# 3D Feature Descriptors for Human Activity Recognition

Akhil C. Acharya, Kaveen H. Bandara, William T. Fowler, Siyu Huang, Xiaohang Miao, Daniel W. Rice
Computer Science Department, North Carolina State University
Raleigh, North Carolina, USA
Email: acachar2, kmhearth, wtfowler, shuang7, xmiao2, dwrice@ncsu.edu

**Abstract—**The time-series nature of video poses a significant challenge to classical classification algorithms. As such, in order to classify videos, one must develop feature descriptors in order to extract critical features that are spatially and temporally invariant. We created a feature descriptor by implementing Histogram of Oriented Optical Flow (HOOF), which combines Optical Flow with Histogram of Oriented Gradients (HOG) which is a feature descriptor for images that captures spatial invariance. In addition, we utilized pyramiding, a method of detecting temporal invariance in our approach. In this study, we first implemented our own versions of HOG, pyramiding, and Optical Flow to understand the mechanics behind these three techniques. We then studied the effects of combining these techniques upon the accuracy of different classifiers, such as SVM, Decision Tree, Logistic Regression, and Random Forest to determine which classifier worked best with the data. We ran these classifiers on the feature descriptors produced from two different data sets, in a five class problem. Our best result of 78% accuracy came from using our Optical Flow with pyramiding feature descriptor and a Logistic Regression classifier.

**Keywords: Human Activity Recognition; 3D Feature Descriptors; HOOF**

## I. INTRODUCTION

Human Activity Recognition is a subfield of machine learning and computer vision that involves determining which activity a person performs in a series of consecutive images, and the results can be used to understand the intent and motive of the activity. This is one of the most important problems in the computer vision domain, because many applications depend on recognizing human activities in videos, including visual surveillance, human-computer interaction, content-based video retrieval, and sports analysis [1].

However, accurately recognizing human activities in videos is a challenging problem, because human activities are complex and highly diverse. The same action performed by the same person could look very different depending on multiple variables, such as illumination, color of clothes, background, angle and distance of filming, etc. Due to varying physical conditions or customs, the same action performed by two different people may appear different as well.

To cope with such variation, we need to develop 3D feature descriptors to extract only the useful features from video frames. If the extracted features are selective and informative, accurate and efficient activity recognition can be achieved.

In this paper, we will analyze the effectiveness of a simple 3D feature descriptors -- Histogram of Oriented Optical Flow (HOOF) -- on classifying human actions from the HMDB51 dataset [4] and KTH dataset [8], as well pre-processing techniques to improve the results. We will also present several state-of-the-art 3D feature descriptors.

## II. PROBLEM STATEMENT

"Detecting humans in films and videos is a challenging problem owing to the motion of the subjects, the camera and the background and to variations in pose, appearance, clothing, illumination, and background clutter [2]." Our project aims to build a feature descriptor, which along with an appropriate classifier and preprocessing method is capable of detecting a specific human action in videos (e.g. surveillance footage), therefore detecting target activities, such as smoking in a public park, or drawing a gun in a bank. A descriptor which can describe actions in video must operate over three dimensions: it must use the two dimensions from each frame, and track the changes in these frames over time. Our project involves researching ways to extract information about people in images and video in such a way that it reduces noise factors, implementing descriptors that perform this function, and evaluating accuracy of models trained using the data generated.

## III. OBJECTIVES AND GOALS

Our ultimate goal is to gain a better understanding of what constitutes a good 3D feature descriptor, and implement one which is capable of capturing information from video useful in classifying human actions. To accomplish this goal, we must explore and implement different approaches to finding features in images (2D) and video (3D), and evaluate their descriptive ability by testing the performance of different classifiers which use the features extracted by these descriptors to train and test classification models. Specifically, for 2D feature description, we researched the ability of Histogram of Oriented Gradients (HOG) to describe objects, and used HOG algorithm as an inspiration for our 3D feature descriptor [2]. We then researched Histogram of Oriented Optical Flows (HOOF) as a 3D feature descriptor and explored pyramiding as a way to reduce noise in the HOOF descriptor.

## IV. LITERATURE REVIEW

In this section, we are going to introduce several feature descriptors commonly used for activity recognition.

HOOF: Histogram of Oriented Optical Flow (HOOF) utilizes the insights gained by Histogram of Oriented Gradients (HOG) [2] to generate a feature descriptor for videos. It uses locally normalized histograms to encode information about a video that are independent of the scale of the moving person as well as the direction of motion. HOOF calculates Optical Flow using any method, such as Lucas-Kanade, Gunnar-Farneback or Horn-Schunck, and then bins the optical flow vectors according to its primary angle from the horizontal axis, and weighted according to its magnitude. Finally, the histogram is normalized to sum up to 1 [3].

SIFT: Scale-Invariant Feature Transform (SIFT) descriptors are computed from normalized image patches, i.e. regions around salient keypoints extracted from the image. Those key points are typically edges, and they help to achieve invariance to image rotation. A descriptor is then computed using those image patches; it is manifested as a 3D histogram of gradient location and orientation, where location is quantized into a 4 by 4 location grid and the gradient angle is quantized into 8 orientations. The resulting descriptor is of dimension 128. Each orientation plane represents the gradient magnitude corresponding to a given orientation [3, 6].

GLOH: Gradient location-orientation histogram (GLOH) is an extension of the SIFT descriptor designed to increase the performance of SIFT. SIFT descriptors are computed and placed in a log-polar location grid with three bins in radial direction and 8 in angular direction. The central bin is not divided in angular directions. This results in 17 location bins (See Figure 1). The gradient orientations are quantized in 16 bins. This gives a 272 bin histogram. Finally, the descriptor is reduced with PCA, and the 128 largest eigenvectors are computed and used for description [6, 8].

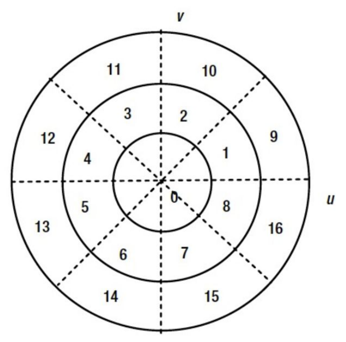

*Figure 1: Log-polar Location Grid Used for GLOH [5]*

## V. IMPLEMENTATION DETAILS

Implementing the 3D Feature Descriptor involves ingesting the video, pre-processing each video's frames, and extracting the sequential optical flow in order to create our final feature vector. These steps are summarized in Figure 2, below. First, video ingestion is performed using OpenCV 3.0's VideoCapture() library function, which allows discrete frames to be obtained from a video file of unbounded size. Next, each frame is pre-processed in order to normalize all possible video frames to a specific size. We chose to use a target size of 160x120, which all frames were resized to using linear interpolation. Next, the images were converted to grayscale using OpenCV's BGR to Gray conversion function.

Next, utilize the pre-processed frames in order to extract a sequential optical flow. This is done by feeding the frames into getSequentialOpticalFlow algorithm (Figure 3). The algorithm iterates through each frame, and finds the histogram of oriented optical flow between each two consecutive frames using the optical flow.

The getOpticalFlow method utilizes Gunnar-Farneback Optical Flow algorithm provided by OpenCV to extract optical flow for each pair of consecutive frames. Gunnar-Farneback Optical Flow calculates optical flow by using the displacement between the neighborhood of two frames. The neighborhood of each frame is approximated by using a polynomial function, and this can be efficiently implemented by a sequence of hierarchical convolutions. Finally, the polynomial coefficients are used for the displacement calculation with the assumption that the entire signal is a single quadratic polynomial [7].

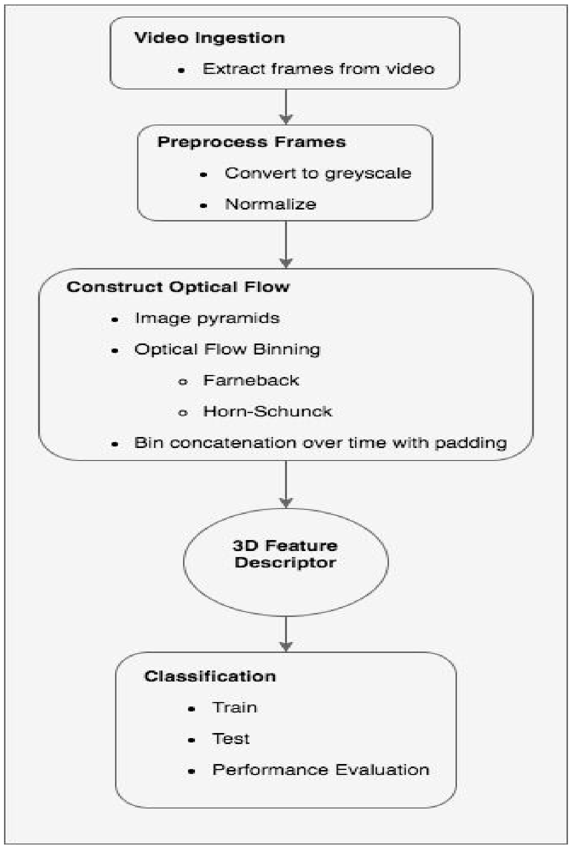

*Figure 2: Process Flow Diagram*

```
getSequentialOpticalFlow(frames):
    histograms = []
    for i in frames.length - 1:
        current = frames[i]
        last = frames[i+1]
        // Gets flow vector given two
frames
        flow = getOpticalFlow(last,
current)
        // getHistogram bins the flow by
        // orientation and magnitude
        histogram = getHistogram(flow)
        histograms.push(flow)
```

*Figure 3: getSequentialOpticalFlow algorithm*

We also implemented a histogram algorithm to convert the orientation of the found flow into a consist-length set of bins. The orientation of the flow is binned into nine bins, 1 bin for each orientation between 0 and 180 degrees in a 20 degree interval. This is done with the magnitude of each flow. The HOOF descriptors for each bin are normalized in order to introduce better invariance to illumination, edge contrast and shadowing. An example of this procedure on two consecutive frames is detailed in Figure 4.

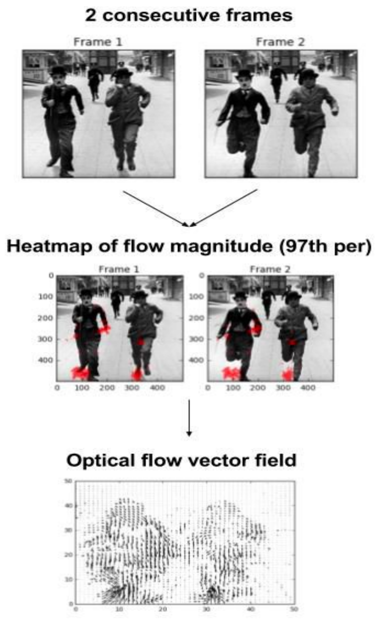
2 consecutive frames
Frame 1
Frame 2
Heatmap of flow magnitude (97th per)
Frame 1
Frame 2
Optical flow vector field

*Figure 4: Process of Calculating Optical Flow on Two Consecutive Frames*

Due to the characteristics of our dataset, which include a significant amount of variation and image noise, require the group to utilize techniques in order to improve accuracy. As well as processing frames as-is (“baseline”), we introduced gaussian image pyramids in order to transform each frame during the optical flow extraction. Image pyramids work by downsampling and blurring each frame in a level-based scheme, with each level indicating a smaller scaled size. Before processing each frame, we generated 3 levels of down samples, and concatenate them to the original frame. Image pyramid helps reduce the spatial and intensity variance of the dataset.

In order to ensure a consistent feature vector size that is invariant of the length of the video, we used four different methods, namely padding with trailing zeroes, padding with reflection, trimming and segmentation. Padding with trailing zeros method pads optical flow vector with zeroes to match the longest frame length in the dataset. Padding with reflection method pads the optical flow vector with the reflection of the vector mirrored along the edge of the array (imagine playing a clip of video in an oscillating manner) to match the longest frame length in the dataset. Trimming method finds the video with the shortest length, and equally trim the heads and tails of the other videos, so that all videos have the same length and retain the middle frames, which contains the core of the activities as validated by empirical testing. Segmentation method takes the shortest video with length n, and slice all other videos in segments of length n; the last segment is padded with trailing zeros if its size is less than n. Then, all segments are labeled identically as their parent and processed as independent data.

Lastly, each optical flow vector of frame pairs is concatenated together in order to yield a final 3D feature vector. These 3D feature vectors can then be fed into a classifiers for training and inference.

We selected 5 human actions, namely “Running”, “Walking”, "Punching", "Hand Clapping", "Hand Waving" from two distinctive datasets -- HMDB51 [4] and KTH [8]. HMDB51 is a large human activity database. It contains snippets of movies and sports videos that are labeled by names of activities. Due to the nature of their origins, the snippets are noisy, as they often contain disorderly backgrounds, camera shaking/panning and disjunct frames caused by film transitions such as cuts. This made it difficult to classify actions. KTH, in contrast, is much more clean, as they are created with the purpose to provide labeled data for activity recognition. This dataset contains videos shot with a still camera in front of an actor, wearing clothes that is easily distinguishable from the monotonic background, performing the specified action repeatedly while facing a different direction on each repetition.

## VI. RESULTS AND ANALYSIS

Figure 5 shows the performances on classifying 5 classes (“Running”, “Hand Clapping”, "Punching", "Walking", "Hand Waving" ) in HMDB51 dataset and KTH dataset, using cross-validation with five folds for training video files. Due to memory constraint on our machines, we took 25 videos per action (125 total) from the HMDB51 dataset, and 8 videos per action (40 total) from the KTH dataset.

Our best results came when we combined pyramid, segmentation, and a Logistic Regression classifier, which classifies 5 classes in KTH and HMDB at accuracies of 78.0% and 55% respectively, which are much higher than random guessing at 20%. Pyramiding being the best result in both tests is expected. As previously defined, pyramiding can help reduce the spatial and intensity variance of the dataset, so it was hypothesized to improve our performances. Of all methods to reduce size invariance of videos, segmentation performed the best on both datasets. Segmentation outperforming the other preprocessing techniques in KTH is expected (78% accuracy), because all the videos consist of the actor repeating the same action multiple times: this effectively gives us a larger dataset of data that is likely to be labelled correctly. Segmentation also worked best in HMDB51 (55%), but due to the noisy nature of the dataset, some segments may not contain the labeled activity. Therefore, we may have created some false labels during segmentation; however, the fact that it outperformed other techniques means the same benefits discussed for KTH also applied in this dataset. Logistic regression may have performed well because it takes into account probabilistic relationships between many independent variables (attributes) and one dependent variable (class), so it may have worked well in reducing the effect of noisy attributes in addition to gracefully handling the large dimensionality of the dataset.

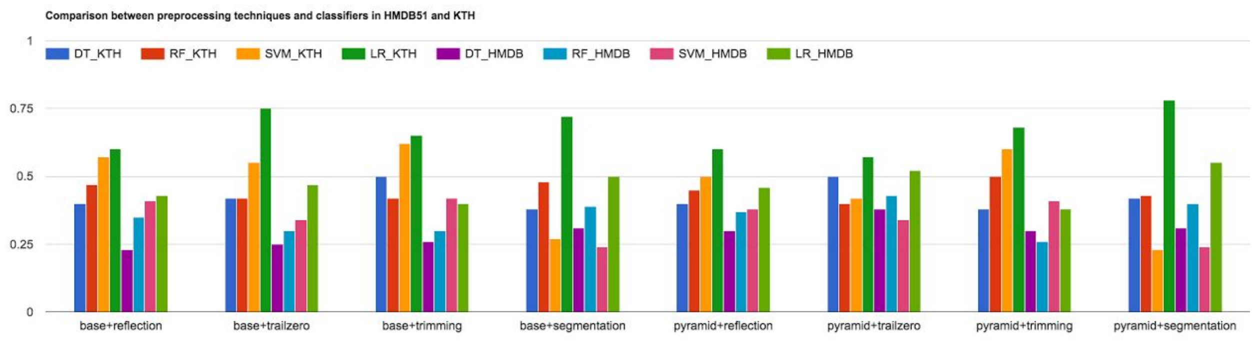

*Figure 5: Performance comparison between preprocessing techniques on HMDB dataset, KTH dataset, and various classifiers. The Y-axis represents the accuracy. The labels {DT, RF, SVM, LR} are mapped to classifiers {Decision Tree, Random Forest, Support Vector Machine, Logistic Regression}. The affixes KTH and HMDB represents datasets that the classifiers are applied to.*

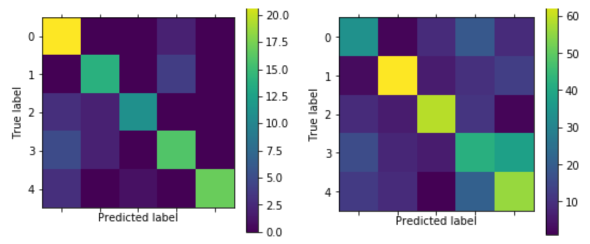

*Figure 6: Confusion Matrix for the most accurate classification method in KTH (left) and HMDB51 (right). The labels {0, 1, 2, 3, 4} are mapped to {run, hand clapping, punching, walking, hand waving}*

Figure 6 shows the confusion matrix for the most accurate classification method, i.e. pyramiding, segmentation and Logistic Regression classifier. Upon observation in the KTH confusion matrix, we can see that walking is occasionally misclassified as running, possibly because they are actions with similar motion. Interestingly, hand clapping is sometimes misclassified as walking, although the occurrences are relatively small. In HMDB51, we can see similar confusion in classifying running and walking; however, most of the confusion seemed to come from the inability to distinguish between walking and waving. Without context, this result means that the walking and waving actions are identical to each other; however, we determine that this is not true by intuition, and we believe this is cause by the noisy nature of the dataset.

## VII. CONCLUSION AND FUTURE DIRECTION

Our results show that the combination of pyramiding in order to account for frame invariance, segmenting in order to create uniform feature vector lengths, and Logistic Regression classifier most accurately classifies the actions in our dataset. We implemented our own versions of HOOF, pyramiding, and Optical Flow calculation to better understand the mechanics of video feature descriptors. Moreover, we also studied the effects of both spatial invariance, fine scale binning in locally normalized histograms, and temporal invariance, pyramiding, on classifiers for video datasets. We considered different ways to overcome the challenge of variable length videos, which lead to variable length feature descriptors, including padding 0s to the end of a feature vector to make each vector's length equal to the length of the longest vector, including only the middle portion of a feature vector, and segmenting videos into uniform lengths, and arrived at the conclusion that pyramiding does provide improvement over base HOOF, that segmentation is the best way to create uniform feature lengths, and that Logistic Regression is the best classifier to use on this kind of data.

**Future work:** Our implementations of pyramiding and Optical Flow are both considerably slower than the standard library implementations, leaving us considerable room to optimize our implementations and provide faster classification. Furthermore, we did not have enough time to combine our pyramiding with our Optical Flow calculations; rather, we tested each one independently, using the library functions for the other where needed. To truly understand the current state of video feature descriptors, we would need to combine our pyramiding and our Optical Flow calculation to provide complete spatial and temporal invariance. We would also like to continue to research the best methods for dealing with variable frame rates and video lengths and coercing those differing frame rates and video lengths to a same size feature vector. We also ran into problems where our feature vectors were too long, and we ran out of memory. We would like to research ways to compress these feature vectors to conserve as much memory as possible.

## VIII. ACKNOWLEDGEMENTS

We'd like to thank Bharathkumar "Tiny" Ramachandra for his mentorship and support for this project.

## IX. REFERENCES

[1] Abdulmunem, Ashwan, Yukun Lai, and Xianfang Sun. "3D GLOH Features for Human Action Recognition." Cardiff University.

[2] Dalal, Navneet, and Bill Triggs. "Histograms of oriented gradients for human detection." 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). Vol. 1. IEEE, 2005.

[3] Chaudhry Rizwan, Avinash Ravichandran, Gregory Hager and Rene Vidal. "Histograms of Oriented Optical Flow and Binet-Cauchy Kernels on Nonlinear Dynamical Systems for the Recognition of Human Actions." Center for Imaging Science, Johns Hopkins University.

[4] HMDB: a large human motion database (http://serre-lab.clps.brown.edu/resource/hmdb-a-large-human-motion-database/)

[5] Abdulmunem, Lai, and Xianfang Sun. "3D GLOH Features for Human Action Recognition"

[6] Abdulmunem, Ashwan, et al. "3D GLOH Features for Human Action Recognition." University of Kerbala.

[7] Farneback, Gunnar. "Two-Frame Motion Estimation Based on Polynomial Expansion." Computer Vision Laboratory, Linkoping University.

[8] Recognition of Human Actions: KTH Action Database (http://www.nada.kth.se/cvap/actions/)