# VISION BASED BEHAVIOR RECOGNITION OF LABORATORY ANIMALS FOR DRUG ANALYSIS AND TESTING

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND

ELECTRONICS ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCES

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Selçuk Sandıkcı

August 2009

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. A. Bülent Özgüler(Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Pınar Duygulu Şahin(Co-supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. Billur Barshan

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Selim Aksoy

Approved for the Institute of Engineering and Sciences:

Prof. Dr. Mehmet Baray
Director of Institute of Engineering and Sciences

# ABSTRACT

## VISION BASED BEHAVIOR RECOGNITION OF LABORATORY ANIMALS FOR DRUG ANALYSIS AND TESTING

Selçuk Sandıkcı
M.S. in Electrical and Electronics Engineering
Supervisors: Prof. Dr. A. Bülent Özgüler
and Assist. Prof. Dr. Pınar Duygulu Şahin
August 2009

In pharmacological experiments, a popular method to discover the effects of psychotherapeutic drugs is to monitor behaviors of laboratory mice subjected to drugs by vision sensors. Such surveillance operations are currently performed by human observers for practical reasons. Automating behavior analysis of laboratory mice by vision-based methods saves both time and human labor. In this study, we focus on automated action recognition of laboratory mice from short video clips in which only one action is performed. A two-stage hierarchical recognition method is designed to address the problem. In the first stage, still actions such as sleeping are separated from other action classes based on the amount of the motion area. Remaining action classes are discriminated by the second stage for which we propose four alternative methods. In the first method, we project 3D action volume onto 2D images by encoding temporal variations of each pixel using discrete wavelet transform (DWT). Resulting images are modeled and classified by hidden Markov models in maximum likelihood sense. The second method transforms action recognition problem into a sequence matching problem

by explicitly describing pose of the subject in each frame. Instead of segmenting the subject from the background, we only take temporally active portions of the subject into consideration in pose description. Histograms of oriented gradients are employed to describe poses in frames. In the third method, actions are represented by a set of histograms of normalized spatio-temporal gradients computed from entire action volume at different temporal resolutions. The last method assumes that actions are collections of known spatio-temporal templates and can be described by histograms of those. To locate and describe such templates in actions, multi-scale 3D Harris corner detector and histogram of oriented gradients and optical flow vectors are employed, respectively. We test the proposed action recognition framework on a publicly available mice action dataset. In addition, we provide comparisons of each method with well-known studies in the literature. We find that the second and the fourth methods outperform both related studies and the other two methods in our framework in overall recognition rates. However, the more successful methods suffer from heavy computational cost. This study shows that representing actions as an ordered sequence of pose descriptors is quite effective in action recognition. In addition, success of the fourth method reveals that sparse spatio-temporal templates characterize the content of actions quite well.

*Keywords:* Action recognition, discrete wavelet transform, hidden Markov models, pose sequences, bag of words.

# ÖZET

## İLAÇ ÇÖZÜMLEMESİ VE TESTİ İÇİN LABORATUAR HAYVANLARININ DAVRANIŞLARININ GÖRÜ TABANLI TANINMASI

Selçuk Sandıkcı
Elektrik ve Elektronik Mühendisliği Bölümü Yüksek Lisans
Tez Yöneticileri: Prof. Dr. A. Bülent Özgüler
ve Yrd. Doç. Dr. Pınar Duygulu Şahin
Ağustos 2009

Farmakolojik deneylerde psikoterapik ilaçların etkilerinin ortaya çıkarılmasında ilaca maruz bırakılmış laboratuar farelerinin davranışlarının görü algılayıcılar ile gözlenmesi yaygın olarak kullanılan bir yöntemdir. Günümüzde pratik olması sebebiyle bu tür gözetim işlemleri insanlar tarafından gerçekleştirilmektedir. Laboratuar hayvanlarının davranış analizini görü tabanlı yöntemler kullanarak otomatikleştirmek hem zaman hem de iş gücünden tasarruf sağlayacaktır. Bu çalışmada sadece tek bir hareket içeren kısa video kliplerden laboratuar farelerinin hareketlerinin otomatik olarak tanınması üzerine odaklanılmıştır. Bu problemi çözmek için iki aşamalı sıradüzensel bir tanıma metodu tasarlanmıştır. İlk aşamada uyuma gibi durağan hareketler hareket alanı miktarına dayanılarak diğer hareket sınıflarından ayrılmıştır. Geriye kalan hareket sınıfları dört alternatif yöntem önerdiğimiz ikinci aşama ile ayırt edilmiştir. Birinci yöntemde ayrık dalgacık dönüşümüyle (ADD) her bir pikseldeki zamansal değişimler kodlanarak 3B hareket hacimlerinin 2B imgeler üzerine izdüşümü alınmıştır.

Oluşan imgeler saklı Markov modelleri ile modellenmiş ve en büyük olabilirlik kriterine göre sınıflandırılmıştır. İkinci yöntem deneğin her bir çerçevedeki pozunu açıkça betimleyerek hareket tanıma problemini dizi eşleştirme problemine dönüştürmektedir. Poz betimlenmesinde deneğin arkaplandan bölütlenmesi yerine zamansal olarak aktif parçaları dikkate alınmıştır. Çerçevelerdeki pozları betimlemek için yönlü gradyanların histogramları kullanılmıştır. Üçüncü yöntemde hareketler, değişik zamansal çözünürlüklerde tüm hareket hacmi kullanılarak hesaplanan düzgelenmiş uzamsal-zamansal gradyanların histogramlarından oluşan bir küme ile betimlenmiştir. Son yöntemde hareketlerin bilinen uzamsal-zamansal şablonlardan oluştuğu ve bu şablonların histogramı ile betimlenebileceği varsayılmaktadır. Hareketlerde bu tür şablonların yerini belirlemek ve betimlemek için sırasıyla çok ölçekli 3B Harris köşe sezicisi ile yönlü gradyanların ve optik akış vektörlerinin histogramları kullanılmıştır. Önerilen hareket tanıma çatısı erişilebilir bir fare hareket veri kümesi üzerinde denenmiştir. Ayrıca her bir yöntemin literatürdeki iyi bilinen çalışmalarla karşılaştırılması sunulmuştur. İkinci ve dördüncü yöntemin hem literatürdeki ilgili çalışmalara hem de bu çalışmadaki diğer iki yönteme göre genel tanıma başarımında daha üstün olduğu görülmüştür, bununla beraber bu yöntemlerin ağır hesaplama maliyetine sahip oldukları saptanmıştır. Bu çalışma davranışların sıralı poz betimleyiciler dizisi şeklinde ifade edilmesinin hareket tanımada oldukça etkili olduğunu göstermiştir. Ayrıca dördüncü yöntemin başarısı seyrek uzamsal-zamansal şablonların hareketlerin içeriğini oldukça iyi nitelendirdiğini ortaya koymuştur.

*Anahtar Kelimeler:* Hareket tanıma, ayrık dalgacık dönüşümü, saklı Markov modelleri, poz dizisi, kelimeler torbası.

# ACKNOWLEDGMENTS

I owe my sincere gratitude to my supervisor Prof. Dr. A. Bülent Özgüler for his supervision, guidance, suggestions, and support throughout my studies leading to this thesis.

I would like to express my deepest thanks to my co-supervisor Assist. Prof. Dr. Pınar Duygulu Şahin for her positive attitude, help, and guidance in this study. I am also thankful to Prof. Dr. Billur Barshan and Assist. Prof. Dr. Selim Aksoy for their revisions and suggestions for my thesis.

I am grateful to Zeynep Yücel and Mehmet Kök for their friendship, support, discussions, and patience in answering all my questions. I also would like to thank all my friends including my colleagues at Aselsan for their friendship, support, and assistance. Without them I could not complete this thesis. My special thanks go to Ayşegül Çiftçi, who has my been greatest motivation, for her endless understanding, patience, and support in my hard times.

I am also thankful to my parents for their perpetual love and for being understanding and supportive all the time. I dedicate my thesis to them.

I am also appreciative of the financial support from EEEAG–105E065 project funded by TÜBİTAK, the Scientific and Technical Research Council of Turkey.

# Contents

# List of Figures

# List of Tables

**To my parents Pakize & Ramazan Sandıkcı...**

# Chapter 1

# Introduction

In pharmacological experiments involving laboratory mice under the influence of psychotherapeutic drugs, behavior pattern of the mice reveals important clues about physiological effects of the drug. To uncover the effects of injected drug, the subject must be monitored and its actions must be recorded in an objective and measurable manner until effects of the drug disappear. Currently in pharmacological experiments, mouse which is subjected to the drug is recorded by a visual sensor, such as a camera, during the experiment, and afterwards behaviors that the subject exhibited are annotated by human observers.

Considering that pharmacological experiments are repeated many times on hundreds of mice for statistical accuracy and consistency, an automated action recognition system is highly desirable, since it would save great amount of both time and human labor. Moreover, different human observers can record different results for the same behavior pattern; however, an automated behavior recognition system would produce more consistent behavior annotations. Another desired specification of a behavior recognition system is that it should be non-intrusive i. e., any disturbance to the subject, such as environmental modifications or implanted electrodes are not allowed. In a non-intrusive monitoring

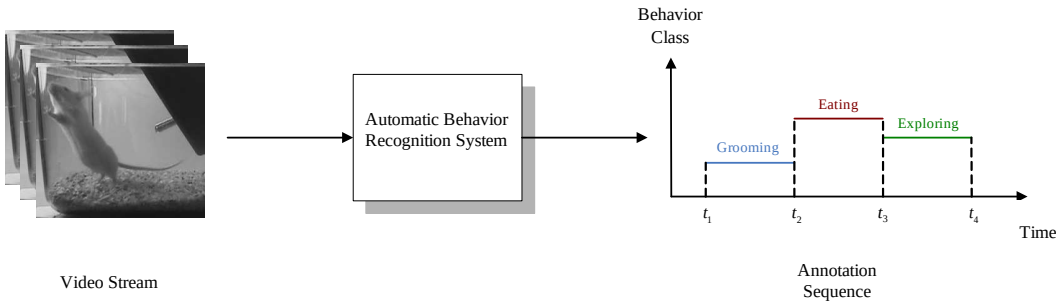



Figure 1.1: Overview of a continuous behavior recognition system.

environment, the behavior pattern of the subject is not affected by external factors during the experiments. A computer vision based action recognition system satisfies above specifications while also being cheap due to the accessibility of low-cost cameras.

## 1.1 Problem Statement

For continuous and automated monitoring of laboratory animals, aforementioned behavior recognition system acquires a video stream in which performed behaviors are desired to be correctly recognized and labeled. Output of the system is an annotation sequence with a time line associated with the given video (see Figure 1.1 for the system overview). In this thesis, as a preliminary study for such a system we addressed the problem of recognizing actions from small video clips, in which exactly one class of behavior is performed, based on computer vision and machine learning technology. The main functionality of our system is depicted graphically in Figure 1.2.

Problem addressed in this thesis is a 3D pattern recognition problem in the general sense and it is stated as follows:

**Given:** A set of training video clips, $\{V_i,\ i = 1, \ldots, \#\text{ of training videos}\}$ with corresponding ground-truth labels, $\{\ell_i |\ \ell_i \in \mathcal{L}\}$, where $\mathcal{L}$ is the set

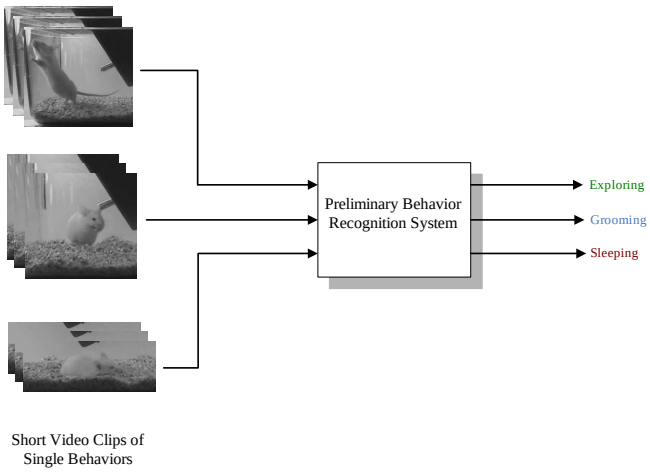



Figure 1.2: Overview of the preliminary system designed in this thesis.

of behavior classes to be classified. In this thesis $\mathcal{L}$ consists of five classes, namely drinking, eating, exploring, grooming, and sleeping.

**Aim:** is to train an action recognition system, which can correctly determine the class $\ell_{\mathrm{T}}$ of a given test video clip $V_{\mathrm{T}}$ with unknown class.

Although the problem seems quite naive, there are a number of challenges that need to be addressed in order to design a robust action recognition system [1]. The first and the most important challenge is the unconstrained motion of the subject i. e., the mouse under observation naturally must not be aware that its behaviors are being visually recorded. Thus, it behaves in the most natural manner e.g., it may turn its back to camera or perform the same behavior with significant variations each time. Note that unlike well-studied human action recognition from carefully recorded datasets, such as KTH [2] and Weizmann [3], mice action recognition introduces a harder pattern recognition problem. In addition, some of the mouse behaviors happen in a burst, making tracking body parts a nontrivial task. In contrast to human's articulated body, a mouse has a highly deformable blob-like body which can stretch and compress significantly. Therefore, one can not easily fit mouse body to a template model or skeleton [4]. Moreover, there are a few body parts to track such as eyes and tail. Peripherals of mouse body are quite small in size to detect and track, and can be occluded

by other body parts. To sum up, mouse introduces lots of challenges due to its body structure and movements.

Recording environment also presents a number of challenges. Designed recognition system must be robust to scale and rotation changes, camera viewpoint angle, and illumination variations. Lastly, cluttered and noisy background arises due to the litter in a mouse cage.

In this thesis, we present a hierarchical classification framework to deal with the behavior recognition of laboratory mice. First stage of our framework is to discriminate still actions such as sleeping from the others. We take advantage of the amount of motion area, that is covered by the subject while performing the behavior, to separate sleeping from other behaviors. Experiments on real mice videos validate our assumption.

The second stage, which is not as simple as the first stage, classifies the remaining four actions, namely, drinking, eating, exploring, and grooming. Regarding significant pattern variations and randomness in these actions, we prefer to follow a bottom-up approach i. e., classification based on low level features that are extracted from raw video data. We propose four alternative methods for the implementation of the second stage:

**1**. Inspired by the work of Töreyin *et. al* [5], we utilize Discrete Wavelet Transform (DWT) to analyze temporal characteristics of individual pixels. Then, we form action *summary* images (ASIs) using the amount of temporal fluctuations at each pixel in the video volume. ASIs are transformed into subimage sequences by blockwise raster scanning. We form multidimensional observation sequences by taking intensity histograms of each subimage in the sequence. Hidden Markov models (HMMs) with continuous observation densities are used to model observation sequences. Classification of action videos with unknown classes is carried out by trained HMMs in the maximum likelihood sense.

**2**. We follow the work of Hatun and Duygulu [6] to represent actions as sequences of pose prototypes. A pose codebook which contains all the possible pose prototypes is constructed by clustering pose descriptors extracted from training videos. As in [6], we describe the pose in a given frame by histograms of oriented gradients [7]. Different from [6], we consider only temporally active points in pose description. After representing an action by a sequence of pose prototypes, the classification problem reduces to matching the pose sequence associated with the action to the nearest known pose sequence in the database. In matching, we used the length of "Longest Common Subsequence" as similarity metric [8].

**3**. We modify the method of Zelnik-Manor and Irani [9] to classify mice actions. According to the method, first a temporal video pyramid is constructed by smoothing and downsampling the video along temporal axis. This operation enables one to analyze video at different temporal resolutions. At each level of the pyramid, normalized 3D gradient vectors are computed for all points in the entire video volume. Points with small temporal gradient vectors are discarded, since they are not believed to participate in the action. We propose an adaptive thresholding scheme for discarding those points. To represent an action, histograms of normalized gradient vectors from each pyramid level are combined to form a feature set. Leave one out cross validation in conjunction with nearest neighbor classifier (1-NN) is employed for classification purposes.

**4**. By following the method of Laptev *et. al* for human action recognition [10], an action is represented by an unordered collection of sparse spatio-temporal templates. We first detect salient points in the action volume by 3D Harris corner detector. Then, we describe each detected point by histograms of oriented gradient and optical flow vectors computed from neighborhood of the detected point. Descriptor of each point is assigned to a visual codeword, which is element of a codebook. The codebook is constructed by quantizing descriptors extracted from training data. At the end, an action is represented by a histogram which

indicates the distribution of codewords within the action volume. We have used 1-NN and radial basis support vector machine classifiers for classification.

## 1.2 Literature Review

There has been a vast amount of research conducted on automated action recognition from video sequences, since it has quite appealing application areas such as security surveillance, pharmacological experiments, patient monitoring, and human computer interaction. As expected, human action recognition is investigated much more thoroughly than animal action recognition. Action recognition methods which do not directly exploit human body structure and movement kinematics are also applicable to animal action recognition. Such methods are referred as "bottom-up" approaches in the literature. In this section, first we discuss related work on human action recognition built on "bottom-up" approaches. Afterwards, we review animal action recognition methods on which researchers rarely focused.

### 1.2.1 Human Action Recognition

Computer vision based human action recognition has been a challenging and active research area for the past two decades (see [11, 12, 13, 14] for surveys on human action recognition). Action recognition methods can be divided into three categories as frame-based, volumetric, and model-based.

In frame-based approaches, action descriptors are built on 2D image features, such as contours and silhouettes. Bobick and Davis [15] described video volumes by Motion Energy Images (MEI) and Motion History Images (MHI), which indicate existence and recency of motion in the video volume, respectively. They applied template matching techniques to MEIs and MHIs to recognize aerobic

actions. In [16], a camera viewpoint invariant action recognition system based on geometrical properties of spatio-temporal (ST) volumes was proposed. 2D image contours are stacked to construct corresponding ST volumes. A similar approach which is based on building space-time shapes from 2D image silhouettes was proposed by Gorelick *et. al* [17]. 3D features such as saliency, shape information and orientation can be extracted by solving a Poisson equation related to constructed space-time volume. Notice that all these approaches rely on clean segmentation of foreground objects, thus they are limited to static or well-modeled background cases.

On the other hand, volumetric approaches treat actions as 3D space-time volumes and extract descriptors from action volume in either localized or global manner. One of such methods was introduced by Zelnik-Manor and Irani who claimed that global histograms of normalized gradients computed from entire video volume are sufficient to discriminate basic actions such as walking and jogging [9]. Efros *et. al* attempted to classify low resolution human videos such as football actions by a novel ST descriptor based on coarse histograms of noisy optical flow measurements over a subject-centric ST volume and normalized cross-correlation [18].

Sparse volumetric representations of actions have also been employed by various researchers. These kind of representations require that ST points which are considered to be "interesting" within the action volume are detected first. Features extracted from vicinity of detected points are believed to reveal important characteristics about content of the action. Representing an action as collections of those features "sparsifies" the description. One of the ST interest point detectors is 3D Harris corner detector which belongs to Laptev [19]. It has been successfully used by [2, 10, 20, 21] for representing actions as collections of small ST templates (cuboids). Another detector based on local maxima of video volumes filtered by temporal Gabor filters and spatial Gaussian smoothing kernels is

proposed by Dollar *et. al* [22]. In [22], proposed detector is used for recognition of facial gestures, human, and mice behavior. Dollar's detector has also been used by [23] in conjunction with probabilistic Latent Semantic Analysis (PLSA) for unsupervised classification of human actions. Scovanner *et. al* [24] extended the well-known Scale Invariant Feature Transform (SIFT) descriptor [25] to 3D for action recognition.

A major problem with sparse action representations is that geometrical relations between ST templates within the action volume are ignored, thus resulting in poorer description. To overcome this shortcoming, a number of methods, which consider geometrical topology of ST keypoints, based on ST correlograms [26], Discriminative Subsequence Mining [27], graph-based Bayesian inference [28] and improved PLSA with implicit shape models [29] were developed. Another issue with sparse representations is that the number of detected points for smooth actions might be too low to describe the action. For instance in our case, Laptev's detector locates only a few points for sleeping actions. In [30] in order to avoid the curse of extreme sparsity, combinations of features at 2D Harris corners detected along both space and time are learned by data mining methods. For a better action representation Wong and Cipolla [31] tried to refine interest point detection process by imposing global information based on non-negative matrix factorization.

Subvolume matching has been investigated for action recognition as an alternative to sparse keypoint approaches. Ke *et. al* [32] developed a real-time event detection system exploiting 3D box features based on optical flow. Shechtman and Irani [33] extended conventional 2D image correlation to 3D spatio-temporal volume correlation to detect complex dynamic actions in video sequences.

Kim and Cipolla [34] applied Canonical Correlation Analysis, which is conventionally used to maximize correlation of two 1D random variables, on video

volumes which are treated as tensors. They achieved significant recognition rates for human actions and hand gestures.

Model-based action representations incorporate temporal kinematics into action recognition framework. Hidden Markov models (HMM), which are one of the well-known state-space models, have been widely utilized in modeling temporal dynamics of actions. Li [35] assumed that human actions can be modeled by HMMs with finite number of states and Gaussian Mixture Model (GMM) observations. Hierarchical HMMs are exploited to represent complex activities, which are considered to be Markov chains of simpler actions in [36]. Peursum *et. al* attacked occlusion and imperfect segmentation problems in action recognition by modifying HMMs for missing observations [37].

### 1.2.2 Animal Action Recognition

There are a few studies for behavior recognition of laboratory mice in the computer vision literature. First of them is performed by Dollar *et. al* [22] as mentioned in the previous subsection. In more detail, they expressed behaviors as histograms of "cuboid" prototypes which are visual words of a previously constructed "cuboid" codebook. Here "cuboid" refers to windowed spatio-temporal data around interest points which are localized by the detector proposed in [22]. For describing "cuboids" they tried a bunch of feature representations, such as normalized pixel values, brightness gradient vectors, and optical flow within the "cuboids". Another work on mice action recognition belongs to Jhuang *et. al* [38]. Their action recognition method imitates biological visual processing architecture of human brain by hierarchical spatio-temporal feature detectors. They showed that their system works for both mice and human action recognition.

Xue ve Henderson [39] constructed affinity graphs using extracted spatio-temporal features to detect Basic Behavior Units (BBUs) in artificially created mice videos. BBUs are assumed to be building blocks of more complex behaviors. They applied Singular Value Decomposition (SVD) to discover BBUs in a given complex behavior. Although their model works quite well for artificial videos, it is hard to predict if it is applicable to real mice videos. In addition to mice action recognition there has been also some vision based research on multiple mice tracking based on optical flow, active contours [40, 41], and contour and blob trackers [4].

Apart from mice, behaviors of some other animals such as lions, bears, and insects attract researchers' attention. Burghardt and Càlič [42] exploited face characteristics and trajectories of lions in annotating locomotive behaviors of lions for animal monitoring in wildlife. For face tracking they combined the method of Viola and Jones [43] and Lucas-Kanade feature tracker. They took advantage of frequency characteristics of horizontal and vertical components of face trajectory to discriminate between behaviors. In [44], to aid biological studies on insects, stereo 3D tracking and spectral clustering of 3D motion features are employed to discriminate basic behaviors of grasshoppers, such as walking, jumping, and standing still. Wawerla *et. al* [45] built "motion shapelets" from low level features such as gradient responses and foreground statistics by Adaboost classifiers to detect grizzly bears at the Arctic Circle.

## 1.3 Contributions

Main contributions of this thesis can be summarized as follows:

- We present a simple method based on motion area to distinguish still actions from the others.

- We propose a novel action recognition method based on DWT and HMMs. First we simplify the action recognition problem to image classification problem by DWT analysis. Then, we make use of HMMs to model inherent randomness in action *summary* images.

- We adopt a pose-based action recognition algorithm to mice case. We modify it to handle the absence of valid background models as in our case. For segmentation of the subject prior to pose description, we employ temporal gradient magnitudes, which is a simple and computationally cheap operation.

- We utilize a human action representation method for mice action recognition in conjunction with adaptive thresholding and 1-NN classification.

- We test the performance of a sparse keypoint based algorithm which is shown to be very successful in human action recognition on mice actions.

- We performed a detailed parameter search for all the methods in the second stage of our framework in order to achieve the highest recognition rate possible.

## 1.4 Outline of the Thesis

The thesis is organized as follows: in Chapter 2, we present the details of the first stage of our action recognition framework. Chapters 3, 4, 5, and 6 explain methods of the second stage in greater detail. In Chapter 7, we test proposed methods on a commonly available mice action dataset [22] and evaluate parameters of the methods. We also compare recognition performance of proposed methods with the algorithms in [22] and [38] that get results on this dataset. Finally in Chapter 8, we conclude this thesis by giving a short summary and providing some future research ideas.

# Chapter 2

# Discrimination of Still Actions

All of the methods which will be explained in the next chapters exploit temporal variations in the pixel intensities. Sleeping being a quite still action can cause serious degeneracy in the classification stage. Therefore, we propose a simple method to determine whether class of a given video is sleeping or not. We integrate this algorithm into our action recognition framework as an initial stage. A given video is first assigned to sleeping or "non-sleeping" class using this method. If it is classified as sleeping, then given video is not proceeded to the second stage. On the other hand, if it turns out to be a "non-sleeping" video, then it is processed by the second stage to uncover its true class. One can consider this procedure as a hierarchical classification framework.

According to the method, behaviors are classified based on area which is spanned by the subject while performing the behavior. The main assumption is that during sleeping an animal is almost still and the spanned area is minimal compared with other behaviors.

In order to determine the spanned area for a given video clip $V$, temporal standard deviation $\sigma_t$ of each pixel in the video volume is computed empirically,

$$\sigma_t(x,y) = \sqrt{\frac{1}{N}\sum_{t=1}^{N}\big(V(x,y,t) - \overline{V}(x,y)\big)^2},$$

where $N$ is the number of frames in $V$ and $\overline{V}(x,y)$ is the empirical mean along temporal axis at pixel $(x,y)$

$$\overline{V}(x,y) = \frac{1}{N}\sum_{t=1}^{N} V(x,y,t).$$

Then, the standard deviations of all pixels are thresholded with a predefined threshold $\epsilon$. Pixels having standard deviation above threshold are considered to be moving pixels. Let $\Psi$ be a black-white image and formed by thresholding operation,

$$\Psi(x,y) = \begin{cases} 0 & \text{if } \sigma_t(x,y) < \epsilon \\ 1 & \text{if } \sigma_t(x,y) \geq \epsilon \end{cases}.$$

Summing values of all the pixels in $\Psi$ gives us the number of moving pixels. Thus, the spanned area $\psi$ is computed as,

$$\psi = \sum_{(x,y)\in\Psi} \Psi(x,y).$$

Prior to classification of a given video as sleeping or "non-sleeping", the training videos in the dataset are divided into two groups as sleeping videos and non-sleeping videos. For all videos in the training set, spanned areas $\{\psi_i \mid i = 1,\ldots,\#\text{ of training videos}\}$ are computed. Then, we fit two univariate Gaussian distributions, $G_S$ and $G_{NS}$ to the computed spanned areas, one for sleeping videos and one for non-sleeping videos. $G_S$ is given as,

$$G_S(\psi_S,\mu_S,\sigma_S) = \frac{1}{\sqrt{2\pi\sigma_S^2}}\ \exp\left(\frac{-(\psi_S-\mu_S)^2}{2\sigma_S^2}\right),$$

where $\mu_S$ and $\sigma_S^2$ are the mean and variance of $G_S$. $G_{NS}$ is similarly formed.

Given a test video $V_{\mathrm{T}}$ with its spanned area $\psi_{\mathrm{T}}$, we estimate the likelihoods $P(\psi_{\mathrm{T}}|G_S)$ and $P(\psi_{\mathrm{T}}|G_{NS})$ i. e., probability that $V_{\mathrm{T}}$ is a sleeping or non-sleeping

video. To evaluate likelihoods, $\psi_{\mathrm{T}}$ is simply substituted in $G_S$ and $G_{NS}$ i. e., $P(\psi_{\mathrm{T}}|G_S) = G_S(\psi_{\mathrm{T}}, \mu_S, \sigma_S)$ and $P(\psi_{\mathrm{T}}|G_{NS}) = G_{NS}(\psi_{\mathrm{T}}, \mu_{NS}, \sigma_{NS})$. Then $V_{\mathrm{T}}$ is classified according to maximum likelihood criterion,

$$\ell_{\mathrm{T}} = \begin{cases} \texttt{sleeping} & \text{if } P(\psi_{\mathrm{T}}\ |G_S) > P(\psi_{\mathrm{T}}\ |G_{NS}) \\ \texttt{non-sleeping} & \text{otherwise.} \end{cases},$$

where $\ell_{\mathrm{T}}$ is the assigned label for $V_{\mathrm{T}}$. In Chapter 7, it will become clear that this simple method works pretty well for the discrimination of sleeping from other behaviors.

# Chapter 3

# Discrete Wavelet Transform and Hidden Markov Models based Approach

When a subject performs an action, its movements in space-time generate temporally varying image points. According to content of the movements, image points exhibit different types of temporal variations. For instance, if the subject performs a periodic action, generated image points are likely to have temporal periodicity in pixel intensities. Similarly, a stationary action such as sleeping or standing still probably give rise to image points which have temporal stationarity. Furthermore, an action may consist of temporal action units. Consider a mouse first scratching its back by its rearfeet rapidly and then licking its forefeet slowly. As a whole, this sequence of action units constitute a *grooming* action. Performing this action will generate a bunch of image points each undergoing different temporal variation schemes. Image points corresponding to rearfeet will have a periodic behavior during scratching, but then they will become stationary while licking forefeet. Therefore, while performing an action, generated

image points may have temporally varying characteristics. Temporal variations of image points exhibit important clues on the type of performed action.

In order to address action recognition of laboratory animals, in this chapter temporal characteristics of image points are analyzed by discrete wavelet transform (DWT) applied along the temporal axis. Only the highband subsignal is considered in analyzing temporal properties of image points, since most of the information is carried in the highband subsignal. A simple score for each image point is computed according to temporal fluctuations at that point. Then, using the scores for all image points, an action *summary* image (ASI) is formed. Each point $(x, y)$ in ASI encodes the amount of temporal activity of the image point $(x, y)$ in the action. Small groups of image points with nonzero values in ASI are likely to arise from background clutter. So they are detected by morphological operations and are set to zero. Afterwards, a bounding box with minimal size which includes all the nonzero points in ASI is determined. A sequence of subimages is obtained by raster scanning the cropped ASI i. e., ASI points inside the bounding box. Taking histograms of intensity values inside each subimage generates a multidimensional observation sequence to be modeled by hidden Markov models (HMMs). For each action video in the dataset an HMM model is trained using extracted observation sequences. Lastly, a given action with unknown class is classified with trained HMMs in the maximum likelihood sense.

## 3.1 Discrete Wavelet Transform and Action *Summary* Image Formation

Discrete wavelet transform (DWT) is a very valuable tool in multiresolution signal analysis. DWT has the ability of capturing both spectral and temporal information. Consider a discrete signal $x[n]$. To decompose $x[n]$ to two subsignals carrying only low and high frequency components, one needs to filter $x[n]$ by

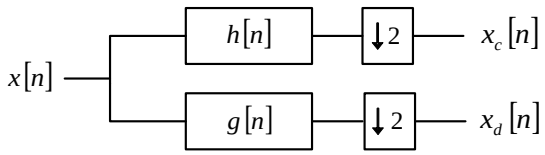



Figure 3.1: Single level wavelet decomposition.

a pair of lowpass and highpass filters, $h[n]$ and $g[n]$, respectively. Afterwards filtered signals are downsampled by 2 to obtain

$$x_c[n] = \sum_{k=-\infty}^{\infty} x[n]\ h[2n-k], \tag{3.1}$$

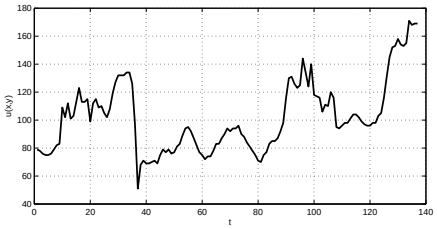
$$x_d[n] = \sum_{k=-\infty}^{\infty} x[n]\ g[2n-k], \tag{3.2}$$

where $x_c[n]$ and $x_d[n]$ are approximation and detail coefficients of the signal $x[n]$. Or synonymously they are called lowband and highband subsignals. Equations (3.1) and (3.2) are mathematical expressions of convolving $x[n]$ with $h[n]$ and $g[n]$ and downsampling by 2. A graphical illustration of discrete wavelet decomposition is shown in Figure 3.1.

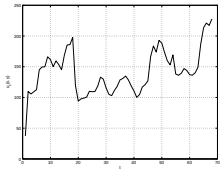
It is worth to mention that if perfect reconstruction, minimal filter length, and compact support are desired properties, lowpass and highpass filters, $h[n]$ and $g[n]$ need to satisfy the relationship,

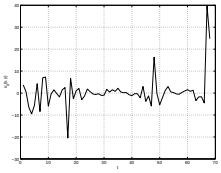
$$\left|H(e^{jw})\right| = \left|G(e^{j(\pi-w)})\right|, \tag{3.3}$$

where $H(e^{jw})$ and $G(e^{jw})$ are frequency responses of $h[n]$ and $g[n]$, respectively. Filters $h[n]$ and $g[n]$ satisfying the relationship (3.3) are called quadrature mirror filters (QMF), since their frequency response is symmetric with respect to $w = \pi/2$. Design of QMF is another issue and is out of the scope of this thesis. For more information on QMF, the reader is referred to [46] and [47]. Filtering $x[n]$ with quadrature mirror filters, $h[n]$ and $g[n]$, results in two subband signals each carrying only low and high frequencies of the original signal $x[n]$. Downsampling subband signals by 2 stretches the frequency responses of those to fullband and generates the highband and lowband subsignals $x_c[n]$ and $x_d[n]$. Notice that




(a) Original 1D temporal signal $u(x, y)$.





(b) Lowband subsignal $u_c(x, y)$. (c) Highband subsignal $u_d(x, y)$.

Figure 3.2: Temporal wavelet decomposition example.

lengths of subband subsignals, $x_c[n]$ and $x_d[n]$ are half of the original signal's due to the downsampling operations.

In this chapter, temporal characteristics of image points while performing an action are analyzed by DWT. Consider an image point at pixel $(x, y)$. Concatenating image points at pixel $(x, y)$ along the temporal axis forms a 1D signal $u(x, y)$. Considering an action volume $V$ as a set of 1D temporal signals, we apply DWT to all elements of that set. An example of 1D temporal signal, $u(x, y)$ and its lowband and highband subsignals, $u_c(x, y)$ and $u_d(x, y)$ are shown in Figure 3.2.

Temporal variations of the signal $u(x, y)$ are encoded in the highband subsignal $u_d(x, y)$. Thus, we are not interested in the lowband subsignal $u_c(x, y)$. A simple measure of temporal variations in $u(x, y)$ is the number of zero crossings in $u_d(x, y)$. Higher number of zero crossings indicates high frequency activity in $u(x, y)$. Before counting zero crossings in $u_d(x, y)$, samples close to zero are forced to be zero not to count small fluctuations as a zero crossing. After counting zero crossing numbers for all 1D temporal signals in the action volume, we form the action *summary* image. Intensity value of pixel at location $(x, y)$ in ASI

(a) Exploring action.

(a) Grooming action.

(a) Drinking action.

Figure 3.3: Sample frames from various actions and their corresponding ASIs.

is set to zero crossing number of 1D signal $u(x, y)$:

$$\text{ASI}(x, y) = \rho(x, y), \forall x, y \in V,$$

where $\rho(x, y)$ is the number of zero crossings in $u_d(x, y)$. Some example frames from various actions and their ASIs are illustrated in Figure 3.3.

Small objects in ASIs are assumed to be generated by background clutter noise. Thus, objects with small area are removed. Then, a bounding box image $BB$ is formed such that all of the pixels with nonzero intensity values in ASI are assured to be inside $BB$:

$$\begin{aligned} &BB(x - x_1 + 1, y - y_1 + 1) = \text{ASI}(x, y), \\ &\forall x, y, \ x_1 < x < x_2, y_1 < y < y_2, \\ &\text{and } \{\text{ASI}(x, y) \neq 0 \mid \forall x, y, 0 < x < x_1 \text{ or } x_2 < x < X, \\ &\qquad\qquad 0 < y < y_1 \text{ or } y_2 < y < Y\} = \{\}, \end{aligned}$$

where the indices $x_1$, $x_2$, $y_1$ and $y_2$ satisfy $0 < x_1 < x_2 < X$ and $0 < y_1 < y_2 < Y$. Here, $X$ and $Y$ are number of rows and columns in ASI. Notice that forming $BB$ from ASI is a simple cropping operation. An example ASI and its determined bounding box image $BB$ is depicted in Figure 3.4.

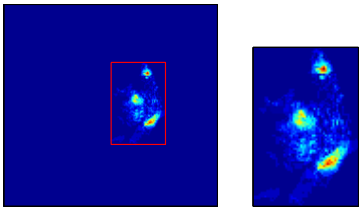


Figure 3.4: An ASI and its bounding box image $BB$.

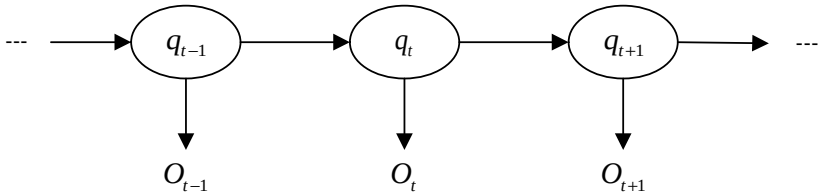



Figure 3.5: An illustration of HMM with 1st order Markov chain. $q_t$ and $O_t$ are state and observation at time $t$, respectively.

## 3.2 Hidden Markov Models

Hidden Markov models (HMMs) have been widely used in speech recognition [48], face recognition [49, 50], gesture recognition [51], and action recognition [35, 52]. HMMs are well-known for their applications on modeling time series.

An HMM is a doubly stochastic process consisting of an *unobservable* stochastic process and an observable process which generates the observable outputs and dependent on the *unobservable* one. Unobservable stochastic process is a Markov chain which includes a finite number of states, a state transition probability matrix, and an initial state probability matrix. Observable process consists of a set of observation symbol probability distributions. A graphical illustration of an HMM with first-order Markov chain as the underlying process is depicted in Figure 3.5. In this section, brief information on HMMs will be given by following Rabiner's excellent tutorial on HMMs [48]. To define an HMM with discrete states and observation symbols, first its elements are identified below:

1. $N$ is the number of distinct states in an HMM. State set is $S = \{S_1, S_2, \ldots, S_N\}$. The state at time $t$ is denoted by $q_t$.

2. $M$ is the number of distinct observation symbols. Observation symbol set is $\Upsilon = \{v_1, v_2, \ldots, v_M\}$.

3. State transition probability matrix is denoted by $A = \{a_{ij}\}$.

$$a_{ij} = P\left[q_{t+1} = S_j \mid q_t = S_i\right],\ i \text{ and } j \in \{1, \ldots, N\}.$$

$a_{ij}$ is the probability of passing from state $S_i$ to $S_j$ at time $t$. Note that row sum of $A$ must be equal to 1,

$$\sum_{j=1}^{N} a_{ij} = 1, \forall i \in \{1, \ldots, N\}. \tag{3.4}$$

4. Set of observation symbol probability distributions is $B = \{b_j(k)\}$,

$$b_j(k) = P\left[O_t = v_k \mid q_t = S_j\right],\ j \in \{1, \ldots, N\} \text{ and } k \in \{1, \ldots, M\},$$

where $O_t$ is the observation symbol at time $t$. Notice that $b_j(k)$ is the probability of generating symbol $v_k$ at state $S_j$ at time $t$.

5. Initial state probability distribution is denoted by $\Pi = \{\pi_i\}$ ,

$$\pi_i = P[q_1 = S_i],\ i \in \{1, \ldots, N\}.$$

$\pi_i$ is the probability that first state is $S_i$. Note that the sum of $\Pi$ must be equal to 1,

$$\sum_{i=1}^{N} \pi_i = 1. \tag{3.5}$$

A compact definition for an HMM is denoted by $\Lambda = (A, B, \Pi)$. A completely defined HMM can generate an observation sequence, $\boldsymbol{O} = O_1 O_2 \ldots O_T$ by the following steps:

1. Start by selecting an initial state $q_1 = S_i$ according to initial state probability distribution $\Pi$.

2. Generate the first observation $O_1 = v_k$ according to observation symbol probability distribution at state $S_i$, $b_i(k)$.

3. Transit to state $q_{t+1} = S_j$ according to state transition probability $a_{ij}$.

4. If $t < T$ increment $t$ by 1 and go back to Step 2 to generate a new observation. Else terminate generating observations.

### 3.2.1 HMMs with Continuous Probability Densities

HMMs can be extended to generate vector observation symbols instead of scalar symbols. In that case observation symbol probability distributions are required to be a mixture of continuous densities

$$b_j(O) = \sum_{m=1}^{M} c_{jm} \mathfrak{N}(O, \boldsymbol{\mu}_{jm}, \boldsymbol{\Sigma}_{jm}),\ 1 \leq j \leq N,$$

where $O$ is observation symbol vector, $c_{jm}$ is the weight of $m^{th}$ mixture component at state $j$ and $\mathfrak{N}$ is a elliptically symmetric density with mean vector $\boldsymbol{\mu}_{jm}$ and covariance matrix $\boldsymbol{\Sigma}_{jm}$ for the $m^{th}$ mixture component in state $j$. In practice, $\mathfrak{N}$ is assumed to be a multivariate Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_{jm}, \boldsymbol{\Sigma}_{jm})$. Mixture weights $c_{jm}$ must satisfy the following constraints so that $b_j$ sums to 1,

$$\sum_{m=1}^{M} c_{jm} = 1,\ 1 \leq j \leq N,$$

$$c_{jm} \geq 0,\ \ 1 \leq j \leq N,\ \ 1 \leq m \leq M.$$

### 3.2.2 Three Canonical Problems for HMMs

If elements $(N, M, A, B, \Pi)$ of an HMM are specified explicitly, then one can use it to generate observation sequences or model an observation sequence with appropriate values of elements. To effectively use HMMs in practice, one needs to solve three canonical problems of HMM:

1. Given an HMM model $\Lambda = (A, B, \Pi)$ and an observation sequence $\boldsymbol{O} = O_1 O_2 \ldots O_T$, what is the probability that the given observation sequence is generated by the given model i. e., P($\boldsymbol{O}|\Lambda$)?

2. Given an HMM model $\Lambda = (A, B, \Pi)$ and an observation sequence $\boldsymbol{O} = O_1 O_2 \dots O_T$, what is the *best* state sequence $\mathcal{Q} = q_1 q_2 \dots q_T$ explaining the observations?

3. Given an observation sequence $\boldsymbol{O}$ and initial estimate of model parameters $(A^{(1)}, B^{(1)}, \Pi^{(1)})$, how do we update model parameters to maximize $\mathrm{P}(\boldsymbol{O}|\Lambda)$?

In a classification framework, Problem 1 and Problem 3 correspond to testing and training problems, respectively. Assume that we are supposed to design a system which can discriminate observation sequences according to their class. By solving Problem 3, we can train HMMs for each class with known observation sequences. In a similar way, by solving Problem 1, we can test the trained HMMs for a new observation sequence with unknown class i. e., we can compute the probability of generating the given observation sequence by each trained HMM. Then, the new observation sequence is assigned to the class of HMM, which gives the highest probability of generating that sequence. The main motivation behind Problem 2 is to uncover hidden dynamics of the model. In a classification framework, uncovering state sequence may be useful for model improvement, such as deciding on the number of hidden states and observation symbols.

There are well-defined methods to solve the three canonical problems of HMMs. Problem 1 can be solved by "Forward-Backward Procedure". To solve Problem 2 one can use Viterbi algorithm. Lastly, solution for Problem 3 is addressed by Baum-Welch algorithm which is a special case of Expectation-Maximization procedure. The reader is referred to Rabiner's tutorial [48] for theoretical details of these methods.

## 3.3 Modeling and Classifying Actions by HMMs

If HMMs are desired to be used in modeling and classification, then first one needs to define the observation sequence and HMM model, which are suitable for the application. Recall that by exploiting DWT and forming ASI for each action, we were able to reduce the action recognition problem to an image classification problem. In view of the successful applications of HMMs on face recognition [53, 54], we prefer to follow the work of [54] to model ASIs by HMMs.

### 3.3.1 Generating Observation Sequences from ASIs

In order to generate an observation sequence $\boldsymbol{O}$ from an ASI, we divide the bounding box image $BB$ into a grid of $N_{SI} \times M_{SI}$ overlapping subimages $\Omega_{SI}$. Tracing the subimages in a raster scan fashion generates a sequence of subimages with length $N_{SI}M_{SI}$. Tracing scheme is illustrated in Figure 3.6. Regarding Figure 3.6, $\Omega_{SI}, X_{SI}, Y_{SI}, x_{SI}$, and $y_{SI}$ denote the subimage, width and height of the subimage, width and height of the overlap region between consecutive subimages, respectively. The lengths $x_{SI}$ and $y_{SI}$ are related to $X_{SI}$ and $Y_{SI}$ by an overlap ratio, $\zeta_{SI}$ i. e., $x_{SI} = \zeta_{SI} X_{SI}$ and $y_{SI} = \zeta_{SI} Y_{SI}$. The relations between $N_{SI}$, $M_{SI}$ and $X_{SI}, Y_{SI}$ are given as,

$$N_{SI} = \left\lfloor \frac{X_{BB} - x_{SI}}{X_{SI} - x_{SI}} \right\rfloor,$$
$$M_{SI} = \left\lfloor \frac{Y_{BB} - y_{SI}}{Y_{SI} - y_{SI}} \right\rfloor,$$

where $X_{BB}$ and $Y_{BB}$ are width and height of bounding box image $BB$. After obtaining the subimage sequence, for each subimage $\Omega_{SI}$, an $m_{SI}$ bin histogram based on pixel intensities is computed. Forming a sequence from computed histograms in the same order with the subimage sequence gives us the observation sequence $\boldsymbol{O} = O_1 O_2 \dots O_T$ to be used in HMMs. Here, observation symbol $O_n$ is a $m_{SI}$-dimensional vector and corresponds to the histogram of the $n^{th}$ subimage.

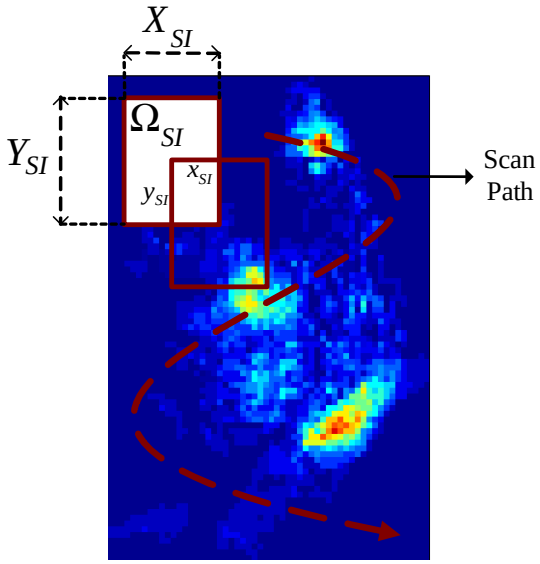



Figure 3.6: Scanning scheme of $BB$ image.

Length of the observation sequence is $T$ which is simply the product of $N_{SI}$ and $M_{SI}$ i. e., $T = N_{SI}M_{SI}$.

### 3.3.2 Training of HMMs

One HMM model $\Lambda_i$ with $N$ distinct states is trained for each action video $V_i$ in the training set. Each HMM model is trained by executing the steps proposed by [54]:

1. Consider the observation sequence $\boldsymbol{O}_i$ extracted from $BB_i$ associated with $V_i$. Cluster the set of observation symbols $\{O_n,\ n = 1, \ldots, T\}$ into $N$ clusters $C_k$ with cluster centers $c_k, k = 1, \ldots, N$. Actually, here each cluster corresponds to a state.

2. Determine state of each observation symbol $O_n$ by assigning it to the nearest cluster center $c_k$ according to Euclidean distance

$$q_n = \underset{k}{\operatorname{argmin}} \|O_n - c_k\|^2 \text{ for } n = 1, \ldots, T \text{ and } k = 1, \ldots, N,$$

where $q_n$ is the state of $O_n$. This step corresponds to assigning each observation symbol to an appropriate state.

3. We assume that the observation symbol probability density $b_k(O)$ at state $k$ is a single Gaussian distribution with mean vector $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$. Empirically estimate the initial values of mean vector and covariance matrix of each cluster (state), $C_k$,

$$\boldsymbol{\mu}_k^{(1)} = \frac{1}{|C_k|} \sum_{O_n \in C_k} O_n,\ 1 \leq k \leq N,$$
$$\boldsymbol{\Sigma}_k^{(1)} = \frac{1}{|C_k|} \sum_{O_n \in C_k} (O_n - \mu_k)^T\ (O_n - \mu_k),\ 1 \leq k \leq N.$$

4. According to [48], random initialization of $A$ matrix and $\Pi$ vector is mostly acceptable in real world applications. So initial estimates, $A_i^{(1)}$ and $\Pi_i^{(1)}$ are initialized randomly with stochastic constraints (3.4) and (3.5).

5. Note that the initial estimates $\{\boldsymbol{\mu}_k^{(1)}, \boldsymbol{\Sigma}_k^{(1)}\ ,\ k = 1, \ldots, N\}$ constitute $B_i^{(1)}$. Using initial estimates $\left(A_i^{(1)}, B_i^{(1)}, \Pi_i^{(1)}\right)$ estimate model parameters $\Lambda_i = (A_i, B_i, \Pi_i)$ by Baum-Welch algorithm.

At the end, for each action video $V_i$ in the training set an HMM model $\Lambda_i$ is trained.

### 3.3.3 Classification by HMMs

To classify a given test action $V_{\mathrm{T}}$, first its $\mathrm{ASI}_{\mathrm{T}}$ and $BB_{\mathrm{T}}$ image are computed as elaborated in section 3.1. Then, observation sequence $\boldsymbol{O}_{\mathrm{T}}$ is formed as according to subsection 3.3.1. Action $V_{\mathrm{T}}$ is assigned to the class of most likely HMM model

$$\underset{j}{\operatorname{argmax}}\ P\left(\boldsymbol{O}_{\mathrm{T}} \mid \Lambda_j\right), 1 \leq j \leq \#\text{ of trained HMMs.}$$

In our framework, the training set is the rest of the dataset with test action omitted. This procedure is repeated for all of the actions in the dataset and overall recognition accuracy is measured to be the average of all classification runs. This classification approach is known as the "leave-one-out cross validation".

# Chapter 4

# Spatial Histograms of Oriented Gradients based Approach

Appearance, shape or *pose* of the subject while performing an action is a very important clue in action recognition. In fact, one can think actions as temporal sequences of poses. There has been a significant amount of research on pose-based action recognition [55, 56, 57, 58, 59, 60]. In this chapter, behavior recognition of laboratory animals is addressed by following a pose-based action recognition method, specifically the method proposed in [6].

The main assumption in this chapter is that an action is an ordered collection of pose *prototypes*, $\mathcal{P} = \{p_1, \ldots, p_K\}$. Pose descriptors extracted from training actions are clustered by k-means clustering algorithm to form the codebook of pose *prototypes*, $\mathcal{P}$. Center of each cluster is treated as a pose *prototype*, $p_k, k = 1, \ldots, K$. Poses in each frame of a given action is described by spatial histograms of oriented gradients (SHOG). In SHOG computation, only a subset of image points, which are believed to participate in the action, are considered. Set of such points will be referred as Region of Interest (ROI) in the rest of this chapter. The points inside ROI are assumed to be temporally *active* i .e., magnitudes of

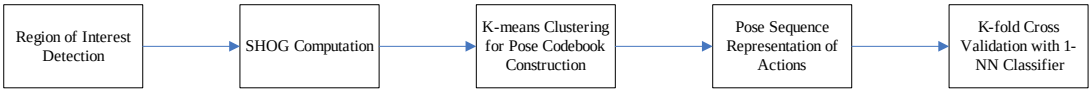



Figure 4.1: Main blocks of pose sequence based approach.

temporal gradient vectors at those points are high. Thus, ROI can be easily located by simple temporal gradient computation and thresholding. One needs to perform the following steps to represent a given action as a pose sequence:

- For each frame in the action,
  - Determine ROI by temporal gradient computation and thresholding.
  - Describe the pose in the frame by SHOG descriptor.
  - Assign the pose in the frame to the nearest pose *prototype*, $p_k$.
- Represent the action as a pose sequence, $\mathcal{S} = s_1 s_2 \dots s_N$, where each element $s_n$ is matched to a pose *prototype*, $p_k \in \mathcal{P}$. Here, $N$ is the number of frames in the action.

K-fold cross-validation with nearest neighbor (1-NN) classifier is used to classify an action represented by the pose sequence, $\mathcal{S}$. Similarity metric in 1-NN classifier is the length of "Longest Common Subsequence", which is commonly used for string matching [8]. Figure 4.1 summarizes main steps of the method.

## 4.1 Region of Interest (ROI) Detection

In conventional pose-based action recognition, firstly one needs to detect the subject in the frame and segment it as clean as possible. Unless a valid model for the background is available at hand, object detection and segmentation are quite nontrivial and difficult tasks. Thus, instead of using detection and segmentation for ROI determination, a simpler and computationally cheaper approach

is used in this chapter. Points which have high temporal gradient are assumed to participate in the action, therefore they are taken as ROI in pose description. To determine such points in the frames, temporal gradient vectors are analyzed. Points whose temporal gradient vectors have relatively higher magnitude are assumed to be *active* points. Consider an action as an image sequence, such that it is a mapping from spatio-temporal domain to pixel intensity domain, $V : \mathbb{R}^2 \times \mathbb{R} \mapsto \mathbb{R}$. To estimate temporal gradient of the action, $V$ is smoothed along the $t$ axis by convolving with a Gaussian kernel $g_s$ with variance $\sigma_s^2$ and temporal derivative of smoothed volume is taken,

$$V_t = \partial_t \left( V(\cdot; t) * g_s(t, \sigma_s^2) \right),$$

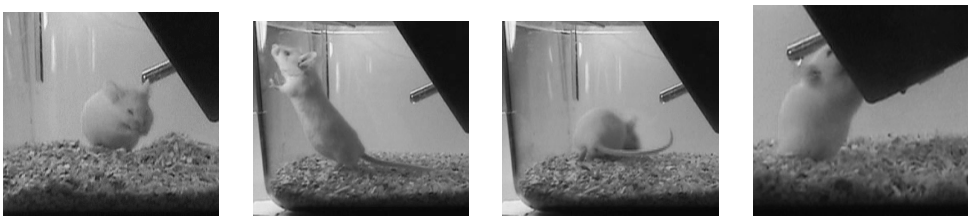
where $g_s$ is the smoothing Gaussian kernel, $g_s(t, \sigma_s^2) = \frac{1}{\sqrt{2\pi\sigma_s^2}} e^{-t^2/2\sigma_s^2}$. Approximating partial derivative operator with 1D discrete filter, $h = [-1\ 0\ 1]$,

$$V_t = \left( V(\cdot; t) * g_s(t, \sigma_s^2) \right) * h.$$

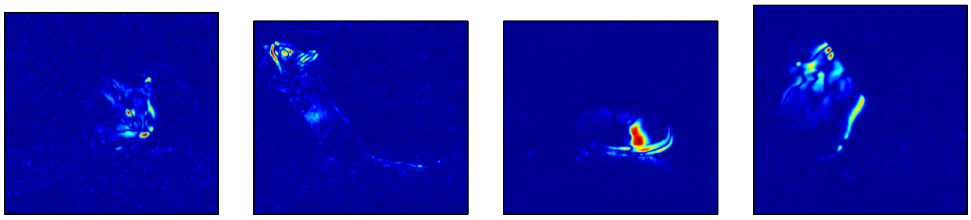
To determine points with relatively high temporal gradients, an adaptive thresholding scheme is employed. For each action, threshold is adjusted according to mean and variance of temporal gradient magnitudes. Points $(x, y)$ satisfying the below inequality are considered to belong ROI at time $t$,

$$|V_t(x, y, t)| - \mu_t > \gamma\sigma_t, (x, y, t) \in V.$$

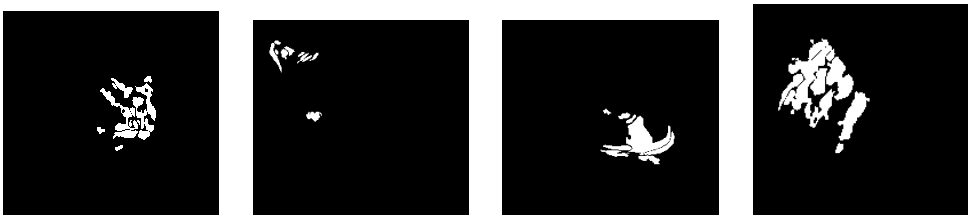
In above inequality, $\mu_t$ and $\sigma_t$ are empirical mean and standard deviation of temporal gradient magnitudes of all points in the action, $\{|V_t(x, y, t)|, (x, y, t) \in V\}$. $|V_t(x, y, t)|$ is the temporal gradient magnitude at point $(x, y)$ in frame at time $t$. $\gamma$ is a constant to adjust the threshold value. Here, magnitudes of temporal gradients are assumed to obey a Gaussian distribution. Points whose temporal gradient magnitude deviates from empirical mean more than $\gamma\sigma_t$ are assumed to have significant amount of temporal variation. In Figure 4.2 (a), some sample frames from various actions are shown. Corresponding temporal gradient magnitudes are illustrated in Figure 4.2 (b). Thresholding temporal gradient magnitudes results in binary ROI images shown in Figure 4.2 (c).



(a) Sample images from various actions.



(b) Corresponding temporal gradient magnitude images.



(c) Thresholded temporal gradient magnitude images.

Figure 4.2: Some examples of ROI detection.

## 4.2 SHOG Computation

SHOG is first proposed by Dalal and Triggs to locate humans in images [7]. In this chapter, following the method of [6] a reduced variety of SHOG is used to describe the pose in the frames. The main intuition behind SHOG descriptors is that shape or pose can be discriminatively described by localized histograms of gradient orientations.

In SHOG descriptor computation, one first needs to estimate spatial gradient field from the given frame. Consider the given frame $I$ as a mapping from spatial domain to intensity domain, $I : \mathbb{R}^2 \mapsto \mathbb{R}$. Its spatial gradient field is a vector field, such that $F_g : \mathbb{R}^2 \mapsto \mathbb{R}^2$. $F_g$ is estimated in a similar fashion to temporal gradient estimation. First, $I$ is smoothed spatially with a bivariate Gaussian to eliminate gradient responses due to noise. Then, derivative of $I$ is taken along $x$ and $y$ axes separately. Convolution with $h = [-1\ 0\ 1]$ discrete filter is used to

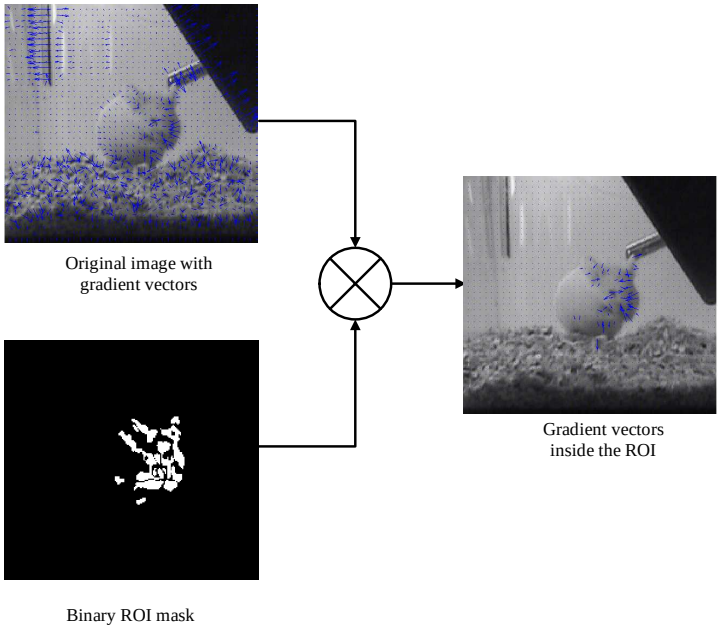



Figure 4.3: Masking operation of spatial gradient vectors outside the ROI.

approximate derivative operator,

$$I_x = \left( I * g_{sp}(\cdot, \sigma_{sp}^2) \right) * h,$$
$$I_y = \left( I * g_{sp}(\cdot, \sigma_{sp}^2) \right) * h^T,$$

where $g_{sp}$ is a 2D Gaussian with zero mean and variance $\sigma_{sp}^2$,

$$g_{sp}(x, y, \sigma_{sp}^2) = \frac{1}{2\pi\sigma_{sp}^2} \, e^{-(x^2+y^2)/2\sigma_{sp}^2}.$$

Estimated gradient field is formed as $F_g = [I_x \; I_y]^T$. Since SHOG descriptor is used to express the pose of subject, gradient vectors arising from background must be discarded. To retain only gradient vectors computed from points on the subject, $F_g$ is masked by the binary ROI image as depicted in Figure 4.3. SHOG computation requires dividing ROI into $n$ *cells* by rectangular or radial partitioning scheme. In this work, radial (circular) partitioning scheme is employed. Center of partitioning scheme is selected as the center of binary ROI image as illustrated in Figure 4.4. After radial partitioning, for each *cell*, $b_i$, an $m$ bin histogram, $h_i$, is computed according to the orientations of gradient vectors in that *cell* as demonstrated in Figure 4.5. While accumulating gradient vectors into histograms bins according to their orientations, each vector is weighted by

Figure 4.4: Center of ROI. Red dot is the center.

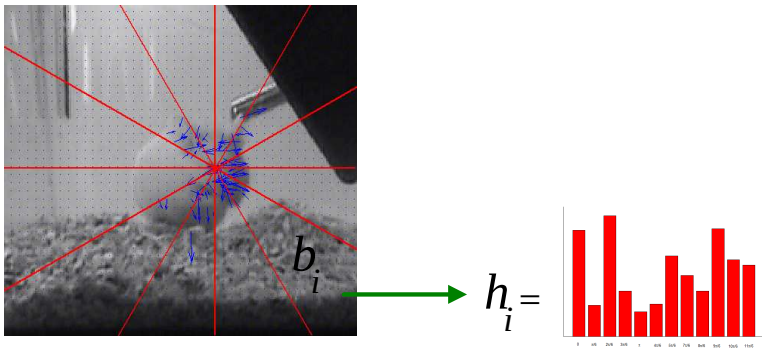



Figure 4.5: SHOG computation process.

its magnitude,

$$h_i(j) = \sum_{k \in b_i} \left|\vec{F_{g_k}}\right| \text{, such that } \theta_j \leq \left(\angle \vec{F_{g_k}}\right) < \theta_{j+1} \text{ and } j = 1, \ldots, m.$$

Each histogram $h_i$ is normalized, such that its $L_2$ norm equals to 1. Combining histograms, $h_i$ from all cells into one final descriptor $f_{\text{SHOG}}$ by simple concatenation operation results in SHOG descriptor which is a $1 \times nm$ vector,

$$f_{\text{SHOG}} = \begin{bmatrix} h_1 & h_2 & \ldots & h_n \end{bmatrix}.$$

## 4.3 K-means Clustering for Pose Codebook Construction

Representing an action as a pose sequence requires a previously constructed pose codebook, or synonymously alphabet or dictionary of poses. A pose codebook is

a set of pose *prototypes*, $\mathcal{P} = \{p_1, \ldots, p_K\}$, where any given frame in an action can be matched to one of the pose *prototypes*. In this chapter, to construct such a codebook, extracted pose descriptors from training actions are clustered into $K$ clusters by k-means clustering algorithm. At the end of this operation, a codebook with $K$ pose *prototypes* being the centers of $K$ clusters is formed.

Assume that $q$ pose descriptors are extracted from the training actions, such that they form a set $Q = \{f_1, f_2, \ldots, f_q\}$. Each element $f_k$ in the set $Q$ is SHOG descriptor of a frame in a training action. If descriptors are assumed to be $d$-dimensional vectors, then $Q$ can be considered as a point cloud in $\mathbb{R}^d$ space. The aim is to determine the most representative descriptors in set $Q$. This problem is solved by k-means clustering algorithm [61], which tries to minimize within-cluster sum of squared distances,

$$\underset{\mathbf{C}}{\operatorname{argmin}} \sum_{i=1}^{K} \sum_{f_j \in C_i} \|f_j - c_i\|^2,$$

where $\mathbf{C}$ is the set of clusters $C_i$, such that $\mathbf{C} = \{C_1, C_2, \ldots, C_K\}$ and $c_i$ is the center of cluster $C_i$.

For clustering point set $Q$ into $K$ clusters, standard k-means algorithm which is explained in Table 4.1 is used. At the end of clustering process, pose codebook $\mathcal{P}$ turns out to be the set of cluster centers, $\mathbf{c} = \{c_1, c_2, \ldots, c_K\}$,

$$\mathcal{P} = \{p_k = c_k, k \in \{1, 2, \ldots, K\}\}.$$

Some example clusters are shown in Figure 4.6. Highlighted regions in images are detected ROIs.

**Given:** a set of points $Q = \{f_1, f_2, \ldots, f_q\}$ and random initial guesses for centers of clusters, $\{c_1^{(1)}, c_2^{(1)}, \ldots, c_K^{(1)}\}$

**Objective:** Find a set of clusters $\mathbf{C} = \{C_1, C_2, \ldots, C_K\}$ minimizing within-cluster sum of squared distances.

**Step1**. Assign each point $f_j$ to the cluster with the closest center,

$$C_i^{(t)} \supset \{f_j : \left\| f_j - c_i^{(t)} \right\|^2 < \left\| f_j - c_k^{(t)} \right\|^2, \forall k = 1, 2, \ldots, K\}$$

**Step2**. If assignments don't change, clustering process is converged. Terminate the process. Else continue with **Step 3**.

**Step3**. Update the cluster centers according to recent assignments, and go back to **Step 1**,

$$c_i^{(t+1)} = \frac{1}{|C_i^{(t)}|} \sum_{f_j \in C_i^{(t)}} f_j$$

Table 4.1: Steps of k-means algorithm for clustering $Q$.

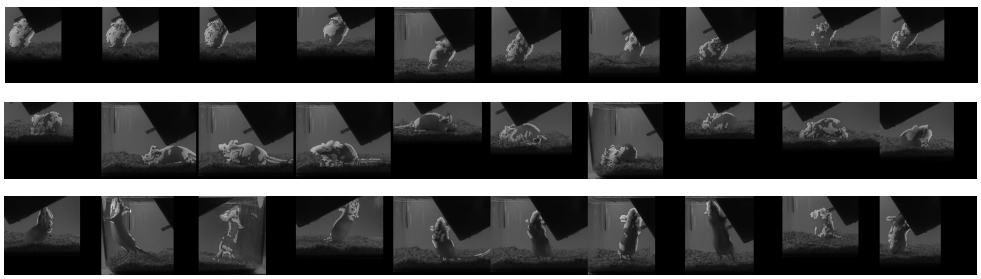


Figure 4.6: Sample clusters obtained by k-means clustering.

## 4.4 Pose Sequence Representation of Actions

Pose sequence representation is the process of describing an action as a sequence of pose *prototypes*, which are elements of a pose codebook. In the previous section, it is explained how to build such a codebook. Consider an action video $V$ consisting of $N$ frames. To construct pose sequence associated with $V$, first for each frame a pose descriptor $f_{\text{SHOG}}$ is computed as explained in sections 4.1 and 4.2. Then, each pose descriptor is assigned to the nearest pose *prototype* according to Euclidean distance metric,

$$s_i = \underset{\mathbf{j}}{\operatorname{argmin}} \| f_{\text{SHOG}_i} - p_j \|^2 \text{ for } j = 1, \ldots, K \text{ and } i = 1, \ldots, N.$$

At the end, $V$ is represented as a pose sequence $\mathcal{S} = s_1 s_2 \ldots s_N$, where each element in the sequence corresponds to the nearest $p_k$ in the codebook.

## 4.5 K-fold Cross Validation with 1-NN Classifier

We used K-fold cross-validation classification scheme, where the dataset is split into K subsets. At each classification run, exactly one subset is separated as the test set and the rest is left as the training set. Classification runs end when all the subsets are used as the test set. Overall classification rate is estimated as the average of K runs. At each classification run, a 1-NN classifier is trained using the pose sequences of training videos, $\{\mathcal{S}_i \mid i \in \{1, 2, \ldots, \#\text{ of training videos}\}\}$. In 1-NN classifier, the similarity metric is the length of Longest Common Subsequence (LCS), which is commonly used to measure similarity of two sequences [8]. Given two sequences, $\mathcal{S}_i$ and $\mathcal{S}_j$ LCS returns concatenation of subsequences, which are common in both sequences. For instance, consider two sequences consisting of letters, 'ABCDE' and 'BCEF'. LCS between these sequences is 'BCE'. In this

**Given:** Two sequences $\mathcal{S}_1$ and $\mathcal{S}_2$ with lengths $m$ and $n$

**Objective:** Find the length of Longest Common Subseqeunce between $\mathcal{S}_1$ and $\mathcal{S}_2$.

**Method:**
Form a $(m+1) \times (n+1)$ matrix $D$

For $i = 1$ to $m+1$
    $D_{i,0} = 0$

For $j = 1$ to $n+1$
    $D_{0,j} = 0$

For $i = 2$ to $i = m+1$
    For $j = 2$ to $j = n+1$
        If $\mathcal{S}_1(i) = \mathcal{S}_2(j)$
            $D_{i,j} = D_{i-1,j-1} + 1$
        Else:
            $D_{i,j} = \max(D_{i,j-1}, D_{i-1,j})$

Length of LCS is $D_{m,n}$.

Table 4.2: Dynamic programming method to compute length of LCS between two sequences.

work, the length of LCS is computed by the dynamic programming method given in Table 4.2.

# Chapter 5

# Global Histograms of 3D Gradients based Approach

In this chapter, method of Zelnik-Manor and Irani [9] is slightly modified to solve action recognition problem for laboratory animals. Motivation of the method is that an action recognition system must handle diversities of real-world actions with no a priori information about the context of the actions. Furthermore, the system must quickly learn new varieties of actions and do not need too many parameters to be tuned. Such a system must employ a general framework to describe action context. Therefore, pursuing a generalized nonparametric activity modeling is more favorable than following a parametric activity model specialized to a restricted set of actions. To build such a generic behavior recognition system, Zelnik-Manor and Irani made general assumptions, which any action likely to obey:

- Behavior representation using motion-based features is invariant to photometric effects, such as illumination changes.
- Behaviors are random space-time processes and can be described by an underlying probability density function (pdf).

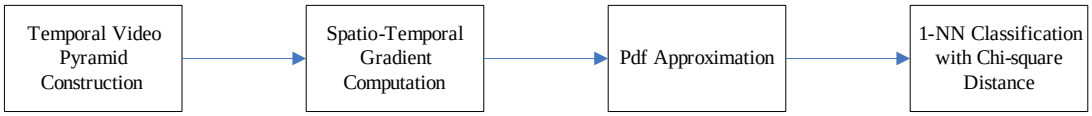



Figure 5.1: Main blocks of spatio-temporal gradients based approach.

- Spatio-temporal gradient vectors computed from raw video intensity values are samples of the underlying process.
- Behaviors consist of different temporal resolutions.
- Two behaviors are similar if their probability density functions at corresponding temporal resolutions are similar.

In order to realize above assumptions, any given action is decomposed into different temporal resolutions by constructing a *temporal pyramid* of the video. Histograms of 3D gradients computed at each level of the temporal pyramid are used to empirically approximate underlying multidimensional probability density functions associated with the action. Classification of a given action is carried out by using a 1-NN classifier based on Chi-square ($\chi^2$) distance between approximated pdfs. An overview of the modified method is given in Figure 5.1.

## 5.1 Temporal Video Pyramid Construction

To capture different temporal resolutions for a given video, a temporal video pyramid has to be built. First level in the pyramid is the given video itself. Video volume at level $(l+1)$, $V^{(l+1)}$, in the pyramid can be constructed by convolving the video volume at level $(l)$, $V^{(l)}$, with a smoothing Gaussian $g_s$ along the temporal axis and downsampling blurred video by 2 along the time

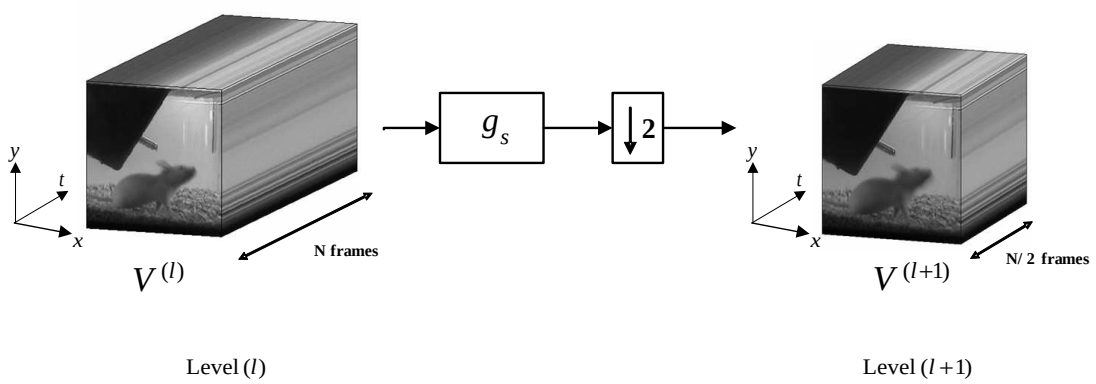



Figure 5.2: Illustration of temporal video pyramid construction.

axis,

$$V_b^{(l)} = V^{(l)}(x, y, t) * g_s,$$
$$V^{(l+1)}(x, y, t) = V_b^{(l)}(x, y, 2t),$$

where $V_b^{(l)}$ is the blurred video volume at level $(l)$. Here, $g_s$ is a 1D Gaussian with zero mean and variance $\sigma_s^2$. Due to downsampling operation by 2, temporal extent of $V^{(l+1)}$ is half of $V^{(l)}$'s. A graphical illustration of temporal pyramid construction is shown in Figure 5.2.

## 5.2 Spatio-temporal Gradient Computation

Spatio-temporal gradient computation is performed for all levels of temporal pyramid,

$$\left(V_x^{(l)}, V_y^{(l)}, V_t^{(l)}\right) = \left(\frac{\partial V^{(l)}}{\partial x}, \frac{\partial V^{(l)}}{\partial y}, \frac{\partial V^{(l)}}{\partial t}\right) \text{ for } l = 1, \ldots, l_{\max},$$

where $l_{\max}$ is the number of levels in the temporal pyramid. Gradient computation results in $l_{\max}$ gradient fields each with 3 components, $(V_x, V_y, V_t)$.

Temporal component of gradient vectors, $V_t$, encodes the rate of change in $V$ along time axis. Space-time points with larger temporal gradients are more likely to correspond to points of interest i. e., points participated in the action.

Likewise, space-time points which are part of the background probably have smaller temporal gradients. To eliminate background and take only foreground information into account, points whose temporal gradient component lower than a threshold are discarded. Notice that this operation is equivalent to a rough spatio-temporal segmentation. A space-time point $(x, y, t)$ in video volume $V^{(l)}$ is discarded if its temporal gradient satisfies below inequality,

$$\left|V_t^{(l)}(x,y,t)\right| - \mu_t^{(l)} < \gamma\sigma_t^{(l)}, (x,y,t) \in V^{(l)}.$$

In the above inequality, $\mu_t^{(l)}$ and $\sigma_t^{(l)}$ are empirical mean and standard deviation of temporal gradient magnitudes, $\{|V_t^{(l)}(x,y,t)|, (x,y,t) \in V^{(l)}\}$. $\gamma$ is a constant to adjust the threshold value. Here, magnitudes of temporal gradients are assumed to obey a Gaussian distribution. Points whose temporal gradient magnitude deviates from empirical mean more than $\gamma\sigma_t^{(l)}$ are assumed to have significant amount of temporal variation.

Gradient vectors are locally normal to the space-time volume, thus orientations of gradient vectors characterizes shape of the volume. On the other hand, gradient magnitudes usually arise from photometric properties, such as contrast and illumination. To suppress the effect of magnitude as much as possible, gradient vectors are normalized such that their $L_2$ norm is 1. In addition, absolute value of normalized gradient vectors is taken, so that invariance to negated contrast differences (e. g. dark object/light background or light object/dark background) and direction of the action (e. g. right-to-left or left-to-right) is ensured. Normalized gradient vectors are computed as,

$$\left(N_x^{(l)}, N_y^{(l)}, N_t^{(l)}\right) = \frac{\left(\left|V_x^{(l)}\right|, \left|V_y^{(l)}\right|, \left|V_t^{(l)}\right|\right)}{\sqrt{\left(V_x^{(l)}\right)^2 + \left(V_y^{(l)}\right)^2 + \left(V_t^{(l)}\right)^2}}\ . \tag{5.1}$$

At the end of gradient computation and normalization, each space-time point $(x, y, t)$ is represented by a $3l_{\max} \times 1$ vector,

$$\left[N_x^{(1)}(x,y,t) \quad N_y^{(1)}(x,y,t) \quad N_t^{(1)}(x,y,t) \quad N_x^{(2)}(x,y,t) \quad \ldots \quad N_t^{(l_{\max})}(x,y,t)\right].$$

## 5.3 Pdf Approximation

In order to think actions as 3D statistical random processes, one needs to find an underlying pdf which generates the observations, which are normalized gradient vectors, $\{N_k^{(l)}, l = 1, \ldots, l_{\max} \text{ and } k \in \{x, y, t\}\}$ in our case. To approximate such a pdf, histograms of observations can be constructed. Since each observation is a $3l_{\max}$-dimensional vector, actual pdf can be approximated by a $3l_{\max}$-dimensional histogram. However, for $l_{\max} = 3$ case and assuming 256 bins for each histogram dimension, we end up a histogram of size $256^9$, which is quite high for today's computers. Instead, one can assume that components of $3l_{\max}$-dimensional gradient vectors are statistically independent from each other. It is a very strong assumption, but individual histogram channels might be discriminative enough [9]. So for a given space-time volume (action), $3l_{\max}$ 1D histograms for each component of the $3l_{\max}$-dimensional gradient vectors are constructed. At the end, $i^{th}$ action in the dataset is represented by $3l_{\max}$ 1D histograms,

$$f_i = \left\{ h_x^{(1)}, h_y^{(1)}, h_t^{(1)}, h_x^{(2)}, \ldots, h_t^{(l_{\max})} \right\},$$

where $h_k^{(l)}$ is a 1D histogram of the normalized gradient component, $N_k^{(l)}$.

In order to analyze the discriminative power of computed descriptors, for the case $l_{\max} = 3$ we construct 9 1D histograms for each action sample, compute the mean histograms for each action class, smooth the mean histograms and plot mean histograms corresponding to same temporal levels. Figure 5.3 shows smoothed mean histograms for each action class and gradient component at 3 temporal scales. Notice that for some gradient components and temporal scales actions can be discriminated easily.

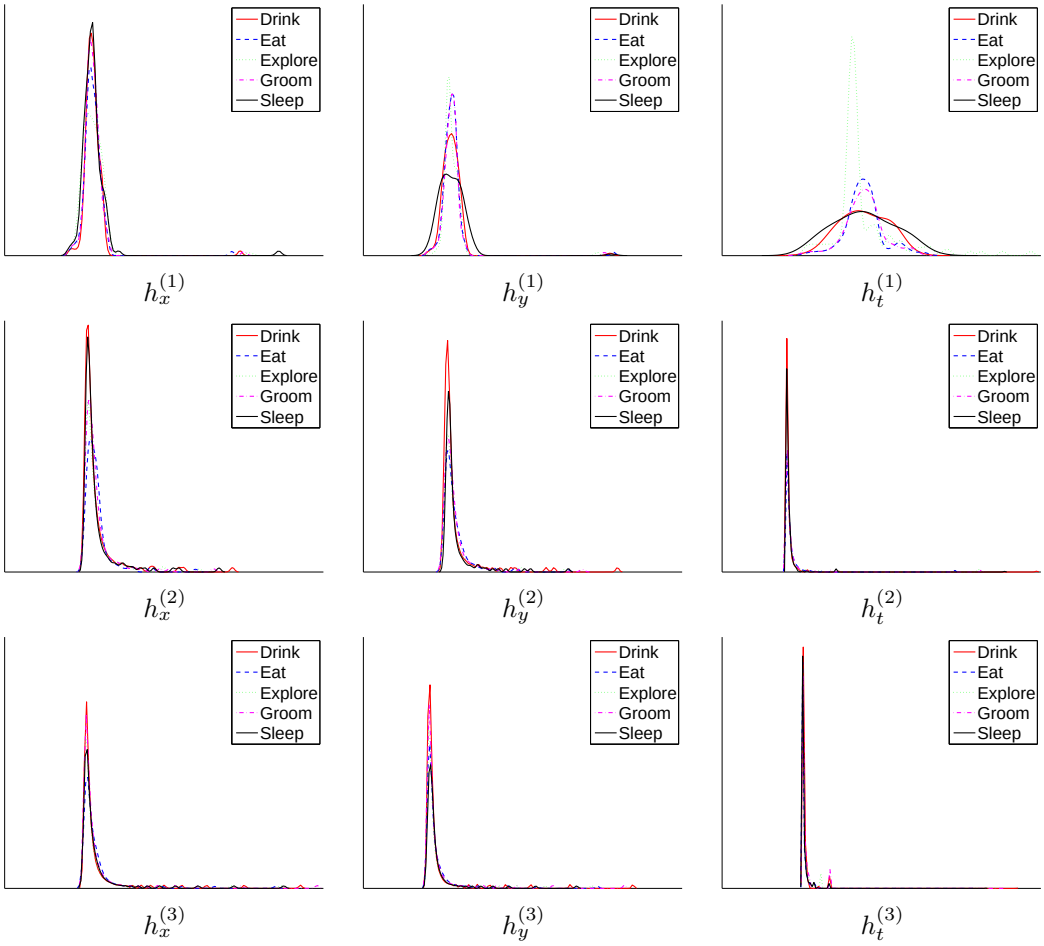



Figure 5.3: Mean histograms of gradient components for 5 actions at temporal scales $l = 1, 2, 3$.

## 5.4 1-NN Classification with Chi-square Distance

For classification, leave-one-out cross validation scheme with 1-NN classifier is employed. Leave-one-out cross validation scheme requires that at each classification run only one action is separated as the test set and the remaining actions are retained as the training set. Classification runs are repeated until each action in the dataset is used exactly once as the test set. Distance metric used in 1-NN classifier is $\chi^2$ distance. For this application, $\chi^2$ distance between actions $V_i$ and $V_j$ is given as,

$$\chi^2(V_i, V_j) = \frac{1}{3l_{\max}} \sum_k \sum_l \sum_m \frac{\left[h_{ik}^{(l)}(m) - h_{jk}^{(l)}(m)\right]^2}{h_{ik}^{(l)}(m) + h_{jk}^{(l)}(m)} \qquad \begin{array}{c} k \in \{x, y, t\}, \\ l = 1, \ldots, l_{\max}, \\ m = 1, \ldots, \ \# \text{ of bins.} \end{array} \tag{5.2}$$

For instance, to classify a given test action $V_{\mathrm{T}}$, its descriptor $f_{\mathrm{T}}$ is computed as explained in the sections 5.1, 5.2, and 5.3. $\chi^2$ distances between $f_{\mathrm{T}}$ and descriptors of remaining actions in the dataset are computed. Since 1-NN classifier is used, class of the action is assigned to the class of the nearest action in the remaining dataset.

# Chapter 6

# Sparse 3D Harris Corners and HOG-HOF based Approach

Sparse keypoints are special points in space-time, where significant amount of variation occurs in both space and time. Therefore, the vicinity of the keypoints exhibits rich information about the content of the video. To represent a given video in terms of keypoints, first one needs to localize those special points in the video. Afterwards, a keypoint is represented by a descriptor, which is extracted from the neighborhood of the keypoint. To represent a video clip in a compact way, extracted keypoint descriptors are labeled according to a previously computed codebook. The codebook is constructed by clustering the keypoint descriptors extracted from training data. At the end, a video is described by a histogram of keypoint types. This approach is the well known *bag-of-words* approach. In classification part, one just needs to classify the histograms associated with the videos.

In this chapter, action recognition for laboratory animals is addressed by applying the *bag-of-words* approach following the method of [10]. Locations of keypoints are determined by using a 3D extension of Harris corner detector

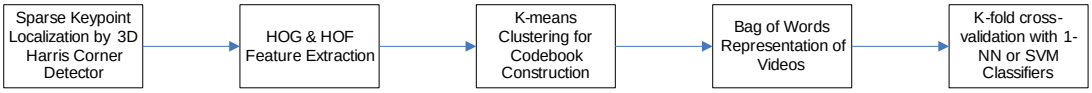



Figure 6.1: Main blocks of sparse keypoint based approach.

[19]. To describe the keypoints, histograms of oriented gradients (HOG) and histograms of oriented optical flows (HOF) are computed from the space-time vicinity of detected keypoints. To form the codebook, HOG-HOF descriptors associated with each localized keypoint are clustered by k-means clustering algorithm. One can assume that after clustering we have $K$ different types of keypoints, where $K$ is the number of clusters in k-means. For a given video with its keypoint features, each keypoint in the video is matched to the closest cluster center in the codebook. As a result, the given video is represented as a set of keypoint *types*. Histograming the *types* of the keypoints will generate a compact descriptor for the given video. Note that this procedure is a simple application of the *bag-of-words* approach, where the *bag* and *words* correspond to the given video and the keypoints, respectively. After these steps, classifying the histogram of keypoint types becomes equivalent to classifying the video. Nearest neighbor (1-NN) and support vector machine (SVM) classifiers are used to classify the histograms associated with the videos. In addition, due to high amount of pattern variations in the dataset K-fold cross validation scheme is carried out. Figure 6.1 summarizes main steps of the method. In the following sections, greater detail on each block will be given.

## 6.1 Sparse Keypoint Localization by 3D Harris Corner Detector

The idea of Harris corner detector is to locate 2D space points $(x, y)$, where gross amount of variation occurs in pixel intensities in both directions [62]. Multi-scale Harris detector is a generalized form of Harris detector over multiple spatial scales

by making use of Gaussian scale-space. Laptev and Lindaberg [19, 21, 63, 64] extended the multi-scale Harris detector to 3D case and proposed 3D Harris corner detector. Conventional and multi-scale Harris corner detectors will be explained as background information in the following subsections.

### 6.1.1 Harris Corner Detector

The image function $I(x, y)$ is assumed to be a mapping from spatial domain to the intensity domain, such that $I : \mathbb{R}^2 \mapsto \mathbb{R}$. If an image patch is taken over the area $(x, y)$ and shifted by $(\delta x, \delta y)$, weighted sum of squared difference (SSD) between the original and shifted patches gives a measure on the similarity of patches, $E$:

$$E(\delta x, \delta y) = \sum_x \sum_y W(x, y)\ [I(x + \delta x, y + \delta y) - I(x, y)]^2, \tag{6.1}$$

where $W(x, y)$ is the weighting function defined over the patch $(x, y)$. $W(x, y)$ can be constant or an isotropic 2D Gaussian. Assuming that the shift $(\delta x, \delta y)$ is small enough, $I(x + \delta x, y + \delta y)$ can be approximated by its Taylor expansion:

$$I(x + \delta x, y + \delta y) \approx I(x, y) + I_x(x, y)\delta x + I_y(x, y)\delta y, \tag{6.2}$$

where $I_x$ and $I_y$ are partial derivatives of $I(x, y)$ with respect to $x$ and $y$, respectively. Substituting (6.2) in (6.1), we obtain:

$$E(\delta x, \delta y) \approx \sum_x \sum_y W(x, y)\ [I_x(x, y)\delta x - I_y(x, y)\delta y]^2. \tag{6.3}$$

Rewriting equation (6.3) as a matrix equation:

$$E(\delta x, \delta y) \approx \begin{bmatrix} \delta x & \delta y \end{bmatrix} \mathbf{H}(x, y) \begin{bmatrix} \delta x \\ \delta y \end{bmatrix}, \tag{6.4}$$

where $\mathbf{H}(x, y)$ is a second moment matrix:

$$\mathbf{H}(x,y) = \sum_x \sum_y W(x,y) \begin{bmatrix} I_x^2(x,y) & I_x(x,y)I_y(x,y) \\ I_x(x,y)I_y(x,y) & I_y^2(x,y) \end{bmatrix}$$
$$= \begin{bmatrix} \sum_x \sum_y W(x,y)I_x^2(x,y) & \sum_x \sum_y W(x,y)I_x(x,y)I_y(x,y) \\ \sum_x \sum_y W(x,y)I_x(x,y)I_y(x,y) & \sum_x \sum_y W(x,y)I_y^2(x,y) \end{bmatrix}.$$

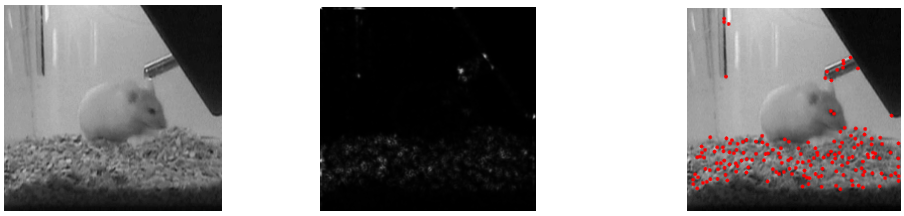
Note that elements of the matrix $\mathbf{H}$ are weighted sums of image derivatives over the patch $(x, y)$. Harris and Stephens claimed that there is a corner at the center of patch $(x, y)$, if the shift $(\delta x, \delta y)$ in any direction causes large increments in $E$. Eigenvalues of $\mathbf{H}$ characterize the response of $E$ to the shift $(\delta x, \delta y)$. Notice that $\mathbf{H}$ is symmetric and positive definite, therefore eigenvalues $\lambda_1$ and $\lambda_2$ are both positive. By further analysis of $\lambda_1$ and $\lambda_2$, one can deduce that:

- If both $\lambda_1$ and $\lambda_2$ are very small i. e., $\lambda_1 \approx 0$ and $\lambda_2 \approx 0$, the pixel at the center of patch $(x, y)$ doesn't have any corner, furthermore it is a flat region.
- If one of the eigenvalues is very small and the other is significantly large, such as $\lambda_1 \gg \lambda_2$ and $\lambda_2 \approx 0$, then the pixel at the center of a patch is an edge.
- If both $\lambda_1$ and $\lambda_2$ are large, then the pixel is a corner point.

Harris and Stephens defined a *corner response* function $R(x, y)$ based on determinant and trace of $\mathbf{H}$ to avoid high computational cost of explicit eigenvalue decomposition:

$$R(x,y) = \det(\mathbf{H}(x,y)) - k\ \mathrm{tr}^2(\mathbf{H}(x,y))$$
$$= \lambda_1\lambda_2 - k\ (\lambda_1 + \lambda_2)^2.$$

In order to locate corner points, one needs to find positive local maxima of $R(x, y)$. Consider the ratios of eigenvalues $\alpha = \lambda_1/\lambda_2$ for a local maxima of



(a) Original image. (b) *Corner response* function. (c) Detected Harris corners.

Figure 6.2: Harris corner detection example on a real world image.

$R(x, y)$. Relation between $k$ and $\alpha$ is given as:

$$k \leq \frac{\alpha}{(1+\alpha)^2} \ .$$

If we let $k = 0.25$, then local maxima of $R(x, y)$ will occur at $\alpha = 1$ i. e., $\lambda_1 = \lambda_2$. Considering that real world images probably don't have too many ideally isotropic corner points, it is better to allow higher values for the ratio of eigenvalues, so that local maxima of $R(x, y)$ will correspond to points with more elongated shape. Commonly $k$ is set to 0.04 in the literature, which results in $\alpha < 23$. Harris corner detector is applied to a real world image shown in Figure 6.2(a). In Figure 6.2(b) and 6.2(c), *corner response* function and detected corner points are illustrated, respectively.

### 6.1.2 Multi-scale Generalization of Harris Corner Detector

The idea behind the multi-scale Harris corner detector is to locate corners at different spatial scales. Consider the image illustrated in Figure 6.3. Notice that leaves of the trees in the image introduce fine level details, where windows of the building give rise to coarser details. In order to detect corners at multiple spatial levels, one needs to construct *scale-space* representation of the image $I(x, y)$ [65, 66, 67, 68], $L : \mathbb{R}^2 \times \mathbb{R}_+ \mapsto \mathbb{R}$,

$$L(x, y, \sigma_l^2) = g_l(x, y, \sigma_l^2) * I(x, y),$$

Figure 6.3: An image with different levels of detail.

where $(*)$ is convolution operator and $g(x, y, \sigma_l^2)$ is a zero-mean 2D Gaussian with variance $\sigma_l^2$,

$$g_l(x, y, \sigma_l^2) = \frac{1}{2\pi\sigma_l^2}\ e^{-(x^2+y^2)/2\sigma_l^2}.$$

Similar to conventional Harris corner detector, a second moment matrix for a given scale $\sigma_l^2$ integrated over a Gaussian window $g_i(x, y, \sigma_i^2)$ [69, 70, 71] is defined as follows,

$$\boldsymbol{\mu_s}(x, y, \sigma_l^2, \sigma_i^2) = g_i(x, y, \sigma_i^2) * \begin{bmatrix} L_x^2 & L_x L_y \\ L_x L_y & L_y^2 \end{bmatrix}, \tag{6.5}$$

where $L_x$ and $L_y$ are partial derivatives of $L$ with respect to $x$ and $y$, respectively,

$$L_x = \partial_x(\ g_l(x, y, \sigma_l^2) * I(x, y)\ ),$$
$$L_y = \partial_y(\ g_l(x, y, \sigma_l^2) * I(x, y)\ ).$$

Due to interchangeability of differentiation and convolution operators, $L_x$ and $L_y$ can be computed by convolving $I(x, y)$ with *Gaussian derivatives*,

$$L_x = \partial_x(g_l(x, y, \sigma_l^2)) * I(x, y),$$
$$L_y = \partial_y(g_l(x, y, \sigma_l^2)) * I(x, y).$$

Integration and scale-space Gaussian kernel variances are proportional to each other with a factor $\zeta_s$, $\sigma_i^2 = \zeta_s \sigma_l^2$. In the literature $\zeta_s$ is selected from the interval $[2, 4]$.

Notice that $\boldsymbol{\mu_s}$, $g_i(x, y, \sigma_i^2)$, and $(L_x, L_y)$ are analogous to $\mathbf{H}$, $W(x, y)$, and $(I_x, I_y)$ in the conventional Harris corner detector, respectively. Similarly, by

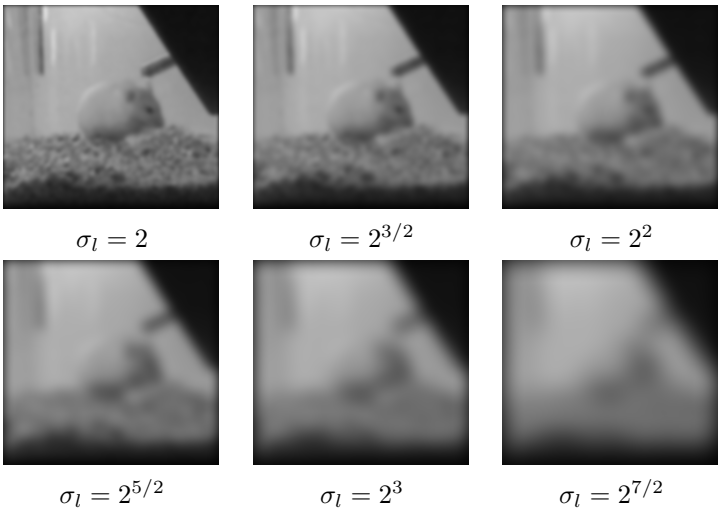



Figure 6.4: Scale-space representation of an image for 6 levels of scale.

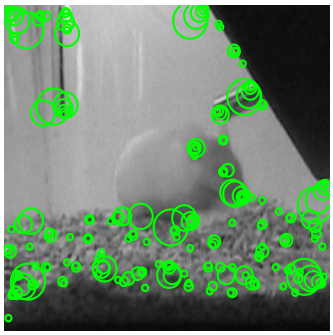


Figure 6.5: Harris Corners detected at multiple spatial scales. Detected corners are at the centers of the circles. Size of the circles indicate the spatial scale at which the corner is detected.

analyzing eigenvalues of multi-scale second moment matrix $\boldsymbol{\mu_s}$, same inferences in subsection 6.1.1 on the cornerness of a point can be made. *Corner response* function in multi-scale Harris corner detector turns out to be,

$$\mathbf{M_s}(x, y, \sigma_l^2, \sigma_i^2) = \det(\boldsymbol{\mu_s}(x, y, \sigma_l^2, \sigma_i^2)) - k \ \mathrm{tr}^2(\boldsymbol{\mu_s}(x, y, \sigma_l^2, \sigma_i^2)).$$

Corners with different size in images can be detected by determining local maxima of $\mathbf{M_s}$ computed at differing values of scales $\sigma_l^2$. Scale-space representation of the image in Figure 6.2(a) is illustrated in Figure 6.4. Detected Harris corners at different spatial scales are shown in Figure 6.5.

### 6.1.3 3D Harris Corner Detector

Laptev and Lindeberg extended the idea of the multi-scale Harris corner detector to spatio-temporal dimensions and proposed 3D Harris corner detector [19, 21, 63, 64, 72]. Analogous to the multi-scale Harris corner detector, 3D Harris corner detector operator gives significant amount of response to points, which exhibit space-time discontinuities.

Consider an image sequence or a video clip as a mapping from spatio-temporal domain to pixel intensity domain, such that $V : \mathbb{R}^2 \times \mathbb{R} \mapsto \mathbb{R}$. Scale-space representation of $V$ at level $l$ is generated by convolution of $V$ with a zero-mean Gaussian kernel with spatial variance $\sigma_l^2$ and temporal variance $\tau_l^2$,

$$L(x, y, t, \sigma_l^2, \tau_l^2) = g(x, y, t, \sigma_l^2, \tau_l^2) * V(x, y, t), \tag{6.6}$$

where $g$ is a separable Gaussian,

$$g(x, y, t, \sigma_l^2, \tau_l^2) = \frac{1}{2\pi\sigma_l^2} \exp\left(\frac{-(x^2+y^2)}{2\sigma_l^2}\right) \frac{1}{\sqrt{2\pi}\tau_l} \exp\left(\frac{-t^2}{2\tau_l^2}\right).$$

To detect space-time corners, $2 \times 2$ second moment matrix in the multi-scale Harris corner detector is extended to a $3 \times 3$ second moment matrix formed of spatial and temporal derivatives of $L$. In a similar way, 2D integrating Gaussian kernel is replaced by a 3D Gaussian kernel $g_i(x, y, t, \sigma_i^2, \tau_i^2)$,

$$\boldsymbol{\mu_{st}} = g_i(x, y, t, \sigma_i^2, \tau_i^2) * \begin{bmatrix} L_x^2 & L_x L_y & L_x L_t \\ L_x L_y & L_y^2 & L_y L_t \\ L_x L_t & L_y L_t & L_t^2 \end{bmatrix}, \tag{6.7}$$

where $L_x$, $L_y$ and $L_t$ are first-order derivatives of $L$ with respect to space-time dimensions,

$$L_x = \partial_x(g_l(x, y, t, \sigma_l^2, \tau_l^2)) * V(x, y, t),$$
$$L_y = \partial_y(g_l(x, y, t, \sigma_l^2, \tau_l^2)) * V(x, y, t),$$
$$L_t = \partial_t(g_l(x, y, t, \sigma_l^2, \tau_l^2)) * V(x, y, t).$$

Scales of scale-space and integrating Gaussian kernels are related to each other by a factor, $\sigma_i^2 = \zeta_{sp}\sigma_l^2$ and $\tau_i^2 = \zeta_{sp}\tau_l^2$.

Large values for eigenvalues of $\boldsymbol{\mu_{st}}$, $\lambda_1$, $\lambda_2$, and $\lambda_3$, indicate presence of a space-time corner. Laptev and Lindeberg simply defined spatio-temporal *corner response* function by considering determinant and trace of $\boldsymbol{\mu_{st}}$ as in the multi-scale Harris corner detector,

$$\begin{aligned} \mathbf{M_{st}} &= \det(\boldsymbol{\mu_{st}}) - k\ \mathrm{tr}^3(\boldsymbol{\mu_{st}}) \\ &= \lambda_1\lambda_2\lambda_3 - k(\lambda_1 + \lambda_2 + \lambda_3)^3. \end{aligned} \tag{6.8}$$

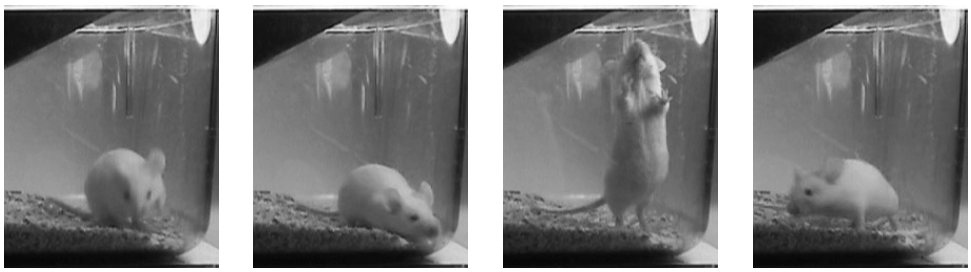
Space-time corners at the spatial scale $\sigma_l$ and the temporal scale $\tau_l$ are located at positive local maxima of $\mathbf{M_{st}}$. To set $k$ to a reasonable value, ratios of eigenvalues, $\alpha = \lambda_2/\lambda_1$ and $\beta = \lambda_3/\lambda_1$ are analyzed as in the 2D case. For the positive local maxima of $\mathbf{M_{st}}$ i. e., $\mathbf{M_{st}} \geq 0$, replacing the ratios of eigenvalues in (6.8),

$$0 \geq \lambda_1^3(\alpha\beta - k(1 + \alpha + \beta)^3).$$

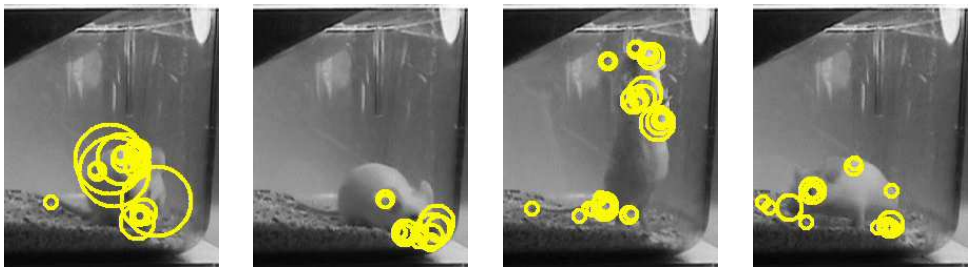
Therefore, $k$ is related to $\alpha$ and $\beta$ through the following inequality,

$$k \leq \frac{\alpha\beta}{(1 + \alpha + \beta)^3}\ .$$

Allowing $\alpha < 23$ and $\beta < 23$ results in $k \approx 0.005$. To detect 3D corners at multiple spatial and temporal scales, $\mathbf{M_{st}}$ is constructed for a set of scales $\sigma_l = 2^{(i+1)/2}, i = 1, 2, \ldots, n$ and $\tau_l = 2^{j/2}, j = 1, 2, \ldots, m$ and local maxima are located. In Figure 6.6(a), some sample images from an image sequence are illustrated. Detected spatio-temporal corners are shown in Figure 6.6(b). Similar to Figure 6.5, locations of detected corners are the centers of the circles. Radius of the circles points out the scale at which the corner is detected.



(a) Sample images from an image sequence.



(b) Detected 3D corners.

Figure 6.6: 3D corner detection for an image sequence with multiple spatial and temporal scales.

## 6.2 HOG & HOF Feature Extraction

Histograms of oriented gradients (HOG) and histograms of optical flow (HOF) are simple but effective descriptors commonly used in computer vision for feature matching, recognition and object detection. The central idea behind HOG and HOF descriptors is to divide an image or video patch into smaller patches, compute histograms of *orientations* of gradient or optical flow vectors from each small patch and lastly concatenate all of the histograms into one descriptor for the given patch. They are highly inspired from Scale Invariant Feature Transform (SIFT) proposed by Lowe [25]. Lowe discovered the discriminative power of localized gradient orientation histograms and built the SIFT descriptor on it. The key point in SIFT is to compute histograms of *orientations*, not the *amplitudes* of gradient vectors, since orientation is much more important than amplitude in characterizing the spatial appearance as suggested by Zelnik-Manor and Irani [73]. Another key idea in SIFT is to compute localized histograms over a given patch to avoid disregarding positional dependency or *structure* of the visual units in the patch.

There has been a number of varieties of HOG descriptors proposed by different researchers. Dalal and Triggs described a densely sampled HOG descriptor in order to detect humans in images [7]. Laptev and Perez used a spatial HOG descriptor in conjunction with a HOF descriptor to detect actions in movies [74]. Kläser, Marszałek, and Schmid extended HOG descriptor to spatio-temporal domain for usage in action recognition [20].

HOF descriptor is very similar to HOG descriptor except that it is based on optical flow vectors instead of gradient vectors. In the HOF case, optical flow vectors between two consecutive frames are estimated by the method proposed by Lucas and Kanade [75] and orientations of optical flow vectors are quantized in a similar fashion to HOG descriptor. HOF descriptors are frequently used among computer vision society for human detection in videos and human action recognition [32, 35, 76, 77].

In this chapter following the method of [10], to describe each detected 3D Harris corner point HOG and HOF descriptors are extracted from the spatio-temporal vicinity (*cuboid*) of the detected corner. Since HOF descriptor is very similar to HOG, details of descriptor computation will be given for only HOG.

### 6.2.1 HOG Descriptor Computation

Consider a 3D Harris corner at point $(x_c, y_c, t_c)$ in the video space-time volume detected at the spatial and the temporal scales, $\sigma_l$ and $\tau_l$, respectively. Extent of the *cuboid* $\Omega$, $(\Delta_x, \Delta_y, \Delta_t)$, is selected as an integral multiple of the spatial and the temporal detection scales, such that $\Delta_x, \Delta_y = 2\zeta\sigma_l$ and $\Delta_t = 2\zeta\tau_l$. $\zeta$ is a scale factor and usually selected empirically. The *cuboid* $\Omega$ is defined as,

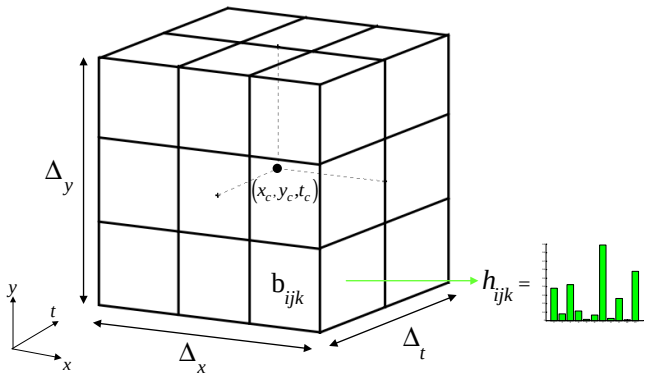



Figure 6.7: Illustration of HOG and HOF computation.

$\forall(x_i, y_i, t_i) \in \Omega$, satisfying,

$$x_c - \zeta\sigma_l \leq x_i \leq x_c + \zeta\sigma_l,$$

$$y_c - \zeta\sigma_l \leq y_i \leq y_c + \zeta\sigma_l,$$

$$t_c - \zeta\tau_l \leq t_i \leq t_c + \zeta\tau_l.$$

An example of a corner point and its surrounding *cuboid* are illustrated in Figure 6.7.

Similar to the state of the art SIFT descriptor, each *cuboid* $\Omega$ is partitioned into $\eta_x \times \eta_y \times \eta_t$ grid of blocks. For instance, the *cuboid* shown in Figure 6.7 is partitioned into $3 \times 3 \times 2$ blocks. Size of each block is equal and for a general block it is given as $(\Delta_{xb}, \Delta_{yb}, \Delta_{tb}) = (2\zeta\sigma_l/\eta_x, 2\zeta\sigma_l/\eta_y, 2\zeta\tau_l/\eta_t)$. Therefore, a block is a small voxel in the video space-time volume consisting of $2\zeta\tau_l/\eta_t$ image patches with size $(2\zeta\sigma_l/\eta_x, 2\zeta\sigma_l/\eta_y)$. From each block $b_{ijk}$, gradient orientation histograms $h_{ijk}$ are computed as shown in Figure 6.7. For a given block $b_{ijk}$ for each image patch in the block, a gradient vector field $F_g : \mathbb{R}^2 \mapsto \mathbb{R}^2$ is computed spatially. Thus, each space-time point $(x, y, t)$ within the block $b_{ijk}$ is associated to a 2D dimensional gradient vector $\vec{G} = [g_x \; g_y]^T$. To construct $h_{ijk}$, orientations of all gradient vectors are quantized by projecting them onto the unit vectors $\vec{\mathbf{u}}_\theta$, where $\theta \in \{0, 90, 180, 270\}$,

$$h_{ijk}(\theta) = \sum_{m \; \in \; b_{ijk}} \Gamma(\vec{\mathbf{u}}_\theta \cdot \vec{G}_m),$$

where $\Gamma$ function is defined as,

$$\Gamma(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}.$$

Lastly, $h_{ijk}$ is normalized such that its $L_2$ norm equals to 1. Combining normalized histograms $h_{ijk}$ from all blocks into one final descriptor $f_{\text{HOG}}$ by simple concatenation operation results in HOG descriptor,

$$f_{\text{HOG}} = \big\{ h_{ijk} \mid \forall i \in \{1, \ldots, \eta_x\}, \forall j \in \{1, \ldots, \eta_y\} \text{ and } \forall k \in \{1, \ldots, \eta_t\} \big\}.$$

Resultant $f_{\text{HOG}}$ descriptor is a $1 \times 4\eta_x\eta_y\eta_t$ vector.

#### Spatial Gradient Computation for HOG Descriptor

To compute gradient vector field $F_g : \mathbb{R}^2 \mapsto \mathbb{R}^2$ for each image patch within a block, each patch is first smoothed by convolution with a 2D Gaussian kernel $g_{sp}$, then convolved with a 1D filter $h = [-1\ 0\ 1]$. Note that the filter $h$ is an approximation to partial derivative operator. Components of gradient vector field are given as,

$$G_x(\cdot;\cdot;t) = (b_{ijk}(\cdot;\cdot;t) * g_{sp}(\cdot;\cdot;\sigma_{sp})) * h,$$
$$G_y(\cdot;\cdot;t) = (b_{ijk}(\cdot;\cdot;t) * g_{sp}(\cdot;\cdot;\sigma_{sp})) * h^T,$$

where $b_{ijk}(\cdot;\cdot;t)$ is the image patch at time $t$ in the block $b_{ijk}$. $G_x(\cdot;\cdot;t)$ and $G_y(\cdot;\cdot;t)$ are horizontal and vertical components of gradient vector field for the image patch $b_{ijk}(\cdot;\cdot;t)$. Smoothing Gaussian kernel has zero mean and standard deviation $\sigma_{sp} = 1$. An example of gradient vector field estimation for a real world image is shown in Figure 6.8.

### 6.2.2 HOF Descriptor Computation

HOF descriptors are computed in a similar fashion to HOG descriptors. Major difference is that optical flow vector field is used instead of gradient vector field

(a) Original real world image. (b) Gradient vectors overlaid on original image.

Figure 6.8: An example of spatial gradient estimation.

in HOF. In addition, number of orientation bins in $h_{ijk}$ is 5 in contrast to HOG case. In HOF case, there is an additional bin in $h_{ijk}$ representing *no motion* i. e., magnitude of the optical flow vector is zero. Therefore, resultant HOF descriptor is a $1 \times 5\eta_x\eta_y\eta_t$ vector. In the following part, Lucas-Kanade optical flow estimation is explained in detail.

### Lucas-Kanade Optical Flow Method

Optical flow estimation is used to determine motion between two image frames at time $t$ and $t + \delta t$. Fundamental assumption in Lucas-Kanade optical flow method is that all pixels within a local image patch are subjected to same flow. Consider an image patch centered at location $(x, y)$. Constant local flow assumption applies as sufficiently small translations of the patch in space and time, $(\delta x, \delta y, \delta t)$, do not introduce any changes on pixel intensities i. e., $I(x, y, t) = I(x + \delta x, y + \delta y, t + \delta t)$ for all the pixels in the patch. Approximating the translated image function by its Taylor expansion around the point $(x, y, t)$,

$$I(x + \delta x, t + \delta y, t + \delta t) \approx I(x, y, t) + I_x\delta x + I_y\delta y + I_t\delta t, \tag{6.9}$$

where $I_x, I_y$, and $I_t$ are partial derivatives of $I$ with respect to $x, y$, and $t$, respectively. Assuming that the patch centered at $(x, y)$ and translated patch are equal in pixel intensities:

$$I_x\delta x + I_y\delta y + I_t\delta t \approx 0. \tag{6.10}$$

Dividing left-hand side of (6.10) by $\delta t$:

$$I_x \frac{\delta x}{\delta t} + I_y \frac{\delta y}{\delta t} + I_t \approx 0,$$

$$I_x v_x + I_y v_y + I_t \approx 0,$$

where $v_x$ and $v_y$ are components of optical flow along $x$ and $y$ axes. Transforming optical flow equation into matrix form,

$$\nabla I \cdot v^T \approx -I_t, \tag{6.11}$$

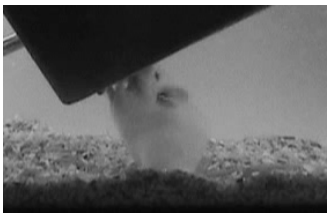
where $v = [v_x \; v_y]$. Constant local flow assumption requires that (6.11) must hold for all pixels of the patch. Therefore, a set of linear equations can be constructed in the form,

$$\begin{bmatrix} \nabla I_1 \\ \nabla I_2 \\ \vdots \\ \nabla I_n \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} -I_{t1} \\ -I_{t2} \\ \vdots \\ -I_{tn} \end{bmatrix}, \tag{6.12}$$

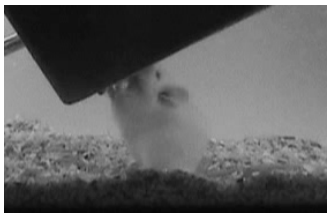
where $n$ is the number of pixels within the patch. Equation (6.12) obeys the general form of system of linear equations i. e., $Ax = b$. Solution for $x$ is given as $x = (A^T A)^{-1}(A^T b)$ using pseudo-inverse of $A$. Applying same technique on (6.12) $[v_x \; v_y]^T$ can be solved as,

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} I_{xi}^2 & \sum_{i=1}^{n} I_{xi} I_{yi} \\ \sum_{i=1}^{n} I_{xi} I_{yi} & \sum_{i=1}^{n} I_{yi}^2 \end{bmatrix}^{-1} \begin{bmatrix} \sum_{i=1}^{n} -I_{xi} I_{ti} \\ \sum_{i=1}^{n} -I_{yi} I_{ti} \end{bmatrix}. \tag{6.13}$$

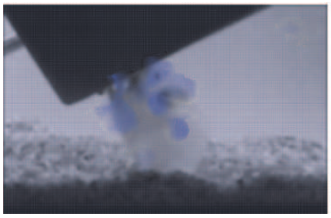
To give more importance to the pixels closer to patch center $(x, y)$ in the solution of $[v_x \; v_y]^T$, partial derivatives $I_x, I_y$, and $I_t$ can be windowed by a weighting function such as a 2D Gaussian. An example of Lucas-Kanade optical flow estimation is illustrated in Figure 6.9.



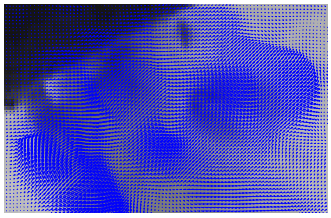
(a) First image



(b) Second image



(c) Lucas-Kanade optical flow vectors



(d) A close look to optical flow vectors

Figure 6.9: An example of Lucas-Kanade optical flow estimation.

## 6.3 K-means Clustering for Codebook Construction

To describe a given video by *bag of words* model, a previously constructed codebook is needed. In this chapter, such a codebook is a set of principal descriptors, *codewords*, where one can describe a video in terms of *codewords*. Construction of required codebook is carried out by k-means clustering algorithm in a very similar way to section 4.3. Same as section 4.3, a descriptor set $Q$ is formed by extracting $q$ descriptors from training videos. The only difference is that descriptors are computed by using 3D Harris corner detector and HOG & HOF process. Resultant codebook is in the form $\mathcal{E} = \{e_1, e_2, \ldots, e_K\}$, where each element $e_k$ is a *codeword*.

## 6.4 Bag of Words Representation of Videos

Bag-of-words is a simple, compact yet effective model widely used in image and video representation [10, 22, 60, 78, 79, 80]. In bag-of-words model, a given video with its keypoint descriptors is expressed as a collection of words from a

previously constructed codebook. In this chapter, codebook is constructed by k-means clustering of the descriptor set $Q$ as explained in previous section.

To represent any given video, first descriptors, $\{f_i \mid i = 1, \ldots, n\}$, are computed for each detected 3D Harris corner in the video volume. Here, $n$ is the number of detected Harris corners. Each descriptor $f_i$ is assigned to the nearest codeword in the codebook,

$$\upsilon_i = \underset{\mathbf{j}}{\operatorname{argmin}} \|f_i - e_j\|^2 \text{ for } j = 1, \ldots, K \text{ and } i = 1, \ldots, n,$$

where $\upsilon_i$ is the *type* of descriptor $f_i$ and $e_j$ is the $j^{th}$ codeword in the codebook $\mathcal{E}$. After all descriptors are assigned, a histogram, $\mathcal{A}$, with $K$ bins, describing the distribution of codewords in the video is computed. Simply each histogram bin indicates the number of occurrences of associated codeword in the video. Lastly, $\mathcal{A}$ is normalized such that it sums to 1. Notice that bag-of-words model does not take into account geometrical and spatial relations between descriptors, thus resulting in an unordered representation.

## 6.5 K-fold Cross-Validation Classification with 1-NN or SVM Classifier

In this chapter similar to section 4.5, K-fold cross validation is employed to measure classification performance. At each classification run in K-fold cross validation, either a 1-NN classifier or radial basis support vector machine classifier [81] is learned using the histograms, $\{\mathcal{A}_i \mid i = 1, \ldots, \#$ of training videos$\}$. Since action recognition in this thesis is a multi-class recognition problem, one-against all SVM classifiers are trained for each behavior using PRTools toolbox of Information and Communication Theory Group in Delft University of Technology [82].

# Chapter 7

# Experimental Results

For the evaluation and comparison of action recognition methods discussed in previous chapters, MATLAB implementations of these methods are tested on a publicly available mice action dataset recorded by Computer Vision group at University of California San Diego (UCSD) [22]. In this chapter, we test the performance of proposed action recognition framework by utilizing the method described in Chapter 2 as the first stage and one of the methods explained in Chapters 3, 4, 5, and 6 as the second stage. We analyze weakness and strength of each method according to its recognition performance. In addition we compare recognition results achieved by each method with those of Dollar *et. al* [22] and Jhuang *et. al* [38] who worked on the same dataset. In this chapter, first we describe the dataset used. Then, we demonstrate the results for the first stage in our framework. Rest of the chapter is composed of sections reserved for the second stage methods and comparisons of those with each other and with related studies. In the following sections for convenience we refer the methods in Chapters 3, 4, 5, and 6 as **DWT-HMM**, **SHOG**, **3DGrads**, and **SparseHOG**, respectively.

| Action class | no. of occurence |
|---|---|
| `drinking` | 17 |
| `eating` | 144 |
| `exploring` | 183 |
| `grooming` | 40 |
| `sleeping` | 46 |

Table 7.1: Distribution of action classes in UCSD mice action dataset.

As it will be explained in next section, occurrences of action classes in the dataset are not same. Therefore, we evaluate recognition performances of methods in two different aspects, overall recognition rate (ORR) and mean of individual behavior recognition rate (MD). ORR is simply defined as,

$$\text{ORR} = \frac{\text{no. of correctly classified video clips}}{\text{no. of video clips in the dataset}}$$

On the other hand, MD is given as,

$$\text{MD} = \frac{1}{|\mathcal{L}|}\sum_{i=1}^{|\mathcal{L}|}\frac{\text{no. of correctly classified video clips with class } \ell_i}{\text{no. of video clips in the dataset with class } \ell_i},$$

where $\mathcal{L}$ is the set of class labels.

## 7.1 Dataset

UCSD mice action dataset consists of short video clips manually cut from seven fifteen–minute videos of the same mouse recorded at different times of a day. In this dataset, there are five action classes, namely `drinking, eating, exploring, grooming`, and `sleeping`. Occurrence of each action class in the dataset is depicted in Table 7.1. Notice that the dataset mostly consists of `eating` and `exploring` action classes. On the other hand, `grooming, sleeping`, and `drinking` action classes comprise approximately 10%, 10%, and 4% of the dataset, respectively.

Although there are five classes in the dataset, we notice serious pattern variations among intra-class behaviors. For instance, during `grooming` action the

mouse may lick its forefeet in a rapid fashion or lick other body parts such as its rearfeet and tail very slowly. Furthermore, if the mouse cleans its head with its forefeet by exhibiting large and quasi-periodic arm movements, then this action is again counted as `grooming`. Similarly, climbing onto walls of cage, wandering inside the cage, digging the litter in the cage, looking around without body movement and combinations of these actions are all annotated as `exploring` action. Therefore, mouse actions turn out to be multimodal patterns.

It is worth to say a few words on properties of the video clips. Each video clip corresponds to one action and lasts about 10–15 seconds. Resolution differs from clip to clip. However, mouse is at the spatial center of the frame most of the time. Average mouse diameter is 120 pixels, but it can stretch or compress significantly due to its highly deformable body.

## 7.2 Experimental Results for the First Stage

We apply the first stage of our framework to UCSD mice action dataset in order to eliminate `sleeping` action. First, dataset is split into two sets as training and testing sets. Using the action videos in training sets Gaussian distributions, $G_S$ and $G_{NS}$ are learned for `sleeping` and `non-sleeping` actions as described in Chapter 2, respectively. In order to estimate mean and variance of $G_S$, we exploit 5 `sleeping` videos. On the other hand, parameters of $G_{NS}$ are estimated from 2, 14, 19, and 4 instances of `drinking, eating, exploring`, and `grooming` actions, respectively. In fact, training set corresponds to 10% of the whole dataset. In Figure 7.1, we plot $G_S$ and $G_{NS}$ learned from training set videos. 100% classification accuracy at discriminating `sleeping` from other action classes proves our assumption that `sleeping` can be classified by just examining the spanned area during behavior.

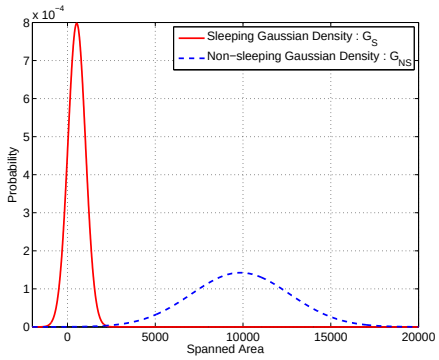



Figure 7.1: Gaussian distributions, $G_S$ and $G_{NS}$ learned from real data.

In our action recognition framework, first stage is common for all the second stage methods. Therefore, classification accuracy of `sleeping` action class is reported as 100% in experimental results of second stage methods.

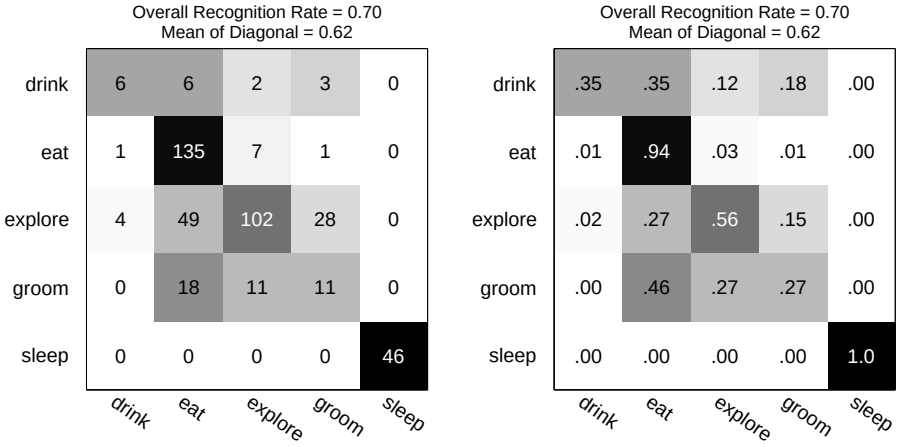
## 7.3 Experimental Results for DWT and HMM based Method

In this section, as second stage of our action recognition framework, **DWT-HMM** method, explained in Chapter 3, is used and tested on UCSD mice action dataset. Recognition performance is illustrated by confusion matrices which are commonly used for visualization in multi-class cases. Each row $i$ in a confusion matrix represents the number of class predictions for an action whose actual class was $i$. In this study, normalized confusion matrices are formed by normalizing the rows of the confusion matrices, such that sum of each row equals 1.

Since we employ "leave one out cross validation" scheme for **DWT-HMM** method in order to classify a test action $V_{\mathrm{T}}$, we first train one HMM model for each video in the remaining of dataset as explained in Chapter 3. Then, we assign the class of $V_{\mathrm{T}}$, $\ell_{\mathrm{T}}$ to the class of HMM model, which gives the highest probability that observation sequence $\boldsymbol{O}_{\mathrm{T}}$ associated with $V_{\mathrm{T}}$ is generated by that model.

Best recognition performance achieved by **DWT-HMM** method is shown in Figure 7.2. As seen from Figure 7.2, only `eating` action is successfully discriminated from other action classes. 100% success rate for `sleeping` action is inherited from the first stage. Remaining actions are heavily confused with each other. We believe that **DWT-HMM** method is able to successfully classify `eating` action, because most of ASIs associated with `eating` actions have similar appearance i. e., `eating` action turns out to be unimodal. On the other hand, intra-class variances of ASIs generated by other actions are quite high. We deduce that there are no consistent patterns in other actions to be modeled by HMMs. In other words ASIs generated by other action classes are too random even for HMMs. Besides, it is our belief that in most cases $T$ is too short for training reliable HMMs. Call that the length of observation sequences was given as $T = N_{SI}M_{SI}$ in subsection 3.3.1. One may increase $N_{SI}$ and $M_{SI}$ by dividing bounding box image $BB$ into smaller subimages. However, decreasing the size of subimages gradually disregards spatial relation between pixels of $BB$, which will eventually decrease the quality of observation symbols and introduce inaccuracies in estimation of HMM parameters. Therefore, we believe that there is a lower bound on the size of subimages, which imposes an upper bound on the length of observation sequences. Considering experimental results performed by **DWT-HMM**, that upper bound is not sufficient to train reliable HMMs in our case.

Although we don't believe that parameter tuning will improve recognition rates of **DWT-HMM**, we search for the optimal parameters. There are five important parameters that can affect the performance of **DWT-HMM** method. First and the most important one is the number of states in HMMs, $N$. Number of histogram bins $m_{SI}$ determines the dimension of observation symbols. $\zeta_{SI}$ is the ratio of overlap between consecutive subimages in raster scanning $BB$ image (see subsection 3.3.1). $N_{SI}$ and $M_{SI}$ are the number of overlapping subimages along horizontal and vertical dimensions. $N_{SI}$ and $M_{SI}$ also determines the




(a) Confusion matrix. (b) Normalized confusion matrix.

Figure 7.2: Best recognition performance for **DWT-HMM** method.

length of observation sequence. Last parameter is the filterbank (FB) used in ASI formation. Confusion matrices shown in Figure 7.2 are the result of **DWT-HMM** method with $N = 4, m_{SI} = 5, \zeta_{SI} = 0.75, N_{SI} = 8,\ M_{SI} = 8$ and using filterbank with Daubechies $4^{th}$ order decomposition filters.

We first analyze the effect of number of states $N$ on recognition performance. Values of other parameters are fixed as $m_{SI} = 5, N_{SI} = 8, M_{SI} = 8, \zeta_{SI} = 0.75$ and we employ $4^{th}$ order Daubechies decomposition filters. We tried the following values; $N = 3, 4, 5, 6, 7$. Figure 7.3 shows the effect of $N$ on ORR and MD. We observe that ORR and MD remain almost constant up to 4 states. Further increasing the number of states causes ORR and MD to decrease. Call that the number of states is selected as the number of clusters in clustering observation symbols during training HMMs. Our intuition is that increasing $N$ beyond 5 causes some of the clusters to be nearly empty or very similar to each other. Two similar clusters mean that the associated states can not be distinguished. On the other hand, an empty cluster introduces inaccuracies in training state transition probabilities of the associated state.

Similarly, the effect of $N_{SI}$ and $M_{SI}$ on overall recognition performance is examined. We keep the values of other parameters constant at $N = 4, m_{SI} =$

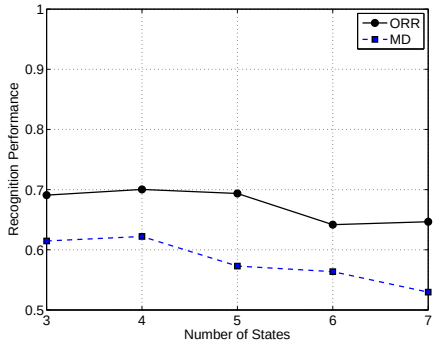



Figure 7.3: Variations of ORR and MD with respect to number of states $N$.

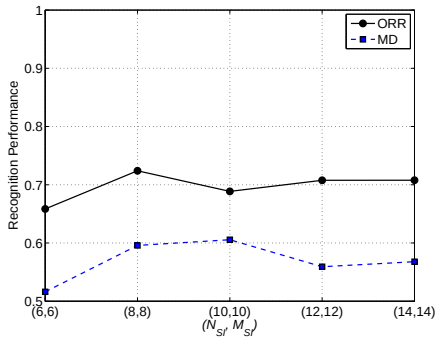



Figure 7.4: Variations of ORR and MD with respect to $N_{SI}$ and $M_{SI}$.

$5, \zeta_{SI} = 0.75$ and test the following values of $N_{SI} = M_{SI} \in \{6, 8, 10, 12, 14\}$. Resultant ORR and MD with different values of $N_{SI}$ and $M_{SI}$ are plotted in Figure 7.4. Highest ORR and MD are observed when $N_{SI}$ and $M_{SI}$ are 8. Further increasing $N_{SI}$ and $M_{SI}$ decreases MD, which means that individual recognition rates for action classes decrease. Constancy of ORR is due to high recognition rates of `eating` action.

In order to find out the effect of overlap ratio $\zeta_{SI}$ on ORR and MD, values of $N, m_{SI}, N_{SI}$ and $M_{SI}$ are fixed to 4, 5, 8, and 8, respectively. We try four different overlap ratios, $\zeta_{SI} = 0, 0.25, 0.50, 0.75$. The effect of $\zeta_{SI}$ is illustrated in Figure 7.5. Increasing $\zeta_{SI}$ forces **DWT-HMM** to stress spatial relations between subimages of $BB$ in HMM training, so that we observe the highest ORR and MD at $\zeta_{SI} = 0.75$.

For analyzing number of histogram bins we test the following values of $m_{SI}$: $m_{SI} = 3, 4, 5, 6, 7, 8, 9$. Other parameters are fixed at $N = 4, N_{SI} = 8, M_{SI} = 8$,

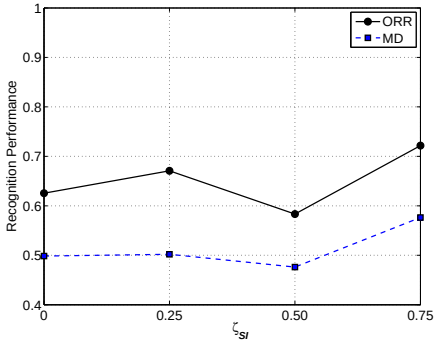



Figure 7.5: Variations of ORR and MD with respect to overlap ratio $\zeta_{SI}$.

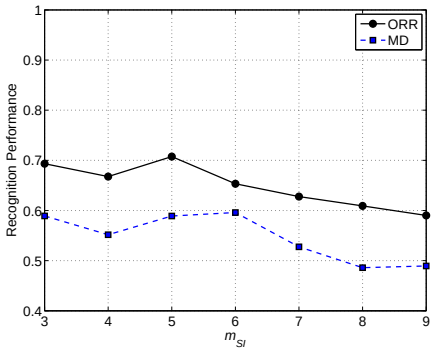



Figure 7.6: Variations of ORR and MD with respect to number of histogram bins $m_{SI}$.

and $\zeta_{SI} = 0.75$. Figure 7.6 shows how ORR and MD are affected from varying values of $m_{SI}$. Similar to effect of number of states, increasing $m_{SI}$ beyond 5 causes ORR and MD decrease. Higher number of histogram bins results in sparser histograms, which makes the clustering process inaccurate in HMM training.

Lastly, we seek for a filterbank that maximize ORR and MD. We use the values $N = 4, N_{SI} = 8, M_{SI} = 8, \zeta_{IS} = 0.75$, and $m_{SI} = 5$. We test **DWT-HMM** with $1^{st}, 2^{nd}, 3^{rd}$, and $4^{th}$ order Daubechies decomposition filters. Resultant ORR and MD are plotted in Figure 7.7. Note that we abbreviate names of filterbanks as 'db***o***' where ***o*** stands for the order. Our expectation was that type of the filterbank should not have any significant effect on ORR and MD, since we are only interested in zero crossing numbers of highband subsignals. Using more complicated filterbanks will not greatly effect the number of zero crossings. Our expectation holds except for 'db3' case as shown in Figure 7.7.

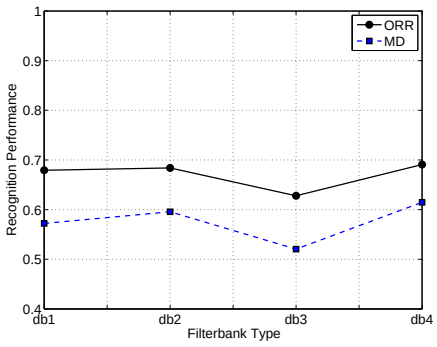
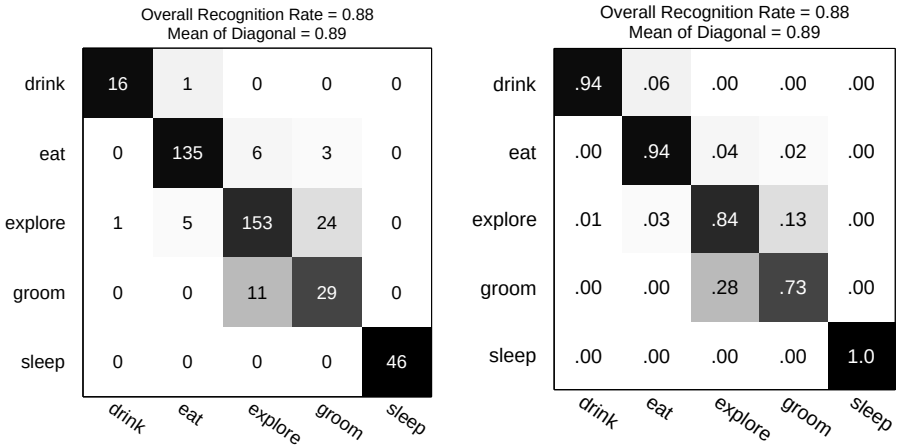



Figure 7.7: ORR and MD with different filterbanks.

## 7.4 Experimental Results for SHOG based Method

In this section, **SHOG** method explained in Chapter 4 is tested on UCSD mice action dataset and recognition performance is illustrated. We prefer to estimate classification accuracy of **SHOG** using K-fold cross validation scheme. We choose the number of subsets, K = 5 for this method. At each validation run first a pose codebook, $\mathcal{P}$ must be constructed from training data as described in section 4.3. Each action volume in the training set is expressed as a pose sequence $\mathcal{S}$ using the constructed codebook $\mathcal{P}$ (see section 4.4 for details). To classify a test action volume $V_{\mathrm{T}}$, a pose sequence $\mathcal{S}_{\mathrm{T}}$ associated with the volume is formed similar to training case. Then, $V_{\mathrm{T}}$ is assigned to the class of training action whose pose sequence is most similar to $\mathcal{S}_{\mathrm{T}}$ according to LCS similarity metric. In Figure 7.8, we illustrate the highest recognition rates by a confusion matrix and its normalized form. Classification success of **SHOG** is quite satisfactory for `drinking, eating`, and `sleeping` actions. However, in this method, `exploring` and `grooming` actions are slightly confused with each other. Confusion matrix illustrated in Figure 7.8(b) is the average of 5 classification runs in K-fold cross validation. We also estimate the classification reliability for each action class as the standard deviation of 5 classification runs. Mean and standard deviation values of classification accuracies are depicted in Table 7.2.




(a) Confusion matrix. (b) Normalized confusion matrix.

Figure 7.8: Highest classification performance for **SHOG** method.

| | Mean± Std. |
|---|---|
| `drinking` | $.94 \pm .10$ |
| `eating` | $.94 \pm .03$ |
| `exploring` | $.84 \pm .04$ |
| `grooming` | $.73 \pm .09$ |
| `sleeping` | $1.00 \pm 0$ |
| ORR | $.88 \pm .02$ |
| MD | $.89 \pm .04$ |

Table 7.2: Mean and standard deviation values of classification rates in **SHOG** method. Here, Std. means standard deviation.

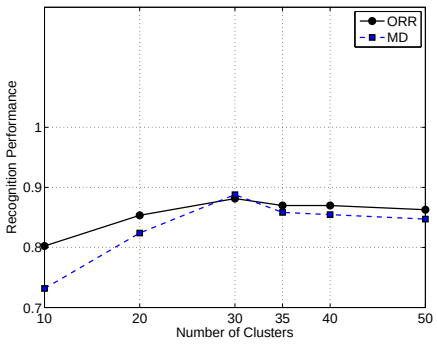
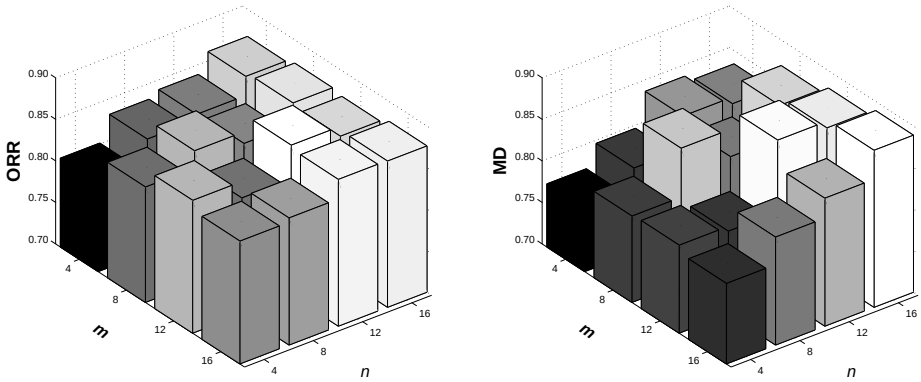



Figure 7.9: Variation of ORR and MD with respect to $K$.

We notice that there are three important parameters in **SHOG** method. Number of radial cells ($n$) and orientation bins ($m$) in SHOG computation directly affect pose description. Last parameter is the number of clusters $K$ in pose codebook construction. Optimal values of $n, m$ and $K$, which results in highest recognition performance, have been determined to be 12, 12, and 30, respectively.

We analyze the effect of number of clusters on classification accuracy. With fixed values of $n = 12$ and $m = 12$, we try the following number of clusters $K$: $K = 10, 20, 30, 35, 40, 50$. In Figure 7.9, ORR and MD are plotted with respect to varying number of clusters. We observe that ORR and MD increase steadily up to $K = 30$. Further increasing $K$ towards 50 causes slight decreases in ORR and MD. Thus, the highest ORR and MD are achieved at $K = 30$.

To examine the effect of $n$ and $m$ on recognition performance, $K$ is set to 30 and four different values of $n$ and $m$ are tested. We tried following values: $n = 4, 8, 12, 16$ and $m = 4, 8, 12, 16$. ORR and MD corresponding to different combinations of $n$ and $m$ values are plotted in Figure 7.10. Considering Figure 7.10, one can notice that higher numbers of radial cells usually result in better recognition performance for both ORR and MD. On the other hand, $m$ has a positive effect on ORR and MD for only high values of $n$. In addition, it is important to note that there is a tradeoff between recognition performance and computational cost. Thus, it is favorable to select values of $n$ and $m$ as 12 which gives the highest ORR and MD with minimal computational cost.




(a) Overall recognition rates. (b) Mean of individual recognition rates.

Figure 7.10: Effects of $n$ and $m$ on ORR and MD.

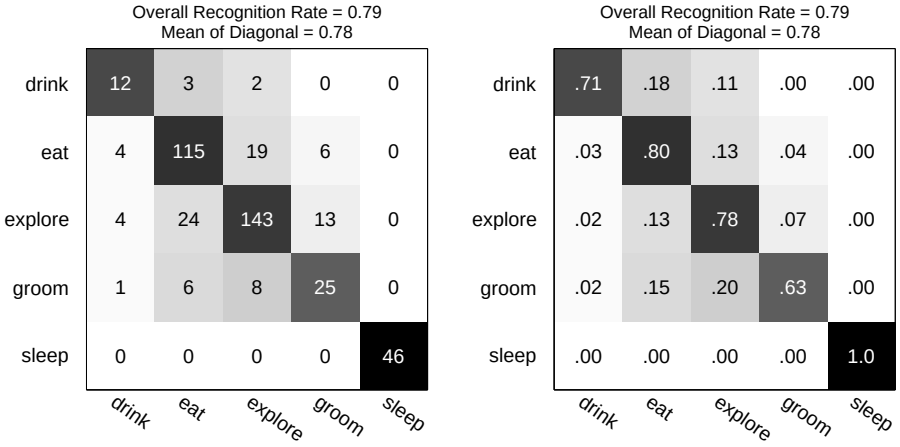
## 7.5 Experimental Results for Global Histograms of 3D Gradients based Method

In this section, we demonstrate our experimental results for **3DGrads** method which is based on global histograms of 3D Gradients. As explained in Chapter 5, we extract a set of one dimensional histograms,

$$\left\{h_x^{(1)}, h_y^{(1)}, h_t^{(1)}, h_x^{(2)}, \ldots, h_t^{(l_{\max})}\right\}$$

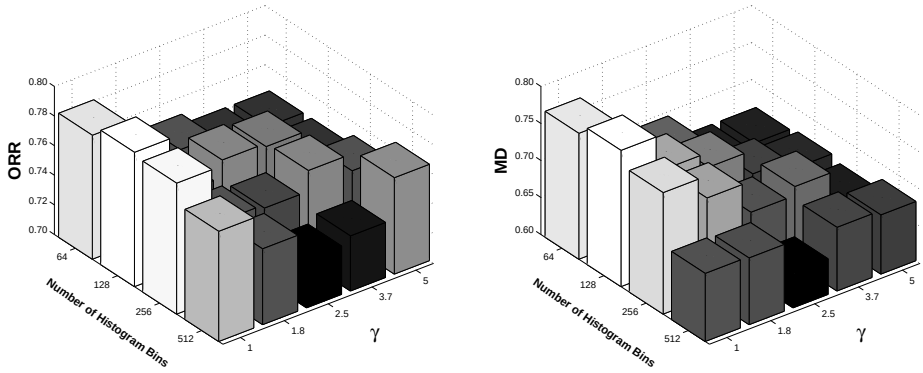
for each video clip in the dataset. Here, we choose $l_{\max} = 3$, so that we represent each video clip by a set of 9 histograms. Then, we end up in confusion tables using leave one out cross validation with 1-NN classifiers. We used $\chi^2$ distance given in section 5.4 to evaluate the similarity of two videos.

In **3DGrads** there are only two tunable parameters, $\gamma$ constant which is used to adjust the threshold and the number of bins in histograms $\{h_k^{(l)}, l = 1, \ldots, l_{\max}$ and $k \in \{x, y, t\}\}$. The highest recognition rates are achieved with $\gamma = 1$ and 128 bins in histograms. In Figure 7.11, the best confusion matrix achieved by **3DGrads** method and its normalized form are illustrated. It is observed that all of the action classes are recognized with fairly high rates.




(a) Confusion matrix. (b) Normalized confusion matrix.

Figure 7.11: Highest recognition performance achieved by the method **3DGrads**.




(a) Overall recognition rates. (b) Mean of individual recognition rates.

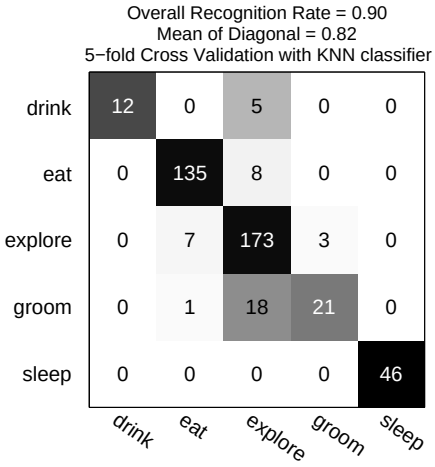
Figure 7.12: Overall recognition rates and mean of individual recognition rates for different values $\gamma$ and number of histograms bins.

We analyze the effects of $\gamma$ and the number of histogram bins on recognition rates. We try the following values $\gamma \in \{1, 1.8, 2.5, 3.7, 5\}$ and the number of bins $\in \{64, 128, 256, 512\}$. Overall recognition rate and the mean of individual recognition rates are shown as bar graphics in Figure 7.12 for all parameter combinations. We observe that different values of parameters do not affect overall recognition rates, where individual behavior recognition rates decrease with increasing $\gamma$ and with increasing number of histogram bins.

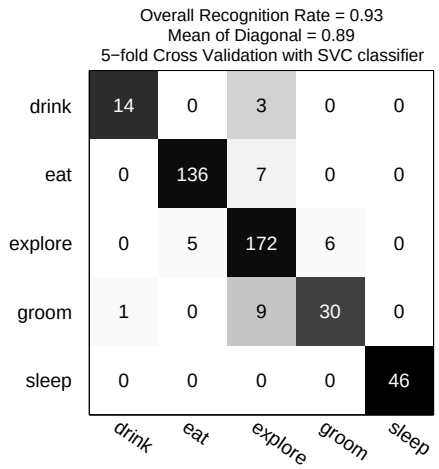
## 7.6 Experimental Results for Sparse Keypoints and HOG-HOF based Method

In this section, we apply the method **SparseHOG** to UCSD mice action dataset. In order to realize K-fold cross validation, first we split the dataset into K sets containing nearly equal number of videos. We choose K = 5 in our experiments. At each classification run, we treat one of the sets as testing data, and the rest as training data. For K= 5, we perform the classification process for 5 times as K-fold cross validation requires.

We extract HOG and HOF descriptors for each detected spatio-temporal Harris corner in training data at each classification run. We utilize the code provided by Ivan Laptev [83] for HOG and HOF descriptor computation. Then, we group extracted descriptors into $K$ clusters by k-means clustering. Clustering operation provides us the codebook, $\mathcal{E} = \{e_1, e_2, \ldots, e_K\}$. We form one histogram, $\mathcal{A}_i$ for each training video in order to train 1-NN and radial basis SVM classifiers. It has been explained in section 6.4, how to form the histograms $\{\mathcal{A}_i\}$. Prior to classification of a test video, $V_{\mathrm{T}}$, HOG and HOF descriptors, $f^k_{\mathrm{HOG_T}}$, $f^k_{\mathrm{HOF_T}}$ for each detected 3D Harris corner,$(x_c, y_c, z_c)^k$ in $V_{\mathrm{T}}$ are extracted in the same manner. We form a histogram $\mathcal{A}_{\mathrm{T}}$ in a similar way to training process for the test video $V_{\mathrm{T}}$. At the end, we classify $\mathcal{A}_{\mathrm{T}}$ by trained 1-NN and radial basis SVM classifiers. Above procedure is repeated for all test videos. At the next classification run, we alternate the test set and repeat the classification. Classification accuracy is then given as the average of 5 classification runs. The best recognition rates achieved by applying the method **SparseHOG** with 1-NN and SVM classifiers to UCSD mice action dataset are depicted as confusion matrices in Figures 7.13 and 7.14. Notice that SVM classifier outperforms 1-NN classifier at recognition of `drinking` and `grooming` actions, where they perform nearly same for the rest

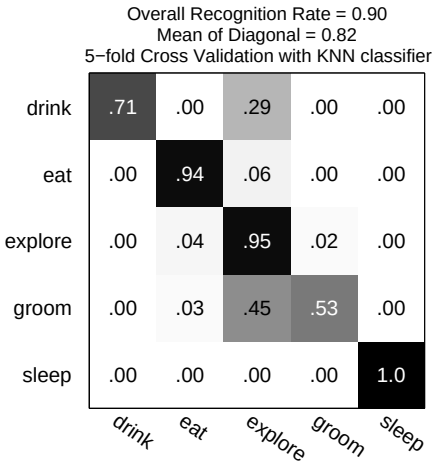



(a) 1-NN classifier.

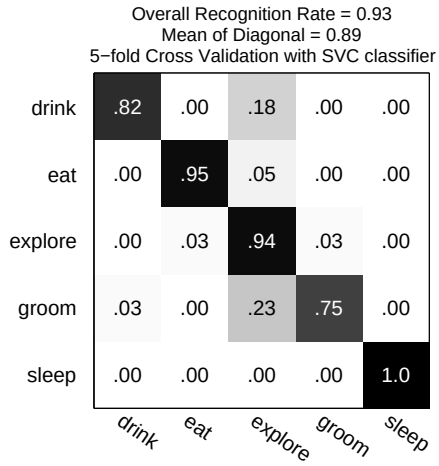



(b) Radial basis SVM classifier.

Figure 7.13: Best confusion matrices achieved by the method **SparseHOG**.




(a) 1-NN classifier.




(b) Radial basis SVM classifier.

Figure 7.14: Best normalized confusion matrices achieved by the method **SparseHOG**.

of actions. Similar to **SHOG** case, we provide classification mean and standard deviation values for each action class, ORR, and MD in Table 7.3.

We notice that in **SparseHOG** there are a number of parameters which may affect the recognition performance. Parameters related to 3D Harris corner detector (see subsection 6.1.3) are number of spatial and temporal scales ($n$ and $m$) and $k$ parameter in Equation (6.8). HOG and HOF descriptor computation process introduces the parameters, descriptor type (HOG, HOF or concatenation of both which we denote as HOGHOF) and $\zeta$ scale factor which determines the

| | Mean± Std. |
|---|---|
| drinking | $.71 \pm .27$ |
| eating | $.94 \pm .04$ |
| exploring | $.95 \pm .05$ |
| grooming | $.53 \pm .05$ |
| sleeping | $1.00 \pm 0$ |
| ORR | $.90 \pm .02$ |
| MD | $.82 \pm .04$ |

(a) 1-NN classifier.

| | Mean± Std. |
|---|---|
| drinking | $.82 \pm .15$ |
| eating | $.95 \pm .05$ |
| exploring | $.94 \pm .07$ |
| grooming | $.75 \pm .08$ |
| sleeping | $1.00 \pm 0$ |
| ORR | $.93 \pm .03$ |
| MD | $.89 \pm .02$ |

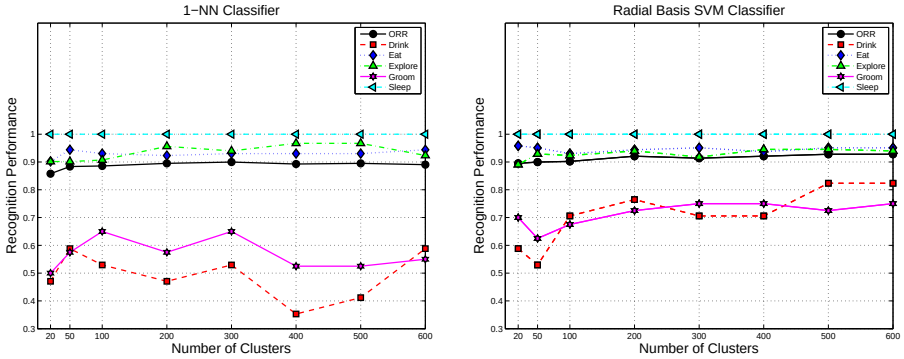
(b) Radial basis SVM classifier.

Table 7.3: Mean and standard deviation values of classification rates in **Sparse-HOG** method. Here, Std. means standard deviation.

descriptor cuboid size (see section 6.2). Another significantly important factor is the number of clusters, $K$ which determines the size of codebook, $\mathcal{E}$. In Figure 7.14(a) we achieve the highest recognition rates for 1-NN classifier with parameter values $n = 6, m = 2, k = 0.0005$, $\zeta = 7$, $K = 600$ and using HOGHOF descriptor. In 1-NN classifier spatial and temporal scales are selected as $\sigma_l = 2^{(i+1)/2}, i = 1, 2, \ldots, 6$, and $\tau_l = 2^{j/2}, j = 1, 2$, respectively. On the other hand, in Figure 7.14(b) optimal parameter values which result in highest recognition rates for SVM classifier are $n = 6, m = 2, k = 0.0005$, $\zeta = 5$, $K = 600$, and utilizing HOGHOF descriptor. Values of spatial and temporal scales for SVM classifier are same as 1-NN case. Among the parameter set stated above, we believe that most important ones are $K, n, m, \zeta$, and the type of the descriptor. Therefore, we analyze the recognition performance of 1-NN and SVM classifiers for different values of these parameters.

### 7.6.1 Effect of Number of Clusters $K$

To find out the effect of number of clusters $K$ in codebook construction on recognition performance, we fix the values of remaining parameters to $n = 6, m = 2, k = 0.0005, \zeta = 5$, and use HOGHOF as our descriptor. We select values of spatial and temporal scales as $\sigma_l = 2^{(i+1)/2}, i = 1, 2, \ldots, 6$ and $\tau_l = 2^{j/2}, j = 1, 2$. Then, we investigate recognition rates for each action class and overall recognition




(a) 1-NN classifier. (b) Radial basis SVM classifier.

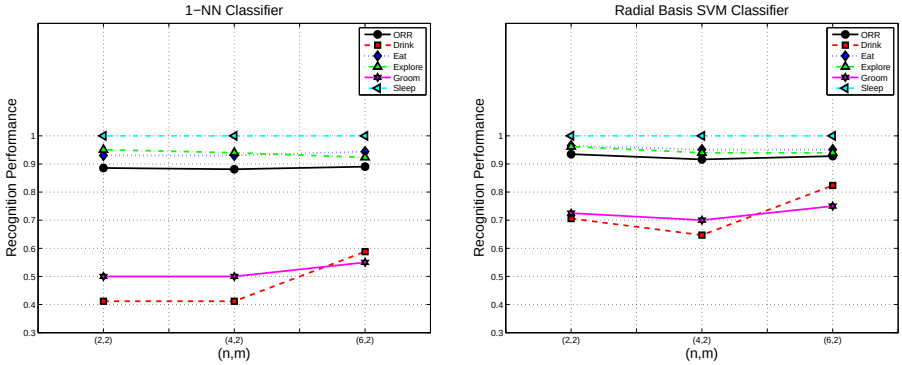
Figure 7.15: Effect of number of clusters $K$ on recognition performances.

rate for different values of $K$. We repeat the experiments for 8 different values of $K \in \{20, 50, 100, 200, 300, 400, 500, 600\}$. Overall recognition rate and individual behavior recognition rates are plotted with respect to varying numbers of clusters in Figure 7.15.

Regarding Figure 7.15(a), varying values of $K$ slightly affect overall recognition rate and classification performances for the actions `eating, exploring`, and `sleeping`. However, success rates for recognizing `grooming` and `drinking` actions fluctuate with increasing $K$. Best recognition rates for `drinking` action are observed for $K = 50$ and $K = 600$, whereas for `grooming` it occurs at $K = 100$ and 300. Effect of $K$ is similar for overall recognition rate and recognition rates of `eating, exploring`, and `sleeping` in SVM classifier case. But different from 1-NN case, classification success rates for `drinking` and `grooming` have tendency to raise with increasing values of $K$. We note that `grooming` is best recognized when $K = 300, 400,$ and 600, where for $K = 500$ and 600 recognition rate for `drinking` reaches its maximum.

### 7.6.2 Effect of Number of Spatial and Temporal Scales

In this subsection, we examine the effects of the number of spatial and temporal scales ($n$ and $m$) on recognition performance. Similar to previous subsection,

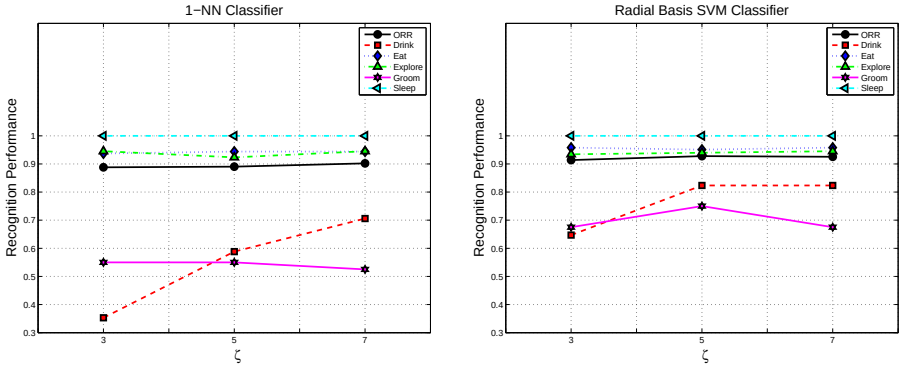



(a) 1-NN classifier. (b) Radial basis SVM classifier.

Figure 7.16: Effect of number of spatial and temporal scales $(n, m)$ on recognition performances.

we fix the values of other parameters and vary the values of the investigated parameters. To analyze relevant parameters, we fix the values of parameters $K$ and $\zeta$ to 600 and 5. We used HOGHOF descriptor for all experiments in this subsection. We measure the recognition performance for three different sets of spatial and temporal scales. In the first set, we select $n = 2$ and $m = 2$ that results in the values, $\sigma_l^2 = 4, 8$ and $\tau_l^2 = 2, 4$. In the second set, $n$ and $m$ are selected as 4 and 2 resulting in $\sigma_l^2 = 4, 8, 16, 32$ and $\tau_l^2 = 2, 4$. Lastly, setting $n = 6$ and $m = 2$ gives us the spatial and temporal scales, $\sigma_l^2 = 4, 8, 16, 32, 64, 128$ and $\tau_l^2 = 2, 4$, respectively. In Figure 7.16, we illustrate overall and individual behavior recognition rates for 1-NN and SVM classifiers.

We observe that recognition rates for all classes reach their maximum at $n = 6$ and $m = 2$ in Figure 7.16. The reason is that by increasing number of spatial and temporal scales we detect more 3D Harris corners which boosts discrimination of codeword histograms $\mathcal{A}$.




(a) 1-NN classifier. (b) Radial basis SVM classifier.

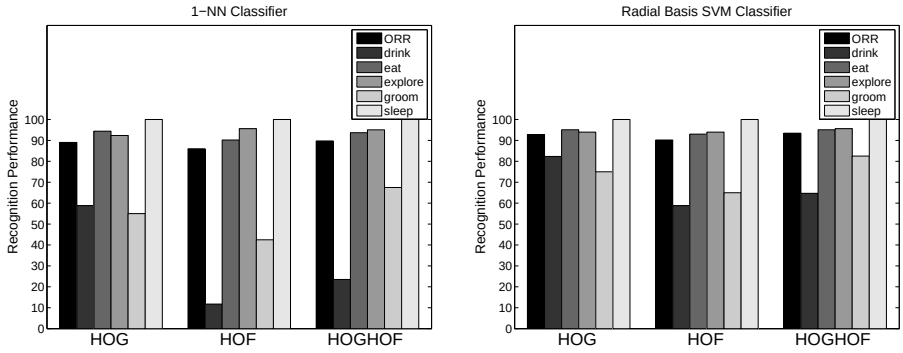
Figure 7.17: Effect of $\zeta$ on recognition performances.

### 7.6.3 Effect of Descriptor Cuboid Scale Factor $\zeta$

Similar to previous subsections, we fix the values of remaining parameters, where we repeat the experiments with varying values of the parameter under investigation. Accordingly, we set $K = 600$, $n = 6$, $m = 2$, and used HOGHOF as cuboid descriptor. We try the following values for the cuboid scale factor, $\zeta \in \{3, 5, 7\}$. We demonstrate the effect of $\zeta$ on overall and individual behavior recognition performance in Figure 7.17. The parameter $\zeta$ affects the size of cuboid from which HOG and HOF descriptors are computed. Considering that HOG and HOF computation process is dependent on gradient and optical flow estimation, which are highly numeric operations, it is hard to explain its effect on recognition rates.

### 7.6.4 Effect of Descriptor Type

Finally we analyze the effect of the type of descriptor on recognition rates. To measure recognition performance with different types of descriptor, we fix $K, n, m,$ and $\zeta$ to 600, 6, 2, and 5, respectively. We test three different types of descriptors, namely HOG, HOF and HOGHOF. Note that HOGHOF is direct concatenation of HOG and HOF descriptors. In Figure 7.18, we show the effect of descriptor type used in **SparseHOG** on recognition performances.



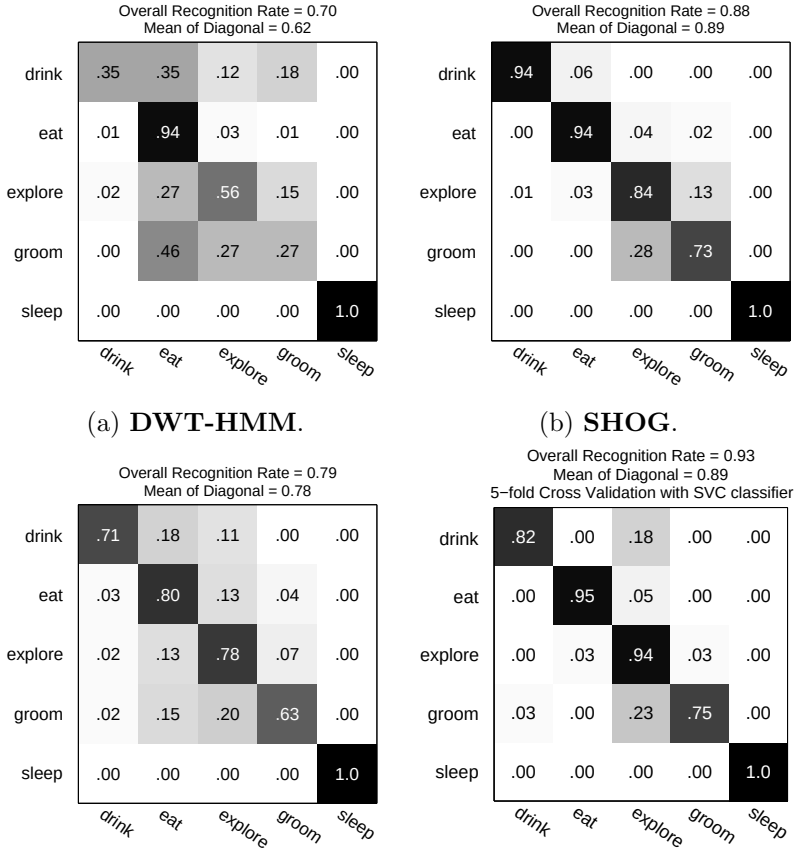

(a) 1-NN classifier. (b) Radial basis SVM classifier.

Figure 7.18: Effect of descriptor type on recognition performances.

Examining Figure 7.18, one can deduce that overall recognition rates and classification success for `exploring, eating`, and `sleeping` change a little bit for different descriptor types. For `drinking` action, HOG descriptor outperforms HOF and HOGHOF descriptors. We believe that since the mouse mostly stays still during `drinking` action, spatial features (HOG) are more discriminative then the temporal ones (HOF). For `grooming` case, HOGHOF performs better than HOG and HOF. Considering that during grooming mouse can exhibit both rapid and slow feet movements, taking both spatial and temporal features (HOGHOF) into account results in the highest recognition rates. For both 1-NN and SVM classifiers HOF desciptor is the worst one at recognition of `grooming`. The reason for that is that the body parts of the mouse move either very fast or slow during grooming. Fast movements give rise to motion blur which causes inaccuracies in optical flow estimation, whereas slow motions result in optical flow vectors close to zero. Optical flow information is useless in both cases.

## 7.7 Comparisons

In this section, we provide comparisons of second stage methods with both each other and related studies in the literature. We would like to thank Piotr Dollar




(a) **DWT-HMM**. (b) **SHOG**.

(c) **3DGrads**. (d) **SparseHOG** with SVM.

Figure 7.19: Comparison of second stage methods with each other.

and Hueihan Jhuang for their kind attitude in sharing details of experimental methodology in their work.

### 7.7.1 Comparisons of Second Stage Methods

In this subsection, second stage methods are compared with each other in terms of recognition accuracy. Normalized confusion matrices for second stage methods, namely **DWT-HMM**, **SHOG**, **3DGrads**, and **SparseHOG** are given in Figure 7.19.

Considering Figure 7.19, the weakest second stage method is **DWT-HMM**, since it only can recognize `eating` and `sleeping` actions with high rates. In **DWT-HMM** remaining action classes are heavily confused with each other.

| Method | ORR | MD | Classification Methodology |
|---|---|---|---|
| **SparseHOG** with SVM | 93% | 89% | 5-Fold Cross Validation |
| **SHOG** | 88% | 89% | 5-Fold Cross Validation |
| **3DGrads** | 79% | 78% | Leave-one-out Cross Validation |
| **DWT-HMM** | 70% | 62% | Leave-one-out Cross Validation |

Table 7.4: ORR and MD comparison between second stage methods.

| | Set1 | Set2 | Set3 | Set4 | Set5 | Set6 | Set7 |
|---|---|---|---|---|---|---|---|
| `drinking` | 3 | 6 | 3 | 3 | 2 | 0 | 0 |
| `eating` | 30 | 30 | 30 | 30 | 2 | 0 | 22 |
| `exploring` | 30 | 30 | 30 | 30 | 30 | 4 | 30 |
| `grooming` | 1 | 5 | 7 | 2 | 19 | 0 | 10 |
| `sleeping` | 0 | 0 | 0 | 0 | 24 | 22 | 0 |

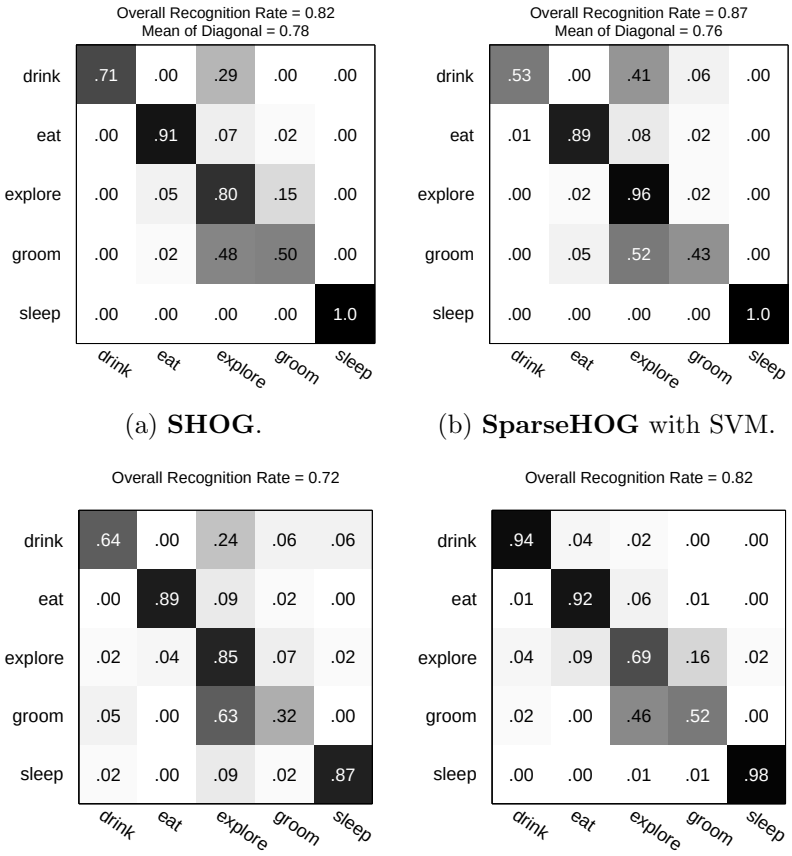
Table 7.5: Distribution of action classes in the sets.

The most successful methods in second stage are **SHOG** and **SparseHOG** with SVM classifier. They successfully discriminate all of the action classes. On the other hand, **3DGrads** method is not as successful as **SHOG** and **SparseHOG**, but at least it achieves recognition rates higher than 60% for all classes.

We also provide comparisons between second stage methods according to ORR and MD. Comparison of second stage methods in terms of ORR and MD is depicted in Table 7.4.

### 7.7.2 Comparisons with Related Studies

In this subsection, we compare two of the most successful second stage methods, **SHOG** and **SparseHOG** to related studies previously worked on UCSD mice action dataset. The methods **SHOG** and **SparseHOG** are implemented with the same methodology followed by related studies. Accordingly, the dataset is split into seven sets. Numbers of action classes in each set is given in Table 7.5. Notice that action classes are irregularly distributed to the sets. K-fold cross validation with K = 7 is applied to the seven sets as the classification scheme. In Figure 7.20, recognition performances of two second stage methods, namely **SHOG** and **SparseHOG** are compared to those of [22] and [38].




(a) **SHOG**. (b) **SparseHOG** with SVM.

(c) Dollar *et. al* [22]. (d) Jhuang *et. al* [38].

Figure 7.20: Comparison of the methods **SHOG** and **SparseHOG** with related studies.

| Method | ORR |
|---|---|
| **SparseHOG** with SVM | 87% |
| **SHOG** | 82% |
| Jhuang *et. al* [38] | 82% |
| Dollar *et. al* [22] | 72% |

Table 7.6: ORR comparison of **SHOG** and **SparseHOG** methods with related studies.

**SHOG** method performs similar or better than [22] for all action classes. We observe that recognition rates of **SHOG** method and [38] for `eating, grooming`, and `sleeping` actions are quite similar. On the other hand, **SHOG** method is better than [38] at classifying `exploring` action, where [38] performs much better than **SHOG** for `drinking` action.

**SparseHOG** method outperforms both [22] and [38] for `exploring` and `sleeping` actions. All three methods recognize `eating` action with very similar rates. Method of [38] is the best at classifying `drinking` and `grooming` actions. However, there is no big difference between [38] and **SparseHOG** for `grooming` action.

We also carry out comparisons based on ORR. In Table 7.6, ORR for **SHOG** and **SparseHOG** methods and related studies is given. We can only provide comparisons in terms of ORR, since neither in [22] nor [38] MD information is given.

## 7.8 Comments on Experimental Results

Throughout this chapter we tested each action recognition method on UCSD mice action dataset and observed its classification performance. As explained in the introduction chapter, mice action recognition problem poses many challenges due to both body and behavior characteristics of mice and recording environment. We believe that comparing action recognition methods on such a challenging dataset reveals weak and strong points of the methods. Our main inference is that simple representations of actions tend to be more successful than complex ones. In this section, we present our discussions on weakness and strength of each method investigated.

The first stage of our action recognition framework meets the requirements and perfectly separates sleeping actions from the others. Simply examining amount of motion addresses the assumption that sleeping is a still action.

Among the four different action recognition methods, **DWT-HMM**, **SHOG**, **3DGrads**, and **SparseHOG**, proposed for the second stage, **SHOG** and **SparseHOG** compete for the highest recognition performance.

**DWT-HMM** method fails to discriminate actions from each other. We observe that projecting 3D action volumes onto ASIs by using DWT discards local temporal characteristics and generates random patterns in ASIs as a result. We employed a theoretically well-grounded tool, HMMs with continuous observations in order to model randomness in ASIs and incorporate spatial coherence in ASIs. However, HMMs fail to model ASIs except for eating action due to shortness of observation sequences and extreme randomness in ASIs.

One of the most successful methods, **SHOG**, accomplishes action recognition task with high rates. As claimed in [6], describing actions as sequences of pose words is quite simple and discriminative. Histograms of oriented gradients are computed over radial grids to describe pose in each frame. Note that this operation both minimizes effect of noise by taking histograms and preserves local structure using a griding scheme. The only problem with this method is that it requires proper segmentation of foreground objects from background, which is considered to be one of the most challenging problems in computer vision area. Instead of attempting to segment the subject in each frame, we use only temporally active points in pose description. By imposing temporal constraints on pose description we also enrich HOG features by taking temporal information into account. As a result, this method performs quite well in action recognition of laboratory mice.

Being a modest method, **3DGrads** considers whole action volume in feature extraction process. Taking histograms of normalized gradient vectors over entire action volume provides robustness to perturbations but also ignores local spatio-temporal structures. One advantage of **3DGrads** is that feature extraction is carried out at different temporal resolutions. In short, **3DGrads** pays the price of being simple and parameter free by moderate recognition rates.

Lastly, **SparseHOG** method achieves quite high recognition rates by modeling actions as sets of sparse spatio-temporal cuboids. Locating such informative and sparse points in the action volume is addressed by 3D Harris corner detector which provides scale and rotation invariance. Describing detected points by HOG and HOF takes into account both spatial and temporal information and is robust to noise. In contrast to arguments in the literature that spatio-temporal corners in real-world actions are quite rare, we observe sufficient number of corners in all action classes except drinking and sleeping. In **SparseHOG**, we utilize radial basis SVM classifiers which are considered to be maximum margin classifiers. The only weakness of **SparseHOG** is that it ignores geometrical relations between detected cuboids inside the action volume. Nevertheless, **SparseHOG** gives the highest recognition rates in our experiments.

Computational complexities of **SHOG** and **SparseHOG** are much higher than other two methods'. The main computational load in **SHOG** is extracting pose descriptors from all training frames and clustering operation. Similarly, in **SparseHOG**, seeking for salient points at multiple spatio-temporal scales, HOG and HOF extraction process and k-means clustering takes great amount of time. Conversely, operations carried out in **3DGrads** and **DWT-HMM** require lower computational cost. In fact, **DWT-HMM** method would require much more time, if number of states, dimension of observation symbols, and length of observation symbols were higher.

Considering experimental results with all methods, we conclude that simple action representation in conjunction with informative and discriminative feature extraction usually gives the most successful results.

# Chapter 8

# Conclusions and Future Work

In this thesis we build a two level hierarchical system to recognize mice actions from short video clips. Designed system is a preliminary work for a general continuous action recognition system which greatly aids pharmacologists in their experiments on mice. The first stage of the system is used to distinguish still actions such as sleeping action from others. We propose four different methods for the second stage which is expected to classify remaining actions as accurate as possible. The first algorithm is a cascade combination of two subsystems based on DWT and HMMs, respectively. The second method exploits subject's pose to achieve classification. The third method relies on spatio-temporal global features extracted from action volumes. Lastly, the fourth method models an action as a combination of learned spatio-temporal templates. We performed experiments on a real mice action dataset and achieved 93% success rate. We have also made comparisons between the methods in this thesis and related studies in the literature.

Major contribution of this thesis is that it provides a comprehensive study of action recognition methods in mice case. In addition, we propose a simple method to discriminate still actions from others. A novel method exploiting temporal

variations in pixel intensities by DWT and modeling those variations by HMMs is designed. We have also adopted three methods which were previously used in human action representation or recognition to mice case. We provide a simple and effective solution to determine the support region for pose description in the first method given in [6]. We combine the second method in [9] with adaptive thresholding and 1-NN classifier to be used in mice action recognition. Lastly, we evaluate the performance of the third method [10] on mice actions. Furthermore, we performed parameter search for all the methods tested.

We have faced many challenges in mice action recognition during this study. Major conclusions that we drew are given below:

- Quantifying the amount of motion in a given video is sufficient enough to identify still actions.
- Accumulating temporal variations of individual pixels all over the time axis discards local temporal information which could be useful in feature extraction.
- Deciding on model complexity and observation symbols is a key issue in hidden Markov models. The observation symbols must be long enough to reliably train HMMs.
- In pose-based action recognition methods, imposing temporal constraints on spatial feature extraction greatly improves classification performance.
- Volumetric features extracted from entire action volume are robust to local perturbations. However, they disregard all positional information (both spatial and temporal).
- Sparse keypoints in the action volume where large variations occur along spatial and temporal axes can be sufficient to model complex actions.

- Localized histograms of orientations of gradient and optical flow vectors are powerful descriptors for local patches.

There are many improvements and extensions that can be incorporated into our framework. We list some of those below:

- We plan to extend our system for action recognition from continuous video streams.
- Spatio-temporally windowed wavelet coefficients can be fed to HMMs as observation symbols for better training.
- In pose-based action recognition, temporal dynamics of pose sequences can be taken into account by using HMMs instead of simple sequence matching methods.
- We intend to extend global gradients based approach to handle scale variations in both space and time by constructing spatial and temporal pyramids. Furthermore, we believe that taking localized histograms of orientations of gradient vectors will improve recognition performance.
- In sparse keypoint approach, considering positional dependency between keypoints may boost classification accuracy.

# Bibliography

[1] S. Belongie, K. Branson, P. Dollár, and V. Rabaud, "Monitoring animal behavior in the smart vivarium," *Measuring Behavior, Wageningen, The Netherlands*, 2005.

[2] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: A local svm approach," in *Proceedings of the 17th International Conference on Pattern Recognition*, vol. 3, 2004.

[3] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," in *IEEE International Conference on Computer Vision*, vol. 2, 2005.

[4] K. Branson, *Tracking multiple mice through severe occlusions*. PhD thesis, University of California at San Diego La Jolla, CA, USA, 2007.

[5] B. U. Toreyin, Y. Dedeoglu, A. E. Cetin, "Flame detection in video using hidden markov models," in *IEEE International Conference on Image Processing*, vol. 2, 2005.

[6] K. Hatun and P. Duygulu, "Pose sentences: A new representation for action recognition using sequence of pose words," in *International Conference on Pattern Recognition*, pp. 1–4, 2008.

[7] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Computer Society Conference on Computer Vision and*

*Pattern Recognition* (C. Schmid and S. Soatto and C. Tomasi, ed.), vol. 2, pp. 886–893, June 2005.

[8] D. S. Hirschberg, "A linear space algorithm for computing maximal common subsequences," *Communications of the ACM*, vol. 18, no. 6, pp. 341–343, 1975.

[9] L. Zelnik-Manor and M. Irani, "Statistical analysis of dynamic actions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1530–1535, 2006.

[10] I. Laptev, M. Marszalek, C. Schmid and B. Rozenfeld, "Learning realistic human actions from movies," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008.

[11] D. M. Gavrila, "The visual analysis of human movement: A survey," *Computer Vision and Image Understanding*, vol. 73, no. 1, pp. 82–98, 1999.

[12] J. K. Aggarwal, and Q. Cai, "Human motion analysis: A review," in *Proceedings of IEEE Nonrigid and Articulated Motion Workshop*, pp. 90–102, 1997.

[13] L. Wang, W. Hu, and T. Tan, "Recent developments in human motion analysis," *Pattern recognition*, vol. 36, no. 3, pp. 585–601, 2003.

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

[15] A. F. Bobick and J. W. Davis, "The recognition of human movement using temporal templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 3, pp. 257–267, 2001.

[16] A. Yilmaz and M. Shah, "Actions sketch: A novel action representation," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, 2005.

[17] L. Gorelick, M. Blank, E. Shechtman, M. Irani, R. Basri, "Actions as space-time shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, pp. 2247–2253, 2007.

[18] A. A. Efros, A. C. Berg, G. Mori, and J. Malik, "Recognizing actions at a distance," in *Proceedings of Ninth IEEE International Conference on Computer Vision*, pp. 726–733, 2003.

[19] I. Laptev, "On space-time interest points," *International Journal of Computer Vision*, vol. 64, no. 2-3, pp. 107–123, 2005.

[20] A. Kläser, M. Marszałek and C. Schmid, "A spatio-temporal descriptor based on 3d-gradients," in *British Machine Vision Conference*, pp. 995–1004, September 2008.

[21] I. Laptev, B. Caputo, Ch. Schultd and T. Lindeberg, "Local velocity-adapted motion events for spatio-temporal recognition," *Computer Vision and Image Understanding*, vol. 108, no. 3, pp. 207–229, 2007.

[22] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in *2nd Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pp. 65–72, 2005.

[23] J. C. Niebles, H. Wang, and L. Fei-Fei, "Unsupervised learning of human action categories using spatial-temporal words," *International Journal of Computer Vision*, vol. 79, no. 3, pp. 299–318, 2008.

[24] P. Scovanner, S. Ali, and M. Shah, "A 3-dimensional sift descriptor and its application to action recognition," in *Proceedings of the 15th International Conference on Multimedia*, pp. 357–360, 2007.

[25] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[26] S. Savarese, A. DelPozo, J. C. Niebles, and L. Fei-Fei, "Spatial-temporal correlatons for unsupervised action classification," in *IEEE Workshop on Motion and Video Computing*, pp. 1–8, 2008.

[27] S. Nowozin, G. BakIr, and K. Tsuda, "Discriminative subsequence mining for action classification," in *IEEE International Conference on Computer Vision*, vol. 22, pp. 25–28, 2007.

[28] O. Boiman and M. Irani, "Detecting irregularities in images and in video," *International Journal of Computer Vision*, vol. 74, no. 1, pp. 17–31, 2007.

[29] S. F. Wong, T. K. Kim, and R. Cipolla, "Learning motion categories using both semantic and structural information," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2007.

[30] A. Gilbert, J. Illingworth, and R. Bowden, "Scale invariant action recognition using compound features mined from dense spatio-temporal corners," in *Proceedings of the 10th European Conference on Computer Vision: Part I*, pp. 222–233, 2008.

[31] S. F. Wong and R. Cipolla, "Extracting spatiotemporal interest points using global information," in *Proceedings of IEEE International Conference on Computer Vision*, pp. 1–8, 2007.

[32] Y. Ke, R. Sukthankar and M. Hebert, "Efficient visual event detection using volumetric features," in *IEEE International Conference on Computer Vision*, vol. 1, pp. 166–173, October 2005.

[33] E. Shechtman and M. Irani, "Space-time behavior based correlation," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 267–296, 2005.

[34] T. K. Kim, S. F. Wong, and R. Cipolla, "Tensor canonical correlation analysis for action classification," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2007.

[35] X. Li, "HMM based action recognition using oriented histograms of optical flow field," *Electronics Letters*, vol. 43, no. 10, pp. 560–561, 2007.

[36] D. Kawanaka, T. Okatani, and K. Deguchi, "Hhmm based recognition of human activity," *IEICE Transactions on Information and Systems*, no. 7, pp. 2180–2185, 2006.

[37] P. Peursum, H. H. Bui, S. Venkatesh, and G. West, "Robust recognition and segmentation of human actions using hmms with missing observations," *EURASIP Journal on Applied Signal Processing*, vol. 13, p. 2110, 2005.

[38] H. Jhuang, T. Serre, L. Wolf, and T. Poggio, "A biologically inspired system for action recognition," in *Proceedings of IEEE International Conference on Computer Vision*, pp. 1–8, IEEE, 2007.

[39] X. Xue and T. C. Henderson, "Video-based animal behavior analysis from multiple cameras," in *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pp. 335–340, 2006.

[40] Z. Kalafatic, "Model-based tracking of laboratory animals," *The IEEE Region 8 EUROCON 2003. Computer as a Tool*, vol. 2, 2003.

[41] Z. Kalafatic, S. Ribaric, and V. Stanisavljevic, "A system for tracking laboratory animals based on optical flow and active contours," in *Proceedings of 11th International Conference on Image Analysis and Processing*, pp. 334–339, 2001.

[42] T. Burghardt and J. Calic, "Analysing animal behaviour in wildlife videos using face detection and tracking," *IEE Proceedings-Vision, Image and Signal Processing*, vol. 153, no. 3, pp. 305–312, 2006.

[43] P. Viola M. Jones, "Robust real-time object detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2002.

[44] M. M. Naeini, G. Dutton, K. Rothley, and G. Mori, "Action recognition of insects using spectral clustering," in *IAPR Conference on Machine Vision Applications*, 2007.

[45] J. Wawerla, S. Marshall, G. Mori, K. Rothley, and P. Sabzmeydani, "Bearcam: Automated wildlife monitoring at the arctic circle," *Machine Vision and Applications*, vol. 20, no. 5, pp. 303–317, 2009.

[46] P. P. Vaidyanathan, *Multirate systems and filter banks*. Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1993.

[47] Z. Yucel, "Watermarking via zero assigned filter banks," Master's thesis, Dept. of Electrical and Electronics Eng., Bilkent University, Ankara, Turkey, August 2005.

[48] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[49] A. V. Nefian and M. H. Hayes III, "Hidden markov models for face recognition," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 5, 1998.

[50] J. T. Chien and C. P. Liao, "Maximum confidence hidden markov modeling for face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 606–616, 2008.

[51] A. D. Wilson and A. F. Bobick, "Parametric hidden markov models for gesture recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 9, pp. 884–900, 1999.

[52] V. Kellokumpu, M. Pietikäinen, and J. Heikkilä, "Human activity recognition using sequences of postures," in *Proceedings of the IAPR Conference on Machine Vision Applications (MVA 2005), Tsukuba Science City, Japan*, pp. 570–573, 2005.

[53] M. Bicego, U. Castellani, and V. Murino, "Using hidden markov models and wavelets for face recognition," in *Proceedings of 12th International Conference on Image Analysis and Processing*, pp. 52–56, 2003.

[54] V. V. Kohir and U. B. Desai, "Face recognition using a dct-hmm approach," in *Proceedings of the 4th IEEE Workshop on Applications of Computer Vision*, (Washington, DC, USA), p. 226, IEEE Computer Society, 1998.

[55] A. Bobick and J. Davis, "An appearance-based representation of action," *International Conference on Pattern Recognition*, vol. 1, p. 307, 1996.

[56] C. Thurau, "Behavior histograms for action recognition and human detection," *Lecture Notes in Computer Science*, vol. 4814, p. 299, 2007.

[57] C. Thurau and V. Hlavac, "Pose primitive based human action recognition in videos or still images," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 0, pp. 1–8, 2008.

[58] L. Wang and D. Suter, "Informative shape representations for human action recognition," in *International Conference on Pattern Recognition*, (Washington, DC, USA), pp. 1266–1269, IEEE Computer Society, 2006.

[59] M. M. Rahman and S. Ishikawa, "Robust appearance-based human action recognition," *International Conference on Pattern Recognition*, vol. 3, pp. 165–168, 2004.

[60] N. Ikizler and P. Duygulu, "Histogram of oriented rectangles: A new pose descriptor for human action recognition," *Image and Vision Computing*, vol. 27, no. 10, pp. 1515–1526, 2009.

[61] S. P. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–136, 1982.

[62] C. Harris and M. Stephens, "A combined corner and edge detection," in *Proceedings of The Fourth Alvey Vision Conference*, pp. 147–151, 1988.

[63] I. Laptev and T. Lindeberg, "Space-time interest points," *IEEE International Conference on Computer Vision*, vol. 1, pp. 432–439, 2003.

[64] I. Laptev and T. Lindeberg, "Interest point detection and scale selection in space-time," in *Scale Space'03*, (Isle of Skye, UK), June 2003.

[65] A. P. Witkin, "Scale-space filtering," in *International Joint Conference on Artificial Intelligence*, pp. 1019–1022, 1983.

[66] J. J. Koenderink and A. J. van Doom, "Representation of local geometry in the visual system," *Biological Cybernetics*, vol. 55, no. 6, pp. 367–375, 1987.

[67] T. Lindeberg, *Scale-Space Theory in Computer Vision*. Norwell, MA, USA: Kluwer Academic Publishers, 1994.

[68] T. Lindeberg, "Scale-space," in *Encyclopedia of Computer Science and Engineering* (B. Wah, ed.), pp. 2495–2504, Hoboken, New Jersey: John Wiley and Sons, 2009.

[69] J. Bigün, G. H. Granlund and J. Wiklund, "Multidimensional orientation estimation with applications to texture analysis and optical flow," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 8, pp. 775–790, 1991.

[70] J. Gårding and T. Lindeberg, "Direct computation of shape cues using scale-adapted spatial derivative operators," *International Journal of Computer Vision*, vol. 17, no. 2, pp. 163–191, 1996.

[71] T. Lindeberg and J. Gårding, "Shape-adapted smoothing in estimation of 3-d depth cues from affine distortions of local 2-d brightness structure," in

*Proceedings of the Third European Conference on Computer Vision*, vol. 1, (Secaucus, NJ, USA), pp. 389–400, Springer-Verlag New York, Inc., 1994.

[72] I. Laptev, *Local Spatio-Temporal Image Features for Motion Interpretation*. PhD thesis, Department of Numerical Analysis and Computer Science (NADA), KTH, 2004.

[73] L. Zelnik-Manor and M. Irani, "Event-based analysis of video," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, p. 123, 2001.

[74] I. Laptev and P. Perez, "Retrieving actions in movies," *IEEE International Conference on Computer Vision*, vol. 0, pp. 1–8, 2007.

[75] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision (darpa)," in *Proceedings of the 1981 DARPA Image Understanding Workshop*, pp. 121–130, April 1981.

[76] N. Dalal and B. Triggs and C. Schmid, "Human detection using oriented histograms of flow and appearance," in *9th European Conference on Computer Vision*, vol. 3952, pp. 428–441, 2006.

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

[78] S. Agarwal and A. Awan, "Learning to detect objects in images via a sparse, part-based representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1475–1490, 2004.

[79] R. Fergus and P. Perona and A. Zisserman, "Object class recognition by unsupervised scale-invariant learning," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, p. 264, 2003.

[80] T. Leung and J. Malik, "Representing and recognizing the visual appearance of materials using three-dimensional textons," *International Journal of Computer Vision*, vol. 43, no. 1, pp. 29–44, 2001.

[81] N. Cristianini and J. Shawe-Taylor, *An introduction to support vector machines: and other kernel-based learning methods*. Cambridge Univ Press, 2000.

[82] R. P. W. Duin, P. Juszczak, P. Paclik, E. Pekalska, D. De Ridder and DMJ. Tax, "Prtools, a matlab toolbox for pattern recognition," *Delft University of Technology*, 2004.

[83] I. Laptev, "Ivan Laptev–IRISA/INRIA Rennes France," July 26, 2009. http://www.irisa.fr/vista/Equipe/People/Laptev/interestpoints.html.