spatial extent, and position [2, 6, 10, 11, 16, 17]. There is however a large variation in how many features one must use and how these features are detected and represented. Most algorithms rely on semi-supervised learning [11, 16, 17] schemes where examples of the desired class of objects must be manually aligned, and then learning algorithms are used to automatically select the features that best separate the images of the desired class from background image patches. More recent approaches learn the model in an unsupervised fashion from a set of unlabeled and unsegmented images [2, 6]. In particular, Fergus *et al* [6] develop a component based object detection algorithm that learns an explicit spatial relationship between parts of an object, but unlike our framework assumes Gaussian likelihoods and spatial relationships and does not model temporal consistency.

In contrast to part-based representations, simple discriminative classifiers treat an object as a single image region. Boosted classifiers [16], for example, while very successful tend to produce a large set of false positives. While this problem can be reduced by incorporating temporal information [17], discriminative classifiers based on boosting do not explicitly model parts or components of objects. Such part-based models are useful in the presence of partial occlusions, out-of-plane rotation and/or local lighting variations [5, 11, 18]. Part- or component-based detection is also capable of handling highly articulated objects [10], for which a single appearance model classifier may be hard to learn. An illustration of the usefulness of component-based detection for vehicles is shown in Fig. 1. While all vehicles have almost identical parts (tires, bumper, hood, etc.) their placement can vary significantly due to large variability in the height and type of vehicles.

Murphy *et al* [12] also use graphical models in the patch-based detection scheme. Unlike our approach they do not incorporate temporal information or

Fig. 1. Variation in the vehicle class of objects is shown. While objects shown here have a drastically different appearance as a whole due to the varying height and type of the vehicle, their components tend to be very homogeneous and are easy to model.

explicitly reason about the object as a whole. Also closely related is the work of [13] which uses AdaBoost for multi-target tracking and detection. However, their Boosted Particle Filter [13] does not integrate component-based object detection and is limited to temporal propagation in only one direction (forward in time). In contrast to these previous approaches we combine techniques from discriminative learning, graphical models, belief propagation, and particle filtering to achieve reliable multi-component object detection and tracking.

In our framework, object motion is represented via temporal constraints (edges) in the graphical model. These model-based constraints for the object and components are learned explicitly from the labeled data, and make no use of the optical flow information. However the model could be extended to use explicit flow information as part of the likelihood model, or as part of the proposal process. In particular, as part of the proposal process, optical flow information can be useful in focusing the search to the regions with "interesting" motion, that are likely to correspond to an object or part/component of an object.

2 Graphical Object Models

Following the framework of [14] we model an object as a spatio-temporal directed graphical model. Each node in the graph represents either the object or a component of the object at time t. Nodes have an associated state vector $X^T = (x, y, s)$ defining the component's real-valued position and scale within an image. The joint probability distribution for this spatio-temporal graphical object model with N components and over T frames can be written as:

$$
\begin{aligned}
P(\mathbf{X}_0^O, \mathbf{X}_0^{C_0}, \mathbf{X}_0^{C_1}, ..., \mathbf{X}_0^{C_N}, ..., \mathbf{X}_T^O, \mathbf{X}_T^{C_0}, \mathbf{X}_T^{C_1}, ..., \mathbf{X}_T^{C_N}, \mathbf{Y}_0, \mathbf{Y}_1, ..., \mathbf{Y}_T) = \\
\frac{1}{Z} \prod_{ij} \psi_{ij}(\mathbf{X}_i^O, \mathbf{X}_j^O) \prod_{ik} \psi_{ik}(\mathbf{X}_i^O, \mathbf{X}_i^{C_k}) \prod_{ikl} \psi_{kl}(\mathbf{X}_i^{C_k}, \mathbf{X}_i^{C_l}) \\
\prod_{i} \phi_i(\mathbf{Y}_i, \mathbf{X}_i^O) \prod_{ik} \phi_i(\mathbf{Y}_i, \mathbf{X}_i^{C_k})
\end{aligned}
$$

where $\mathbf{X}_t^O$ and $\mathbf{X}_t^{C_n}$ is the state of the object, O, and object's n-th component, C_n, at time t respectively ($n \in (1, N)$ and $t \in (1, T)$); $\psi_{ij}(\mathbf{X}_i^O, \mathbf{X}_j^O)$ is the temporal compatibility of the object state between frames i and j; $\psi_{ik}(\mathbf{X}_i^O, \mathbf{X}_i^{C_k})$ is the spatial compatibility of the object and it's components at frame i; $\psi_{kl}(\mathbf{X}_i^{C_k}, \mathbf{X}_i^{C_l})$ is the spatial compatibility between object components at frame i; and $\phi_i(\mathbf{Y}_i, \mathbf{X}_i^O)$ and $\phi_i(\mathbf{Y}_i, \mathbf{X}_i^{C_k})$ denote the local evidence (likelihood) for the object and component states respectively.

Our framework can be viewed as having five distinct components: (i) a graphical model, (ii) an inference algorithm that infers a probability distribution over the state variables at each node in the graph, (iii) a local evidence distribution (or image likelihood), (iv) a proposal process for some or all nodes in a graphical model, and (v) a set of spatial and/or temporal constraints corresponding to the edges in a graph. We will now discuss each one of these in turn.

2.1 Building the Graphical Model

For a single frame we represent objects using a two-layer spatial graphical model. The fine, component, layer contains a set of loosely connected "parts." The coarse, object, layer corresponds to an entire appearance model of the object and is connected to all constituent components. Examples of such models for pedestrian and vehicle detection are shown in a the shaded regions of Fig. $2a$ and $2b$ respectively. In both cases objects are modeled using four overlapping image components. For the vehicle the components are: top-left (TL), top-right (TR), bottom-right (BR) and bottom-left (BL) corners; while for the pedestrian, they are: head (HD), left arm (LA), right arm (RA) and legs (LG) (see Fig. $3ab$).

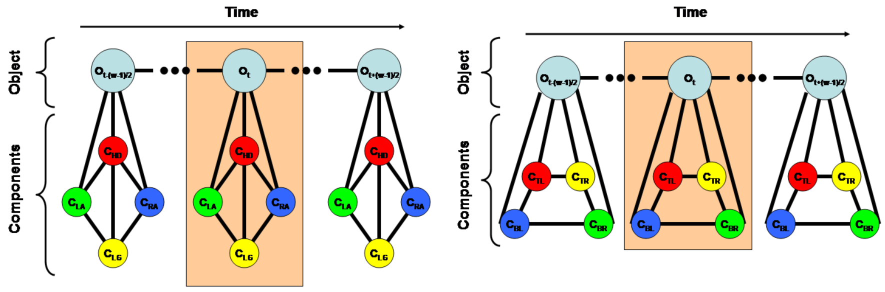
To integrate temporal constraints we extend the spatial graphical models over time to an arbitrary-length temporal window. The resulting spatio-temporal graphical models are shown in Fig. $2a$ and $2b$. Having a two-layer graphical model, unlike the single component layer model of [14], allows the inference process to reason explicitly about the object as a whole, as well as helps reduce the complexity of the graphical model, by allowing the assumption of independence of components over time conditioned on the overall object appearance. Alternatively, one can also imagine building a single object layer model, which would be similar to the Boosted Particle Filter [13] (with bi-directional temporal constraints).

2.2 Learning Spatial and Temporal Constraints

Each directed edge between components i and j has an associated potential function $\psi_{ij}(\mathbf{X}_i, \mathbf{X}_j)$ that encodes the compatibility between pairs of node states. The potential $\psi_{ij}(\mathbf{X}_i, \mathbf{X}_j)$ is modeled using a mixture of M_{ij} Gaussians (following [14])

$$\psi_{ij}(\mathbf{X}_i, \mathbf{X}_j) = \lambda^0 \mathcal{N}(\mathbf{X}_j; \mu_{ij}, \Lambda_{ij}) + (1 - \lambda^0) \sum_{m=1}^{M_{ij}} \pi_{ijm} \mathcal{N}(\mathbf{X}_j; F_{ijm}(\mathbf{X}_i), G_{ijm}(\mathbf{X}_i))$$

where λ^0 is a fixed outlier probability, μ_{ij} and Λ_{ij} are the mean and covariance of the Gaussian outlier process, and $F_{ijm}(\mathbf{X}_i)$ and $G_{ijm}(\mathbf{X}_i)$ are functions that return the mean and covariance matrix respectively of the m-th Gaussian



(a) (b)

Fig. 2. Graphical models for the (a) pedestrian and (b) vehicle detection and tracking. Spatio-temporal models are obtained by replicating a spatial model (shown by the shaded region) along the temporal domain to a w-length window and then connecting the object layer nodes across time.

mixture component. π_{ijm} is the relative weight of an individual component and $\sum_{m=1}^{M_{ij}} \pi_{ijm} = 1$. For experiments in this paper we used $M_{ij} = 2$ mixture components.

Given a set of labeled images, where each component is associated with a single reference point, we use standard iterative Expectation-Maximization (EM) algorithm with K-means initialization to learn $F_{ijm}(\mathbf{X}_i)$ of the form:

$$F_{ijm}(\mathbf{X}_i) = \mathbf{X}_i + \left[\frac{\mu^x_{ijm}}{\mu^s_{ijm}}, \frac{\mu^y_{ijm}}{\mu^s_{ijm}}, \mu^s_{ijm}\right]^T \tag{1}$$

where μ^x_{ijm}, μ^y_{ijm}, μ^s_{ijm} is the mean position and scale of component or object j relative to i. $G_{ijm}(\mathbf{X}_i)$ is assumed to be diagonal matrix, representing the variance in relative position and scale. Examples of the learned conditional distributions can be seen in Fig. 3*cde*.

2.3 AdaBoost Image Likelihoods

The likelihood, $\phi(\mathbf{Y}, \mathbf{X}^i)$ models the probability of observing the image $\mathbf{Y}$ conditioned on the state $\mathbf{X}^i$ of the node i, and ideally should be robust to partial occlusions and the variability of image statistics across many different inputs. To that end we build our likelihood model using a boosted classifier.

Following [16] we train boosted detectors for each component. For simplicity we use AdaBoost [16] without a cascade (training with a cascade would likely improve the computational efficiency of the system). In order to reduce the number of false positives produced by the detectors, we use a bootstrap procedure that iteratively adds false positives that are collected by running the trained strong classifier over the set of background images (not containing the desired object) and then re-training the detectors using the old positive and the new extended negative sets.

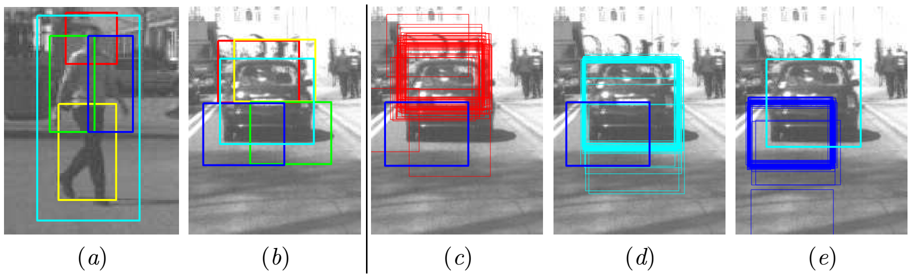


Fig. 3. Components for the (a) pedestrian and (b) vehicle object models (entire appearance model is in cyan) and learned conditional distributions from (c) Bottom-Left (BL) to Top-Left (TL) component, (d) Bottom-Left (BL) to the whole appearance model, and (e) whole appearance model to the Bottom-Left (BL) component.

Given a set of labeled patterns the AdaBoost procedure learns a weighted combination of base weak classifiers, $h(I) = \sum_{k=1}^{K} \alpha_k h_k(I)$, where I is an image pattern, and $h_k(I)$ is the weak classifier chosen for the round k of boosting, and α_k is the corresponding weight. We use a weak classifier scheme similar to the one discussed in [16]: $h_k(I) = p_k \left(\left[\sqrt[\beta_k]{(f_k(I))^{\beta_k}} < \theta_k \right] \right)$, where $f_k(I)$ is a feature of the pattern I computed by convolving I with the delta function over the extent of a spatial template; θ_k is a threshold, p_k is the polarity indicating the direction of inequality, and $\beta_k \in [1, 2]$ allowing for a symmetric two sided pulse classification.

The output of the AdaBoost classifier is a confidence $h_k(I)$ that the given pattern I is of the desired class. It is customary to consider an object present if $h_k(I) \geq \frac{1}{2} \sum_{k=1}^{K} \alpha_k$. We convert this confidence into a likelihood function by first normalizing the α_k's, so that $h(I) \in [0, 1]$, and then exponentiating

$$\phi(\mathbf{Y}, \mathbf{X}^i) \propto \exp(h(I)/T) \tag{2}$$

where image pattern I is obtained by cropping full image $\mathbf{Y}$ based on the state of the object or component $\mathbf{X}^i$; and T is an artificial temperature parameter that controls the smoothness of the likelihood function, with smaller values of T leading to peakier distribution. Consequently we can also anneal the likelihood by deriving a schedule with which T changes. We found an exponential annealing schedule $T = T_0 \upsilon^{\kappa}$, where T_0 is the initial temperature, υ is a fraction $\in (0, 1)$, and κ is the annealing iteration, to work well in practice. AdaBoost classifiers are learned using a database of 861 vehicles and 662 pedestrians [11]. The number of negative examples after bootstrapping tends to be on the order of 2000 to 3000.

Depending on an object one may or may not have a likelihood or a proposal process for the object layer nodes. For example if the whole appearance of an object is indeed too complicated to model as a whole (e.g. arbitrary size vehicles) and can only be modeled in terms of components, we can simply assume a uniform likelihood over the entire state space. In such cases the object layer nodes simply fuse the component information to produce estimates for the object state that are consistent over time.

It is worth noting that the assumption of local evidence independence implicit in our graphical model is only approximate, and may be violated in the regions where object and components overlap. In such cases the correlation or bias introduced into the inference process will depend on the nature of the filters chosen by the boosting procedure. While this approximation works well in practice, we plan to study it more formally in the future.

2.4 Non-parametric BP

Inferring the state of the object and it's components in our framework is defined as estimating belief in a graphical model. We use a form of non-parametric belief propagation [8] Pampas to deal with this task. The approach is a generalization of particle filtering [4] which allows inference over arbitrary graphs rather then a simple chain. In this generalization the 'message' used in standard belief propagation is approximated with a kernel density (formed by propagating a particle set through a mixture of Gaussians density), and the conditional distribution used in standard particle filtering is replaced by product of incoming messages. Most of the computational complexity lies in sampling from a product of kernel densities required for message passing and belief estimation; we use efficient sequential multi-scale Gibbs sampling and epsilon-exact sampling [7] to address this problem.

Individual messages may not constrain a node well, however the product over all incoming messages into the node tends to produce a very tight distribution in the state space. For example, any given component of a vehicle is incapable of estimating the height of the vehicle reliably, however once we integrate information from all components in the object layer node, we can get a very reliable estimate for the overall object size.

More formally a message m_{ij} from node i to node j is written as

$$m_{ij}(\mathbf{X}_j) = \int \psi_{ij}(\mathbf{X}_i, \mathbf{X}_j)\phi_i(\mathbf{Y}_i, \mathbf{X}_i) \prod_{k \in A_i/j} m_{ki}(\mathbf{X}_i) d\mathbf{X}_i, \tag{3}$$

where A_i/j is the set of neighbors of node i excluding node j and $\phi_i(\mathbf{Y}_i, \mathbf{X}_i)$ is the local evidence (or likelihood) associated with the node i, and $\psi_{ij}(\mathbf{X}_i, \mathbf{X}_j)$ is the potential designating the compatibility between the states of node i and j. The details of how the message updates can be carried out by stratified sampling from belief and proposal function see [14].

While it is possible and perhaps beneficial to perform inference over the spatio-temporal model defined for the entire image sequence, there are many applications for which this is impractical due to the lengthy off-line processing required. Hence, we use a w-frame windowed smoothing algorithm where w is an odd integer ≥ 1 (see Fig. 2). There are two ways one can do windowed smoothing: in an object-detection centric way or a tracking-centric way. In the former we re-initialize all nodes every time we shift a window, hence the temporal integration is only applied in the window of size w. In the tracking centric way we only initialize the nodes associated with a new frame, which tends to enforce temporal consistency from before $t-(w-1)/2$. While the later tends to converge faster and produce more consistent results over time, it is also less sensitive to objects entering and leaving the scene. Note that with $w = 1$, the algorithm resembles single-frame component-based fusion [18].

2.5 Proposal Process

To reliably detect and track the object non-parametric BP makes use of the bottom-up proposal process, that constantly looks for and suggests alternative hypothesis for the state of the object and components. We model a proposal distribution using a weighted particle set. To form a proposal particle set for a component, we run the corresponding AdaBoost detector over an image at a number of scales to produce a set of detection results that score above the $\frac{1}{2}\sum_{k=1}^{K}\alpha_k$ threshold. While this set tends to be manageable for the entire appearance model, it is usually large for non-specific component detectors (a few thousand locations can easily be found). To reduce the dimensionality we only keep the top P scoring detections, where P is on the order of a 100 to 200. To achieve breadth of search we generate proposed particles by importance sampling uniformly from the detections. For more details on the use of the proposal process in the PAMPAS framework see [14].

3 Experiments

Tests were performed using a set of images collected with a single car-mounted grayscale camera. The result of vehicle detection and tracking over a sequence of 55 consecutive frames can be seen in Fig. 5. A 3-frame spatio-temporal object model was used and was shifted in a tracking-centric way over time. We run BP with 30 particles for 10 iterations at every frame. For comparison we implemented a simple fusion scheme that averages the best detection result from each of the four components (Fig. 5(b) 'Best Avg.') to produce an estimate for the vehicle position and scale independently at every frame. The performance of the simple fusion detection is very poor suggesting that the noisy component detectors often do not have the global maximum at the correct position and scale. In contrast, the spatio-temporal object model consistently combines the evidence for accurate estimates throughout the sequence.

The performance of the pedestrian spatio-temporal detector is shown in Fig. 6. A 3-frame spatio-temporal object model is run at a single instance in time for two pedestrians in two different scenes. Similar to the vehicle detection we run BP with 30 particles for 10 iterations. For both experiments the temperature of the likelihood is set to $T_0 = 0.2$.

While in general the algorithm presented here is capable of detecting multiple targets, by converging to multi-modal distributions for components and objects, in practice this tends to be quite difficult and requires many particles. Particle filters in general have been shown to have difficulties when tracking multi-modal distributions [13]. The Pampas framework used here is an extension of particle filtering, and the message update involves taking a product over particle sets, consequently, Pampas suffers from similar problems. Furthermore, belief propagation over a loopy graph such as ours may further hinder the modeling of multi-modal distributions. To enable multi-target tracking then we employ a peak suppression scheme, where modes are detected one at a time, and then the response of the likelihood function is suppressed in the regions where peaks have already been found. An example of this obtained by running a purely spatial graphical model over the image containing 6 vehicles is shown in Fig. 4.

4 Conclusion

In this paper we present a novel object detection and tracking framework exploiting boosted classifiers and non-parametric belief propagation. The approach provides component-based detection and integrates temporal information over an arbitrary size temporal window. We illustrate the performance of the framework with two classes of objects: vehicles and pedestrians. In both cases we can reliably infer position and scale of the objects and their components. Further work needs to be done to evaluate how the method copes with changing lighting and occlusion. Additional work is necessary to develop a mutli-target scheme that incorporates a probabilistic model of the entire image.

The algorithm developed here is quite general and might be applied to other objection tracking and motion estimation problems. For example, we might formulate a parameterized model of facial motion in which the optical flow in different image regions (mouth, eyes, eyebrows) are modeled independently. These motion parameters for these regions could then be coupled via the graphical model and combined with a top-level head tracker. Such an approach might offer improved robustness over previous methods for modeling face motion [1].

References

1. Recognizing facial expressions in image sequences using local parameterized models of image motion, M. J. Black and Y. Yacoob. *International Journal of Computer Vision*, 25(1), pp. 23–48, 1997.
2. M. Burl, M. Weber and P. Perona. A probabilistic approach to object recognition using local photometry and global geometry, *ECCV*, pp. 628–641, 1998.
3. J. Coughlan and S. Ferreira. Finding deformable shapes using loopy belief propagation, *ECCV* Vol. 3, pp. 453–468, 2002.
4. A. Douce, N. de Freitas and N. Gordon. Sequential Monte Carlo methods in practice, *Statistics for Engineering and Information Sciences*, pp. 3–14, Springer Verlag, 2001.
5. P. Felzenszwalb and D. Huttenlocher. Efficient matching of pictorial structures, *CVPR*, Vol. 2, pp. 66–73, 2000.
6. R. Fergus, P. Perona and A. Zisserman. Object class recognition by unsupervised scale-invariant learning, *CVPR*, 2003.

7. A. Ihler, E. Sudderth, W. Freeman and A. Willsky. Efficient multiscale sampling from products of Gaussian mixtures, *Advances in Neural Info. Proc. Sys. 16*, 2003.
8. M. Isard. PAMPAS: Real-valued graphical models for computer vision, *CVPR*, Vol. 1, pp. 613–620, 2003.
9. M. Jordan, T. Sejnowski and T. Poggio. Graphical models: Foundations of neural computation, *MIT Press*, 2001.
10. K. Mikolajczyk, C. Schmid, and A. Zisserman. Human detection based on a probabilistic assembly of robust part detectors, *ECCV*, 2004.
11. A. Mohan, C. Papageorgiou and T. Poggio. Example-based object detection in images by components, *IEEE PAMI*, 23(4):349–361, 2001.
12. K. Murphy, A. Torralba and W. Freeman. Using the forest to see the trees: A graphical model relating features, objects, and scenes, *Advances in Neural Info. Proc. Sys. 16*, 2003.
13. K. Okuma, A. Teleghani, N. de Freitas, J. Little and D. Lowe. A boosted particle filter: Multitarget detection and tracking, *ECCV*, 2004.
14. L. Sigal, S. Bhatia, S. Roth, M. Black and M. Isard. Tracking loose-limbed people, *CVPR*, 2004.
15. E. Sudderth, A. Ihler, W. Freeman and A. Willsky. Nonparametric belief propagation, *CVPR*, Vol. 1, pp. 605–612, 2003; (see also MIT AI Lab Memo 2002-020).
16. P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features, *CVPR*, 2001.
17. P. Viola, M. Jones and D. Snow. Detecting pedestrians using patterns of motion and appearance, *ICCV*, pp. 734–741, 2003.
18. B. Xie, D. Comaniciu, V. Ramesh, M. Simon and T. Boult. Component fusion for face detection in the presence of heteroscedastic noise, *Annual Conf. of the German Society for Pattern Recognition (DAGM'03)*, pp. 434–441, 2003.

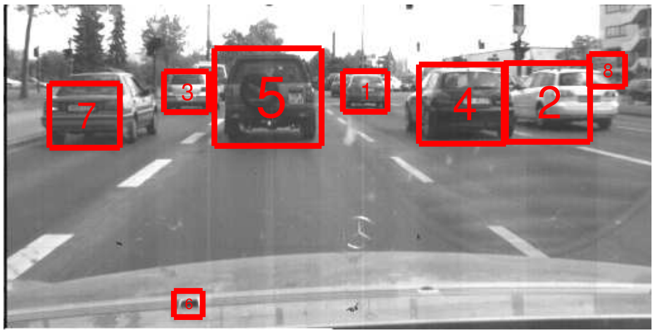


Fig. 4. Vehicle component-based spatio-temporal object detection for multiple targets. The algorithm was queried for 8 targets. The mean of 30 samples from the belief for each object are shown in red. Targets are found one at the time using an iterative approach that adjusts the likelihood functions to down weight regions where targets have already been found.

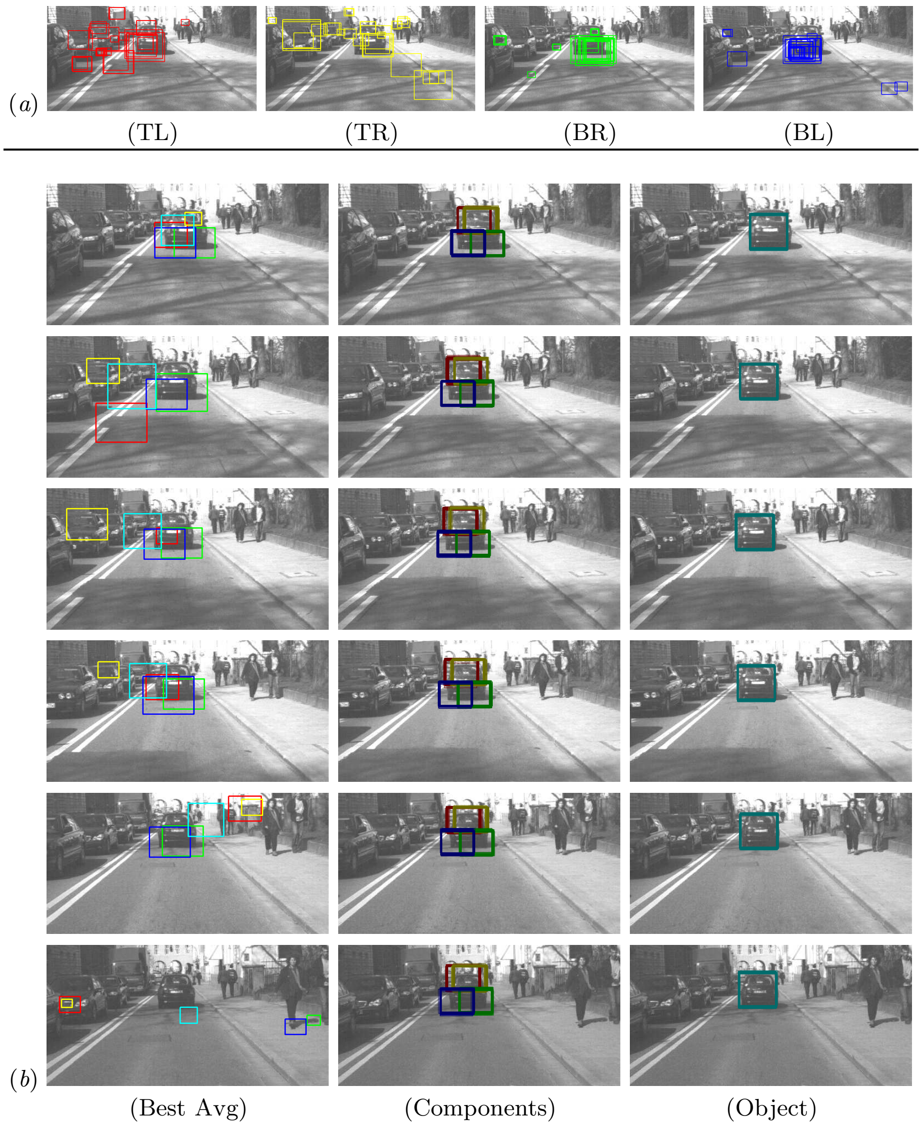


Fig. 5. Vehicle component-based spatio-temporal object detection and tracking. (a) shows samples from the initialization/proposal distribution, and (b) 30 samples taken from the belief for each of the four component (middle) and an object (right). The detection and tracking was conducted using a 3-frame smoothing window. Frames 2 through 52 are shown (top to bottom respectively) at 10 frame intervals. For comparison (b) (left) shows the performance of a very simple fusion algorithm, that fuses the best result from each of the components by averaging.

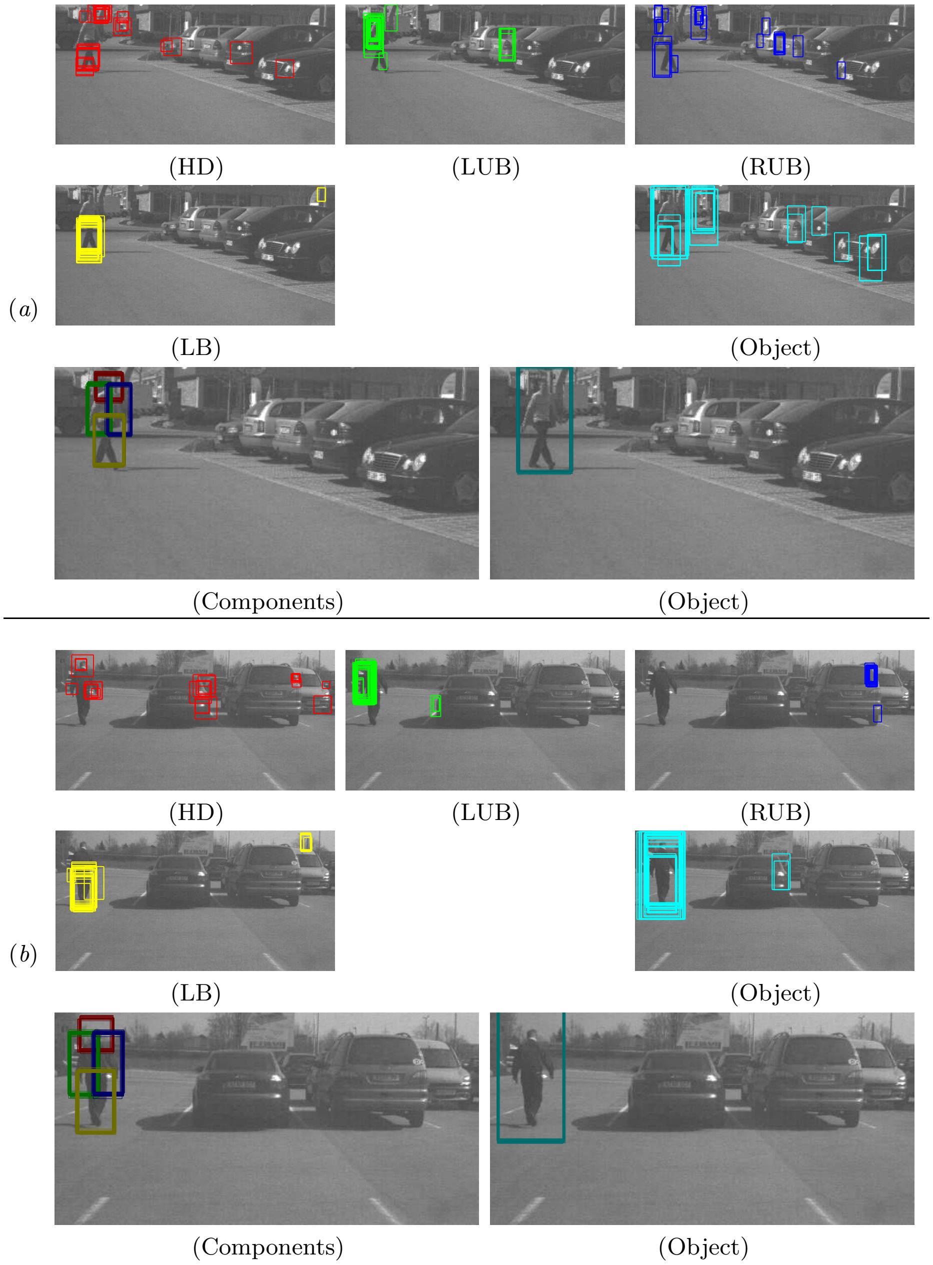


Fig. 6. Pedestrian component-based spatio-temporal object detection for two subjects (a) and (b). (top) shows the initialization/proposal distribution, and (bottom) 30 samples taken from the belief for each of the four component and an object. The detection was conducted using a 3-frame temporal window.