

MIDGROUND OBJECT DETECTION IN REAL WORLD VIDEO SCENES

B.Valentine, S.Apewokin, L.Wills, S.Wills

Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, GA, U.S.A.

A. Gentile

Dipartimento di Ingegneria Informatica
University of Palermo
Palermo, Italy

Abstract

*Traditional video scene analysis depends on accurate background modeling to identify salient foreground objects. However, in many important surveillance applications, saliency is defined by the appearance of a new non-ephemeral object that is between the foreground and background. This **midground** realm is defined by a temporal window following the object's appearance; but it also depends on adaptive background modeling to allow detection with scene variations (e.g., occlusion, small illumination changes). The human visual system is ill-suited for midground detection. For example, when surveying a busy airline terminal, it is difficult (but important) to detect an unattended bag which appears in the scene. This paper introduces a midground detection technique which emphasizes computational and storage efficiency. The approach uses a new adaptive, pixel-level modeling technique derived from existing backgrounding methods. Experimental results demonstrate that this technique can accurately and efficiently identify midground objects in real-world scenes, including PETS2006 and AVSS2007 challenge datasets.*

1. Introduction

Most surveillance applications use background modeling to factor out elements of the scene that are stationary or changing in uninteresting ways (such as swaying tree branches or rippling waves). This allows salient objects in the foreground to be more readily monitored, tracked, and analyzed for normal or anomalous behaviour.

However, many crucial surveillance tasks require attention to new stationary objects that appear and persist over time in the midst of a rapidly changing, cluttered scene (e.g., a suitcase left unattended in a crowded airline terminal for several minutes, a person or vehicle loitering near a busy street). Salient objects in these scenarios are between the foreground and the background, in a *midground* realm. The human visual system has evolved to detect rapidly moving objects, and is ill-suited to perceiving changes over longer time scales. Automated

video surveillance systems hold the potential to “tune in” to changes within a specified temporal window.

This paper introduces a midground detection technique that explicitly models a precisely defined temporal window during which foreground objects that have become stationary are classified as midground. This separates scene elements into three categories: long-lived, persistent background elements, short-lived (ephemeral), moving foreground elements, and newly stationary, persistent (non-ephemeral) midground objects.

Our approach to midground object detection employs a pixel-level, adaptive multimodal background model, annotated with temporal information. For computational efficiency and high throughput, we use integer arithmetic operations to produce the model. Our results demonstrate a fast and effective adaptive implementation that is able to detect midground objects in real-world scenes.

2. Related Work

Several techniques exist for background modeling and subtraction. These include frame differencing, temporal median, gradients, Gaussian probability density functions, and linear predictors. (See [1], [2], and [3] for recent comprehensive surveys.) These techniques vary in computational complexity and segmentation accuracy.

The simplest, frame differencing, utilizes information from a small set of recent frames. Sliding window based methods use the k most recent frames of the video and computes the median/mean of the buffer to represent background pixel values. Foreground is determined either by fixed thresholds or a measure of standard deviation from the window mean/median. In [4], edge features are used to identify foreground and background regions. Morphological filtering and density measures are applied to the edge images to locate foreground objects. Wallflower [5] uses a three-tiered approach combining pixel, region, and frame-level processing.

Stauffer and Grimson's mixture of Gaussians (MoG) modeling technique [6] is a popular multimodal approach that models each pixel as a weighted sum of 3-5 Gaussian probability density functions (pdf). A pixel in the current frame is compared against the set of weighted distributions

to determine a match. Lower weighted distributions are eventually replaced with new pixel distributions as more information from the video scene is obtained. This approach handles multimodal background and can incorporate new objects into the background based on a user-defined learning rate, but it is compute intensive. Appiah and Hunter [7] use temporal lowpass filters instead of Gaussian pdfs to implement multimodal backgrounding on a single-chip FPGA. The background weight, match, and updating schemes are similar to MoG, with simplifications to reduce floating-point calculations. In contrast to MoG [6] and [7] which use nonlinear updates of weights and pixel values, our multimodal background method uses a linear parameter updating scheme that supports efficient storage of a pixel’s long-term history. Our background model also incorporates aging factors to enable long-term adaptation by preventing long-lived modes from inappropriately dominating the model.

Matthew, et al. [8] and Sacchi, et al. [9] record a measure of the pixel’s history in time to aid in stationary object identification. In [8], time-based pixel history values are used within the framework of a MoG background subtraction technique to detect static objects. In [9], a specifically sized FIFO queue is used to measure the duration of observed non-background pixels. Our multimodal approach maintains multiple pixel models per pixel, each annotated with temporal information.

In [10], we compared the performance of our *multimodal mean* backgrounding algorithm with several pixel-based background modeling techniques executing in a resource-constrained real-time embedded environment. Our multimodal mean technique delivers accuracy comparable to multimodal MoG techniques but with a 6X improvement in execution time and an 18% reduction in required storage. This technique is summarized and extended to midground object detection in Section 3.

3. Midground Modeling

Our midground extraction algorithm uses an adaptive multimodal background model that maintains a set of mode averages for each pixel. The model includes temporal information about when and how often a pixel value was observed, which is used to detect midground. This section describes the multimodal mean background model and then uses it to define midground. Implementation details for efficient adaptation and storage in an embedded environment are presented in subsections 3.3 and 3.4. Based on this implementation, subsection 3.5 analyzes the storage and computation requirements of this technique. Section 4 presents experimental evaluation, demonstrating efficient midground object detection in real-world scenes, including scenarios from the PETS 2006 [11] and AVSS 2007 i-Lids [12] challenge datasets.

3.1 Multimodal Mean Background Model

If a particular pixel is multimodal (e.g., it is part of rustling tree leaves), it will display values that are clustered at more than one point in the color space. Our background model maintains an average value for each such cluster of values, or *mode*. Each image pixel value is represented as a three-component color representation space (e.g., RGB, HIS). In the following, $I_{t,x}$ represents the x color component of a pixel in frame t (e.g., $I_{t,red}$ denotes the red component of I_t).

The background model for a given pixel maintains a set of mean pixel representations, called *cells*. Each cell $Cell_{i,t}$ contains three mean color component values $\mu_{i,t,x}$ that have been computed over t frames for each color component x . The number of cells per pixel depends on the multimodal nature of the pixel; i may range from 1 (for unimodal pixel values) to the number of modes of the pixel.

A pixel I_t in the current frame t matches a background cell computed from previous frames $Cell_{i,t-1}$ if the cell’s mean for each color component x is within a predefined threshold E_x of the corresponding color component of I_t :

$$\left(\bigwedge_x |I_{t,x} - \mu_{i,t-1,x}| \leq E_x \right) \quad (1)$$

(In our experiments described below, $E_x = 20$, for $x \in \{R, G, B\}$.)

In addition to the mean color component values, each cell contains a birthday B_i , which contains the frame number on which the cell was created, and an observation count $OC_{i,b}$ which keeps track of how many times the cell has been matched since it was created. These give information on the *age* of a cell $Cell_{i,t}$ at a given frame t :

$$Age_{i,t} = t - B_i \quad (2)$$

and the *observation density* of the cell:

$$OD_{i,t} = OC_{i,t} / Age_{i,t} \quad (3)$$

3.2 Midground Definition

Cell age and observation density are used to classify pixels as midground. In particular, a pixel I_t is considered to be in the midground if it matches a cell whose age is within a certain interval $[A_{min}, A_{max}]$ and its observation density is above an observation density threshold T_{OD} . That is, I_t is a midground pixel if it matches a cell $Cell_{i,t-1}$, such that:

$$\begin{aligned} A_{min} &\leq Age_{i,t-1} \leq A_{max} \\ OD_{i,t-1} &\geq T_{OD} \end{aligned} \quad (4)$$

The T_{OD} specifies how much occlusion to tolerate in detecting a midground object. In our experiments, we vary the observation density threshold as well as the frame rate (and therefore, the corresponding minimum and maximum age limits) to analyze their effects on midground detection, as described in Section 4.

3.3 Decimation for Efficient Adaptation

To avoid expensive floating point computations, each cell's means are represented as a running sum for each color component $S_{i,t,x}$ and a count $C_{i,t}$ of how many times a matching pixel value has been observed in t frames. Both components are periodically *decimated* to facilitate long-term adaptation (described below). At any given frame t , the mean color component value is computed as $\mu_{i,t,x} = S_{i,t,x} / C_{i,t}$. When a pixel I_t matches a cell, the background model is updated by adding each color component to the corresponding running sum $S_{i,t,x}$ and incrementing the count $C_{i,t}$ (and the observation count $OC_{i,t}$). As the background gradually changes, (e.g., due to lighting variations), the running averages will also adapt.

In addition, to enable long-term adaptation of the background model, all cells are periodically *decimated* by halving both the sum and the count every d (the decimation rate) frames. To be precise, when I_t matches a cell $Cell_{i,t-1}$, the cell is updated as follows:

$$\begin{aligned} S_{i,t,x} &= (S_{i,t-1,x} + I_{t,x}) / 2^b \\ C_{i,t} &= (C_{i,t-1} + 1) / 2^b \\ OC_{i,t} &= (OC_{i,t-1} + 1), \end{aligned} \quad (5)$$

where $b = 1$ if $t \bmod d = 0$, and $b=0$, otherwise.

Decimation is used to decay long-lived background components so that they do not permanently dominate the model, allowing the background model to adapt to the appearance of newer stationary objects or newly revealed parts of the background. Once a cell's count is decimated below a predefined cell threshold T_c , the cell is removed. This cleans up cells corresponding to previous midground or background elements that have disappeared from the scene. (In our experiments, $d = 100$ frames/decimation.)

3.4 New Cell Creation

Pixel values that do not match existing cells initiate a new cell creation in a special "seed" cell. When a pixel I_t does not match a cell, if either a seed cell does not yet exist or the current seed cell has a count less than the cell threshold T_c , then a new seed cell is created (the existing low-count seed cell is replaced). The seed cell's running sums are initialized to the color components of I_t , the count and observation count are initialized to 1, and the birthday is initialized to t . Once a seed cell is matched T_c or more times, it becomes a regular cell and any subsequent pixel that does not match a cell will create a new seed cell. (In our experiments, $T_c = 4$.)

3.5 Storage and Computation Requirements

Figure 1 shows the structure of a given cell, at frame t . The $S_{i,t}$ fields contain running sums of each color component, the $C_{i,t}$ field holds the decimated count while $OC_{i,t}$ holds

the observation count, and $Bday$ is the frame number at which the cell was created.

$S_{i,t,x}$	$S_{i,t,y}$	$S_{i,t,z}$	$C_{i,t}$	$OC_{i,t}$	$Bday$
-------------	-------------	-------------	-----------	------------	--------

Figure 1: *Pixel Cell Data Structure*

All fields are represented as 32 bit integers. P = number of sets (pixels), K = average cells per set, M = bytes per cell, and I = average instructions per cell per frame. The storage requirement for the background model is given as:

$$storage_{MG} = P \times K \times M \quad (6)$$

For the experiments reported in this paper, the average image resolution $P = 400,000$ with $K = 1.87$ cells/set average, and $M = 24$ bytes per cell (six words). This results in a representation storage requirement of **18 Mbytes**. Only integer operations are performed in this computation. The computational requirement for the midground model is defined as:

$$computation_{MG} = P \times K \times I \times fps \quad (7)$$

A more precise expression includes mid-set matching effects and decimation costs. However since sets are usually small and decimation is infrequent, this model is a good approximation. For the experiment conducted in this paper at one frame per second (fps), the computational requirement is approximately **15 Mops/second**. While other processes (image extraction, preprocessing, and post-midground processing) are not considered, the midground model computation and storage is well within the capabilities of today's embedded processors.

4. Experimental Results

Four test sequences are used to evaluate our midground detection technique (see Figure 2). The first sequence "Blocks" is an idealized sequence in which blocks are added and removed from a uniformly illuminated scene with no foreground activity or dynamic background. It is designed to be a "best case" for testing the algorithm. In the selected sequence, two background blocks are removed and a new block is added prior to the sampled frame. Only one true midground object should appear in the scene. In 'Outdoors', a busy outdoor scene includes dynamic background (waving trees, sky), many foreground objects, and two true midground objects (a trash can and a small paper bag) that are frequently occluded. "PETS" 2006 dataset S7 [11] includes a train terminal where a small bag is left unattended. i-Lids (AVSS) [12] is a tube (subway) station where a large bag is placed. In this pixel level evaluation, no attempt is made to determine whether the bag is unattended. Only a cell's age and observation density are used to determine its presence in midground. This technique can be extended to use this information with a suitable proximity algorithm.

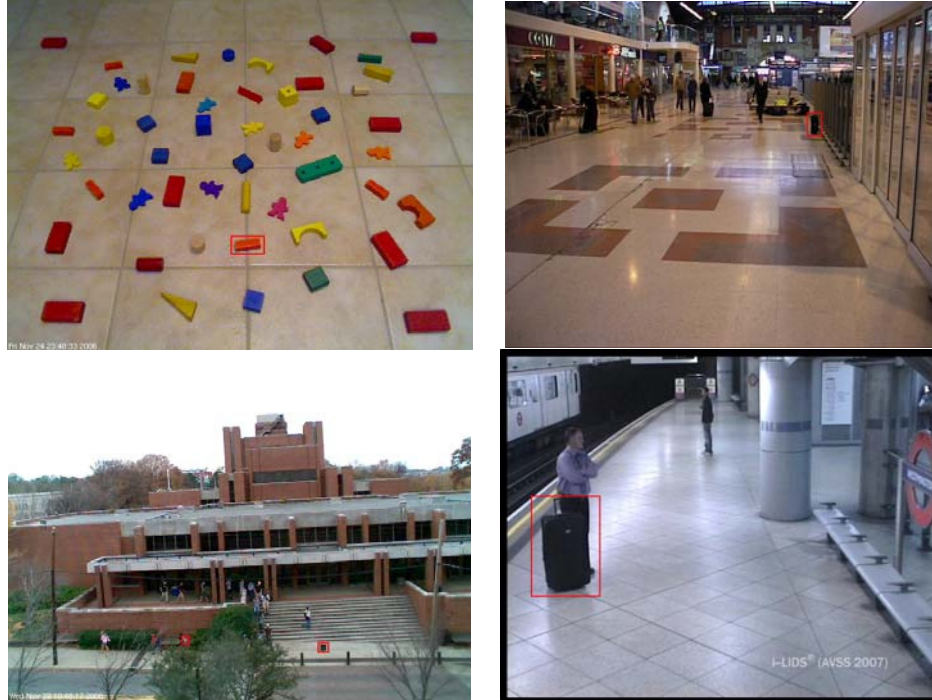


Figure 2: Test sequences with midground regions highlighted

Our experiments evaluate our midground modeling technique in a variety of real-world scenes. They specifically explore the effect on midground detection of varying the observation density threshold (which affects tolerance to occlusion) and of varying the frame rate.

The sequences listed in Table 1, include the sequence name, length (in seconds), frame rates, midground time window (in seconds), and number of midground objects. For the high frame rate PETS and AVSS sequences, the frames were downsampled to 1, 2, 4, 8, 16, and 25 frames per second. When experimenting with varying frame rates, the midground (MG) window in time is scaled by the frame rate to compute the MG frame window $[A_{min}, A_{max}]$ defined in Section 3.2.

Table 1: Test Sequence Summary

sequence	length	fps	MG win	MG objs
Blocks	800s	1	200-250s	1
Outdoors	1000s	1	200-250s	2
PETS	137s	1-25	20-40s	1
AVSS i-Lids	117s	1-25	20-40s	1

The effect of varying observation density threshold is shown in Figure 3. For each sequence the threshold is set from 30% – 90%. Lower threshold tests contain greater midground false positives resulting primarily from transient cell values of objects that have left the scene. In “Blocks,” the removed blocks appear in the midground at lower T_{OD} values since their corresponding cells satisfy the midground age requirements. At higher T_{OD} ,

they are excluded. A similar reduction in false positive noise is exhibited in the other sequences until 80%, above which false negatives begin to appear within the midground objects. All sequences are evaluated at 1 fps.

The impact of varying frame rate was explored in a second experiment, as shown in Figure 4. In these sequences, midground (defined with a constant temporal window and $T_{OD}=80\%$) is evaluated at frame rates of 1, 2, 4, 8, 16, and 25 fps. While the frame rate appears to have little effect on the PETS sequence, false positive noise is significantly reduced in the AVSS sequence as frame rate is reduced. Normally, higher frame rates reduce the impact of random image noise through averaging. In this sequence, the higher frame rates created more permanent objects (by exceeding T_C). Since computation is linearly proportional to frames processed, it is highly desirable to operate at the lowest possible frame rate for a given application and scene.

Since the MG window is an independent parameter, it can be adjusted to precisely meet the application requirements. In “Outdoors”, the window has been evaluated over relatively long windows (tens of minutes) whereas the transportation sequences typically require shorter windows (tens of seconds). Since the representation does not explicitly include the temporal window or T_{OD} , the evaluation function can include additional factors in midground determination. For example, human proximity information combined with a bag’s midground representation to trigger an alarm following a prescribed interval.

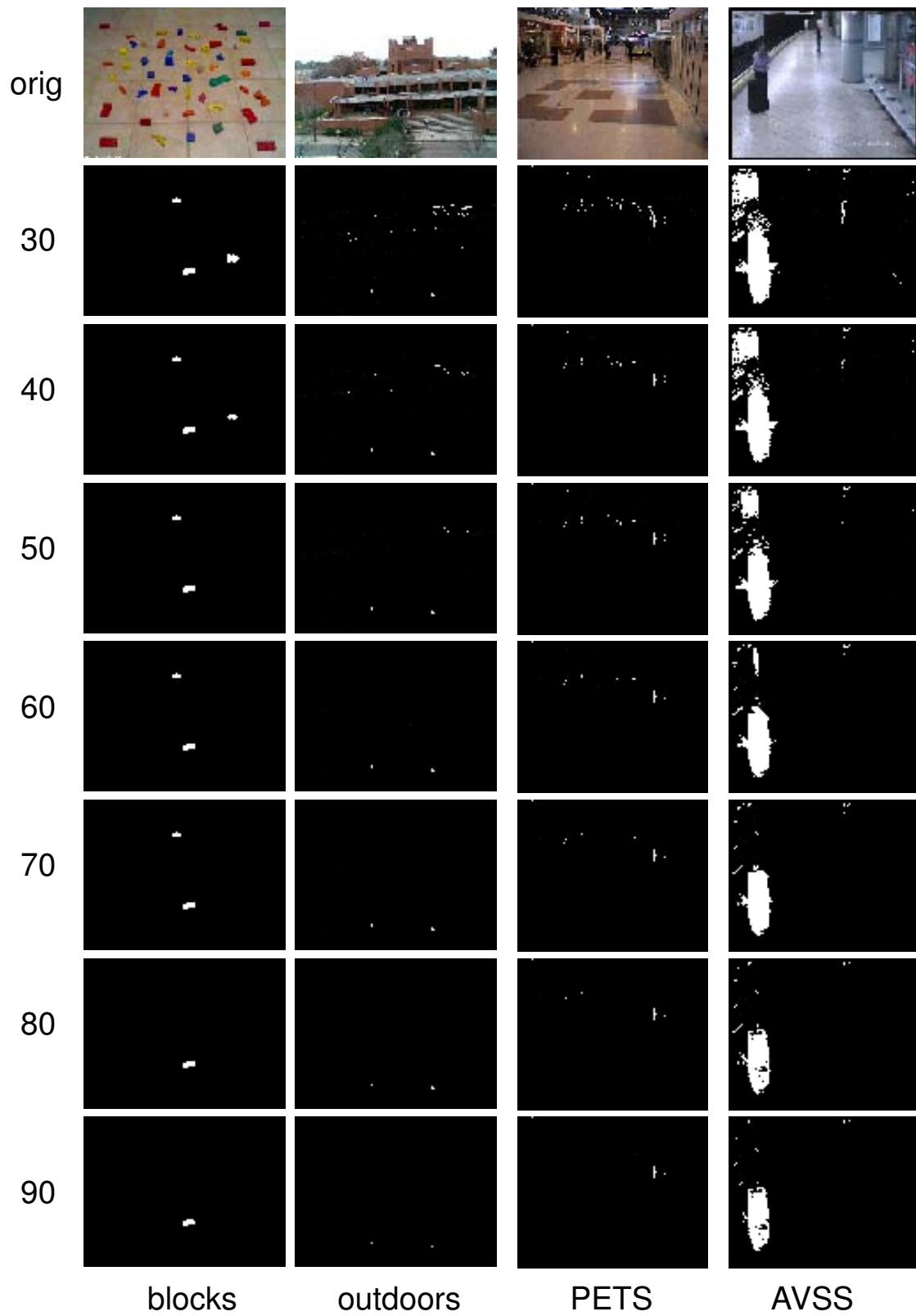


Figure 3: *Effect of observation density threshold*

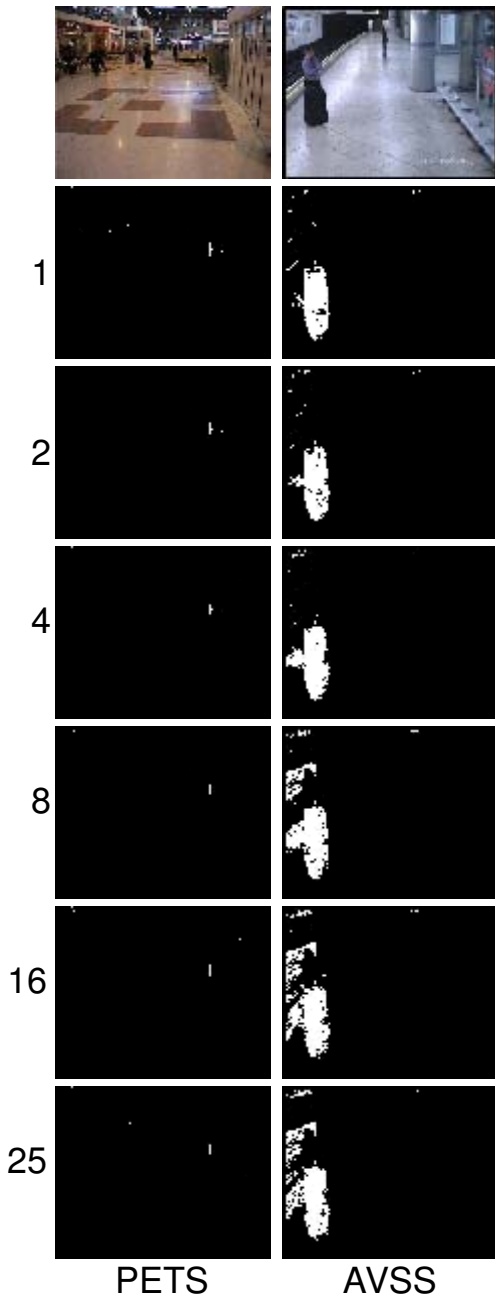


Figure 4: Effect of frame rate

5. Conclusions

This paper presents a new technique for precisely selecting midground regions in a scene based on a temporal window. It builds on a powerful but efficient multimodal model which has modest storage and computational requirements. For a 400,000 pixel image at one fps, the algorithm requires only 18 Mbytes storage and 15 Mops/sec processing throughput. Using realistic test sequences, this technique is shown to detect

midground objects with high accuracy. These tests show that a high T_{OD} ($>80\%$) can be used to minimize false positives. The technique performs well at relatively low frame rates (~ 1 fps) suggesting that it can be applied to a downsampled image stream for greater efficiency.

This midground detection technique can serve as a valuable complement to more complex scene analysis and event detection methods. Its low cost and high accuracy can provide scene modeling in addition to traditional foreground and background information. Ongoing research explores ways to adapt this technique to more difficult imaging environments (e.g., rapid illumination variations). Work is underway to *link* corresponding cells which vary only in illumination to better respond to scene complexities.

References

- [1] Cheung, S. and Kamath, C. "Robust techniques for background subtraction in urban traffic video," *Video Comm. and Image Processing*, Vol. 5308, pp. 881-892, SPIE Electronic Imaging, San Jose, Jan. 2004.
- [2] Radke, R.J., Andra, S., Al-Kofahi, O., Roysam, B., "Image change detection algorithms: A systematic survey," *IEEE Trans. on Image Processing*, 14(3):294-307, March 2005.
- [3] Piccardi, M., "Background subtraction techniques: a review," *IEEE International Conference on Systems, Man and Cybernetics*, Vol. 4, pp. 3099-3104, October 2004.
- [4] Kiratiratanapruk, K., Dubey, P., and Siddhichai, S., "A gradient-based foreground detection technique for object tracking in a traffic monitoring system" *Proc. of AVSS*, pp. 377-381, Sept. 2005.
- [5] Toyama, K., Krumm, J., Brummitt, B., and Meyers, B., "Wallflower: Principles and Practices of Background Maintenance," in *Proc. of ICCV (1)*, pp. 255-261, 1999.
- [6] Stauffer, C. and Grimson, W. E. L., "Learning Patterns of Activity Using Real-Time Tracking," *IEEE PAMI*, 22(8), pp. 747-757, Aug. 2000.
- [7] Appiah, K., and Hunter, A., "A single-chip FPGA implementation of real-time adaptive background model," *IEEE International Conference on Field-Programmable Technology*, pp. 95-102, December 2005.
- [8] Mathew, R., Yu, Z., and Zhang, J., "Detecting New Stable Objects in Surveillance Video," *IEEE Workshop on Multimedia Signal Processing*, pp.1-4, Oct. 2005.
- [9] Sacchi, C., and Regazzoni, C.S., "A distributed surveillance system for detection of abandoned objects in unmanned railway environments" *IEEE Trans. on Vehicular Technology*, 49(5), pp. 2013-2026, Sept. 2000.
- [10] Apewokin, S., Valentine, B., Wills, S., Wills, L., and Gentile, A., "Multimodal Mean Adaptive Backgrounding for Embedded Real-Time Video Surveillance," *Embedded Computer Vision Workshop (ECVW07)*, June 2007.
- [11] PETS 2006 dataset, Sequence Name S7-T6-B: Video 1, online: <http://www.cvg.rdg.ac.uk/PETS2006/data.html>.
- [12] i-Lids dataset for AVSS 2007, Available online at: www.elec.qmul.ac.uk/staffinfo/andrea/avss2007_d.html.