Machine Vision and Applications (2007) 18:41–64
DOI 10.1007/s00138-006-0047-x

ORIGINAL PAPER

An adaptive focus-of-attention model for video surveillance and monitoring

James W. Davis · Alexander M. Morison · David D. Woods

Received: 16 June 2006 / Accepted: 2 September 2006 / Published online: 13 October 2006

Abstract In current video surveillance systems, commercial pan/tilt/zoom (PTZ) cameras typically provide naive (or no) automatic scanning functionality to move a camera across its complete viewable field. However, the lack of scene-specific information inherently handicaps these scanning algorithms. We address this issue by automatically building an adaptive, focus-of-attention, scene-specific model using standard PTZ camera hardware. The adaptive model is constructed by first detecting local human activity (i.e., any translating object with a specific temporal signature) at discrete locations across a PTZ camera's entire viewable field. The temporal signature of translating objects is extracted using motion history images (MHIs) and an original, efficient algorithm based on an iterative candidacy-classification-reduction process to separate the target motion from noise. The target motion at each location is then quantified and employed in the construction of a global activity map for the camera. We additionally present four new camera scanning algorithms which exploit this activity map to maximize a PTZ camera's opportunity of observing human activity within the camera's overall field of view. We expect that these efficient and effective algorithms are implementable within current commercial camera systems.

Keywords Computer vision · Machine vision · Motion detection · Surveillance · Security · Camera scanning · Motion history images

J. W. Davis (✉) · A. M. Morison
Department of Computer Science and Engineering,
Ohio State University, 2015 Neil Avenue,
Columbus, OH 43210, USA
e-mail: jwdavis@cse.ohio-state.edu

A. M. Morison
e-mail: morisona@cse.ohio-state.edu

D. D. Woods
Cognitive Systems Engineering Laboratory,
Institute for Ergonomics, Ohio State University,
Columbus, OH 43210, USA
e-mail: woods.2@osu.edu

1 Introduction

In the area of security and surveillance, many types of sensors (e.g., video, acoustic, seismic) exist to provide information about the current state of the world. In particular, video cameras perhaps provide the most rich and useful information as compared with other sensor technologies. This richness of information accounts for the pervasiveness of video cameras within our society. Video cameras are employed everywhere for monitoring traffic flow, overseeing border control, providing indoor security, and enhancing outdoor surveillance.

In security surveillance specifically, there exists a large population of video surveillance cameras which provide pan/tilt/zoom (PTZ) functionality. The need for high coverage with low cost makes PTZ cameras an ideal, flexible solution, for this application. Surveillance coverage, nevertheless, is still difficult. In particular, indoor and urban environments are difficult from a coverage perspective where scene structure (e.g., buildings, trees) can limit camera coverage. Some solutions employ master–slave camera systems or omnidirectional cameras to combat coverage issues. However, most commercial surveillance systems currently consist of individual PTZ cameras. As a result, in this work we focus on enhancing current functionality for single PTZ cameras.

In addition, master–slave camera solutions necessarily increase the number of video streams (for master–slave systems) or have reduced functionality (no zoom for omnidirectional cameras). Increasing the number of cameras is not an appropriate solution because, in surveillance, the number of observable video streams is often already greater than the attention capacity of surveillance staff. This attention bottleneck changes the nature of surveillance from an anticipating, proactive agency to a reactive process plagued by surprise. In addition, the elimination of zoom capability directly effects surveillance staff which typically use the capability to obtain multiple views of regions of interest to gain the maximum quantum of information possible (e.g., a wide view for scene structure and current surrounding activity, a narrow view for faces and license plate numbers). Our approach attempts to reduce the attention bottleneck (introduced by large numbers of video streams) and is designed for use with standard PTZ camera systems in both new and pre-existing security surveillance installations.

In order to reduce human attention demands without eliminating functionality, many PTZ camera systems provide basic or naive automatic scanning technology, which sends each camera on a repeated predefined path. Example scan mechanisms may include, Continuous Pan, Frame Scan, Random Scan, and Recorded Path Scan (see Table 1). This technology attempts to relieve attention demands by reducing the need for human–camera interaction; this is a useful direction in the ideal case of cameras with a large, uniform interest, coverage area (e.g., a parking lot). Although in more complex environments, these generic algorithms typically result in sub-optimal scanning of the space due to camera position (e.g., mounted on pole, building roof, underneath overpass), obstructing scene structure (e.g., buildings, trees), or non-uniform user interest across a scene (e.g., high security areas vs. low profile areas). Each of these factors reduces user attention due to the mis-sampling of the complete scene (i.e., higher than desired sampling of low interest areas and lower than desired sampling of high interest areas). In our discussions with security staff, this forces personnel to adapt in unpredictable ways (e.g., turn the functionality off, sample the video stream less frequently) to overcome the hobbling effects of these algorithms, even though these methods were designed to "help" personnel.

Table 1 Current automatic camera scanning technology from Pelco [25]

Automatic scanning technology	Description
Continuous Pan	Camera pans clockwise continuously at a user determined rate (deg/s). The tilt position is constant and is determined by the current tilt position when the option is initiated.
Frame Scan	Camera pans clockwise one frame (i.e., the number of degrees in the current field of view), pauses for a user determined time, and then repeats. The tilt position is constant and is determined by the current tilt position when the option is initiated.
Random Scan	Camera pans in a random direction for a random number of degrees, pauses for a random quantity of time and then repeats; the tilt position is constant and is determined by the current tilt position when the option is initiated.
Recorded Path Scan	A user recorded path scan is stored in memory. The camera runs the scan path exactly as recorded and then repeats.

In this work we develop a more scene-specific, adaptive, focus-of-attention camera navigation model for video surveillance by automatically learning locations of high activity and directing the camera to sample these areas more frequently. The algorithm detects targets from a single motion history image (MHI) [3] at each local view from the full viewable field of a PTZ camera. This information is then accumulated over time at each local view to create a scene-specific model of interest (activity map). We also present several new scanning algorithms which take advantage of this activity map to efficiently navigate a PTZ camera around its entire viewable field, maximizing the opportunity for observing human activity based on prior observations. This naturally eliminates the issues associated with the aforementioned current scanning technology (i.e., uniform sampling across an entire scene containing a non-uniform distribution of human interest). Currently, our system does not consider multiple zoom factors; however, the work presented here naturally extends to multiple scales and is discussed in Sect. 7.

We begin with a review of computational aspects related to our approach, including human activity detection/recognition, outdoor scene modeling, and focus-of-attention models for surveillance (Sect. 2). We then present an overview of our approach (Sect. 3). Next, we provide a detailed description of our algorithm for measuring human activity using a iterative candidacy-classification-reduction technique for MHIs, which is then accumulated over time into a single activity map (Sect. 4). Next, we describe possible applications of the activity map to new camera scan path navigation methods (Sect. 5). We then provide experimental results for the detection (candidacy-classification-reduction approach), the resulting activity map, and a performance evaluation of the camera navigation algorithms (Sect. 6). Then, we discuss limitations of the current activity map and possible extensions (Sect. 7). Lastly, we provide a summary and comment on future directions of the work (Sect. 8).

2 Related work

Our work spans several areas of computer and machine vision including human detection/activity recognition, scene modeling, and scene scanning as focus-of-attention. We briefly examine related work in each of these areas and conclude with a comparison to our proposed approach.

2.1 Human activity detection and recognition

In this work we define human activity as any human generated motion (e.g., walking pedestrian, biking pedestrian, moving vehicle). In particular, we detect human activity by recognizing characteristic translating motion patterns. In general, however, human activity can range from basic presence in a scene to specific actions (e.g., person walking or entering a building).

Many techniques exist for blob-based person detection and most rely on a background model of simpler or greater complexity. Methods employing background subtraction include a single or multi-modal Gaussian background model [28,32], a two-stage color and gradient method [14], a Markov Chain Monte Carlo method [34], and a contour saliency map (CSM) approach which combines a standard background subtraction with a contour completion algorithm [10]. Alternatively, detection techniques can take advantage of motion or temporal change. Specifically, temporal differencing has been used by [16] to classify objects as either humans or cars using dispersedness and area features. More recently, work by [33] uses MHIs forward and backward in time simultaneously to extract object silhouettes for each frame. Lastly, template-based methods are another well explored and popular area, and includes techniques such as AdaBoost, where an ensemble classifier is created using rectangular filters on intensity and difference images to extract pedestrians [30] and further studied by [9] to automatically learn the optimal ensemble of filters. In [6], histogram of oriented gradient (HOG) templates are used for pedestrian detection. Futhermore, pedestrian detection can be accomplished using coarse-to-fine matching of edge templates [12], or using wavelets in combination with support vector machines (SVM) to learn characteristic pedestrian templates for detection [24].

Beyond mere person detection, the domain of human activity recognition includes a broad range of approaches, which are succinctly described in [1,4,11, 31]. Specific categories of approaches for activity recognition include, frequency-based Fourier analysis [17], temporal differencing (object and temporal classification) [16], feature-based properties (e.g., stride) [7], spatio-temporal patterns [22], spatio-temporal templates from image sequences [2,18], and Hidden Markov Models (HMMs) [5]. All of these activity recognition techniques exploit temporal information to identify categories of human motion.

2.2 Scene analysis and modeling

Current scene modeling and analysis techniques can be split into two distinct categories. The first category performs scene segmentation using scene structure based on user-defined rules (e.g., [29]), or use additional knowledge gained through user-assisted classification of examples based on information such as, color, texture, and patterns (e.g., [20]). The second category uses semantic-based information, such as pedestrian entry points, pedestrian exit points, and pedestrian pathways, by tracking pedestrians and clustering resulting trajectories, start positions, and end positions [15,19,28]. All of these models extract high-level scene information using scene properties (e.g., structure or human pathways).

2.3 Scene scanning as focus-of-attention

As discussed previously, current PTZ cameras provide a dramatic increase in functionality, which raises a fundamental issue. If a camera is directable over a large viewable field but has a small observable window (i.e., can monitor only one of many locations at a time), where *should* the camera look? This question is addressed by recent work in scene scanning, using focus-of-attention as a model. These systems utilize two cameras simultaneously (master–slave camera systems) to provide surveillance personnel a focus and a surround. The focus, or “slave” view, in most systems is implemented with a PTZ camera which is able to acquire a close-up view of a location designated by the “master,” or overview camera with a large fixed field of view. Current systems implement the master camera, which provides the surround view, with either a very large field of view static camera [35] or an omnidirectional camera [23].

2.4 Relations to proposed approach

Our proposed approach creates a global scene model for a single PTZ camera by dynamically measuring local

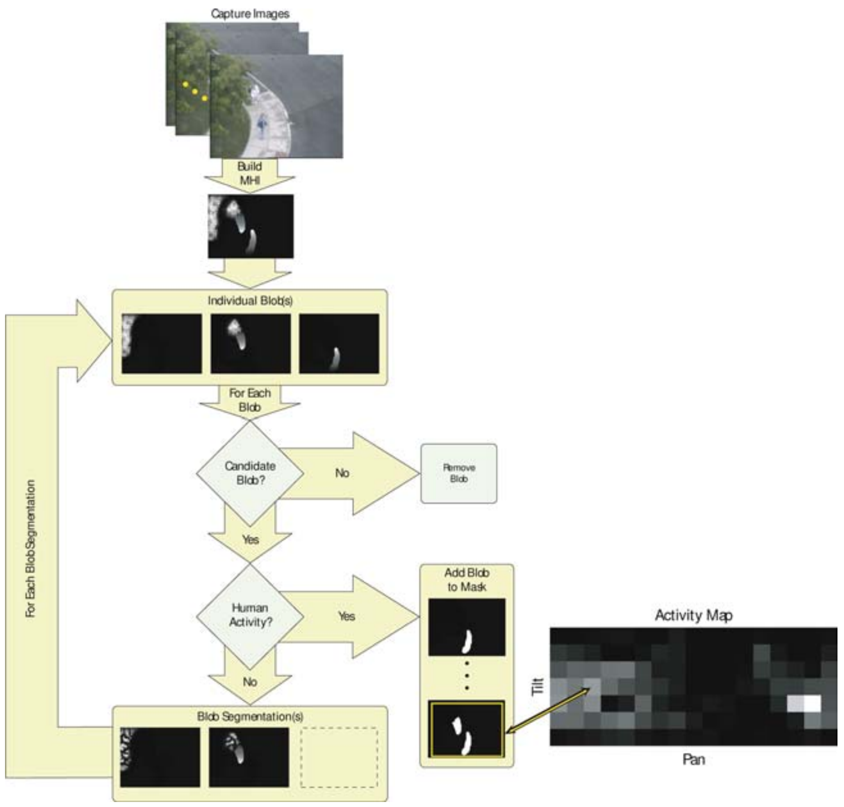

Fig. 1 Conceptual local/global overview

human activity using MHIs to build a single activity map, which is then used as an input to a set of automatic camera navigation algorithms.

The proposed approach differs from previous work to address person/activity detection shortcomings related to noise sensitivity, background modeling, and specific templates. However, other work related to human activity detection/recognition, as described earlier, could be employed in our framework. Further, current scene modeling methods typically require some form of user interaction. This is not necessary with our system, although user interaction is possible. Finally, focus-of-attention systems typically require multiple cameras that increases cost and may demand more user attention. Our approach uses only a single camera.

3 Algorithm overview

Our complete algorithm is composed of two distinct modules, which are a local/global detection algorithm and an activity-based navigation algorithm, as shown in Figs. 1 and 2, respectively.

The detection algorithm shown in Fig. 1 begins by capturing a short sequence of images from a single pan/tilt camera position and generating an MHI. Then, using connected components, we separate the MHI into individual blobs, which are analyzed independently using an iterative process. The process begins by validating each blob for candidacy as a target translating object (i.e., must have minimum spatial size and temporal length). All blobs that fail candidacy are immediately removed from the MHI. If, however, a blob qualifies as a candidate, the blob is passed to the classification stage where it is evaluated for translating motion. An MHI blob classified as translating motion is added to the final segmentation image for the image sequence. If instead the MHI blob is classified as noise, then the MHI blob is passed on to a reduction stage, which removes those pixels least likely to compose translating object motion. The reduction stage is necessary as misclassification can stem from the connection of a valid translating object region with a noise region. The reduction operation partitions the blob based on MHI gradient magnitude (measures the rate of temporal change between neighboring pixels), resulting in either a single MHI blob or multiple MHI blobs of smaller size. In either case, any remaining MHI blobs are passed back to the candidacy stage where the process repeats. This candidacy–classification–reduction iteration continues on a blob until the blob is either classified as translating motion (and added to the final segmentation image) or is removed by the candidacy stage. The local algorithm is complete when all MHI blobs are classified as translating motion or removed. A single activity measurement (summation of motion pixels) is then computed on the final segmentation image.

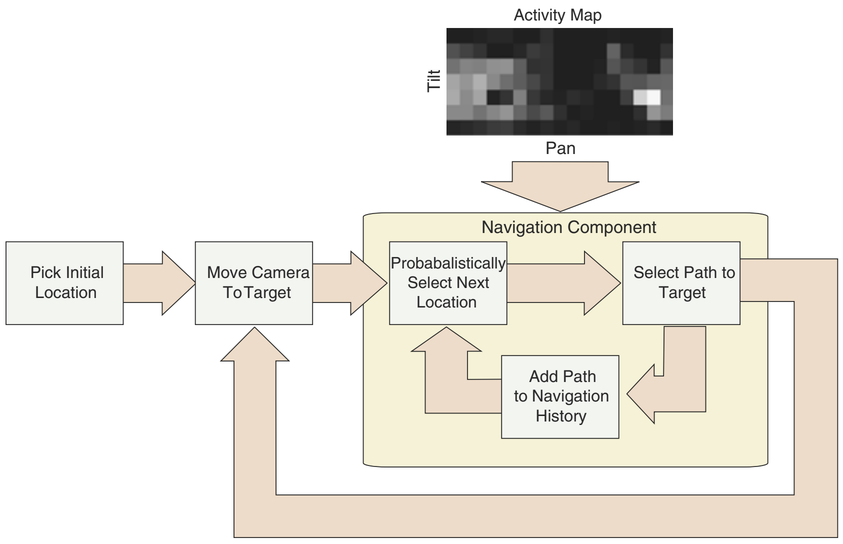

Fig. 2 Conceptual navigation overview

The detection process and resulting activity measurements are collected at each location across the entire view space to create a global activity map. The activity map is populated with activity measurements by visiting each discrete pan/tilt location in a random order and performing the local detection algorithm. The activity map is then refined using several passes of the scene.

The navigation algorithm incorporates the information from the constructed activity map to direct the focus of the camera through the scene. A generic navigation algorithm is depicted in Fig. 2 and begins with selecting an initial starting location and moving to this position. Next, we select a new location to move the camera. Here, the selection is guided by the particular scanning algorithm (we provide several approaches later) and the activity map. Depending on the algorithm, an "optimal" path to the new location is also selected based on the underlying activity map. Both the target location and path are stored in a history buffer to prevent the camera from moving back to the same location too quickly (we quantify this later for comparison of the different navigation methods). The algorithm then selects a new location and the above process repeats.

4 Measuring human activity

In order to measure human activity at a fixed camera position (i.e., pan/tilt, currently at a fixed zoom), we employ a series of steps to extract translating object regions in a single MHI. In Sect. 4.1, we introduce categories of motion within MHIs. We then describe the algorithm for detecting the human activity category, beginning with capturing a sequence of frames and generating the MHI (Sect. 4.2). Following this we extract individual blob regions and examine them for candidacy (Sect. 4.3). Next, we use the MHI representation to classify all regions as translating objects or noise (Sect. 4.4). Due to the possible connection of translating object motion and noise, we incorporate a reduction step, which separates translating object motion from noise (Sect. 4.5). The result from the reduction step is then returned to the candidacy stage for re-evaluation. This process continues until all MHI pixels are classified as belonging to a translating object or removed. The final segmentation image is then quantified into a single activity measurement. The local activity measurements at different locations are then used to form a global activity map (Sect. 4.6).

4.1 Categories of motion

Examination of MHIs from a commercial PTZ video surveillance camera tend to show three distinct categories of MHI patterns or signatures. We label these general categories as human activity, environmental noise, and camera noise. Examples of each type of motion are provided in Table 2. In our categorization we distinguish environmental noise from camera noise; however, camera noise is a consequence of interaction between the camera sensor technology (CCD sensor), scene structure (spatial patterns), and transmission format (i.e., NTSC). Therefore, overlap can occur between environmental and camera noise, although the two are conceptually separate and unique. For example, a tree shaking

Table 2 Examples of human activity, environmental noise, and camera noise

Human activity	Environmental noise	Camera noise
Pedestrian walking	Tree shaking	Brick work
Person biking	Smoke/steam	Building edges
Moving vehicle	Reflections	Lamp posts

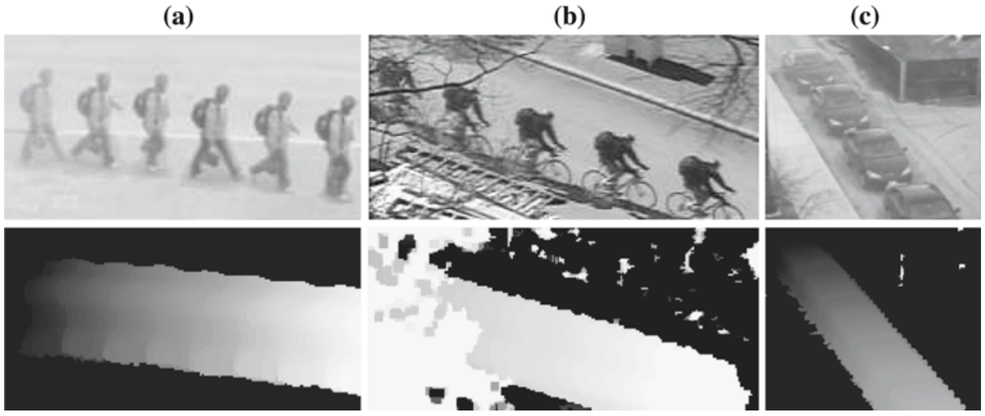

Fig. 3 Timelapsed images and raw MHIs for classes of Human Activity. **a** Pedestrian, **b** cyclist, and **c** moving vehicle

due to wind is considered as environmental noise, and differs from a still tree with NTSC edge aliasing (of leaves and branches), which we consider as camera noise.

Our algorithm attempts to separate human (or human generated) activity from environmental and camera noise. In typical surveillance videos, the target motion usually corresponds to pedestrians, groups of pedestrians, moving vehicles, cyclists, rollerbladers, skateboarders, etc. In this work, we define human activity as any translating object with a minimum spatial size and temporal length. In Fig. 3 we provide a few examples of such human activity using a timelapsed image and corresponding MHI. The MHI provides a compact motion representation that visually captures the motion trail of translating objects over time. Rather than using a specific activity detector (e.g., pedestrian templates), we feel that an MHI-based approach can be simple and effective at separating general human activity patterns from the remaining noise categories. Certainly other color and texture patterns could also be employed in conjunction with the MHI to help identify the target regions, and this additional information will be addressed in future work. Currently, we examine the usefulness of a motion-based representation. We reemphasize here that we are including more pedestrians in our definition of human activity, which is desirable and motivated from interviews and discussions with security surveillance personnel.

4.2 Motion history image

An MHI is a single image which represents extended temporal motion (>2 frames). Typically, an MHI uses either background subtraction or temporal differencing to extract frame-to-frame motion. In this work, temporal differencing is used to extract motion between subsequent temporal frames. An MHI is computed as follows. Let $I_{\text{seq}} = [I_1(x,y)\cdots I_N(x,y)]$ be an image sequence of length N. Let $D_{\text{seq}} = [D_1(x,y)\cdots D_{N-1}(x,y)]$ contain the temporal differencing of I_{seq}, where

$$D_t(x,y) = I_{t+1}(x,y) - I_t(x,y) \tag{1}$$

For RGB images, one could use pixel-wise Euclidian distance. We then binarize the differencing result with

$$B_t(x,y) = \begin{cases} 1 & \text{if } |D_t(x,y)| > T_{\text{Diff}} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

A low T_{Diff} threshold is selected to increase sensitivity to all possible motion, which, of course, results in a larger quantity of included noise. The tradeoff of more noise for higher sensitivity is offset by the recursive nature of the algorithm to be discussed in Sect. 4.5. Combating increased noise is also assisted by removing regions below a minimum size ($T_{\text{MinFrameSize}}$) from each image in $B_t(x,y)$. In addition, to strengthen larger regions, we perform a single close morphological operation to fill the interior of all binary blobs in $B_t(x,y)$. The binary sequence is used to update the MHI for each timestep (t) as follows:

$$\text{MHI}(x,y) = \begin{cases} t & \text{if } B_t(x,y) == 1 \\ \text{MHI}(x,y) & \text{otherwise} \end{cases} \tag{3}$$

An example of a walking person is shown in Fig. 4a using a time-lapse image (higher intensity is more recent in time), with the corresponding MHI shown in Fig. 4b. The long trail of the person in the MHI visually captures a basic property of translating objects. We refer to this property as "temporal consistency", in that, a translating object within an MHI will have a trail consisting of an intensity fade of extended temporal length. Furthermore, an intensity fade for a semi-rigid, constant velocity, translating object will have equal quantities of all MHI timestamps. In addition, we conjecture that non-translating objects, or noise, will not exhibit strong

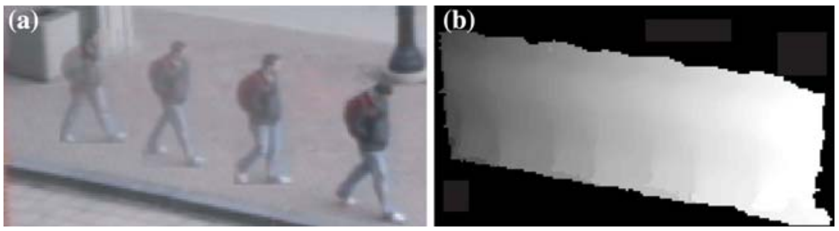

Fig. 4 Example MHI. **a** Time-lapse image and **b** MHI of a person walking across a scene

temporal consistency, given the nature of noise sources (generally static). We therefore, use temporal consistency to classify each MHI blob as a translating object or noise. In the ideal case of a blob consisting of only a translating object with minimal contributing noise (we describe later how we manage significant noise contribution), the classification process is as follows.

4.3 MHI candidacy

First, for each MHI blob, we determine whether the blob is a potential candidate for human activity (i.e., translating motion). We define blob candidacy with two properties of temporal consistency previously discussed, which are a minimum spatial size (i.e., $size(\text{MHI\_blob}) > T_{\text{SpatialSize}}$, e.g., 400 pixels for a 320×240 image) and a minimum temporal length (i.e., $Duration(\text{MHI\_blob}) > T_{\text{TemporalLength}}$, e.g., 0.5 s). If an MHI blob fails either of these two criteria then the blob is not considered a candidate for human activity and is removed from the MHI. These thresholds can be tightened or relaxed depending on the application to identify more or less temporally/spatially significant signatures, respectively. All blobs that are selected as candidate human activity are passed to the classification stage for evaluation of translating motion.

4.4 MHI classification

In this stage, we examine the intensity fade of each MHI blob (timestamp distribution). For classification, if we assume that an MHI blob is indeed human activity (and therefore is a valid translating object), then the resulting MHI blob will have a trail of relatively equally spaced (i.e., spatially and temporally) timestamps. For example, consider the ideal case of a square moving left-to-right across a scene with constant velocity. The histogram of this ideal MHI blob would show equal quantities of each timestamp in the blob (i.e., uniform distribution within $0 < t < N - 1$). This timestamp distribution, however, is dependent on the MHI blob size. Therefore, for comparison of temporal histograms of different blobs, we normalize the histogram by the size of the MHI blob.

Classification of a candidate MHI blob (translating vs. noise) is accomplished using a histogram similarity measure to quantify the degree of match between the candidate MHI blob timestamp distribution and the ideal timestamp distribution (uniform) for that blob. The resulting similarity measure must be greater than a threshold value $T_{\text{HumanActivity}}$, for the MHI blob to be classified as human activity. We provide experiments later examining several different similarity measures and thresholds (Sect. 6.1).

The previous candidacy and classification approaches are designed for classifying MHIs in the presence of minimal noise. In actuality, however, MHIs are more complex than this idealized situation due to the presence and overlap of environmental noise (e.g., shaking trees, illumination changes) and camera noise (e.g., spatial frequency aliasing). These noise sources, when attached to the MHI intensity fade of a true translating object, will cause the classification algorithm to fail (i.e., classify as noise).

In Fig. 5 we provide a synthetic example of three possible classes of MHI blobs which are, (a) translating activity only, (b) noise only, and (c) an overlap of noise and translating activity. Using the classification technique previously described, the blob in Fig. 5a would be classified as translating activity (uniform distribution), while Fig. 5b, c would be classified as noise (not similar to a uniform distribution). Notice that Fig. 5b would be classified correctly as noise; however part Fig. 5c in some sense would be classified both correctly and incorrectly due to the presence of a large noise region. Unfortunately, classifying this MHI blob as noise and removing all of the blob pixels would eliminate the translating object. We need to remove the noise region to successfully detect the translating object. In Fig. 5d, we present the ideal extraction of the translating activity from Fig. 5c. The histogram of the segmented translating activity (Fig. 5d), is more similar to the histogram in (a) than in either (b) or (c), although is still not ideal due to the shape and overlap of the noise region. However, this may be handled by the classifier with an appropriate threshold. We therefore need a method to separate the blob into noise and translating object, from which we can then detect the translating object.

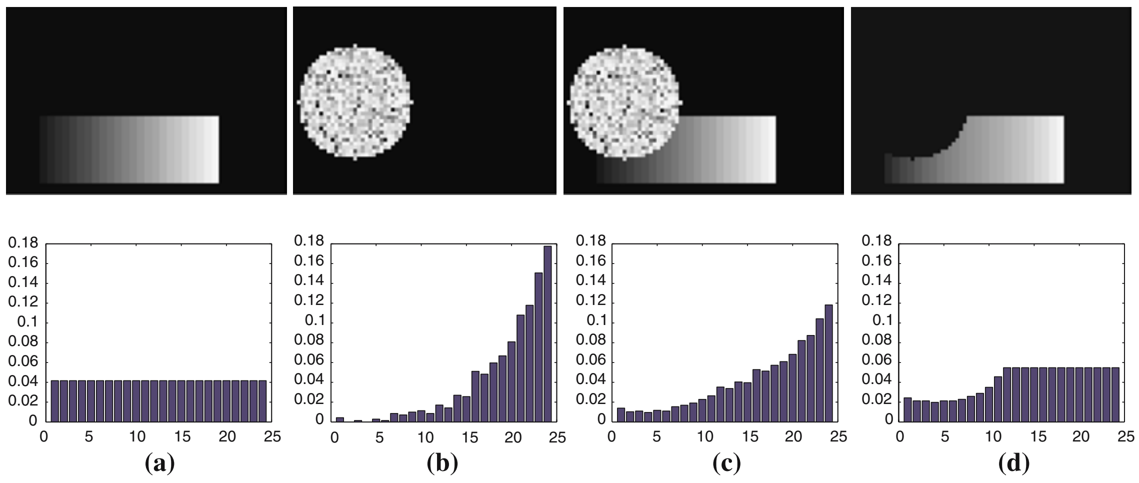

Fig. 5 Examples of MHI classification for **a** translating object, **b** non-translating object (i.e., noise), **c** combination of translating object and noise, and **d** translating object after reduction. *Top image* is the MHI and *bottom image* is the timestamp histogram

4.5 MHI reduction

This section is motivated, as stated previously, by the possibility of a candidate MHI blob simultaneously consisting of human activity and noise, resulting in the misclassification of the entire region as noise. We cannot, however, know *a priori* that an MHI blob classified as noise contains human activity or not and therefore we assume that all noise MHI blobs potentially contain human activity. Consequently, all noise MHI blobs are processed by this stage of the algorithm in an attempt to reduce all MHI blobs to human activity.

When human activity and noise regions are connected, and therefore are contained in a single MHI blob, we must find a method to segment the single region in such a way as to separate the noise from the human activity. With this goal in mind, we use the gradient magnitude image of the MHI blob (similar to the approach in [3]). The MHI gradient magnitudes can be used to identify areas of large temporal disparity between neighboring pixels, which is precisely what we expect to see at boundaries of objects and within noise regions. The gradient magnitudes provide a means to consistently remove pixels in order of confidence related to translating objectness and are the basis for our MHI reduction algorithm. Note that currently only the gradient magnitude (not direction) information is used.

For a single MHI, the gradient image is constructed by convolving Sobel masks (Fig. 6) with the MHI to obtain the F_x and F_y gradient images. The resulting gradient magnitude image is defined as $\sqrt{F_x^2 + F_y^2}$. Several issues may exist with this initial gradient magnitude image after these simple steps, and may include invalid gradient magnitudes at blob-background boundaries and zero

(a)

-1	0	1
-2	0	2
-1	0	1

(b)

1	2	1
0	0	0
-1	-2	1

Fig. 6 Sobel masks. **a** X-direction, **b** Y-direction

gradients within large regions of uniform timestamp. We first remove gradients larger than a threshold value T_{MaxGrad} (e.g., gradient magnitudes > 0.5 s). These large gradient pixels correspond to blob-background boundary pixels and some noise pixels. These pixels are removed from both the gradient image and MHI. We then eliminate the most current timestamp pixels from the gradient magnitude image and MHI, because these pixels do not form part of the MHI intensity fade. Finally, the zero gradient magnitude pixels (within regions of the same non-zero timestamp) are filled in using the following process. Each zero gradient magnitude pixel is assigned the average gradient value of the 8-connected neighbors having a gradient magnitude >0. Note, in order for a pixel to be filled, the pixel must necessarily have at least one non-zero gradient magnitude neighbor. Initially, some pixels may have all zero gradient magnitude neighbors and therefore we perform this process iteratively until all possible zero gradient magnitude blob pixels are assigned a non-zero value. Any zero gradient magnitude pixels remaining after this operation are removed from the MHI image.

The MHI reduction process uses the updated gradient magnitude information computed by the above process. In order to tightly control our reduction method, we select the maximum gradient magnitude for the given MHI blob and use this pixel as the seed/start pixel for

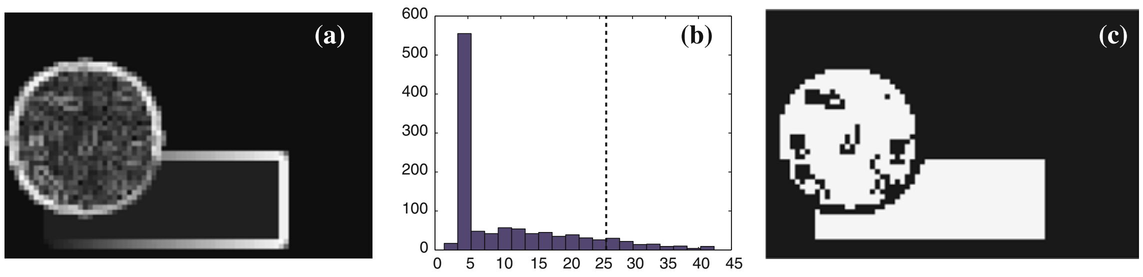

Fig. 7 **a** Gradient magnitude image of translating square with noise region. **b** Histogram of gradient magnitudes with 10% threshold. **c** Two regions resulting from reduction steps

the reduction process. From this seed pixel we recursively grow out 8-connected from the seed pixel to all other pixels in the gradient magnitude image with magnitude greater than a growing threshold (T_{grow}). We take a conservative approach at each reduction step, selecting a small T_{grow}, based on a small percentage (e.g., 10%) of the number of pixels in the current MHI blob. We believe this will provide a more accurate reduction as a smaller T_{grow} will result in a fewer number of pixels selected at each reduction step, and accordingly, finer resolution on the reduction. In Sect. 6.1, we provide experimental results for different T_{grow} values. Once growing is complete, all pixels collected by the growing operation are removed from the current MHI blob. After applying a connected components algorithm, there may be multiple MHI blobs. Each of the resulting blobs is returned to the candidacy stage (Sect. 4.3) for consideration as human activity. After the candidacy, classification, and reduction stages complete, all blobs are either classified as human activity or removed by the candidacy test. The result is a final binary segmentation image showing areas of human activity in the image.

To illustrate this process, we refer again to the ideal example of a box translating from left-to-right across a scene with an added noise region, as shown in Fig. 5c. In Fig. 7a, we show the raw gradient magnitude image of the MHI immediately after applying the gradient masks. Notice that the higher gradient magnitudes (higher intensity) reside within the noise region and on the blob and noise boundaries. In particular, the object-noise boundary is clearly visible in this image. After removing the initial invalid gradients and filling missing gradients, we select the seed pixel (current maximum gradient pixel). In Fig. 7b, we show a histogram of the gradient values and highlight the selected T_{grow} for this blob with a vertical dashed line. Finally, we show in Fig. 7c an intermediate binary result in the reduction process. This binary image shows the result immediately after the reduction algorithm completely separates the noise and translating object regions (after 10 reduction iterations). As the iterative process (candidacy–classification–reduction) continues to execute, the circular noise region is repeatedly classified as noise and returned to the reduction stage where pixels are removed, separating the region into many smaller subregions which are finally removed from the MHI based on the candidacy stage. The separated translating object region is returned to the candidacy stage and, with the correct selection of threshold values, the region will be classified as translating object and added to the final segmentation image.

We lastly convert the final binary segmentation image into a single activity measurement number to give a relative indication of the quantity of activity that occurred at that particular location. Possible measurements include counting the number of segmented blobs, counting the number of pixels, or counting a scaled number of pixels based on depth of view. For our work we selected a simple summation of pixels (total motion energy) due to the difficulty of, segmenting overlapping blobs, blob fragmentation due to noise/partial occlusion, and scaling difficulty without knowledge of the ground plane.

4.6 Global activity map

We presented a method to capture a coarse measurement of human activity at a single camera view by segmenting human activity (i.e., translating objects) from noise using MHIs. In order to use this local information for global camera navigation we must employ the local activity measurement at a set of discrete camera positions. We create an "activity map" for a camera's full viewable field using the local activity measure at several camera locations. In order to create this activity map, we divide the full field into $m \times n$, discrete pan/tilt locations, which naturally results in an $m \times n$ rectilinear activity map. Due to the PTZ camera and the rectilinear activity map, as the tilt angle decreases from 0° (horizontal) to −90° (vertical) the quantity of overlap between neighboring pan positions increases. Larger overlap results in higher emphasis of these locations (−90°). However, the rectilinear activity map simplifies construction (and the navigation algorithms in Sect. 5) and produces reasonable results. Building the map consists of visiting each location in the activity map in a random order, applying the activity detection algorithm at each location, and accumulating the activity measures over time. An example activity map is displayed in Fig. 8.

The time to complete one full pass of the activity map is dependent upon several factors including, the number of activity map locations, the digitization rate/duration (i.e., length of the sequence), and the processing time to segment the MHI. In our current implementation the number of activity map locations is 119 (7×17), and each video sequence (for each location) is collected at a rate of 12 Hz for approximately 6 s (75 frames). The median execution time to process a single activity map location using the recursive MHI reduction method implemented in Matlab running on a P4 2.8 GHz computer is <1s (calculated from 59 test images).

Once the activity map is initially constructed for a particular PTZ camera, we must consider updating the activity map overtime. In general, the activity map can be updated as often as desirable to capture the evolution of a given scene over time. Updating is an important and complex issue involving both when and how to include new activity information. For example, new activity information can be collected and updated continuously, only at selected times, based upon the activity measured, or using a combination of all of these. Currently, we collect data for one complete pass of the activity map and then update each location simultaneously by simply adding in the new measurements to the previous measurements for each location. This process is repeated over several hours/days. Additional considerations exist for online updating of the activity map during camera navigation. During automatic navigation, activity map locations are visited based on activity level which results in a disparity between number of visits for high versus low activity locations. This visitation disparity will result in slow (or no) online updating of low activity regions. In order to address this issue, modification to the automatic navigation will be necessary to visit low activity regions specifically for the purpose of online updating of the activity map. Possible modifications to the navigation include a periodic random jump to lower activity locations, performing one complete pass of the activity map at designated times, or selecting a maximum temporal threshold before a location must be revisited. The selection of how to continually update the map over time is application- and context-specific and is the focus of future work.

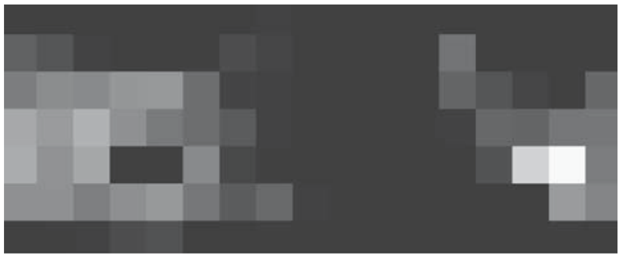
Fig. 8 A 7×17 activity map (*rows* correspond to tilt positions and *columns* correspond to pan positions) after 10 complete passes of the scene, where *brighter areas* correspond to locations with more activity

5 Activity-based camera navigation

In this section, we present several navigation methods that can exploit the activity map to improve scanning efficiency over a camera's complete viewable field as compared to current automatic scanning algorithms. For each of the methods we describe the activity map interpretation and the scanning procedure. The methods presented include probabilistic jump, probabilistic walk, inhibited probabilistic walk, and reinforcement learning paths.

5.1 Probabilistic jump

For this navigation method the activity map is normalized (activity map values sum to 1) and considered as a pseudo-probability distribution. Each subsequent location is selected using a probabilistic roulette wheel sampling of the activity map. Notice that as the number of random jumps (i.e., time) approaches infinity the percentage of samples at each location will converge to the normalized activity map. We limit the scanning memory/history to only the previous location. In other words, the probabilistic selection will not permit the algorithm to jump to the same location two times in a row. This is accomplished by probabilistically selecting a new location until the location selected is not equal to the current location. Once the next location is determined the camera moves directly (and quickly) to this new location and the algorithm repeats.

This navigation algorithm, converges to the activity map and therefore provides the maximum likelihood of observing human activity based on prior observation. However, this technique tends to be disorienting to some human observers due to the quick, and possibly distant,"jump" between consecutive locations. Eliminating the disorienting movement between consecutive locations motivates the following navigation algorithms.

5.2 Probabilistic walk

This navigation method is similar to the previous jump method, except that the next location is selected probabilistically based on the 8-connected neighborhood of the current location. This implicitly creates a path because the distance between a location and any neighbor is a straight line of unit distance (i.e., activity map resolution determines unit distance). Similar to probabilistic jump, the history is limited to only the previous location.

This navigation algorithm eliminates the quick "jump" between consecutive positions making this algorithm less disorienting to some human observers. One significant issue for this algorithm, however, is the tendency over short temporal windows to become stuck in local maxima. Correcting the issue of becoming stuck in local maxima motivates the following algorithm.

5.3 Inhibited probabilistic walk

The next navigation method is a variation of probabilistic walk and uses an inhibition/suppression mechanism to control the history. The approach was originally motivated by the saliency/saccade modeling method of [13]. The approach maintains an implicit history of recently visited locations using a spatio-temporal inhibition mask to decrease, and slowly recover, the normalized activity map values of the 5-connected neighbors in the opposite direction of the next location chosen (see Fig. 9). This inhibition mask forces the focus-of-attention away from recently visited areas by modifying (i.e., lowering) the probability of visiting these locations (i.e., reducing the activity map values). The probabilities are increased back to original values using an inverted exponential decay function

$$\text{Inhibit}(t) = 1 - \exp(-\alpha * (t - t_0)) \tag{4}$$

where t_0 is the time the location was initially inhibited, and α is the inhibition rate (e.g., $\alpha = 0.1$). Note, other functions can be used to create the suppression/recovery

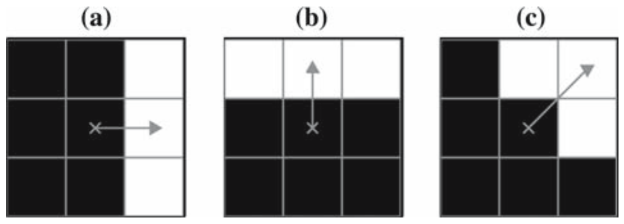

Fig. 9 Example inhibit patterns when the direction the camera moves is **a** horizontal, **b** vertical, **c** at an angle. *Black* represents the activity map locations that are inhibited and *white* represents locations that are not inhibited

such as linear and exponential; however, in our work we found the inverted exponential decay gave encouraging results.

This navigation technique eliminates the local maxima issue associated with the previous probabilistic walk method. However, this technique requires selection of the inhibition function and the parameter α which directly affects the resulting navigation.

5.4 Reinforcement learning paths

A well-known machine learning robot navigation method for determining "optimal paths" given rewards received along possible paths is reinforcement learning. The goal of reinforcement learning is to determine optimal actions given a set of rewards to maximize cumulative reward. Specifically, Q-learning [21] is an iterative method for learning a function (Q) which provides the optimal policy (i.e., the correct action at a given state) when rewards and actions are deterministic. The function Q gives a single number which indicates the discounted cumulative reward for all state-action pairs. In the case of a single location with large reward (a goal location), maximization of the function Q (i.e., selecting the state-action pair with maximum Q value at all states) will give the optimal path from any location to that goal location. The Q function for learning the optimal action at any state is given by

$$Q(s, a) = R(s) + \gamma \max_{a} Q(s', a') \tag{5}$$

where $Q(s, a)$ is the discounted cumulative reward for state s and action a, given reward function R, and the discounted (e.g., $\gamma = 0.9$) cumulative reward at the next state $Q(s', a')$. Additional details can be found in [21].

Our task of navigating a scene given an activity map is a natural domain for Q-learning, where the activity map is the reward function (R), the activity map locations are the states, and the move to any 8-connected neighbor is the set of possible actions. We simply select a set of M goal locations (G) based on a probabilistic selection of locations from the activity map. Then, for each goal location $g \in G$ we have a separate reward function R_g, which is the activity map modified to give g extremely high reward (extremely high activity). Each reward function R_g is then input in to the Q-learning algorithm. The Q-learning algorithm consequently finds the optimal path from each state s (pan/tilt locations in the activity map) to each $g \in G$ using R_g, which are then stored and used for navigation.

Navigation of the scene consists of probabilistically selecting $m \leq M$ goal locations based on the original activity map values. The subset m is compared against the goal location history. If one or more goal locations in m do not exist in the history then we randomly select one of these. If however, all goal locations in m are in the history, we select the "oldest" goal location in m according to the history. Once a new goal location is determined, the path is provided from the Q-learning results (performed offline prior to navigation). The goal location history is then updated with each goal location that is visited along the path from the current location to the new goal location.

This algorithm navigates the space using optimal paths between goal locations. However, the goal locations must be selected and the algorithm must retrain when the activity map is changed (training time can be considerable).

6 Experiments

In this section we examine the performance of our algorithms related to measuring local activity, generating global activity maps, and applying our navigation techniques with multiple cameras and varying views. In the local activity section (Sect. 6.1) we describe and provide examples of the results for human activity extraction and provide experimentally determined threshold values. Following this (Sect. 6.2), we examine global activity maps generated from three different cameras. Lastly (Sect. 6.3), the navigation techniques are compared using paths generated by each algorithm.

The specific equipment used consists of three Pelco Spectra III SE series dome cameras (see Fig. 10) [25] mounted outdoors on three different buildings on the main campus of Ohio State University. Two cameras have overlapping views and one camera has a completely disparate view. Additionally, the cameras are mounted at varying heights (two, three, and four stories). Images are captured with a Matrox Meteor frame-grabber using RGB at a size of 320×240 pixels with a frame rate of $\sim$ 12 fps. In Fig. 11a–c, we show spherical panoramas of the three camera views, each created by stitching together 119 separate local-view images, into a single viewable field for each of the cameras.

6.1 Measuring local human activity

In Sect. 4, we presented our MHI-based approach for segmenting translating motion from (noisy) scenes. The main thresholds of interest are the classification threshold (T_{class}) and growing threshold (T_{grow}) used for reduction. In this section, we compare the performance of different threshold values and similarity measures using manually segmented MHI data.

Fig. 10 Pelco Spectra III SE series camera

Due to the dependent structure of our algorithm (i.e., tightly coupled candidacy–classification–reduction stages) for extracting human activity, we evaluate the classification and growing thresholds together. The evaluation used a training set of 59 MHIs (generated from 59 different image sequences), each containing one or more MHI blobs (noise and object). These training sequences were collected across the three cameras, at different times-of-day and days-of-week. Example time-lapsed images (for display purposes only) are provided in the left-most column of Fig. 12, followed by the raw MHIs in column two. Note the different locations, orientations, and field of view. Then, in column three, we show the ground truth manual segmentations of each MHI.

In this work, we selected four histogram similarity metrics to evaluate as candidates for the classifier. The similarity measures selected were Bhattacharya distance, Jeffrey divergence (i.e., the symmetric form of Kullback–Leibler divergence), Minkowski-form distance, for Match distance (equivalent to Earth mover's distance for the class of 1D histograms [27]). For these histogram similarity measures, H is the size-normalized MHI blob timestamp distribution and K is the ideal MHI blob timestamp distribution (uniform). The metrics are defined as follows

$$D_{\text{Bhatta}}(H,K) = \sum_i \sqrt{(h_i)(k_i)} \tag{6}$$

$$D_{JD}(H,K) = \sum_i h_i \log \frac{h_i}{m_i} + k_i \log \frac{k_i}{m_i} \tag{7}$$

$$D_{\text{Mink}}(H,K,L) = \left(\sum_i |h_i - k_i|^L \right)^{\frac{1}{L}} \tag{8}$$

$$D_{\text{Match}}(H,K) = \sum_i |\widehat{h_i} - \widehat{k_i}| \tag{9}$$

where, $m_i = \frac{h_i+k_i}{2}$, $\widehat{h_l} = h_1 + h_2 + \cdots + h_l$, and $\widehat{k_l} = k_1 + k_2 + \cdots + k_l$. We selected the L_2 norm (L = 2) for the Minkowski-form distance.

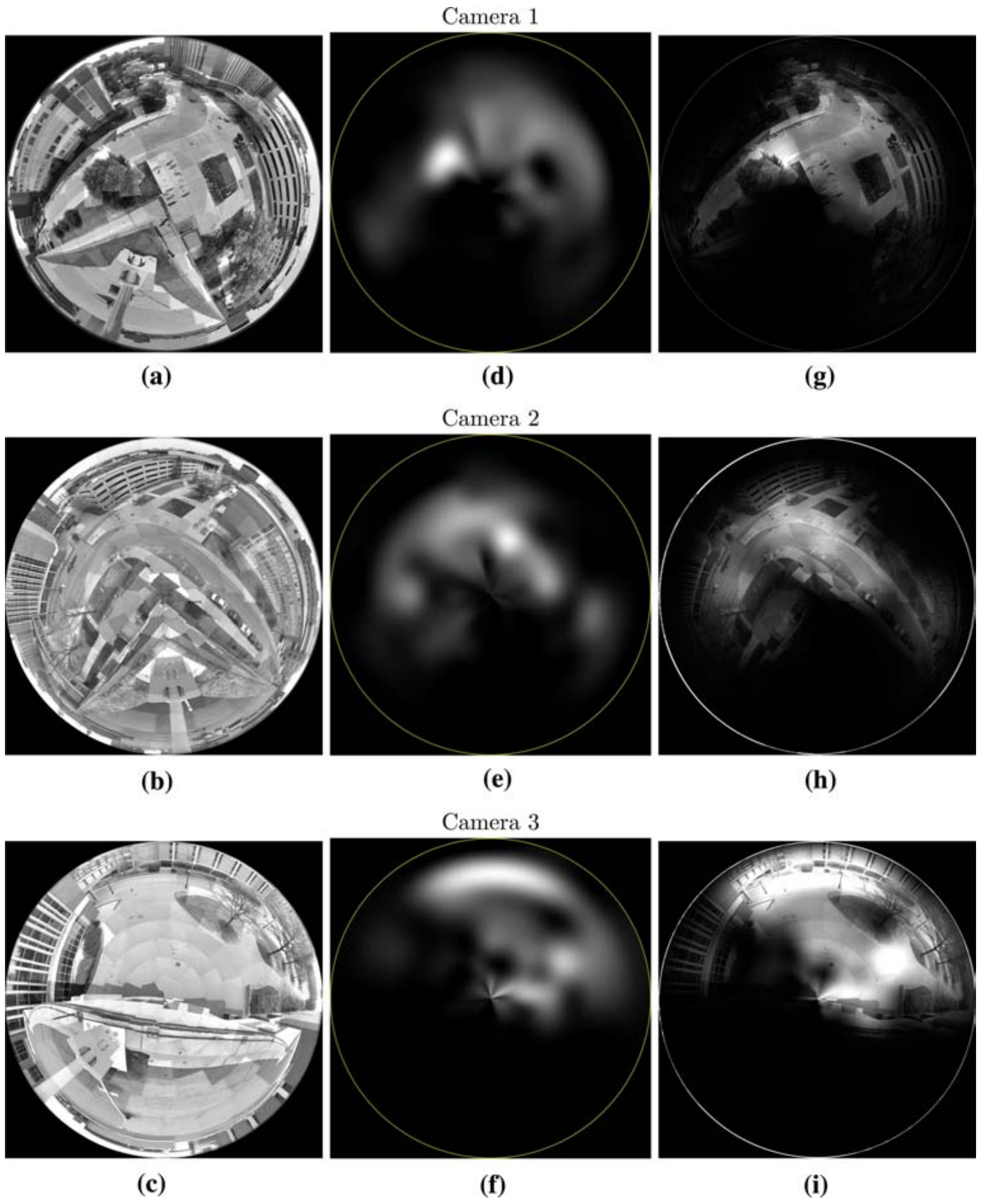

Fig. 11 Spherical representations of **a**–**c** camera views, **d**–**f** activity maps, and **g**–**i** overlays

As discussed, we evaluated the classification and reduction thresholds (T_{class}, T_{grow}) simultaneously over a wide range of values. The value for T_{class} was varied over a range suitable for each similarity measure (e.g., 0–1 for Bhattacharya Distance, 0–2 for Minkowski-form distance). The value for T_{grow} was varied over the range of 4 – 18% of the MHI blob size.

For evaluation, precision/recall data was collected for the segmentation results for each combination of T_{class} and T_{grow} compared to the manually segmented MHIs. The result of these experiments are summarized in Fig. 13 using a plot of the precision versus recall for each threshold combination. For all measures, except for match distance which significantly underperforms, the results are very similar. In order to accurately compare these results, we use the F-measure (F) [26] (or harmonic mean) of precision (P) and recall (R)

$$F = \frac{2PR}{P+R} \tag{10}$$

The F-measure provides a single number for comparison across all combinations of thresholds. The optimal thresholds for each similarity metric are provided in Table 3. An evaluation of this table concludes that Bhattacharya distance and Jeffrey divergence are essentially equivalent for our dataset (consistent with Fig. 13). We selected the Bhattacharya distance since the optimal T_{grow} value was larger, meaning that more pixels

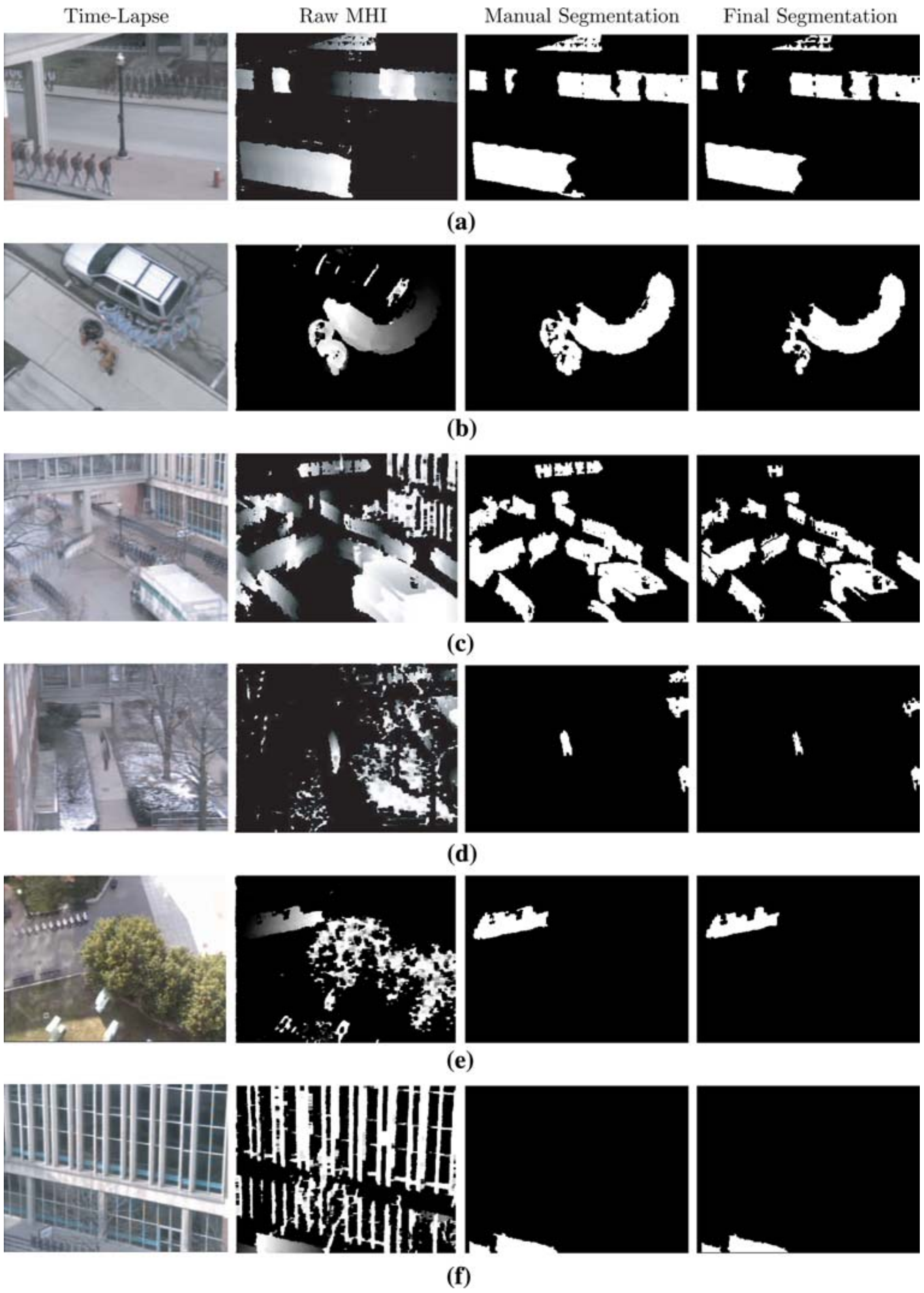

Fig. 12 Training data examples

are selected at each reduction step, possibly resulting in fewer iterations of the candidacy–classification–reduction stages and likely shorter execution time. The automatically segmented data (using the selected classification and growing thresholds) are shown in the rightmost column of Fig. 12. The results are very similar to the manually segmented MHIs.

To further evaluate our algorithm, we collected 10 additional passes of the full scene for each camera (10 × 119 MHIs) and computed the segmentations. In Figs. 14 and 15 we display a sample of successful and problematic candidacy-classification-reduction results.

In the successful segmentation images of Fig. 14, there is significant noise contribution in each of the cases, in particular the image in Fig. 14d is almost completely filled with noise from the surrounding building, tree, and ground cover. For all six image sequences, however, the final segmentation image includes none of these

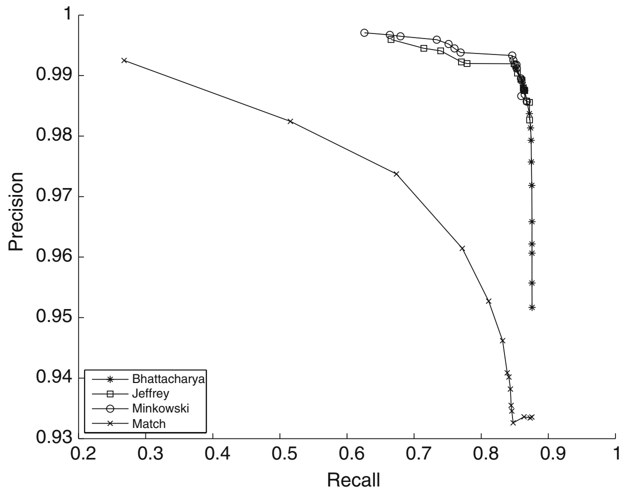

Fig. 13 Precision-recall curve for Bhattacharya distance, Jeffrey divergence, Minkowski-form distance, and Match distance

Table 3 The optimal results for each similarity measure with corresponding precision–recall range, *F*–measure, classification threshold, and growing threshold

Similarity measure	Precision-recall range	*F*-measure	T_{class}	T_{grow}
Bhattacharya distance	0.986–0.870	0.925	0.79	0.10
Jeffrey divergence	0.986–0.872	0.925	0.36	0.07
Minkowski-form distance	0.986–0.868	0.923	0.24	0.15
Match distance	0.934–0.876	0.904	15.0	0.05

regions. Furthermore, in all of these sequences, valid target objects exist within/between noise regions and are often significantly corrupted by the noise. Although, even with the large interaction between target objects and noise, the final segmentation images are an accurate reflection of the target objects in their respective scenes. The final segmentation, however, often includes a small number of non-target pixels due to the connection of valid translating objects with minor noise and the nature of our classification. For instance, in Fig. 14e the top middle blob in the final segmentation is human motion; however, the small lower portion is part human activity and part noise. Perhaps the most encouraging result is Fig. 14b, where a person is walking behind a tree in the bottom of the image and is nearly completely occluded for the entire length of the sequence. Adding even more difficulty is the resulting noise from the tree leaves due to shaking and edge aliasing. With all of these difficult contributing factors, our algorithm accurately catches two large regions where the person's movement is unobstructed, a promising result. A similar result is provided in Fig. 14f in the upper-left corner of the sequence, where a person is also walking behind a tree and our algorithm correctly includes the unobstructed translating object pixels. Also, notice the disparity of view angles, specifically, Fig. 14a and Fig. 14e, with a nearly top-down view, versus Fig. 14c and Fig. 14d with a more oblique view. Finally, these images show a large variation in depth of view from near field in Fig. 14a and Fig. 14e to objects in the far field in Fig. 14c and Fig. 14d. In these variations, the algorithm robustly segments the translating motion.

The previous results are extremely encouraging; however, our algorithm does have difficulty in some cases and we provide examples in Fig. 15. In Fig. 15a, the combination of spatial aliasing from brick work and concrete block with the camera shaking happens to produce a small region in the MHI with a timestamp distribution similar to a translating object. In Fig. 15b, a glass window provides a reflection of multiple people walking through the scene below and is, as expected, captured by our algorithm. The next image in Fig. 15c fails to capture a person walking directly towards the camera (located in the left-center of the scene) due to a small and broken MHI trail (eliminated by the algorithm based on size). Lastly, we present in Fig. 15d a detected region that is part of the roof on which the camera is mounted. This region is very close to the camera (3 ft.) and is highly textured, and produces a small region with evenly distributed timestamps (as in Fig. 15a).

The overall segmentation results of the testing images show the desired sensitivity to the differences between translating objects and noise in MHIs. From the highly varying data presented (view angle, scene complexity, occlusion, and camera noise), we believe our algorithm is a simple, yet robust, method for dynamically extracting translating motion from a scene.

6.2 activity map construction

For evaluation of the global activity map, we separate our analysis into three separate categories, *spatial* consistency for a single camera, *temporal* consistency for a single camera, and *spatial* consistency of *multiple* cameras with overlapping views. The spatial consistency of a single camera is evaluated qualitatively using a spherical panorama of the camera view with an activity map overlay. The temporal consistency is assessed using the mean and standard deviation activity map images for each camera, and also using the evolution of an activity map for a single camera. Finally, spatial consistency of two cameras viewing the same area is demonstrated.

The activity map experiments employ the same testing data used in the previous experiments (10×119 segmented MHIs for each camera). We evaluate the spatial consistency of our activity maps with a visual inspection of the corresponding high activity map areas with known

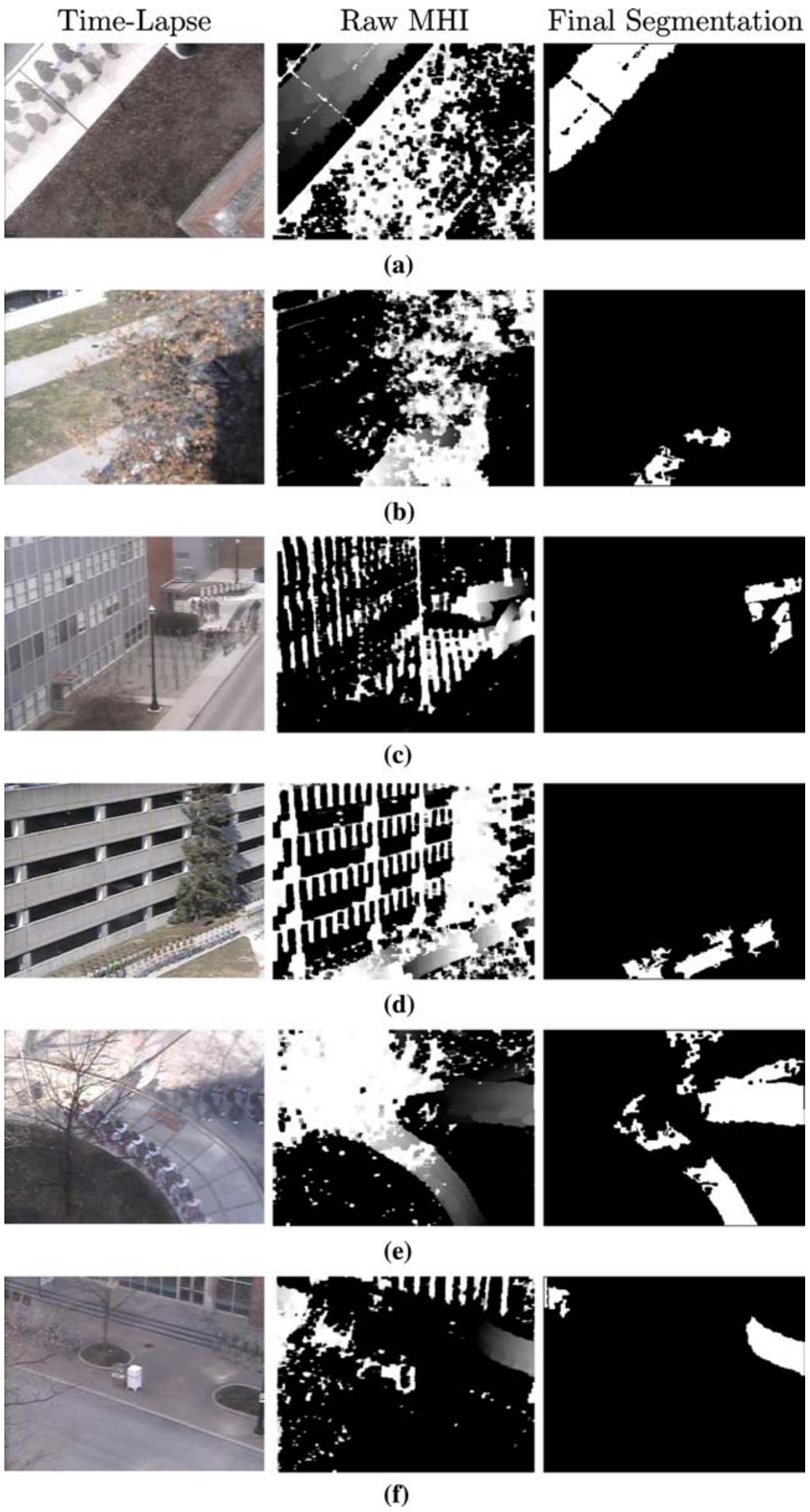

Fig. 14 Examples of successful MHI segmentations for testing data

high-traffic areas. For visualization purposes we provide spherical panoramas of the camera views and spherical panorama activity map overlays for each of the cameras in Fig. 11. The spherical panorama for each camera in Fig. 11a–c shows the full viewable area for the camera with walkways, roads, and buildings. The middle column (Fig. 11d–f) shows the warped (spherical) activity map for each camera where higher intensity corresponds to more activity. We then blended the spherical panoramas from Fig. 11a–c with the activity maps from Fig. 11d–f,

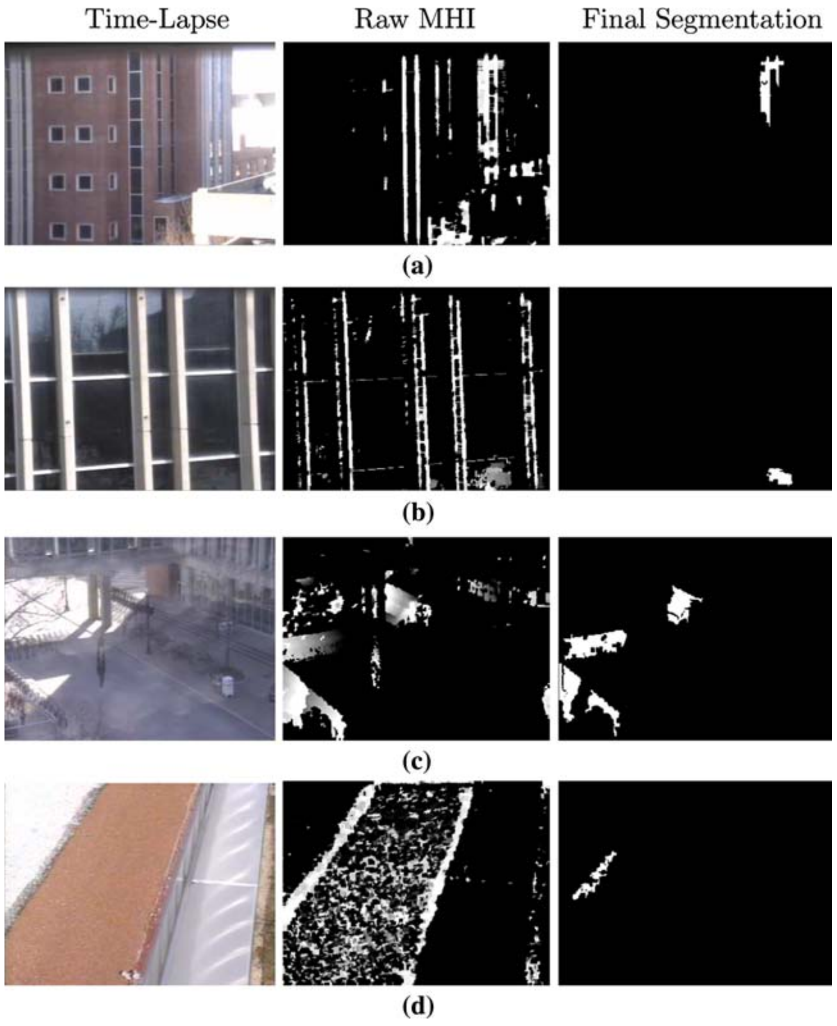

Fig. 15 Examples of problematic MHI segmentations for testing data

which results in the overlays displayed in the right-most column (Fig. 11g–i). Through inspection of the overlays, several areas are worth noting. First, the building areas are close to or completely black, meaning little to no activity occurs at these locations. Specifically, for each panorama the roof to which the camera is mounted is completely removed, which is desired. Also, the walkways and roadways in each scene are bright, indicating high activity areas. Several interesting features are noticeable from these maps. For example, in Fig. 11(g) the overlay shows the highest activity area which resides close to a tree; however, manual inspection of the segmentation results shows that this activity corresponded to pedestrian motion at this location (valid motion). Also for this overlay, although difficult to see, the large rectangular patch of grass at 70° from vertical (clockwise) has nearly a zero activity value (this low-activity region is easier to see from the corresponding spherical activity map in the middle column of the same figure). This result shows that our activity map is an accurate representation of the translating objects in the scene, as people typically walk around this area of grass. In addition, in the overlay of Fig. 11i, we see a noticeable intensity area on the windows of the building on the left. In this instance, the side of the building is composed of large glass panes which from the position and angle of the camera reflect the ground well. This is exactly the reason for this activity region, as reflections of walking pedestrians are captured by the camera and naturally pass as human activity. Also, to the right of this region we see a dark area on the walkway. Closer examination of the dataset found that little to no pedestrian activity occurred in this area when the video sequences were captured. The high semantic correlation between the overall scene structure and the activity maps demonstrates the robustness of the algorithm.

The temporal consistency of an activity map is evaluated by examining the mean and standard deviation of the activity map for each camera and the temporal evolution of an activity map for a single camera. Consistency in this instance does not refer to similar values at all instances in time at the same location, but instead,

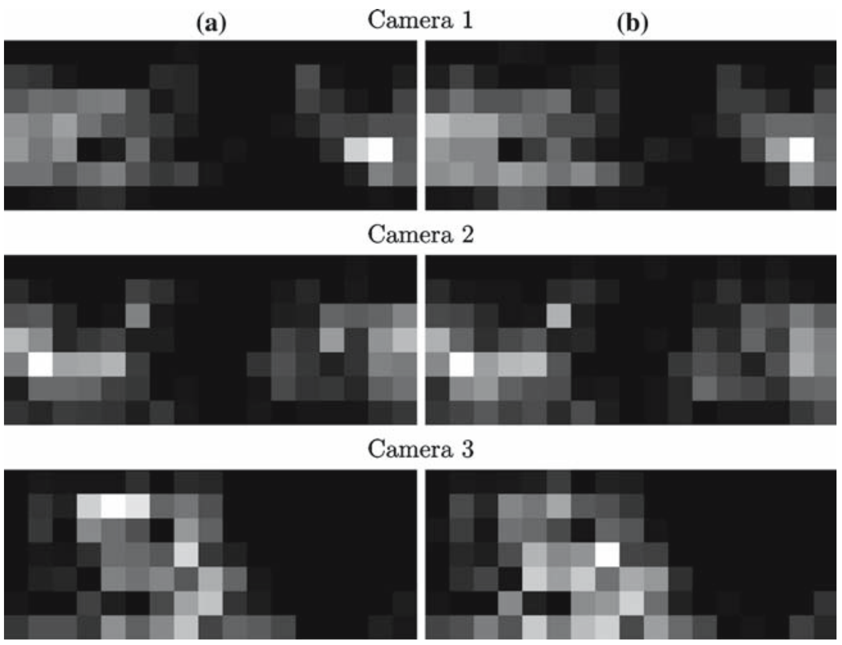

Fig. 16 For 10 total passes of three separate outdoor cameras we display an image of the **a** mean and the **b** standard deviation of the activity map value at each location

refers to the expected quantity of variation in activity at a location based upon the average activity at that location. In other words, high activity areas will see large variation because a person or vehicle may or may not be moving through the scene at a specific time. Similarly, in areas of low activity (i.e., sides of building and rooftops) we expect little to no variation since human activity rarely if ever occurs in these locations. As a result, the mean and standard deviation values for each location are expected to be directly proportional to the activity map values.

In Fig. 16, we display the mean and standard deviation values (scaled for display) for the activity maps in Fig. 11 (using all 10 passes). These maps demonstrate that high-activity areas have the expected high variability in activity level at any given moment. In addition, low-activity areas have consistently low activity the majority of the time. This could be used to provide a measure of anticipation for expected activity at a location. We also provide the evolution of an activity map over 10 passes for a single camera in Fig. 17. The evolution of the activity map at each time is the cumulative sum of the new activity map with all previous activity maps (maps are normalized for display only). Here we see that over time high-activity locations increase in intensity and low activity areas remain at consistently

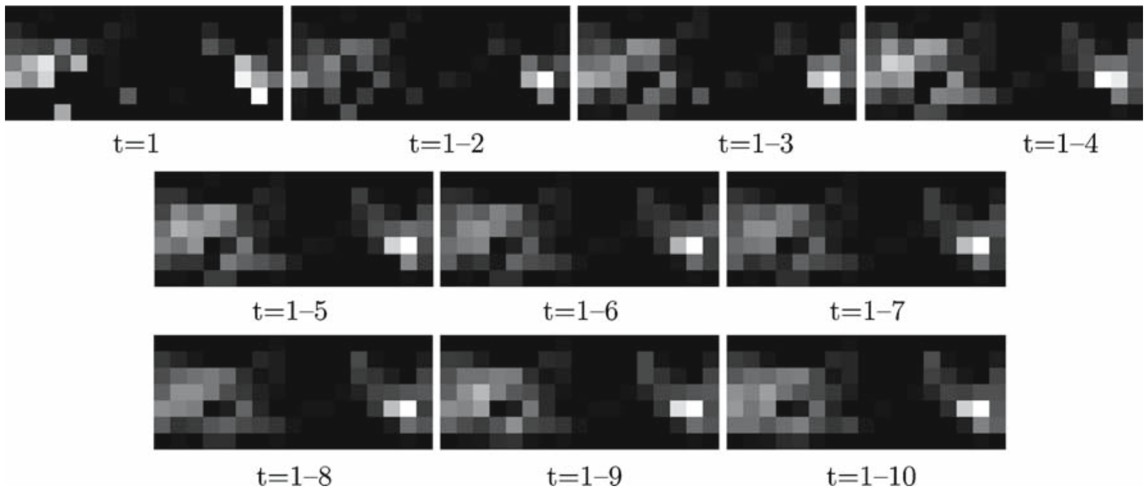

Fig. 17 Cumulative evolution at each timestep of the global activity map for a single camera

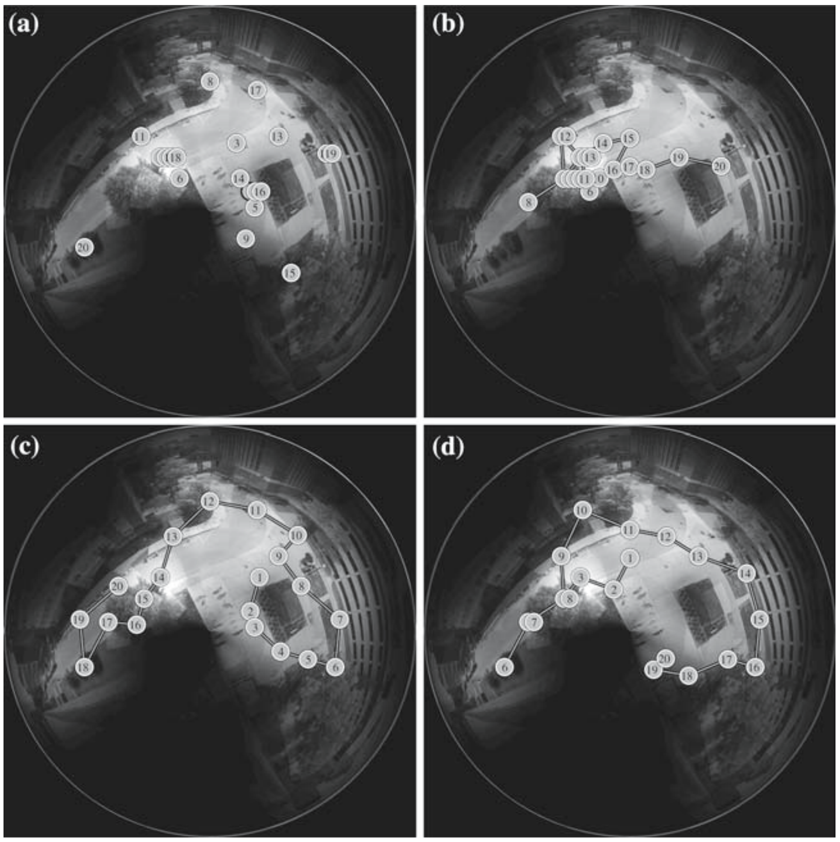

Fig. 18 Paths for 20 camera moves for four navigation techniques. **a** Probabilistic jump, **b** probabilistic walk, **c** inhibited probabilistic walk, and **d** reinforcement learning paths (goal locations are 1, 3, 7, 17, and 20)

low activity. The result appears to stabilize to a fairly consistent map which tends to vary slightly in the high intensity area but rarely changes in the low activity areas.

Finally we examined activity maps created using two separate cameras with significantly overlapping views. For the activity map overlays in Fig. 11g, h, the maximum activity for both cameras occurs on the roadway/walkway in-between the two cameras. This is significant because no two passes were collected simultaneously for these cameras. In other words, independent of the exact time-of-day and day-of-week, the activity map converged to similar activity emphasis across the same physical space but measured from a different position (and time). This supports the validity of our local activity measure, and the accuracy and precision of the resulting global activity map. Hence, the activity map results indicate that we do indeed capture an appropriate activity model for the full viewable field of a single PTZ camera and that the results are robust across both space and time.

6.3 Navigation

We next compared the four activity-based navigation techniques (probabilistic jump, probabilistic walk, inhibited probabilistic walk, and reinforcement learning paths) presented in Sect. 5. For inhibited probabilistic walk, we set the inhibition rate $\alpha = 0.1$, and for reinforcement learning paths the number of goal location states was set to 9, which were selected probabilistically from the activity map. We note here, that for all navigation techniques, zero-valued locations in the activity map are set to a value of 10% of the minimum non-zero activity map value (i.e., all locations must be reachable). We provide a comparison of the techniques as opposed to a strict evaluation because the optimal technique will depend on the application and context.

We provide a sample path of each navigation technique by overlaying the sample path on a spherical panorama activity map blend (see Fig. 18). For each algorithm we display 20 iterations (camera moves) about the scene, and show the sampled activity map locations using disks labeled in sequential order (to designate multiple samples on a location we shift the disks and labels). This provides a visual comparison of each navigation technique, as well as showing the underlying physical space (walkways, buildings). In addition, for the three “walking” techniques we also provide a line indicating the path between sequential locations.

The first navigation technique, probabilistic jump (Fig. 18a), illustrates how the highest probability locations are sampled multiple times whereas the lower

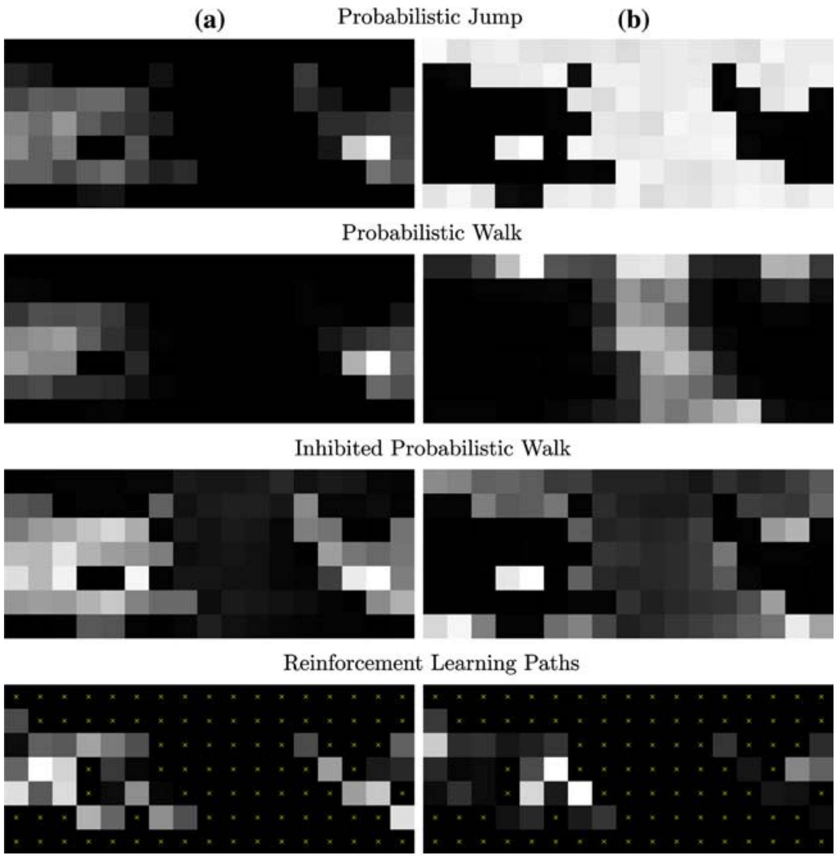

Fig. 19 Visualization of **a** sample rate (*brighter* corresponds to higher sampling) and **b** mean time between samples (*brighter* means a longer time between samplings) for each navigation method ('*x*' means the location was never sampled by the algorithm)

activity areas are sampled less. The "jumping" however, is usually not ideal for some human observers. The next navigation method, probabilistic walk (Fig. 18b), tends to stay in the local maxima around a single high activity area. However, the algorithm, as desired, moves in a manner more suitable for human observation and avoids low activity areas. The next navigation technique, inhibited probabilistic walk (Fig. 18c), is intended to eliminate the problem of becoming stuck in local maxima, as it tends to move away from recently sampled locations. Most notably, the patch of grass in this panorama overlay (discussed previously) is avoided. The algorithm walks around this low-activity area (locations 1–9) to remain in higher activity locations. Finally, the reinforcement learning paths (Fig. 18d) move between 5 of the 9 goal locations along paths of maximum reward. By design, these paths avoid low-activity areas and move along areas of high reward. Similar to the result for inhibited probabilistic walk, this method also avoids the low-activity grass patch.

First, we compared the experimental sample rate (i.e., the number of times a location is visited over a set number of camera moves, $\frac{\#\text{samples}}{\text{time}}$) of each navigation technique with the original input activity map. Optimally, the sample rate for a technique will match exactly the activity map values (i.e., higher sampling of higher activity locations). Next, we compared the mean time between samples (i.e., the average time or number of steps between two consecutive visits of a location) for each navigation method with the activity map value. The mean time between samples for all locations should vary inversely with the activity map values (i.e., higher activity locations should have shorter time between consecutive visits). Lastly, we provide a qualitative comparison of properties and limitations for all four navigation techniques.

To quantitatively compare the navigation techniques, we employed the final activity map shown in Fig. 17 (for the camera displayed in Fig. 11a). Each navigation method performed one million camera moves (to the next location) in the activity map, where at each iteration the time and location is recorded. The resulting data was analyzed to determine how well each navigation technique mimics the input activity map in terms of sample rate and mean time between samples.

In Fig. 19(a), we provide a visual comparison of the sample rates for a single activity map for each navigation technique. In order to compare the results, we computed the similarity between the sample rates (normalized) with the input activity map (normalized) using the Bhattacharya distance. The results are shown in Table 4. Immediately apparent, and expected, is the

Table 4 Similarity of sample rate versus activity map value using Bhattacharya distance

Navigation method	Similarity value
Probabilistic jump	1.00
Probabilistic walk	0.97
Inhibited random walk	0.95
Reinforcement learning paths	0.81

result for probabilistic jump which matches exactly with the original activity map. This is a direct consequence of the probabilistic sampling of the activity map with no spatial or temporal restriction on the next location selected (except for no temporally consecutive sampling of the same location). Also expected is the performance of the reinforcement learning paths navigation method which is least similar compared to the original activity map. The reason for this is the limited number of goals (9) and pre-defined paths from all locations to all goal locations, resulting in a large number of unsampled activity map locations (see Fig. 19a). This could be improved by increasing the number of goal states (but at large training cost) or by adding a periodic random jump. The two remaining navigation methods, probabilistic walk and inhibited probabilistic walk, fall in between these two extremes with probabilistic walk appearing more similar to the original activity map than inhibited probabilistic walk. This result is a consequence of the inhibition which essentially provides a history driving the algorithm away from recently visited locations. We also tested other inhibition rates ($\alpha = 0.05, 0.01$) and other inhibition functions (exponential and linear), but the results were similar. A useful observation which ties these two techniques together is that inhibited probabilistic walk is equivalent to probabilistic walk when the inhibition function is a dirac delta function, which fundamentally removes the inhibition.

The previous similarity metric on sample rate is useful but does not capture a very important and desirable feature of navigation which is the nature of paths formed over time. This is most apparent in the comparison between probabilistic walk and inhibited probabilistic walk. Utilizing the mean time between samples for each location, we gain a sense of the quantity of time that passes before a high/low activity location is revisited, or how long the path is before the location is revisited. Notice this does not provide an evaluation of the path itself, but provides an indication of the length of the path. We examine the relative mean time between samples using Fig. 19b and the absolute mean time between samples versus activity map value in Fig. 20. The relative mean time between samples in Fig. 19b compared to the activity map (Fig. 16) shows that for all navigation methods, except for reinforcement learning paths, the mean time between sample values is smaller at higher activity locations and larger at lower activity locations. The result for reinforcement learning paths is similar; however, due to the strict paths, many low-activity locations were not sampled resulting in zero values. Notice that the mean time between samples is, in general, inversely proportional to the sample rate, indicating high sampled locations have relatively short times between samples and vice versa.

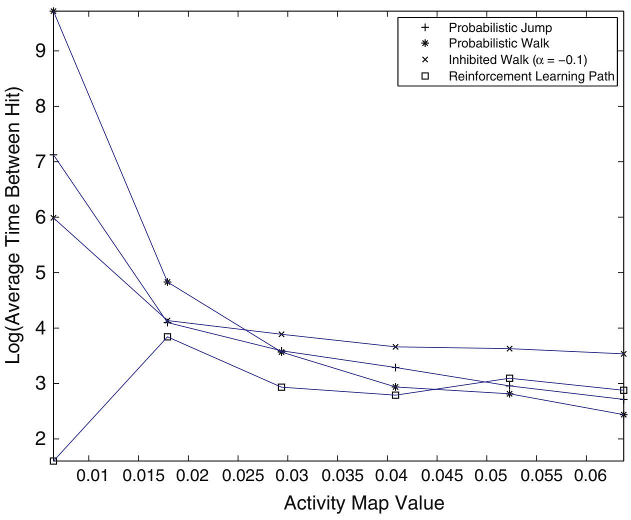

Fig. 20 activity map value versus Log (mean time between samples)

In Fig. 20, we show the absolute mean time between samples across different activity values for each navigation method. The variation of these navigation methods provides an understanding of how the algorithm will navigate a given activity map. In the case of probabilistic jump, walk, and inhibition, the path lengths for small activity values are large and path lengths for large activity values are small. Specifically, in the case of probabilistic walk, due to the restriction of moving to only neighboring locations the disparity between large and small activity values is most significant. Hence, in the short-term, probabilistic walk will tend to get stuck in local maxima of the activity map, rarely visiting lower-value activity map locations. Combating the inherent nature of probabilistic walk to find and favor local maxima, the inhibited probabilistic walk suppresses recently visited areas. As a result, there is less disparity between mean time between samples for large and small activity map values. Note, however, that the path lengths for higher activity map values are still shorter than for small activity map values (i.e., favoring higher activity areas), which affirms that the activity map information is being

Table 5 Properties and limitations of each navigation algorithm

Navigation	Properties	Limitations
Probabilistic jump	• Optimal according to previous observation • No training	• Discontinuous scanning
Probabilistic walk	• Spatio-temporal consistency • Optimal over very long temporal windows • No training	• Sampling distribution over short temporal window is suboptimal (i.e., short paths) • Easily caught in local maxima in activity map
Inhibited random walk	• Sample rate distribution over short temporal window (i.e., longer paths) • No training	• Inhibition function and rate selection (α)
Reinforcement path learning	• Optimal path between two locations	• Long training time • Each update requires retraining

used. Finally, in the case of reinforcement learning paths, the path length for small activity map values is zero as this algorithm never visits these locations. Also different is that the mean time between samples does not necessarily decrease with higher activity value. This is likely due to the selection of goal locations and the limited number of paths between goal locations.

Lastly, we provide a list of properties and limitations in Table 5 for each navigation technique. We reiterate here that this analysis provides neither a single optimal technique nor a ranking of navigation techniques. This is intentional due to the fact that the optimal navigation algorithm will vary depending on application and situational context.

7 Discussion

In this work, some limitations exist with the activity map presented in Sect. 4.6. The current system, as noted previously, generates a single activity map for a single PTZ camera, which fundamentally compresses time into a single representation. Consequently, the natural rhythms of daily life are lost. A natural extension of the activity map is to include a temporal dimension to capture more seasonal/periodic behavior patterns [8]. We plan to expand the activity map to incorporate time by constructing the activity map at specific times (e.g., one map each half hour), resulting in a single activity map at each time step. This addition will provide the ability to more accurately represent the activity of a given camera where levels of activity vary greatly across time (e.g., a parking lot, a university campus, etc.).

Discussions with surveillance personnel have also shown the necessity for consideration of camera navigation with multiple magnifications and resolutions. A simple method to expand the current activity map to include multiple magnifications and resolutions is to build an activity map at a set of magnifications (e.g., $1\times, 2\times, 4\times$, etc.) and a set of resolutions (e.g., $7 \times 17, 14 \times 34, 28 \times 68$, etc.). This will result in a pyramid of activity maps for both magnification and resolution, which can then be used to navigate the camera where location selection will occur across different levels of the magnification and resolution pyramids. The quantity of magnification and resolution for a location could also be adaptive to the quantity of activity detected. These extensions to the activity map will provide a more accurate and useful model of a PTZ camera's complete viewable field.

8 Summary and conclusion

We presented an adaptive, focus-of-attention, scene-specific model using standard PTZ camera hardware to address the lack of scene-specific information used in current automatic camera scanning algorithms. Our activity model is constructed by first detecting and quantifying local human activity (translating motion) at discrete pan/tilt locations over a camera's complete viewable field, and then using the activity measurements to create a global activity map. We also presented four automatic camera navigation techniques that employ the activity map (past observations) to continually select locations that maximize the opportunity of observing (future) human activity.

The activity detection method for each local view is based on an iterative candidacy–classification–reduction process using MHIs. The candidacy and classification stages are used to validate and then classify an MHI blob as either target translating motion or noise. If a blob is classified as noise, a reduction algorithm is applied to extract any possible sub-regions containing translating objects. A single activity measurement is then calculated based on the final binary segmentation image. Activity measurements at each discrete pan/tilt location are then used to construct the global activity map. This activity map is exploited by a camera navigation algorithm (e.g., probabilistic jump, probabilistic walk, inhibited probabilistic walk, or reinforcement learning Paths) to maximize the opportunity of observing human activity. The activity map is treated as a pseudo-probability distribution for probabilistic walk, probabilistic jump, and inhibited probabilistic walk, and is interpreted as a reward function for reinforcement learning paths.

Experiments examined the performance of our algorithms related to measuring local activity, generating global activity maps, and applying our navigation techniques. After selecting optimal detection thresholds determined from precision–recall results with manually segmented data, sample results on test data showed the desired sensitivity to differences between translating objects and noise. Next, we presented a consistency analysis of our activity map that demonstrated robustness across both space and time. The results showed that the high/low activity areas had high/low variability in activity level, that low activity areas were generally quiescent over time, and that two overlapping camera views yielded similar high activity areas. Lastly, we compared the four navigation methods using sample rate and mean time between samples (path length). The similarity (using Bhattacharya distance) between the sample rates and the activity map showed that Probabilistic Jump matched exactly, whereas reinforcement learning paths was most dissimilar due to the limited number of goal states and several unsampled locations. For the mean time between samples comparison, probabilistic jump, walk, and inhibition showed path lengths that varied inversely with activity map value, while with reinforcement learning paths, the path length was fairly constant across activity level.

Overall, our current approach shows very promising results. In future work, we plan to examine additional features (e.g., gradient direction, color, texture) for the detection component, which may result in more robust segmentation but at an increased computational cost. We will also investigate methods for temporal and spatial updating of the activity map, which is necessary for long-term operation of the system. We additionally plan to incorporate multiple zoom factors to construct a multi-scale activity map for navigation. Finally, we plan to include anomaly detection and alerting mechanisms based on past global observations (activity map) and local current activity.

Acknowledgements This research was supported in part by the National Science Foundation under grant no. IIS-0428249, and through collaborative participation in the Advanced Decision Architectures Consortium sponsored by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0009. We additionally thank Christopher M. Gaal at The MathWorks for his software support.

References

1. Aggarwal, J., Cai, Q.: Human motion analysis: a review. In: Nonrigid and Articulated Motion Workshop, pp. 90–102 (1997)
2. Bobick, A., Davis, J.: The recognition of human movement using temporal templates. IEEE Trans. Patt. Anal. Mach. Intell. **23**(3), 257–267 (2001)
3. Bradski, G., Davis, J.: Motion segmentation and pose recognition with motion history gradients. In: Proceedings of Workshop on Applications of Computer Vis. (2000)
4. Cedras, C., Shah, M.: Motion-based recognition: a survey. Image Vis. Comp. **13**(2), 129–155 (1995)
5. Chang, I., Huang, C.: The model-based human body motion analysis system. Image Vis. Comp. **18**(14), 1067–1083 (2000)
6. Dalal, N., Triggs, B.: An effective pedestrian detector based on evaluating histograms of oriented image gradients in a grid. In: Proceedings of Computer Vision and Pattern Recognition (2005)
7. Davis, J.: Visual categorization of children and adult walking styles. In: Proceedings of international conference on Audio- and Video-based Biometric Person Authentication, pp. 295–300 (2001)
8. Davis, J., Keck, M.: Modeling behavior trends and detecting event anomalies using seasonal Kalman filters. In: Proceedings of Workshop on Human Activity Recognition and Modeling (2005)
9. Davis, J., Keck, M.: A two-stage template approach to person detection in thermal imagery. In: Proceedings of Workshop on Applications of Computer Vis. (2005)
10. Davis, J., Sharma, V.: Background-subtraction in thermal imagery using contour saliency. Int. J. Comp. Vis. (to appear)
11. Gavrila, D.: The visual analysis of human movement: a survey. Comput. Vis. Image Understand. **73**(1), 82–98 (1999)
12. Gavrila, D.: Pedestrian detection from a moving vehicle. In: Proceedings of European Conference Computer Vision, pp. 37–49 (2000)
13. Itti, L., Baldi, P.: A principled approach to detecting surprising events in video. In: Proceedings of Computer Vision and Pattern Recognition, pp. 631–637 (2005)
14. Javed, O., Shafique, K., Shah, M.: A hierarchical approach to robust background subtraction using color and gradient information. In: Workshop on Motion and Video Computing, pp. 22–27, (2002)
15. Johnson, N., Hogg, D.: Learning the distribution of object trajectories for event recognition. In: British Machine Vision Conference, pp. 583–592 (1995)
16. Lipton, A., Fujiyoshi, H., Patil, R.: Moving target classification and tracking from real-time video. In: Proceedings of Workshop on Applications of Computer Vision, pp. 8–14 (1998)
17. Little, J., Boyd, J.: Recognizing people by their gait: the shape of motion. Videre **1**(2), 2–32 (1998)
18. Liu, Y., Collins, R., Tsin, Y.: Gait sequence analysis using frieze patterns. In: Proceedings of European Conference Computer Vision, pp. 657–671 (2002)
19. Makris, D., Ellis, T.: Automatic learning of an activity-based semantic scene model. In: Advanced Video and Signal Based Surveillance, pp. 183–188 (2003)
20. Minka, T., Picard, R.: Interactive learning using a "society of models". In: Proceedings of Computer Vision Pattern Recognition, pp. 447–452 (1996)
21. Mitchell, T.: Machine Learning. McGraw-Hill, New York (1997)
22. Niyogi, S., Adelson, E.: Analyzing and recognizing walking figures in XYT. In: Proceedings of Computer Vision and Pattern Recognition, pp. 469–474 (1994)
23. Remote Reality, http://www.remotereality.com/vtprod/index.html
24. Oren, M., Papageorgiou, C., Sinha, P., Osuma, E., Poggio, T.: Pedestrian detection using wavelet templates. In: Proceedings of Computer Vision and Pattern Recognition, pp. 193–199 (1997)

25. Pelco Spectra III SE. http://www.pelco.com/products/default.aspx? id=337
26. Van Rijsbergen C.: Information Retrieval. Department of Computer Science, 2nd edn. University of Glasgow (1979)
27. Rubner, Y., Guibas, L.: Tomassi, C.: The earth mover's distance as a metric for image retrieval. Int. J. Comp. Vis. **40**(2), 99–121 (2000)
28. Stauffer, C., Grimson, W.E.L.: Adaptive background mixture models for real-time tracking. In: Proceedings of Computer Vision and Pattern Recognition, pp. 246–252 (1999)
29. Strat, T., Fischler, M.: Context-based vision: recognizing objects using information from both 2-d and 3-d imagery. In: IEEE Transaction Pattern Analysis and Machine Intelligence, pp. 1050–1065 (1991)
30. Viola, P., Jones, M., Snow, D.: Detecting pedestrians using patterns of motion and appearance. In: Proceedings of International Conference Computer Vision, pp. 734–741 (2003)
31. Wang, L., Hu, W., Tan, T.: Recent developments in human motion analysis. Pattern Recogn. **36**, 585–601 (2003)
32. Wren, C., Azarbayejani, A., Darrell, T., Pentland, A.: Pfinder: real-time tracking of the human body. IEEE Transaction on Pattern Analysis and Machine Intellignece **19**(7), 780–785 (1997)
33. Zhaozheng, Y., Robert, C.: Moving object localization in thermal imagery by forward-backward MHI. In: IEEE International Workshop on Object Tracking and Classification. Beyond the Vision Spectrum (2006)
34. Zhao, T., Nevatia, R.: Stochastic human segmentation from a static camera. In: Workshop on Motion and Video Computing, pp. 9–14 (2002)
35. Zhou, X., Collins, R., Kanade, T., Metes, P.: A master–slave system to acquire biometric imagery of humans at distance. International Workshop on Video Surveillance, pp. 113–120 (2003)