Efficient Action Recognition with MoFREAK

Chris Whiten, Robert Laganière
VIVA Lab
University of Ottawa
Ottawa, Canada
cwhit025@uottawa.ca, laganier@uottawa.ca

Guillaume-Alexandre Bilodeau
LITIV Lab
École Polytechnique de Montréal
Montréal, Canada
gabilodeau@polymtl.ca

***Abstract*—Recent work shows that local binary feature descriptors are effective for increasing the efficiency of object recognition, while retaining comparable performance to other state of the art descriptors. An extension of these approaches to action recognition in videos would facilitate huge gains in efficiency, due to the computational advantage of computing a bag-of-words representation with the Hamming distance rather than the Euclidean distance. We present a new local spatio-temporal descriptor for action recognition that encodes both the appearance and motion in a scene with a short binary string. The first bytes of the descriptor encode the appearance and some implicit motion, through an extension of the recently proposed FREAK descriptor. The remaining bytes strengthen the motion model by building a binary string through local motion patterns. We demonstrate that by exploiting the binary makeup of this descriptor, it is possible to greatly reduce the running time of action recognition while retaining competitive performance with the state of the art.**

***Keywords*-action recognition; spatiotemporal feature description; local binary descriptor**

I. Introduction

Recognizing actions in video sequences has garnered a great deal of interest in recent years. With recent advances in natural user interface design, algorithms for gesture recognition are in very high demand. Furthermore, robust and efficient classification of surveillance video footage is desirable for efficient traversal of large volumes of data. Although these are among the top applications for action recognition, recent works have strayed from the path of applicability, favouring algorithms with high recognition accuracy but colossal and sometimes unrealistic running times.

Our main contribution is in the presentation of a novel binary spatio-temporal feature descriptor that achieves significant running time improvements over state of the art methods, while remaining highly accurate. We extend recent work [1] in feature description for object recognition, where spatial neighbourhoods are described by compact binary strings. We introduce an implicit temporal component into the descriptor, while removing components that are of less value for recognizing actions. Furthermore, we adapt a recent approach [2] to action recognition for increased efficiency, where the geometric structure of motion is encoded by dense sampling of self-similarities. Our extension of this work improves efficiency by avoiding dense sampling through keypoint detection, while removing the dependency of requiring future frames to construct the descriptor. By removing such a requirement, the descriptors can be computed in an online fashion. Throughout the construction of this descriptor, emphasis is placed on ensuring the entire descriptor remains binary, gifting us with highly optimized processing and feature matching. This yields significant computational gains in approaches such as standard bag-of-words models, where thousands of matches must often be made at each frame.

II. Related Work

The avenues of research for action recognition generally fall into three categories: higher level representations, trajectory analysis, and local features with a bag-of-words representation.

Many techniques build complex representations in attempt to encode semantically meaningful descriptions. These representations are designed through a combination of low-level approaches, constructing a hierarchy of action representations. Higher level models are attractive due to their high recognition performance and intuitive construction. However, these systems suffer from an excessive amount of required computation time, making them unsuitable for most applications. Fathi and Mori [3] use tracking to localize an action, classifying short subsequences of the target to build weak classifiers from low-level actions. Adaboost [4] is then used to combine these into strong, higher-level classifiers. Jhuang et al. [5] introduce a biologically inspired system which builds a hierarchy of spatiotemporal feature detectors with increasing complexity. Basic features are detected, with methods such as optical flow [6], and a hierarchy of increasingly complex features are built on the basic features. Sadanand and Corso [7] construct a large bank of action detectors, which are manually selected to construct an "action space". Each query video is then embedded in this space based on its response to each detector, producing a single feature per video.

Tracking interest points and analyzing the motion trajectory has recently been shown to be another effective

approach for action recognition [8], [9], [10]. Efros et al. [9] track a target actor and compute motion descriptors based on its optical flow [6] after actor stabilization. The optical flow is projected onto several channels, corresponding to different motion components, and blurred to create four channels for the motion descriptor. Messing et al. [8] use the KLT feature tracker [6] on 3-dimensional Harris corners [11] to build descriptors based on the velocity history of keypoints. Wang et al. [10] use dense sampling to compute dense trajectories from the optical flow field. While they demonstrate state of the art results, the approach is infeasible for larger datasets due to the complexity of computing the Euclidean distance to match descriptors in a dense sampling paradigm.

The contribution in this paper falls under the domain of recognition with local spatio-temporal features. Using local spatio-temporal features with a bag-of-words model has remained an effective method for producing efficient and accurate action recognition algorithms. These approaches are appealing due to their increased efficiency over more complex algorithms, their competitive accuracy, and their ease of implementation. Laptev and Lindeberg [11] introduced this approach by extending the Harris corner detector into the temporal dimension. Detected corners in the spatial domain are rejected if there is insufficient change across time. Laptev et al. [12] improve upon that approach by detecting keypoints at multiple scales. Each detected point is described by histograms of oriented gradients (HOG) and optical flow. Chen et al. [13] continue this methodology by computing the SIFT features [14] and optical flow [6] at each frame, amalgamating the two descriptors into a single descriptor named MoSIFT which independently encodes both the motion and appearance of a scene. Detected SIFT locations with insufficient optical flow are rejected as keypoints, leaving only spatio-temporal keypoints on strong motions. Kliper-Gross et al. [2] capture the local geometry of motion with self-similarity [15] across the temporal domain. Each frame is densely sampled, where each pixel centers a self-similarity computation with a geometric pattern in one frame from the past and one frame from the future, generating a 64 trinary digit descriptor encoding the motion of the pixel's local neighbourhood. Our work is inspired by approaches that encode appearance and motion independently, while combining them into one simple feature; in particular, we followed the MoSIFT framework [16], one of the most successful methods for event detection at NIST TRECVID workshops. We also draw inspiration from the alternative motion representation followed in [2], where motion is captured through self-similarities.

III. MoFREAK

Several space-time descriptors, such as [12] and [13], leverage gradient-based methods to encode the appearance of an action, while using optical flow to capture its motion. While these works have demonstrated encouraging recognition accuracy, they falter in practice due to the complexity of computing these descriptors. A binary setup represents a desirable alternative for efficient keypoint detection, description, and matching.

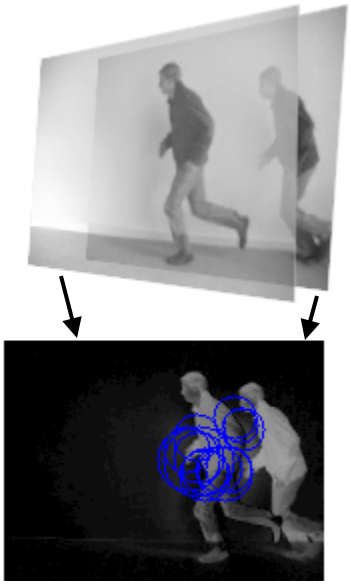

Figure 1. Keypoints are detected on the difference image between each frame and a frame few time steps away in the past, permitting implicit motion encoding and robustness to static environments.

Alahi et al. recently presented FREAK [1], a local binary descriptor with the potential to replace gradient-based appearance encodings. FREAK is constructed with short binary strings, based on intensity comparisons within a sampling pattern inspired by the human retina. FREAK descriptors are efficient to compute and compare, while presenting competitive accuracy compared to SIFT [14] in the object recognition domain. These results inspire us to explore its potential in the action recognition domain.

We present MoFREAK, a compact binary spatio-temporal feature descriptor which is inspired by recent advances in descriptor representation for both object and action recognition.

A. Appearance modelling

Several approaches for action recognition have successfully encoded action appearance by directly porting successful feature descriptors from object recognition into the action recognition domain [17], [11], [18]. We adopt a similar methodology in constructing MoFREAK, employing the recently proposed FREAK descriptor [1] as an appearance component. FREAK is advantageous by virtue of its binary nature, making it efficient to compute and match descriptors, while retaining high recognition accuracy.

At each time step t throughout a query video sequence, we compute the absolute difference image $A_t(x, y)$ between

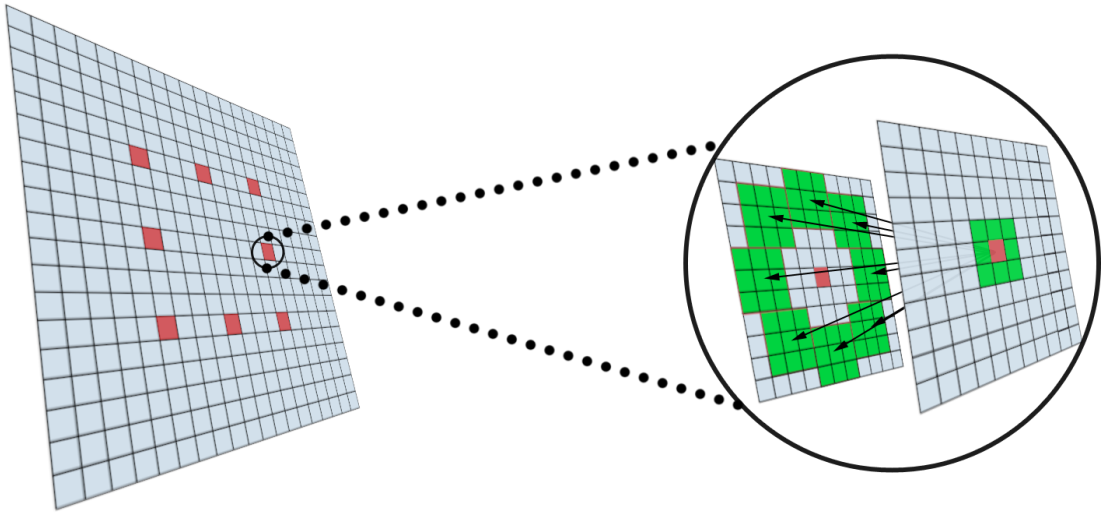

Figure 2. **Left:** Self-similarity computations are computed in 8 neighbouring locations around a detected keypoint. **Right:** A 3×3 patch is evaluated against a set of image patches in the past frame to compute 1-byte of the descriptor's motion component.

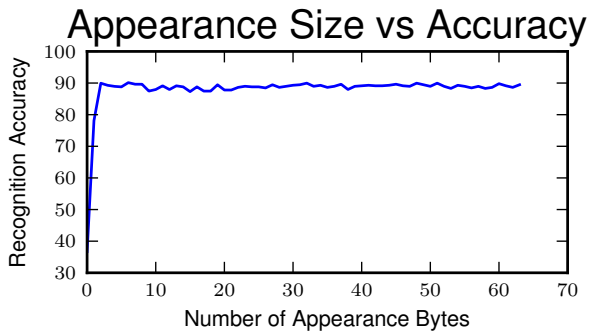

Figure 3. We compare the effect of the appearance component size on recognition accuracy. Significant increases are noted between 0 bytes and 1 byte of appearance data, and between 1 and 2 bytes of appearance data. The effect levels out when the appearance component is greater than 2 bytes, leaving us with a larger descriptor with no significant performance increase.

$F_t(x, y)$, the current frame, and $F_{t-\mathcal{D}}(x, y)$, the frame $\mathcal{D}$ time steps in the past.

$$A_t(x, y) = |F_t(x, y) - F_{t-\mathcal{D}}(x, y)| \quad (1)$$

The selection of parameter $\mathcal{D}$, denoting the temporal distance between two image frames, can greatly influence the results of this descriptor. For video sequences with a high frame rate, a small $\mathcal{D}$ will result in the comparison of two nearly identical images, resulting in a poor discriminator for motion. Conversely, videos with a low frame rate and a large $\mathcal{D}$ will have images which are temporally very far apart being compared. This scenario causes the descriptor to be overly sensitive and unable to capture subtle movements. Throughout the experiments in this paper, a $\mathcal{D}$ of 5 was used on videos with frame rates ranging from 25 to 30 frames per second.

On $A_t(x, y)$, we detect the location and scale of numerous keypoints with the BRISK detector [19] which is a fast scale-invariant keypoint detector. This process is demonstrated in Figure 1. From these detections, we extract FREAK descriptors within the frame $F_t(x, y)$ at the detected locations and scales. Using the difference image allows the detector to find keypoints that are interesting in both the spatial and the temporal domains, owing to the difference image's ability to implicitly encode both appearance and motion. Using the difference image has the added side-effect of performing rudimentary background subtraction, which assists in avoiding spurious features and avoids overfitting to static environments.

As described in [1], the context of the bytes in the FREAK descriptor increases from coarse to fine as the byte index increases. The first 16 bytes mainly involve the perifoveal receptive fields, roughly corresponding to a human's peripheral vision. The remaining bytes help distinguish between finer details, which are of less interest in action recognition since we do not want to match specific appearances, such as a specific actor in a scene. We have experimented with different descriptor sizes, the results of which can be viewed in Figure 3. For all possible sizes $x \in [0, 64]$, MoFREAK descriptors have been extracted from the videos in the KTH dataset using x bytes from the FREAK descriptor. Since FREAK only returns 64 bytes, larger descriptors cannot be tested. MoFREAK descriptors from each tested x are used

to evaluate recognition performance (using the approach described in Section IV). We find that maintaining only the first 2 bytes of the FREAK descriptor, while discarding the remaining 62 bytes, leaves us with an appearance model that is compact, efficient to match, and discriminative enough to robustly recognize actions. It is important to note that, while only 2 bytes of appearance data were used in the experiments in the paper, it is likely that there exist application domains where the appearance of a video sequence requires the descriptor to be more detailed to discriminatively capture and describe spatiotemporal regions. In such a scenario, it would be advantageous to use more than the suggested 2 bytes of appearance data.

B. Motion modelling

While the first bytes of the descriptor implicitly encode motion through the absolute difference image, it is insufficient for most action recognition applications due to its inability to capture the directionality of the motion at each keypoint. A common approach for encoding this information is optical flow, as can be seen in many popular action recognition algorithms, such as [13], [5], [3], [9]. While optical flow provides accurate motion representation, it is generally costly to compute and produces a dense set of floating point vectors, which is expensive to match and impede on progress towards real-time recognition.

Recently, Motion Interchange Patterns have been proposed for modelling motion by a series of self-similarity patch computations [2]. Each frame is densely sampled such that each pixel yields a 64 trinary digit descriptor, where each trinary digit corresponds to the result of SSD computations in the local space-time neighbourhood. We adopt a binary variation on this technique with an extended neighbourhood pattern to build an 8 byte motion descriptor.

To construct the motion component for a MoFREAK descriptor, the detected keypoint is resized to a 19×19 patch, on which we compute a series of self-similarity computations across the spatio-temporal domain. A self-similarity approach is taken to remain appearance invariant [15], which enables the model to learn the geometric structure of the motion, rather than encoding the details of the specific actor or environment. From this 19×19 patch, we perform identical computations on 3×3 patches centered at eight spatial locations. The computations at each of the eight spatial locations return a single byte of descriptor data, leaving us with a full 8-byte motion descriptor at the end of the process. These computations are centered at pixel locations $(5, 5)$, $(5, 9)$, $(5, 13)$, $(9, 5)$, $(9, 13)$, $(13, 5)$, $(13, 9)$, and $(13, 13)$, forming the pattern shown on the left-hand side of Figure 2. 19×19 patches are used because such a size allows for minimal overlapping information when rooting the smaller 3×3 patches at each of the eight spatial locations, while still being evenly spaced.

We denote the patch $p_t(x, y)$ as the 3×3 patch in the current frame F_t centered at spatial location (x, y). We wish to evaluate how the intensity values in $p_t(x, y)$ have changed, relative to its spatial neighbourhood in past frames. Moving five frames into the past, we define eight separate 3×3 patches at the frame F_{t-5} which encode possible locations in F_{t-5} of the structure in p_t(x, y). These eight new patches, $p_i(x, y)\ \forall\ i \in [1, 8]$, are defined at the following spatial locations relative to the location of p_t: $(-4, 0)$, $(-3, 3)$, $(0, 4)$, $(3, 3)$, $(4, 0)$, $(3, -3)$, $(0, -4)$, and $(-3, -3)$. Each p_i will be included in an SSD computation against p_t, leaving us with 8 calculations. This process is visualized in the right-hand side of Figure 2. We convert the resulting SSD value into a binary decision by comparing its value against threshold θ, giving rise to a natural 8-bit descriptor for this set of computations. In our experiments, a θ value of 288 was used, setting the motion threshold for bit activation to be an average intensity difference of 32 for each of the 9 pixel pairs when comparing two 3×3 patches. We define $b(i)$, the value of bit i, by the following equation:

$$b(i) = \begin{cases} 1 & : SSD(p_i, p_t) < \theta \\ 0 & : SSD(p_i, p_t) \geq \theta \end{cases} \tag{2}$$

The construction of this 8-byte descriptor is a simple set of independent SSD computations, making it efficient to compute and highly parallelizable. Encoding the intensity changes over several image patches within a keypoint's neighbourhood is advantageous compared to analyzing the movement of a single location, since complex motions over a larger area can be evaluated with our compact encoding. The choice of SSD comparisons in 8 directions between frames is motivated with compactness in mind. By using the pattern shown on the right-hand side of Figure 2, motion is captured with a single bit from each region surrounding the source pixel, which permits encoding the entire motion of the descriptor in a single byte.

IV. Recognition

For the recognition task, we use a standard bag-of-words representation. Studies have shown [20] that k-means clustering for codeword selection tends to overfit to the densest region of the feature space, resulting in a clustering that is often just as poor, if not worse than random cluster selection. Our experiments were consistent with that hypothesis in the action recognition domain, leading us to use class-balanced random clusters for visual codebook selection. With n classes and k clusters, $\frac{k}{n}$ descriptors are randomly selected from each class to be codewords. Since we are working with binary strings, the efficient Hamming distance is used to measure descriptor similarity.

For classifying the final bag-of-words features, a support vector machine (SVM) is used with the histogram intersection kernel. For bag-of-words features a and b, the histogram

	box	clap	wave	jog	run	walk
box	97	0	2	0	0	1
clap	5	95	0	0	0	0
wave	7	3	90	0	0	0
jog	0	0	0	79	9	12
run	0	0	0	11	83	6
walk	1	0	0	3	0	96

Table I
KTH CONFUSION MATRIX

Approach	**Accuracy**
Schüldt et al. [21]	71 %
Dollár et al. [18]	81 %
Laptev et al. [12]	91 %
MoSIFT [13]	87 %
Kovashka & Grauman [23]	95 %
Kliper-Gross et al. [2]	93 %
MoFREAK	90 %

Table II
RECOGNITION RESULTS ON THE KTH DATASET [21] FOR SIMILAR SPATIO-TEMPORAL FEATURE-BASED APPROACHES. MOFREAK RETAINS COMPETITIVE ACCURACY WITH SIMILAR STATE OF THE ART METHODS, DESPITE THE SIGNIFICANT INCREASE IN COMPUTATIONAL EFFICIENCY.

intersection kernel is defined as

$$K_{\cap}(a,b) = \sum_{i=1}^{n} min(a_i, b_i) \tag{3}$$

Although the χ^2 kernel is very popular in the action recognition literature, we found the difference in accuracy between the two kernels to be negligible, while the histogram intersection is more computationally efficient.

We selected the bag-of-words + SVM approach purely based on its simplicity and popularity. By using a generic action recognition setup, it is simple to evaluate the MoFREAK descriptor against future descriptors to benchmark performance and accuracy.

V. EXPERIMENTAL RESULTS

We evaluate performance on two standard benchmark datasets for action recognition. KTH [21] is a simple, single actor dataset that has been benchmarked by several approaches. In contrast, HMDB51 [22] is a recent dataset that is among the most difficult datasets due to its varying complexities. Furthermore, we compare the running times required by alternative approaches to process KTH, showcasing the significant computational advantage of MoFREAK.

All of our experiments were conducted on standard consumer hardware with unoptimized code. The experiments were run on Windows 7 with an Intel Core i7 870 2.93 GHz CPU. No GPUs or other hardware optimizations were used, and the C++ code is not highly optimized.

A. Experiments

KTH dataset: The KTH dataset [21] consists of 599 video sequences, each of which contains a single actor performing one of six actions (hand waving, clapping, boxing, walking, jogging, and running). 25 separate actors are used, each performing all six actions under different conditions (scaling, outdoors, indoors, and changed clothing). Each sequence is low resolution (160×120), and the average sequence length is 20 seconds.

MoFREAK features are detected and computed on every frame in each sequence, beginning at the sixth frame to allow computation of the absolute difference image. Then, the entire sequence is represented by a bag-of-words model with 600 randomly selected clusters, 100 clusters for each action class. We evaluate the performance of MoFREAK on this dataset through leave-one-out cross validation. For each of the 25 actors, we remove one for testing and train on the remaining 24, computing the recognition performance on that actor. We then average the results across all possible actors to achieve our final reported accuracy.

The confusion matrix for classification on KTH is displayed in Table I. The most significant areas of confusion are between walking, jogging, and running. Specifically, our system appears to occasionally confuse jogging with these two similar actions. To evaluate the effect of the MoFREAK descriptor, we evaluate against similar methods which employ spatio-temporal features with a bag-of-words model in Table II. We find that MoFREAK has comparable performance to the state of the art for comparable methods. For instance, we perform better than MoSIFT in accuracy and we are also significantly more efficient, which we expand upon in Section V-B. The results for MoSIFT have been generated from the binary provided by the authors for generating MoSIFT points from a video sequence. The remainder of the pipeline is identical to that used in MoFREAK's experiments.

HMDB51 dataset: The HMDB51 dataset [22] is a significantly more difficult dataset than KTH [21]. HMDB51 consists of 51 actions, each of which have at least 101 video representations, with a total of $6,849$ video clips to recognize. Due to the numerous sources that the videos are sampled from, such as movies and YouTube, the quality varies greatly from video to video. This dataset is currently among the most difficult datasets for benchmarking action recognition, with most algorithms achieving recognition accuracy in the neighbourhood of 20 %.

To evaluate our descriptor on this dataset, the 70 - 30 training/testing split presented in [22] is followed. This evaluation methodology selects 70 videos for training and 30 videos for testing from all 51 actions. The videos were hand-selected to contain videos of varying quality, environments, and camera motion. When computing features, we use a codebook of 5100 randomly selected features, 100 from each

Approach	Accuracy
Laptev et al. [12]	20.4 %
Jhuang et al. [5]	22.8 %
Sadanand & Corso [7]	26.9 %
Kliper-Gross et al. [2]	29.1 %
MoFREAK	18.4 %

Table III
RECOGNITION COMPARISON TO PREVIOUS RESULTS ON THE DIFFICULT HMDB51 DATASET [22].

Approach	Running Time (mins)	Accuracy
Fathi and Mori [3]	10,129	90%
Jhuang et al. [5]	1,198	90 %
MoSIFT [13]	449	87 %
MoFREAK	185	90 %

Table IV
COMPARING THE RUNNING TIMES OF PROCESSING KTH [21]. MOFREAK AND MOSIFT [13] WERE COMPUTED WITH OUR C++ IMPLEMENTATION, WHILE THE REMAINING APPROACHES ARE REPORTED RESULTS.

action, for the bag of words representation.

A summary of reported results are presented in Table III. These results are all reported on the more difficult, original HMDB51 dataset, rather than the stabilized alternative. The current state of the art is achieved with Motion Interchange Patterns [2], which includes a motion stabilization component within the algorithm. We note that, while we do not have greater accuracy than the remaining approaches, we remain close while achieving significant performance speedups, as discussed in Section V-B.

B. Computational Cost

One of the main focal points of this approach is ensuring a low computational cost, granting feasibility on large datasets. Action Bank [7] reports high accuracy on several datasets, but its applicability is limited, due to the overhead involved in building the action detectors and the amount of time required to process each video. Although the time required to build the detectors is not reported, processing a single UCF50 video requires, on average, 204 minutes. Processing a comparable video with our unoptimized implementation of MoFREAK requires approximately 1 minute, a $200\times$ speed increase over Action Bank.

To showcase the performance gains of MoFREAK, we compare the running time of MoFREAK against similar spatio-temporal descriptor approaches with reported running times, outlined in Table IV. This evaluation is done on the KTH dataset, since there an insufficient number of approaches report running times on other datasets to make a valid comparison. We have also run an implementation of MoSIFT [13] for our comparisons. Fathi and Mori [3] report a computationally efficient method in which classification takes between 0.2 and 4 seconds per frame on KTH [21]. Using the midpoint of 2.1 for computation on the $11,576$ seconds of KTH footage at 25 frames per second, this adds up to over $10,000$ minutes. Assuming a best-case scenario of 0.2 seconds for each frame, the method still takes 965 minutes to process. Jhuang et al. [5] report that a typical run takes over 2 minutes per video sequence, processing only 50 frames per sequence, summing to over 1198 minutes for KTH. In contrast, we processed the entirety of KTH in 185 minutes, which is a shorter time period than the actual length of the KTH dataset. While these reported running times would not remain exact on modern day hardware, they are still indicative of the complexity of the solutions. Furthermore, while tight computational optimizations and GPU implementations may lead to near real-time code in restricted domains, MoFREAK would also be a good candidate for such optimizations, leading to even further computational gains.

VI. CONCLUSION

We have presented a compact, 10-byte binary spatio-temporal keypoint descriptor, MoFREAK, which simultaneously encodes the local appearance and the geometric structure of the motion of a local space-time neighborhood. The descriptor takes advantage of recent advances in binary feature descriptors to avoid costly gradient computations that are typically present in space-time descriptors. The C++ source code for this descriptor is available online. [1]

We have shown that this approach has significant computational advantages, being faster than the state of the art by several orders of magnitude, while retaining competitive recognition accuracy. On the KTH dataset [21], we achieved 90 % recognition accuracy, while being several times faster than previous methods.

Throughout this paper, our focus has been on the descriptor rather than the overall recognition system, leading us to use a very simple bag-of-words representation with a simple SVM classifier. A more extensive classification setup may lead to recognition accuracy greater than what we have reported, and is an area of future work.

REFERENCES

[1] A. Alahi, R. Ortiz, and P. Vandergheynst, "FREAK: Fast Retina Keypoint," in *CVPR*, 2012.

[2] O. Kliper-Gross, Y. Gurovich, T. Hassner, and L. Wolf, "Motion interchange patterns for action recognition in unconstrained videos," in *ECCV*, 2012. [Online]. Available: http://www.openu.ac.il/home/hassner/projects/MIP

[3] A. Fathi and G. Mori, "Action recognition by learning mid-level motion features," in *CVPR*, 2008.

[1] Source code available: http://www.eecs.uottawa.ca/~laganier/projects/mofreak

[4] R. E. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions," in *Machine Learning*, 1999.

[5] H. Jhuang, T. Serre, L. Wolf, and T. Poggio, "A biologically inspired system for action recognition," in *ICCV*, 2007.

[6] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," 1981.

[7] S. Sadanand and J. J. Corso, "Action bank: A high-level representation of activity in video," in *CVPR*, 2012.

[8] R. Messing, C. Pal, and H. Kautz, "Activity recognition using the velocity histories of tracked keypoints," in *ICCV*, 2009.

[9] A. A. Efros, A. C. Berg, E. C. Berg, G. Mori, and J. Malik, "Recognizing action at a distance," in *ICCV*, 2003.

[10] H. Wang, A. Kläser, C. Schmid, and L. Cheng-Lin, "Action Recognition by Dense Trajectories," in *CVPR*, 2011. [Online]. Available: http://hal.inria.fr/inria-00583818

[11] I. Laptev and T. Lindeberg, "Space-time interest points," in *ICCV*, 2003.

[12] I. Laptev, M. Marszaek, C. Schmid, B. Rozenfeld, I. Rennes, I. I. Grenoble, and L. Ljk, "B.: Learning realistic human actions from movies," in *CVPR*, 2008.

[13] M. Chen and A. Hauptmann, "Mosift: Recognizing human actions in surveillance videos," Carnegie Mellon University, Tech. Rep. CMU-CS-09-161, 2009.

[14] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, 2004.

[15] E. Shechtman and M. Irani, "Matching local self-similarities across images and videos," in *CVPR*, 2007.

[16] M. Chen, H. Li, and E. Hauptmann, "Informedia@ trecvid 2009: Analyzing video motions," in *TRECVID*.

[17] X. Sun, M. Y. Chen, and A. Hauptmann, "Action Recognition via Local Descriptors and Holistic Features," in *CVPR*, 2009.

[18] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in *VS-PETS*, 2005.

[19] S. Leutenegger, M. Chli, and R. Siegwart, "BRISK: Binary robust invariant scalable keypoints," in *ICCV*, 2011.

[20] F. Jurie and B. Triggs, "Creating efficient codebooks for visual recognition," in *ICCV*, 2005. [Online]. Available: http://dx.doi.org/10.1109/ICCV.2005.66

[21] C. Schüldt, I. Laptev, and B. Caputo, "Recognizing human actions: A local svm approach," in *ICPR*, 2004.

[22] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: a large video database for human motion recognition," in *ICCV*, 2011.

[23] A. Kovashka and K. Grauman, "Learning a hierarchy of discriminative space-time neighborhood features for human action recognition," in *CVPR*, 2010.