Automatic Bootstrapping and Tracking of Object Contours

John Chiverton, Xianghua Xie, *Member, IEEE*, and Majid Mirmehdi, *Senior Member, IEEE*

***Abstract*—A new fully automatic object tracking and segmentation framework is proposed. The framework consists of a motion-based bootstrapping algorithm concurrent to a shape-based active contour. The shape-based active contour uses finite shape memory that is automatically and continuously built from both the bootstrap process and the active-contour object tracker. A scheme is proposed to ensure that the finite shape memory is continuously updated but forgets unnecessary information. Two new ways of automatically extracting shape information from image data given a region of interest are also proposed. Results demonstrate that the bootstrapping stage provides important motion and shape information to the object tracker. This information is found to be essential for good (fully automatic) initialization of the active contour. Further results also demonstrate convergence properties of the content of the finite shape memory and similar object tracking performance in comparison with an object tracker with unlimited shape memory. Tests with an active contour using a fixed-shape prior also demonstrate superior performance for the proposed bootstrapped finite-shape-memory framework and similar performance when compared with a recently proposed active contour that uses an alternative online learning model.**

***Index Terms*—Active contour, level set, object segmentation, object tracking, online learning, shape modeling.**

I. Introduction

OBJECT contour tracking is a complicated process due to many factors, including variations in object appearance and deformations of object shape, e.g., for articulated objects or changing perspectives of 3-D objects. Therefore, object tracking approaches need to incorporate some form of shape modeling to enable successful contour localization and tracking.

An ideal medium for shape modeling is the active-contour model, which has been extensively investigated in conjunction with prior shape knowledge since at least [1] and [2]. These spline-based approaches are limited by topological constraints unlike level-set-based active-contour approaches, e.g., in [3]. Principal component analysis (PCA) is often used in these techniques to compress and summarize the important components of a set of characteristic level sets [3], [4] or control points modeled using an active shape model (ASM) [5]. These techniques use a set of representative 2-D contours of an object, which is learned *a priori* to enable more accurate contour localization in object segmentation or object tracking techniques.

Many prior shape-based tracking methods have been demonstrated to be quite robust, providing accurate outlines of the shape of the object being tracked. For example, Cremers [6] used a dynamical shape prior to enable tracking under high-noise conditions where the prior shape information was combined with a second-order autoregressive model of the transformation of the shapes of a person walking. Dambreville *et al.* [7] demonstrated the inability of a linear PCA shape space to fully describe the nonlinear deformations prevalent in deformable or articulated objects in static images and video data. The authors therefore chose to embed their shape prior in kernel space using kernel PCA and presented impressive segmentation and tracking results.

However, preparation of extensive prior shape knowledge is not always convenient and even cumbersome. Furthermore, many methods can encounter difficulties if the tracked object is protean and cannot be easily approximated by the current reduced dimensional shape space, i.e., a realistic prospect for articulated objects and their 2-D image projections. Some promising alternatives are available. Yilmaz *et al.* [8] proposed a tracking method that adapts to previously unseen shapes but only utilized shape information when an occlusion was detected. Yezzi and Soatto [9] described a framework that used a moving average of the shape information without reference to shapes seen in much earlier frames that may otherwise have provided useful information for much later frames.

Recently, Fussenegger *et al.* [10] described an approach that was able to update a reduced dimensional shape space online using robust incremental PCA [11]. Such methods are still dependent on manual extraction of relevant information about the objects to be tracked, requiring at least a manual segmentation of the object in an initial frame. In addition, while using a fixed-shape-prior-learned *a priori* provides a distinct advantage of making the object tracking process more robust, a dynamically learned prior can result in the active-contour model becoming more dependent on the manual adjustment of parameter values [12].

Online learning is not limited to shape-based techniques. Nummiaro *et al.* [13] used online learning for tracking and learning of a color distribution of the tracked object. Their model was based on a linear tradeoff between an existing model distribution and the color distribution for the current frame.

Manuscript received January 30, 2011; revised June 29, 2011; accepted August 24, 2011. Date of publication September 08, 2011; date of current version February 17, 2012. This work was initially supported by the University of Bristol through a grant from the U.K. Leverhulme Trust. The work of J. Chiverton was supported by an institutional research grant from Mae Fah Luang University, Thailand. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Yongyi Yang.

J. Chiverton is with the School of Information Technology, Mae Fah Luang University, Chiang Rai 57100, Thailand (e-mail: jpchiverton@gmail.com).

X. Xie is with the University of Wales Swansea, Department of Computer Science, Swansea SA2 8PP, U.K. (e-mail: x.xie@swansea.ac.uk).

M. Mirmehdi is with the Department of Computer Science, University of Bristol, Bristol BS8 1UB, U.K. (e-mail: majid@cs.bris.ac.uk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2011.2167343

1057-7149/$26.00 © 2011 IEEE

However, no attempt was made to learn the shape, and tracking was limited to an ellipse region that was manually initialized.

Tu *et al.* [14] also used an appearance-based model where a number of key frames were manually provided to help constrain the tracking system enabling head pose tracking. Online updating of the histogram information was used for individual frames, but new observations were not combined for future head pose tracking. Again, shape information was not included in the model formulation where the histograms for the mean-shift tracking were computed in a rectangular region of the image space.

Pan and Schonfeld [15] investigated the use of higher order particle filters to provide improved motion estimates obtained from a tracking system using an adaptive block-matching technique. Earlier variations of their tracking system included an active contour to adapt the tracked ellipse to the relatively static shape of the human head for individual frames in [16].

Gai and Stevenson [17] used robust appearance modeling in a tracking framework with the use of a student's t distribution version of PCA but without modeling shape information. The authors compared their work with that in [18], which was described as robust because of the use of an incremental PCA approach for (online) learning the appearance of the tracked object. However, the PCA subspace used by Ross *et al.* did not explicitly include a step or modeling to assist in the automatic rejection of outliers. In contrast to this, De La Torre and Black [19] described a robust approach to subspace learning using robust M-estimation. Their work was found to be computationally more demanding but equivalent in power to the work by Skocaj *et al.* [11]; a variant of which was later used by Fussenegger *et al.* [10] in their online adaptive active-contour framework.

A number of authors have combined feature tracking with object shape or contour tracking. Feature tracking helps to improve the performance of the process of object tracking and is relatively robust even if the object undergoes significant changes in shape or photometric properties, e.g., due to changing light conditions. Furthermore, shape information, such as an active contour combined with feature tracking, enables an object tracking approach to provide additional information to the tracking system. This may include new unseen shape information and further regional information such as the distribution of the colors of the object.

Smith and Brady [20] proposed an approach based on feature tracking combined with hulls attracted to edge strength. The technique provided a way of extracting the tracked object shape in the form of a radial map enveloped around clustered motion features. This limited shape information could be enhanced with the use of edges from an optional edge detector. McCane [21] described an approach that imposed constraints on the motion of features dependent on immediate neighbors connected by an edge in a minimum spanning tree (MST), which was also used by Smith and Brady, [20] but McCane [21] described this part in detail. A spider, which extends the concept of a 1-D boundary snake to the 2-D linking of features, was used as the basis of the technique. The spider was built using a Delaunay triangulation to identify neighboring corner features connected via live-wire paths (attracted to high image gradients); then, the MST was calculated on this graph resulting in a spider. Consistency of the image gradients across frames enabled identification of important edges. The results from two image sequences illustrated that this approach could identify some simple shape information suited to tracking objects in video data.

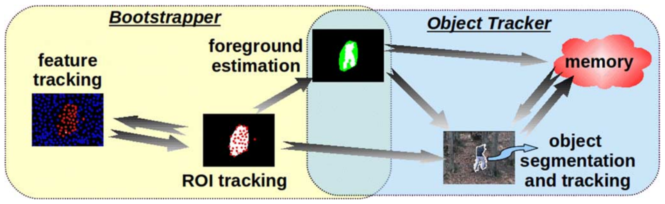


Fig. 1. Overview of the proposed system. The two major parts of the system are a bootstrapper and an object tracker. The bootstrapper is motion based and automatically extracts shape information for a subset of the frames. The object tracker continues the shape extraction process using a shape-based active contour. Both the bootstrapper and the object tracker subsystems perform shape extraction and tracking, with the object tracker performing the final role in this part of the process.

Gouet and Lameyre [22] used corner-feature tracking in combination with an active contour. Features were tracked using interframe spatial appearance matching rather than optical flow (i.e., similar to [20] and [21]), primarily to overcome problems associated with tracking through occlusions. An active contour was then defined around the envelope of the tracked features. The system required manual initialization for the first frame. Olszewska *et al.* [23] utilized an ASM on a set of corner features. The ASM was used to track objects, which in turn identified suitable bounds for corner points in each new frame unless an occlusion had been detected. As for [22], the ASM required manual initialization before tracking could commence.

Most of these techniques assume accurate corner identification and tracking, unlike that in [21], which indirectly checks the consistency of tracked features across frames via the edges connecting neighboring features. Gouet and Lameyre [22] describe an approach that somewhat mitigates inaccurate corner tracking by assigning high confidence to corner points within the converged active-contour region. Unfortunately, this does assume that the active contour can converge to a suitable minima without the use of a shape prior, which is often not possible when the video data consists of complex photometric information in the foreground or background.

A. *Our Approach*

This paper proposes an online active-contour-based shape learning model, which is fully automatic. Unlike our previous work [12], the system applies an original automated bootstrapping stage, and it is further combined with a novel multilevel approach to feature, region, and object tracking. The system also proposes a finite-sized shape memory, which automatically eliminates unnecessary shape information. An overview of the system is shown in Fig. 1.

Overall, the proposed framework tackles a number of important and unsolved issues in shape-based object tracking, i.e., bootstrapping and online learning of an object's changing shape. The objective of the framework is to track automatically a single moving object in the video data while dynamically learning the shape of the object over time, continuously remembering the most important shape information. This provides an extensive framework for tracking of shapes in video data, placing signif-

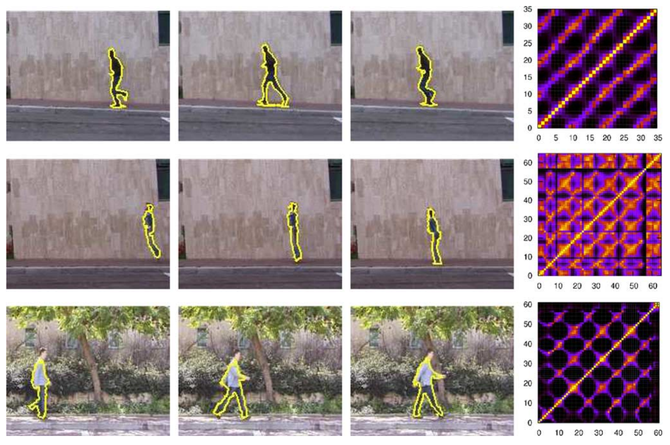


Fig. 2. Tracking results and corresponding similarity weight matrices illustrating the commonality of tracked shapes over time. Results from [12]. Original data from [29].

icant emphasis on past shape modeling defined and learned online, in conjunction to the tracking process.

B. Paper Organization

The next section presents the proposed methodology. Section II-A describes the feature tracking, clustering, and higher level region-based tracking technique. Section II-B describes how shape information is automatically extracted from the tracked regions. Sections II-C and II-D describe the initialization and updating of the object shape memory. Section II-E describes the shape-based active contour, and Section II-F provides an algorithmic overview of the entire system. Section III presents results and discussion. Section IV closes the paper with a discussion and some conclusions.

II. Proposed Framework

There are two main components of the proposed tracking framework, as shown in Fig. 1: a bootstrapper stage, which is a motion-based sparse-in-time shape and region tracker, and an active-contour stage that tracks shapes,which is guided by the output of the bootstrap stage.

The object shape is bootstrapped here by assuming that, for at least a subset of the frames, the object will possess some form of differential photometric properties from the region immediately surrounding the object. Two alternative approaches are described that bootstrap this information purely from crude regional information: one operating over each object region as a whole and the other performing a local differential analysis. Shapes that are extracted using these processes are then analyzed for consistency across individual frames using a shape overlap metric. Consistent shapes are then stored in an associative memory for tracking by the shape-based active-contour tracking framework. The similarity of learned shape information over time can be observed in similarity weight matrices of the contents of the memory (see Fig. 2). Shapes that are similar to previously observed shapes can be forgotten.

Each region from which the shape information is extracted also defines a point set of features that are required to satisfy correspondence between frames. We track individual features based on the feature detector in [24] using measurements from estimates of optical flow combined with individual Kalman filters for each tracked feature. Features with distinct motions are then spatially clustered using alpha shapes[1] on the Delaunay triangulations of features, which are motion clustered using random sample consensus (RANSAC) [26]. RANSAC is used to identify the dominant background motion (see, e.g., [27] and [28]); hence, objects of interest are assumed to possess outlier feature motion, at least for a subset of the frames being analyzed. These alpha shapes then define spatial regions that are tracked also with individual Kalman filters.

Errors due to inaccurate feature tracking and natural motions present in the video frames (such as nonplanar motions) are handled implicitly via a normalized measure of intersection of the feature sets (between frames) by virtue of a discrete overlap measure. These regions then define plausible regions from which an object should be tracked, i.e., an object that has motion that is significantly different from the background motion.

Following the bootstrapper stage is an object tracking stage that consists of a shape-based active-contour object tracker. The bootstrapper stage provides an initial memory from which the object tracker can perform a more detailed analysis of the data and infer the tracked object shape. The object tracker for each new frame provides a new object shape that can be included in the shape memory. The shape memory is then used to infer future object shapes. Some shapes in the shape memory will not be used very often, and shapes that have been used rarely are automatically removed. This means that the shape memory remains finite and that any future observations of shapes will be compared with shapes that have already been found to be useful. This results in more efficient shape memory usage and relatively stable shape memory where the integrity of the memory is more likely to remain intact for a longer (indefinite) period because irrelevant or badly extracted object shapes are more likely to be forgotten before first use so that the tracked shape is not adversely affected. The robustness of the active-contour framework is enhanced further with the use of the alpha shapes as an initial prior for each individual frame to guide the tracking process and to reduce the likelihood of the propagation of errors in the object contour tracking process. Furthermore, the proposed method is not exclusively for video acquired using a stationary camera as no such explicit assumption (e.g., background subtraction) is used in the development of the system.

A. Feature Tracking, Clustering, and Region Tracking

Potential foreground features are identified based on nonmembership of the dominant background motion using RANSAC [26]. After this, they are spatially clustered by performing a Delaunay triangulation on the set of feature points and subsequently isolated from the nonbackground points as follows: clusters of features are spatially grouped by disconnecting any edges connecting background points with nonbackground points, then performing a graph-based connected component analysis to identify isolated subgraphs. Alpha hulls [25] of these subgraphs form envelopes surrounding spatially isolated feature sets, which can be used to identify sets of image pixels.

[1]An alpha-shape is a hull that can be convex or concave [25].

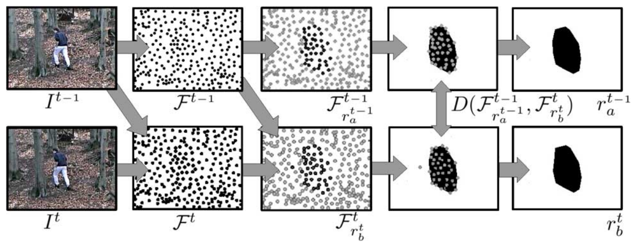


Fig. 3. Overview of the process of feature tracking and clustering. Initially, Kalman-filtered Shi and Tomasi image features are detected as sparse feature sets $\mathcal{F}^t$ at each time frame t. The dominant global motion is then detected and used to eliminate background feature points, which provides a sparse definition of foreground ROIs $\mathcal{F}^t_{r_b^t}$. Foreground ROIs are then spatially isolated via connected component labeling on a Delaunay triangulation of the features. Alpha shapes are drawn around the resulting isolated foreground components to form regularly and densely sampled definitions of the respective ROIs r_b^t.

The image features in [24] are used to detect a sparse feature set, and the motions of the detected features are estimated using pyramid-based Lucas–Kanade optical flow [30]. An acceleration-model Kalman filter is assigned to each feature to perform individual motion estimation and to reduce the effect of measurement errors. Each clustered set of pixels enveloped by an alpha hull defines an image region; the motion of which is also individually tracked using a Kalman filter. This region-based information provides feedback to the individual features, including correction of nonbackground membership motion clusters. This reduces the effect of inaccurate motion estimation and tracking for individual features.

The corresponding regions r_a^{t-1} and r_b^t from different frames, e.g., $t-1$ and t, will envelope corresponding features from each time instance $\mathcal{F}^{t-1}_{r_a^{t-1}}$ and $\mathcal{F}^t_{r_b^t}$. For the ideal case, the two sets of features from two corresponding regions at different time instances should be equivalent, i.e., $r_a^{t-1} \sim r_b^t$ if $\mathcal{F}^{t-1}_{r_a^{t-1}} \sim \mathcal{F}^t_{r_b^t}$. Correspondences between regions from different frames are therefore calculated based on a discrete overlap measure, e.g., the Dice coefficient $D(.,.)$, of the corresponding features between frames

$$D\left(\mathcal{F}^{t-1}_{r_a^{t-1}}, \mathcal{F}^t_{r_b^t}\right) = \frac{2\left|\mathcal{F}^{t-1}_{r_a^{t-1}} \cap \mathcal{F}^t_{r_b^t}\right|}{\left|\mathcal{F}^{t-1}_{r_a^{t-1}}\right| + \left|\mathcal{F}^t_{r_b^t}\right|} \tag{1}$$

where $\mathcal{F}^{t-1}_{r_a^{t-1}}$ is a set of features at time t. Correspondence between regions r_a^{t-1} and r_b^t will be declared if

$$D\left(\mathcal{F}^{t-1}_{r_a^{t-1}}, \mathcal{F}^t_{r_b^t}\right) > \eta \tag{2}$$

where η is a threshold set depending on the feature stickiness. If it evaluates as true, then r_a^{t-1} has many features in common with r_b^t. This provides an efficient and unambiguous approach to determining correspondences between regions. An overview of this process is illustrated in Fig. 3.

Next, the sets of pixels defined by each of the alpha hulls can be used to automatically extract the shape of the object that has generated the motion of the underlying features.

B. Bootstrapping the Shape Information: Shape Extraction

The photometric properties of the foreground region are initially unknown and have to be extracted dynamically from the given image information. This information may change over time due to, e.g., changes in illumination. Therefore, an approach that can adapt to the dynamic statistical properties of the photometric information is necessary. Many techniques seek to model the background statistics of the image, such as background subtraction techniques. However, this information is often difficult to extract and is not usually relevant to a moving observer.

The object shape from an alpha hull can be extracted using gradient-based edge detection techniques, such as that in [20]. However, gradient information is typically susceptible to noise or even textured image regions. Therefore, we propose two approaches of increasing computational intensity and accuracy that statistically estimate the foreground from the potential mixture of foreground and background enveloped by an alpha hull.

The first approach, i.e., referred to here as single background–foreground boosting (SB–FB), estimates the foreground using a single foreground–background model, which assumes that the tracked object possesses different photometric properties from any part of the background immediately surrounding the object. This assumption is sufficient for a sparse-in-time shape estimation technique, where the object being tracked will be different sufficiently (photometrically) from the background.

The second technique, i.e., referred to here as multiple background–foreground boosting (MB–FB), estimates the object shape using an assumption of local differential properties between the tracked object shape and smaller local regions surrounding the object. This latter approach is more robust, but it is computationally more intensive as it includes an instance of the first approach for each subregion (of which there are many).

1) SB–FB: Each motion and spatially clustered sets of image features are grouped and enveloped together by an alpha shape or hull. An alpha shape is assumed to completely surround a set of coherently moving image features and define the region of interest (ROI) $\mathfrak{T}$ of the image. $\mathfrak{T}$ not only contains foreground $\mathfrak{F}$, exhibiting photometric statistical properties of the object to be tracked, but also pixels corresponding to the immediate background $\mathfrak{B}$, i.e.,

$$\mathfrak{T} \subset (\mathfrak{F} \cup \mathfrak{B}). \tag{3}$$

A further region surrounding $\mathfrak{T}$, i.e., narrow background band $\mathfrak{N}$, can be defined, as illustrated in Fig. 4. $\mathfrak{N}$ will almost certainly contain a majority of pixels exhibiting photometric statistical properties of the background in the immediate pixel neighborhood as follows:

$$\mathfrak{N} \subset \mathfrak{B}. \tag{4}$$

Each pixel in these regions will then possess some photometric information.

Let $I^j_{\mathbf{x}} : \mathbb{R}^2 \times \mathbb{R}^+ \to \mathbb{R}^n$ be an n-dimensional image intensity at pixel $\mathbf{x} \in \mathbb{R}^2$ in the image frame $t \in \mathbb{R}^+$, where, e.g., $n = 3$ for RGB color images. Also consider the binary mask functions $\mathfrak{f} : \mathbb{R}^2 \to \{0, 1\}$ for the foreground and $\mathfrak{b} : \mathbb{R}^2 \to \{0, 1\}$ for the background, referred to here by $\mathfrak{f}_{\mathbf{x}}$ and $\mathfrak{b}_{\mathbf{x}}$, related by $\mathfrak{b}_{\mathbf{x}} = 1 - \mathfrak{f}_{\mathbf{x}}$ (for each image frame t, although t is not included to shorten the formulations). The set of foreground pixels is given

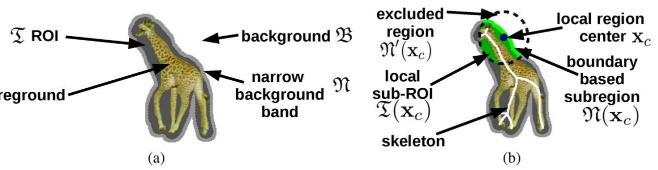

$\mathfrak{T}$ ROI
$\mathfrak{F}$ foreground
background $\mathfrak{B}$
narrow background band $\mathfrak{N}$
(a)
excluded region $\mathfrak{N}'(\mathbf{x}_c)$
local sub-ROI $\mathfrak{T}(\mathbf{x}_c)$
skeleton
local region center $\mathbf{x}_c$
boundary based subregion $\mathfrak{N}(\mathbf{x}_c)$
(b)

Fig. 4. Illustration of the bootstrap regions. The labels in (a) are applicable to SB–FB and MB–FB and in (b) is applicable to the regions for MB–FB only. (a) illustrates the global ROI $\mathfrak{T} \subset (\mathfrak{F} \cup \mathfrak{B})$, narrow background band $\mathfrak{N}$, and background $\mathfrak{B}$ regions with the true foreground $\mathfrak{F}$ enveloped in the ROI $\mathfrak{T}$. (b) illustrates an example of one of the many localized ROIs centered at points $\mathbf{x}_c$ along the boundary of the ROI $\mathfrak{T}$. These points are the centers of disks that are used to create localized ROIs $\mathfrak{T}(\mathbf{x}_c)$. Also illustrated is the local background region $\mathfrak{N}(\mathbf{x}_c)$ and the medial axis of $\mathfrak{T}$ used to identify suitable radii for the disks that define the localized ROIs.

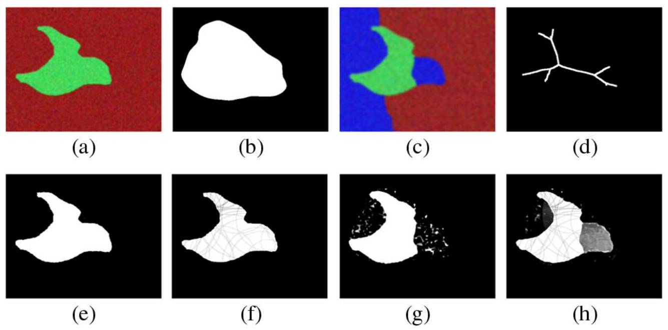


Fig. 5. SB–FB versus MB–FB. (a) Test shape with distinctive global characteristics. (b) ROI mask. (c) Test shape with variable local characteristics. (d) Medial axis of the ROI mask. (e) SB–FB result applied to (a). (f) MB–FB result applied to (a). (g) SB–FB result applied to (c). (h) MB–FB result applied to (c).

by $\mathfrak{F} = \{\mathbf{x}|\mathfrak{f}_\mathbf{x}\}$ and background pixels $\mathfrak{B} = \{\mathbf{x}|\mathfrak{b}_\mathbf{x}\}$, resulting in $\Omega = \mathfrak{F} \cup \mathfrak{B}$ as the set of pixels in the image space.

The foreground probability density function (pdf) $p(I_\mathbf{x}|\mathfrak{F})$, the background pdf $p(I_\mathbf{x}|\mathfrak{B})$, and the corresponding prior probabilities $P(\mathfrak{F})$ and $P(\mathfrak{B})$ can be associated with the foreground and background pixels, respectively. The probability of a particular pixel intensity corresponding to a foreground or background pixel can be then calculated via the Bayes theorem as follows:

$$P(\mathfrak{F}|I_\mathbf{x}) = \frac{p(I_\mathbf{x}|\mathfrak{F})P(\mathfrak{F})}{p(I_\mathbf{x})} \quad \text{and} \quad P(\mathfrak{B}|I_\mathbf{x}) = \frac{p(I_\mathbf{x}|\mathfrak{B})P(\mathfrak{B})}{p(I_\mathbf{x})} \tag{5}$$

where the marginal pdf is defined by

$$p(I_\mathbf{x}) = p(I_\mathbf{x}|\mathfrak{F})P(\mathfrak{F}) + p(I_\mathbf{x}|\mathfrak{B})P(\mathfrak{B}). \tag{6}$$

$p(I_\mathbf{x}|\mathfrak{B})$ is approximated by learning the statistics in the narrow band, i.e., $p(I_\mathbf{x}|\mathfrak{B}) \approx p(I_\mathbf{x}|\mathfrak{N})$, which is a reasonable approximation because: the narrow band $\mathfrak{N}$ is by definition a subset of the background by (4); $\mathfrak{N}$ is immediately adjacent to (envelopes) the object of interest so that the pdf of the intensities of the pixels in $\mathfrak{N}$ should be very similar to the pdf of the pixels in $\mathfrak{N}$ combined with the remaining relevant background pixels in $\mathfrak{T}$.

This means that all analysis can be limited to a subset of the image space, i.e., the ROI $\mathfrak{T}$ in combination with the narrow band $\mathfrak{N}$. The advantage to this is that it increases the foreground/background class separability because nonrelevant background information is automatically excluded, and in practice, all calculations are limited to the ROI and a small region surrounding the ROI. Therefore, the marginal pdf $p(I_\mathbf{x})$, which is valid for the entire sample space under consideration, is defined over the ROI and the narrow band so that $p(I_\mathbf{x}) \cong p(I_\mathbf{x}|\mathfrak{T}, \mathfrak{N})$, resulting in the statistical properties of the ROI being learned directly. Furthermore, by using Bayes in (5), the following can be stated:

$$P(\mathfrak{F}|I_\mathbf{x}) = 1 - P(\mathfrak{B}|I_\mathbf{x}) = 1 - \frac{p(I_\mathbf{x}|\mathfrak{B})P(\mathfrak{B})}{p(I_\mathbf{x})} \tag{7}$$

which, using the approximations now defined, can be calculated with

$$P(\mathfrak{F}|I_\mathbf{x}) = 1 - \frac{p(I|\mathfrak{N})P(\mathfrak{B})}{p(I_\mathbf{x}|\mathfrak{T}, \mathfrak{N})}. \tag{8}$$

The background prior probability $P(\mathfrak{B})$ is the only unknown term, which cannot be measured or estimated directly from the data. However, the above model is the two-class mixture model $(\mathfrak{B}, \mathfrak{F})$ [see (5) and (6)]. Therefore, $P(\mathfrak{B})$ can be estimated using maximum likelihood (ML) for a two-class mixture model where $P(\mathfrak{B}) = \sum_{\forall \mathbf{x} \in \mathfrak{T}, \mathfrak{N}} P(\mathfrak{B}|I_\mathbf{x})/|\mathfrak{T} \cup \mathfrak{N}|$ (see, e.g., [31], for a detailed explanation and derivation of ML parameter estimation for mixture models in general). Thus, an initial value for $P(\mathfrak{B})$ is selected, e.g., $P(\mathfrak{B})^{[0]} = 0.5$. Setting the iteration variable $l = 0$, $P(\mathfrak{B}|I_\mathbf{x})^{[l]} = 1 - P(\mathfrak{F}|I_\mathbf{x})^{[l]}$ is then calculated using (8). Then, an updated value for the background prior probability is calculated via

$$P(\mathfrak{B})^{[l+1]} = \frac{\sum\limits_{\forall \mathbf{x} \in \mathfrak{T}, \mathfrak{N}} P(\mathfrak{B}|I_\mathbf{x})}{|\mathfrak{T} \cup \mathfrak{N}|}. \tag{9}$$

where $|\mathfrak{T} \cup \mathfrak{N}|$ is the number of pixels in the ROI and the narrow band combined. The iteration variable l is next incremented, and (8) and (9) are repeatedly applied until convergence. After convergence, the foreground region can be defined as the set F of pixels that have higher probability of being foreground over the probability of being background, i.e.,

$$F = \{\mathbf{x}|P(\mathfrak{F}|I_\mathbf{x}) > P(\mathfrak{B}|I_\mathbf{x}), \qquad \mathbf{x} \in \mathfrak{T}\}. \tag{10}$$

This result suggests that $F = \mathfrak{F}$, which would imply a recursive formulation. However, (8) removes the need to calculate $P(\mathfrak{F}|I_\mathbf{x})$ directly.

A check can be made to determine if a foreground object has successfully been extracted by assessing the number of pixels in F. If the following is true:

$$|F| < |\mathfrak{T}| \times \gamma \tag{11}$$

where γ is a weight, then the algorithm is likely to have successfully extracted the pixels of the foreground. An obvious disadvantage of this test is that the algorithm may completely fail in providing an estimate of the foreground. This can be understood because if the number of pixels in F are similar to the number of pixels in $\mathfrak{T}$, then that might indicate that either the ROI was not well defined or the background pixels might possess similar photometric properties in conparison with the foreground. The next section describes an alternative approach that overcomes this limitation by analyzing the ROI in multiple overlapping small regions. The overall foreground is then calculated from all the successful individual subregions so that individual subregions that fail to provide a valid foreground estimate for their respective subregions do not immediately affect the final result.

2) MB–FB: The nonlocal (but object specific) approach described above is sufficient for many situations, particularly when a sparse-in-time shape extraction approach is required. However, some parts of a tracked object will often possess similar photometric properties in conparison with the background that immediately surrounds the object.

A more localized approach described here identifies the localized probabilistic differences between the background and the foreground. This is done by estimating the foreground from the foreground–background mixture in local regions surrounding the tracked object.

The local and global shape extraction techniques are illustrated in Figs. 4 and 5. The local regions are defined here with centers $\mathbf{x}_c$ on the boundary of the alpha shape by circular regions with radii given by twice the distance to the closest point on the medial axis of the alpha shape as follows:

$$r_{\mathbf{x}_c} = 2. \min_{\forall \mathbf{x}_s} \langle \mathbf{x}_c, \mathbf{x}_s \rangle \tag{12}$$

where $\mathbf{x}_s$ is a point on the medial axis of the alpha shape. The area defined by each disk with center $\mathbf{x}_c$ and radius $r_{\mathbf{x}_c}$ will cover a subregion of the alpha shape (i.e., $\mathfrak{T}(\mathbf{x}_c) \subset \mathfrak{T}$, see Fig. 4) and a further region outside of the alpha shape $\mathfrak{N}'(\mathbf{x}_c)$. This exterior region is reduced in size (to minimize the possibility of including extraneous background) via intersection with a global narrow band, as used previously, i.e.,

$$\mathfrak{N}(\mathbf{x}_c) = \mathfrak{N}'(\mathbf{x}_c) \cap \mathfrak{N}, \quad \text{so that} \quad \mathfrak{N}(\mathbf{x}_c) \subset \mathfrak{N}. \tag{13}$$

These boundary-based subregions can be used as part of the foreground–background model so that (8) and (9) can be applied to obtain a local foreground estimate as follows:

$$F_{\mathbf{x}_c} = \{\mathbf{x} | P(\mathfrak{F}_{\mathbf{x}_c} | I_\mathbf{x}) > P(\mathfrak{B}_{\mathbf{x}_c} | I_\mathbf{x}), \qquad \mathbf{x} \in \mathfrak{T}(\mathbf{x}_c)\} \tag{14}$$

where $P(\mathfrak{F}_{\mathbf{x}_c} | I_\mathbf{x})$ is calculated with the analogous local form of (8).

A check can be made to determine if the pixels of the foreground object subregion have successfully been extracted by assessing the number of pixels in $F_{\mathbf{x}_c}$, similar to (11), i.e.,

$$C(\mathbf{x}_c) = \begin{cases} \text{true} & \text{if} \sum_{\mathbf{x} \in \mathfrak{T}} P(\mathfrak{F}_{\mathbf{x}_c} | I_\mathbf{x}) < |\mathfrak{T}(\mathbf{x}_c)| \times \gamma, \\ \text{false} & \text{otherwise}. \end{cases} \tag{15}$$

If (15) returns "false," the algorithm is unlikely to have successfully extracted the pixels of the foreground subregion. Previously in (11), if the foreground had not been extracted successfully, then the entire foreground was rejected. Now, because subregions are being extracted, only the subregions that satisfy (15) are included in the final foreground estimate. Each pixel $\mathbf{x}$ may then have multiple foreground probability estimates $\{P(\mathfrak{F}_{\mathbf{x}_c} | I_\mathbf{x}) | C(\mathbf{x}_c)\}$ whenever $C(\mathbf{x}_c)$ is true. The mean of these foreground estimates can then be used as an estimate of the foreground posterior probability $P(\mathfrak{F} | I_\mathbf{x}) = \mathbb{E}[P(\mathfrak{F}_{\mathbf{x}_c} | I_\mathbf{x})]$ at a particular pixel location $\mathbf{x}$. Regions not satisfying (15) are considered zero foreground probability and are automatically excluded as follows:

$$P(\mathfrak{F} | I_\mathbf{x}) = \frac{\sum_{\forall \mathbf{x}_c | C(\mathbf{x}_c)} P(\mathfrak{F}_{\mathbf{x}_c} | I_\mathbf{x})}{|\{\mathbf{x}_c | C(\mathbf{x}_c), \qquad \mathbf{x} \in \mathfrak{F}_{\mathbf{x}_c}\}|} \tag{16}$$

where the denominator is the number of foreground subregions with $\mathbf{x} \in \mathfrak{F}_{\mathbf{x}_c}$ that satisfy $C(\mathbf{x}_c)$. As shown in Fig. 4(b), there is a substantial overlap between adjacent foreground subregions so that the result of (16) should be robust even for situations where $C(\mathbf{x}_c)$ is false for many $\mathbf{x}_c$.

C. Building a Collection of Plausible Object Shapes

The foreground extraction techniques described in the previous section are used to automatically extract the foreground of tracked objects from crude estimates of the regions surrounding the moving objects. These foreground shapes $\mathfrak{F}^t$ (where t is sampled real time) are estimated from the alpha hull of the outlier motion features $\mathfrak{T}^t$.

However, many frames may not result in useful object shapes due to a number of reasons, e.g., the alpha hulls are not guaranteed to cover the entire object all of the time, or individual features may be tracked inaccurately, which may result in erroneous alpha hulls; furthermore, photometric information may be ambiguous even in the locally defined approach. For these reasons, $\mathfrak{F}^t$ should only be remembered for some time instances. The first test for remembering a given shape is given by (11). A further test is also performed to determine if the shape at the time instance t is similar enough to the corresponding extracted shape at time instances $t-n, \ldots, t-2, t-1$. The similarities of the shapes are calculated using the Dice coefficient as follows:

$$D(\mathfrak{F}^{t-j}, \mathfrak{F}^t) = \frac{2|\mathfrak{F}^{t-j} \cap \mathfrak{F}^t|}{|\mathfrak{F}^{t-j}| + |\mathfrak{F}^t|} \tag{17}$$

where $\mathfrak{F}^{t-j}$ is translated to have the same center of gravity as $\mathfrak{F}^t$. Obviously, this measure is not rotation or scale invariant, but it is used here to show that if a shape has relatively high similarity with the shapes from the last n time instances, then $\mathfrak{F}^t$ must be a reasonably representative shape. Shape $\mathfrak{F}^t$ is included in memory $\mathfrak{F}^t \in Q$ if the following is true:

$$\mathbf{D}(\mathfrak{F}^t) = \begin{cases} \text{true} & \text{if } D(\mathfrak{F}^{t-i}, \mathfrak{F}^t) > \theta \quad \forall i, \qquad 1 \le i \le n \\ \text{false} & \text{otherwise}. \end{cases} \tag{18}$$

Threshold θ is set low enough to enable shapes that have undergone some deformation or rotation to be included, and it is set high enough to prevent completely erroneous shapes from being included.

D. Shape Memory for Object Tracking

Prior shape information is particularly useful in active-contour models to reduce the likelihood of the active contour deforming to unlikely configurations of shape. However, the prior shape information is difficult to obtain without manually segmenting and preparing suitable templates to be used for statistical modeling.

Despite this, the initial definition of the shape memory $Q = (q_\mathbf{x} = \mathfrak{f}_\mathbf{x} | \mathbf{x} \in \mathfrak{F}^t, \mathbf{D}(\mathfrak{F}^t), \forall t)$ (as a tuple) can be used as a first estimate of a sequence of shapes that an object tracker requires for full object tracking and segmentation. The foreground extraction stage can be run in batch mode across a set of images to provide an initial shape memory estimate. Alternatively, it can be run simultaneous to a shape-based active-contour model.

Furthermore, an object tracking process is highly likely to generate new configurations of shape that have not been previously observed, i.e., either as part of the training stage or in actual object tracking. It is therefore prudent to also remember this online information to enhance future object tracking. Thus, for a given time instance t, the shape-based active-contour framework tracks the object, resulting in a partition of the image space $\mathcal{P}^t = (\mathfrak{F}^t, \mathfrak{B}^t)$. The first element of the partition defines a new object shape $q_{\mathbf{x}} = \mathfrak{f}_{\mathbf{x}} | \mathbf{x} \in \mathfrak{F}^t$ that is included in the shape memory as follows:

$$Q^{t+1} = Q^t \cup (q_{\mathbf{x}} = \mathfrak{f}_{\mathbf{x}} | \mathbf{x} \in \mathfrak{F}^t, \qquad \mathfrak{F}^t \in \mathcal{P}^t). \tag{19}$$

However, continually including every tracked shape in the shape memory (from the shape-based active contour) is a potentially hazardous process as errors in object shape are likely to propagate into the object tracking process. If this occurs, the shape memory is likely to become dominated by nonrelevant shapes, resulting in a degeneration of the shape memory. To prevent this, a more selective shape memory can be designed.

Some observations regarding the object tracking process in relation to a more useful shape memory are the following.

1) At the start of object tracking, we have a somewhat (but limited) representative shape memory of the object being tracked.
2) At any time instance, shapes recently included in the shape memory are likely to be representative of the object being tracked recently.
3) Shapes found to be repeatedly similar to the object being tracked may continue to be similar to the shape of the object being tracked.
4) Some observed shapes may never be similar to a future object shape.
5) Computer memory and processing power are finite.

These observations enable us to build shape memory online that prioritizes recently observed and recently similar object shapes.

Therefore, the definition of the shape memory Q^t can be extended to include variables for each object shape $q \in Q^t$. These variables are $t_1 = t$, t_2, and h, which are the time when it was generated, the time when it was last used, and a count of the number of times an object shape has been recalled for use in the tracking framework, respectively. This last discrete frequency variable h is incremented if the currently tracked object shape is similar to a past observed shape.

The parameters enable some shapes to be prioritized over other shapes in terms of importance in the object tracking framework. A simple approach to ordering the object shapes is given by

$$o(t_1, t_2, h) = t_1 + t_2 \times h. \tag{20}$$

This order function o gives high priority to recently generated or observed shapes and shapes that have been used many times. The shape memory can then be redefined as an ordered sequence $\zeta_1, \ldots, \zeta_n$ of n four tuples $\zeta_v = (q^v, t_1^v, t_2^v, h^v)$, where ζ_v denotes the vth tuple. Each tuple includes an object shape q^v and the three parameters. The ordering of the tuples is defined by $o(\zeta_v) \geq o(\zeta_w)$ or equivalently $o(t_1^v, t_2^v, h^v) \geq o(t_1^w, t_2^w, h^w)$, where v and w are integer shape-order indicators so that $v > w$ if $\zeta_v > \zeta_w$.

The final two observations indicate that some shapes can be forgotten, and this ordering provides a simple approach to forgetting. If there are $n+1$-ordered shapes and only n object shapes should be remembered, then shape ζ_{n+1} is forgotten as

$$o(\zeta_{n+1}) = \min_{\forall v} o(\zeta_v) \tag{21}$$

i.e., $o(\zeta_v) \geq o(\zeta_{n+1}) \forall v$. Hence, the shape memory has constant length $\#Q = n$.

The construction of this heuristic helps to fulfill the five observations regarding the useful shape memory defined earlier. The principal idea of prioritization of some shapes over others, i.e., accomplished via (20), contributes to implementing observations (2) and (3), i.e., time sensitivity and repeated observations. The forgetting part not only helps to meet the requirement of finite memory in observation (5) but also helps to eliminate irrelevant shape information so that observations (2)–(4), i.e., the removal of irrelevant shapes, are also assisted by the forgetting part. The remaining observation (1) is a necessary requirement of a memory associated with a particular type of tracked objects where the initial memory content is provided here (fully automatically) by the sparse-in-time shape extraction technique described earlier.

E. Shape-Based Active Contour

The active contour is evolved using a combination of prior shape information Q^t (i.e., learned online, as described above) and the image information I^t for each individual frame t. The partition of the image space $\mathcal{P}^t = (\mathfrak{F}^t, \mathfrak{B}^t)$ into foreground and background pixels is dependent on the contents of the current shape memory Q^t, and the current image frame $I^t_{\mathbf{x}} \forall \mathbf{x} \in \Omega$. $\mathcal{P}^t$ is obtained here by maximizing the probability density of the partition given the shape memory and the image data as follows:

$$\max_{\mathcal{P}} p(\mathcal{P}^t | Q^t, I^t). \tag{22}$$

This is achieved here by four steps (dropping t).

Step 1) Predict: Find the shape in the memory with the maximum probability given the following current partition:

$$\max_{v} \left\{ p(q^v | \mathcal{P}) \right\}. \tag{23}$$

Step 2) Estimate: Evolve the contour toward shape q^v as follows:

$$\max_{\mathcal{P}} \left\{ p(\mathcal{P} | q^v, I) \right\}. \tag{24}$$

Step 3) Update: Update the shape memory using (19).

Step 4) Forget: Forget the lowest priority shape q^{n+1} in the shape memory using (21).

The prediction stage can be considered equivalent to finding a shape in the shape memory, which is most similar to the current partition. Therefore, (23) can be given by (see, e.g., [32])

$$\max_v \{p(q^v|\mathcal{P})\} \equiv \max_v \left\{ \exp\left(\frac{1}{2\sigma^2} \int_\Omega (\Phi_v(\mathbf{x}) - \Phi \cdot \mathcal{A}(\mathbf{x}))^2 \, dx \right) \right\} \quad (25)$$

where $\Phi \cdot \mathcal{A}(\mathbf{x})$ is the signed distance of the currently evolving shape aligned to the signed distance of shape q^v, i.e., $\Phi_v(\mathbf{x})$, and σ is a measure of variation between the shapes. Gradient descent is used to optimize the alignment of the shapes $\mathcal{A}(\mathbf{x})$ in (25), where the negative logarithm is taken similar to [32] and optimized with respect to rotation and translation but not scale due to difficulties with the convergence.

Using the Bayes theorem and assuming conditional independence of the image data I from shape q^v due to the current partition information $\mathcal{P}$, i.e., $p(I|v,\mathcal{P})p(v,\mathcal{P}) = p(I|\mathcal{P})p(v,\mathcal{P})$, the pdf to be maximized in the estimation stage (24) is then given by

$$p(\mathcal{P}|q^v, I) \propto p(I|\mathcal{P})p(q^v|\mathcal{P})p(\mathcal{P}) \quad (26)$$

where the partition pdf $p(\mathcal{P})$ enforces smoothness of the partitioned image space using an exponential distribution with exponent given by the gradient magnitude of the level set $|\nabla\Phi(\mathbf{x})|$ calculated along the boundary of the level set $\delta \cdot \Phi(\mathbf{x})$ (see, e.g., [33]), i.e.,

$$p(\mathcal{P}) \propto \exp\left(-\lambda_k \int (\delta \cdot \Phi(\mathbf{x})) \, |\nabla\Phi(\mathbf{x})| \right) \quad (27)$$

where λ_k is a contour length weight and δ is the Dirac delta function. The smoothness on the labeling enforced by $p(\mathcal{P})$ helps to condition the overall posterior distribution toward a unique solution so that the individual image intensities $I_\mathbf{x}$ in the image likelihood $p(I|\mathcal{P})$ can be assumed to be distributed independent of neighboring image intensities $p(I|\mathcal{P}) = \prod_{\forall\mathbf{x}} p(I_\mathbf{x}|\mathcal{P})$. Furthermore, the image intensities at each pixel $\mathbf{x}$ are considered to be generated by either foreground $\mathfrak{f}_\mathbf{x} = 1$ or background $\mathfrak{b}_\mathbf{x} = 1$ so that the individual intensity likelihoods can be split into a product of a foreground and a background term $p(I_\mathbf{x}|\mathcal{P} = \mathfrak{F} \cup \mathfrak{B}) = p(I_\mathbf{x}|\mathfrak{F})^{\mathfrak{f}_\mathbf{x}} p(I_\mathbf{x}|\mathfrak{B})^{\mathfrak{b}_\mathbf{x}}$. Thus, the overall image likelihood $p(I|\mathcal{P})$ is given by

$$p(I|\mathcal{P}) = \prod_{\forall\mathbf{x}} p(I_\mathbf{x}|\mathcal{P}) = \prod_{\forall\mathbf{x}} p(I_\mathbf{x}|\mathfrak{F})^{\mathfrak{f}_\mathbf{x}} p(I_\mathbf{x}|\mathfrak{B})^{\mathfrak{b}_\mathbf{x}}. \quad (28)$$

The shape prior $p(q^v|\mathcal{P})$ needs to pull the currently evolving partition $\mathcal{P}$ to a partition with a shape similar to the model shape q^v. This can be done via a pdf based on the sum of squared differences between the two sources of shape information. The two sources of shape information therefore have to be converted to the same form. The signed distance form is not closed under addition or subtraction and is therefore not suitable for use in a sum-of-squared-differences operation. Therefore, we use the Heaviside function of the signed distances to create two binary maps, where $H : \mathbb{R} \to \{0, 1\}$ is the Heaviside function with $H \cdot \Phi(\mathbf{x}) = 1$ if $\Phi(\mathbf{x}) \geq 0$. The shape prior is then given by

$$p(q^v|\mathcal{P}) = \prod_{\forall\mathbf{x}} \exp\left(-\lambda_s \left(H \cdot \Phi(\mathbf{x}) - H \cdot \Phi_v \cdot \mathcal{A}(\mathbf{x}) \right)^2 \right) \quad (29)$$

where λ_s is the shape weight that can be used to control the contribution of the model shape information. A prior based on the sum of squared differences of the Heaviside functions of the level sets has been used previously by other authors, e.g., Cremers *et al.* [34]. This prior is particularly attractive here because it contributes a single constant factor to or from points in the level set; as a result, the changing level set is likely to remain a valid level set even after a number of iterations. Other calculations based on the raw level-set values may not always produce a valid contribution to the changing level set, resulting in rapid nonregular degeneration of the level set.

1) Initialization and Tracking: The shape and position of the evolving contour encapsulated by the signed distance function $\Phi(\mathbf{x})$ has to be initialized at the arrival of every new image frame $t = t + 1$, $\tau = 0$, where τ is the gradient-descent optimization time variable. As $\tau \to \infty$, the active contour converges to a region in the image space $\Phi^{t,\tau\to\infty}$. This region typically surrounds the object being tracked. However, often the contour converges to a local minima, thereby producing an erroneous track of the object shape. Despite this, the converged contour may overlap the region occupied by the object to be tracked. Furthermore, the regional information provided by the alpha hull $\mathfrak{M}^{t+1}$ will likely contain the entire object to be tracked.

Therefore, the initialization of the object contour for every new image frame is a combination of the alpha hull for the new frame $\mathfrak{M}^{t+1}$ and the converged active contour for the previous frame

$$\Phi^{t+1,\tau=0}(\mathbf{x}) = f\left(\Phi^{t,\tau\to\infty} \cdot \mathcal{A}_K(\mathbf{x}), \mathfrak{M}^{t+1}(\mathbf{x}) \right) \quad (30)$$

where $\mathcal{A}_K(\mathbf{x})$ is a Kalman-filtered translational motion estimate calculated from the mean of the optical flow weighted by the spatially normalized image gradient magnitude (as a measure of confidence in the optical flow values) along the boundary of the contour (i.e., the zero level set) (see [12]). The motion estimate $\mathcal{A}_K(\mathbf{x})$ provides a useful initial location from which the active contour can evolve, and (30) can be interpreted as providing a useful initial estimate of object shape and location or as an interframe shape prior, so that $\hat{\Phi}^{t+1} = \Phi^{t+1,\tau=0}$. A linear function is used here with proportions $0 < \lambda < 0.5$ so that

$$\hat{\Phi}^{t+1} = \lambda \left(\Phi^{t,\tau\to\infty} \cdot \mathcal{A}_K(\mathbf{x}) \right) + (1 - \lambda) \left(\mathfrak{M}^{t+1}(\mathbf{x}) \right). \quad (31)$$

2) Implementation: The expectation–maximization algorithm [35] in combination with a finite Gaussian mixture model is used to learn the photometric properties of the foreground and background regions.

The image model described by (26) can be transformed into a level-set-based active-contour formulation by taking the negative logarithm, the variational derivative, and then performing gradient descent. This is equivalent to maximizing (26) to obtain an object segmentation that closely matches the image information and the shape model.

Taking the negative logarithm converts the probabilistic form of (26) into energy $\mathcal{E} = -\ln p(\mathcal{P}|, q^v, I)$ so that

$$\begin{aligned}\mathcal{E} &= -\ln p(I|\mathcal{P}) - \ln p(q^v|\mathcal{P}) - \ln p(\mathcal{P}) + C\\ &= -\int_{\mathbf{x}\in\Omega} (\mathfrak{f}_{\mathbf{x}} \ln p(I_{\mathbf{x}}|\mathfrak{F}) + \mathfrak{b}_{\mathbf{x}} \ln p(I_{\mathbf{x}}|\mathfrak{B})\\ &\qquad - \lambda_s \left(H\cdot\Phi(\mathbf{x}) - H\cdot\Phi_v\cdot\mathcal{A}(\mathbf{x})\right)^2\\ &\qquad -\lambda_k \delta\cdot\Phi(\mathbf{x}))\,|\nabla\Phi(\mathbf{x})| + C)\,dx \qquad (32)\end{aligned}$$

where C is a constant and λ_s and λ_k are weights for the shape and regularization of the length of the contour, respectively. The first two terms can be reformulated in terms of the Heaviside and level-set functions: $H\cdot\Phi(\mathbf{x}) = \mathfrak{f}_{\mathbf{x}}$, and $(1 - H\cdot\Phi(\mathbf{x})) = \mathfrak{b}_{\mathbf{x}}$. Minimization of the energy can then be performed via gradient descent on the variational derivative with respect to the gradient-descent time parameter τ, $\partial\mathcal{E}/\partial\Phi = -(\partial\Phi/\partial\tau)$, i.e.,

$$\begin{aligned}\frac{\partial\Phi}{\partial\tau} &= \delta_0\cdot\Phi(\mathbf{x})\Big(-\ln p(I_{\mathbf{x}}|\mathfrak{F}) + \ln p(I_{\mathbf{x}}|\mathfrak{B})\\ &\qquad +2\lambda_s\left(H\cdot\Phi(\mathbf{x}) - H\cdot\Phi_v\cdot\mathcal{A}(\mathbf{x})\right) + \lambda_k\cdot\mathcal{K}\Big) \qquad (33)\end{aligned}$$

where $\mathcal{K} = \nabla\cdot\nabla\Phi(\mathbf{x})/|\nabla\Phi(\mathbf{x})|$ is the curvature. Finite differencing with a nonoscillatory upwind scheme is used to perform the gradient descent (see, e.g., [36]).

To overcome potential difficulties with the stability of the numerical scheme, the Courant–Friedrichs–Lewy (CFL) condition is used to control the time step in the gradient-descent process. A secondary effect of using the CFL condition is a more robust object tracking framework overall. Weights controlling the relative contribution of the different components in (26) can be set constant across a wider variety of data sets where the system remains stable despite potentially large variations in the object tracking problem, such as variations in image properties or varying deformations of shape.

A slight modification to the CFL condition was introduced to provide improved robustness to variations in initial conditions, i.e., the initialization contour derived from the alpha shapes. This was due to a potentially very different shape prior from the initialization contour. If the shape prior is very different, then the CFL condition will enforce a very small time step to ensure stability. However, a small time step reduces the power of the force generated by the image likelihoods, i.e., $p(I|\mathfrak{F})$ versus $p(I|\mathfrak{B})$, and the contour motion in a single time step becomes very small. This in turn results in the system identifying positive convergence criteria such as constant contour length. Therefore, the CFL condition was modified to enforce a minimum time step.

F. Overall System and Algorithm

The overall system can be summarized by the predict, estimate, update, and forget steps described in Section II-E and in particular by (19), (21)–(24). The system is also summarized here in algorithm form. Initially, the image frames are processed by the bootstrap-stage algorithm shown in Algorithm 1.

Algorithm 1 Bootstrap algorithm

```
Let time $t = 0$ and the first shape detection time $t' = -1$.
loop
    Fetch image frame $I^t$.
    Compute feature points $\mathcal{F}^t$.
    Compute feature point sets $r_b^t$ using spatial clustering.
    Compute regions via alpha shapes over the point sets.
    Determine corresponding regions $r_a^{t-1}$ and $r_b^t$
        using $D(\mathcal{F}^{t-1}_{r_a^{t-1}}, \mathcal{F}^t_{r_b^t})$ in (1).
    Compute foreground shapes using (10) and (11).
    Determine plausible object shapes for inclusion in
        shape memory using $D(\mathfrak{F}^{t-j}, \mathfrak{F}^t)$ from (17).
    if $t' == -1$ && $\#Q == 1$ then
        Set $t' = t$ (the first frame with an extracted shape).
    end if
    Let $t = t + 1$.
end loop
```

Following the bootstrap stage, the online shape-based active contour, which is shown in Algorithm 2, is applied using the generated image masks and the initial shape memory, which is then updated as each frame is processed.

Algorithm 2 Online shape-based active contour.

```
Let $t = t'$ the first shape detection time.
loop
    Fetch image frame $I^t$.
    Initialize level set $\Phi^{t+1,\tau=0}(\mathbf{x})$ using (30).
    while Contour length not constant do
        Evolve the level set with the gradient descent $\partial\Phi/\partial\tau$
            in (33) using Finite Differencing (FD).
        Every 10 FD iterations:
            Update foreground and background
                likelihoods $p(I|\mathfrak{F})$ and $p(I|\mathfrak{B})$.
            Update model shape using $\max_v\{p(q^v|\mathcal{P})\}$
                from (25).
    end while
    Update the shape memory using (19).
    Forget lowest priority shape using (21).
    Let $t = t + 1$.
end loop
```

III. Results and Discussion

A. Shape Extraction Testing

The two shape extraction techniques, i.e., SB–FB and MB–FB, described in Sections II-B-1 and II-B2 were tested using a variety of test images, with some results shown in Fig. 6. For comparison, results are also shown for the region-based active-contour (RAC) technique in [33], except extended to a full probabilistic model of the foregrounds and backgrounds. The probabilistic models are used instead of just the foreground and background mean values, increasing the ability to model complex variations in the foreground or background color distributions. Also shown are the manually created foreground masks used to initialize the three techniques.

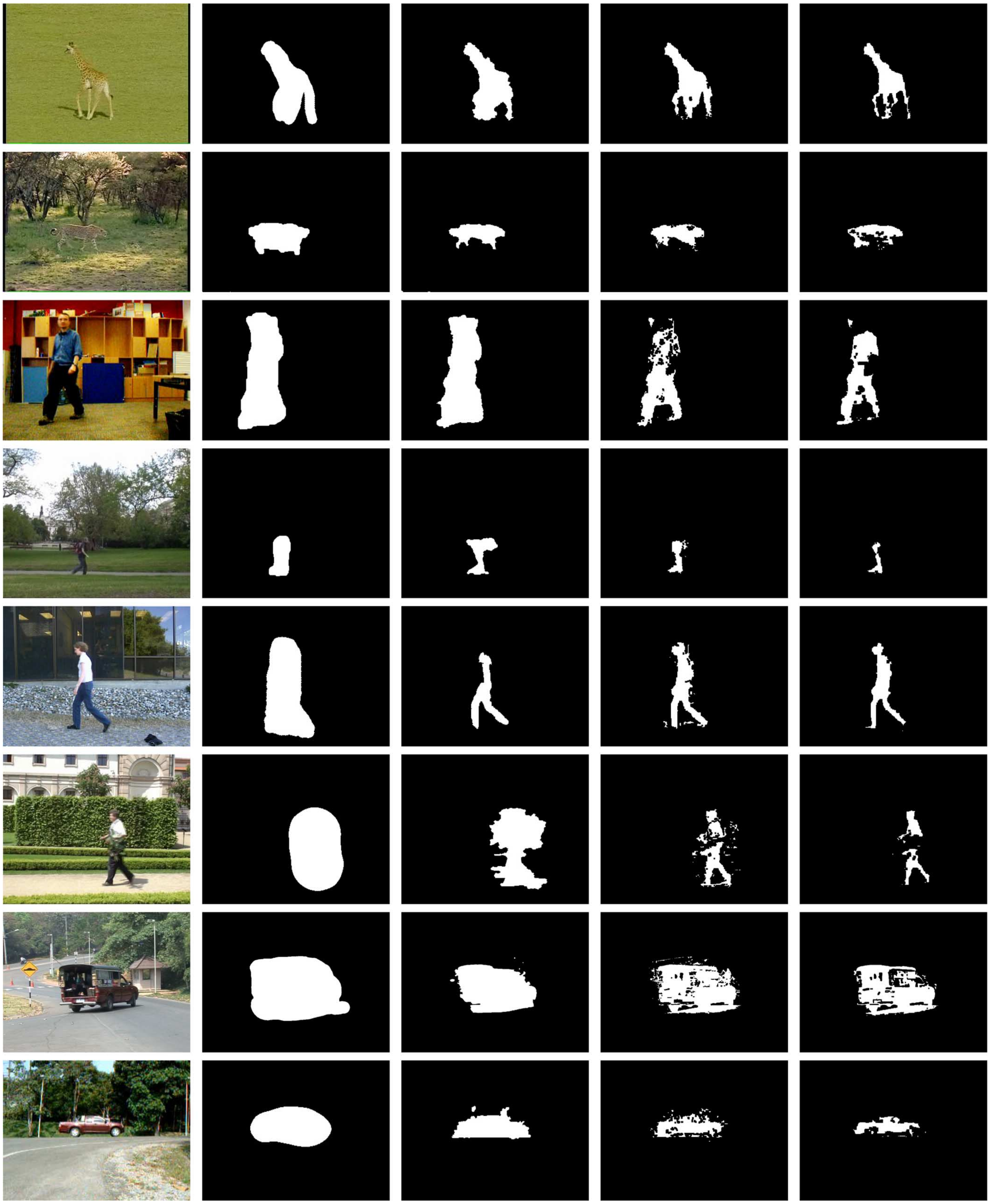


Fig. 6. Shape extraction using three different region-based methods. Column 1: original images. Column 2: the initial foregrounds. Column 3: RAC[33] results. Column 4: SB-FB results. Column 5: MB-FB results. Original data in row 5 is from [37] and rows 4 and 6 from [38].

These results appear to demonstrate that MB–FB generally performs better, at least qualitatively. Quantitatively, SB–FB has the best mean Dice coefficient (77.9%) compared with MB–FB (72.1%) and RAC [33] (77.5%). However, the true positive rate

TABLE I
COMPARISON OF SHAPE EXTRACTION COMPUTATION TIMES FOR STILL IMAGES (ALL COMPUTATION TIMES IN SECONDS)

	SB-FB	**MB-FB**	**RAC[33]**
traffic	0.30	36.72	77.00
person with plant	0.36	63.83	34.43
person in office	0.34	27.30	6.31
cheetah	0.26	15.57	21.38
backpack in park	0.13	12.33	18.43
walking right to left	0.33	44.84	82.51

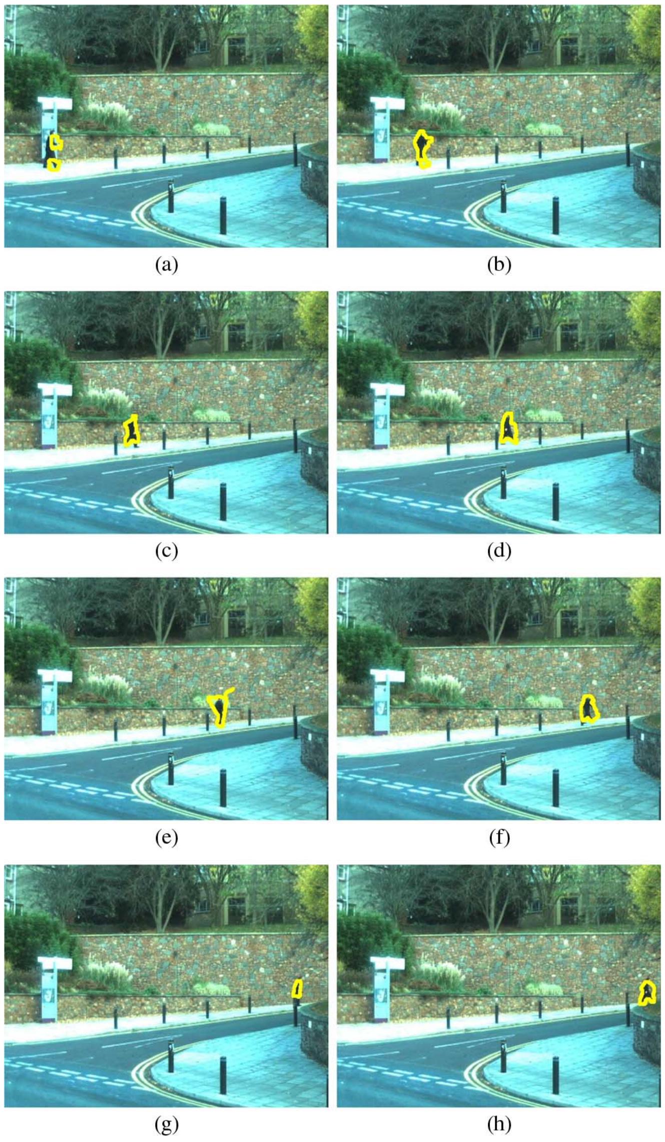


Fig. 7. Results for tracking a walking person. Time instances (frame number) are shown below each frame. (a) $t = 0$. (b) $t = 50$. (c) $t = 100$. (d) $t = 150$. (e) $t = 200$. (f) $t = 250$. (g) $t = 300$. (h) $t = 319$.

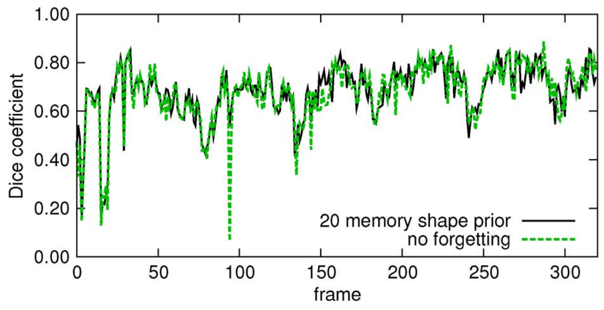


Fig. 8. Performance characterization on the object tracking result shown in Fig. 7 in conparison with object tracking utilizing a full memory (without forgetting).

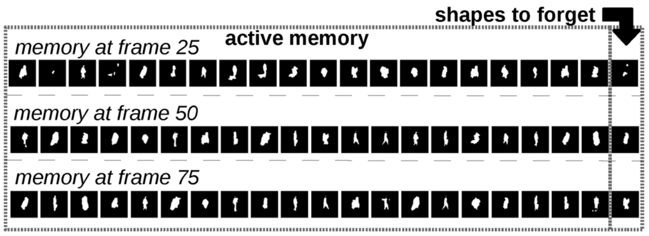


Fig. 9. Contents of memory ($\#Q = 20$) for the sequence shown in Fig. 7. Each row corresponds to the contents of the memory at particular time instances or image frames.

or sensitivity of MB–FB (94.3%) is significantly better than the sensitivity for SB–FB (84.5%) and RAC[33] (72.6%).

Computation times for the static single-image shape extraction processes are shown in Table I. All techniques were implemented in C++ on a notebook running an Intel Dual-Core 1.7-GHz processor with 3-GB random access memory. The computation times for SB–FB are significantly lower in comparison with the other techniques. The (qualitative) accuracy of SB–FB was found to be somewhat inferior for many test images (in relation to MB–FB), but the improvements in the computation times of SB–FB over MB–FB combined with the dynamic nature of the shape memory resulted in SB–FB being used to build the initial shape memory in the experiments that follow.

B. Object Tracking and the Shape Memory

The length of the finite memory $\#Q$ was determined by observing the dominant frequencies in shape similarity matrices such as those shown in Fig. 2. Many shapes are repeatedly observed at least twice over a window of length 20, corresponding to just under a second for a 25-frames-per-second video. Therefore, in the experiments that follow, $\#Q = 20$.

The results for tracking a person walking can be seen in Fig. 7. Performance characterization using the Dice coefficient in combination with the ground truth of the moving object $(\text{Dice mean} = 0.68, \text{ standard deviation} = 0.11)$ is shown in Fig. 8. Also shown is the performance characterization on a tracked object without forgetting any information $(\text{Dice mean} = 0.68, \text{ standard deviation} = 0.12)$, illustrating that the forgetting part proposed here enables the object to be tracked without having to utilize all past object shape configurations.

The contents of the memory for results shown in Fig. 7 at a number of corresponding time instances are shown in Fig. 9. This illustration includes the contents of the active shape memory and the candidate shapes to be forgotten. The shapes to be forgotten are poor representations for the object being tracked. Note that many of the early erroneous shapes shown

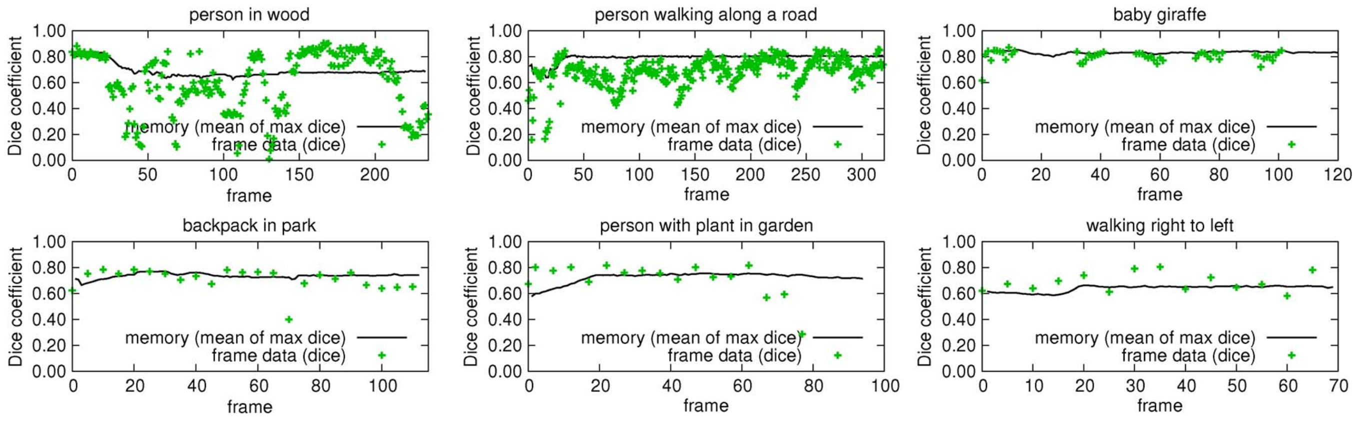


Fig. 10. Attempt to provide an objective assessment of the usefulness of the shape memory for a number of different sequences calculated using the mean of the maximum Dice coefficients (34), where the Dice coefficient is calculated between every ground-truth shape and every shape in the memory at every time instance where $\#Q = 20$. Also shown (data points) are Dice coefficient values for individual frames. These plots help to demonstrate that the shape memory continues to contain shapes that are representative of shapes that will be seen in the actual video data, i.e., the ground truth. The straight lines indicate that the maximum Dice coefficient does not change over time and that the shape memories for each data set could be considered as stable and converged to a similar point for all the data sets shown here.

for the memory at frame 25 are no longer in the shape memory at the later time instances.

a) Shape memory usefulness: An objective assessment of the usefulness of the shape memory is difficult to achieve, as can be seen by the evolving contents of the memory in Fig. 9. However, it is possible to compare the contents of the memory at different time instances with the best match (in terms of shape) and with the ground truth of the tracked object in that video sequence. This is shown in Fig. 10, where the mean of max Dice coefficients is calculated with

$$d^t = \frac{1}{\#Q} \sum_{\forall q \in Q} \max_{\forall g} D(g, q) \tag{34}$$

where g is a ground-truth shape for the particular image sequence being assessed. As can be seen from these plots, the value of d^t appears to converge to an approximately constant Dice coefficient value of between 0.6 and 0.8, i.e., $d^t \rightarrow C$, where $C \in [0.6, 0.8]$ when $t > \#Q$. This appears to suggest that the contents of the shape memory converges to a constant level of apparent usefulness.

Also shown are data points representing Dice coefficient values for corresponding time instances. These data points have quite significant variation in Dice coefficient values, but the contents of the memory remains relatively constant. This demonstrates that the memory contains a consistent representation of shape irrespective of individual variations in object segmentation performance.

C. Occlusion Handling and Other Object tracking Results

An object tracking result for a person walking with an almost complete occlusion can be seen in Fig. 11, where tracking continues successfully through the occlusion. A comparison was made with RAC[33], except that it was extended to object tracking in video by including a Kalman-filtered optical flow tracker (see Section II-E1) (KOFTRAC). Performance characterization was then undertaken with the Dice coefficient. The result of this comparison is shown in Fig. 12. Various other object tracking results can be seen in Fig. 13.

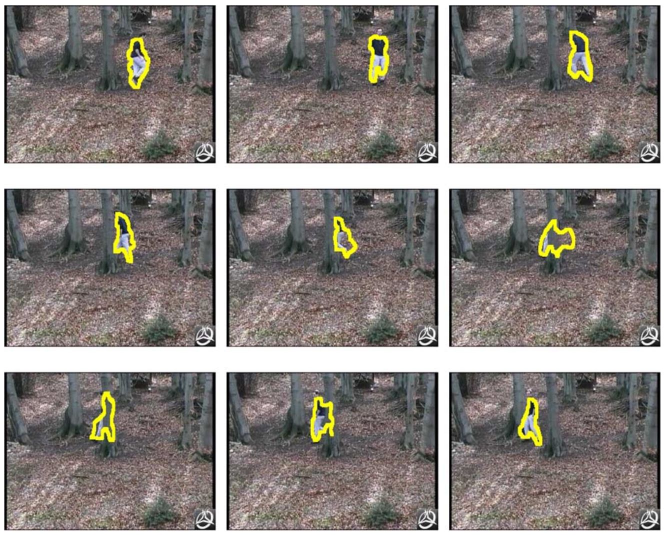


Fig. 11. Tracking result for person walking with a close-to-complete occlusion. Data from [38].

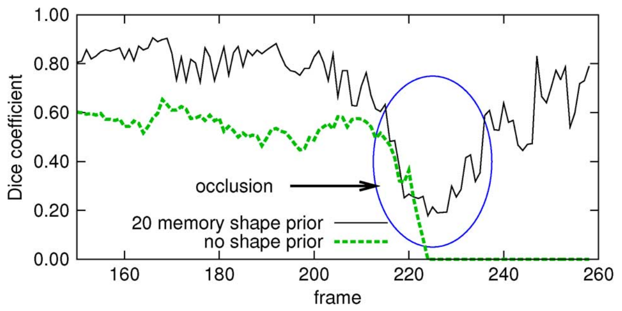


Fig. 12. Performance characterization for the result shown in Fig. 11. Also shown here is a comparison with KOFTRAC demonstrating that the shape prior enables the object tracking to continue despite the tracked object almost disappearing completely from view.

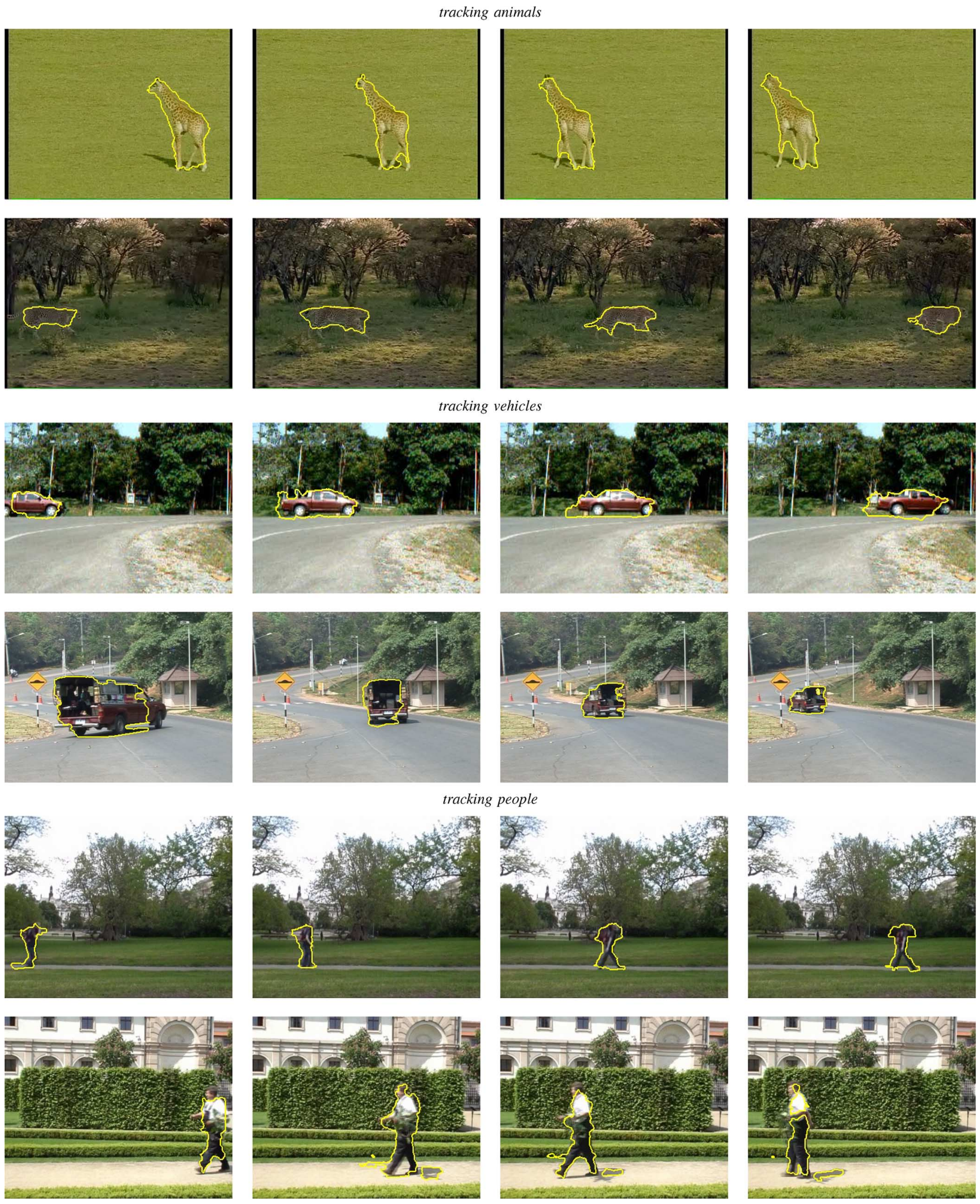


Fig. 13. Various object tracking results.

D. Comparison With Nonadaptive Shape Priors and Various Other Configurations

Active-contour techniques need to use shape priors to improve the accuracy of object segmentation. This is particularly relevant for object tracking, where the tracked object may change significantly in appearance and or shape. However, an object may transform into a shape or a set of shapes that are quite different from the shape priors that are provided via manual supervised training methods (see, e.g., [3], [4], and [32]). In Fig. 14, we show a comparison of our method to when the shape prior is fixed.

The fixed-shape prior contained only a single template, which was a manually defined object shape. This is an oversimplification of many proposed techniques, but it does demonstrate a possibility for when supervised training of the shape information is not sufficient for the range of possible deformations that might occur for some types of objects, e.g., for animals or other

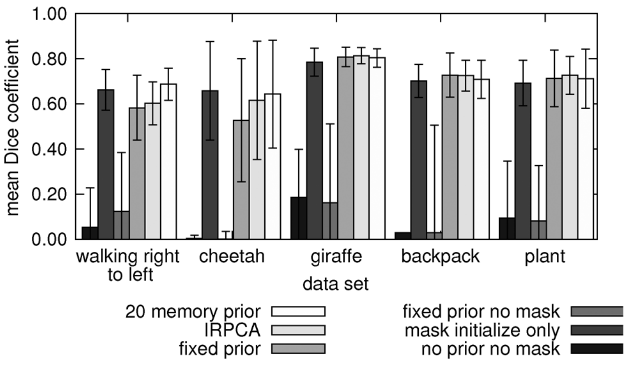


Fig. 14. Comparison of a number of configurations of object segmentation and tracking techniques using the mean of the Dice coefficients for each image frame. Error bars are one standard deviation of the Dice coefficient values. The 20 memory prior is the technique proposed here. IRPCA has been used recently for online learning of shape memory by [10], which provides slightly reduced performance over two of the data sets.

highly deformable objects. The fixed-shape prior provides good object segmentation and tracking performance only when it is combined with the initialization prior at the start of every frame ($\mathfrak{M}$). Furthermore, the use of a manually defined template and an initializing contour mean that it is no longer an automatic technique.

Results for a shape memory based on an incremental and robust PCA (IRPCA) feature space are also shown in Fig. 14. The shape memory based on IRPCA has been used recently by [10]. The results obtained for the IRPCA-based memory appear to be either similar or not as good as the results obtained for our proposed system.

a) Overall robustness of object segmentation and tracking: The overall performance for tracking with and without the finite dimensional shape prior can be assessed, which is also shown in Fig. 14. In particular, the effect of the alpha hulls or ROIs as the initializing prior contour that is combined with the contour from a preceding iteration provide an excellent initialization for the contour for each tracked frame. Also shown are results for the case of no shape priors. These results indicate the importance of good initialization, which is provided by the initializing shape prior. However, the intraframe finite memory shape prior provides improved overall performance.

E. Length of Shape Memory and Alternative Shape Ordering

The effect on the performance of the object tracker regarding the length of the shape memory was also investigated in combination with an alternative approach to prioritizing the shapes. Assessment was undertaken using various sizes of shape memory and then compared with a completely random ordering of the shapes in the shape memory. The tracking system with these different configurations was applied to a number of different video data sets. These tests revealed that the length of shape memory had little effect on the overall performance of the system with slight preference for shape memories with length 20 for a number of the data sets. The alternative ordering of the shapes in the shape memory also had little effect on the overall performance of the system. The exception to this was for the data set with an almost complete occlusion where the completely random ordering of the shape memory failed to retain the important shape information over the number of frames that the occlusion occurred. This can be understood because, normally, the shape memory keeps including reasonably good instances of the shape into the memory during normal tracking for both approaches, but the ordering becomes particularly important when an event such as an occlusion occurs.

F. Computation Times

The same computing platform that was used for the tests shown in Section III-A was also used for all the other experiments shown here. These experiments have shown that the system requires substantial computation time (mean computation time for a number of sequences is 60 seconds per frame). This is mostly due to the requirement to align the shape information and optimization of the active contour.

IV. Discussion and Conclusion

A new fully automatic object segmentation and tracking framework has been proposed consisting of new techniques applicable not only to object tracking in video data but also to static image object segmentation. The results show that the system is capable of automatically detecting and tracking moving objects in video data without any manual intervention. Furthermore, comparison with other existing techniques or configurations of techniques demonstrate similar or superior object tracking performance for the work described here. These developments proposed are not only theoretically interesting as the potential applications for such a system are quite wide, including security and multimedia.

All of the tests performed here utilized the same settings or parameter values, demonstrating fully automatic object segmentation and tracking performance, although manual parameter setting may be necessary for other data. In general, parameters and various other configurations can severely affect the performance of object segmentation and tracking systems. This can be a source of incredible difficulty in developing and/or implementation of these systems. Nevertheless, the contributions described here help toward fully automating the object segmentation and tracking process.

Future work will include finding ways to reduce computation times and to improve the accuracy of the system. Extension of our system to multiple object tracking is also ongoing. Multiple object tracking is an important research topic in the computer vision field, e.g., in [39], and in human visual experiments, e.g., in [40]. Level-set techniques have been proposed that provide convenient and even satisfying formulations for multiple regions, e.g., in [10]. This paper will also necessarily combine feature tracking; all of which will require methods involving accurate correspondence analysis.

References

[1] A. Blake, M. Isard, and D. Reynard, "Learning to track curves in motion," in *Proc. IEEE 33rd Decision Control*, 1994, vol. 4, pp. 3788–3793.

[2] A. Baumberg and D. Hogg, "An efficient method for contour tracking using active shape models," in *Proc. IEEE Workshop Motion Non-Rigid Articulated Objects*, 1994, pp. 194–199.

[3] M. Leventon, W. Grimson, and O. Faugeras, "Statistical shape influence in geodesic active contours," in *Proc. IEEE Int. Conf. Comp. Vis. Pattern Recognit.*, 2000, pp. 316–323.

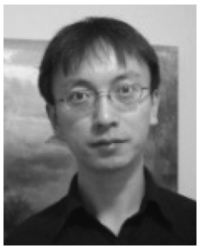
[4] A. Tsai, A. Yezzi, W. Wells, C. Tempany, D. Tucker, A. Fan, W. Grimson, and A. Willsky, "A shape-based approach to the segmentation of medical imagery using level sets," *IEEE Trans. Med. Imag.*, vol. 22, no. 2, pp. 137–154, Feb. 2003.

[5] T. Cootes, D. Cooper, C. Taylor, and J. Graham, "A trainable method of parametric shape description," *Image Vis. Comput.*, vol. 10, no. 5, pp. 289–294, Jun. 1992.
[6] D. Cremers, "Dynamical statistical shape priors for level set based tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 8, pp. 1262–1273, Aug. 2006.
[7] S. Dambreville, Y. Rathi, and A. Tannenbaum, "A framework for image segmentation using shape models and kernel space shape priors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 8, pp. 1385–1399, Aug. 2008.
[8] A. Yilmaz, X. Li, and M. Shah, "Contour-based object tracking with occlusion handling in video acquired using mobile cameras," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 11, pp. 1531–1536, Nov. 2004.
[9] A. Yezzi and S. Soatto, "Deformotion: Deforming motion, shape average and the joint registration and approximation of structures in images," *Int. J. Comput. Vis.*, vol. 53, no. 2, pp. 153–167, 2003.
[10] M. Fussenegger, P. Roth, H. Bischof, R. Deriche, and A. Pinz, "A level set framework using a new incremental, robust active shape model for object segmentation and tracking," *Image Vis. Comput.*, vol. 27, no. 8, pp. 1157–1168, Jul. 2009.
[11] D. Skočaj and A. Leonardis, "Weighted and robust incremental method for subspace learning," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2003, vol. 2, pp. 1494–1501.
[12] J. Chiverton, M. Mirmehdi, and X. Xie, "On-line learning of shape information for object segmentation and tracking," in *Proc. Brit. Mach. Vis. Conf.*, 2009, pp. 1–11.
[13] K. Nummiaro, E. Koller-Meier, and L. Gool, "Object tracking with an adaptive color-based particle filter," in *Proc. 24th DAGM Symp. Pattern Recognit.*, 2002, pp. 353–360.
[14] J. Tu, H. Tao, and T. Huang, "Online updating appearance generative mixture model for meanshift tracking," *Mach. Vis. Appl.*, vol. 20, no. 3, pp. 163–173, Feb. 2009.
[15] P. Pan and D. Schonfeld, "Visual tracking using high-order particle filtering," *IEEE Signal Process. Lett.*, vol. 18, no. 1, pp. 51–54, Jan. 2011.
[16] N. Bouaynaya and D. Schonfeld, "A complete system for head tracking using motion-based particle filter and randomly perturbed active contour," in *Proc. SPIE—IVCP*, San Jose, CA, Mar. 2005, vol. 5685, pp. 864–873.
[17] J. Gai and R. Stevenson, "Studentized dynamical system for robust object tracking," *IEEE Trans. Image Process.*, vol. 20, no. 1, pp. 186–199, Jan. 2011.
[18] D. Ross, J. Lim, R. Lin, and M. Yang, "Incremental learning for robust visual tracking," *Int. J. Comp. Vis.*, vol. 77, no. 1–3, pp. 125–141, May 2008.
[19] F. DeLaTorre and M. Black, "A framework for robust subspace learning," *Int. J. Comput. Vis.*, vol. 54, no. 1–3, pp. 117–142, Aug./Sep. 2003.
[20] S. Smith and J. Brady, "ASSET-2: Real-time motion segmentation and shape tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 8, pp. 814–820, Aug. 1995.
[21] B. McCane, "Snakes and spiders," in *Proc. IEEE Int. Conf. Patt. Recog.*, 2000, vol. 1, pp. 652–655.
[22] V. Gouet and B. Lameyre, "SAP: A robust approach to track objects in video streams with snakes and points," in *Proc. Brit. Mach. Vis. Conf.*, 2004, pp. 737–746.
[23] J. Olszewska, T. Mathes, C. DeVleeschouwer, J. Piater, and B. Macq, "Non-rigid object tracker based on a robust combination of parametric active contour and point distribution model," in *Proc. SPIE—Vis. Commun. Image Process.*, 2007, p. 650 82A.
[24] J. Shi and C. Tomasi, "Good features to track," in *Proc. IEEE Int. Conf. Comp. Vis. Pattern Recognit.*, 1994, pp. 593–600.
[25] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel, "On the shape of a set of points in the plane," *IEEE Trans. Inf. Theory*, vol. IT-29, no. 4, pp. 551–559, Jul. 1983.
[26] M. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981.
[27] M. Irani, B. Rousso, and S. Peleg, "Detecting and tracking multiple moving objects using temporal integration," in *Proc. Eur. Conf. Comput. Vis.*, 1992, pp. 282–287.
[28] S. Hannuna, X. Xie, M. Mirmehdi, and N. Campbell, "Generic motion based object segmentation for assisted navigation," in *Proc. Int. Conf. Comp. VISAPP*, 2009, pp. 450–457.
[29] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 12, pp. 2247–2253, Dec. 2007.
[30] J.-Y. Bouguet, Pyramidal implementation of the Lucas–Kanade feature tracker Microprocessor Res. Labs., Intel Corp., Santa Clara, CA, Tech. Rep., 1999.
[31] C. Bishop, *Neural Networks for Pattern Recognition*. London, U.K.: Oxford Univ. Press, 1995.
[32] N. Paragios, M. Taron, X. Huang, M. Rousson, and D. Metaxas, "On the representation of shapes using implicit functions," in *Statistics and Analysis of Shapes*, H. Krim and A. Yezzi, Eds. Basel, Switzerland: Springer Birkhäuser, 2006, pp. 167–199.
[33] T. Chan and L. Vese, "Active contours without edges," *IEEE Trans. Image Process.*, vol. 10, no. 2, pp. 266–277, Feb. 2001.
[34] D. Cremers, S. Osher, and S. Soatto, "Kernel density estimation and intrinsic alignment for shape priors in level set segmentation," *Int. J. Comput. Vis.*, vol. 69, no. 3, pp. 335–351, Sep. 2006.
[35] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Statist. Soc. B.*, vol. 39, no. 1, pp. 1–38, 1977.
[36] G. Aubert and P. Kornprobst, *Mathematical Problems in Image Processing*. New York: Springer-Verlag, 2006.
[37] H. Sidenbladh and M. Black, "Learning the statistics of people in images and video," *Int. J. Comput. Vis.*, vol. 54, no. 1–3, pp. 181–207, Aug./Sep. 2003.
[38] F. Korč and V. Hlaváč, "Detection and tracking of humans in single view sequences using 2-D articulated model," in *Human Motion*, B. Rodo, R. Klette, and D. Metaxas, Eds. New York: Springer-Verlag, 2007, pp. 105–130.
[39] T. Yang, S. Li, and J. Li, "Real-time multiple objects tracking with occlusion handling in dynamic scenes," in *Proc. IEEE Int. Conf. Comp. Vis. Pattern Recognit.*, 2005, vol. 1, pp. 970–975.
[40] R. St Clair, M. Huff, and A. Seiffert, "Conflicting motion information impairs multiple object tracking," *J. Vis.*, vol. 10, no. 4, pp. 18.1–18.13, Apr. 2010.

John Chiverton received the M.Res. and Ph.D. degrees in computational medical image analysis from the Centre for Vision, Speech and Signal Processing, University of Surrey, Surrey, U.K., in 2002 and 2006, respectively.

Since 2009, he has been a Lecturer with the School of Information Technology, Mae Fah Luang University, Chiang Rai, Thailand, following a two-year postdoctoral research position with the Department of Computer Science, University of Bristol Bristol, U.K. His research interests include computer vision and computational medical imaging topics.

Dr. Chiverton is a member of the IET.



Xianghua Xie (S'03–M'06) received the M.Sc. (with commendation) and Ph.D. degrees in computer science from the University of Bristol, Bristol, U.K., in 2002 and 2006, respectively.

He was previously a Research Associate with the Department of Computer Science, University of Bristol. Since 2007, he has been a Lecturer (RCUK Academic Fellow) with the Department of Computer Science, Swansea University, Swansea, U.K. His current research interests are video analysis, texture analysis, image segmentation, surface inspection, deformable models, and human pose estimation and tracking.

Majid Mirmehdi (SM'08) received the B.Sc. (Hons) and Ph.D. degrees in computer science from the City University London, London, U.K., in 1985 and 1991, respectively.

He is currently a Professor of Computer Vision with the Department of Computer Science, University of Bristol, Bristol. He is the author of more than 150 refereed conference and journal publications. His research interests include natural scene analysis and medical imaging.

Dr. Mirmehdi was elected as a Fellow of the International Association for Pattern Recognition in 2010. He is the Editor-in-Chief of IET *Computer Vision journal (from January 2012) and an Associate Editor of the Pattern Analysis and Applications* journal. He is currently a member of the IET and serves on the Executive Committee of the British Machine Vision Association.