HUMAN ACTIVITY RECOGNITION USING DYNAMIC VISION SENSORS

A Project Report

submitted by

STEFANIE ANNA BABY

in partial fulfilment of the requirements
for the award of the degree of

BACHELOR OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

MAY 2017

THESIS CERTIFICATE

This is to certify that the thesis titled **Action Recognition Using Dynamic Vision Sensor**, submitted by **Stefanie Anna Baby** (Roll Number: EE13B120), to the Indian Institute of Technology, Madras, for the award of the degree of **Bachelor of Technology**, is a bonafide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. 1
Kaushik Mitra
Assistant Professor
Dept. of Electrical Engineering
IIT-Madras, 600 036

Place: Chennai

Date: 5th May 2017

ACKNOWLEDGEMENTS

First and foremost, I sincerely thank Professor Kaushik Mitra for providing me this engaging and compelling project to work on. I am grateful for his enthusiasm and effort to guide and correct me at each step of my work. I also thank Bimal Vinod of the DVS team and all the members of the Computer Vision Lab, Department of Electrical Engineering, IIT Madras for their valuable guidance throughout the course of the project. Finally, I thank my parents, my sister and all my friends at IIT Madras for being a constant source of support and strength throughout my undergraduate curriculum.

ABSTRACT

KEYWORDS: Dynamic Vision Sensor; Human Action recognition; Motion Descriptors; Optical Flow

Dynamic Vision Sensors are novel cameras built to overcome several disadvantages posed by conventional cameras such as low temporal resolution, high memory storage requirements, high power consumption and computational time. These factors are primarily caused due to the fact that there is huge data redundancy on storing information frame by frame. DVS cameras instead capture and output only those pixels in a scene where motion is detected - in the form of a new encoding known as Address Event Representation (AER). Also, the events are recognised at the same time as it is happening, at a temporal scale of microsecond, which is much better than the typical frame rate of conventional cameras. Realising these advantages of DVS in terms of both space and time efficiency, this project explores the possibility of better recognising human actions using DVS when compared to action recognition by conventional cameras. We look at performance of DVS when compared to conventional videos for Human Activity Recognition (HAR) using existing feature descriptors in literature. We also propose a new method for HAR and combine it with an existing descriptor to increase DVS' overall recognition rate.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

IITM	Indian Institute of Technology, Madras
HoG	Histogram of Gradients
HoF	Histogram of Oriented Optical Flow
DVS	Dynamic Vision Sensors
HAR	Human Activity Recognition
BoF	Bag of Features
SVM	Support Vector Machine
KNN	K- Nearest Neighbours
ANN	Artificial Neural Networks
LK	Lucas - Kanade method (for optical flow)
HS	Horn - Schunck method (for optical flow)
STIP	Space time interest points
LOO	Leave One Out
GM	Geometric Mean
AM	Arithmetic Mean

NOTATION

V_x	Horizontal component of optical flow
V_Y	Vertical component of optical flow
I_x	Partial derivative of intensity with respect to x
I_y	Partial derivative of intensity with respect to y
I_t	Partial derivative of intensity with respect to t
Δ	Laplacian operator : $\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$

CHAPTER 1

INTRODUCTION

1.1 Dynamic Vision Sensors

Dynamic Vision Sensors are a recent innovation in machine vision, built to provide several advantages over the typically used conventional cameras. Unlike traditional vision that works on frame based data storage, these sensors mimic how neuromorphic retinal vision happens in humans, using a new encoding scheme that records only transient data at pixel level. Classical vision cameras produce a constant volume of data in the form of frames at a fixed rate, irrespective of the extent of scene activity that is taking place. This mechanism has the inherent disadvantage of data redundancy, as is especially evident from the information stored in consecutive frames. Dynamic vision sensors, as the name suggests are built to sense only the moving parts in a video, thereby removing static and redundant data from the very beginning (Delbruck, 2008).

DVS sensors have several qualities that are worth noting. First of all, in these silicon retina sensors, information is stored in the form of events at pixel level using intensity change as the criterion for firing. This ensures that events are recorded only at the address of pixels that sense relative change in temporal luminance. Other pixels where intensity remained constant do not generate events. This helps hugely in data compression since none of the static scenes are recorded. In traditional CMOS cameras, irrespective of the scene activity, a constant volume of data generated in the form of frames at a fixed rate. But DVS cameras respond only to relative movement to asynchronously and in real time generate a stream of on-off events.

Based on the transient responses in the video, data from DVS is often sparse and directly usable. This is because these cameras perform an intrinsic round of preprocessing during data recording itself. DVS recordings can be considered similar to that obtained from conventional cameras after preprocessing for silhouette extraction by background subtraction and contour detection. So DVS can save huge costs in computational effort

and speed when it comes to applications like object tracking and motion detection. For such scenarios, background data is irrelevant and often an unnecessary distraction.

DVS sensors have an inherent usefulness in its mechanism that it is not the absolute value of illuminance, but the relative change that fires events. The firing of pixels in DVS generates two types of events - On and Off - based on whether the intensity change detected was positive or negative and greater than a certain threshold. This means that lighting effects are inherently taken care of by the sensor, giving a relatively robust encoding. The bandwidth of these cameras are lower as well since no absolute irradiance values are recorded.

Because the pixels operate in parallel, DVS cameras have a very high temporal resolution of one microsecond with very low latency (Cho and Lee, 2015). This would also mean that they capture new information that cannot be obtained by the classical CMOS cameras.The scene is captured using the parallelly operating pixels at a frequency resolution of Megahertz, making these sensors ideal for real time applications like fast robotics and visualising vibrations.

Furthermore, DVS cameras have very efficient power consumption, about ten to hundred times lower than conventional CMOS cameras. This can be effectively used in sensitive applications where power availability is limited, as is the case in sensors used in surveillance, security, mobile devices,remote vision systems etc. It is the sparsity of DVS data and its efficient mechanism for analog to digital conversion, different from the usual AD converters, that gives it such low power requirement.

Finally, since the value of relative intensity can vary immensely within a scene, these silicon retina cameras have a huge dynamic range of illumination at about 120 dB. Contrast this with conventional high speed vision systems, where dynamic range is typically limited to around 50 dB.

1.2 Applications

With the above presented advantages that the bio- inspired DVS data provide, the uses of DVS cameras are many. They can be effectively used in applications like tracking fast moving objects, acquiring and analysing traffic data, steering Unmanned Aerial Vehicle

(UAV) by obstacle detection and avoidance, car collision avoidance system etc. They can also be used for human action recognition which can have crucial use in e-health services. DVS data enables low-cost recognition of activities at real-time capacity with low latency. Also, its low power consumption makes it ideal for use in areas that require sensors to be always on.

1.3 DVS Data Generation and Representation

An event in a DVS camera is generated when the temporal illuminance change at a pixel is greater in magnitude than a preset threshold. For this, the logarithm of change in intensity is measured and is compared with the threshold to decide triggering of events.

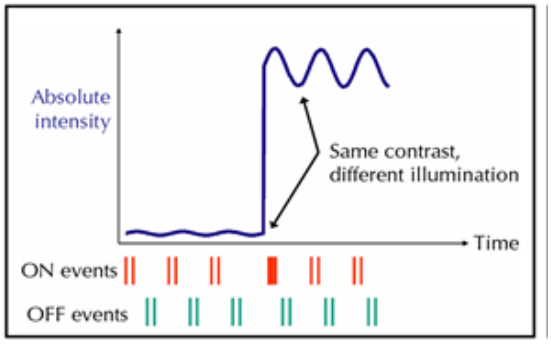

Figure 1.1: Generating Address Events [1]

The representation of an event is done as a four-tuple, explaining the location (x and y pixel address), time stamp (of microsecond resolution), and polarity (positive and negative) of the event. These sensors are also more similar in working compared to biological vision, since it channels positive and negative events as separate signals and outputs them asynchronously.

1.4 Human Action Recognition

Human Action Recognition is an important field in Computer Vision. Several scenarios require recognising and categorising human actions and poses like in gait analysis,

[1]Taken from the homepage of Inilabs (URL :`http://inilabs.com/products/dynamic-vision-sensors/`)

health care, visual human- machine interfaces such as video game systems and robots, to name a few. But traditional cameras pose several hindrances to these critical and in-demand situations. They store too much data that is irrelevant and redundant. Also, the static background that is stored in conventional video recordings make visual identification of the object of interest a computationally intensive task. These additional and often unuseful pieces of information distract the recognition algorithm from the target and need to be removed before helpful features can be extracted. In the light of these requirements, we attempted action recognition using information from DVS with the focus on tapping its inherently relevant features.

CHAPTER 2

PREVIOUS WORK

Steps in Action Recognition

For any type of recognition task, the first critical step is object tracking. This is followed by extraction of useful features after suitable pre processing of raw data. Finally, the action is identified using appropriate classifier(s). Since the output of DVS generate only the moving contours in a video, this data can be used directly for feature extraction and subsequent classification. We will now review some of the existing techniques in literature for feature extracting techniques.

2.1 Feature Extraction

For a typical video on humans, two types of features are classically extracted - descriptors based on motion and those based on shape. Because DVS video contains both shape and motion based information, we need to extract features that reflect both of these in order to predict with good classification accuracy. There are several more descriptors in literature that work on extracting scene, color/ hue and texture based features in a video. But texture and hue information is unavailable in DVS data because of its binary encoding scheme. The scene context based descriptors can also not be used with DVS videos since scenes usually are static in a video, unless there is significant camera motion. Nevertheless, volume based features like motion and shape often provide sufficient information required to perform decent recognition and are more popular than surface features like color and texture .

2.1.1 Motion Based Representations

One common and efficient descriptor for object motion is optical flow. In optical flow methods, the relative motion in a sequence of scenes is captured by describing velocity

of the apparent motion at every voxel position. The consecutive scene frames are assumed to be sufficiently similar, in order to calculate the differential motion by Taylor Series aproximation. We assume the displacement of image is small and constant, such that over a small time Δt the brightness of the displaced pixel remains same. That is, if Δt is the time between consecutive frames and a voxel at location (x, y, t) having intensity $I(x, y, t)$ has moved by $(\Delta x, \Delta y)$, then the brightness constancy assumption gives us:

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t) \tag{2.1}$$

The right hand side of this equation can be aproximated in terms of first order terms in its Taylor Series Expansion.

$$I(x + \Delta x, y + \Delta y, t + \Delta t) = I(x, y, t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t$$

These two equations imply that:

$$\frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t = 0$$

$$\Rightarrow \frac{\partial I}{\partial x}\frac{\Delta x}{\Delta t} + \frac{\partial I}{\partial y}\frac{\Delta y}{\Delta t} + \frac{\partial I}{\partial t} = 0$$

$$\Rightarrow \frac{\partial I}{\partial x}V_x + \frac{\partial I}{\partial y}V_y + \frac{\partial I}{\partial t} = 0$$

where V_x and V_y represent the x and y components of velocity. So we have:

$$I_x V_x + I_y V_y + I_t = 0 \tag{2.2}$$

where I_x , I_y and I_t are the partial derivatives of intensity with respect to x, y and t.

Since there are two unknowns V_x and V_y in this equation, we need an additional constraint to solve for the optical flow. We describe two methods for estimating it - viz. Lucas-Kanade and Horne Schunk algorithm.

Lucas - Kanade Optical Flow

In this method, we assume that displacement of a point p is the same as that of other points within a small neighbourhood around it (Lucas *et al.*, 1981). So we use equation

(2.2) keeping V_x and V_y constant for all points within a window around p. This gives us n simultaneous equations in V_x and V_x, changing the problem from an under-determined system to an overdetermined one. If $q_1, q_2, ...q_n$ are the n pixel points within the window, then we have:

$$I_x(q_1)V_x + I_y(q_1)V_y + I_t(q_1) = 0$$

$$I_x(q_2)V_x + I_y(q_2)V_y + I_t(q_2) = 0$$

$$\vdots$$

$$I_x(q_n)V_x + I_y(q_n)V_y + I_t(q_n) = 0$$

where $I_x(q_i)$, $I_y(q_i)$ and $I_t(q_i)$ are the three partial derivatives of intensity evaluated at point (q_i).

Since this is an overdetermined system of linear equations, we can solve it be least squares principle.
Denote:

$$A = \begin{bmatrix} I_x(q_1) & I_y(q_1) \\ I_x(q_2) & I_y(q_2) \\ \vdots & \vdots \\ I_x(q_n) & I_y(q_n) \end{bmatrix} \qquad v = \begin{bmatrix} V_x \\ V_y \end{bmatrix} \qquad B = \begin{bmatrix} I_t(q_1) \\ I_t(q_2) \\ \vdots \\ I_t(q_n) \end{bmatrix}$$

Thus, we have

$$Av = B$$

So

$$A^T Av = A^T B$$

i.e

$$v = (A^T A)^{-1} A^T B$$

Hence, at each pixel location, the flow is found using Eq.(2.2) on all the pixels within a window centred at the point of interest. For solving the system, we can also have weighted least squares instead of ordinary least squares by using an nXn weight matrix that gives more importance to the pixels that are closer to the central pixel.

Horn - Schunck Method for Optical Flow

This method combines Equation (2.2) with an additional constraint that imposes smoothness of the computed flow. It is a global constraint that assumes optical flow to be smooth over the entire image (Horn and Schunck, 1981).

The value of flow is computed by minimising the objective function:

$$f = \int\int [(I_x V_x + I_y V_y + I_t)^2 + \alpha^2(\|\nabla V_x\|^2 + \|\nabla V_y\|^2)]dxdy$$

Here, the second term within the integral is forced to be small using the smoothness regularisation parameter α. Larger values of α force the flow to be smoother and vice versa. To find the optimal V_x and V_y values, we differentiate the objective function with respect to these two variables. This gives us two equations, namely:

$$(I_x V_x + I_y V_y + I_t)I_x - \alpha^2 \Delta V_x = 0$$

$$(I_x V_x + I_y V_y + I_t)I_y - \alpha^2 \Delta V_y = 0$$

Here, Δ is the Laplacian operator given by:

$$\nabla = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

To implement the Laplacian operate, we approximate it using the formula:

$$\nabla V_x(x, y) = \bar{V_x}(x, y) - V(x, y)$$

where $\bar{V_x}$ is the average velocity in x direction computed at location (x, y) using its neighbourhood. So, the smoothness constraint helps in filling missing flow values using its surrounding pixels. But this makes it less robust and more sensitive to noise (Bruhn *et al.*, 2005). In the DVS recordings that we used (details of which are given in Chapter 4), often contain spurious events were fired in the static background. Hence we noted that performance of Horn Schunck on our DVS datasets was slightly lower than that given by Lucas - Kanade method where only local neighbourhood is considered for flow estimation.

Motion Boundary Histograms

In optical flow method, local motion is captured in terms of absolute value of velocity. This implies that a flow value would be attributed to even static background objects if there is camera motion present in the video. This is the case with the dataset we used as well, where videos were shot in an unconstrained setting with significant camera motion. Since camera motion is often a simple translational movement, one easy method to remove it is by taking the derivative of optical flow in the horizontal and vertical directions (Dalal *et al.*, 2006). This would remove any simple background movement in the data which is often a source of distraction and clutter for the recognition algorithm. What remains is a cleaner encoding that captures only the relative motion, which is then binned based on the flow's orientation to give a new feature known as Motion Boundary Histograms.

2.1.2 Object Shape Representation

A very simple and efficient feature for describing shapes of objects is Histogram Of Gradients (HoG) (Dalal and Triggs, 2005). In this method the gradient of the intensity image is found in the x and y directions after greyscaling the input video if it has color. This gives for each input frame, two feature-sets of the same size as the given frame. Denote the two extracted sets of features as G_x and G_y. G_x and G_y are then used to find the magnitude and angle of gradients at each pixel location of the frame. These two frames of features are then subdivided into dense blocks (overlapping or non overlapping) within which the gradient values are segregated into bins based on their angles. After binning, the histograms are locally normalised within each block or cell to account for the effect of illumination. The locally normalised histograms are then concatenated to create the final feature vector. Performance of HoG descriptors depend on several parameters like block size, number of bins within a block, percentage overlap between blocks, type of normalisation (L1 norm, L2 norm, L1- square root norm) etc.

Other features in literature for describing shape include centroid distance features, Fourier transform Descriptor for closed contours, Cartesian coordinate features etc. We have used only HoG in our implementation owing to its simplicity and classification efficiency.

2.2 Classification

This is the third and final step of the recognition problem. For image classification, commonly used classifiers are K nearest neighbours, support vector machines (SVM), bag of visual words, bag of Features followed by SVM and artificial neural networks. We will now briefly look into each of these methods.

2.2.1 K Nearest Neighbours (k-NN)

K nearest neighbours - as the name suggests is an algorithm that looks at K most related neighbours in the training set for a given test instance of features. KNN classifier can be used for both classification and regression. It outputs a class based on majority voting in the former case while in case of regression, it gives out a value based on ordinary or weighted average of the values of the k nearest neighbours. Typically, the weight used is the inverse of the distance of the neighbour from the test point, so that closer neighbours are assigned greater weights. So KNN is a non- parametric method where only local features decide the output. For measuring the closeness between samples, Euclidean distance is commonly used. Some other closeness measures are Hamming distance (used in finding overlap in text classification) and correlation (which is used in gene analysis).

2.2.2 Support Vector Machines

Support Vector Machines are one of the most common classifiers used for image recognition. It is a supervised algorithm where labelled training data is used to build a separating boundary or hyperplane between classes. In its simplest form, it is a linear, binary classifier. The hyperplane is typically constructed by maximising its margin from each instance of training data. SVMs can also find non -linear classification boundaries by mapping the original features into a higher dimensional space where the two classes are linearly separable. Some common kernels used for this mapping are polynomial, Gaussian and sigmoid. To enable multi-class classification using SVMs, we break the multiple classes into several binary class problems. That is, if there are N classes in our recognition task, we either train N separate SVMs (one-vs-all approach) or N(N-1)/2

SVMs (one-vs-one method) to determine the class of a given test sample.

2.2.3 Bag of Words

Bag of Visual Words (O'Hara and Draper, 2011) is an image classification method in which images from the training data are used to create a code-book to encode local patches in images as codewords. After each image in the train set is abstracted by a set of local patches, k-means algorithm is run on the patches to extract k mutually exclusive clusters. Then for a given image, its histogram is generated as the count of its image patches belonging to each cluster. The histogram is then used as the feature for training a classifier such as SVM.

CHAPTER 3

DATASET USED

Before explaining our proposed method, we describe the dataset we have used for testing. We have taken the UCF11 dataset, also known as UCF YouTube Action Data (Liu *et al.*, 2009) for human activity recognition. This dataset contains eleven action classes , viz. basketball shooting, biking/cycling, diving, golf swinging, horse back riding, soccer juggling, swinging, tennis swinging, trampoline jumping, volleyball spiking, and walking with a dog.

Figure 3.1: YouTube Action Data Set [1]

The Data Set has more than 100 videos in each of the classes, which is further subdivided in to 25 groups. This grouping allows us to perform Leave One (group) Out cross validation twenty five times on the actions. Unless mentioned otherwise, we have performed all our experiments under this setting of 25 LOO cross validations as was suggested by the creators of the data .

We chose the UCF11 dataset for our experiments because it is one of the only few human action data whose benchmark DVS counterpart is publicly available (Hu *et al.*, 2016).

[1]Image source: `http://crcv.ucf.edu/data/UCF_YouTube_Action.php`

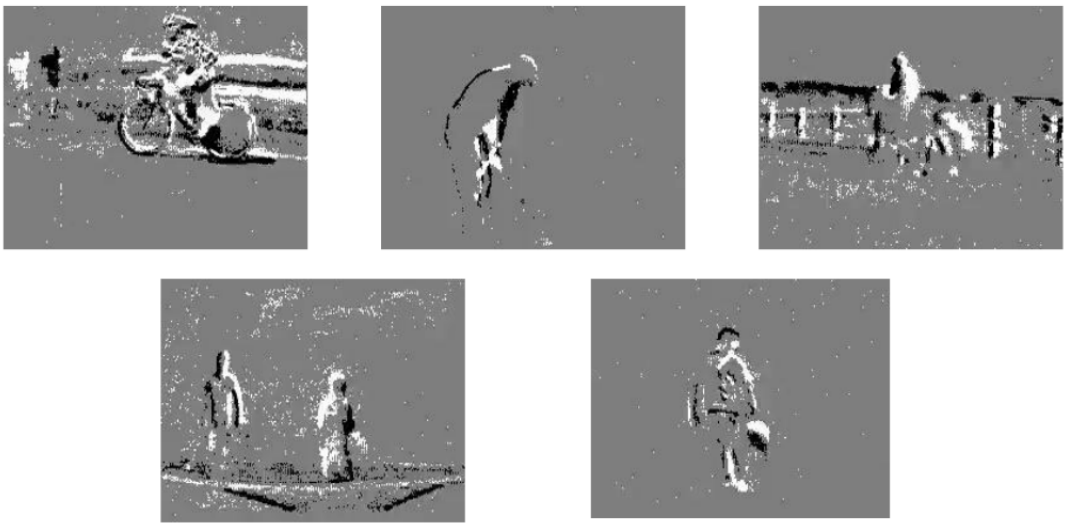

Figure 3.2: Some sample DVS images of the UCF11 data. The actions shown are biking, golf swing, horse riding, trampoline jumping and soccer

The DVS data was created by the authors Hu *et al.* (2016) by re- recording the existing benchmark UCF11 videos played on a monitor using a DAViS240C vision sensor. Since the data was not directly recorded from the wild, this would mean that time resolution greater than that provided by the UCF11 video is not available in DVS under this simulated setting. Nonetheless, we proceeded to use this dataset since our focus is not to tap the temporal resolution that DVS offers, but to analyse its performance on HAR using its special spatial encoding.

CHAPTER 4

PROPOSED METHOD

Averaged Motion Maps

In this section, we propose a simple method that that focuses to capture the both movement and shape in video using three averaged slices of frames. The three slices represent the activities averaged along the three dimensions of the video, to give us xy, xt and ty slices.

Our method is a simple encoding that relies on the fact that DVS has already captured mostly the moving foreground object which is our subject of interest. So time averaging the frames along the length of the video gives the mean pose and stance of the object as the xy slice. Similarly, to obtain the xt slice, the *n*th column of xt matrix is filled by the row sum of the *n*th frame in the video. Likewise, the ty slice is created by filling its *k*th row with the column sum of the *k*th frame in the video. These two slices record the manner the object had moved over the video's duration.

From the three motion maps, we extract Bag of Features where the feature points taken are speeded up robust features (SURF) extracted through grid search on the slices. SURF is similar to SIFT (scale invariant feature transform) detector in its overall approach, but differs in its implementation technique which aims at speeding the feature detection. This is followed by k-means clustering of the train data's features to create a visual vocabulary of k words. Then the features from each video are pooled or binned into the k clusters and l_2 noramlised. Typically SVM classifier is then used on the encoded features to predict the activity that was performed. Since the features in the three slices complement one another, with the xy slice encoding the object's shape and pose while the xt and yt slices describing its motion, a fusion of the slices either before or after BOF- gives the best performance on HAR.

Figures 4.2, 4.3 and 4.5 show the motion maps for 11 randomly picked videos from the 11 classes.

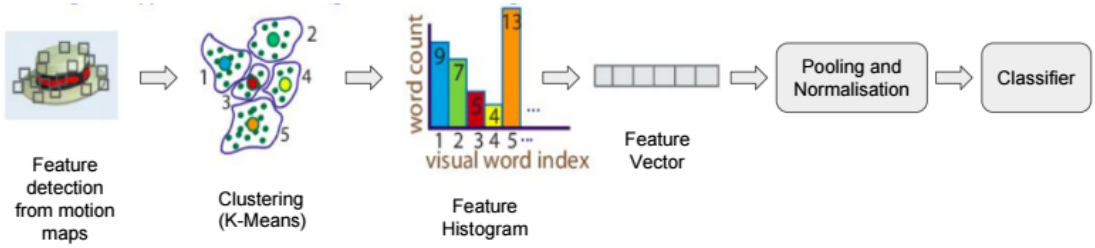

Figure 4.1: The proposed method: Averaged Motion Maps followed by Bag of Visual Words

Figure 4.2: Motion map 1 : Time-averaged xy slices

In the xy slice, we note that much of the shape and pose of the object is captured. Similarly xt and ty slices show rhythmic patterns typical for a given action category. Notable ones among these are winding river-like ty slice for *swinging* action in figure 4.3 and the rhythmic up and down spikes in the xt slice for *trampoline* class in figure 4.5.

Since the number of frames vary per video, bi-linear interpolation was done on the final xt and ty slices to reshape them to a fixed size from all the training videos. These slices represent the three motion maps of the video.

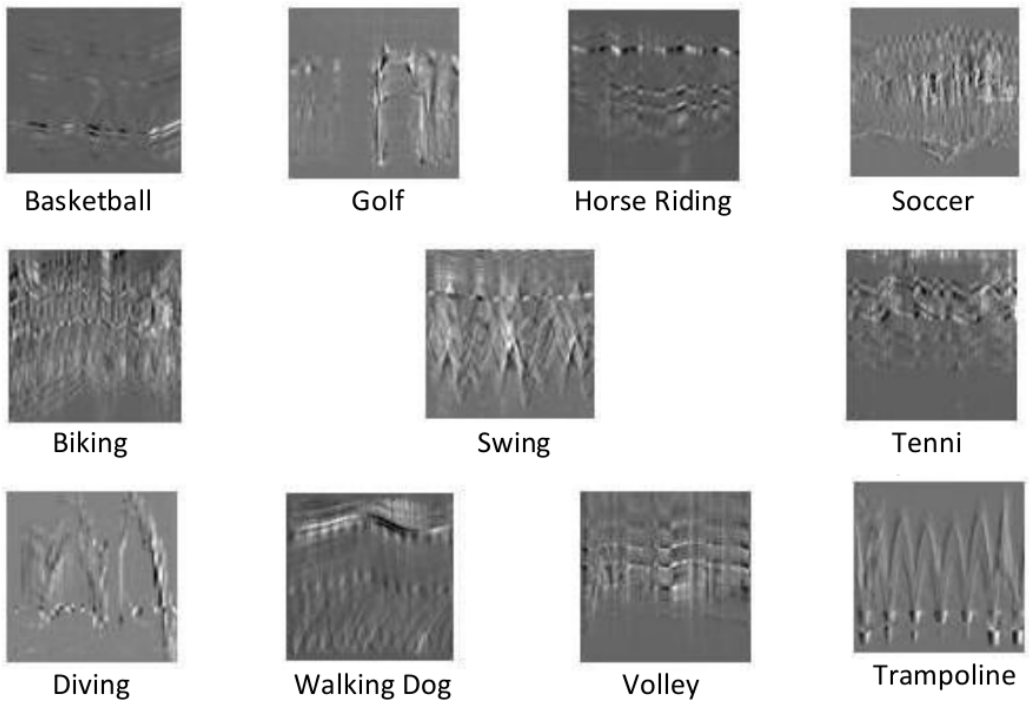

Figure 4.3: Motion map 2 : y-averaged xt slices

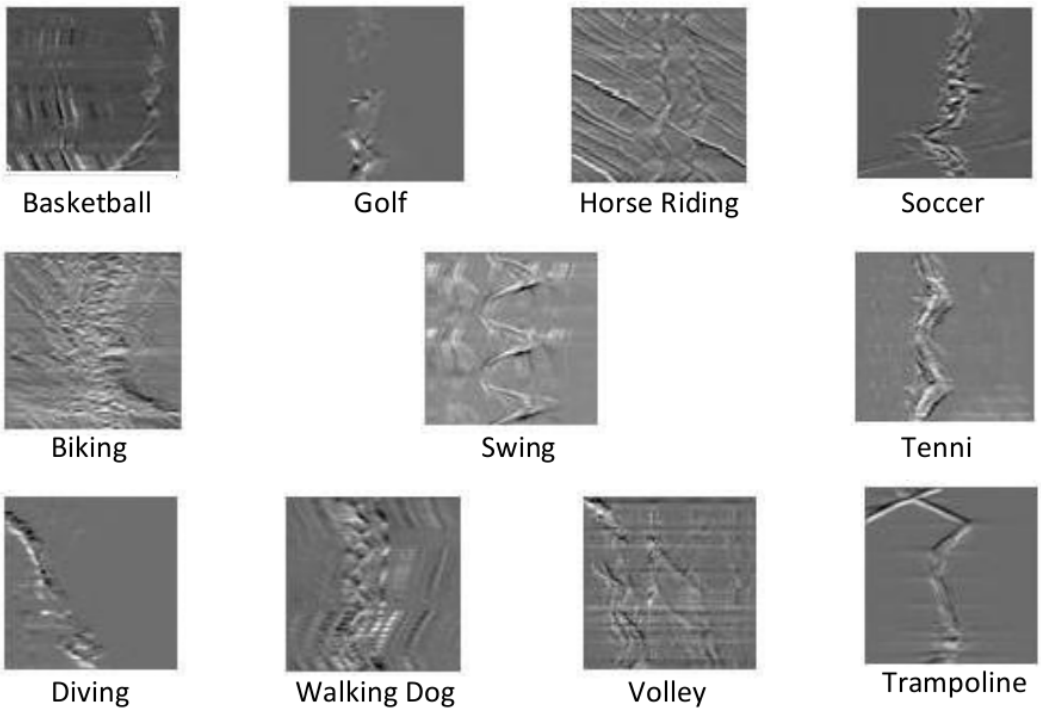

Figure 4.4: Motion map 3 : x-averaged ty slices

In the following section, we note the performance of HAR using individual motion maps as well as that using existing motion descriptors like HoG and HoF. Then we look at combining the three motion maps effectively using fusion at three different levels and improve its HAR further. We will also look at improving the recognition rates of existing descriptors. Finally we combine the best existing feature descriptor for HAR with our descriptor and perform activity recognition in DVS as well as conventional videos. We assess the loss in HAR on using DVS data compared to conventional videos and conclude whether activity recognition is indeed possible in practice with the sparse encoding of DVS.

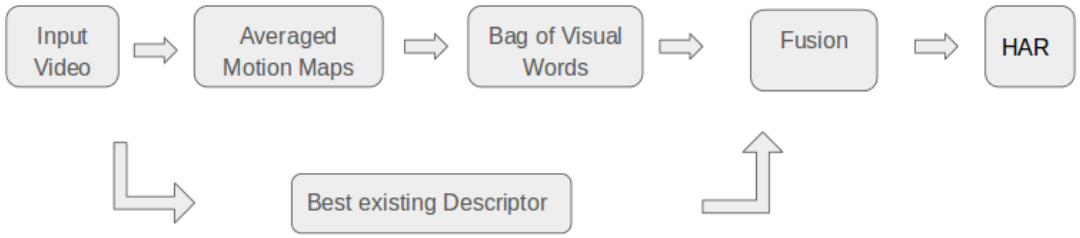

Figure 4.5: Outline of the proposed method

CHAPTER 5

EVALUATION

The UCF YouTube Action Data is in general considered to be extremely challenging due to the large amount of variation it provides in appearance and pose of the object along with several other factors like background clutter, camera motion, scale of object etc. The methods we used each tackle one more of these issues. DVS data by itself helps remove most of the background clutter and other distractions present in the actual scene. Scale of the object is taken care when we extract the SURF features from averaged motion maps. Optical flow also helps to focus on the foreground object is we assume most of the movement that happens is due to our object of interest. Histogram of Gradient captures the pose and overall stance of the subject while Motion Boundary Histograms remove linear camera motion by taking gradient of optical flow in the horizontal and vertical directions.

For HoG, we used three frames uniformly from the length of the video as our key frames for feature extraction. The initial 4 frames were excluded before picking key frames, since not much activity happens in the very beginning of the video

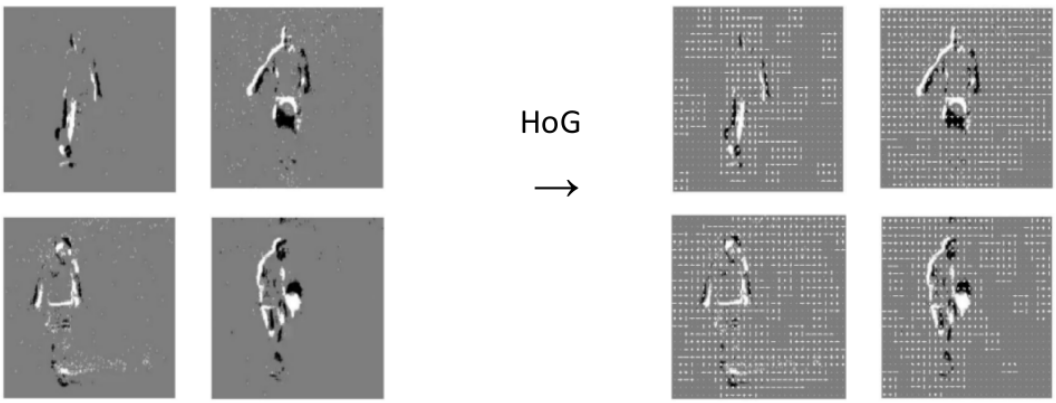

Figure 5.1: Visualisation of Histogram of Gradients on class- Soccer

Similarly, for finding optical flow, we used 3 uniformly chosen frames from the videos and the flow was computed between the chosen frames and their immediately preceding ones. For UCF11 data, we observed that Horn-Schunck method gave two

percent lower recognition rate than Lucas Kanade algorithm. From here onwards, all further flow calculations were made using LK method.

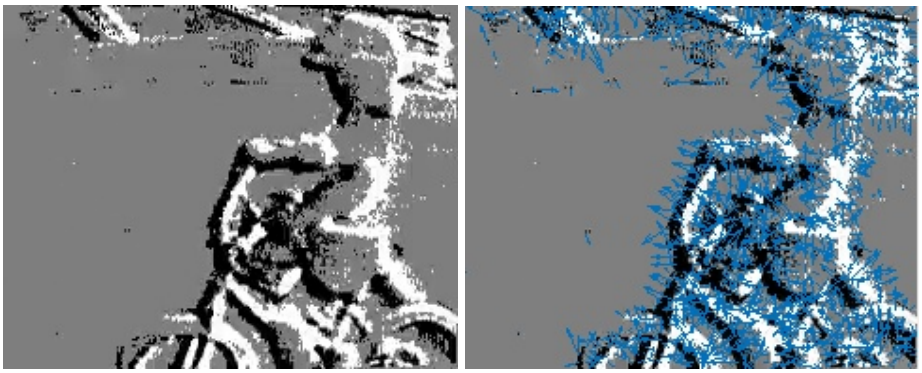

Figure 5.2: Optical flow shown on a biking video from DVS

Finally, for our proposed method, all the frames were used from the videos and the three averaged motion maps were obtained.

5.1 Results

In this section, we present the recognition rates under Leave One Out Cross validation method performed 25 times on the above three feature descriptors.

Table 5.1: Results of HoG, HoF and Averaged Motion Maps

Video type	HoG	HoF	xy slice	xt slice	ty slice
Original UCF11	0.5327	0.5119	0.4573	0.4253	0.3518
DVS data	0.4378	0.3926	0.4943	0.4680	0.4698

Here, bag of features was performed on the 3 slices with a codebook of dimension 500 while the training was done on 100,000 random train features. This was followed by training one vs one SVM for the multi- class classification. On HoG and HoF, we trained SVM directly after flattening the feature values into a 1D array from the three frames. KNN classifier was also used for the final predictions, but it gave consistently about 5 percent lower HAR rates. The results show that DVS gives a better recognition rate on using our proposed method, while HoG and HoF perform better on the original

data. For making the xy map, we noted that taking sum of absolute difference between consecutive frames gave a better outline of the object instead of simple averaging. So that is what we have used here. The reason our method worked worse on the actual UCF11 data is due the fact that it has background clutter and scene information for distracting bag of features encoding.

5.2 Combining Averaged Motion Maps

Next we looked at combining the features from the three slices appropriately to further boost HAR. For this we used the three fusion methods that are described in the paper by Peng *et al.* (2016). It says that fusion of features is done typically at three different levels. The first one is at descriptor level, where the desciptors from different algorithms are concatenated as a single feature before Bag of features is performed. The second type of fusion is representation level fusion, where BoF is trained separately on each descriptor and then the encoded representation is used for training the final classifier. So this fusion happens at the video level, where global representation of each description is first individually learnt. The final fusion method is score level fusion, where the BoF framework and also the classifier is trained separately on each descriptor. The predictions of the classifiers are then combined based on their scores.

The classifiers' scores are usually the posterior probability of a given test video to belong to a particular action class. We use three types of score merging methods for our predictions, which are predicting the class with highest on arithmetic mean of scores, highest geometric mean and class with maximum individual score among the classifiers. Reddy and Shah (2013) also suggests similar fusion algorithm and termed fusion of features before the before the final classifier as early fusion. So descriptor and representation level fusion are early fusion techniques while score level fusion that combines results of different classifier is late fusion. We used multi-class linear SVM with one- vs- one coding scheme as the classifier for all three slice descriptors. The classification score is computed based on the inverse of distance of the test sample from the separating hyperplane. The results of the these fusion methods on our three averaged

[1]Image courtesy- Peng *et al.* (2016)

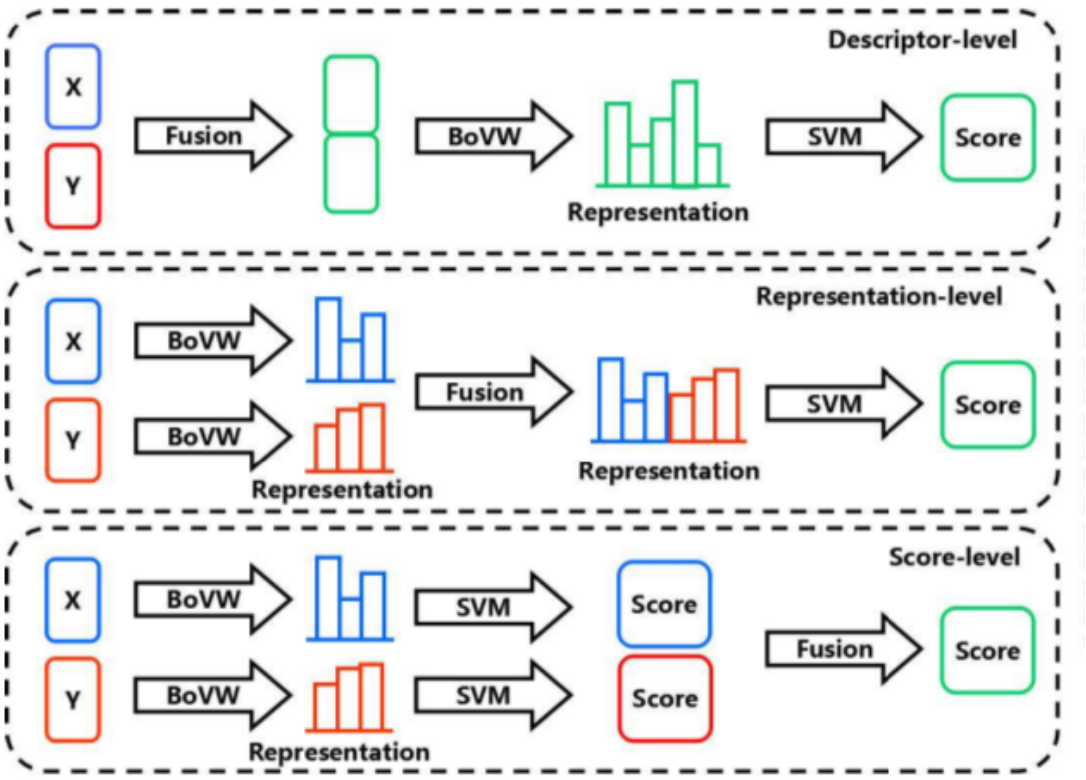

Figure 5.3: Fusion of features at three different levels[1]

motion maps are listed below (tables 5.2 and 5.3). The fusion method that works best ofcourse depends on the dataset used.

Table 5.2: Results of different fusion methods on the 3 averaged motion maps with bag of features of vocabulary size 500

Video type	Descriptor fusion	Representation fusion	Score (GM)	Score (AM)	Score (maximum)
Original UCF11	0.4447	**0.5722**	0.5220	0.5069	0.4491
DVS data	0.5459	**0.6564**	0.6093	0.6030	0.5471

Here we see that representation level learning is what gave the best performance on HAR. This is followed by late fusion at score level with geometric mean performing best, followed by arithmetic mean at a slightly lower HAR rate and finally maximum score of classifier. The simple descriptor level fusion gave the least performance although its results are comparable with score level fusion using maximum classifier score. Also we note that simply training a BoF model with a larger vocabulary size had increased performance by two to three percent in each fusion method.

Table 5.3: Results of different fusion methods on the 3 averaged motion maps with bag of features of vocabulary size 4000

Video type	Descriptor fusion	Representation fusion	Score (GM)	Score (AM)	Score (maximum)
Original UCF11	0.4793	**0.5716**	0.5509	0.5364	0.4950
DVS data	0.5992	**0.6790**	0.6501	0.6426	0.6055

CHAPTER 6

IMPROVING HAR RATE - FURTHER ANALYSIS

As the next step, we look at improving the performance of average motion maps as well as existing motion descriptors using further detection techniques. Then we combine the best result of the proposed method with most discriminating feature in existing literature to get an overall better HAR rate.

Our use of HoG and HoF descriptors in the previous section was not optimal since we did not make use of all the frames in the video. A simple method of improving their performance is by using key frames instead of uniform ones. However, the framework that has most dominated research in action recognition is Bag of Visual Words on local features. For finding local features, the most successful extractors are Space Time Interest Points (STIPs) and Dense Trajectories. In STIP method, motion descriptors such as HoG, HoF and MBH are taken from a cuboid around the extracted points. This is followed k-means clustering to generate the code-book, feature encoding, pooling and normalisation and finally classification. Space Time Interest Points itself could be found using several methods such as Dollar detector (Dollár *et al.*, 2005), Laptev detector (Laptev, 2005), selective STIPs (Chakraborty *et al.*, 2012) etc. Dense trajectories on the other hand, tracks interest points across frames to give a path of interest. So in this method, the descriptors are extracted from a volume around the detected path. The volume is hence no longer a cuboid, but a winding tube along time.

In the following sections, we report the performance of using STIP with average motion maps as well as that of using dense trajectory on HoF, HoG and MBH. Finally we combine the best features given by our proposed method with the best existing feature descriptor to give an overall improved recognition rate.

6.1 Combining Averaged Motion Maps with STIPs

We used selective STIP as our detector on UCF11 data, where STIP points found by basic Harris Detectors are further selected to avoid redundancy in the local descriptors

by suppressing unnecessary points obtained from the background of the scene, imposing local and temporal constraints and adapting to scale. Selective STIPa are shown to have better recognition rates compared to other STIP detectors like Dollar and Laptev.

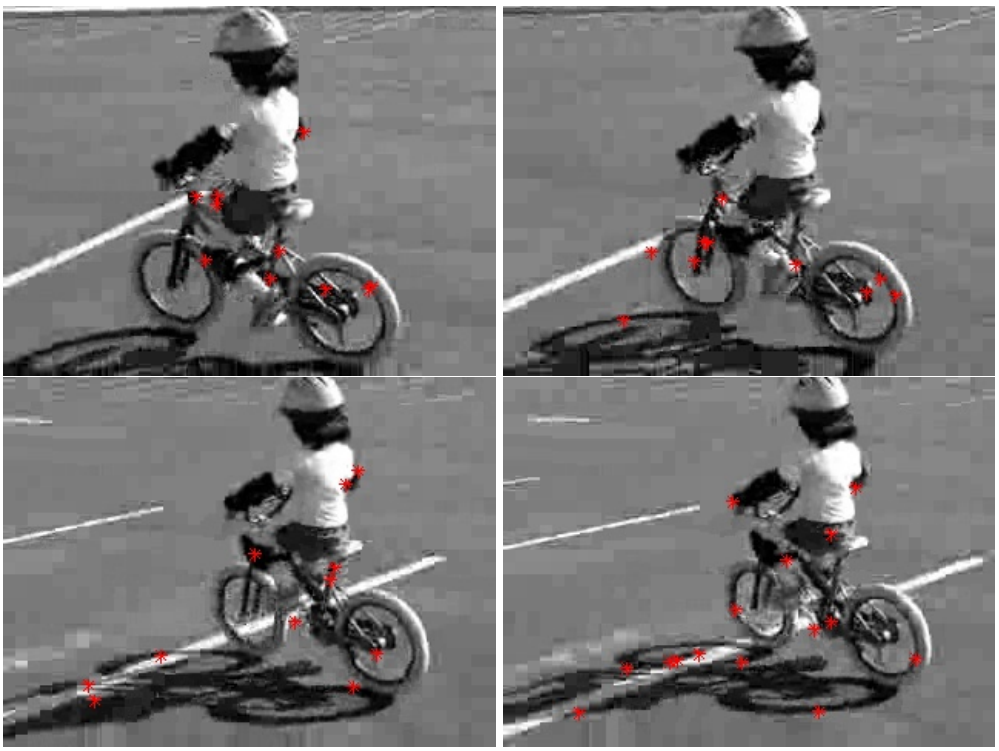

Figure 6.1: Illustration of selective STIPs on a biking video

For each STIP point , the surrounding cuboid of size 13 X 13 X 13 pixels was taken. Bag of features was trained with a 500 dimension codebook and number of nearest neighbours used for clustering was kept at one. For building the vocabulary we used 100,000 random cuboids from the train set and leave one out cross validation was performed 25 times. The results of the same are given in tables 6.1 and A.1.

Table 6.1: Results of using average motion maps with selective STIPs

Video type	xy slice	xt slice	ty slice
Original UCF11	0.4026	0.3995	0.3719
DVS data	0.3825	0.3832	0.3907

Here we again note the same order of performance on using different fusion methods with selective STIP. Descriptor level fusion was not performed this time based on the previous trend. Since STIP did not improve performance any more than the initial

Table 6.2: Results of fusion methods on the 3 averaged motion maps with selective STIP

Video type	Representation fusion	Score (GM)	Score (AM)	Score (maximum)
Original UCF11	**0.5161**	0.4472	0.4454	0.4391
DVS data	**0.4413**	0.4290	0.4271	0.4133

framework, we will use the previous feature vector without cuboids for average motion maps for the final combination.

6.2 Using Motion Descriptors with Dense Trajectories

Dense trajectories are created by dense sampling of image which is shown to have better performance when compared to sparse interest points (Wang *et al.*, 2011). In contrast to interest points, dense tracking provides trajectories that trace the point of interest over time.

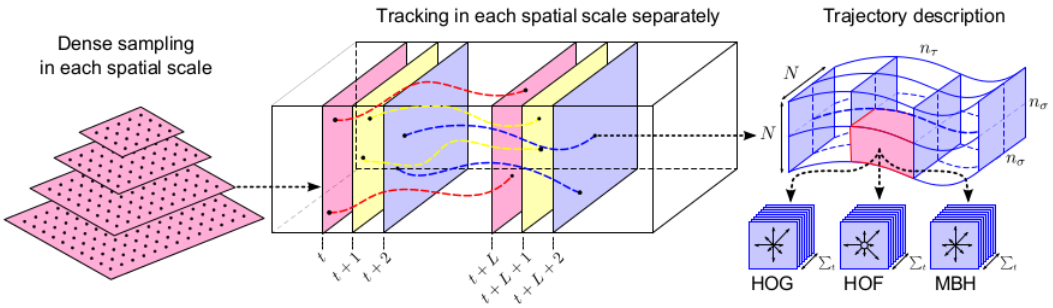

Figure 6.2: Illustration of tracking dense trajectory followed by feature extraction. (Wang *et al.*, 2013)

Along the the path tracked, descriptors like HoG , HoF or MBH are computed using an NXN neighbourhood of pixels. The points are tracked from one frame to another using dense optical flow field and the trajectory length is limited to 15 frames to avoid drifting of tracked points.

In our experiments, we took the tracking volume to be 32 X 32 pixels X 15 frames. This volume is further divided into cells of size 16 X 16 pixels X 5 frames. So each tracked tube gives 2X2X3 cells. Within each cell, the histograms of descriptors are

[1]Image source : Wang *et al.* (2013)

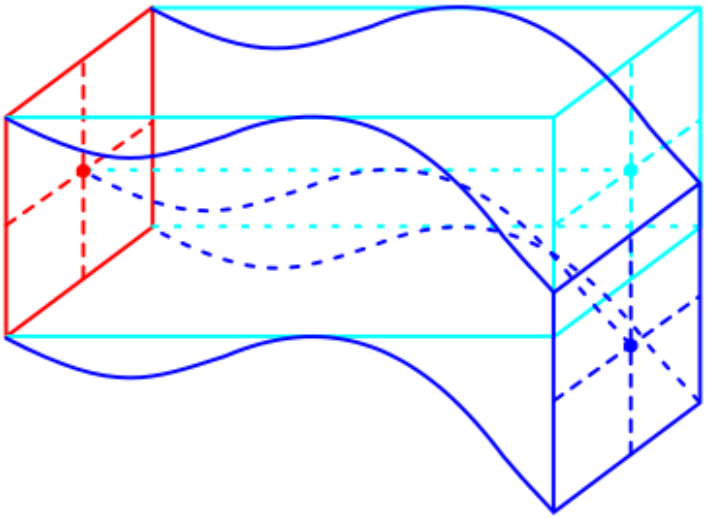

Figure 6.3: Illustration of difference in volume used for feature extraction by dense trajectory (dark blue) and interest point (light blue) [1]

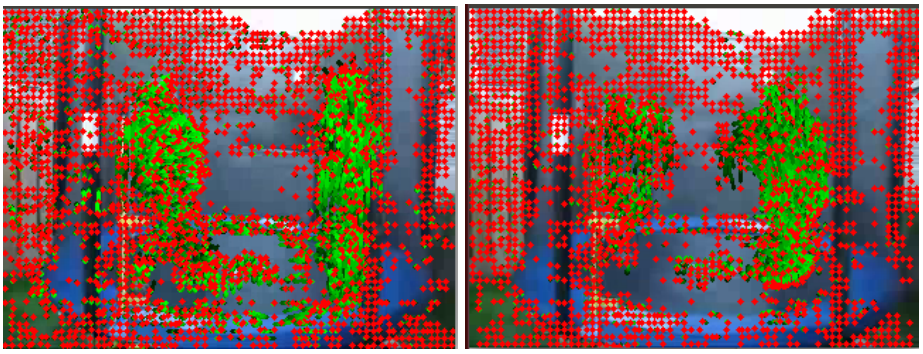

Figure 6.4: Dense tracking of 2 people on a trampoline. The red dots are the densely tracked points while the green arrows show their optical flow or trajectory

found. For HoG and MBH, we used 8 orientation bins per cell and the magnitude of the feature values was used for weighting. For HoF, an additional bin was added to account for pixels whose flow value was smaller than a threshold. All the descriptors were also l_2 normalised before performing bag of features. So in total, HoG gave feature descriptors of size 96 per tracked volume (2X2X3cells per tracked path times 8 bins) while HoF produced 108 features (2X2X3cells times 9 bins). MBH also gave 96 features similar to HoG, but in both horizontal and vertical directions. Thus, overall it had twice the number of features for a chosen trajectory. Bag of features was individually performed on each descriptor. Since each video produced about 500000 dense tracks, most of them in close proximity to one another, BoF was done on a subset of training features on one lakh trajectories randomly selected. The codebook dimension was kept at 500. After

learning the cluster centers, all features of the video are used to generate the histograms of same 500 bins. Finally the pooled features are l_2 normalised and SVM was classifier is trained. The results for 25 cross validations under Leave One Out method for the UCF11 dataset are given below:

Table 6.3: Results of HoG, HoF and MBH with Dense Trajectories using BoF of size 500

Video type	HoG	HoF	MBH
Original UCF11	0.6375	0.5528	0.6866
DVS data	0.5126	0.5810	0.6212

So we see that dense trajectories indeed outperform the same descriptors when used with uniform frames. Also, we note that motion boundary histograms give the best performance out of the three descriptors on HAR. This is a result known for HAR in conventional videos, and the same inference is seen here for DVS data as well. To improve performance further, we trained bag of features for codebook of vocabulary size 4000. Again Leave One Out cross validation was performed, this time on a single fold, on the UCF11 dataset. The results are enlisted in table 6.4

Table 6.4: Results of HoG, HoF and MBH with Dense Trajectories with BoF of size 4000

Video type	HoG	HoF	MBH
Original UCF11	0.6935	0.6452	0.7742
DVS data	0.6290	0.7097	0.6613

The HAR rates have improved by 5 to 10 percent in each case, due to the increased feature size. It is interesting to note that optical flow gave better recognition rate with DVS data wth dense trajectories when compared to actual UCF11 videos, while MBH which is based on the derivative of optical flow performed better with the original videos. This can be explained based on the fact that there is less clutter in DVS data aiding in flow calculations, but the irrelevant flow values that was computed for the background image in the original videos were mostly uniform and got removed on computing MBH. Since MBH consistently outperformed the other two motion descritpors, we use this feature from now on to combine with our proposed descriptor.

6.3 HAR on DVS at different frame rates

In an attempt to boost the performance of DVS even with existing motion descriptors, we created frame based videos out of DVS' events stream at different frame rates. Then the performance of MBH with dense trajectories was recorded on 25 fold leave one out cross validation. Three different frame rates were used for this setup, viz. 30, 20 and 10 fps. For lower frame rates, we observed that the transition from one frame to next was too abrupt for dense trajectories to track any point. Although in theory, higher frame rate videos can also be created out of DVS events, we did not attempt this since the DVS data we used was created by shooting its original videos played at 30 fps. Thus there is no further information that can bw captured by making videos of greater frame rate. This is ofcourse not a limitation of DVS, but of the dataset we had and could have been a study in itself if the data was directly created using a DVS camera from the wild. Nonetheless, we report the results of MBH using dense tracking on the 3 frame rates mentioned. BOF framework was used on the MBH features for a codebook size of 500 words. Table

30 fps	20 fps	10 fps
0.6212	0.6759	0.6564

So, we can gained a 5 percent increase in HAR rate by simply changing the frame rate of DVS videos. As a final setup, we combine this MBH descriptor with dense trajectory at 20 fps with our proposed average motion map descriptor under representation level fusion. Since MBH with dense tracking tracks a large number of points per video (around 60,000 on average), its bag of features was trained for only 500 dimension codebook. For each of the 3 averaged motion maps, BoFs of size 4000 was built.

6.4 Final Fusion - Representation level learning of MBH with Averaged Motion Maps

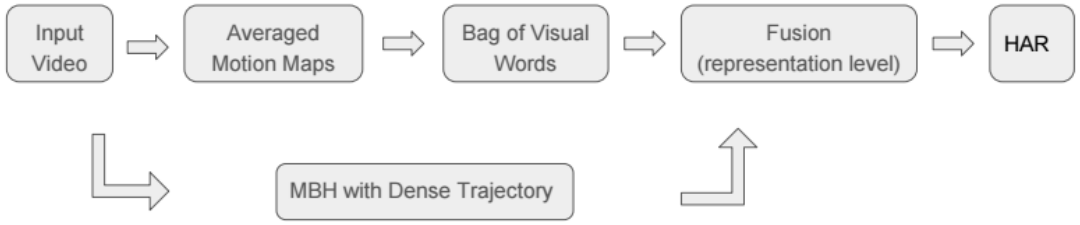

Figure 6.5: Illustration of Final Fusion

The table below shows our final results. For a more elaborate analysis on this combining, please refer the appendix.

Video type	HAR rate
Original UCF11	0.7299
DVS	0.7387

Table 6.5: Final Fusion : MBH with Dense trajectory combined with Averaged Motion Maps under Representation level learning

We conclude this section by noting that DVS had performed just as well as actual videos in HAR based on the feature descriptors that are available for its encoding. Further features based on scene, texture and hue have enabled better recognition rates with actual videos. But these are more complex and unavailable for use from the very beginning with DVS data. Hence if we respect the limitations that come with DVS, we note that within the framework of its possible descriptors it is just as useful for HAR as conventional videos and could efficiently used in place of the latter.

CHAPTER 7

CONCLUSION AND FUTURE WORKS

In this project, we analysed the performance of DVS data in Human Activity Recognition (HAR) compared to its conventional frame-based counterpart using traditional feature extraction techniques. We also proposed a new encoding technique that is suited especially for DVS data in light of its sparse and concise recording scheme. Finally we evaluated how DVS data's performance gets affected on using different frame rates for segregating events. Our results have showed how DVS for HAR is a promising application.

However looking beyond the results, there were some fundamental questions that we left unaddressed in our approach. In every recognition algorithm we tried, we have used bag of features as the final step before classifying the actions as is the state of the art norm today. But the question still remains whether BOF really is the best method to be used, for pooling features or even otherwise. Although our algorithm performed better with BOF followed by SVM approach as opposed to directly training SVM, there are some inherent limitations to BOF that are worth noting. Location based relations are not preserved in BOF, since pooling destroys all spatial information between the extracted features in the image. Valuable relations - such as a person standing on top of the diving board - that would directly allow us to classify the action as *diving* is lost on using BOF. This can be further worked upon, using methods like spatial correlogram and matching. Also, some of the features extracted by BOF from the slices have no semantic meaning to it. They cannot be inherently explained by a cursory examination by humans, nor could we attach them to any particular human activity. If BOF had learnt features that are in actuality irrelevant -then this is something that needs to be looked into for further improving its performance. Interestingly, DVS has the advantage with BOF that its image itself has mainly the moving, foreground object. Hence there are lesser incorrect features that BOF could learn. Nevertheless, improving BOF in general is a challenge that still remains unsolved.

Also, we noted that similar to recognition rates shown by conventional videos, dense trajectories gave the best results on using traditional features in DVS as well. Much of

the success of dense tracking comes from the fact that it generates too many interest points given any video sample. A visualisation of the interest points found by dense sampling showed that some of these are randomly fired noisy events in DVS unrelated to the object in foreground. A simple median filtering pre-processing before finding dense interest points however did not improve recognition rate. Decreasing the frame rate to an extent helped us bypass this issue. However, in order to truly address the problem, perhaps a new method specifically for finding and tracking DVS interest points should itself be invented. This would act as the initial step for improving the performance of HAR on using optical flow, MBH as well as Dense trajectories with dynamic vision sensors.

APPENDIX A

FINAL FUSION METHOD ELABORATED

We have put the final fusion results for only representation level combining in the main section. This was based on the previous analysis where representation level learning consistently outperformed all other fusion methods. Although lower than representation fusion, here we list the HAR rate for the remaining fusion methods as well.

Table A.1: Early and late fusion of MBH with averaged motion maps

Video type	Representation fusion	Score fusion (GM)	Score fusion (AM)	Score fusion (maximum)
Original UCF11	**0.7299**	0.7167	0.6847	0.6859
DVS data	**0.7387**	0.7299	0.7280	0.7041

So indeed representation level fusion best discriminates UCF11 dataset. Then the expected trend of score level fusion follows with Geometric Mean performing better than Arithmetic Mean.

The confusion matrices for representation fusion in both UCF11 and its DVS version are shown in figure A.1. We note how the accuracy figures for each class are very similar in both cases, once again reinforcing our idea that DVS can be used successfully to replace conventional videos for HAR.

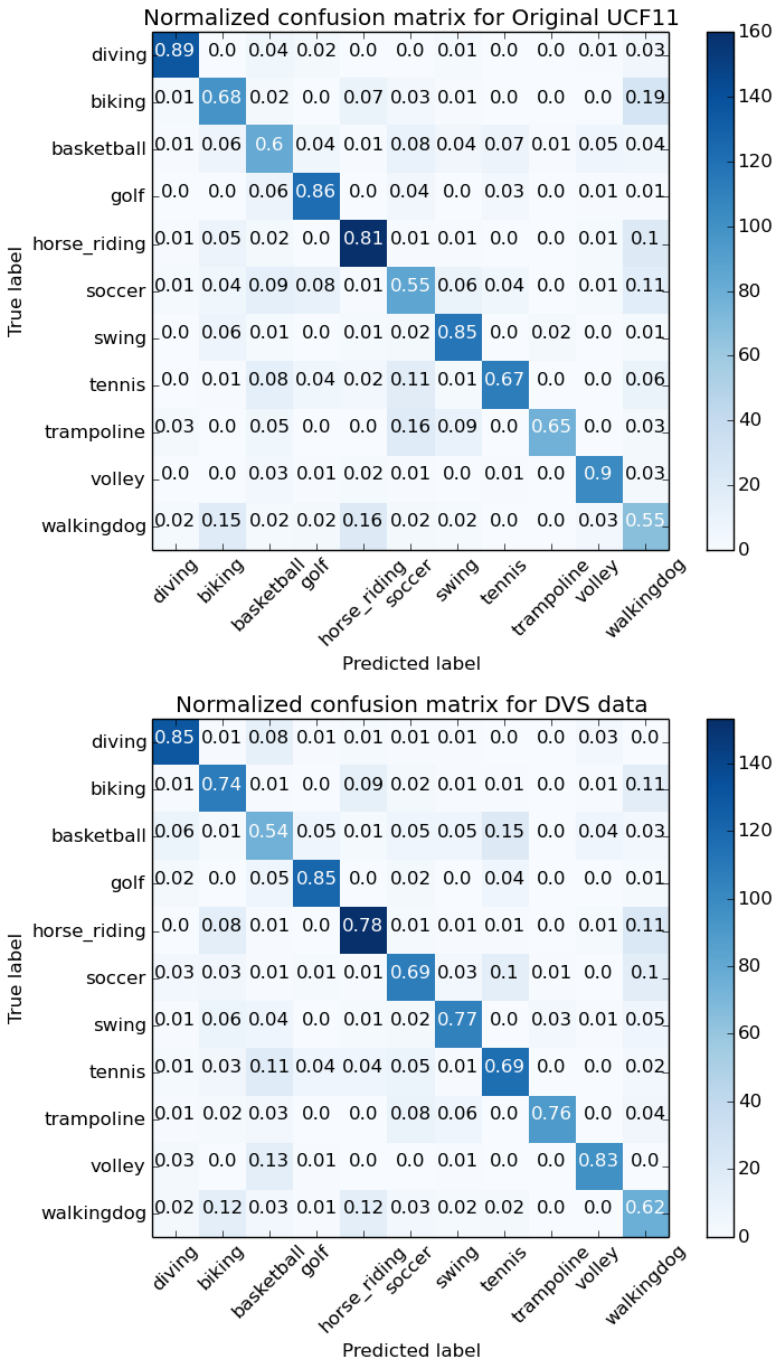

Figure A.1: Confusion matrix for representation fusion in original UCF11 and its DVS counterpart

REFERENCES

1. **Bruhn, A.**, **J. Weickert**, and **C. Schnörr** (2005). Lucas/kanade meets horn/schunck: Combining local and global optic flow methods. *International journal of computer vision*, **61**(3), 211–231.

2. **Chakraborty, B.**, **M. B. Holte**, **T. B. Moeslund**, and **J. Gonzàlez** (2012). Selective spatio-temporal interest points. *Computer Vision and Image Understanding*, **116**(3), 396–410.

3. **Cho, D.-i. D.** and **T.-j. Lee** (2015). A review of bioinspired vision sensors and their applications. *Sensors and Materials*, **27**(6), 447–463.

4. **Dalal, N.** and **B. Triggs**, Histograms of oriented gradients for human detection. *In Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1. IEEE, 2005.

5. **Dalal, N.**, **B. Triggs**, and **C. Schmid**, Human detection using oriented histograms of flow and appearance. *In European conference on computer vision*. Springer, 2006.

6. **Delbruck, T.**, Frame-free dynamic digital vision. *In Proceedings of Intl. Symp. on Secure-Life Electronics, Advanced Electronics for Quality Life and Society*. 2008.

7. **Dollár, P.**, **V. Rabaud**, **G. Cottrell**, and **S. Belongie**, Behavior recognition via sparse spatio-temporal features. *In Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on*. IEEE, 2005.

8. **Horn, B. K.** and **B. G. Schunck** (1981). Determining optical flow. *Artificial intelligence*, **17**(1-3), 185–203.

9. **Hu, Y.**, **H. Liu**, **M. Pfeiffer**, and **T. Delbruck** (2016). Dvs benchmark datasets for object tracking, action recognition, and object recognition. *Frontiers in Neuroscience*, **10**.

10. **Laptev, I.** (2005). On space-time interest points. *International journal of computer vision*, **64**(2-3), 107–123.

11. **Liu, J.**, **J. Luo**, and **M. Shah**, Recognizing realistic actions from videos âĂIJin the wildâĂİ. *In Computer vision and pattern recognition, 2009. CVPR 2009. IEEE conference on*. IEEE, 2009.

12. **Lucas, B. D.**, **T. Kanade**, *et al.* (1981). An iterative image registration technique with an application to stereo vision.

13. **O'Hara, S.** and **B. A. Draper** (2011). Introduction to the bag of features paradigm for image classification and retrieval. *arXiv preprint arXiv:1101.3354*.

14. **Peng, X.**, **L. Wang**, **X. Wang**, and **Y. Qiao** (2016). Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice. *Computer Vision and Image Understanding*, **150**, 109–125.

15. **Reddy, K. K.** and **M. Shah** (2013). Recognizing 50 human action categories of web videos. *Machine Vision and Applications*, **24**(5), 971–981.

16. **Wang, H.**, **A. Kläser**, **C. Schmid**, and **C.-L. Liu**, Action Recognition by Dense Trajectories. *In IEEE Conference on Computer Vision & Pattern Recognition*. Colorado Springs, United States, 2011. URL `http://hal.inria.fr/inria-00583818/en`.

17. **Wang, H.**, **A. Kläser**, **C. Schmid**, and **C.-L. Liu** (2013). Dense trajectories and motion boundary descriptors for action recognition. *International journal of computer vision*, **103**(1), 60–79.