A Symbolic Framework for Recognizing Activities in Full Motion Surveillance Videos

Manohar Karki[*], Saikat Basu[*], Robert DiBiano[*], Supratik Mukhopadhyay[*], Jerry Weltman[*], Malcolm Stagg[†],
[*]Louisiana State University Baton Rouge LA, 70803,USA
[†]Microsoft Corporation
Redmond, Washington, 98052, USA

Abstract—**We present a symbolic framework for recognizing activities of interest in real time from video streams automatically. This framework uses regular expressions to symbolically represent (possibly infinite) sets of motion characteristics obtained from a video. It uniformly handles both trajectory-based and periodic articulated activities and provides polynomial time graph algorithms for fast recognition. The regular expressions representing motion characteristics can either be provided manually or learnt automatically from positive and negative examples of strings (that describe dynamic behavior) using offline automata learning frameworks. Confidence measures are associated with recognition using Levenshtein distance between a string representing a motion signature and the regular expression describing an activity. We have used our framework to recognize trajectory-based activities like vehicle turns (U-turns, left and right turns, and K-turns), vehicle start and stop, a person running and walking, and periodic articulated activities like hand waving, boxing, hand clapping and digging in videos from the VIRAT public dataset, the KTH dataset, and a set of videos obtained from YouTube. Our framework is fast (it runs at nearly 3 times real time) and on the KTH dataset, it is shown to outperform three of the latest existing approaches.**

I. Introduction

Intelligence obtained from recognizing activities underlying the dynamics of moving objects [1], [2] in surveillance videos is a key enabler for many video analysis applications [3]. While many deployed surveillance systems provide automatic tracking, describing the activities of tracked objects still generally requires human intervention. Analysts are prone to miss events, and even if no events were missed, manually keeping a log of everything that happens in a video would not be viable. It is therefore essential to develop techniques to automatically analyze the motions and behaviors of objects in video streams. Potentially important events could then be flagged in real time, giving analysts a more manageable amount of data to handle.While in the past few years there has been a slew of research progress in real time activity recognition from videos, the general problem is inherently hard. Both trajectory-based activities and periodic articulated activities can have differences that are nuanced or maybe completely different. Periodic articulated ac based activities that we describe here consist of human body movements while standing still and the trajectory based activities describe the actual movement of vehicles or other objects in different frames of a video. Existing approaches have targeted individual activity recognition problems with specialized complex descriptor matching (such as bag-of-words or histogram-of-gradients), probabilistic logic, and classification algorithms. Complex descriptor matching can be computationally expensive. Despite the recent progress in activity recognition, there has been no uniform framework that can be efficiently used to solve a variety of problems and can be seamlessly integrated with reasoning platforms to provide inferences at a higher level of abstraction such as anomalies. We propose a framework for automatically recogning activities in real time from a video stream. The frameworks breaks down the activities in a series of symbols before recognizing them, hence, we call it a symbolic framework. Regular expressions are able to uniformly handle both trajectory-based and periodic articulated activities and provide polynomial time graph algorithms for fast recognition. The regular expressions representing motion characteristics for activities are in many cases simple enough to be provided manually (e.g., by an expert analyst or by crowd sourcing, etc.) providing a *generative* framework; or they can be learnt automatically from positive and negative examples of strings describing dynamic behavior using offline automata learning frameworks like libalf [4]. Confidence measures are associated with recognition using Levenshtein distance between a string representing a motion signature and a regular expression describing an activity [5]. Since regular languages described by regular expressions are as expressive as monadic second-order logic over strings (MSO-S) [6], we get for free, a rich logical framework that can be integrated with reasoning platforms for performing high level inference by linking together activities. We have used our framework to recognize trajectory-based activities like vehicle turns (U-turns, left and right turns, and K-turns), start and stop, a person running and walking, and periodic articulated activities like hand waving, hand clapping, boxing and digging holes in the ground observed in videos from the VIRAT public dataset [7], the KTH dataset [8], and a set of videos obtained from YouTube. Further details and the experimental results of these example provided in Section V.

II. Related Work

Our approach builds on top of our robust tracking framework that uses a combination of foreground background segmentation and a color histogram model to compensate for trajectory failures. Frameworks have been developed that consider activities as resulting from the execution of a dynamical

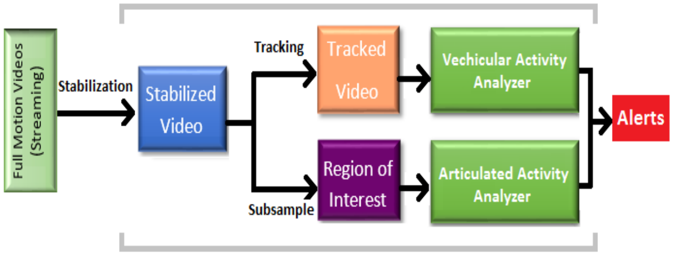

Full Motion Videos (Streaming)
Stabilization
Stabilized Video
Tracking
Tracked Video
Subsample
Region of Interest
Vechicular Activity Analyzer
Articulated Activity Analyzer
Alerts

Fig. 1. The architecture of the proposed framework.

system [9], based on 3D Markov Random Field to recognize trajectory-based and articulated activities like dancing, talking, etc. [10]. In contrast, we use a symbolic regular expression-based framework for representing possibly infinite sets of motion signatures. Using a normalized edit distance measure, representation and retrieval of object trajectories has also been proposed [11]. A generative mixture model was designed for activity recognition using the velocity data collected from trajectories [12]. Context-free grammars have proven to be useful to recognize complex activities before like in [13]. Time complexity for recognition of context-free languages is $O(n^3)$ whereas that for regular ones is linear and hence computationally less expensive. Besides, context-free languages cannot be directly tied to a logical framework unlike regular expressions. Other frameworks include representing contexts to facilitate activity recognition [14] and logic programming [15]. Our regular expression-based framework is equivalent to MSO-S which lies at the boundaries of decidability and thus has more expressive power compared to first order logic (over strings) [16]. Deep Learning and Convolution Neural Networks have made strides in image analysis and video recognition such as [17], [18] and[19]. These techniques require huge amount of data and state of the art processing capabilities and whereas our approach works with smaller data size and mediocre processors. A Symbolic Framework is also more robust to addition of new classes as the entire model doesn't need to be trained again. Dynamic Bayesian Networks have been used for detecting activities in [20] and [21]. We use a measure based on the Levenshtein distance to quantify the accuracy (or uncertainty) of recognition of an activity. Markov Logic Network has been used to recognize interesting activities in video [22]. The power of finiite state machines to describe dynamic systems and the ease of use is described in [23]. [24] present a scalable classifier system that works on a high dimesional problems using an encoder called XCS based on finite state machines. These advancements make the use of symbolic framework a solid approach in problems such as recognizing activites. Also, [25] propose a framework based on qualitative spatio-temporal graphs and graph isomorphism (similarity). [26] represent sequences of complex arm-hand acting by a robot using regular languages. We represent human motion by using directional histograms that are fitted with Gaussians instead.

III. The Proposed Framework

Activities are associated with possibly infinite sets of motion signatures (think of the numerous ways in which a U-turn can be made). In our framework, activities are represented by regular expressions describing their motion signature. Incoming full motion video is first input to a stabilization engine for jitter correction and background noise subtraction. The stabilized video is then input to a tracker. The output of the tracker is input to an activity analyzer. The analyzer smooths the data and from the tracked video, extracts strings describing motion signatures of moving objects in it. The strings are then matched against regular expressions representing activities using approximate pattern matching algorithms for "soft" matching. The various components of the symbolic framework are described in detail below. An overview of the Framework is presented in Fig. 1.

A. Regular Expressions, Directions, and Periodicity

A regular expression [27] describes a pattern representing a (possibly infinite) set of strings (i.e., a language) over an alphabet. In addition to supporting efficient string search operations which are polynomial in the number of bytes of data to be searched, regular expressions offer great flexibility in symbolically describing sets of strings and can be conveniently represented by finite state automata that allows efficient manipulation using graph algorithms. In terms of expressiveness, regular languages (i.e., those described by regular expressions) are as expressive as monadic second-order logic over strings (MSO-S) [6]. MSO-S subsumes temporal logics like LTL [28] that are popularly used for describing dynamic behavior sequences. We will use the flexibility of the framework of regular expressions to describe and classify object motions and recognize underlying activities. Fig. 2 shows an example where we use the characters a through h to represent unit vectors in both the cardinal (N, S, E, W) and ordinal (NW, NE, SW, SE) directions respectively. The optimal unit length depends on scale and noise but our experiments used numbers on the order of 1 pixel width. Any arbitrary trajectory can be approximated by the combination of multiple unit vectors. Using this language of motion description, the string abc represents a movement to the east, then from there to the northeast, then from there to the north. In practice, we use 24 characters to more finely quantize direction, rather than 8. These unit vectors are represented by the characters a through x.

Articulated activities with periodic motion like digging, gesturing, etc., can also be mapped to regular expressions as shown in Fig. 6 and illustrated in Section III-D. Here, we fit Gaussians with the running average of the directional histogram differences as data. The widths of Gaussians are subsequently mapped to character strings to extract the periodicity information in them.

B. Classifying Motions

To classify motion signatures obtained from a video, one first needs to track moving objects. From the trajectories,

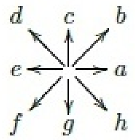


Fig. 2. Cardinal and ordinal directions are mapped to characters.

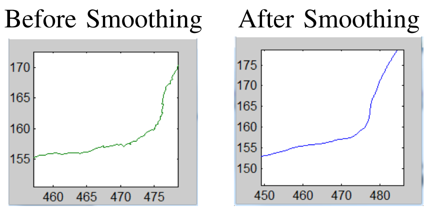


Fig. 3. The position (x, y) of a left-turning vehicle before and after smoothing.

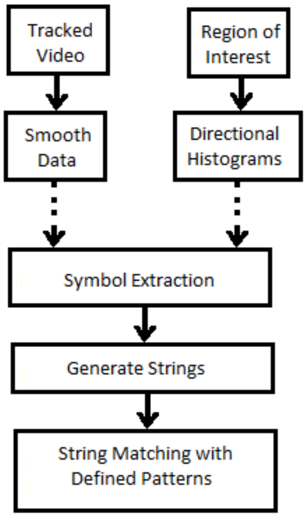


Fig. 4. Classifying Motions

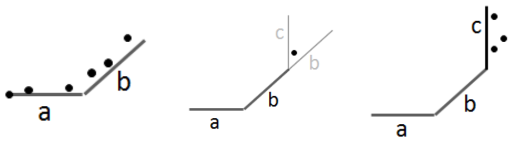


Fig. 5. Symbolic mapping as trajectory updates.

symbols are extracted corresponding to motion characteristics in individual frames. Finally, the strings obtained from the video are (approximately) matched against regular expressions representing activities.

1) Tracking: We use an in-house tracking algorithm based on a combination of foreground-background segmentation and a color histogram model, a detailed description of which is beyond the scope of this paper.

2) Symbol Extraction: Symbol extraction is the process of translating tracked object positions into strings of characters.

Symbol extraction involves two steps: smoothing of the data and mapping of the data into character symbols.

3) Data Smoothing: Object tracking is imperfect; the trajectories of real-world motion usually do not follow any smooth pattern. For example, even if a car makes a smooth turn, on close inspection the trajectory appears jagged due to small additive noise. Raw trajectories, if left unsmoothed, are often incorrectly mapped.

The data must be smoothed to improve accuracy. To smooth the data, we use a simple Gaussian filter with a width of α.

This straightforward approach to smoothing has a profound effect, not only serving to eliminate faulty trajectories that would lead to erroneous classification, but also supports greater accuracy in the data mapping process, as the derivative is less noisy (Fig.reffig:smoothing).

C. Mapping Vehicle Motions to Strings

Data mapping is the process of converting motion vectors into strings of characters which may be efficiently matched by regular expression patterns.

In the ideal case, the vehicle moves exactly one unit per frame in one of the prescribed directions. In this case, we can simply generate as output one character per frame, and the mapping conforms precisely to the actual motion. However, this is usually not the case. Subpixel movement is often significant, so an object may move less than a unit between frames. Conversely it may move multiple units. We accumulate sub-unit movement until it can be mapped into a single unit vector.

Figure 5 (left) shows a smoothed trajectory as a series of dots. Superimposed on this, is the trajectory as it would be approximated by a sequence of symbols. In this case, motion from the first three frames is approximated by horizontal movement (symbol a), followed by the next three frames approximated by symbol b. Figure 5 (middle) shows the next sub-unit movement. At this point, there are two choices: keep accumulating movements, or map the movement to a symbol. The decision depends on the amount of error that would result from such a mapping. We take the derivative at the current point and compare this slope against the slopes of eight (or 24) possible symbols that could be generated. If the slope is between the slopes of two symbols, they become the candidates for the next mapping. In the figure, the candidates for the next mapping are b and c, shown in gray. We pick whichever minimizes the error (i.e., Euclidean distance) between the approximated trajectory and the actual movement. In this case, error is minimized by not generating a new symbol, and we continue to accumulate movement each frame.

Figure 5 (right) shows what happens after two more positions have been accumulated. Now symbol c can be mapped because it is the candidate that minimizes error. The strings representing other trajectory-based motion signatures for vehicles such as start and stop and those for humans (i.e., those associated with walking, running, etc.) can be determined similarly.

D. Mapping Articulated Motions to Strings

Articulated motions are associated with human activities like digging, waving, boxing, clapping, etc. We use the directional histograms of pixel intensities in the frames of a target video to map them into a regular expression. The directional histograms help in capturing the periodic motion in the articulated activities like clapping, gesturing, digging etc. Examples articulated motions is shown in Fig. 10 (c). Here, the directional histogram for the x-axis in the lower part of the video is periodic. Similarly, for the hand waving video in Fig. 10 (d), the horizontal directional histogram for the upper part of the video is periodic in nature.

1) Representing Articulated Activities with Directional Histograms: Directional histograms are computed separately for the x and y axes for a 2-dimensional image. The average of the i (Intesity) values on each pixel along the x and y axis describes a histogram along that axis. Our tracking framework identifies the region of interest from the trajectories of humans or vehicles. Histogram differences are then calculated frame-by-frame from the video sequence. A running average of the histogram differences that decays with time (e.g.,150 frames) is accumulated in bins. The first bin represents the difference between consecutive frames and the second bin represents the difference of histograms that are 2 frames apart and so on. A low value in the n^{th} bin would mean the frames that are n bins apart are similar and suggest that the motion repeats every n frames.

2) Mapping the histogram data into strings: For articulated activities, directional histograms are fitted with Gaussians whose standard deviations are quantized into 10 levels and then mapped to characters. We fit a linear composition of a mixture of Gaussians with the horizontal and vertical histograms as data as shown in Fig. 6. A Gaussian function is given by $f(x) = ae^{-(x-b)^2/2c^2}$ for some constants a, b, and c. Here, the periodicity information can be approximately mapped to the value $2c^2$. So, we get the corresponding period of motion as $P_i = 2{c_i}^2$. Thus we can derive strings of the form $P_1 P_2 P_3 ... P_n$.

E. Defining Patterns of Motion

The general form of a turn expression consists of three parts: a straight segment before the turn, a curved segment during the turn and a second straight segment indicating the turn has stopped. By varying the maximum/minimum lengths of these parts we can define what should be considered a turn. Our method does make some general assumptions about scale but scale need not be accurate beyond an order of magnitude. After a turn or other motion has been positively identified, the corresponding characters are consumed. Consider the case of the northbound left turn; the vehicle moves from east to north, traveling the intervening northeast direction. The string matching such a motion must begin with a, end with c, and may contain only a, b, and c as intervening characters; thus the corresponding pattern is $/a[abc]^+c/$. The general form of the left-turn expression is as follows:

$$a^s\{\{a,b,c\}^l \cup \{a,b,c\}^{l+1} \cup \{a,b,c\}^{l+2} \cup \ldots \cup \{a,b,c\}^u\}c^f \tag{1}$$

This expression classifies left turns that begin facing due east. A similar expression is used for each of the other 7 (or 23) directions. Right-turn patterns are simply the reverse of left-turn patterns. U-turn expression are similar except the starting and ending directions are 180 degrees apart, and there are a combination of more symbols in the middle. The general expression has four parameters *s*, *l*, *u* and *f* that can be tuned based on data by the analyst.

- *s*, *f*: minimum length of the start and finish of the turn
- *l*, *u*: lower and upper limit for the middle of the turn

Figure 7 shows how an actual turn is encoded. In this analysis, we assume that *s*=1, which means the start of the turn can be just one symbol; *f*=3, which means that the finish of the turn must be a string of at least three symbols that are 90 degrees from the start. The middle of the turn is any combination of the three symbols in between and including the start and finish symbol, with a minimum lower length of *l* = 10 and maximum upper length of *u* = 60. The figure shows a trajectory of a left turn from due east to north, so the start symbol is a, the middle portion is a combination of 35 *a*'s, *b*'s, and *c*'s, and the finish is a string of *c*'s. The general form of a K-turn expression is as follows:

$$\{\{a\Sigma^* b\} p \Sigma^* q \{a\Sigma^* b\}\}^+ \tag{2}$$

where, *p* and *q* are characters 180 degrees apart and *a* and *b* are mutually opposite characters skewed by 2 to 3 characters and Σ is the alphabet. The symbol generation rate can be used to check activities like start, stop, walking, running. For articulated activities we can derive regular expressions as well, e.g., the regular expression derived for waving is of the form $\{\{c\} \cup \{b\}^*\}^+$ A detailed discussion on the way these

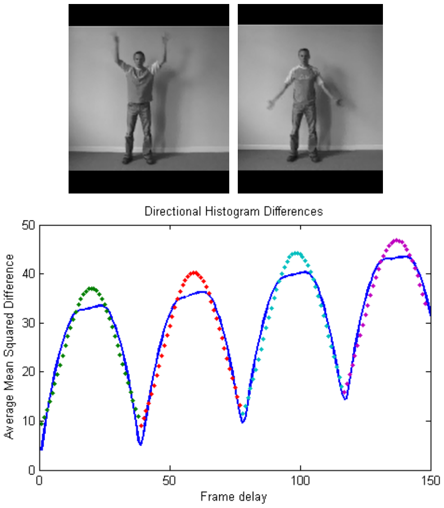
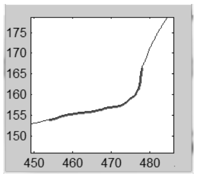


Fig. 6. Running average of Mean Squared Difference of Directional Histograms, fitted with Gaussians for Handwaving action.



baaaaaaa**abaaaabaabaabaabaaaaabaabaaababbbbcbccc**cbbcbcbcbcbbcbbbbcabc

Fig. 7. This graph shows the tracking data from an actual turn and the string to which it is mapped.

regular expressions are derived is presented in section III-D and further illustrated in section IV.

Algorithm 1 Levenshtein Distance Computation

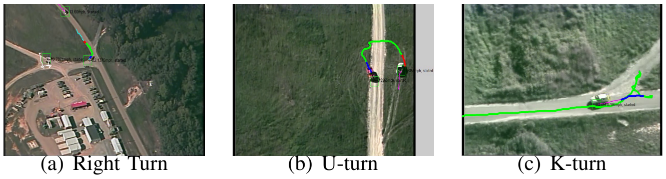
```
 1: $i \leftarrow 0$;
 2: $LD \leftarrow String$.Length();            ▷ Livenshtein Distance
 3: MAPTOSTATES();   ▷ Map characters of Input String to
    the states of Automaton
 4: if $FinalStateReached = AcceptingState$ then
 5:     $StringAccepted$ = TRUE;
 6: else
 7:     while $CurrentState \neq AcceptingState$ do
 8:         $i \leftarrow i + 1$;
 9:         BACKTRACK($i$);    ▷ Backtracks i steps through
    the automaton
10:         $LD \leftarrow LD - i$;
11:         DFS($i$);    ▷ Performs Depth First Search upto a
    depth of i
12:     end while
13: end if
```

F. Confidence Measure and Approximate String Matching

Standard pattern matching algorithms for regular expressions provide hard matching. To obtain soft matching, we use an approximate matching algorithm. This algorithm provides a confidence measure for a string approximately matching a regular expression based on how closely it matches the expression. It computes the confidence measure using the Levenshtein distance between a string and a regular expression. The Levenshtein distance LD is well knowned distance measure to calculate the distance between a string s and a regular expression R. It is given by $min_{s' \in \mathcal{L}(R)} \bar{LD}(s', s)$ where $\mathcal{L}(R)$ is the regular language associated with R and $\bar{LD}$ is the standard Levenshtein distance between two strings, and s and s' have the same length. We have designed an algorithm for computing the Levenshtein distance LD between a regular expression R and a string s; the algorithm is based on repeated depth-first graph search and with time complexity $O(k^2)$ where k represents length of the string s. This quadratic complexity can be attributed to the fact that for a string of length k, the while loop in line 7 can iterate over k states and for each sate the BACKTRACK() in line 9 can iterate for at most k states. Given the Levenshtein distance LD, the equation for computing the confidence measure of a string s matching a regular expression R is given as:

$$CM = \frac{length(s) - LD}{length(s)} \tag{3}$$

The above method is illustrated in Algorithm Listing 1 in which outputs the Levenshtein distance between the given string and the regular expression. A DFS (Depth First Search) up to a depth of i means that at each iteration, we traverse only the nodes of the automaton that lie within a depth of i from the starting node and the DFS stops when the accepting state is reached, which means the input string is identified as an activity.



(a) Right Turn (b) U-turn (c) K-turn

Fig. 8. Detection of Right turn, U-turn and K-turn by our Algorithm.

IV. From Strings to Regular Expressions

While in many cases, the regular expressions representing activities are simple enough to be provided manually, we can also use offline automata learning algorithms for learning regular patterns from positive and negative examples of strings encoding motion characteristics.

A. Learning the regular expressions by classification of strings

Once we have formed the strings representing positive and negative examples, we use a learning algorithm to infer a finite state automaton representing the regular language that accepts strings belonging to the language. For this, we use the RPNI (Regular Positive Negative Inference) offline learning algorithm [4] to learn the finite state automaton. This is implemented using the libalf (Automata Learning Framework) library, which is open-source, modular and comprehensive. Once, we input the strings belonging to the positive and negative classes, the library function calls upon the learning algorithm to infer an automaton that accepts all the positive examples and none of the negative ones. Next, the automaton is converted to a regular expression using the JFLAP library available online. The input strings representing motion signatures can be matched against the regular expressions to detect trajectory-based or articulated activities.

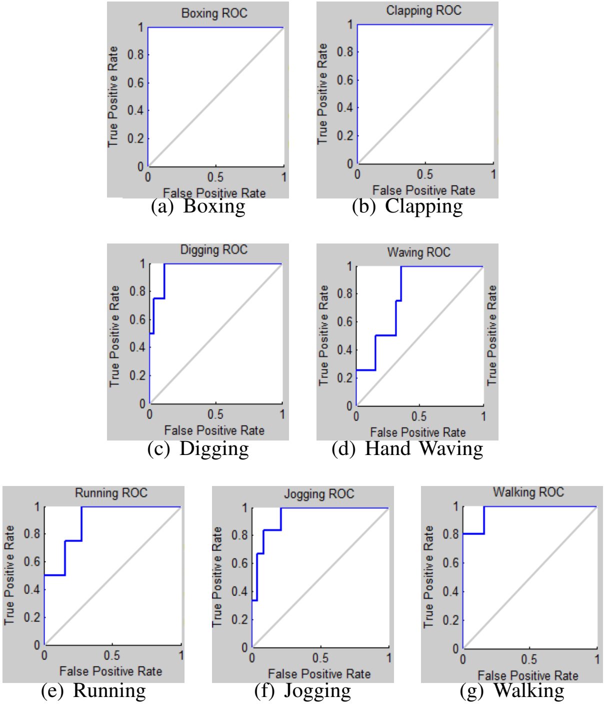
V. Experimental Results

Fig 9 shows the ROC curves for the different activities recognized by our activity recognizing module. We use multiple binary classifiers (simple softmax classifiers) concurrently, each producing a probability per activity. By comparing these probabilities, we find the most likely activity. The digging examples are obtained from YouTube while the other articulated activity examples are from the KTH dataset [8]. Examples in

	False Positives	Correct Detections	Total Expected
Left turn	8	4	5
Right Turn	2	7	8
U-Turn	2	2	2
K-turn	1	1	1
Walking	4	16	21
Starting	0	5	6
Stopping	1	3	4
Running	0	1	1

TABLE I

DETECTION RATES USING THE STRING MATCHING APPROACH (WITHOUT LEARNING), FOR VIRAT DATASET.



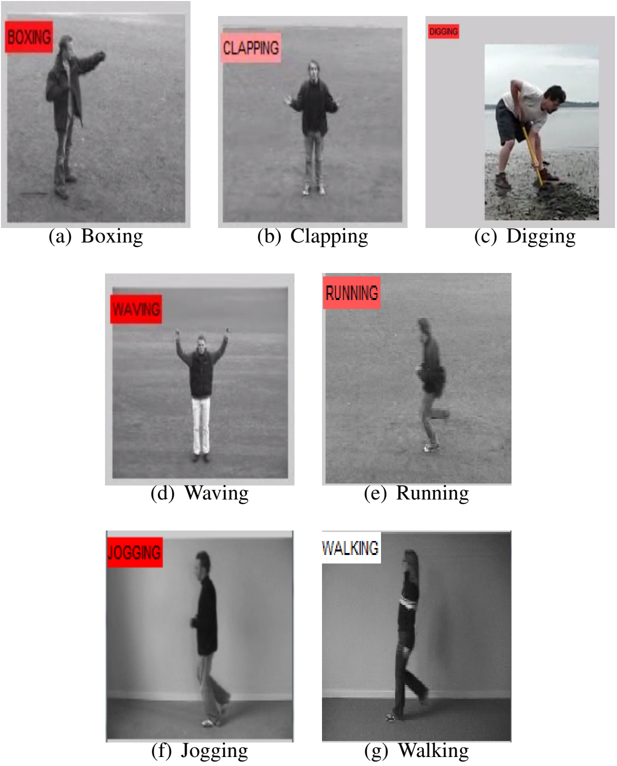
(a) Boxing (b) Clapping

(c) Digging (d) Hand Waving

(e) Running (f) Jogging (g) Walking

Fig. 9. ROC curves for different activities.

these datasets are segmented to contain one activity each. For all these activities, the best alogrithms such as [29] produce a detection rate of 95.83% while the approach in [30] produces a detection rate of 94%.[31] report their best result to be 92.1% at 4.6 fps (frames per second). Our framework outperforms these approaches by producing correct detection rates of 96%. On a 29.97 fps video, our algorithm is able to process frames at 86 fps on an Intel Centrino machine, that is nearly three times real time. Table I shows the results of using our framework for trajectory based activity examples obtained from the Virat Public Dataset [7] where the “Total Expected” column is based on ground truth. On the other hand, the Automata Learning Framework was able to correctly detect 67 out of 94 test samples at 21 false positives for both trajectory and articulated activities. We did not include u-turns and k-turns, because of lack of examples. For the set of articulated activites, we did statistical significance test on our results for the samples and accepted them with a p-value of 0.03 . We evaluated them on separate and equal number of samples from each class, where applicable, as there are no separate test examples. So, The learning framework shows promising results for turns and is expected to perform better in cases where we have a larger training set. Fig. 8 shows screenshots of some trajectory-based turn detection examples. The first image is right turn where the vehicle is travelling south-east and makes a turn towards south-west. The second image shows a trajectory of a car travelling north and then making a u turn and go the opposite direction and the final image is a car that makes a K-turn to go the opposite direction. Fig. 10 shows the results from the activity classification module.



(a) Boxing (b) Clapping (c) Digging

(d) Waving (e) Running

(f) Jogging (g) Walking

Fig. 10. Screen shots of different activities detected by our algorithm.

VI. CONCLUSION

This paper presents a rich symbolic framework based on regular expressions for recognizing diverse types of activities in surveillance videos. Though simple, the framework not only provides fast algorithms for activity recognition but is also flexible enough to admit both generative and learning-based approaches. We plan to integrate our framework with reasoning engines to provide inference capabilities at a higher level of abstraction. We also plan to compare to incoporate the Probabilistic Finite State Automata (PFA) [32] and make comparisions with the current approach as PFAs can be learnt from a set of strings using Expectation-Maximization.

VII. ACKNOWLEDGEMENT

The project is partially supported by Army Research Offi ce (ARO) under Grant #W911-NF1010495. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the ARO or the United States Government.

References

[1] P. Turaga, R. Chellappa, V. Subrahmanian, and O. Udrea, "Machine recognition of human activities: A survey," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 18, no. 11, pp. 1473 – 1488, nov. 2008.

[2] J. Aggarwal and M. Ryoo, "Human activity analysis: A review," *ACM Comput. Surv.*, vol. 43, no. 3, pp. 16:1–16:43, apr 2011.

[3] M. Shah, O. Javed, and K. Shafique, "Automated visual surveillance in realistic scenarios," *IEEE MultiMedia*, vol. 14, pp. 30–39, January 2007. [Online]. Available: http://dl.acm.org/citation.cfm?id=1262177.1262230

[4] B. Bollig, J.-P. Katoen, C. Kern, M. Leucker, D. Neider, and D. R. Piegdon, "libalf: The automata learning framework," in *In Proceedings of CAV*, 2010.

[5] P. Muzatko, "Approximate regular expression matching," Czech Technical University, Tech. Rep., August 1996.

[6] P. Tesson and D. Therien, "Logic meets algebra: The case of regular languages," in *In Proceedings of LICS*, 2005.

[7] S. Oh, A. Hoogs, A. G. A. Perera, N. P. Cuntoor, C.-C. Chen, J. T. Lee, S. Mukherjee, J. K. Aggarwal, H. Lee, L. Davis, E. Swears, X. Wang, Q. Ji, K. K. Reddy, M. Shah, C. Vondrick, H. Pirsiavash, D. Ramanan, J. Yuen, A. Torralba, B. Song, A. Fong, A. K. R. Chowdhury, and M. Desai, "A large-scale benchmark dataset for event recognition in surveillance video," in *CVPR*, 2011, pp. 3153–3160.

[8] C. Schüldt, I. Laptev, and B. Caputo, "Recognizing human actions: A local svm approach," in *ICPR (3)*, 2004, pp. 32–36.

[9] B. Li, M. Ayazoglu, T. Mao, O. Camps, and M. Sznaier, "Activity recognition using dynamic subspace angles," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, june 2011, pp. 3193 –3200.

[10] W. Choi, K. Shahid, and S. Savarese, "Learning context for collective activity recognition," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, june 2011, pp. 3273 –3280.

[11] L. Chen, M. T. Özsu, and V. Oria, "Symbolic representation and retrieval of moving object trajectories," in *Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval*. ACM, 2004, pp. 227–234.

[12] R. Messing, C. Pal, and H. Kautz, "Activity recognition using the velocity histories of tracked keypoints," in *Computer Vision, 2009 IEEE 12th International Conference on*, 29 2009-oct. 2 2009, pp. 104 –111.

[13] M. S. Ryoo and J. K. Aggarwal, "Recognition of composite human activities through context-free grammar based representation," *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, vol. 2, pp. 1709–1718, 2006.

[14] Y.-G. Jiang, Z. Li, and S.-F. Chang, "Modeling scene and object contexts for human action retrieval with few examples," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 21, no. 5, pp. 674 –681, may 2011.

[15] V. Shet, D. Harwood, and L. Davis, "Vidmap: video monitoring of activity with prolog," in *Advanced Video and Signal Based Surveillance, 2005. AVSS 2005. IEEE Conference on*, sept. 2005, pp. 224 – 229.

[16] H.-D. Ebbinghaus and J. Flum, *Finite model theory*. springer, 2005.

[17] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in Neural Information Processing Systems*, 2014, pp. 568–576.

[18] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition." in *ICML*, 2014, pp. 647–655.

[19] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[20] A. Hoogs and A. G. A. Perera, "Video activity recognition in the real world," in *Proceedings of AAAI*, 2008.

[21] Z. Zeng and Q. Ji, "Knowledge based activity recognition with dynamic bayesian network," in *ECCV (6)*, 2010, pp. 532–546.

[22] S. D. Tran and L. S. Davis, "Event modeling and recognition using markov logic networks," in *ECCV (2)*, 2008, pp. 610–623.

[23] J. Van Gurp and J. Bosch, "On the implementation of finite state machines," *Variability in Software Systems The Key to Software Reuse*, p. 45, 2000.

[24] M. Iqbal, W. N. Browne, and M. Zhang, "Extending xcs with cyclic graphs for scalability on complex boolean problems," *Evolutionary Computation*, 2015.

[25] M. Sridhar, A. G. Cohn, and D. C. Hogg, "Unsupervised learning of event classes from video." in *AAAI*, 2010.

[26] I. Gori, S. R. Fanello, F. Odone, and G. Metta, "A compositional approach for 3d arm-hand action recognition," in *Robot Vision (WORV), 2013 IEEE Workshop on*. IEEE, 203, pp. 126–131.

[27] J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, 1979.

[28] A. Pnueli, "The temporal logic of programs," in *FOCS*, 1977, pp. 46–57.

[29] M.-y. Chen and A. Hauptmann, "Mosift: Recognizing human actions in surveillance videos," 2009.

[30] N. Ikizler, R. G. Cinbis, and P. Duygulu, "Human action recognition with line and flow histograms," in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*. IEEE, 2008, pp. 1–4.

[31] H. Wang, M. M. Ullah, A. Klaser, I. Laptev, and C. Schmid, "Evaluation of local spatio-temporal features for action recognition," in *BMVC 2009-British Machine Vision Conference*. BMVA Press, 2009, pp. 124–1.

[32] E. Vidal, F. Thollard, C. De La Higuera, F. Casacuberta, and R. C. Carrasco, "Probabilistic finite-state machines-part i," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 7, pp. 1013–1025, 2005.