Published in final edited form as:
J Multimed. 2007 August ; 2(4): 20–33.

Flux Tensor Constrained Geodesic Active Contours with Sensor Fusion for Persistent Object Tracking

Filiz Bunyak, **Kannappan Palaniappan**, and **Sumit Kumar Nath**
Department of Computer Science, University of Missouri-Columbia, MO 65211-2060, USA, Email: {bunyak,palaniappank}@missouri.edu, naths@ecse.rpi.edu

Gunasekaran Seetharaman
Dept of Electrical and Computer Engineering, Air Force Institute of Technology, OH 45433-7765 USA, Email: guna.seetharaman@afit.edu

Abstract

This paper makes new contributions in motion detection, object segmentation and trajectory estimation to create a successful object tracking system. A new efficient motion detection algorithm referred to as the flux tensor is used to detect moving objects in infrared video without requiring background modeling or contour extraction. The flux tensor-based motion detector when applied to infrared video is more accurate than thresholding ”hot-spots”, and is insensitive to shadows as well as illumination changes in the visible channel. In real world monitoring tasks fusing scene information from multiple sensors and sources is a useful core mechanism to deal with complex scenes, lighting conditions and environmental variables. The object segmentation algorithm uses level set-based geodesic active contour evolution that incorporates the fusion of visible color and infrared edge informations in a novel manner. Touching or overlapping objects are further refined during the segmentation process using an appropriate shape-based model. Multiple object tracking using correspondence graphs is extended to handle groups of objects and occlusion events by Kalman filter-based cluster trajectory analysis and watershed segmentation. The proposed object tracking algorithm was successfully tested on several difficult outdoor multispectral videos from stationary sensors and is not confounded by shadows or illumination variations.

Index Terms

Flux tensor; sensor fusion; object tracking; active contours; level set; infrared images

I. Introduction

In real world monitoring tasks, persistent moving object detection and tracking remains a challenging problem due to complex scenes, lighting conditions, environmental variables (particularly in outdoor settings with weather), clutter, noise, and occlusions. Fusing scene information from multiple sensors and sources is a useful core mechanism to deal with these problems. Recent developments in micro-optics and micro-electomechanical systems (MEMS), VCSELS, tunable RCLEDS (resonant cavity LEDS), and tunable micro-bolometers indicate that hyperspectral imaging will rapidly become as ubiquitous as visible and thermal videos are today [1], [2].

This paper is based on “Geodesic Active Contour Based Fusion of Visible and Infrared Video for Persistent Object Tracking,” by F. Bunyak, K. Palaniappan, S. Nath and G. Seetharaman, which appeared in the Proceedings of the 8th IEEE Workshop on Applications of Computer Vision (WACV 2007), Texas, USA, February 2007.

On board lidars and radars have been used successfully in unmanned autonomous vehicles, extending their versatility well beyond what was demonstrated in the 1990's based on dynamic scene analysis of visible video only. Most of the autonomous vehicles competing in the recent DARPA Grand Challenge events used one or more lidar sensors to augment the video imagery, demonstrating intelligent navigation using fusion of multiple information sources [3]. Autonomous navigation in city traffic with weather, signals, vehicles, pedestrians, and construction will be more challenging.

Effective performance in persistent tracking of people and objects for navigation, surveillance, or forensic behavior analysis applications require robust capabilities that are scalable to changing environmental conditions and external constraints (ie visibility, camouflage, contraband, security, etc.) [4]. For example, monitoring the barrier around sensitive facilities such as chemical or nuclear plants will require using multiple sensors in addition to a network of (visible) video cameras. Both infrared cameras and laser-scanner based lidar have been used to successfully enhance the overall effectiveness of such systems. In crowds or busy traffic areas even though it may be impractical to monitor and track each person individually, information fusion that characterizes objects of interest can significantly improve throughput. Airport surveillance systems using high resolution infrared/thermal video of people can extract invisible biometric signatures to characterize individuals or tight groups, and use these *short-term* multispectral blob signatures to resolve cluttered regions in difficult video segmentation tasks.

This paper presents a new moving object detection and tracking system for surveillance applications using fusion of visible and infrared information. A preliminary version of the paper has been published in [5]. Infrared imagery is less sensitive to illumination related problems such as uneven lighting, moving cast shadows or sudden illumination changes (i.e. cloud movements) that cause false detections, missed objects, shape deformations, false merges etc. in visible imagery. But use of infrared imagery alone often results in poor performance since generally these sensors produce imagery with low signal-to-noise ratio, uncalibrated white-black polarity changes, and "halo effect" around hot or cold objects [6]. "Hot spot" techniques that detect moving objects by identifying bright regions in infrared imagery are inadequate in the general case, because the assumption that the objects of interest, people and moving cars are much hotter than the surrounding is not always true. The proposed system illustrated in Figure 1 is summarized below. A new efficient *motion detection* algorithm referred to as the flux tensor is used to detect moving objects in infrared video without requiring background modeling or contour extraction. The *object segmentation* algorithm uses level set-based geodesic active contour evolution that incorporates the fusion of visible color and infrared edge information. Touching or overlapping objects are further refined during the segmentation process using an appropriate shape-based model. *Multiple object tracking* module resolves frame-to-frame object correspondences, *cluster trajectory analysis* extension handles groups of objects and occlusion events.

This paper is organized as follows. In Section II motion detection using a flux tensor framework is explored. In Section III motion constrained object segmentation using geodesic active contours is discussed. In Section IV edge based video sensor fusion within a level set based active contour framework is explored. In Section V shape-based refinement of the object masks is presented. In Section VI multiple object tracking using correspondence graphs and its extention to handle groups of objects and occlusion events are described. Section VII presents the results, Section VIII offers the concluding remarks.

II. Motion Detection Using a Flux Tensor Framework

Motion blob detection is performed using our novel *flux tensor* method which is an extension to 3D grayscale structure tensor. Both the grayscale structure tensor and the proposed flux tensor use spatio-temporal consistency more efficiently, thus produce less noisy and more spatially coherent motion segmentation results compared to classical optical flow methods [7]. The flux tensor is more efficient in comparison to the 3D grayscale structure tensor since motion information is more directly incorporated in the flux calculation which is less expensive than computing eigenvalue decompositions as with the 3D grayscale structure tensor.

A. 3D Structure Tensors

Structure tensors are a matrix representation of partial derivative information. As they allow both orientation estimation and image structure analysis they have many applications in image processing and computer vision. 2D structure tensors have been widely used in edge/corner detection and texture analysis, 3D structure tensors have been used in low-level motion estimation and segmentation [7], [8].

Under the constant illumination model, the optic-flow (OF) equation of a spatiotemporal image volume $\mathbf{I}(\mathbf{x})$ centered at location $\mathbf{x} = [x, y, t]$ is given by Eq. 1 [9] where, $\mathbf{v}(\mathbf{x}) = [v_x, v_y, v_t]$ is the optic-flow vector at $\mathbf{x}$,

$$\frac{d\mathbf{I}(\mathbf{x})}{dt} = \frac{\partial \mathbf{I}(\mathbf{x})}{\partial x} v_x + \frac{\partial \mathbf{I}(x)}{\partial y} v_y + \frac{\partial \mathbf{I}(x)}{\partial t} v_t = \nabla \mathbf{I}^T(\mathbf{x}) \mathbf{v}(\mathbf{x}) = 0 \tag{1}$$

and $\mathbf{v}(\mathbf{x})$ is estimated by minimizing Eq. 1 over a local 3D image patch $\mathbf{\Omega}(\mathbf{x}, \mathbf{y})$, centered at $\mathbf{x}$. Note that v_t is not 1 since spatio-temporal orientation vectors will be computed. Using Lagrange multipliers, a corresponding error functional $e_{ls}(\mathbf{x})$ to minimize Eq. 1 using a least-squares error measure can be written as Eq. 2 where $W(\mathbf{x}, \mathbf{y})$ is a *spatially invariant* weighting function (e.g., Gaussian) that emphasizes the image gradients near the central pixel [8].

$$e_{ls}(\mathbf{x}) = \int_{\Omega(\mathbf{x},\mathbf{y})} \left(\nabla \mathbf{I}^T(\mathbf{y})\, \mathbf{v}(\mathbf{x})\right)^2 W(\mathbf{x},\mathbf{y}) d\mathbf{y} + \lambda \left(1 - \mathbf{v}(\mathbf{x})^T \mathbf{v}(\mathbf{x})\right) \tag{2}$$

Assuming a constant $\mathbf{v}(\mathbf{x})$ within the neighborhood $\mathbf{\Omega}(\mathbf{x}, \mathbf{y})$ and differentiating $e_{ls}(\mathbf{x})$ to find the minimum, leads to the standard eigenvalue problem (Eq. 3) for solving $\hat{\mathbf{v}}(\mathbf{x})$ the best estimate of $\mathbf{v}(\mathbf{x})$,

$$\mathbf{J}(\mathbf{x}, \mathbf{W})\, \hat{\mathbf{v}}(\mathbf{x}) = \lambda \hat{\mathbf{v}}(\mathbf{x}) \tag{3}$$

The 3D structure tensor matrix $\mathbf{J}(\mathbf{x}, \mathbf{W})$ for the spatiotemporal volume centered at $\mathbf{x}$ can be written in expanded matrix form, without the spatial filter $W(\mathbf{x}, \mathbf{y})$ and the positional terms shown for clarity, as Eq. 4.

$$\mathbf{J} = \begin{bmatrix} \int_\Omega \frac{\partial \mathbf{I}}{\partial x}\frac{\partial \mathbf{I}}{\partial x} d\mathbf{y} & \int_\Omega \frac{\partial \mathbf{I}}{\partial x}\frac{\partial \mathbf{I}}{\partial y} d\mathbf{y} & \int_\Omega \frac{\partial \mathbf{I}}{\partial x}\frac{\partial \mathbf{I}}{\partial t} d\mathbf{y} \\ \int_\Omega \frac{\partial \mathbf{I}}{\partial y}\frac{\partial \mathbf{I}}{\partial x} d\mathbf{y} & \int_\Omega \frac{\partial \mathbf{I}}{\partial y}\frac{\partial \mathbf{I}}{\partial y} d\mathbf{y} & \int_\Omega \frac{\partial \mathbf{I}}{\partial y}\frac{\partial \mathbf{I}}{\partial t} d\mathbf{y} \\ \int_\Omega \frac{\partial \mathbf{I}}{\partial t}\frac{\partial \mathbf{I}}{\partial x} d\mathbf{y} & \int_\Omega \frac{\partial \mathbf{I}}{\partial t}\frac{\partial \mathbf{I}}{\partial y} d\mathbf{y} & \int_\Omega \frac{\partial \mathbf{I}}{\partial t}\frac{\partial \mathbf{I}}{\partial t} d\mathbf{y} \end{bmatrix} \tag{4}$$

A typical approach in motion detection is to threshold **trace**(**J**) (Eq. 5); but this results in ambiguities in distinguishing responses arising from stationary versus moving features (e.g., edges and junctions with and without motion), since **trace**(**J**) incorporates total gradient change information but fails to capture the nature of these gradient changes (i.e. spatial only versus temporal).

$$\mathbf{trace}(\mathbf{J}) = \int_{\Omega} \| \nabla I \|^2 d\mathbf{y} \tag{5}$$

To resolve this ambiguity and to classify the video regions experiencing motion, the eigenvalues and the associated eigenvectors of **J** are usually analyzed [10], [11]. However eigenvalue decompositions at every pixel is computationally expensive especially if real time performance is required.

B. Flux Tensors

In order to reliably detect only the moving structures *without* performing expensive eigenvalue decompositions, the concept of the *flux tensor* is proposed. Flux tensor is the temporal variations of the optical flow field within the local 3D spatiotemporal volume.

Computing the second derivative of Eq. 1 with respect to *t*, Eq. 6 is obtained where, $\mathbf{a}(\mathbf{x}) = [a_x, a_y, a_t]$ is the acceleration of the image brightness located at **x**.

$$\begin{aligned} \frac{\partial}{\partial t}\left(\frac{d\mathbf{I}(\mathbf{x})}{dt}\right) &= \frac{\partial^2 \mathbf{I}(\mathbf{x})}{\partial x \partial t} v_x + \frac{\partial^2 \mathbf{I}(\mathbf{x})}{\partial y \partial t} v_y + \frac{\partial^2 \mathbf{I}(\mathbf{x})}{\partial t^2} v_t \\ &+ \frac{\partial \mathbf{I}(\mathbf{x})}{\partial x} a_x + \frac{\partial \mathbf{I}(\mathbf{x})}{\partial y} a_y + \frac{\partial \mathbf{I}(x)}{\partial t} a_t \end{aligned} \tag{6}$$

which can be written in vector notation as,

$$\frac{\partial}{\partial t}(\nabla \mathbf{I}^T(x)\mathbf{v}(\mathbf{x})) = \frac{\partial \nabla \mathbf{I}^T(\mathbf{x})}{\partial t}\mathbf{v}(\mathbf{x}) + \nabla \mathbf{I}^T(\mathbf{x})\,\mathbf{a}(\mathbf{x}) \tag{7}$$

Using the same approach for deriving the classic 3D structure, minimizing Eq. 6 assuming a constant velocity model and subject to the normalization constraint $\|\mathbf{v}(\mathbf{x})\| = 1$ leads to Eq. 8,

$$e_{ls}^F(\mathbf{x}) = \int_{\Omega(\mathbf{x},\mathbf{y})} \left(\frac{\partial(\nabla \mathbf{I}^T(\mathbf{y}))}{\partial t}\mathbf{v}(\mathbf{x}) \right)^2 W(\mathbf{x},\mathbf{y})\, d\mathbf{y} + \lambda\left(1 - \mathbf{v}(\mathbf{x})^T \mathbf{v}(\mathbf{x})\right) \tag{8}$$

Assuming a constant velocity model in the neighborhood **Ω**(**x**, **y**), results in the acceleration experienced by the brightness pattern in the neighborhood **Ω**(**x**, **y**) to be zero at every pixel. As with its 3D structure tensor counterpart **J** in Eq. 4, the 3D flux tensor $\mathbf{J_F}$ using Eq. 8 can be written as

$$\mathbf{J_F}(\mathbf{x},\mathbf{W}) = \int_{\Omega} W(\mathbf{x},\mathbf{y}) \frac{\partial}{\partial t}\nabla \mathbf{I}(\mathbf{x}) \cdot \frac{\partial}{\partial t}\nabla \mathbf{I}^{\mathbf{T}}(\mathbf{x}) d\mathbf{y} \tag{9}$$

and in expanded matrix form as Eq. 10.

$$\mathbf{J_F} = \begin{bmatrix} \int_{\Omega}\left\{\frac{\partial^2 \mathbf{I}}{\partial x \partial t}\right\}^2 d\mathbf{y} & \int_{\Omega}\frac{\partial^2 \mathbf{I}}{\partial x \partial t}\frac{\partial^2 \mathbf{I}}{\partial y \partial t} d\mathbf{y} & \int_{\Omega}\frac{\partial^2 \mathbf{I}}{\partial x \partial t}\frac{\partial^2 \mathbf{I}}{\partial t^2} d\mathbf{y} \\ \int_{\Omega}\frac{\partial^2 \mathbf{I}}{\partial y \partial t}\frac{\partial^2 \mathbf{I}}{\partial x \partial t} d\mathbf{y} & \int_{\Omega}\left\{\frac{\partial^2 \mathbf{I}}{\partial y \partial t}\right\}^2 d\mathbf{y} & \int_{\Omega}\frac{\partial^2 \mathbf{I}}{\partial y \partial t}\frac{\partial^2 \mathbf{I}}{\partial t^2} d\mathbf{y} \\ \int_{\Omega}\frac{\partial^2 \mathbf{I}}{\partial t^2}\frac{\partial^2 \mathbf{I}}{\partial x \partial t} d\mathbf{y} & \int_{\Omega}\frac{\partial^2 \mathbf{I}}{\partial t^2}\frac{\partial^2 \mathbf{I}}{\partial y \partial t} d\mathbf{y} & \int_{\Omega}\left\{\frac{\partial^2 \mathbf{I}}{\partial t^2}\right\}^2 d\mathbf{y} \end{bmatrix} \tag{10}$$

As seen from Eq. 10, the elements of the flux tensor incorporate information about temporal gradient changes which leads to efficient discrimination between stationary and moving image features. Thus the trace of the flux tensor matrix which can be compactly written and computed as,

$$\mathbf{trace}(\mathbf{J_F}) = \int_{\Omega} \| \frac{\partial}{\partial t}\nabla \mathbf{I} \|^2 d\mathbf{y} \tag{11}$$

and can be directly used to classify moving and non-moving regions without the need for expensive eigenvalue decompositions. If motion vectors are needed then Eq. 8 can be minimized to get $\hat{\mathbf{v}}(\mathbf{x})$ using

$$\mathbf{J}_{\mathbf{F}}(\mathbf{x},\mathbf{W})\,\widehat{\mathbf{v}}(\mathbf{x})=\lambda\,\widehat{\mathbf{v}}(\mathbf{x}) \tag{12}$$

In this approach the eigenvectors need to be calculated at just moving feature points.

C. Flux Tensor Implementation

To detect motion blobs, only the trace of flux tensor $\mathbf{trace}(\mathbf{J}_{\mathbf{F}})=\int_{\Omega(\mathbf{y})}\|\frac{\partial}{\partial t}\nabla\mathbf{I}\|^2 d\mathbf{y}$ needs to be computed. That requires computation of I_{xt}, I_{yt} and I_{tt} and the integration of squares of I_{xt}, I_{yt}, I_{tt} over the area $\mathbf{\Omega}(\mathbf{y})$. The following notation is adopted for simplicity:

$$I_{xt}=\frac{\partial^2 I}{\partial x\partial t},\quad I_{yt}=\frac{\partial^2 I}{\partial y\partial t},\quad I_{tt}=\frac{\partial^2 I}{\partial t\partial t} \tag{13}$$

The calculation of the derivatives is implemented as convolutions with a filter kernel. By using separable filters, the convolutions are decomposed into a cascade of 1D convolutions. For numerical stability as well as noise reduction, a smoothing filter is applied to the dimensions that are not convolved with a derivative filter e.g. calculation of I_{xt} requires smoothing in y-direction, and calculation of I_{yt} requires smoothing in x-direction. I_{tt} is the second derivative in temporal direction; the smoothing is applied in both spatial directions. As smoothing and derivative filters, optimized filter sets presented by Scharr et. al. in [12], [13] are used.

The integration is also implemented as an averaging filter decomposed into three 1D filters. As a result, calculation of trace at each pixel location requires three 1D convolutions for derivatives and three 1D convolutions for averages in the corresponding spatio-temporal cubes.

A brute-force implementation where spatial and temporal filters are applied for each pixel separately within a spatio-temporal neighborhood would be computationally very expensive since it would have to recalculate the convolutions for neighboring pixels. For an efficient implementation, the spatial (x and y) convolutions are separated from the temporal convolutions, and the 1D convolutions are applied to the whole frames one at a time. This minimizes the redundancy of computations and allows reuse of intermediate results.

The spatial convolutions required to calculate I_{xt}, I_{yt} and I_{tt} are I_{xs}, I_{sy} and I_{ss} where s represents the smoothing filter. Each frame of the input sequence is first convolved with two 1D filters, either a derivative filter in one direction and a smoothing filter in the other direction, or a smoothing filter in both directions. These intermediate results are stored as frames to be used in temporal convolutions, and pointers to these frames are stored in a First In First Out (FIFO) buffer of size $n_{FIFO} = n_{Dt} + n_{At} - 1$ where n_{Dt} is the length of the temporal derivative filter and n_{At} is the length of the temporal averaging filter. For each input frame, three frames I_{xs}, I_{sy} and I_{ss} are calculated and stored. Once n_{Dt} frames are processed and stored, FIFO has enough frames for the calculation of the temporal derivatives I_{xt}, I_{yt} and I_{tt}. Since averaging is distributive over addition, $I_{xt}^2+I_{yt}^2+I_{tt}^2$ is computed first and spatial averaging is applied to this result and stored in the FIFO structure to be used in the temporal part of averaging. Once flux tensor trace of n_{At} frames are computed, temporal averaging is applied. Motion mask FG_M is obtained by thresholding and post-processing averaged flux tensor trace. Post-processing include morphological operations to join fragmented objects and to fill holes.

III. Motion Constrained Object Segmentation Using Geodesic Active Contours

Motion blob detection produces a coarse motion mask FG_M as described in Section II-B. Each pixel of an infrared image frame $\mathbf{I}_{IR}(\boldsymbol{x}, t)$ is classified as moving or stationary by thresholding trace of the corresponding flux tensor matrix $\mathbf{trace}(\mathbf{J_F})$ and a motion blob mask $FG_M(t)$ is obtained.

Two problems with motion blob detection are: (1) holes: motion detection produces holes inside slow moving homogeneous objects, because of the aperture problem. (2) inaccurate object boundaries: motion blobs are larger than the corresponding moving objects, because these regions actually correspond to the union of the moving object locations in the temporal window, rather than the region occupied in the current frame. Beside inaccurate object boundaries this may lead to merging of neighboring object masks and consequently to false trajectory merges and splits at the tracking stage.

Motion constrained object segmentation module refines the coarse $FG_M(t)$ obtained through flux tensors by using fusion of multi-spectral image information and motion information, in a level set based geodesic active contours framework. Using mathematical morphology, holes in the motion blobs (disjoint foreground regions in FG_M) are filled. Geodesic active contours are started from the motion blob boundaries and evolved toward moving object boundaries defined by fusion of visible and infrared edges.

Contour evolution results in tighter object contours and separates most of the merged neighboring objects. Obtained object masks are further refined using shape information. Since the geodesic active contour segmentation relies on edges between background and foreground rather than the color or intensity differences such as in [14], the method is more stable and robust across very different appearances, non-homogeneous backgrounds and foregrounds. Starting the active contour evolution from the motion segmentation results prevents early stopping of the contour on local non-foreground edges.

The motion constrained object segmentation process is summarized in Algorithm 1, level set based geodesic active contours, computation of edge indicator functions, shape-based segmentation refinement process are elaborated in the Sections III-A, IV, and V respectively.

A. Level Set Based Geodesic Active Contours

Active contours evolve/deform a curve $\mathscr{C}$, subject to constraints from a given image. Active contours are classified as parametric [15], [16] or geometric [17]–[19] according to their representation. Parametric active contours (i.e. classical snakes) are represented explicitly as parametrized curves in a Lagrangian formulation, geometric active contours are implicitly represented as level sets [20] and evolve according to an Eulerian formulation [21]. Main advantages of level set based active contours over parametric active contours are computational simplicity and topological flexibility.

In this module level set based geodesic active contours [19] are used to refine motion blobs obtained using flux tensor method (Section II-B). These contours are topologically flexible and are effectively tuned to edge/contour information. Topological flexibility is critical in order to recover individual objects, because during the coarse motion segmentation, neighboring objects may have been merged into a single motion blob.

In level set based active contour methods the curve $\mathscr{C}$ is represented implicitly via a Lipschitz function φ by $C = \{(x,y)|\varphi(x,y)=0\}$, and the evolution of the curve is given by the zero-level

curve of the function $\varphi(t, x, y)$. Evolving $\mathscr{C}$ in a normal direction with speed F amounts to solving the differential equation [14],

$$\frac{\partial \varphi}{\partial t} = |\nabla \varphi| F; \quad \varphi(0,x,y) = \varphi_0(x,y) \tag{14}$$

In [17]–[19] geodesic active contour evolution is formulated as Eq. 15

$$\frac{\partial \varphi}{\partial t} = g_F(\mathbf{I})(c + \mathcal{K}(\varphi))|\nabla \varphi| + \nabla \varphi \cdot \nabla g_F(\mathbf{I}) \tag{15}$$

where $g_F(\mathbf{I})$ is the fused edge stopping function (Eq. 27), c is a constant, and $\mathscr{K}$ is the curvature term,

$$\mathcal{K} = div\left(\frac{\nabla \varphi}{|\nabla \varphi|}\right) = \frac{\varphi_{xx}\varphi_y^2 - 2\varphi_x\varphi_y\varphi_{xy} + \varphi_{yy}\varphi_x^2}{(\varphi_x^2 + \varphi_y^2)^{\frac{3}{2}}} \tag{16}$$

The force $(c+\mathcal{K})$ acts as the internal force in the classical energy based snake model. The constant velocity c pushes the curve inwards or outwards depending on its sign (inwards in our case). The regularization term $\mathscr{K}$ ensures boundary smoothness. The external image dependent force g_F (**I**) (Section IV) is the fused edge indicator function and is used to stop the curve evolution at visible or infrared object boundaries. The term $\nabla g_F \cdot \nabla \varphi$ introduced in [19] is used to increase the basin of attraction for evolving the curve to the boundaries of the objects and to pull back the contour if it passes the boundary [21].

The level set function φ is initialized with the signed distance function of the motion blob contours (FG_M) and evolved using the geodesic active contour speed function Eq. 15 with edge stopping function g_F (**I**) which fuses information from visible and infrared imageries. Formulation of edge stopping function and fusion of information from visible and infrared imageries are elaborated in the next section.

IV. Video Sensor Fusion Combining Visible and Infrared Edge Information

Contour feature or edge indicator functions are used to guide and stop the evolution of the geodesic active contour when it arrives at the object boundaries. The edge indicator function is a decreasing function of the image gradient that rapidly goes to zero along edges and is higher elsewhere.

The magnitude of the gradient of the infrared image is used to construct an edge indicator function g_{IR} as shown below where $G_\sigma(x, y) * \mathbf{I}_{IR}(x, y)$ is the infrared image smoothed with a Gaussian filter,

$$g_{IR}(\mathbf{I}_{IR}) = exp(-|\nabla G_\sigma(x,y) * \mathbf{I}_{IR}(x,y)|) \tag{17}$$

Although the spatial gradient for single channel images lead to well defined edge operators, edge detection in multi-channel images (i.e. color edge strength) is not straight forward to generalize since gradients in different channels can have inconsistent orientations. In [22], Ruzon and Tomasi classify color edge detection algorithms into three categories based on when the individual channel responses are fused: (1) output fusion methods, (2) multi-dimensional gradient methods, and (3) vector methods. In output fusion methods gray-scale edge detection is carried out in each channel independently then combined using methods such as weighted sum. Multi-dimensional gradient methods estimate a single estimate of the orientation and strength of an edge at a point. In vector methods, color information is treated as a vector through

the whole process of edge detection such as in the case of edge detection based on vector order statistics [23], [24].

1) Tensor-based Color Edge Indicator Functions

Color edge indicator functions in this work are based on multi-dimensional gradient methods. The main issue in multi-dimensional gradient methods is the combination of the individual channel gradients into a final multi-dimensional gradient. Simple methods use operations such as sum, weighted sum, max, min etc. to produce the final multi-dimensional/color gradients. But the summation of the individual channel gradients discard the correlation between the channels and may result in cancellation effects [25]. Pioneering work on how to combine the gradients of each channel is done by DiZenzo [26] who considered the multi-channel image as a vector field and computed the tensor gradient. Tensor based methods have since been used in various color feature detection algorithms. And many paper study and extend scale and affine invariance [27], [28] or photometric invariance [29] [30] of these features. More information on color features and multi-channel gradients can be found in [25] [22] [31]–[33]. Two types of tensors particularly important for multi-dimensional/color gradients are 2D color structure tensor and Beltrami color metric tensor explored below.

2D Color Structure Tensor

The 2D color structure tensor is defined in Eq. 18. Many color feature detectors [34]–[36] can be related to the eigenvalues of the color structure tensor matrix $\mathbf{J_C}$ (Eq. 19). Since these eigenvalues are correlated with the local image properties of edgeness and cornerness i.e. $\lambda_1 \gg 0$, $\lambda_2 \approx 0$ and $\lambda_1 \approx \lambda_2 \gg 0$ respectively. Some local descriptors that can be obtained from the two eigenvalues Eq. 19 derived from the 2D color structure tensor are summarized in Table I [25].

$$\mathbf{J_C} = \begin{bmatrix} \sum_{i=R,G,B} \left(\frac{\partial \mathbf{I}_i}{\partial x}\right)^2 & \sum_{i=R,G,B} \frac{\partial \mathbf{I}_i}{\partial x}\frac{\partial \mathbf{I}_i}{\partial y} \\ \sum_{i=R,G,B} \frac{\partial \mathbf{I}_i}{\partial x}\frac{\partial \mathbf{I}_i}{\partial y} & \sum_{i=R,G,B} \left(\frac{\partial \mathbf{I}_i}{\partial y}\right)^2 \end{bmatrix} \tag{18}$$

$$\lambda_{1,2} = \frac{1}{2}(J_C(1,1)+J_C(2,2)) \pm \sqrt{(J_C(1,1)-J_C(2,2))^2+(2J_C(1,2))^2} \tag{19}$$

Beltrami Color Metric Tensor

The Beltrami color metric tensor operator for a 2D color image defines a metric on a two-dimensional manifold $\{x, y, R(x, y), G(x, y), B(x, y)\}$ in the five-dimensional spatial-spectral space $\{x, y, R, G, B\}$. The color metric tensor is defined in Eq. 20 and reformulated as a function of the 2D color structure tensor in Eq. 21 where $\mathcal{I}_2$ is the 2 × 2 identity matrix and $\mathbf{J_C}$ is the 2D color structure tensor. The magnitude of ε can be considered as a generalization of the gradient magnitude, and *det*(ε) can be taken as the edge indicator function [37]–[39].

$$\varepsilon = \begin{bmatrix} 1+\sum_{i=R,G,B} \left(\frac{\partial \mathbf{I}_i}{\partial x}\right)^2 & \sum_{i=R,G,B} \frac{\partial \mathbf{I}_i}{\partial x}\frac{\partial \mathbf{I}_i}{\partial y} \\ \sum_{i=R,G,B} \frac{\partial \mathbf{I}_i}{\partial x}\frac{\partial \mathbf{I}_i}{\partial y} & 1+\sum_{i=R,G,B} \left(\frac{\partial \mathbf{I}_i}{\partial y}\right)^2 \end{bmatrix} \tag{20}$$

$$\varepsilon = \mathcal{I}_2 + \mathbf{J_C} \tag{21}$$

The Beltrami color edge stopping function can then be defined as,

$$g_{RGB}(\mathbf{I}_{RGB}) = exp(-abs(\mathbf{det}(\varepsilon))) \tag{22}$$

$$\begin{aligned}\mathbf{det}(\varepsilon) &= Beltrami(\mathbf{I}_{\mathbf{RGB}}) \\ &= 1+\mathbf{trace}(\mathbf{J}_C)+\det(\mathbf{J}_C) \\ &= 1+(\lambda_1+\lambda_2)+\lambda_1\lambda_2\end{aligned} \tag{23}$$

where λ_1 and λ_2 are the eigenvalues of $\mathbf{J_C}$.

Comparison of Color Feature Detectors

Several common color (edge/corner) feature indicator functions were evaluated for comparison purposes, including Harris [34], Shi-Tomasi [35], Cumani [36] feature detectors and determinant of the Beltrami metric tensor [39].

*The Harris operator (*Eq. 24) [34] uses the parameter *k* to tune edge versus corner responses (i.e. $k \rightarrow 0$ responds primarily to corners).

$$\begin{aligned}Harris(\mathbf{I}_{\mathbf{RGB}}) &= \mathbf{det}(\mathbf{J}_C) - k\,\mathbf{trace}^2(\mathbf{J}_C) \\ &= \lambda_1\lambda_2 - k(\lambda_1+\lambda_2)^2\end{aligned} \tag{24}$$

*The Shi-Tomasi operator (*Eq. 25) [35] responds strongly to corners and filters out most edges (since one of the eigenvalues is nearly zero along edges). This is not suitable for a geodesic active contour edge stopping function.

$$ShiTomasi(\mathbf{I}_{\mathbf{RGB}}) = min(\lambda_1,\lambda_2) \tag{25}$$

*The Cumani operator (*Eq. 26) [36] responds nicely to both edges and corners.

$$Cumani(\mathbf{I}_{\mathbf{RGB}}) = max(\lambda_1,\lambda_2) \tag{26}$$

Although any robust accurate color edge response function can be used, the determinant of the Beltrami color tensor [39] has been selected for our application based on its robustness and speed. The best operators for the geodesic active contour edge stopping functions will respond to all salient contours in the image. In our experiments, as shown in Figure 2, the Beltrami color (edge) features was the most suitable function and is fast to compute. The Harris operator misses some salient contours around the pedestrians, the Shi-Tomasi operator responds primarily to corners and is not suitable as an edge stopping function, the Cumani operator produces a contour map that is nearly the same as the Beltrami color metric tensor map but is slightly more expensive to compute.

2) Fusion of Infrared and Color Edge Indicator Functions

Edge indicator functions obtained from infrared imagery and visible color imagery can be fused using various methods such as weighted sum or tensor-based fusion (i.e. infrared imagery can be considered as a fourth channel and the metric tensor in Eq. 18 can be defined in the six-dimensional spatial-spectral space $\{x, y, R, G, B, IR\}$). The fusion should satisfy two conditions: (1) fused edge stopping function $g_F(x, y)$ should respond to the strongest edge at location (x, y) in either IR or visible imagery, and (2) infrared imagery should have more weight in the final decision than any single channel of the visible imagery, since moving infrared edges are highly salient for tracking. In order to satisfy these conditions and not to miss any infrared edges, independent of the gradients in the visible channels, the *min* statistic Eq. 27 is used as the fusion operator. The fused edge stopping function g_F is defined as the minimum of the two *normalized* (0, 1) edge stopping functions $g_{IR}(x, y)$, and $g_{RGB}(x, y)$,

$$g_F(\mathbf{IR},\mathbf{RGB},x,y)=min\{g_{IR}(x,y),g_{RGB}(x,y)\} \quad (27)$$

This fusion method ensures that the curve evolution stops where there is an edge in the visible imagery or in the infrared imagery. *min* fusion operator handles cases where the visible RGB appearance of the moving object is similar to the background but there is a distinct infrared signature, and when the background and foreground have similar infrared signatures but distinct visible appearances.

V. Shape-based Segmentation Refinement

Evolving geodesic active contours from motion blob boundaries toward object boundaries defined by fusion of visible and infrared edges results in tighter object contours and separates most of the merged neighboring objects. But non-foreground edges such as background edges or shadow/illumination boundaries may result in early stopping of the active contours. Or close neighboring objects may lack non-edge spacing between them where the active contour can move. In such cases geodesic active contours result in under-segmentation of object clusters. Shape-based segmentation refinement breaks clusters in the refined foreground mask FG_R based on shape information. Shape-based refinement relies on an ”ellipse like” shape model for people. Regions where people join are narrower compared to the size of the individual people that form the group/cluster.

The process is illustrated in Figure 3, summarized in Algorithm 2 and elaborated in the following paragraphs.

Each disjoint region R of the refined mask FG_R is eroded and a set of markers (disjoint regions) is obtained. If R is a compact ellipse like region the erosion results in a single marker, no further processing is done and R is returned intact. If R contains narrow regions possibly corresponding to locations where people join, erosion removes those regions and the process results in $n > 1$ marker regions. R is partitioned using a process similar to morphological reconstruction. An n-layer image is allocated and each marker M_i is assigned to a layer. Similar to morphological reconstruction, successive conditional dilations (Eq. 28) are applied to the marker regions M_i’s (dilation is constrained to lie within the initial mask FG_R). While in reconstruction the process is stopped when the dilations cease to change, here the process is stopped when the dilated markers overlap with each other and the overlap regions are identified as partition separators.

$$P_i=(M_i \oplus se_2) \cap FG_R \quad (28)$$

Unlike morphological opening this reconstruction operation separates the objects but does not deteriorate their shapes. To avoid false fragmentation, validity of the partitioning is checked using an heuristic based on maximum, minimum, and relative sizes of the obtained partitions.

VI. Multi-Object Tracking Using Object Correspondence Graphs

Moving object tracking is a fundamental step in the analysis of object behaviors [40]. Persistence in tracking is of particular importance for long term behavior analysis. Moving objects can be tracked implicitly i.e. through contour evolution [41] or explicitly through correspondence analysis. Active contour-based tracking methods track objects by evolving their contours dynamically in successive frames [40]. They provide an effective description of the tracked objects, but are highly sensitive to initialization and unless an additional reinitialization step such (e.g. feature-based re-initialization in [42]) is included to the process, they can not handle large displacements. Therefore, while active contours are used in the object segmentation stage (Section III), the proposed system adopts an explicit region correspondence based tracking algorithm that is able to handle larger object displacements.

The tracking module outlined in Algorithm 3 is an extension of our previous works in [43]–[45]. Object-to-object matching (correspondence) is performed using a multi-stage overlap distance $\mathcal{D}_{MOD}$, which consists of three distinct distance functions for three different ranges of object motion as described in [44].

Correspondence information is arranged in an acyclic directed graph O_{GR} Trajectory-Segments are formed by tracing the links of O_{GR} and grouping "inner" nodes that have a single parent and a single child. For each Trajectory-Segment, parent and children segments are identified and a label is assigned. Segment labels encapsulate connectivity information and are assigned using a method similar to connected component labeling. Events such as appearance, disappearance, split and merge are also identified in this stage based on the number of parent and child segments.

Trajectory segment generation is followed by trajectory segment filtering and cluster trajectory analysis. Trajectory segment filtering prunes spurious trajectory segments based on features such as object size, trajectory length, displacement, and duration, or other spatio-temporal measures [43]. Factors such as under-segmentation, group interactions, partial or full occlusions result in temporary merging of individual object trajectories. Cluster trajectory analysis module resolves these merge-split events where n_p parent trajectory segments TPs, temporarily merge into a single trajectory segment TM, then split into n_c child trajectory segments TCs, and recovers individual object trajectories TS *s*. Currently only symmetric cases where $n_p = n_c$ are considered.

Most occlusion resolution methods rely heavily on appearance. But for far-view video, elaborate appearance-based models cannot be used since objects are small and not enough support is available for such models. Prediction and cluster segmentation is used to recover individual trajectories. Rather than predicting individual trajectories for the merged objects from the parent trajectories alone, at each time step, object clusters are segmented, new measurements are obtained, and object states are updated. This reduces error accumulation particularly for long lasting merges that become more frequent in persistent object tracking.

Segmentation of the object clusters into individual objects is done using a marker-controlled watershed segmentation algorithm applied to the object cluster masks [46], [47]. The use of markers prevents over-segmentation and enables incorporation of segmentation results from the previous frame. The cluster segmentation process is shown in Figure 4 and the overall cluster trajectory analysis process is summarized in Algorithm 4, where $\mathbf{X}$ indicates *state* matrix that consists of a temporal sequence of position and velocity information for each individual object segment, and $\hat{\mathbf{X}}$ indicates estimated *state* matrix.

VII. Results and Analysis

The proposed system is tested on thermal/color video sequence pairs from OTCBVS dataset collection [49]. Data consists of 8-bit grayscale bitmap thermal images, and 24-bit color bitmap images of 320 × 240 pixels. Images were sampled approximately at 30Hz. and registered using homography with manually-selected points. Thermal sequences were captured using a Raytheon PalmIR 250D sensor, color sequences were captured using a Sony TRV87 Handycam.

Figure 5 shows flux tensor trace for sample frames in OTCBVS sequences 3:1 and 3:4. While both IR image sequences particularly sequence 3:1 contain non-moving hot/bright regions such as part of the ground, roof tops, windows, lamp post etc., flux tensor trace successfully identifies only the moving pedestrians. Higher responses are produced for the object boundaries, and inside of the larger objects contain holes due to aperture problem.

Figure 6 shows different moving object detection results. MoG refers to the background estimation and subtraction method by mixture of Gaussians [50] [51]. Flux refers to the flux tensor method presented in Section II. The parameters for the mixture of gaussians (MoG) method are selected as follows: number of distributions $K = 4$, distribution match threshold $T_{match} = 2.0$, background threshold $T = 70\%$, learning rate $\alpha = 0.02$. The parameters for the flux tensor method use a neighborhood size $W = 9$, and trace threshold $T = 4$. Visible imagery (Figure 6c) is very sensitive to moving shadows and illumination changes. Shadows (Figure 6c row 3) can alter object shapes and can result in false detections. Illumination changes due to cloud movements covers a large portion of the ground (Figure 6c row 4) which results in many false moving object detections, making detection and tracking of pedestrians nearly impossible. As can be seen from Figures 6d,e infrared imagery is less sensitive to illumination related problems. But infrared imagery is more noisy compared to visible imagery and suffers from ”halo” effects (Figure 6d). The flux tensor method (Figure 6e) produces less noisy and more compact foreground masks compared to pixel based background subtraction methods such as MoG (Figure 6d), since it integrates temporal information from isotropic spatial neighborhoods.

Figure 7 illustrates two sample cases where the shape-based segmentation refinement splits the pedestrian clusters, under-segmented during active contour evolution. In the first case (top row), the two pedestrians walking side-by-side have no spacing between them where the geodesic active contour can move. In the second case (bottom row), the geodesic active contour stops early because of the background edges.

Figure 8 shows evolution of level set function $\varphi(x, y)$ and the resulting contour during object segmentation. Level set evolution moves the contour from the motion blob boundaries inwards toward actual object boundaries defined by fusion of visible and infrared images. The process results in tighter object boundaries, and individual objects.

Figure 9 shows visible, IR and fused edge indicator functions used in the object segmentation process. Figure 10 illustrates effects of contour refinement and fusion of visible and infrared information. Level set based geodesic active contours refine object boundaries and segment object clusters into individual objects or smaller clusters which is critical for persistent object tracking. When used alone, both visible and infrared video result in total or partial loss of moving objects (i.e. top left person in Figure 10b due to low color contrast compared to background, parts of top right person and legs in Figure 10c due to lack of infrared edges). A low level fusion of the edge indicator functions shown in Figure 10d results in a more complete mask, compared to just combining visible and infrared foreground masks (i.e. legs of top right and bottom persons). Figure 11 illustrates the effects of contour refinement and merge resolution on object trajectories. Level set based geodesic active contours can separate clusters caused by under-segmentation (Figure 11a) but cannot segment individual objects during occlusions (Figure 11b). In those cases merge resolution recovers individual trajectories using prediction and previous object states (Figure 11 second row). In occlusion events no single parameter (i.e. color, size, shape etc.) can consistently resolve ambiguities in partitioning as evident in Figure 11b first row.

VIII. Conclusion

This paper presented a moving object detection and tracking system based on the fusion of infrared and visible imagery for persistent object tracking. Outdoor surveillance applications require robust systems due to wide area coverage, shadows, cloud movements and background activity. The proposed system fuses the information from both visible and infrared imagery within a geodesic active contour framework to achieve this robustness.

A new efficient motion detection algorithm referred to as the flux tensor is used to detect moving objects in infrared video without requiring background modeling or contour extraction. The flux tensor-based motion detector when applied to infrared video is more accurate than thresholding ”hot-spots”, and is insensitive to shadows as well as illumination changes in the visible channel. The novel flux tensor also produces less noisy and more spatially coherent results compared to classical pixel based background subtraction methods. The object segmentation algorithm uses level set-based geodesic active contour evolution that incorporates the fusion of visible color and infrared edge information in a novel manner and refines the initial motion mask. Touching or overlapping objects are further refined during the segmentation process using an appropriate shape-based model.

Multiple object tracking using correspondence graphs is extended to handle groups of objects and occlusion events by Kalman filter-based cluster trajectory analysis and watershed segmentation. The proposed object tracking algorithm was successfully tested on several difficult outdoor multispectral videos from stationary sensors and is not confounded by shadows or illumination variations.

References

1. Zolper, J. Integrated microsystems: A revolution on five frontiers. Proc. of the 24th DARPA-Tech Conf; Anahiem, CA. Aug. 2005;
2. Lemnios Z, Zolper J. Informatics: An opportunity for microelectronics innovation in the next decade. IEEE Circuit & Devices Jan;2006 22(1):16–22.
3. Seetharaman G, Lakhotia A, Blasch E. Unmanned vehicles come of age: The DARPA grand challenge. Special Issue of IEEE Computer Dec;2006 :32–35.
4. Brown W, Kaehr R, Chelette D. Finding and tracking targets: Long term challenges. Air Force Research Technology Horizons 2004;5(1):9–11.
5. Bunyak, F.; Palaniappan, K.; Nath, S.; Seetharaman, G. Geodesic active contour based fusion of visible and infrared video for persistent object tracking. 8th IEEE Workshop on Applications of Computer Vision (WACV 2007); Feb. 2007;
6. Davis J, Sharma V. Background-subtraction in thermal imagery using contour saliency. Int Journal of Computer Vision 2007;71(2):161–181.
7. Nath, S.; Palaniappan, K. Adaptive robust structure tensors for orientation estimation and image segmentation. LNCS-3804: Proc. ISVC’05; Lake Tahoe, Nevada. Dec. 2005; p. 445-453.
8. Nagel, HH.; Gehrke, A. Spatiotemporally adaptive estimation and segmentation of OF-Fields. LNCS-1407: ECCV98; Germany. June 1998; p. 86-102.
9. Horn BP, Schunck BG. Determining optical flow. Artificial Intell Aug;1981 17(1–3):185–203.
10. Palaniappan, K.; Jiang, H.; Baskin, TI. Non-rigid motion estimation using the robust tensor method. IEEE Comp. Vision and Patt. Recog. Workshop on Articulated and Nonrigid Motion; Washington, DC. June 2004;
11. Zhang J, Gao J, Liu W. Image sequence segmentation using 3-D structure tensor and curve evolution. IEEE Trans Circuits Syst Video Technol May;2001 11(5):629–641.
12. Scharr, H. Optimal filters for extended optical flow. LNCS: First Int. Workshop on Complex Motion; Berlin, Germany. Oct. 2004; Springer-Verlag; p. 66-74.
13. Scharr, H.; Stuke, I.; Mota, C.; Barth, E. Estimation of transparent motions with physical models for additional brightness variation. 13th European Signal Processing Conference, EUSIPCO; 2005.
14. Chan T, Vese L. Active contours without edges. IEEE Trans Image Proc Feb;2001 10(2):266–277.
15. Kass M, Witkin A, Terzopoulos D. Snakes: Active contour models. Journal International Journal of Computer Vision January;1988 1(4):321–331.
16. Cohen LD. On active contour models and balloons. Computer Vision, Graphics, and Image Processing Image Understanding 1991;53(2):211–218.
17. Caselles V, Catte F, Coll T, Dibos F. A geometric model for active contours. Numerische Mathematik 1993;66:1–31.

18. Malladi R, Sethian JA, Vemuri B. Shape modelling with front propagation:A level set approach. IEEE Trans Patt Anal Mach Intell 1995;17(2):158–174.
19. Caselles V, Kimmel R, Sapiro G. Geodesic active contours. Int Journal of Computer Vision 1997;22 (1):61–79.
20. Sethian, JA. Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science. Cambridge University Press; Cambridge, UK: 1999.
21. Xu C, Yezzi A, Prince J. A summary of geometric level-set analogues for a general class of parametric active contour and surface models. Proc IEEE Workshop on Variational and Level Set Methods in Computer Vision 2001:104–111.
22. Ruzon MA, Tomasi C. Edge, junction, and corner detection using color distributions. IEEE Trans Patt Anal and Mach Intell 2001;23(11):1281–1295.
23. Trahanias PE, Venetsanopoulos AN. Vector order statistics operators as color edge detectors. IEEE Transactions on Systems, Man, and Cybernetics-PartB: Cybernetics Feb;1996 26(1):135–143.
24. Hai, Tao; Huang, Thomas S. Color image edge detection using cluster analysis. IEEE Int Conf on Image Processing (ICIP'97) 1997:834–836.
25. Stokman, H.; Gevers, T.; Weijer, J. Color feature detection. In: Lukac, R.; Plataniotis, KN., editors. Color Image Processing: Methods and Applications. CRC Press; 2006.
26. Silvano, Di Zenzo. A note on the gradient of a multi-image. Computer Vision, Graphics, and Image Processing 1986;33(1):116–125.
27. Mikolajczyk K, Schmid C. Scale & affine invariant interest point detectors. International Journal on Computer Vision 2004;60(1):63–86.
28. Nicu, Sebe; Gevers, Theo; Dijkstra, S.; van de Weijer, Joost. Evaluation of intensity and color corner detectors for affine invariant salient regions. Beyond Patches Workshop Interaction at CVPR. 2006
29. van de Weijer J, Gevers Th, Geusebroek JM. Edge and corner detection by photometric quasi-invariants. IEEE Trans Pattern Analysis and Machine Intelligence April;2005 27(4)
30. van de Weijer J, Gevers Th, Smeulders AWM. Robust photometric invariant features from the color tensor. IEEE Trans Image Proces 2006;15(1)
31. Mohr C, Schmid R, Bauckhage C. Evaluation of interest point detectors. International Journal of Computer Vision 2000;37(2):151–172.
32. Koschan A, Abidi M. Detection and classification of edges in color images. Signal Processing Magazine, Special Issue on Color Image Processing 2005;22(1):64–73.
33. Naik SK, Murthy CA. Standardization of edge magnitude in color images. IEEE Trans on Image Processing Sep;2006 15(9):2588–2595.
34. Harris, C.; Stephens, M. A combined corner and edge detector. Proc. 4th Alvey Vision Conf; Manchester. 1988. p. 147-151.
35. Jianbo, Shi; Tomasi, Carlo. Good features to track. IEEE Conf. on Comp. Vis. and Patt. Recog. (CVPR); Seattle. June 1994;
36. Cumani A. Edge detection in multispectral images. CVGIP: Graphical Models and Image Processing 1991;53(1)
37. Brook A, Kimmel R, Sochen NA. Variational restoration and edge detection for color images. Journal of Mathematical Imaging and Vision May;2003 18(3):247–268.
38. Sochen NA, Kimmel R, Malladi R. A general framework for low-level vision. IEEE Trans Image Proc 1998;7(3):310–318.
39. Goldenberg R, Kimmel R, Rivlin E, Rudzsky M. Fast geodesic active contours. IEEE Trans Image Proc Oct;2001 10(10):1467–1475.
40. Hu W, Tan TN, Wang L, Maybank SJ. A survey on visual surveillance of object motion and behaviors. IEEE Trans on Systems, Man, and Cybernetics - Part C August;2004 34(3):334–352.
41. Paragios N, Deriche R. Geodesic active regions and level set methods for motion estimation and tracking. Computer Vision and Image Understanding March;2005 97(3):259–282.
42. Knoll A, Panin G. Fully automatic real-time 3d object tracking using active contour and appearance models. Journal of Multimedia 2006 2006;1(7):62–70.

43. Bunyak, F.; Subramanya, SR. Maintaining trajectories of salient objects for robust visual tracking. LNCS-3212: Proc. ICIAR'05; Toronto. Sep. 2005; p. 820-827.

44. Bunyak, F.; Palaniappan, K.; Nath, SK.; Baskin, TI.; Dong, G. Quantitive cell motility for *in vitro* wound healing using level set-based active contour tracking. Proc. 3rd IEEE Int. Symp. Biomed. Imaging (ISBI); Arlington, VA. April 2006; p. 1040-1043.

45. Nath, SK.; Bunyak, F.; Palaniappan, K. Robust tracking of migrating cells using four-color level set segmentation. LNCS-3212: Proc. ACIVS'05; Antwerp, Belgium. Sep. 2006; p. 3212

46. Vincent L, Soille P. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. IEEE Trans Patt Anal Mach Intell 1991;13(6):583–598.

47. Beucher, S.; Meyer, F. The morphological approach to segmentation: the watershed transformation. In: Dougherty, ER., editor. Math Morph and its Applications to Image Proc. Marcel Dekker; NY: 1993. p. 433-481.

48. Bar-Shalom, Y.; Li, XR.; Kirubarajan, T. Estimation with Applications to Tracking and Navigation: Theory, Algorithms, and Software. John Wiley & Sons, Inc; 2001.

49. Davis, J.; Sharma, V. Fusion-based background-subtraction using contour saliency. IEEE Int. Workshop on Object Tracking and Classification Beyond the Visible Spectrum; San Diego, CA. June 2005;

50. Stauffer C, Grimson E. Learning patterns of activity using real-time tracking. IEEE Trans Pattern Anal and Machine Intel 2000;22(8):747–757.

51. Zhuang X, Huang Y, Palaniappan K, Zhao Y. Gaussian mixture density modeling, decomposition and applications. IEEE Trans Image Proc Sep;1996 5(9):1293–1302.

Biographies

Filiz Bunyak received her B.S. and M.S. degrees in control and computer engineering from Istanbul Technical University, Istanbul, Turkey and her Ph.D. degree in computer science from University of Missouri-Rolla, Rolla, MO, USA in 2005. In 2005 she joined University of Missouri-Columbia computer science department as post-doctoral researcher. Her research interests are image processing and computer vision, with emphasis on visual surveillance, video tracking, biomedical image processing, data fusion, segmentation, level set methods, and mathematical morphology.

Kannappan Palaniappan received the B.A.Sc. and M.A.Sc. degrees in systems design engineering from the University of Waterloo, Waterloo, ON, Canada, and the Ph.D. degree in electrical and computer engineering from the University of Illinois, Urbana-Champaign, in 1991. From 1991 to 1996, he was with the NASA Goddard Space Flight Center, working in the Laboratory for Atmospheres, where he coestablished the High-Performance Visualization and Analysis Laboratory. He developed the first massively parallel algorithm for satellite-based hurricane motion tracking using 64 K processors and invented the Interactive Image SpreadSheet system for visualization of extremely large image sequences and numerical model data over highperformance networks. Many visualization products created with colleagues at NASA have been widely used on television, magazines, museums, web sites, etc. With the University of Missouri, he helped establish the NSF vBNS highspeed research network, the NASA Center of Excellence in Remote Sensing, ICREST, and MCVL. He has been with UMI-ACS, University of Maryland, and worked in industry for Bell Northern Research, Bell Canada, Preussen Elektra Germany, the Canadian Ministry of Environment, and Ontario Hydro. His research interests include satellite image analysis, biomedical imaging, video tracking, level set-based segmentation, nonrigid motion analysis, scientific visualization, and content-based image retrieval. Dr. Palaniappan received the highest teaching award given by the University of Missouri, the William T. Kemper Fellowship for Teaching Excellence in 2002, the Boeing Welliver Faculty Fellowship in 2004, the University Space Research Association Creativity and Innovation Science Award (1993), the NASA Outstanding Achievement Award (1993), the Natural Sciences and Engineering Research Council of Canada scholarship (19821988),

and the NASA Public Service Medal (2001) for pioneering contributions to scientific visualization and analysis tools for understanding petabyte-sized archives of NASA datasets.

Sumit Nath received his M.Tech degree from the Indian Institute of Science, Bangalore, India in 1998 and his Phd from the University of Ottawa, Canada, 2004. He joined Dr. K. Palaniappan's group at the University of Missouri-Columbia as a Post-Doc researcher in 2004. Currently, he is a pursuing his post-doc research at the Electrical Computer and Systems Engineering Department at Rensselaer Polytechnic Institute, Troy. He is primarily interested in all aspects of computer vision, image processing and compression, with special emphasis on level sets, segmentation, optic flow estimation, Voronoi diagrams and Tomography.

Guna Seetharaman has been an associate professor of Electrical and Computer Engineering at Air Force Institute of Technology, since June 2003. He was with The Center for Advanced Computer Studies, University of Louisiana at Lafayette, until 2003, as an associate professor of computer engineering. He was a CNRS Research Visiting Professor at the Institute for Electronics Fundamentals, University of Paris XI, Orsay, on sabbatical and short-term fellowships.

Dr. Seetharaman earned his Ph.D. degree in Electrical and Computer Engineering, in 1988 from the University of Miami, Coral Gables, FL. He holds a M.Tech., in Electrical Engineering (1982) from Indian Institute of Technology, Chennai. He earned his B.E., Electronics and Communication Engineering, in 1980 from The University of Madras, India. He established and successfully ran the Computer Vision Laboratory, and Intelligent Robotics Laboratory (co-established) at The University of Louisiana at Lafayette. His research has been funded by NSF, ONR, DOE, AFOSR and The Board of Regents of Louisiana. His recent focus has been on integrated Microsystems for 3D imaging and displays; and high performance embedded computing algorithms for image processing systems. He also participated in the DARPA Grand Challenge as a charter member of Team CajunBot. He has published more than 120 articles in: Computer Vision, low-altitude aerial imagery, SIMD-Parallel Computing, VLSI-signal processing, 3D Displays, Nano-Technology, and 3D Video analysis.

He co-organized the DOE/ONR/NSF Sponsored Second International Workshop on Foundations of Decision and Information Fusion, in 1996 (Washington DC), and the IEEE Sixth International Workshop on Computer Architecture for Machine Perception, New Orleans, 2003. He guest edited IEEE COMPUTER special issue devoted to Unmanned Intelligent Autonomous Vehicles, Dec 2006. He also guest-edited a special issue of the EURASIP Journal on Embedded Computing in the topics of Intelligent Vehicles. He is an active member of the IEEE, and ACM. He is also a member of Tau Beta Pi, Eta Kappa Nu and Upsilon Pi Epsilon.

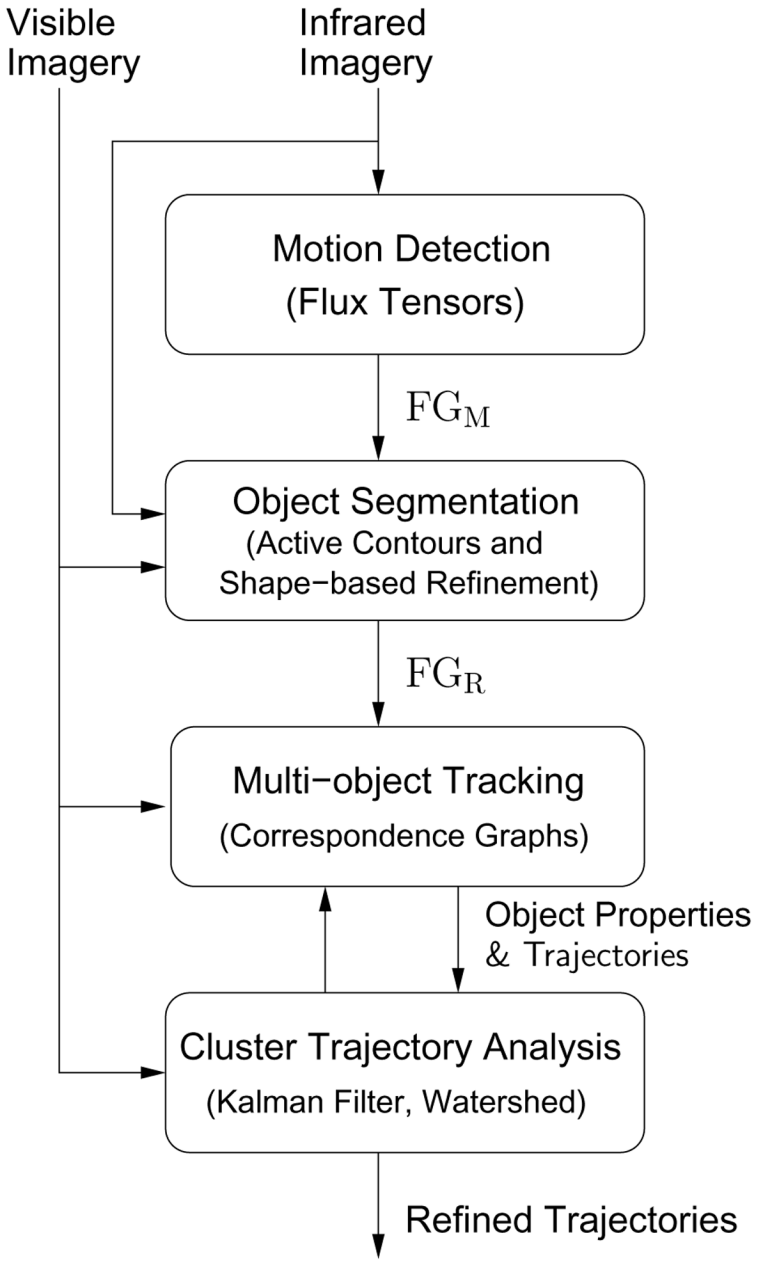


Figure 1.
Multi-spectral data fusion system for persistent object tracking.

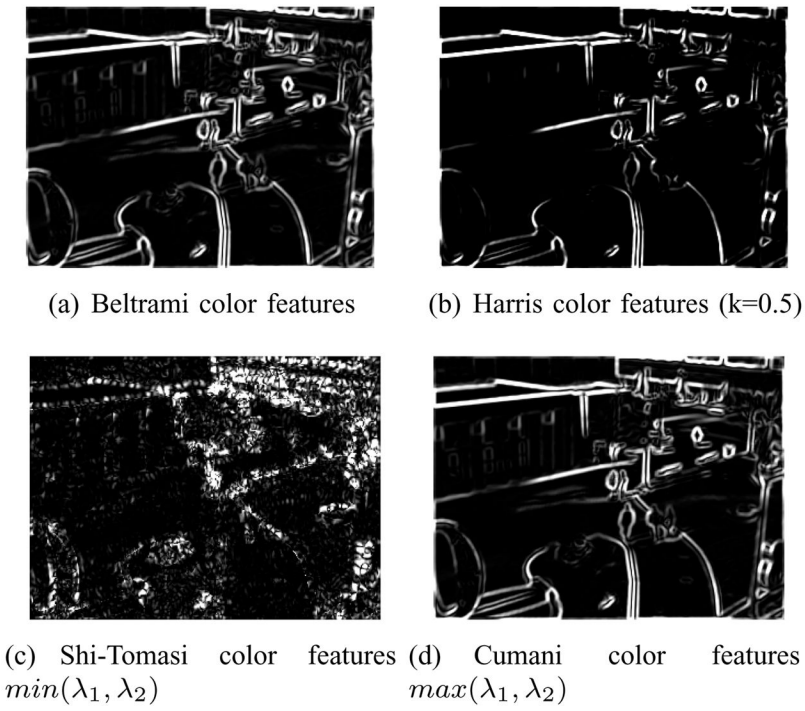


Figure 2.
Color features for frame #1256 obtained using Beltrami, Harris, Shi-Tomasi, and Cumani operators.

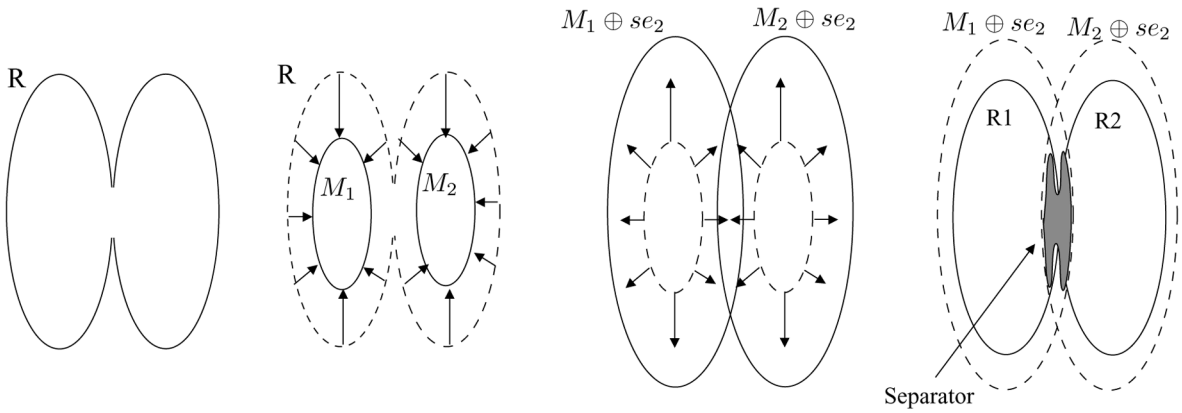


Figure 3.
Shape-based segmentation refinement. R: original region/cluster, $M_{1,2}$: marker regions obtained by eroding R, $M_{1,2} \oplus se_2$: dilated markers, $R_{1,2}$: final segmentation.

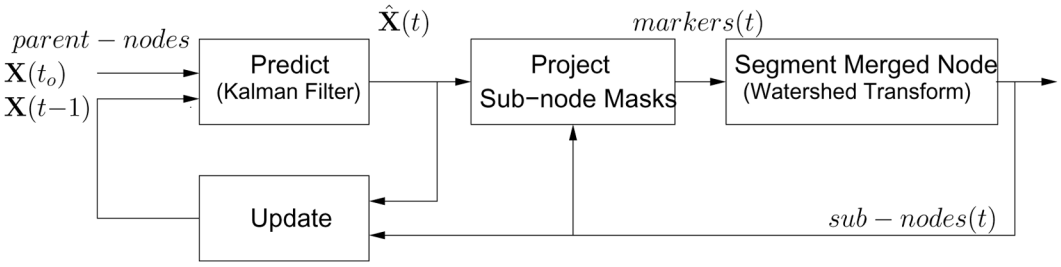


Figure 4.
Cluster segmentation using Kalman filter and watershed segmentation.

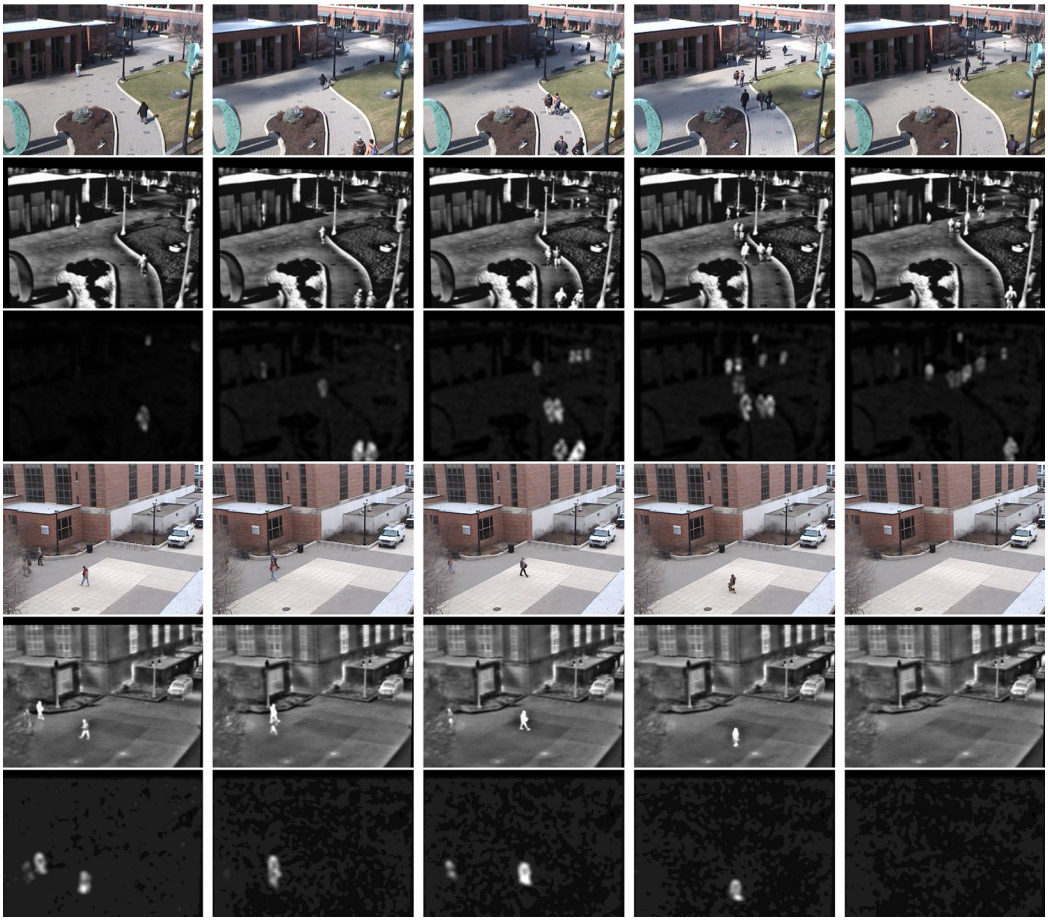


Figure 5.
Flux tensor based motion detection results. Top three rows: visible image, IR image, flux tensor trace for OTCBVS benchmark sequence 3:1 frames #8,#432,#836,#1254,#1740. Flux tensor based motion detection results. Bottom three rows: visible image, IR image, flux tensor trace for OTCBVS benchmark sequence 3:4 frames #20,#124,#294,#1722,#3000. Averaging window size $n_{At} = 7$, derivative filter length $n_{Dt} = 5$. Flux response is scaled for improved print visibility.

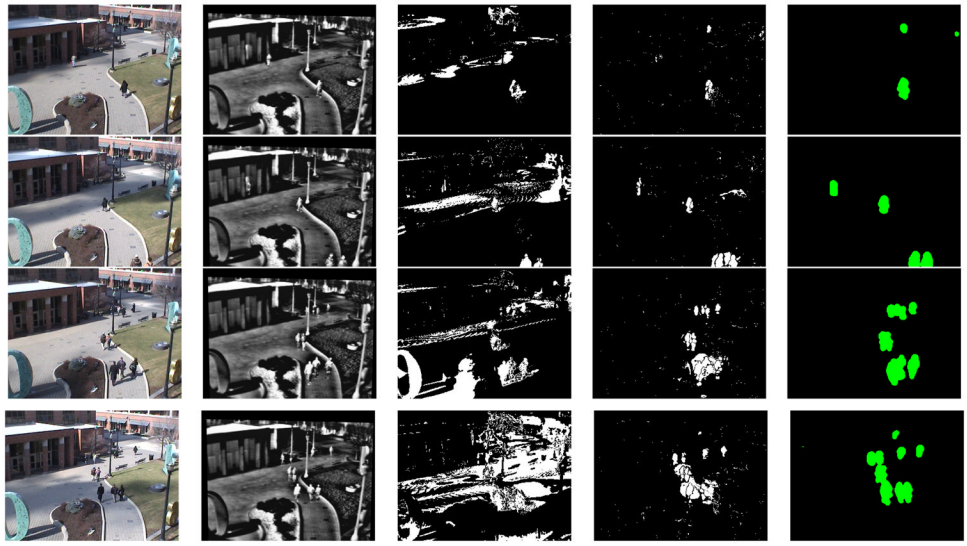


Figure 6.
Moving object detection results for OTCBVS benchmark sequence 3:1. Top to bottom frames #120, #400,#1048,#1256.

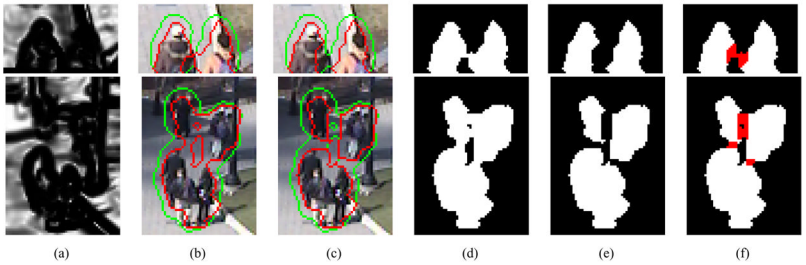


Figure 7.
Shape-based segmentation refinement results for OTCBVS benchmark sequence 3:1 frames #408 (top) and #1528 (bottom). a) edge stopping function, b) contours before shape-based refinement, green: flux tensor results, red: contours refined using active contours, c) contours after shape-based refinement, green: flux tensor results, red: contours refined using active contour evolution + shape-based refinement, d) mask after active contour evolution, e) mask after active contour evolution + shape based refinement, f) partition separators.

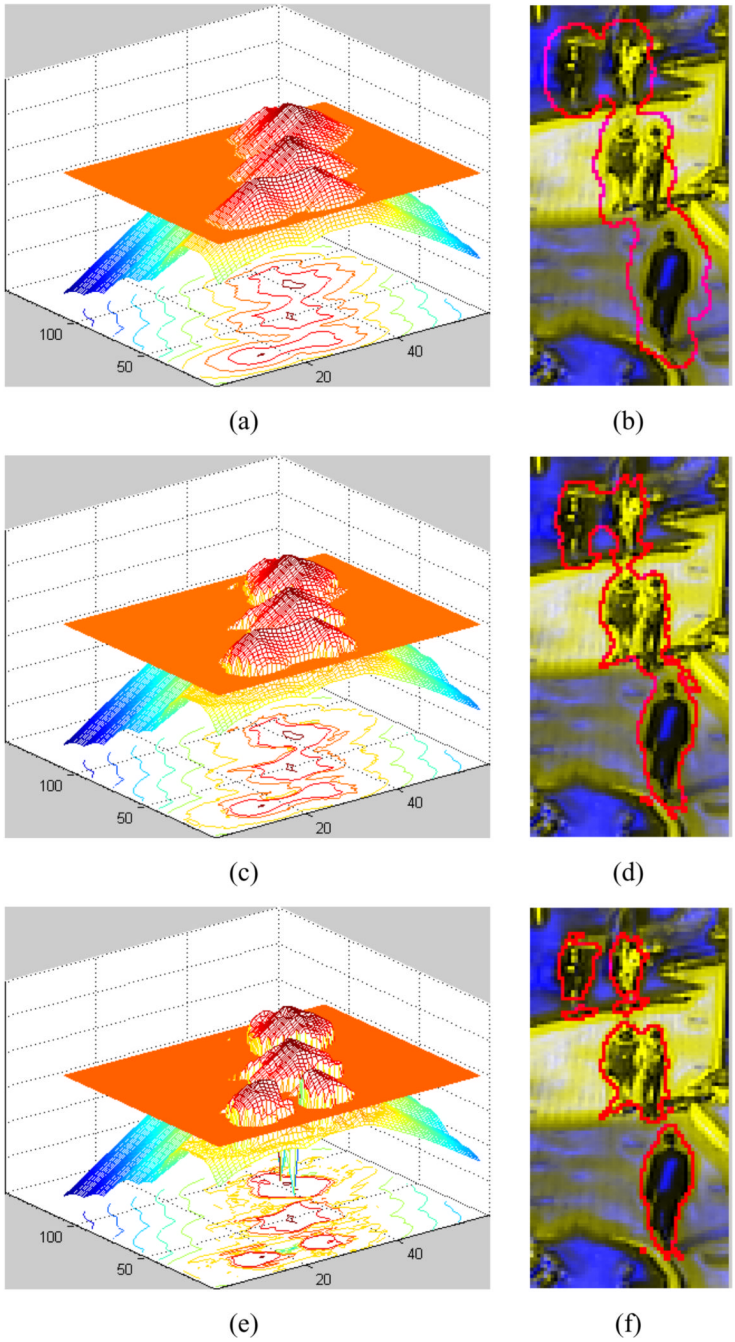
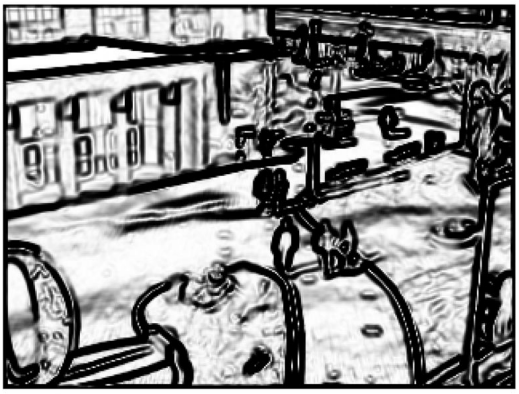


Figure 8.
Level set based active contour evolution for OTCBVS benchmark sequence 3:1 frame #1256. Left: Level set function $\varphi(x, t)$, horizontal plane is zero level. Right: Segmentation result superimposed on the original image. Red and Green channels: original image, Blue channel: fused edge stopping function $g_F(x, y)$, Red contour: level set contour ($\varphi(x, y) = 0$).



(a) g_{RGB}

(b) g_{IR}

(c) g_{Fusion}

Figure 9.
Edge indicator functions for OTCBVS benchmark sequence 3:1 frame #1256.

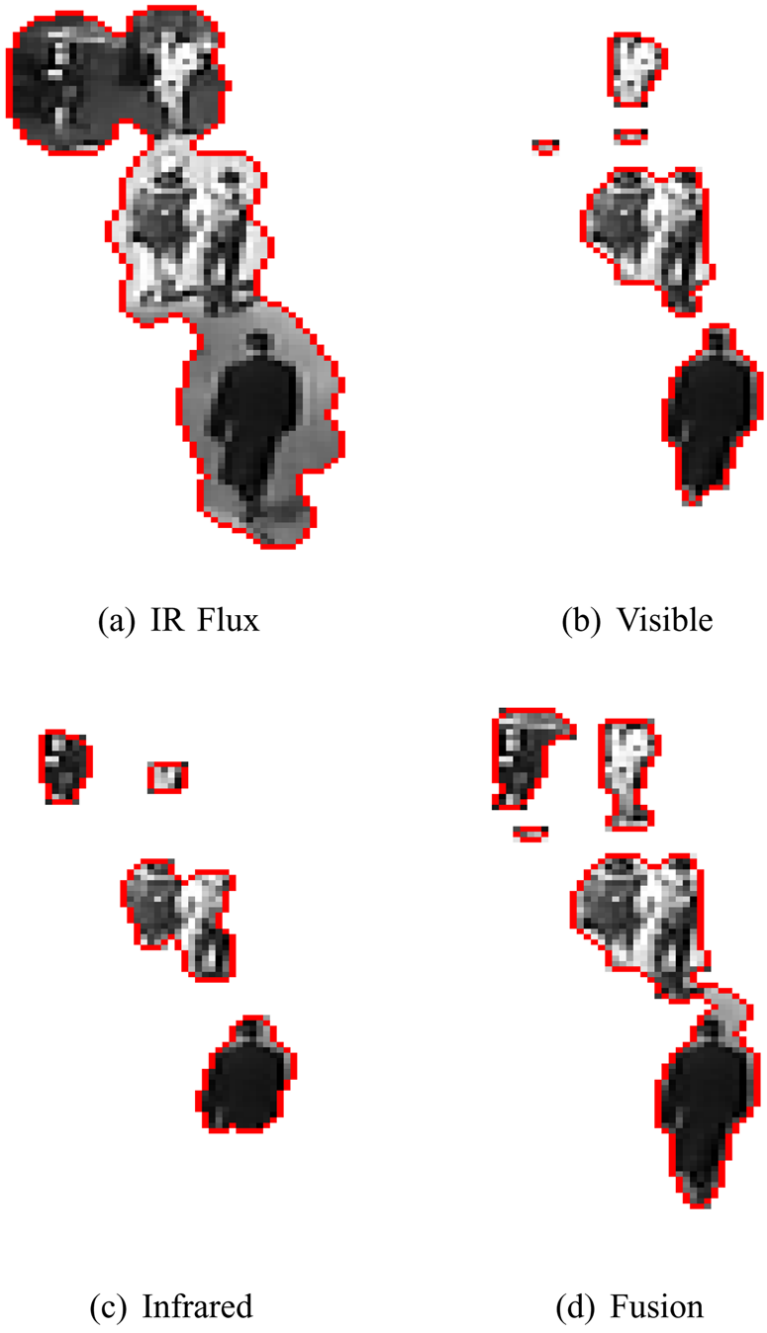


Figure 10.
(a) Motion blob #2 in frame #1256 using IR flux tensors. Refinement of blob #2 using (b) only visible imagery, (c) only infrared imagery, (d) using fusion of both visible and IR imagery.

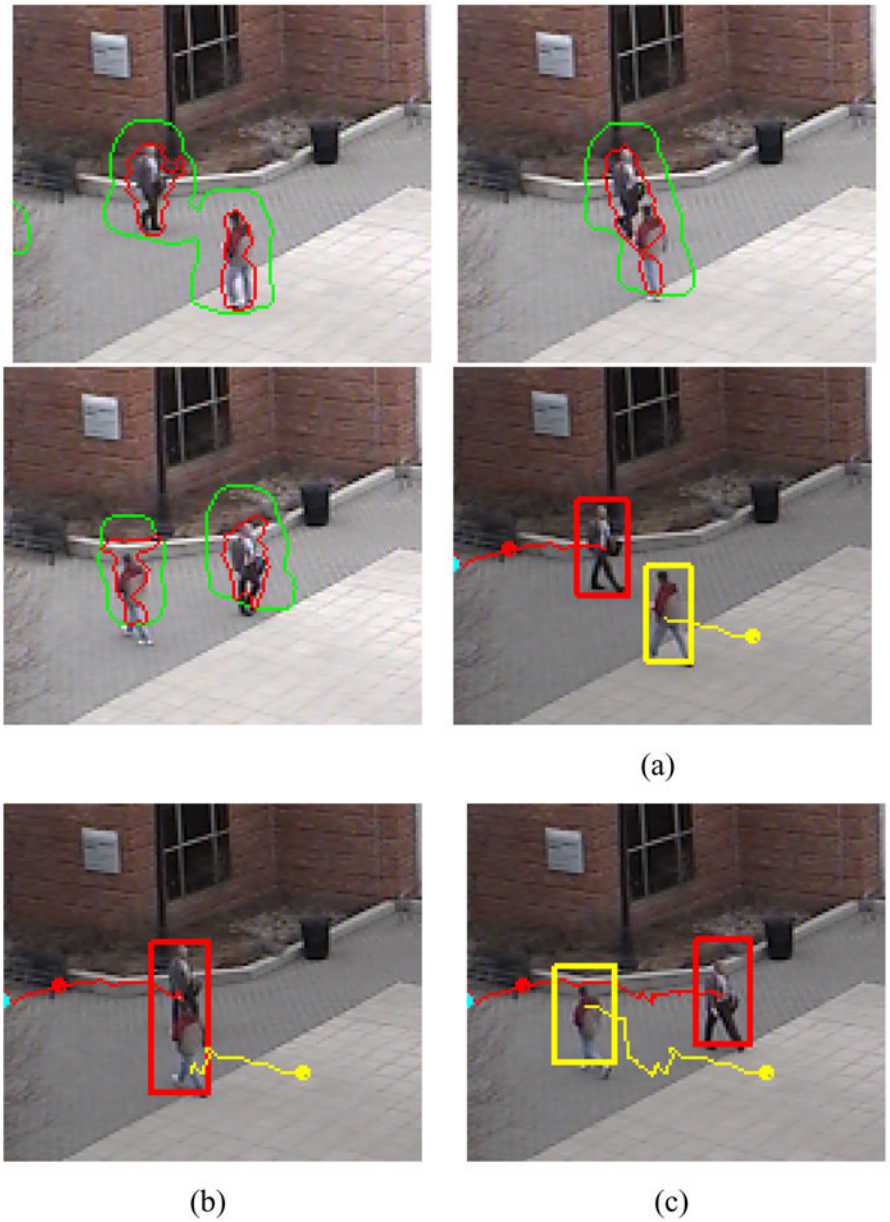


Figure 11.
Merge resolution. Left to right: frames #41, #56, and #91 in OTCBVS benchmark sequence 3:4. Top row: motion constrained object extraction results. Flux tensor results are marked in green, refined contours are marked in red. Bottom row: object trajectories after merge resolution.

TABLE I

Local descriptors that can be obtained from the two eigenvalues λ_1, λ_2 derived from the 2D color structure tensor [25].

$\lambda_1 + \lambda_2$	total local derivative energy
λ_1	derivative energy in the most prominent direction
$\lambda_1 - \lambda_2$	line energy
λ_2	amount of energy orthogonal to the prominent local orientation

Algorithm 1

Object Segmentation Algorithm

Input: Visible image sequence $\mathbf{I}_{RGB}(\boldsymbol{x}, t)$, infrared image sequence $\mathbf{I}_{IR}(\boldsymbol{x}, t)$, foreground mask sequence $FG_M(\boldsymbol{x}, t)$ with $N_M(t)$ regions

Output: Refined foreground (binary) mask sequence $FG_R(\boldsymbol{x}, t)$ with $N_R(t)$ regions

for each time t **do**

  Compute edge indicator functions $g_{IR}(\boldsymbol{x}, t)$ and $g_{RGB}(\boldsymbol{x}, t)$ from infrared $\mathbf{I}_{IR}(\boldsymbol{x}, t)$ and visible $\mathbf{I}_{RGB}(\boldsymbol{x}, t)$ images.

  Fuse $g_{IR}(\boldsymbol{x}, t)$ and $g_{RGB}(\boldsymbol{x}, t)$ into a single edge indicator function $g_F(\boldsymbol{x}, t)$.

  Initialize refined mask, $FG_R(t) \leftarrow 0$

  Identify disjoint regions $R_i(t)$ in $FG_M(t)$ using connected component analysis.

  for each region $R_i(t)$ $\{i = 1, 2, \ldots N_M(t)\}$ in $FG_M(t)$ **do**

    Fill holes in $R_i(t)$ using morphological operations.

    Initialize geodesic active contour level sets $C_i(t)$ using contour of $R_i(t)$.

    Evolve $C_i(t)$ using $g_F(t)$ as edge stopping function.

    Check stopping/convergence condition to subpartition $R_i(t) = \{R_{i,0}(t), R_{i,1}(t), \ldots, R_{i,N_{R_i}(t)}(t)\}$ into $N_{R_i}(t) \geq 1$ foreground regions and one background region $R_{i,0}(t)$.

    Refine mask FG_R using foreground partitions as $FG_R = FG_R \cup R_{i,j}$; $j = 1: N_{R_i}(t)$

  end for// $N_M(t)$ regions

end for// T frames

Algorithm 2

Shape-based Segmentation Refinement Algorithm

Input: $R \in FG_R(\boldsymbol{x}, t)$ // Disjoint region in refined foreground mask

Output: R_i's ($R = \cup_{i=1:n} R_i$, $n \geq 1$) // Sub-regions in R

$R_E = R \ominus se_1$ // erode the region

$L = Label(R_E)$ // apply connected component labeling

$\boldsymbol{M_i} \leftarrow$ Disjoint Regions in R_E, $R_E = \cup M_i$ // disjoint regions in R_E will be used as markers

if if ($\# M_i s \leq 1$) **then**

  Return R // keep R intact

else

  for each $M_i \in R_E$ **do**

    $P_i = (M_i \oplus se_2) \subset FG_R, se_2 > se_1$ // conditional dilation of the markers/parts

  end for

  $R_{rec} = \Sigma P_i$

  $Separators \leftarrow 0$

  $Separators(R_{rec} > 1) \leftarrow 1$ // overlapped regions are locations where the object should be splitted

  $Split \leftarrow ValidateSplit(R, Separators)$

  if if ($Split = = 0$) **then**

    Return R // keep R intact

  else

    $R(Separators = = 1) \leftarrow 0$

    $R_i \leftarrow$ Disjoint Regions in R

    Return R_i's

  end if

end if

Algorithm 3

Tracking Algorithm

Input: Image sequence $\mathbf{I}(\boldsymbol{x}, t)$, and refined foreground mask sequence $FG_R(\boldsymbol{x}, t)$

Output: Trajectories and Temporal Object Statistics

for each frame $\mathbf{I}(\boldsymbol{x}, t)$ at time t **do**

  Use the refined foreground mask, $FG_R(t)$ from the motion constrained object segmentation module.

  Partition $FG_R(t)$ into disjoint regions using connected component analysis $FG_R(t) = \{R_1(t), R_2(t), \ldots, R_{N_R}(t)\}$ that ideally correspond to N_R individual moving objects.

  for each region $R_i(t)$ $\{i = 1, 2, \ldots N_R(t)\}$ in $FG_R(t)$ **do**

    Extract blob centroid, area, bounding box, support map etc.

    Arrange region information in an object correspondence graph O_{GR} Nodes in the graph represent objects $R_i(t)$, while edges represent object correspondences.

    Search for potential object matches in consecutive frames using multi-stage overlap distance D_{MOD}.

    Update O_{GR} by linking nodes that correspond to objects in frame $\mathbf{I}(\boldsymbol{x}, t)$ with nodes of potential corresponding objects in frame $\mathbf{I}(\boldsymbol{x}, t-1)$. Associate the confidence value of each match, $C_M(i, j)$ with each link.

  end for

end for

Trace links in the object correspondence graph O_{GR} to generate moving object trajectories.

Algorithm 4

Cluster Trajectory Analysis

Input: Merged trajectory segment TM, parent and child trajectory segments TP(1: n_p), TC(1: n_p).

Output: Individual trajectory segments TS(1: n_p), updated parent and child trajectory segments TP(1: n_p), TC(1: n_p).

1: Initialize state matrix $\mathbf{X}(t_0)$ consisting of position and velocity informations for each individual trajectory segments TS(1: n_p) using the parent segments' states, $\mathbf{X}(t_0) \leftarrow$ TP(1: n_p). states

2: **for** each node TM. node(t) of the merged segment **do**

3: Predict using Kalman filter [48] $\hat{\mathbf{X}}(t)$, the estimated state matrix of the trajectory segments TS(1: n_p) corresponding to individual objects in the object cluster, from the previous states $\mathbf{X}(t-1)$.

4: Project masks of the sub-nodes TS(1: n_p). node(t−1) from the previous frame, to the predicted positions on the current merged node TM. node(t), and use these projected masks as markers for the watershed segmentation.

5: Using watershed segmentation algorithm and the markers obtained through motion compensated projections of the sub-nodes from the previous frame, segment the merged node TM. node(t) corresponding to a cluster of objects into a set of sub-nodes corresponding to individual objects TS(i). node(t), $i = 1: n_p$.

6: Use refined positions of the sub-nodes obtained after watershed segmentation to update the corresponding states $\mathbf{X}(t)$.

7: **end for**

8: Update object correspondence graph $O_{G\mathcal{R}}$ by including sub-node informations such as new support maps, centroids, areas etc.

9: Update individual trajectory segments's parent and children links (TS(1: n_p). parents, TS(1: n_p). children), parent segments' children links (TP(1: n_p). children), and children segments' parent links (TC(1: n_p). parents) by matching TSs to TPs and TCs.

10: Propagate parent segments' labels to the associated sub-segments, which are subsequently propagated to their children (TP(1: n_p). label → TS(1: n_p). label → TC(1: n_p). label).