Duration Dependent Codebooks for Change Detection

Brandon A. Mayer
Brandon_Mayer@brown.edu
Joseph L. Mundy
mundy@lems.brown.edu

Brown University
School of Engineering
Rhode Island, USA

Abstract

This paper describes a supervised system for pixel-level change detection for fixed, monocular surveillance cameras. Per-pixel intensity sequences are modeled by a class of Hidden Semi-Markov Models, Duration Dependent Hidden Markov Models (DDHMMs), to accurately account for stochastically periodic phenomena prevalent in real-world video. The per-pixel DDHMMs are used to assign discrete state labels to pixel intensity sequences which summarize the appearance and temporal statistics of the observations. State assignments are then used as a features for constructing per-pixel code books during a training phase to identify changes of interest in new video.

The per-pixel intensity model is validated by showing superior predictive performance to pixel representations commonly used in change detection applications. A new data set is presented which contain dynamic, periodic backgrounds with larger time scale variability than previous data sets and the proposed method is compared to state-of-the-art change detection methods using the new videos.

1 Introduction

Natural scenes are composed of complex, dynamic events that make it difficult for a change detection system to distinguish between changes of interest and background. To further compound the problem, it is impossible to define what a system should consider as a relevant change without considering the context of the application. For example, are cars moving along a highway foreground or background? If the goal of the applicaiton is to count the number of cars entering and exiting a restricted area, it is necessary for the system to account for every car in the scene. However, if the system is to monitor a busy highway for irregular traffic activity such as a collision, then the system will need to consider common traffic patterns as normal and not declare routine traffic activity as significant change.

These difficulties are pervasive, but so far not extensively addressed by the change detection community. Most state-of-the-art change detection systems are designed to be unsupervised (no interaction with an operator) and utilize blind, on-line learning to update a background model. This paradigm misplaces the responsibility of defining what is or isn't a meaningful change on the system's designer, not the end user. While it may be argued that unsupervised change detection systems provide a low-level foreground/background segmentation to be used as input for classification systems operating at higher levels of the semantic

hierarchy, the underlying change detection system would still need to be customized to ensure the proper primitive information is passed to the next module.

On the surface, unsupervised change detection seems like the more desirable, challenging task. However, unsupervised systems by necessity, implicitly encode the designer's definition of a meaningful change that is constrained by a target application or data set. The goal of the research reported here is to design a change detection system that can be easily adapted to different definitions of change for scenes of varying complexity.

2 Related Work

Gaussian Mixture Models (GMMs) are closely related to DDHMMs and are a standard per-pixel model used in change detection [8, 10]. However, a GMM can only model typical pixel intensities, not intensity sequences; the order in which intensities are observed at a given pixel has no effect on the probability of the observation sequence.

Heras et. al [5] build a coarse temporal background model using two GMMs, one with a slow and another with a fast learning rate, to determine if a pixel intensity belongs to a static or dynamic background. Stable edges in the scene are also tracked to account for objects that have entered or been removed from the frame. However, the short and long term learning rates must be specified apriori and such a coarse quantization of time limits the systems ability to disambiguate events in the same scene that reoccur with different rates.

An alternative approach taken by the non-parametric algorithms ViBe [2] and PBAS [2] is to explicitly store a history of pixel intensities at every location. Both algorithms randomly add and discard observations from the history and define unique heuristics for thresholding the difference between the intensities stored in memory and novel observations to segment foreground and background. While each algorithm provides a global parameter governing the number of observations to store at each pixel, it must be set prior to deployment and implicitly sets an upper bound on the periodic dynamics the algorithm can associate with the background. Moreover, a background model represented by a pool of intensities cannot be queried by higher-level algorithms to provide information about the temporal semantics of the background.

Similar to the proposed approach, the CBBGS algorithm [9] creates codebooks for each pixel location in a video sequence. Code words are feature vectors whose elements are functions of the target pixel's color and intensity in frames of video during a training period. CBBGS initially stores all code words created during training, but in a second pass of the training data, a value called the Maximum Negative Run-Length (MNRL) is computed which is defined as the maximum number of frames a code word is absent prior to reoccurring. Any code word with a MNRL less than a pre-defined threshold (set in their experiments as half the length of the training sequence) is considered foreground and discarded from the codebook. The MNRL threshold causes many of the same problems associated with ViBe and PBAS. While the code words provide a sparse representation of low level appearance compared to explicitly storing pixel intensities, the global MNRL threshold similarly places an upper bound on the class of periodic phenomena which can be modeled by the CBBGS per-pixel code books.

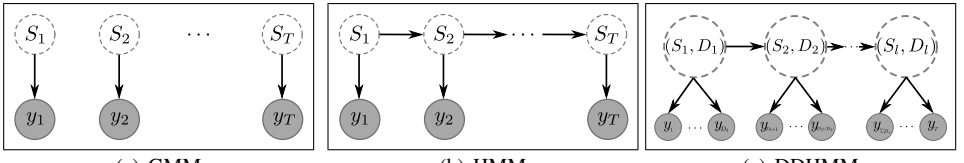
Local Binary Patterns (LBP) are a per-pixel texture descriptor represented as a histograms of binary values computed by thresholding the intensity of neighboring pixels by the intensity of the target pixel. [13] extends the LBP descriptor to the Spatial-Temporal domain to create the per-pixel STLBP histogram: a weighted sum of LBP histograms computed

in the current and previous frame of a video sequence. While the algorithm avoids false positive detections from dynamic content mistakenly made by the STLBPs static counterpart, e.g. such as waving trees and rippling water, the local nature of the temporal neighborhood cannot model intensity observations that reoccur with a long period.

While DDHMMs have been used in computer vision applications [4, 11, 12] they are typically used for high level, activity recognition or event detection to assign a semantic label to a frame of video. The proposed algorithm uses DDHMMs as a bottom-up, pixel-level model of intensity sequences.

3 Model

3.1 Duration Dependent Hidden Markov Models



(a) GMM (b) HMM (c) DDHMM

Figure 1: Visualization of temporal independence assumptions made by GMMs, HMMs, and DDHMMs.

A Duration Dependent Hidden Markov Model (DDHMM) models a sequence of observations, $Y = (y_1, y_2, \ldots, y_T)$, using a sequence of latent state pairs: $((S_1,D_1), (S_2,D_2), \ldots, (S_l,D_l))$ where S_i is a state label and D_i is a random variable that represents the time spent in state S_i. Note that capital letters denote random variables and lower case letters represent specific variable assignments. The probabilistic graphical model for the DDHMM is shown in Figure 1(c) where dotted circles represent random variables and the shaded nodes represent observed quantities. The topology of the graphical model is variable as the number of state-duration tuples will change depending on the particular configuration of the duration random variables.

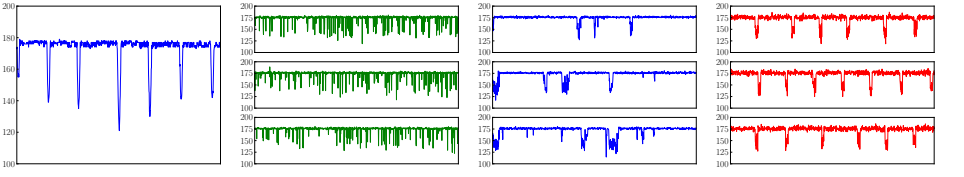
The observation and state sequences are related through three fundamental distributions: the duration $p(D_i = d_i \mid S_i = s_i)$, state transition $p(S_i = s_i \mid S_{i-1} = s_{i-1})$ and emission $p(y_t \mid S_i = s_i)$ distributions. The likelihood of an observation sequence given a particular latent state sequence is

$$p(y_1,\ldots,y_T|(s_1,d_1),\ldots,(s_l,d_l)) = p(s_1)p(d_1 \mid s_1)\prod_{m=1}^{d_1} p(y_m \mid s_1)\cdots$$
$$\prod_{i=2}^{l-1} p(d_i \mid s_i)p(s_i \mid s_{i-1})\prod_{j=1}^{d_i} p(y_{r_i+j} \mid s_i)p(D_l \geq d_l)p(s_l \mid s_{(l-1)})\prod_{k=1}^{d_l} p(y_{r_{(l-1)}+k} \mid s_l) \quad (1)$$

Where $r_i = \sum_{m=1}^{i} d_m$ and $p(s_1)$ is an initial distribution of state labels. The observation sequence is assumed to be left-censored, i.e., the last tuple (s_l,d_l) is distributed according to the state survival distribution $p(D_l \geq d_l \mid s_l)$, to mitigate the effect of the length of the observation sequence on the probability of a particular state sequence [6].

In all experiments the duration and transition distributions are multinomial with zero self-transition probability. The emission distributions are univariate Gaussian with mean and standard deviations $\{\mu_{s_i}, \sigma_{s_i}\}$.

3.2 Model Validation



(a) Training Sequence (b) Samples from GMM (c) Samples from HMM (d) Samples from DDHMM

Figure 2: Sample time series drawn after training GMM, HMM and DDHMM models using an intensity sequence obtained from a single pixel location in the Swing sequence.

The power of the proposed DDHMM model is illustrated in Figure 2. Figure 2(a) is a time series of intensities extracted from the Swing sequence at the pixel marked in red in Figure 5. GMMs and HMMs with three states each are learned using standard maximum likelihood. Three intensity sequence samples were drawn from the GMM and HMM and were visualized in Figures 2(b) and 2(c) respectively. Figure 2(d) shows three sampled waveforms after learning the parameters of the DDHMM model using the algorithm outlined in Section 4. The GMM does not model temporal behavior and therefore cannot reflect the periodic motion of the swinging child. While the HMM offers a first order temporal approximation, the model is incapable of capturing the dynamics of the original intensity sequence. The sampled sequences from the DDHMM however closely mimic the original sequence and mirror the longer term ground - short term child appearance oscillation.

4 Learning Per-Pixel DDHMM Parameters

A simple single-pass, greedy algorithm is used for learning the parameters and complexity of the per-pixel DDHMMs. The algorithm defines three additional parameters: $\sigma_{init}, \sigma_{min}, \gamma$: the initial and minimum standard deviations for a state label, and a multinomial smoothing parameter (to avoid zero probabilities). In all experiments $\sigma_{init} = 15.0$, $\sigma_{min} = 3.0$, $\gamma = 1.0$. Given the first intensity observation y_1, a state is created with parameters $\{\mu_1 = y_1, \sigma_1 = \sigma_{init}\}$, the current state label s_1 is recorded, and a duration counter is initialized to $d = 1$. For all subsequent observations, the learning algorithm chooses one of three options: To extend the current temporal segment by associating a new observation with the current state label, to initialize a new temporal segment by associating the new observation with a previously observed state label, or to initialize a new temporal segment by creating a new state and adding it to the model in order to account for a previously unobserved appearance or temporal dynamic.

The best local state label assignment is made by choosing a state label from the set

existing states according to $s_t^e = \arg\max_s \{p(s_t \mid y_t, s_{t-1}, d)\}$ with

$$p(s_t \mid y_t, s_{t-1}, d) = \begin{cases} p(y_t \mid s_t)p(s_t \mid s_{t-1})p(D = d \mid s_t) & \text{if, } s_t \neq s_{t-1} \\ p(y_t \mid s_t)p(D \geq d \mid s_{t-1}) & \text{else.} \end{cases} \quad (2)$$

In Eq. 2, $p(y_t \mid s_t)p(s_t \mid s_{t-1})p(D = d \mid s_t)$ is the probability of transitioning to a new state and $p(y_t \mid s_t)p(D \geq d \mid s_{t-1})$ is the probability of staying in the same state given the previous state s_{t-1} and new observation y_t.

These choices are compared with the benefit of adding a new state, s_t^n, with an emission distribution build around the observed intensity. s_t^n is hypothesized with parameters $\{\mu = y_t, \sigma = \sigma_{init}\}$ and the probability of transitioning to this new state is computed as:

$$p(s_t^n \mid s_{t-1}) = \frac{\gamma}{K + 1 + \sum_{\forall s' \neq s_{t-1}} \#(s_{t-1} \rightarrow s')} \quad (3)$$

Where $\sum_{\forall s' \neq s_{t-1}} \#(s_{t-1} \rightarrow s')$ is the number of previously observed non-self transitions from s_{t-1} to any other state. This is equivalent to having extended the multinomial transition distribution to have included a s_t^n with γ prior counts.

The algorithm then chooses between two DDHMMs, the DDHMM with the existing set of states, and a DDHMM which is a copy of the current model but extended to include the hypothesized state s_t^n. To encourage a parsimonious allocation of resources and avoid overfitting, the cost of making the decision to select the DDHMM with the extra state is regularized to favor simpler models, i.e., DDHMMs with fewer states.

The *AIC* score [1] is a common objective function used for such model section and is defined as $AIC = 2\kappa - 2ln(Y, \theta)$ where κ is a measure of model complexity and $ln(Y, \theta)$ is the maximal log-likelihood of observations with respect to model parameters θ. Using a duration histogram with maximal length D_{max}, the multinomial transition and Gaussian emission distributions, κ could be computed as $K^2 - K + KD_{max} + 2K$ where K is the number of states of the model. However, the transition histograms are dynamically allocated, extended as needed while the model is updated, and are typically very sparse. Therefore, κ is approximated as $\kappa = K^2 + K$ and the *AIC* score of the current DDHMM, AIC^c, and the new candidate model (the current model extended by adding s_t^n), AIC^n, as:

$$\begin{aligned} AIC_t^c &= 2(K^2 + K)\ln(p(s_t^e \mid y_t, s_{t-1}, d)) \\ AIC_t^n &= 2((K+1)^2 + K + 1)\ln(p(y_t \mid s_t^n)p(s_t^n \mid s_{t-1})p(D = d \mid s_{t-1}) \end{aligned} \quad (4)$$

If $AIC_t^n < AIC_t^c$, s_t^n is added to the set of existing states and s_t^n becomes the current state. Regardless if the current or extended model is selected, the multinomial distributions are updated by adding a single count to the correct duration-transition bin and if $s_t \neq s_{t-1}$, d is reset to one, otherwise it is incremented. The parameters of the emission distribution for state s_t are updated via standard sequential maximum likelihood using σ_{min} as a minimum standard deviation for each appearence distribution [3].

An unoptimized multithreaded C++ implementation, running on a 3.46 GHz Intel i7 processor, achieves real-time performance. Specifically, continuously updating a DDHMM at each pixel for a video sequence containing seventeen hundred frames with resolution 240×320 pixels takes an average of 31 milliseconds per frame.

5 DDHMM Code Book Classifier

While the probability of an intensity sequence at any pixel could be computed using the DDHMM framework, the probability decays exponentially with the length of the intensity sequence. Thus, some form of length normalization procedure is needed to define a consistent threshold, and it is unclear what principle can be used to define such a normalization under all situations. Therefore, a code book based approach is taken to compare atomic elements of intensity sequences obtained from the implicit temporal segmentation provided by the DDHMM state assignments described in section 4.

The DDHMM learning algorithm described in Section 4 is run continuously during the training and testing phases to associate every pixel intensity observation with a DDHMM state label. The resulting per-pixel state assignments are used to compose code words of the form (s_p, d, s_n); the previous state label, the duration of s_p and the next state, respectively. The user provides the system with video footage containing normal scene dynamics and the code words produced at each pixel are recorded in a codebook (one for every pixel). Any code word observed during testing that does not exist in the code book is considered a deviation from normal scene dynamics and is flagged as a change of interest. In all experiments, the duration d was further quantized into the ranges $([1,6), [6,10), [10,25), [25,\infty))$ to further compress the per-pixel codebooks and a 7×7 median filter was applied to each frame of the change detection output as a post-processing step.

5.1 Change Localization by State Merging and Splicing

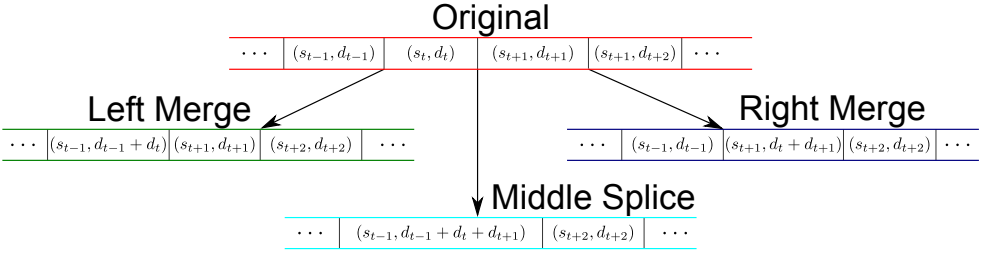


Figure 3: *Visualizing Merge and Splice Hypotheses*: The localization module does not modify the underlying state assignment but merely suppresses false positive artifacts arising from changes in the scene disrupting the duration of otherwise normal states and helps localize the change point in time.

In the proposed model, normal background sequences can exhibit long DDHMM state duration times: (s_t, d_t), which may persist for hundreds of frames. If a change occurs in the middle of such normal state durations then the entire period is considered to be change. This error occurs because no normal background states possesses the appropriate intensity distributions with shorter durations on each side of the change time interval. These false positive errors can be eliminated by "splicing", where the change interval is replaced by the normal background state, with the duration that spans the full interval. Thus, the intensity observations outside the change interval are considered normal even given the splitting of the normal state duration by changes. Figure 3 illustrates the splicing procedure, where there are three cases to consider: left merge - the change interval starts at the same time as the

long duration normal state; right merge - the change interval ends at the same time as the normal state duration; middle-splice - the change is fully interior to the long duration state time interval.

6 Evaluation

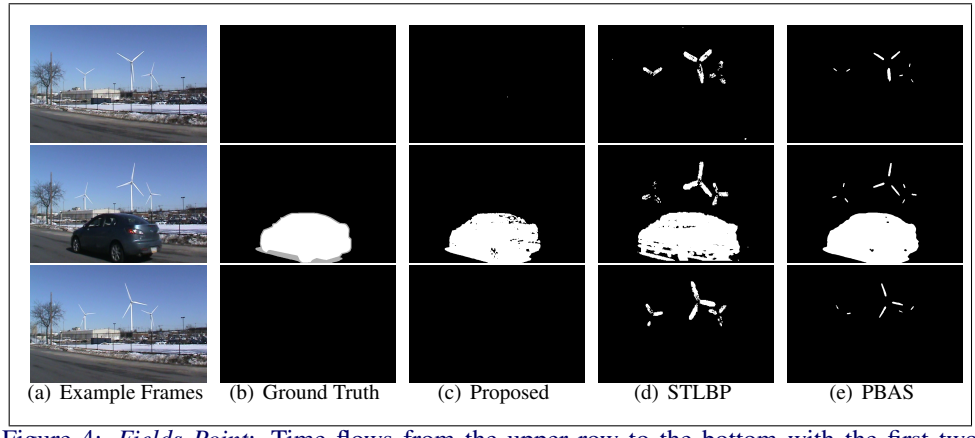


Figure 4: *Fields Point*: Time flows from the upper row to the bottom with the first two columns showing example frames of the video sequence and their corresponding ground truth labels in the same row. The third through fifth columns show the change detection results of the proposed, STLBP, and PBAS algorithms for the example frame in the corresponding row. The proposed method is the only algorithm able to learn that the spinning blades are a normal part of the scene and can still detect the previously unobserved cars.

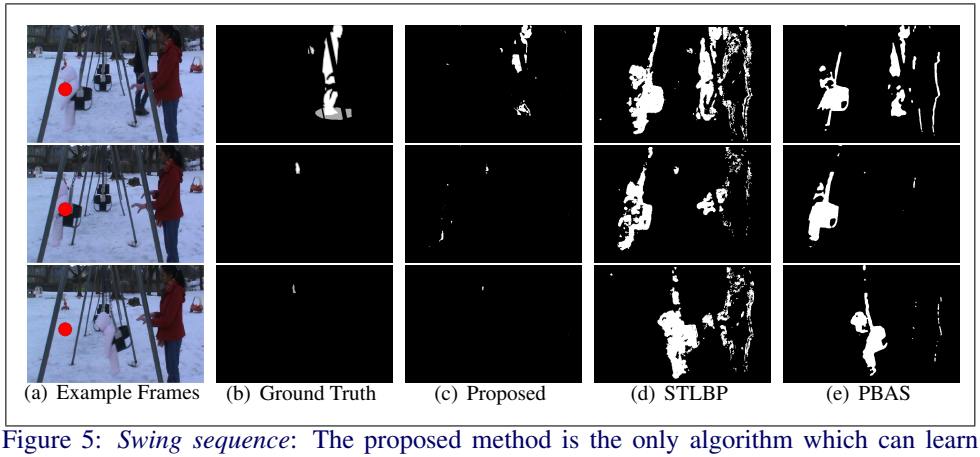


Figure 5: *Swing sequence*: The proposed method is the only algorithm which can learn the swinging child is a normal part of the scene but still detect the previously unobserved pedestrian.

Figures 4 and 5 show example frames from the Fields Point Turbine and Swing video sequences respectively. The images in the second columns visualize hand labeled ground

truth corresponding to the example video frame in the same row. A white pixel represents a true positive, black true negative, and gray is unknown due to shadow or ambiguous object boundaries. In both figures the last three columns show the detections made by the proposed method, STLBP, and PBAS, in that order. Again, a white pixel means the algorithm labeled the observation as a change of interest and a black pixel is the contrary. The parameters for the PBAS algorithm were set to the values reported in [7].

The Fields Point Turbine scene in Figure 4 monitors the entrance to a facility with wind turbines spinning in the background. Cars passing on the road are considered a change of interest whereas the spinning blades are a normal dynamic feature in the scene. The proposed method is able to model the spinning turbine, avoiding false positives resulting from normal blade movement which the other state-of-the-art algorithms mistake as meaningful change.

The Swing video sequence shown in Figure 5 shows a mother pushing her daughter on a swing set and eventually, a previously unobserved pedestrian enters and exits the scene. This seemingly innocuous footage contains interesting periodic phenomena that modern change detection algorithms cannot model. The mother's motions are repetitive as she pushes the child with a periodic rhythm. The mother and daughter on the swing set are considered normal, they are using the swing set for the entirety of the video sequence, and the pedestrian is a change of interest.

The change detection results for the proposed method show that the system is capable of learning the normal intensity sequences associated with the swinging child and mother by correctly labeling their pixel locations in the scene as normal. The system also correctly labels the majority of the pedestrian as change. The other state-of-the-art methods have no mechanism to associate observations caused by periodic motion with normal scene activity. To note one limitation of the proposed system, there are some false negatives on the pedestrian's legs. The pedestrian's pants are a similar color to the black harness of the swing that moves with comparable velocity, resulting in coincident state sojourn times. These mistakes are to be expected using only per-pixel intensity observations.

It may be argued that the parameters of the competing algorithms could be adjusted to better suit this scene. For example, it is possible to tune PBAS to maintain more background samples, allowing it to keep track of reoccurring pixel intensities that occur over a larger time scale. However, this global parameter adjustment greatly increases the storage and computational cost of PBAS. In contrast, the proposed method automatically adjusts the complexity of each DDHMM locally to the demands of the observations at each pixel. For example, the number of states created by the learning algorithm proposed in Section 4 are visualized for the Swing and Fields Point sequences in Figure 6. The algorithm adapts the complexity of each per-pixel intensity representation while the underlying DDHMM model compactly encodes the appropriate time scale for repeating intensity sequences.

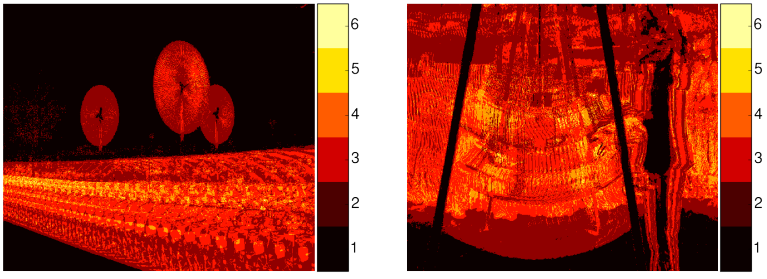
(a) Fields Point

Method	TPR	Precision	FPR	FNR
PBAS	99.3%	68.8%	0.82%	0.012%
ViBe	90.5%	54.7%	1.37%	0.174%
STLBP	91.9%	38.5%	2.68%	0.148%
SGMM-SOD	96.5%	67.2%	0.86%	0.064%
CBBGS	91.8%	40.2%	2.49%	0.149%
Proposed	98.6%	82.2%	0.39%	0.025%

(b) Swing

Method	TPR	Precision	FPR	FNR
PBAS	89.3%	30.2%	2.42%	0.13%
ViBe	80.8%	17.0%	4.63%	0.23%
SPLBP	76.6%	8.6%	9.55%	0.27%
SGMM-SOD	92.2%	23.9%	3.45%	0.09%
CBBGS	84.9%	30.8%	2.25%	0.18%
Proposed	80.7%	41.1%	1.36%	0.23%

Table 1: Quantitative change detection results showing True Positive Rate (TPR), Precision, False Positive Rate (FPR) and False Negative Rate (FNR).



(a) Fields Point sequence. (b) Swing sequence.

Figure 6: Number of DDHMM states instantiated at each pixel.

Table 1 quantifies the performance of the proposed and competing algorithms for the Fields Point and Swing video sequences. The competing algorithms contain a large number of false positives in every frame of video for both scenes. The blades of the turbine in the Fields Point video and the swinging child in the Swing scene are always incorrectly labeled as change and thus report each frame of video to the end user for further inspection. These algorithms would require additional reasoning modules to reliably filter their output for use in a surveillance application. However, the proposed method is able to discriminate between the normal repeating phenomena and previously unobserved events resulting in superior precision and false positive rate.

7 Conclusion

This paper presented a novel real-time algorithm for learning the complexity and parameters of Duration Dependent Markov Models at each pixel in surveillance video. Using the state assignments made by the local DDHMMs, a codebook based classifier was used to detect changes of interest in scenes with stochastically repeating phenomena with arbitrary time-scales. Further directions will explore computationally feasible methods for relaxing the proposed algorithm's spatial independence assumptions. Additionally, the DDHMM model offers a powerful local representation that could be used as a basis for higher level scene segmentation as a precursor to scene understanding.

References

[1] H. Akaike. A new look at the statistical model identification. *Automatic Control, IEEE Transactions on*, 19(6):716–723, Dec 1974. ISSN 0018-9286. doi: 10.1109/TAC.1974.1100705.

[2] O. Barnich and M. Van Droogenbroeck. Vibe: A universal background subtraction algorithm for video sequences. *Image Processing, IEEE Transactions on*, 20(6):1709–1724, June 2011. ISSN 1057-7149. doi: 10.1109/TIP.2010.2101613.

[3] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, 2006.

[4] T.V. Duong, H.H. Bui, D.Q. Phung, and S. Venkatesh. Activity recognition and abnormality detection with the switching hidden semi-markov model. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 838–845 vol. 1, 2005. doi: 10.1109/CVPR.2005.61.

[5] Ruben Heras Evangelio, Michael Paetzold, Ivo Keller, and Thomas Sikora. Adaptively splitted gmm with feedback improvement for the task of background subtraction. *IEEE TRANSACTIONS on Information Forensics and Security*, Accepted for publication.

[6] Yann Guédon. Estimating hidden semi-markov chains from discrete sequences. *Journal of Computational and Graphical Statistics*, 12(3):pp. 604–639, 2003. ISSN 10618600. URL http://www.jstor.org/stable/1391041.

[7] M. Hofmann, P. Tiefenbacher, and G. Rigoll. Background segmentation with feedback: The pixel-based adaptive segmenter. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pages 38–43, 2012. doi: 10.1109/CVPRW.2012.6238925.

[8] P. Kaewtrakulpong and R. Bowden. An improved adaptive background mixture model for real-time tracking with shadow detection. In *Proceedings of 2nd European Workshop on Advanced Video Based Surveillance Systems*, volume 5308, 2001.

[9] Kyungnam Kim, Thanarat H. Chalidabhongse, David Harwood, and Larry S. Davis. Real-time foreground-background segmentation using codebook model. *Real-time Imaging*, 11:172–185, 2005. doi: 10.1016/j.rti.2004.12.004.

[10] Chris Stauffer and W. E L Grimson. Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2, pages –252 Vol. 2, 1999. doi: 10.1109/CVPR.1999.784637.

[11] K. Tang, Li Fei-Fei, and D. Koller. Learning latent temporal structure for complex event detection. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1250–1257, 2012. doi: 10.1109/CVPR.2012.6247808.

[12] D. Tweed, R. Fisher, J. Bins, and T. List. Efficient hidden semi-markov model inference for structured video sequences. In *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on*, pages 247–254, 2005. doi: 10.1109/VSPETS.2005.1570922.

[13] Shengping Zhang, Hongxun Yao, and Shaohui Liu. Dynamic background modeling and subtraction using spatio-temporal local binary patterns. In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pages 1556–1559, Oct 2008. doi: 10.1109/ICIP.2008.4712065.