Time scales in video surveillance

Nathan Jacobs, *Student Member, IEEE,* and Robert Pless, *Member, IEEE*

***Abstract*—Events in surveillance video occur over many time scales, but common approaches to background subtraction and video representation are implicitly based on a single temporal scale. In this work we derive a set of causal filters which define a temporal scale-space representation for the activity at each pixel. This scale-space can be maintained and continuously updated in real-time, and, for static cameras viewing dynamic scenes, has several interesting properties. In particular, it directly characterizes interesting temporal features and supports approximate reconstruction of the video history under challenging noise conditions. The temporal scale-space grounds novel approaches to several applications, including a natural visualization tool to summarize recent video behavior in a single image, and a tool to directly report how long the object has been present in a scene without re-examining any video data.**

***Index Terms*—change detection, background modeling, video surveillance, video analysis**

I. Introduction

Video surveillance is becoming dramatically more widespread; millions of cameras are already deployed in Britain and the United States in order to monitor traffic, borders and critical infrastructure. Each of these cameras creates a stream of data which must be automatically analyzed, observed by a human, archived, discarded, or some combination of these. While much research has sought tools to compress and transmit video data to minimize artifacts for human observation [19], or to analyze data in real time to extract relevant features from raw image data for subsequent processing [8], [11], relatively little has reasoned explicitly about the temporal structure of the video.

This is perhaps surprising, because surveillance systems must often answer inherently temporal questions, such as "How long has a particular object been in the scene?" These questions may be explicit, such as in airport left-bag situations. They may also arise implicitly, such as in deciding when to incorporate a new object into the background model for a scene. The complete answer to such question requires the semantic segmentation of the scene into specific objects. For general scenes, however, segmentation of unknown objects is not yet a robust operation, so we consider two simplifying choices. First, instead of modeling the scene at the object level, we consider a temporal decomposition of the intensity at each pixel. Second, to make the computation and storage feasible, we focus on a variable resolution temporal decomposition, which retains greater accuracy for events that happened more recently.

N. Jacobs and R. Pless are with the Department of Computer Science and Computer Engineering, Washington University, St. Louis, MO, 63130 USA e-mail: jacobsn—pless@cse.wustl.edu.

Recent evidence suggests that biological visual systems make similar choices. Recordings in the lateral geniculate nucleus (LGN) of the cat shows the temporal precision of the neural representation depends on the temporal extent of the event. Although these neurons are thought to code for motion rather than object identity or permanence, they maintain a constant relative accuracy with an error in temporal representation that is proportional to the temporal extent of the event [6].

Thus, we explore a scale-space representation that provides the ability to reconstruct the recent history of the video signal, with an accuracy that degrades as the signal is reconstructed further back in time. This representation, which is the response to a collection of simple causal filters with different filter constants, is simple to maintain and update online. In this paper we argue that this provides both a compelling decomposition of the video data, describes interesting features for video analysis, and supports new types of tools for both automatic and human-in-the-loop video surveillance.

This scale-space representation is useful in the context of both human and machine analysis of video data. Human surveillance operators are being integrated into larger surveillance systems, with a pre-processing step that only shows interesting video data to the operator. But when an operator is responsible for thousands of cameras instead of dozens, it is unreasonable to expect them to remember what the scene typically looks like or how it commonly varies. Section VI illustrates how the temporal scale space provides robust and simple visualization tools for the operator to quickly understand the context of the scene. Additionally, this representation can be incorporated into the pre-processing steps for other surveillance algorithms—and offers explicit control over both how long an object needs to be visible to be considered more than noise, and how long an object must be in the scene before being considered part of the background.

After a review of related work in video surveillance and background representation, we present our temporal scale-space and a design problem for choosing appropriate time scales. To explore this representation, we derive the equations to approximately reconstruct the original video from the data available at any given time and explore the accuracy and sensitivity to noise of this reconstruction. We conclude by presenting two potential application domains, first a traffic scene where the visualization tools immediately present a view of how long each vehicle has been in the scene, and second, an experiment run on the PETS 2006 dataset[1] in a challenging "left bag" detection problem.

[1]http://www.pets2006.net, April 25th, 2006

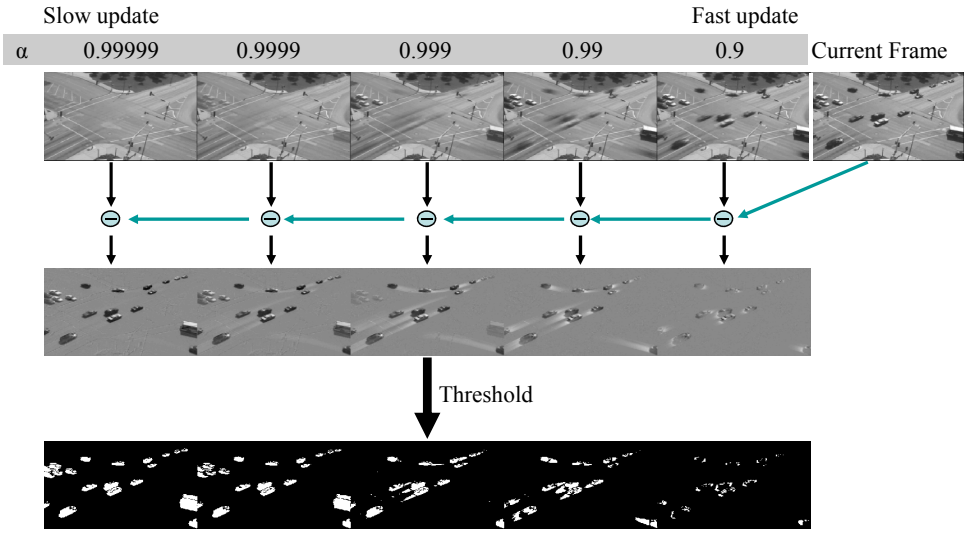


Fig. 1. Background models at different time scales (top) constructed from video of a traffic scene are used to estimate the foreground regions (bottom) of a new video frame. Background models with short time scales only detect change that occurred recently.

II. Prior Work

The foundations of scale-space theory were developed by Lindeberg and Fagerström [15] who show that the temporal scale-space used in this paper is the only one that respects the causal nature of time, has a continuous scale parameter, and does not introduce local-maxima as scale increases. Bretzner and Lindeberg show [5] that tracking in a temporal scale-space can increase the robustness to temporary occlusions.

Otherwise, there has been relatively little work to explicitly include temporal filters of varying extents within video surveillance applications. The most recent work with temporal filters has been in gesture and action recognition; two examples are defining a vector-valued image summarizing recent motion at each pixel in order to recognize a small library of actions [4] and the reverse problem of finding all instances of a given action by searching for a specific spatio-temporal template [3].

Deeper analysis of the time sequence of pixel intensity values over time has been suggested for the problem of tracking objects on the water, but was deprecated because of a lack of efficient tools to develop temporal signatures of every pixel in the scene [1].

Within the surveillance domain, foreground detection is usually performed with reference to a background model. This background model may be based upon pixel intensity statistics, either a Gaussian mixture model [20], a non-parametric distribution [10], or a predictive model for the time sequence [21]. Alternatively, the model may be based on local estimates of the optic flow [17], [16], or parametric models of the distributions of spatio-temporal derivatives [18]. All of these background models can be updated to reflect changes in the scene, and allow the update rule to weight the recent scene appearance more strongly. While this permits these methods to accommodate to slowly changing backgrounds, the drift rate (how much the more recent frames dominate the background model) must be set ahead of time.

An alternative approach to background modeling represents scene dynamics at a global scale, rather than modeling the appearance at each pixel independently. Work on dynamic textures [9] represents each image in a particular video by a set of coefficients, and uses an ARMA model to characterize the time sequence of coefficients. This was extended to segmenting pixels on foreground objects by finding pixels whose changes did not fit this dynamic texture [22].

Explicit reasoning about foreground objects allows them to become another layer of background when they stop and remain stationary in the scene. Reasoning about these persistent has been done using a set of code-books to keep track of image features that appeared at different times [13], or by assigning pixel locations to an ordered set of layers [23], [14], but these methods are computationally expensive and it is unclear if they are robust to noise.

Recent work in foreground detection has explored using a pair of background models to detect changes at particular temporal intervals [2], [7]. These approaches maintain two background models, one for short-term changes and one for long-term changes. A pixel is labeled as having changed in a chosen temporal interval if it is labeled as foreground with respect to the long-term model but not with respect to the short-term model. Our work differs in that we explore the use of a full temporal scale-space by maintaining many background models. We believe this representation will be useful beyond the temporal-interval change detection problem. Additionally, we provide an analysis of using models with different filter constants, a method for choosing the filter constants, and a method for giving a specific estimate of the change time using the scale-space.

The contribution of the current paper is to offer a very lightweight method to maintain a background model simultaneously at multiple time scales. This builds on our previous work [12] but is recast in terms of a temporal scale-space and

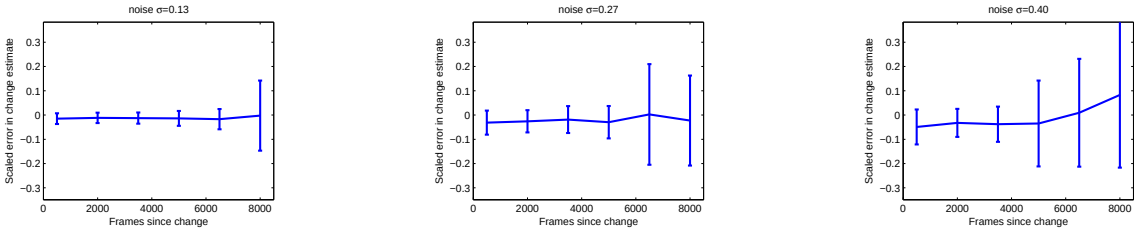


Fig. 2. The accuracy of estimating the change time, r_x, depends on the level of noise and the time since the change occurred. These plots show of the scaled error in estimating the change time, the absolute error divided by the actual time since the change, for six values of r (error bars are one standard deviation). The difference between the plots is the standard deviation of the Gaussian noise added to the signal ($\sigma = (.13, .27, .4)$).

extended to include an error analysis of the signal reconstruction in the presence of noise.

III. Temporal Decomposition

We create a multi-resolutional temporal decomposition of a video sequence by filtering pixel intensity values. This view-based approach is particularly useful when the camera is static, as is often the case in surveillance applications. The decomposition is constructed by maintaining multiple exponentially-weighted moving averages or, in other words, filtering the sequence with multiple causal low-pass filters, each with different filtering constants.

Given an image sequence $I(x,t)$, where x is the pixel index and t denotes time, we create a set of low-pass filtered sequences $\mathbf{L} = \{L_1, \ldots, L_N\}$ defined by the following recursive equation:

$$L_i(x,t) = \alpha_i L_i(x,t-1) + (1-\alpha_i) I(x,t) \qquad (1)$$

for $t > 0$ and $L_i(x,0) = B_x$, an application dependent initialization constant, for example the median background intensity at x.

The filtering constant $\alpha_i \in [0,1]$ determines the amount of the current image $I(x,t)$ to include in $L_i(x,t)$. The set $\mathbf{L}$ of low-pass filtered images depends on the set of filter constants $\mathbf{A} = \{\alpha_1, \ldots, \alpha_N\}$. The selection of $\mathbf{A}$ depends on the video frame-rate and temporal scales of interest.

As an alternative to (1), L_i can be written as the following linear equation:

$$L_i(x,t) = (1-\alpha_i) \sum_{j=1}^{t} \alpha_i^{t-j} I(x,j) + \alpha_i^t B_x. \qquad (2)$$

This form of L_i is used as a basis for the image sequence reconstruction process and subsequent methods.

Fig. 1 shows an example of this temporal decomposition at a single time step for a video of a traffic intersection. It also shows that using different time scales gives different results for the foreground detection problem; background images with shorter time scales (smaller filter constants) only detect changes that occurred in the recent past.

In the remainder of this paper we explore applications of this temporal decomposition. Note that while descriptions are in terms of a temporal decomposition of pixel intensity values, it is possible to decompose other signals generated from video sequences, *i.e.* a binary foreground-background detection sequence.

IV. Signal Reconstruction

Given a set of N exponentially-weighted moving average images it is possible to reconstruct the original video exactly by inverting (2) (if the original video is N or fewer frames). This is of little practical value because memory use is unbounded. However, given additional constraints on the form of the signal, it is possible to approximately reconstruct significantly more frames with constant memory. We give an example of one such constraint that enables us to estimate when a pixel most recently changed. The constraint also provides intuition for the less computationally intensive method described in Section V.

We constrain the reconstructed signal at each pixel to be piecewise constant with only two pieces. While not a practical model over long durations it is useful for modeling short-term image changes (*e.g.* a person or vehicle occluding the background).

Specifically, we make the assumption that the signal at each pixel has the following form:

$$I(x,t) = \begin{cases} f_{x,1} & \text{if } t < r_x \\ f_{x,2} & \text{if } t \geq r_x \end{cases}. \qquad (3)$$

Reconstructing the signal reduces to determining the time when the pixel changed, r_x, and the pixel intensity before and after the change, respectively $f_{x,1}$ and $f_{x,2}$. With this signal model, (2) can be simplified as follows (see appendix for derivation):

$$\hat{L}_i(x,t) = f_{x,2} + (f_{x,1} - f_{x,2}) \alpha_i^{t+1-r_x}. \qquad (4)$$

Given the known low-pass filter responses $L_i(x,t)$ we seek the the parameters $f_{x,1}$, $f_{x,2}$, and r_x for the piecewise-constrained response $\hat{L}_i(x,T)$ that most closely approximate the actual signal responses $L_i(x,T)$ in the least-squares sense:

$$\underset{r_x, f_{x,1}, f_{x,2}}{\arg\min} \sum_{i=1}^{N} (L_i(x,T) - \hat{L}_i(x,T))^2 \qquad (5)$$

The set of filter constants $\mathbf{A}$ affects the accuracy of the estimate of the change time r. To explore this we conducted an experiment with a synthetic signal, a two-piece piecewise constant signal corrupted by different levels of additive Gaussian noise. The piecewise signal value changes from one to zero at time step 2000 (i.e., $(r, f_1, f_2) = (2000, 1, 0)$). The standard deviation of the noise ranges from 0.13 to 0.40. We use a fixed set of filters spaced logarithmically between $1-e^{-2}$

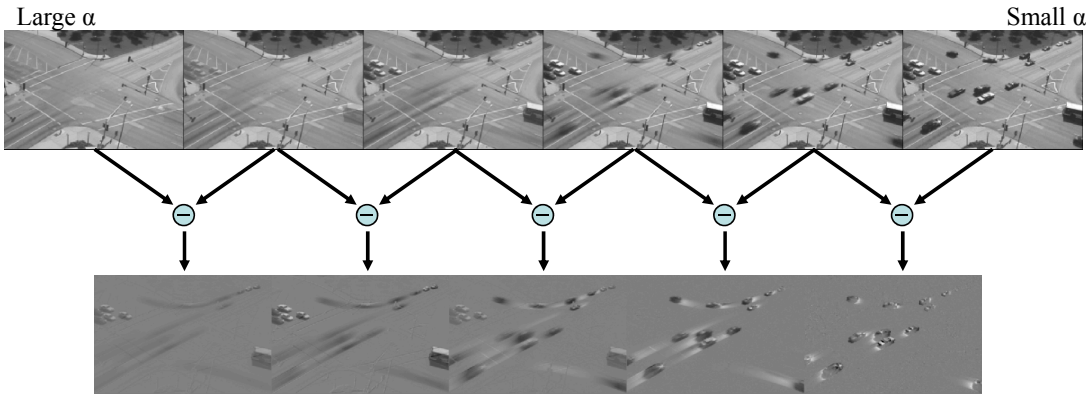


Fig. 3. (top) The time scale decomposition at one time step of the video of a traffic intersection. The difference between background images with small filter constants (bottom right) highlight recent changes, for example the cars moving through the intersection. With larger filter constants (bottom left), the highlighted changes occurred a longer time ago, for example the cars waiting for the light to change. These difference images are used in two applications described in Section VI.

and $1 - e^{-8}$. The results, shown in Fig. 2, show that at the lowest noise level this set of filter constants can be used to estimate the time of recent changes with a small error. The error increases significantly when the time since the change is above 6000. As the level of noise increases the error in estimating recent changes increases and the point at which large estimation errors begin occurs more sooner.

Reconstructing a signal from the temporal decomposition is instructive but in practice performing the reconstruction is computationally intensive—requiring a solution to (5) for each pixel. In the next section we show a method for extracting useful information from $\mathbf{L}$ without this step.

V. Change Detection without Reconstruction

Determining when the most recent change occurred in a video, which we define as determining r_x in (4) for all pixels, is useful for many applications. However, the computationally intensive signal reconstruction step described in Section IV is unnecessary when an exact estimate of r_x is not needed. In this section we describe a filter, based on a combination of two low-pass filter responses, and a corresponding approximation that is significantly less computationally intensive.

We define the difference of low-pass filter (DoLP) $D_{i,j}(x,t) = L_i(x,t) - L_j(x,t)$ which, if we assume a two-piece piecewise constant signal, can be simplified as follows:

$$D_{i,j}(x,t) = (f_{x,1} - f_{x,2})(\alpha_i^{t+1-r_x} - \alpha_j^{t+1-r_x}). \quad (6)$$

Fig. 3 shows an example of difference of low-pass filter responses for a video of a traffic intersection. We now explore properties of this filter when applied to piecewise constant signals.

A. Estimating Recent Changes

This section presents a method for rapidly estimating the change time of a signal using a set of difference of low pass filter (DoLP) responses. Fig. 4 shows an example of the responses of a set of DoLP filters, with α_i and α_j set to consecutive elements of $\mathbf{A} = 1 - \left\{e^{-1}, e^{-2}, \ldots, e^{-N}\right\}$, to a unit step function input. This structure is also evident in the responses to real signals that are not perfect unit step functions, see Fig. 8 (more fully described in Section VI).

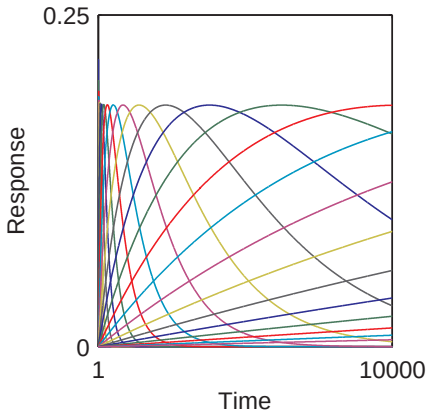
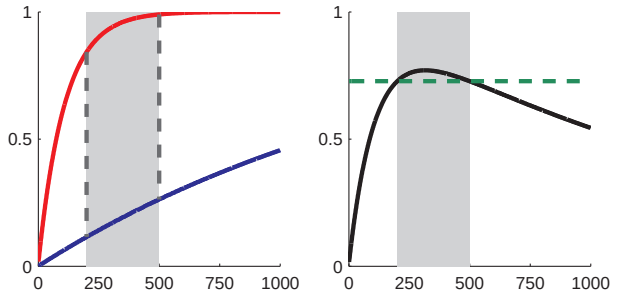


Fig. 4. **Using the difference of low-pass filter (DoLP) responses to approximate the change time.** The figure shows the time-series response of a set of DoLP filters to an input function that changed at time zero, $r_x = 0$. The DoLP filter constants are logarithmically-spaced, $\mathbf{A} = \{\alpha_1, \ldots, \alpha_n\} = 1 - \left\{e^{-1}, e^{-2}, \ldots, e^{-N}\right\}$. A given DoLP filter response is the maximum response for a contiguous temporal interval and this interval can be used as an estimate of when the change in the input signal occurred. The temporal interval for each DoLP filter can be computed numerically in advance and stored in a lookup table (see Table I for an example).

Our approach to estimating the change time is motivated by the observation that a given DoLP filter response, $D_{i,i+1}$, is the maximum response within a set of DoLP filters for a contiguous temporal interval. The maximum response at a particular time-step, t, can be determined on-line and the corresponding interval can be used as an estimate of the change time, r_x, of the input signal. The temporal interval for

TABLE I
LOOKUP TABLE OF TEMPORAL INTERVALS FOR A DOLP FILTER SET

α_i	α_{i+1}	**Start frame**	**End frame**
$1-e^{-2}$	$1-e^{-3}$	6	17
$1-e^{-3}$	$1-e^{-4}$	18	49
$1-e^{-4}$	$1-e^{-5}$	50	136
$1-e^{-5}$	$1-e^{-6}$	137	372
$1-e^{-6}$	$1-e^{-7}$	373	1015
$1-e^{-7}$	$1-e^{-8}$	1016	5000



(a) Low-pass filter responses (b) Difference of low-pass filter response

Fig. 5. **Responses of a set of filters automatically derived to detect pixels that changed between 250 and 500 frames ago (gray region).** Both plot show responses to a unit step function input at $t = 0$. (a) The responses (red and blue) by the two low-pass filters. The region between the two vertical line corresponds to the temporal region of interest. (b) Shows the difference of the two low-pass filters in (a). As desired, the threshold (green) is exceeded only when $250 < t < 500$.

each DoLP filter in a filter set can be computed numerically in advance and stored in a lookup table (see Table I for an example). This interval only needs to be computed once because it does not depend on the magnitude of the signal. We populate the lookup table entry for a DoLP filter, $D_{i,i+1}$, by simulating the responses of all filters in the filter set to a unit-step function and determining the values of t that cause $D_{i,i+1}$ to be larger than all other responses.

We have shown how to use a generic set of filter constants to provide an estimate of the change time. In applications where the temporal intervals of interest are known in advance it is possible to provide a better estimate. In Section V-B we show how to choose filter constants for determining if a change occurred in a specific temporal interval.

B. Designing Filtering Constants

In some applications the specific temporal intervals of interest are known in advance and the approximation inherent with a generic set of filtering constants can be avoided. In this section, instead of choosing the maximum response from a set of responses, as in the previous section, we use a single DoLP response tuned to detect changes that occurred in a specific temporal interval. We show a method for choosing a pair of filter constant values and a threshold to apply to the DoLP response that meet the application requirements. Using this method temporal accuracy is improved without additional runtime cost.

Given an application that requires detecting changes that happened between t_1 and t_2 time-steps in the past we want to design a pair of filters that can be used to accurately detect changes in this interval. The design problem is to determine filter constants, α_1 and α_2, and a threshold c such that the DoLP response for the pair of filters exceeds the threshold $c < D_{1,2}(x,t)$ when the change in the underlying signal occurred in the specified interval $t_1 \leq r_x \leq t_2$ (see Fig. 5). Note that using the threshold c replaces the process of finding the maximal response in the method described in the previous section.

Combining these requirements with the simplified DoLP response (6) results in the following constraints on the solution to the design problem:

$$(f_{x,1} - f_{x,2})(\alpha_i^{t_1+1} - \alpha_j^{t_1+1}) = c \quad (7)$$

$$(f_{x,1} - f_{x,2})(\alpha_i^{t_2+1} - \alpha_j^{t_2+1}) = c \quad (8)$$

The design problem is ill-posed so we need an additional constraint. Empirically we find choosing α_1 such that $L_1(x,t_2) = 0.99 * f_{x,2}$ (*i.e.* constraining the intersection of the red curve with the right-most dashed line in Fig. 5.(a)) results in design parameters that work well in practice. Using α_1 we solve numerically to find the non-trivial value of α_2 (the trivial solution is $\alpha_2 = \alpha_1$). Given α_1 and α_2 and assuming that the input is a unit step function it is straightforward to solve for the specific value of c needed to properly threshold the signal.

Since the filters, α_1 and α_2, are designed for $f_{x,1} = 0$ and $f_{x,2} = 1$ the threshold c will be incorrect for other signals. To compensate for this, the threshold c needs to be rescaled, at runtime, according to the actual signal. As an approximation, we use a low-pass filter response with a large filter constant to estimate $f_{x,1}$ and a filter response with a small filter constant to estimate $f_{x,2}$.

VI. SAMPLE APPLICATIONS

Security personnel are often tasked with monitoring multiple video streams. A key question that arises when a new video stream is presented is "How long has that object been there?". This section highlights how tools to answer this question directly can be derived using only the temporal decomposition (and not, for example, rewinding or reviewing the entire video). Maintaining the temporal video decomposition allows this to be computed on demand, in real-time.

We consider two different cases. The first is the visualization of the recent history of an entire scene that uses colors or multiple images to indicate how long each part of the scene has been constant. The second is the local analysis of a single pixel location in an indoor train station scene which is part of the PETS 2006 "left bag" data set.

A. Temporal visualization

The first application uses video from a static camera mounted far above an intersection. The scene contains multiple time scales of interest—moving vehicles, vehicles that have stopped momentarily, vehicles that are waiting at a light, and vehicles that have parked. The ability to evaluate potential

threats within this scene would benefit from knowledge of the status of each car. We create filtered images that separate objects that have been static for a long time from those that are moving. We maintain four background models using low-pass filters with alpha values: $\mathbf{A} = \alpha_1, \alpha_2, \ldots, \alpha_4 = 1 - \left\{e^{-1}, e^{-3}, e^{-5}, e^{-7}\right\}$.

Fig. 6(a) shows three frames of the input sequence, and Fig. 6(b) shows the decomposition of each frame—with the different filtered images stacked as a column. These filtered images are created by compositing a background image with information extracted from a set of three difference of low-pass filtered images ($D_{1,2}, D_{2,3}$, and $D_{3,4}$). For illustration purposes, we define a background image as the average of all images in the sequence (although in a continuously operating online system, we could use the image L_4 or the low-pass filtered image with the largest filtering constant). Pixel values that differ from background by more than 25 gray values are considered to be foreground. At each frame, each foreground pixel has a largest response in one of the three difference image; that foreground pixel is drawn onto the corresponding filtered image. In the first column, the scene is new, each pixel location just changed from the background, and all objects are therefore drawn on the top image. In the second column, corresponding to frame 60, the bus and opposite cars are shown at the second time scale, while the moving cars show up at the shortest time scale. Finally, in the third column, the bus and cars have still not moved, and are drawn in the filtered image corresponding to the longest time scale. Fig. 6(c) shows a false color image where the color indicates the length of time that each pixel has been static, and summarizes each column of Fig. 6(b) in one image. An example of this false color visualization on another scene is shown in Fig. 7.

B. Left-bag detection

Without hypothesizing a background model it is still possible to estimate when an object appeared in the scene, even in the case of substantial noise and short occlusions. Fig. 8(a,b,c) shows several frames of the PETS 2006 "left bag" challenge. Fig. 8(d) shows the pixel intensity profile of a pixel (the center of the square in the image) that views the bag. Fig. 8(e) shows the $D_{i,i+1}$ (difference of low pass filter images) for $i \in \{1..27\}$, where $\alpha_i = 1 - e^{-(0.5+\frac{i}{2})}$. The piece-wise constant reconstruction of the pixel intensity using *only* the $D_{i,i+1}$ values (following (5)) indicates a time that the bag was left that is accurate to within the error of a human estimate. Note that this accuracy is achieved despite significant noise before the bag is left, and 5 occlusions (people walking in front of the bag) after the bag has been left.

VII. Discussion

This paper considers the use of a temporal scale-space as a lightweight representation of the recent video data. This scale-space creates a robust, multi-resolution decomposition of the recent appearance history at each pixel, and later processing stages can choose the most relevant time scale. At each time scale, the temporal decomposition provides a natural robustness to much shorter impulse noise, which often arises due to brief occlusions in video. Unlike (the more common) spatial scale-spaces, the values of the pixel measured by filters with varying temporal scales is directly useful as a feature defining the recent history of that pixel. We have illustrated how these features correlate with scene properties of interest (such as how long ago an object appeared), and the strength of this approach is that this correlation is immediate and maintained in real time.

This work has taken an intentionally minimalist approach in an attempt to highlight the potential of simple causal temporal filters. Direct visualization of these filter responses provides a form of visual context for a video surveillance operator. Furthermore, as visual surveillance algorithms continue to be deployed in more varied and realistic environments, better representation of temporal variations at different scales are vital for scenes with consistent background motions, and lighting variations such as flashing lights at night or intermittent sunlight during the day.

Appendix A
Simplification using piecewise assumption

In this appendix we show how to derive (4) from the recursively-defined low-pass filter response (2). We make the assumption that the signal, $I(x,j)$, at a pixel, x, is two-piece piecewise constant and that the first signal value, $f_{x,1}$, is equal to the initialization constant, B_x. The derivation is as follows: $\hat{L}_i(x,t) =$

$$
\begin{aligned}
&= (1-\alpha_i)\sum_{j=1}^{t}\alpha_i^{t-j}I(x,j) + \alpha_i^t B_x \\
&= (1-\alpha_i)\left[\sum_{j=1}^{r_x-1}\alpha_i^{t-j}f_{x,1} + \sum_{j=r_x}^{t}\alpha_i^{t-j}f_{x,2}\right] + \alpha_i^t f_{x,1} \\
&= \alpha_i^t(1-\alpha_i)\left[\sum_{j=1}^{r_x-1}\alpha_i^{-j}f_{x,1} + \sum_{j=r_x}^{t}\alpha_i^{-j}f_{x,2}\right] + \alpha_i^t f_{x,1} \\
&= \alpha_i^t(1-\alpha_i)\left[\sum_{z=0}^{r_x-2}\alpha_i^{-(z+1)}f_{x,1} + \sum_{z=0}^{t-r_x}\alpha_i^{-(z+r_x)}f_{x,2}\right] \\
&\qquad + \alpha_i^t f_{x,1} \\
&= \frac{1-\alpha_i}{1-\alpha_i^{-1}}\left[\alpha_i^{t-1}(4)f_{x,1}(1-\alpha_i^{1-r_x}) + \alpha_i^{t-r_x}f_{x,2}(1-\alpha_i^{r_x-t-1})\right] \\
&\qquad + \alpha_i^t f_{x,1} \\
&= f_{x,2} + (f_{x,1} - f_{x,2})\alpha_i^{t+1-r_x}
\end{aligned}
$$

References

[1] Vitaly Ablavsky. Background models for tracking objects in water. In *Proc. IEEE International Conference on Image Processing*, pages III: 125–128, 2003.

[2] Didier Aubert, Frédéric Guichard, and Samia Bouchafa. Time-scale change detection applied to real-time abnormal stationarity monitoring. *Real-Time Imaging*, 10:9–22, 2004.

[3] Moshe Blank, Lena Gorelick, Eli Shechtman, Michal Irani, and Ronen Basri. Actions as space-time shapes. In *Proc. IEEE International Conference on Computer Vision*, pages 1395–1402, 2005.

[4] Aaron F. Bobick and James W. Davis. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):257–267, 2001.

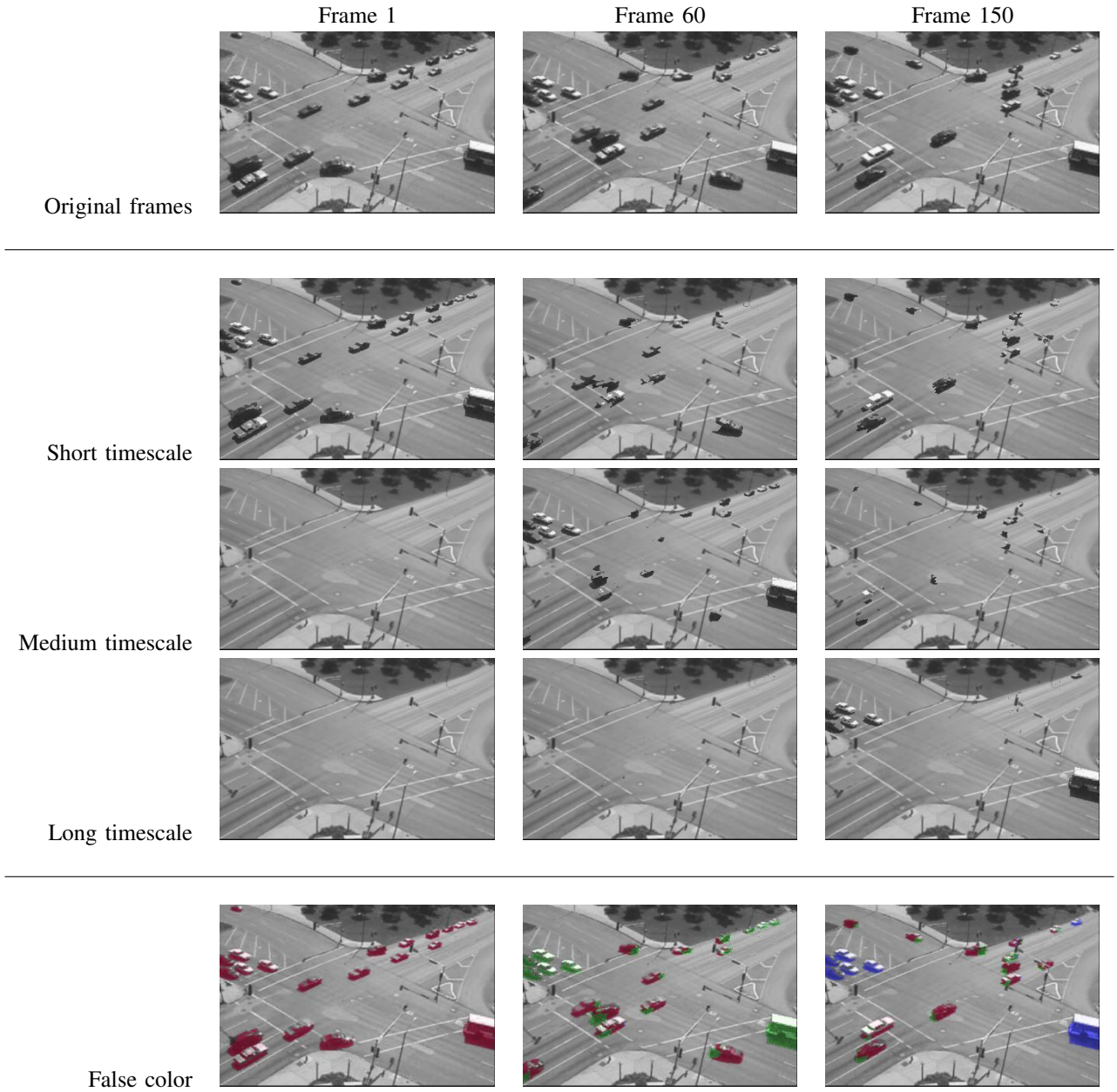


Fig. 6. **Examples of video frames filtered by thresholding difference of low-pass filter responses at each pixel.** (a) Three frames from a video of an intersection. The bus and the cars across the intersection have just arrived and do not move for the duration of the video but other traffic continues to move. (b) The rows correspond to difference of low-pass filters tuned to detect different temporal ranges with lower rows detecting longer temporal ranges. The columns are the image generated for the above original frame. As can be seen, the images of the stationary bus and cars move from the most to the least recent time scale image as time progresses but the non-stationary vehicles remain in the most recent time scale image.

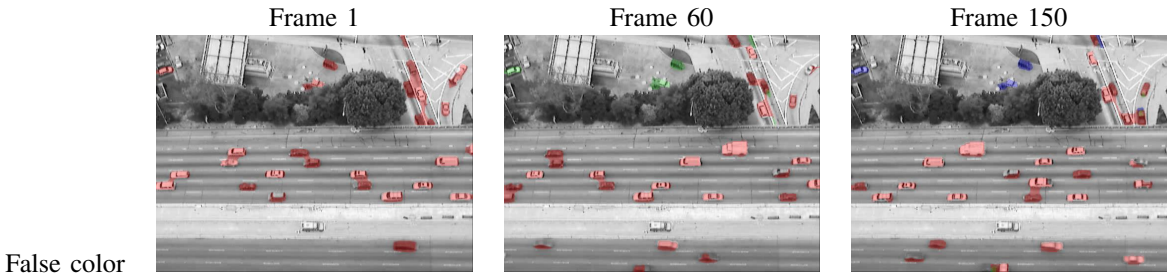
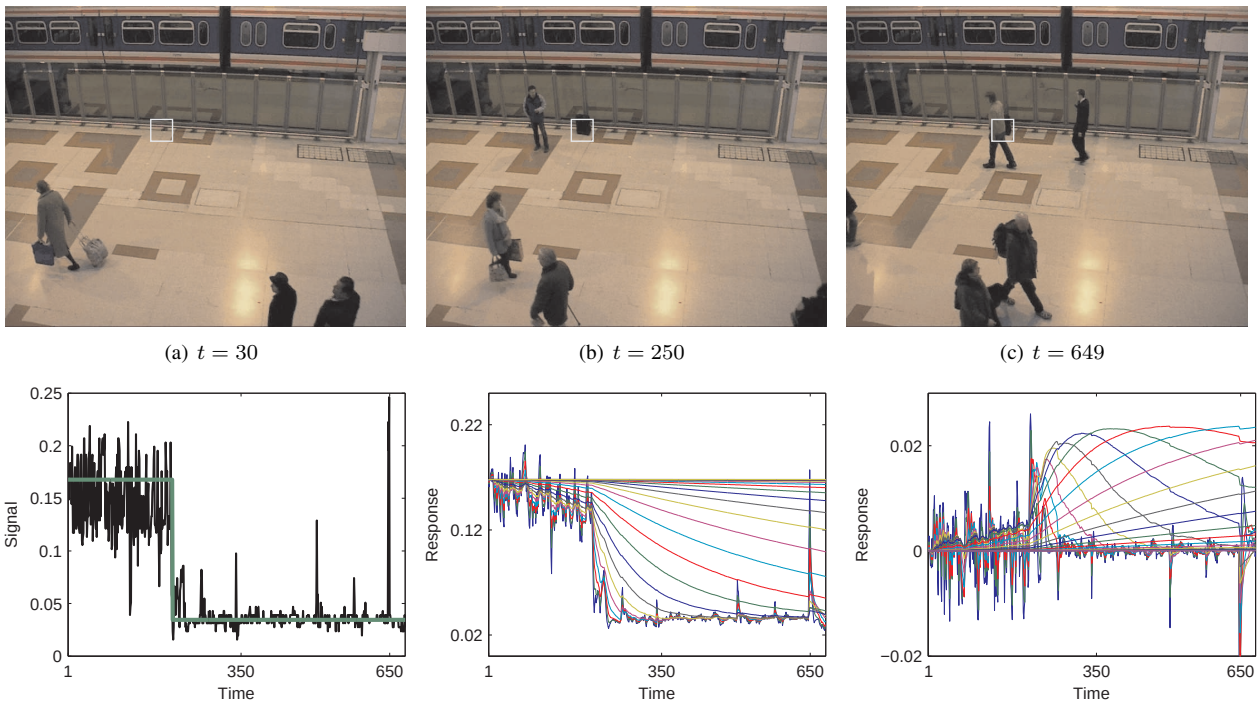


Fig. 7. The same filtering as shown in Figure 6, applied to another video sequence. By frame 150, the temporal decomposition of the scene gives an annotation in each image of which vehicles are moving and which continue to be stationary.



(a) $t = 30$ (b) $t = 250$ (c) $t = 649$

(d) Signal and reconstruction (e) LP filter responses (f) DOLP filter responses

Fig. 8. **An example of reconstruction and change estimation on real video data.** (a,b,c) Three example frames from a video in which a suitcase is left in front of a train. (d) The pixel intensity value over time of the pixel highlighted in (a,b,c). Notice the signal is noisy and the package is occluded several time by pedestrians (*e.g.* c). The reconstruction (green) is computed from the filter responses (e) at the end of the video and accurately highlights the time the bag was left. The difference of low-pass filter responses over time (f) have the structure expected from Fig. 4.

[5] Lars Bretzner and Tony Lindeberg. On the handling of spatial and temporal scales in feature tracking. In B.t. Haar Romeny, Luc Florack, Jan Koenderink, and Max Viergever, editors, *Scale-Space Theory in Computer Vision*, Lecture Notes in Computer Science 1252, pages 128–139. Springer-Verlag, 1997.

[6] Daniel A. Butts, Chong Weng, Jianzhong Jin, Chun-I Yeh, Nicholas A. Lesica, Jose-Manuel Alonso, and Garrett B. Stanley. Temporal precision in the neural code and the timescales of natural vision. *Nature*, 449(7158):92–95, 2007.

[7] Steven Cheng, Xingzhi Luo, and Suchendra M. Bhandarkar. A multiscale parametric background model for stationary foreground object detection. In *Proc. IEEE Workshop on Applications of Computer Vision (WACV)*, Austin, Tx, February 2007.

[8] Robert Collins, Alan Lipton, Takeo Kanade, Hironobu Fujiyoshi, David Duggins, Yanghai Tsin, David Tolliver, Nobuyoshi Enomoto, and Osamu Hasegawa. A system for video surveillance and monitoring. Technical Report CMU-RI-TR-00-12, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, May 2000.

[9] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto. Dynamic textures. *International Journal of Computer Vision*, 51(2):91–109, 2003.

[10] Ahmed Elgammal, Ramani Duraiswami, David Harwood, and Larry S. Davis. Background and foreground modeling using nonparametric kernel density for visual surveillance. *Proceedings of the IEEE*, 90(7):1151–1163, July 2002.

[11] Arun Hampapur, Lisa Brown, Jonathan Connell, Ahmet Ekin, Norman Haas, Max Lu, Hans Merkl, Sharath Pankanti, Andrew Senior, Chiao-Fe Shu, and Ying Li Tian. Smart video surveillance: exploring the concept of multiscale spatiotemporal tracking. *Signal Processing Magazine, IEEE*, 22(2):38–51, March 2005.

[12] Nathan Jacobs and Robert Pless. Real-time constant memory visual summaries for surveillance. In *Proc. ACM International Workshop on Visual Surveillance and Sensory Networks*, Santa Barbara, CA, October 2006.

[13] K Kim, D Harwood, and L Davis. Background updating for visual surveillance. In *International Symposium on Visual Computing (ISVC)*, pages 337–346, 2005.

[14] Mi-Suen Lee. Multi-layered background models for improved background-foreground segmentation. United States Patent 20030058237, 2003. What is claimed is: 1. A method, comprising: retrieving an image of a scene comprising a plurality of pixels; obtaining a background model of said scene; and creating a new layer in said background model if an object in said background model is moved.

[15] Tony Lindeberg and Daniel Fagerstroëm. Scale-space with causal time direction. *Proc. European Conference on Computer Vision*, 1064:229–240, 1996.

[16] Antoine Mittal and Nikos Paragios. Motion-based background subtraction using adaptive kernel density estimation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 302–309, 2004.

[17] Antoine Monnet, Anurag Mittal, Nikos Paragios, and Visvanathan Ramesh. Background modeling and subtraction of dynamic scenes. In *Proc. IEEE International Conference on Computer Vision*, pages 1305–1312, 2003.

[18] Robert Pless, John Larson, Scott Siebers, and Ben Westover. Evaluation of local models of dynamic backgrounds. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 73–78, 2003.

[19] Iain Richardson. *H. 264 and Mpeg-4 Video Compression*. Wiley, New York, 2003.

[20] Chris Stauffer and W E L Grimson. Adaptive background mixture models for real-time tracking. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 2246–2252, 1999.

[21] Kentaro Toyama, John Krumm, Barry Brumitt, and Brian Meyers. Wallflower: Principles and practice of background maintenance. In *Proc. IEEE International Conference on Computer Vision*, pages 255–261, 1999.

[22] J Zhong and S Sclaroff. Segmenting foreground objects from a dynamic textured background via a robust kalman filter. In *Proc. European Conference on Computer Vision*, pages 44–50, 2003.

[23] Yue Zhou and Hai Tao. A background layer model for object tracking through occlusion. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 1079–1085, 2003.