Video Object Tracking for Behavioral Analysis

by

Manu Chandraprabha Sukumaran Nair

(Under the Direction of Suchendra M. Bhandarkar)

Abstract

An object tracking-based behavior analysis technique for identifying Traumatic Brain Injury (TBI) among rodents is proposed. TBI is typically caused by external factors, resulting in severe structural brain damage, severely impacting cognitive reasoning and the overall functioning of the brain. Most existing methods used to identify post-TBI behavior use expensive methods in highly constrained environments like functional Magnetic Resonance Imaging (fMRI) to generate brain scans which are used to identify the nature and assess the extent of the TBI. In contrast, we propose a visual tracking-based method that computes the trajectory of the rodent and quantifies the differences in pre-TBI and post-TBI behavior via analysis of the variations in motion trajectories to assess the extent of TBI in a given rodent. We explore several feature representation schemes and machine learning models to capture the subtle cues that signify the variations in the motion trajectory with encouraging results.

INDEX WORDS: Object Tracking, Object Detection, Feature representation, Behavior Analysis, Traumatic Brain Injury

Video Object Tracking for Behavioral Analysis

by

Manu Chandraprabha Sukumaran Nair

A Thesis Submitted to the Graduate Faculty
of The University of Georgia in Partial Fulfillment
of the
Requirements for the Degree

Master of Science

Athens, Georgia

2019

Video Object Tracking for Behavioral Analysis

by

Manu Chandraprabha Sukumaran Nair

Major Professor:	Suchendra M. Bhandarkar
Committee:	Tianming Liu Yi Hong

Electronic Version Approved:

Suzanne Barbour
Dean of the Graduate School
The University of Georgia
May 2019

Video Object Tracking for Behavioral Analysis

Manu Chandraprabha Sukumaran Nair

2019

Dedication

Amma and Acha, this is for you

Acknowledgments

I would like to express my deepest gratitude to Dr. Suchendra M. Bhandarkar for his support and guidance throughout the tenure of my research. Thank you for pushing me to reach higher and do better. I would also like to express my sincere thanks to Dr. Tianming Liu and Dr. Yi Hong for their valuable feedback and insights. Your suggestions have been instrumental in enhancing this thesis work.

I am grateful to Dr. Lohitash Karumbaiah and his team at the Regenerative Bioscience Center, UGA, for providing me with the crucial data and information that was required for this research.

I also wish to thank the faculty and staff at the Institute for Artificial Intelligence for the opportunity provided to me. It has been a remarkable learning experience.

I am thankful to all my friends I made here for their feedback and encouragement. It was nice sharing the laboratory with some of you, and meeting the rest of you outside. I will always cherish the friendship we shared.

Special mention and thank you to my friend Dr. Arun C S Kumar, who was a doctoral student at the VPCL lab during the course of my thesis. Thank you

for motivating me, and for being the first person I could turn to. Your words of encouragement and criticism alike have been invaluable. You have been a great friend.

None of this would have possible without the love and support of my family. My father, I owe it all to you. I miss you more than I can say. My mother, who has been the pillar of strength and guidance throughout my life. I couldn't have finished this without you. My sister, who is always there for me, cheering me up, and always giving my work a good going over. My brother-in-law, who always believed in me, pushing me to be the best I can be. Thank you. Your love has seen me through it all.

Contents

List of Figures **x**

List of Tables **xv**

1 Introduction **1**

2 Related Work **6**

3 Dataset **12**

3.1 Video Annotation Tool 13

4 Video Object Tracking **16**

4.1 Object Detection 16

4.2 Object Tracking via Optical Flow 17

5 Proposed Approach **23**

5.1 LDOF Results 24

5.2 Kalman filter+LDOF Tracker Results 24

6 Classification Techniques: An Overview **27**

6.1 Support Vector Machines 27

6.2 Kernel SVM 28

6.3 Multilayer Perceptron 31

6.4 OneVsRest Classifier 32

7 Feature Representation **34**

7.1 Learning Behavioral Cues 34

7.2 Matching trajectories via Shape Context 38

7.3 Bag-of-Features 44

8 Results and Evaluation **56**

9 Conclusion **68**

9.1 Future Work 69

Bibliography **70**

List of Figures

3.1 **An illustration of the controlled environment.** 13

3.2 **Snapshots explaining the functioning of the video annotating tool (plugin). (a-c) explains how the video is loaded, and (d) explains how the mouse hover is done with a video interface, making the annotation of such large videos easier.** 15

4.1 **Examples of optical flow: columns 1 and 2 are the image pairs under consideration and their corresponding optical flow is plotted as a color map in column 3.** 19

4.2 **Overview of the Kalman filter pipeline.** 22

5.1 **Proposed Object Tracker Architecture.** 24

5.2 **LDOF results (Left: LDOF tracker results, Right: Corresponding object trajectory (red and violet indicate the start and end points of the trajectory, respectively)).** 25

5.3 **Kalman filter+LDOF tracker results (Left: Kalman filter+LDOF tracker results, Right: Corresponding object trajectory (red and violet indicate the start and end points of the trajectory, respectively)).** 26

6.1 **SVM Classification example.** 29

6.2 **Kernel SVM Classification example.** 30

6.3 **Multilayer Perceptron. n_0 forms input layer, $[n_1, n_2, n_3]$ forms the hidden layer, and n_4 forms the output layer.** . . . 32

7.1 **Illustration of trajectory parameters considered, using first five trajectory positions for the pre-TBI case of rodent subject A3. (x_1, y_1) through (x_5, y_5) denote the subject positions at consecutive time instants, θ_1 through θ_4 denote the angle between consecutive paths in the trajectory, and d_1 through d_4 denote the trajectory path lengths at consecutive time instants.** 35

7.2 **Relative trajectory distance illustrated using first five trajectory positions of pre-TBI and post-TBI cases of rodent subject A3. P_1 through P_5 and $P_1^{'}$ through $P_5^{'}$ denote the subject positions at consecutive time instants for the pre-TBI and post-TBI cases, respectively. d_1 through d_4 denote the relative trajectory Euclidean distance at consecutive time instants.** 36

7.3 **Probability density functions (PDFs) for relative trajectory distances for pre-TBI *vs.* pre-TBI, post-TBI *vs.* post-TBI and pre-TBI *vs.* post-TBI cases.** 38

7.4 **Shape Context computation and matching. (a, b) Sampled edge points of two shapes. (c) Diagram of log-polar histogram bins used in computing the shape contexts. We use 5 bins for log r and 12 bins for θ. (d-f) Example Shape Contexts for reference samples marked by o, $\diamond$, $\triangleleft$ in (a,b). Each Shape Context is a log-polar histogram of the coordinates of the rest of the point set measured using the reference point as the origin (Dark = large value). Note the visual similarity of the Shape Contexts for o and $\diamond$, which were computed for relatively similar points on the two shapes. By contrast, the Shape Context for $\triangleleft$ is quite different. (g) Correspondences found using bipartite matching, with costs defined by the χ^2 distance between histograms. (Figure courtesy: Belongie et al. [22]).** 40

7.5 **Shape Context Sample Results: $A3(pre-TBI)$ *vs.* rest (a-f), and $A3(post-TBI)$ *vs.* rest (g-i).** 42

7.6 **Trajectory Pairs (Left: pre-TBI, Right: post-TBI) of 3 subjects.** 49

7.7 **Left: A Bag-of-Feature example using rodent pose, Right: The Bag-of-Feature histogram for the subject, using pose as the event for feature representation.** 50
7.8 **Temporal BoF histograms (based on episode distance) for the same rodent subject A3 (First row: pre-TBI and post-TBI (Tracking results), Second row: pre-TBI and post-TBI (Ground Truth)).** 51
7.9 **Temporal BoF histograms (based on rate of change in episode distance) for the same rodent subject A3 (First row: pre-TBI and post-TBI (Tracking results), Second row: pre-TBI and post-TBI (Ground Truth)).** 52
7.10 **Temporal BoF histograms (based on rate of change in pose) for the same rodent subject A3 (First row: pre-TBI and post-TBI (Tracking results), Second row: pre-TBI and post-TBI (Ground Truth)).** 53

8.1 **Trajectory Pairs (Left: pre-TBI, Right: post-TBI) of 3 subjects.** 58
8.2 **tSNE distributions of (i) position coordinates, distance vectors and slope vectors of estimated trajectory (pre-TBI and post-TBI) and (ii) with pose included.** 59
8.3 **Shape Context matching for subject A3 (pre-TBI and post-TBI).** 60

8.4 **Left: A Bag-of-Feature example using rodent pose, Right: The Bag-of-Feature histogram for the subject, using pose as the event for feature representation.** 61

8.5 **BoF Trajectory Distance histograms for subject A3 (Time Series) (a) pre-TBI (Tracking), (b) pre-TBI (GT), (c) post-TBI (Tracking), (d) post-TBI (GT), (e) pre-TBI *vs.* post-TBI (Tracking) and (f) pre-TBI *vs.* post-TBI (GT).** 63

8.6 **BoF Distance rate (acceleration) histograms for subject A3 (Time Series) (a) pre-TBI (Tracking) (b) pre-TBI (GT), (c) post-TBI (Tracking) (d) post-TBI (GT),(e) pre-TBI *vs.* post-TBI (Tracking) and (f) pre-TBI *vs.* post-TBI (GT).** . 64

8.7 **BoF Pose histograms for subject A3 (Time Series) (a) pre-TBI (Tracking), (b) pre-TBI (GT), (c) post-TBI (Tracking), (d) post-TBI (GT), (e) pre-TBI *vs.* post-TBI (Tracking) and (f) pre-TBI *vs.* post-TBI (GT).** 65

8.8 **BoF Pose rate histograms for subject A3 (Time Series) (a) pre-TBI (Tracking), (b) pre-TBI (GT), (c) post-TBI (Tracking), (d) post-TBI (GT), (e) pre-TBI *vs.* post-TBI (Tracking) and (f) pre-TBI *vs.* post-TBI (GT).** 66

List of Tables

7.1 **Classification accuracy using Relative Trajectory Distances.** 36

7.2 **Relative Trajectory Distances (pre-TBI** *vs.* **post-TBI).** . . . 37

7.3 **Mean and Standard Deviation for the Relative Trajectory Distance values.** . . . 37

7.4 **Affine Cost, Shape Context Cost and Error Values for the Shape Context computation for the rodent subject** $A3(pre-TBI)$ *vs.* **rest case.** . . . 43

7.5 **Affine Cost, Shape Context Cost and Error Values for the Shape Context computation for the rodent subject** $A3(post-TBI)$ *vs.* **rest case.** . . . 43

7.6 **Euclidean distances between Temporal BoF descriptors (Trajectory distance episodes) for the rodent subject A3 (pre-TBI** *vs.* **pre-TBI). Both rows and columns correspond to pre-TBI episodes.** . . . 54

7.7 **Euclidean distances between Temporal BoF descriptors (Trajectory distance episodes) for the rodent subject A3 (post-TBI *vs.* post-TBI). Both rows and columns correspond to post-TBI episodes.** 55

7.8 **Euclidean distances between Temporal BoF descriptors (Trajectory distance episodes) for the rodent subject A3 (pre-TBI *vs.* post-TBI). Rows correspond to pre-TBI episodes and columns correspond to post-TBI episodes.** 55

8.1 **Relative Trajectory Distances (pre-TBI *vs.* post-TBI) for 6 rodent subjects.** 57

8.2 **Mean and Standard Deviation for the Relative Trajectory Distance cases.** 59

8.3 **Classification accuracy using Relative Trajectory Distance (D: Trajectory distance, DR: Distance rate, P: Pose distance, PR: Pose rate.)** 67

Chapter 1

Introduction

Behavior analysis is the science of behavior pertaining to humans and non-human animals. It is a scientific study which seeks to understand the principles underlying how biological, pharmacological and environmental factors influence behavior. Behavior analysis has been a challenging topic of research within the computer vision research community.

The observation and assessment of the behavior of animals such as laboratory rodents has made significant contributions towards our understanding of human cognition and neurological mechanisms. Rodent models are also used to study the behavioral impairment after an injury, such as Traumatic Brain Injury (TBI). TBI, caused by sudden trauma to the head by an external force, results in disruption in the normal functioning of the brain, and can be temporary if the TBI is mild, and permanent if the impact on the brain is particularly violent. TBI can occur during accidents, and are especially prevalent during motor-vehicle accidents. Everyone

is at risk for TBI, but the rates are generally higher among older adults, children, prisoners and persons in combat zones and athletic arenas such as football and boxing. TBI can also occur during incidents of domestic violence. The very nature of brain injuries make tracking them especially difficult. In the case of TBI, the injury need not be penetrative such as a bullet wound or a sharp object that penetrates into the brain tissue, but can be purely intra-cranial with no visible external bruises. Moreover, these injuries are additive. In scenarios where the brain is subjected to multiple external assaults, together or spaced apart in time, each impact will compound the damage from the previous ones. The impact of this injury is devastating on the injured as it encompasses a broad spectrum of symptoms and disabilities, a wide range of physical and psychological effects, and at times can even result in death.

According to Taylor et al. [2], TBI is a major cause of death and disability in the United States, contributing to about 30 percent of all injury-related deaths. An estimated 2.8 million people in the United States sustain a TBI annually. Of these, 2.5 million are involved in Emergency Department (ED) visits where they are treated and released, another 282,000 require hospitalizations, and 56,000 die of TBI-related causes. This amounts to an estimated 153 deaths a day from TBI-related causes. The highest rate of TBI-related deaths is observed among adults aged 75 or older, with the number of TBI-related deaths greater among older males than among older females, followed by children aged 0-4 years, and adolescents aged 15-24 years. Falls are the primary cause for TBI-related hospitalizations among adults aged 75 years or older, accounting for an increase in TBI-related ED

visits since 2007, which suggests an urgent need to enhance fall-prevention efforts among the older adult population.

Motor-vehicle crashes, the leading cause of TBI-related deaths in 2007, have decreased substantially by 2013, suggesting significant progress on that front. As of 2013, falls are the leading cause of TBI, followed by getting struck by or striking against an object, and motor-vehicle crashes. Falls remain the leading cause for TBI-related deaths, followed by intentional self-harm as the second leading cause. During the span of 6 years, from 2007-2013, rates of TBI-related ED visits increased by 47 percent, hospitalization rates decreased by 2.5 percent and death rates decreased by 5 percent. Due to the observed rise in the incidents involving TBI, it is prudent to conduct more studies that investigate the long-term effects of such a chronic and life-changing injury. The cost of TBI treatment is prohibitive for the injured, as well as his/ her family. The effects of TBI, if one survives it, can last for a few weeks or months to the rest of one's life. Coupled with loss of productivity and mounting costs of health-care, TBI can have lasting effects on all persons involved.

This thesis addresses the analysis of the behavior of rodent subjects impacted by TBI. The proposed approach tries to analyze the impact of TBI on the given rodent subject by studying the real-time movement behavior of the rodent subject, which is then evaluated against a set of available benchmark parameters. This is achieved by tracking the motion of the rodent subject, treating it as the object of interest (OOI) in a given video.

The analysis of motion in videos is a pre-requisite for a variety of applications such as the analysis of the semantic content in videos, automated video editing tasks, summarization of videos etc. The three major steps in video-based behavior analysis are (1) finding the object(s)-of-interest in the video, (2) tracking the object(s)' movement from frame to frame, and (3) analysis of the object(s)' tracks to recognize their behavior. Video tracking refers to the technique for tracking the movement of object(s) in a given video. It has a variety of uses, such as human-computer interaction, security and surveillance, video communication and compression, augmented reality, video-based traffic control, medical imaging and video editing.

Video tracking is a computationally intensive process due to the fact that there is a lot of information contained in the video. The need for object recognition for aiding the tracking process makes the underlying computation even more complex. The primary objective of video tracking is to associate object(s)-of-interest across the frames of the video. The task is rendered more complex if the object being tracked moves much faster relative to the video frame-rate, and/ or its orientation varies significantly over time. Under such situations, the normal approach taken by video tracking methods is to employ a motion model that can provide a description of how the object appears subject to different kinds of motion. The motion model is a 2D transformation in the case of tracking a planar object. In the case of tracking a rigid 3D object, the 3D position and orientation of the object is exploited by the motion model.

The proposed approach involves an efficient combination of optical flow computation and Kalman filtering. The proposed work utilizes a locally generated dataset as the ground truth. The proposed work models the underlying uncertainty in the problem domain by posing the problem as one involving object masking and temporal regression, which increases the robustness of the result. The proposed system is implemented as an architecture which combines optical flow computation with Kalman filtering. The video dataset for our experiments was provided by Regenerative Bioscience Center (RBC), UGA, and the ground truth data for training the tracking model consists of a subset of video frames generated from this dataset. We have also developed an annotation tool for generating the ground truth annotations. The proposed approach aims to come up with an exact detection of the OOI. The performance of the proposed method is tested on randomly selected video frames generated from the video dataset, which were not considered during the training phase.

Chapter 2

Related Work

Over the past three decades, there have been several works related to object tracking. The suitability of an algorithm for the given object tracking task greatly depends on the following factors: object appearances, object shapes, number of objects, object and camera motions, and illumination conditions. The main goal of a video object tracking algorithm is to delineate the motion of the object by localizing its position in every frame of a video. This involves detecting the object and establishing correspondences between the object instances across the frames of the video. These two tasks can be done either separately or jointly. In the first case, the possible region(s) for the object occurrence in the video frames are obtained by means of an object detection algorithm, and this information is then exploited by the tracker to determine the object correspondence across the frames. In the second case, the object regions and correspondences are jointly estimated

by means of an iterative updating of the object location and region information obtained from the previous frame(s).

In either of the aforementioned tasks, the object is represented using a shape and/or appearance model. The choice of the object representation model defines the changes the object representation can undergo. For instance, an object represented by a point-based model can have only translational motion. In case of geometric representation of an object, parametric models such as affine or projective transformations may be used. These models can be used for approximating the motion of rigid objects. In the case of non-rigid objects, the best choice is contour representation, and both parametric and non-parametric models can be used for motion specification. The methodology of object tracking may be broadly classified into three categories viz., kernel tracking, point tracking and silhouette tracking. Kernel tracking makes use of template and density-based or multi-view appearance based models. Point tracking, on the other hand, makes use of deterministic or statistical methods. Silhouette tracking relies on contour evolution and matching of contour-based shape models.

The mean-shift algorithm (MSA) [1] and cam-shift algorithm (CSA) [3] are two of the earliest works in the field of video tracking; both can be categorized as kernel-based tracking methods. The MSA tries to find the center of mass, or in other words the densest sub-region within the entire given object region. Given a set of data samples, which could be formulated by a probability density function (PDF) in R^N, the MSA tries to find the modes of the PDFs connecting the data samples. The PDF is based on an underlying feature space representation. The

feature space used to define the PDF could be the color space, scale space, time space etc. The CSA is an iterative version of MSA. The CSA performs the MSA on each frame of the video. While the CSA utilizes continuously adaptive probability distribution, the MSA utilizes a static distribution. Thus, the distribution followed by the CSA varies over each frame of a video whereas the one used by the MSA varies only when there is a significant variation in the feature space. In the CSA, the mode of moments is computed using spatial moments whereas the MSA utilizes the current and target distributions, and tries to arrive at the maximum ratio of the two. Both the MSA and CSA are good for tracking objects which follow a linear motion model, assuming the object shape remains constant throughout the motion. They are not suitable in cases where the object shape changes continuously, or the motion is not uniform.

Image intensity or color features are usually used for creating the templates for kernel tracking. The intensity of an image is very sensitive to illumination changes. Hence, intensity-based gradients and color-based gradients are preferred candidates for the underlying features [4]. Since the MSA and CSA use a brute-force search, they have high computation cost. An efficient approach for template-based kernel tracking is proposed by Schweitzer et al. [5]. Feiguth and Terzopoulos [6] generate object models by utilizing the color histogram within the rectangle containing the object region to model the kernel.

Point feature tracking has been popular since the 1970s. The Moravec interest operator [7], Harris interest point detector [8], KLT detector [9], and SIFT detector [10] are some examples of point feature trackers. Optical flow has been shown

to be one of the most successful features for tracking moving objects. Some of the popular approaches for optical flow computation include Horn and Schunck [11], Lucas and Kanade [12], Black and Anandan [13], Szeliski and Couglan [14] and Large Displacement Optical Flow [15].

The recent works proposed by Zheng et al. [16], Zhao et al. [17] and Yin et al. [18] rely on data from sensors mounted on the body of the subject, and while they are not noise-free, they are far more accurate for the class of problems tackled in those respective works. The technique of Zheng et al. [16] relies on using large amount of annotated data for the application of data mining. However, our proposed model does not have large amounts of data and our criteria for categorization is not quite straight forward or even describable in the first place, as in Zheng et al. [16]. The method used in Zheng et al. [16] is similar to our method in that both rely on Kalman filtering and post processing to remove noise; however, while Zheng et al. [16] relies on pre-computed trajectories for mining, our method computes trajectories from raw data. The proposed method also attempts to match motion trajectories using Shape Context, similar to that of Zheng et al. [16]; the major difference is that our trajectories unlike Zheng et al. [16] are repetitive and noisy and thus are challenging. While the applications are different, the overall pipeline of Zheng et al. [16] is very similar to our proposed one, except that we do it without supervision and with the help of very little data.

Zhao et al. [17] propose several methods to use pre-computed trajectories from sensors for performing data compression. They present frameworks that exploit reference based spatial and spatio-temporal compression. Though a variety of

frameworks are engineered in Zhao et al. [17], they are mostly built for data compression and similar to Zheng et al. [16], and also rely on availability of large amounts of data and supervision.

Yin et al. [18] combine activity trace from activity sensors with GPS trace to model generic human activities. The human activities modeled in Yin et al. [18] are more sophisticated, describable, and semantically meaningful, whereas the trajectories that we compute from the rodent's motion have no clear semantic categorization. In addition, as noted above, our method approximates trajectories without motion sensors by just using image sequences and is bound to be much more noisy.

Moreover, we attempt to find differences in trajectories, as opposed to learning or modeling a known (ground truth) categorization of trajectories [16, 17, 18]. We also face another major challenge: even if we find ways to procure large amounts of data, we will not be able to obtain annotations, as the purpose of the research itself is to identify potential cues that human subjects fail to recognize, so that it could form the basis for automatic categorization. Thus there is no straightforward way to generate ground truth annotations for our data that can be quantified as a specific behavior or specific action.

In addition, unlike in [16, 17, 18], our goal is to eliminate the need for mounting sensors on the body of the rodent which would hamper its natural activity, thereby defeating the very purpose of this research. Thus, we rely on approximating the trajectories from infrared sensors, instead of mounting accelerometers or motion detection sensors to gather motion cues; the lack of motion sensors consequently

adds significant amounts of noise to the data. On the other hand, we obtain the data that represent the natural behavior of the rodent.

There have been several works which have tried to study the effects on laboratory rodents once they have been induced with TBI. Yu et al. [19] established that the velocity and impact of the Controlled-Cortical Impact (CCI) lead to either mild, moderate, or severe TBI in the laboratory rodents. The authors also claimed that rodents with the largest volume of damaged brain tissue exhibited the worst behavioral impairments. Washington et al. [20] claimed that CCI produces severity-dependent differences in lesion volume and also showed that CCI-impacted rodents do not show any behavioral changes in the open-field tests. In the work by Sarma et al. [21], the rodents were subjected to repeatable direct injury to the left fronto-parietal cortex via CCI, and the extent of TBI was studied by estimating the brain lesion volume.

Chapter 3

Dataset

Our dataset comprises of videos in which laboratory rodents are introduced to a controlled environment as shown in the Figure 3.1. The rodents were expected to solve or attempt to solve the puzzle of getting out of the glass box which is similar to a maze solving problem. The experiment was conducted on 6 rodent subjects, once before impacting them with TBI (i.e., pre-TBI case) and once after (i.e., post-TBI case), resulting in a total of 12 videos. Capturing changes in motion cues of a rodent pre-TBI *vs.* post-TBI is expected to give insights pertaining to the extent of impact of TBI on the rodent subject's behavior. The videos used in the research have a resolution of 240 $\times$ 320 pixels, and were recorded using an infrared (IR) camera. The infrared cameras were the primary choice for video acquisition as rodents are nocturnal, and their natural behavior is better observed under little to no light. The video dataset for our experiments was provided by Regenerative Bioscience Center (RBC), UGA.

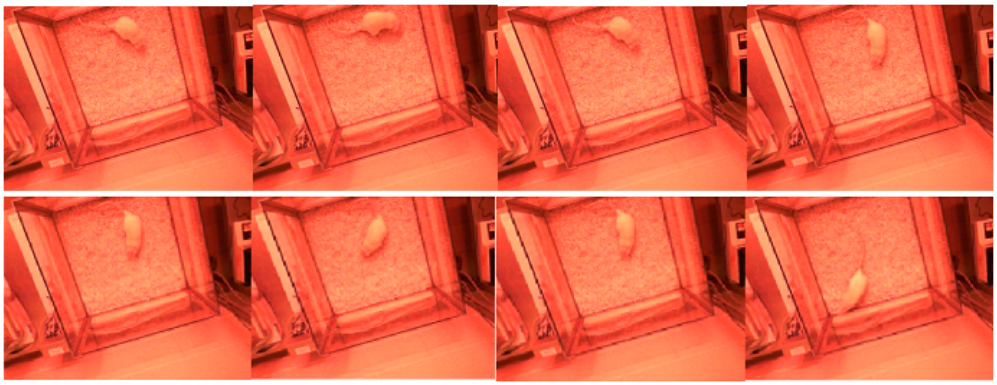

Figure 3.1: **An illustration of the controlled environment.**

3.1 Video Annotation Tool

For the purpose of annotating the object's position over the length of the video, an annotation module was developed using Tkinter, the Python GUI library. Most object annotation tools require the user to provide a bounding box that compactly encloses the object. In our case, it is infeasible, as we deal with over 40,000 frames containing the OOI across multiple videos. Thus, in order to perform the annotation, we developed a GUI scheme that procures the 2D points based on mouse hovering. Mouse hover simply requires the user to place the mouse cursor over the OOI and move it along with the OOI as the OOI moves in the remainder of the video. The scheme does not require mouse clicks or bounding boxes which are painstaking to annotate. The frame rate of the video is slowed down in the toolkit so that the user has sufficient time to follow the OOI (i.e., rodent subject) as accurately as possible. The video annotation tool also comes with the following

features: (a) an option to load the video from the directory through a GUI, (b) an option to select the format in which the annotations are to be saved to the system, (c) an option to enter the desired name for the output file, (d) options to change the frame rate of the video during the mouse hovering process thereby enabling the user to slow the video down further if necessary or to increase the speed of the video, and (e) an option to pause the video between frames. Figure 3.2 demonstrates the working of the video annotation tool.

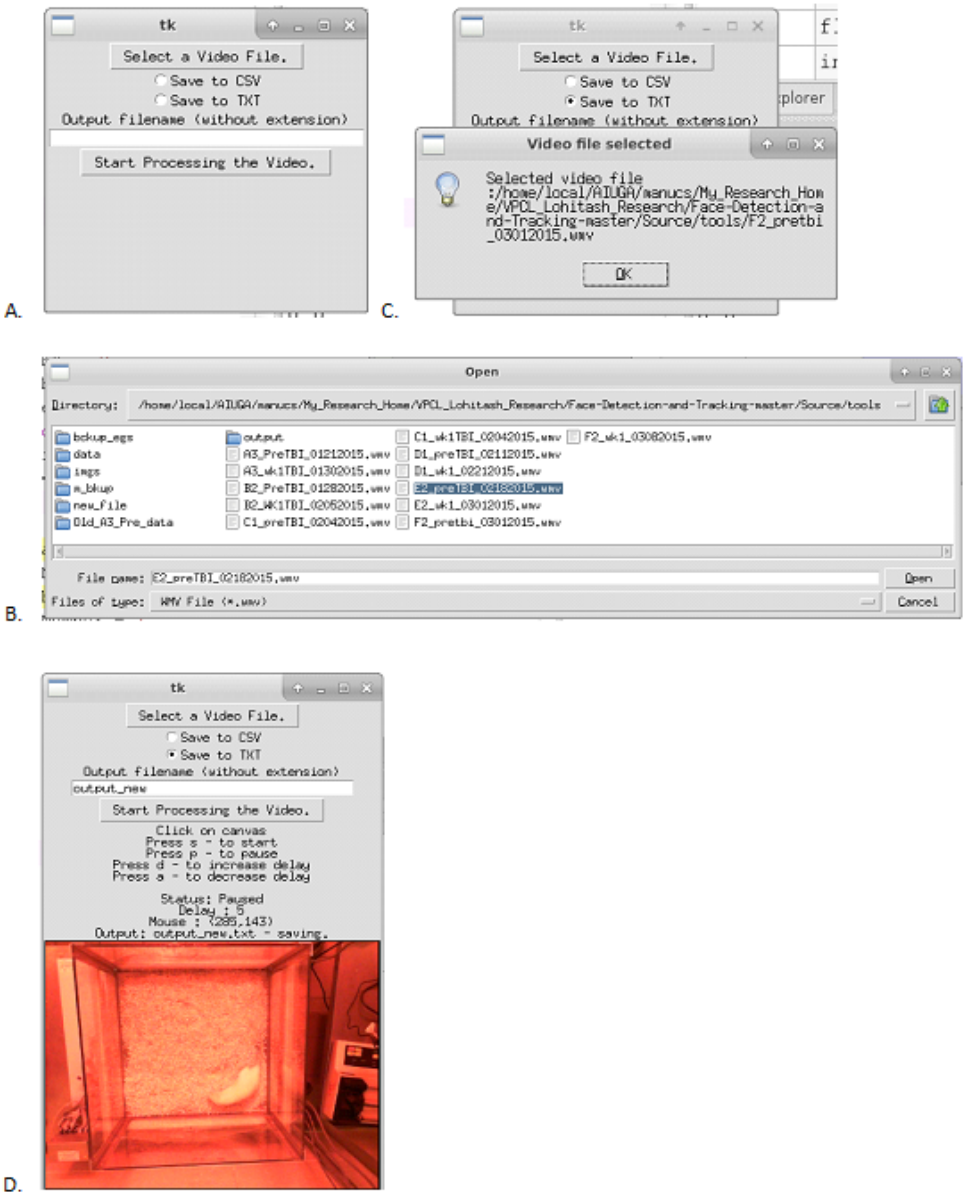

Figure 3.2: **Snapshots explaining the functioning of the video annotating tool (plugin). (a-c) explains how the video is loaded, and (d) explains how the mouse hover is done with a video interface, making the annotation of such large videos easier.**

Chapter 4

Video Object Tracking

4.1 Object Detection

Object detection is the task of finding the different objects in an image and classifying them. It is challenging mainly because of: (a) within-object intra-class variation; for example, a car though considered as a single object for the sake of object detection, is actually a large class of objects that could be further subclassified based on attributes such as the type of car (sedan, SUV, or hatchback), model, make, design, changes in appearances over time (vintage *vs.* new), to cite a few; (b) variation in appearance due to pose or viewpoint of the object; for example, the front view of a truck is completely different from its rear view. Humans inherently apply context learned over time to identify and associate these subcategories with the same object category, but it is very challenging to design such an object detection system; and finally (c) rodents (the class of objects that we deal

with, in this thesis) have non-rigid shapes and are particularly challenging to detect since the underlying shape cues are highly varying and extremely challenging to model, and requires more complex reasoning.

4.2 Object Tracking via Optical Flow

4.2.1 Overview of Optical Flow

Optical flow or optic flow captures the apparent motion of object(s), surface(s) or edges(s) of a scene when the observer is in motion relative to the scene. Imagine that the observer is traveling in a vehicle. The observer views the buildings, trees, people etc. as moving objects irrespective of whether these objects are actually in motion or not. Also, the objects closer to the observer appear to move faster compared to objects farther away.

The computed optical flow depends on both the relative speed and the distance between the observer and the target. The magnitude of the optical flow doubles in either case; whether the observer's speed is doubled, or the distance between the target and the observer is halved. The optical flow is at its largest when the target object's motion is tangential to the observer's motion, or when the target object is vertically above or below the observer. In the case of an object that is in front of the observer, though the object may seem to be at rest relative to the observer, the edges of the object, which are not exactly in front of the observer will seem to move, and the object will appear to grow in size. Figure 4.1 shows an example of optical flow. The Lucas-Kanade (LK) optical flow and Large Displacement Optical

Flow (LDOF) are two popular optical flow algorithms used extensively in several computer vision applications.

The literature pertaining to object tracking can be categorized into two broad categories: point feature-based methods, and continuous optical flow-based methods. Point feature-based methods are coarse-grained but fast to compute whereas continuous optical flow-based methods are fine-grained and slower (since correspondence is computed for each pixel in the source image). Since it computes pixel-to-pixel correspondence, the LDOF algorithm is usually very slow. However, the LDOF algorithm has been shown to be successful in object tracking applications where the object's features vary significantly from those of the object's background.

4.2.2 Large Displacement Optical Flow

The LDOF algorithm employs a coarse-to-fine variational framework for optical flow estimation between two image frames. The LDOF algorithm incorporates descriptor matches in addition to the standard brightness and gradient constancy constraints, for dealing effectively with large displacements of small as well as large structures. Descriptor matches are obtained by matching densely sampled histogram of oriented gradients (HOG) descriptors in the two images. The displacement field is then computed by minimizing the energy functional.

In order to minimize the energy functional using a coarse-to-fine pyramidal scheme, at each level in the pyramid, the non-linear Euler-Lagrange equations are solved via multiple fixed-point iterations. At each iteration, the sparse linear

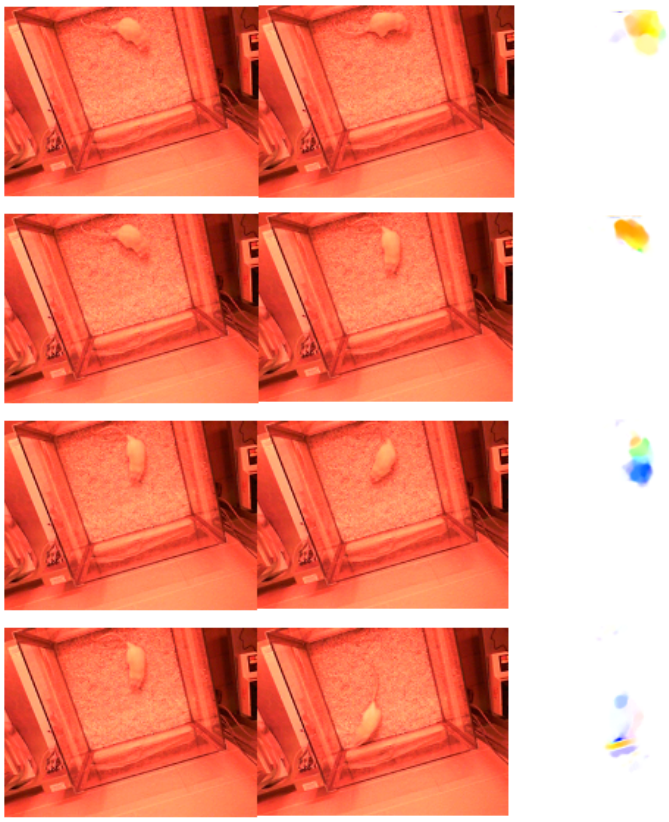

Figure 4.1: **Examples of optical flow: columns 1 and 2 are the image pairs under consideration and their corresponding optical flow is plotted as a color map in column 3.**

system is solved efficiently via successive over-relaxation (SOR) steps. Intuitively, at each pyramid level, the pixel displacement estimates and the pairwise/ unary robust pixel potential terms are updated iteratively, i.e., given the current pixel motion estimates and the displacement field, the robust unary and pairwise potential terms are computed, and vice versa. Once a fixed-point for the computation at a particular level in the pyramid has been reached, the computed flow field is up-sampled to initialize the computation at a finer pyramid level. Figure 4.1 shows an example output of the LDOF algorithm.

In the proposed framework, the LDOF algorithm is used for extracting the temporal cues underlying the trajectory of the OOI in the video. This is accomplished by computing the optical flow-based correspondences of the OOI across the video frames which then serve as the backbone for the Kalman filter-based tracking algorithm.

4.2.3 Kalman Filtering

The Kalman filter is a traditional approach for object tracking. Proposed by Rudolph Kalman in the 1960s, Kalman filter provides a computational means for estimating and predicting the state of a continuously changing system. Consider the example of a GPS system in a vehicle. The measurement of the GPS is affected by noise (usually called the measurement noise). The strength of the noise can be known by knowing the variance associated with the system state (denoted by σ^2), assuming a normal distribution for the noise. The variance associated with

the GPS sensor is precomputed and is accounted for in the Kalman filter-based tracking algorithm.

Now, imagine the case of the sensor losing the GPS signal. The navigation system is totally unclear as to where it is now positioned. Thus, there clearly is an uncertainty with regard to the state of the GPS location. When the variance associated with the GPS location is quite large, its probability distribution curve becomes almost flat. In this situation, a Kalman filter will exploit the previous state of the system and use a set of system variables along with a likelihood probability to predict the next possible state of the system. Figure 4.2 shows the schematic diagram of Kalman filter. The current object position is fed as input to the Kalman filter. The Kalman predictor stage makes use of the current state and measurement noise covariance in order to compute an *a priori* estimate of the next state. The Kalman corrector stage computes an updated measurement in order to obtain a better *a posteriori* estimate of the next state. The predictor and corrector stages operate hand-in-hand in a cyclic manner.

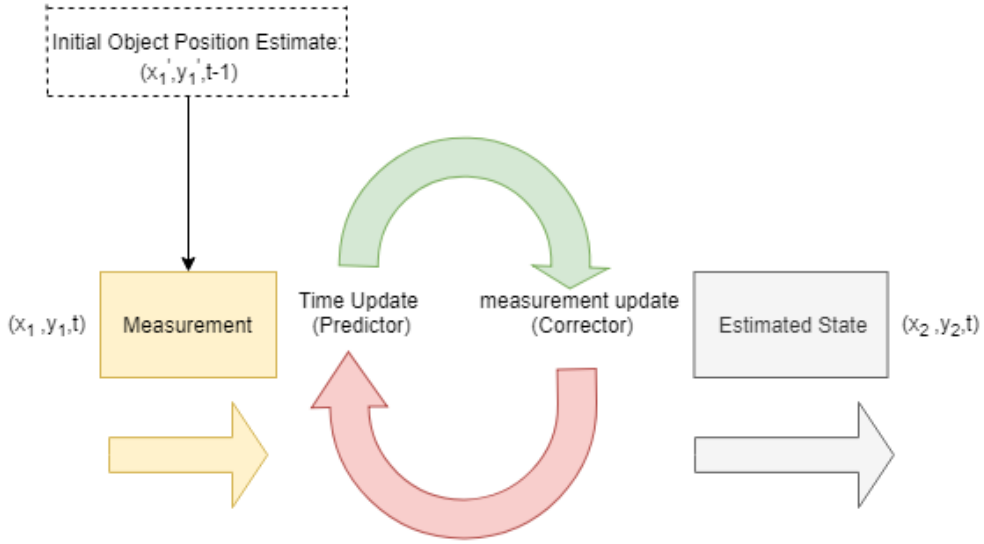

Figure 4.2: **Overview of the Kalman filter pipeline.**

Chapter 5

Proposed Approach

The proposed approach makes use of a combination of two well known methods in computer vision for the video tracking problem i.e., LDOF and the Kalman filter. The proposed architecture is shown in Figure 5.1. In our experiments, the estimated next state is stored, and in case the next position estimated by the LDOF stage happens to be an outlier, the previously stored estimated next state is used as the input to the Kalman filter stage. In order to do the outlier detection, for each given trajectory (pre-TBI or post-TBI), we use the mean and the standard deviation computed using the corresponding ground truth positions' data. In order to compare the performance of standalone LDOF-based tracker and the proposed Kalman filter+LDOF tracker, the respective tracking results and the corresponding object trajectories are observed over 85 consecutive frames as shown in Figure 5.2 and Figure 5.3.

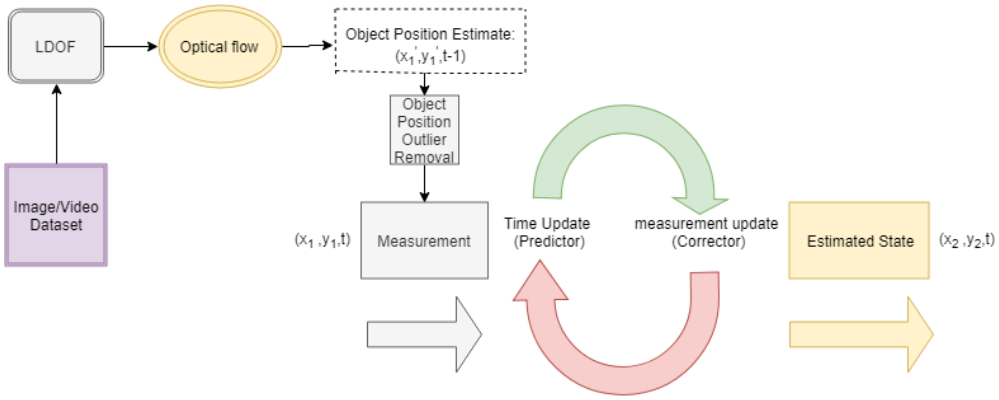

Figure 5.1: **Proposed Object Tracker Architecture.**

5.1 LDOF Results

The results of the LDOF algorithm for the pre-TBI and post-TBI cases are shown in Figure 5.2. As we can see from the results, the LDOF algorithm loses track of the object in case of illumination noise, and also during sudden pauses in movement, since the LDOF algorithm tracks the object using optical flow that requires the object to be in constant motion.

5.2 Kalman filter+LDOF Tracker Results

The Kalman filter+LDOF tracker results are shown in Figure 5.3. The Kalman filter uses a probability model for predicting the object position based on the object's previous state: which consists of its previous position, previous velocity, and previous acceleration. As we can see from Figure 5.3, the Kalman filter+LDOF

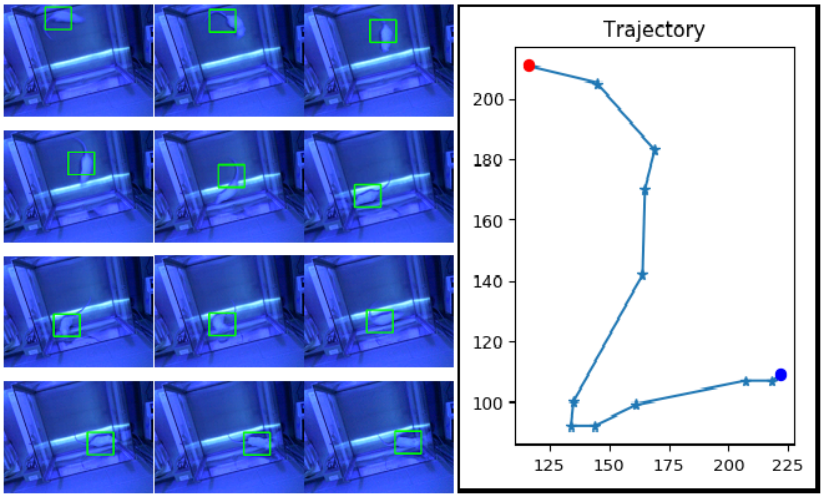

Figure 5.2: **LDOF results (Left: LDOF tracker results, Right: Corresponding object trajectory (red and violet indicate the start and end points of the trajectory, respectively)).**

tracker is more accurate in tracking the trajectory of the object, and it even captures a minor change in motion direction towards the end of the observed trajectory.

The difference in behavior of the trajectory of a rodent in its post-TBI case, as compared to its pre-TBI case, can be effectively studied only if the behavioral cues that distinguish one from the other can be identified. In order to evaluate the reliability of the behavioral cues obtained, we also require efficient classification algorithms which can verify how far these cues are helpful in classifying whether a given rodent subject belongs to pre-TBI case or post-TBI case.

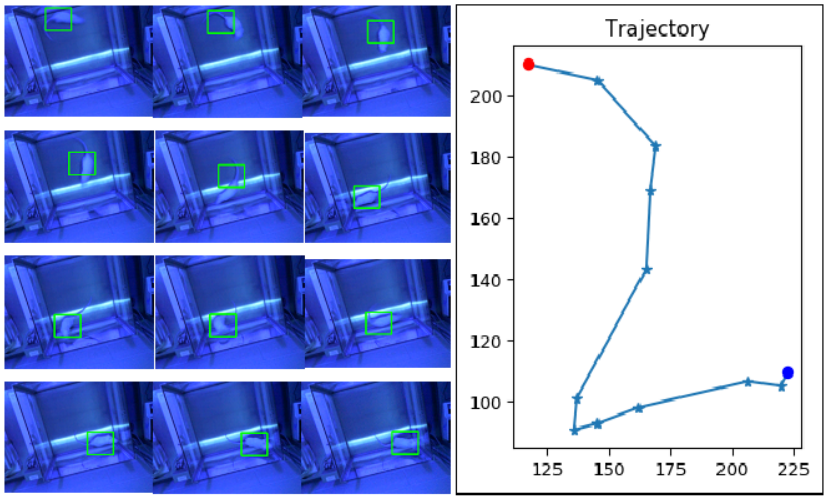

Figure 5.3: **Kalman filter+LDOF tracker results (Left: Kalman filter+LDOF tracker results, Right: Corresponding object trajectory (red and violet indicate the start and end points of the trajectory, respectively)).**

Chapter 6

Classification Techniques: An Overview

The most popular classification algorithms which can be used as evaluation prototypes for classification based on behavioral cues are Support Vector Machine (SVM), Multilayer Perceptron (MLP) and OneVsRest classifier.

6.1 Support Vector Machines

A Support Vector Machine is a discriminative classification technique that deals with high-dimensional data. It is a classification algorithm that works well on small data sets, providing significant accuracy while requiring less computational power. The goal of a SVM is to design a hyperplane that best splits or classifies the data points into the given classes, in such a way that maximizes the margin between the data points of different classes. In other words, we can split the data

in several ways, but there will be only one way to split it while maintaining the widest gap or margin between the classes, which is what SVM accomplishes.

The hyperplane computed by the SVM classifies the classes/ groups while optimizing the margin between the nearest data points between the distinct classes. The points that are nearer to this hyperplane are called support vectors. The positions of these support vectors are crucial in determining the position of the hyperplane. The distance between the support vectors and the hyperplane needs to be as large as possible. If an existing support vector moves or gets deleted, the optimal hyperplane will also change.

When dealing with the classification of two classes in 2D feature space, the hyperplane can be visualized as a line that maintains the maximum distance between the nearest data points of these two classes. With a larger dimensional feature space, the hyperplanes are multidimensional. Also, with a larger number of classes, more hyperplanes are needed. However, once we are dealing with more than three classes in high-dimensional feature space, it can be quite hard or impossible to visualize all the hyperplanes. Maximizing the margin is a constrained optimization problem, and can be solved using the technique of Lagrange multipliers. Figure 6.1 illustrates how SVM classifier works on data.

6.2 Kernel SVM

When dealing with data that is not linearly separable, we utilize the idea that linearly non-separable features mostly become separable once they are projected

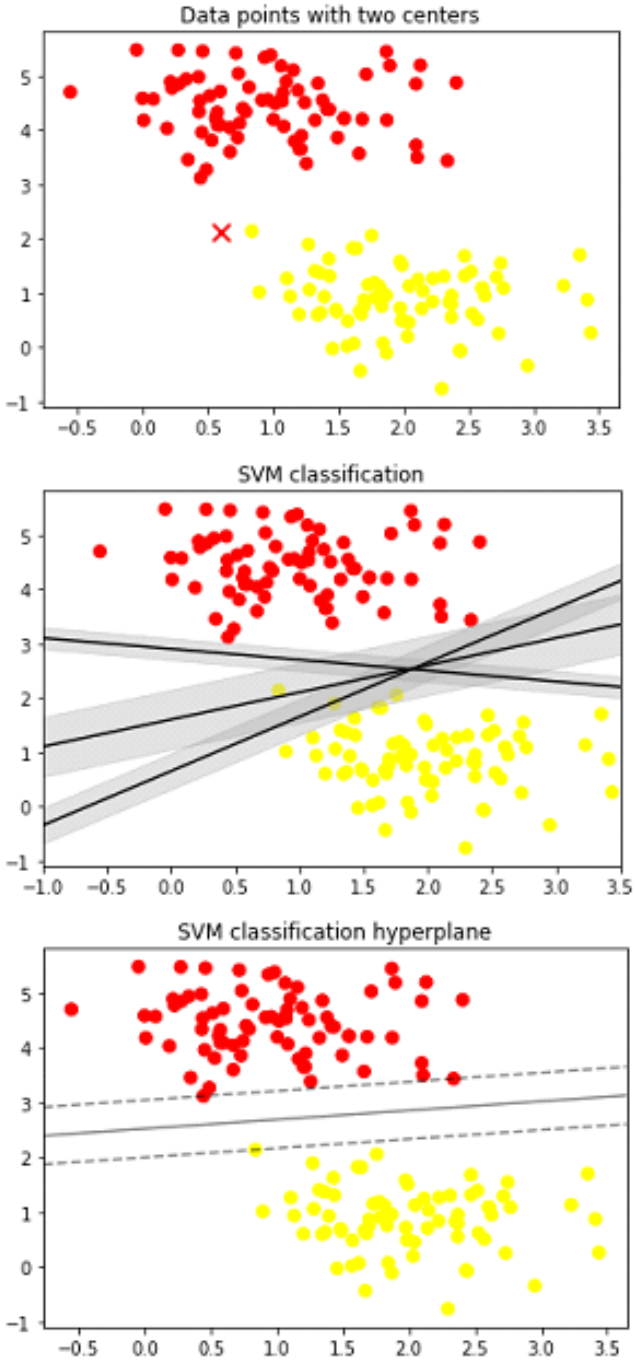

Figure 6.1: **SVM Classification example.**

to high-dimensional space. This is the concept behind SVM using kernel functions. In our experiments, we used Radial Basis Function (RBF) [40] as the kernel. Figure 6.2 illustrates how Kernel SVM classifier works on data.

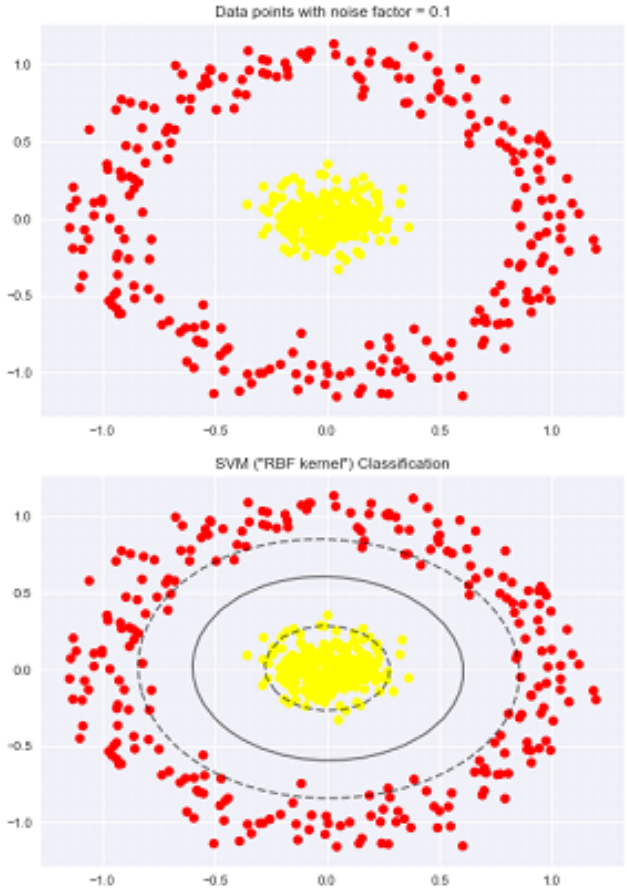

Figure 6.2: **Kernel SVM Classification example.**

6.3 Multilayer Perceptron

A perceptron is a classifier which classifies the input by separating two categories using a straight line. The input to a perceptron is a feature vector 'u' (with an added bias denoted by 'b'). The MLP computes the output 'v' as a linear combination of input weights and sometimes operating on it using a non-linear function $f(u)$. The output of a perceptron is of the form:

$$v = w * u + b \tag{6.1}$$

where, $w = [w_{11}, w_{12}, w_{21}, w_{22}]$ is the weight vector.

A multilayer perceptron is a feed-forward neural network consisting of multiple perceptrons, that is widely used as a classifier and logistic regressor. MLP consists of a minimum of three layers, each having atleast one perceptron. The first layer is known as the input layer, the second layer is the hidden layer and the third layer is known as the output layer. The input layer receives the input signal, and the output layer computes the prediction or decision corresponding to the input signal. The number of hidden layers for MLP can vary depending on the problem and the data it is applied to. The hidden layer(s) form the heart of the computation involved in a MLP, and constitute for the nonlinear nature of MLP. MLP can be regarded as a nonlinear function which can be used to approximate any continuous function. Figure 6.3 shows a MLP having a single hidden layer. In Figure 6.3, n_0 forms the input layer, the layers n_1, n_2 and n_3 form the hidden layer, and n_4 forms the output layer.

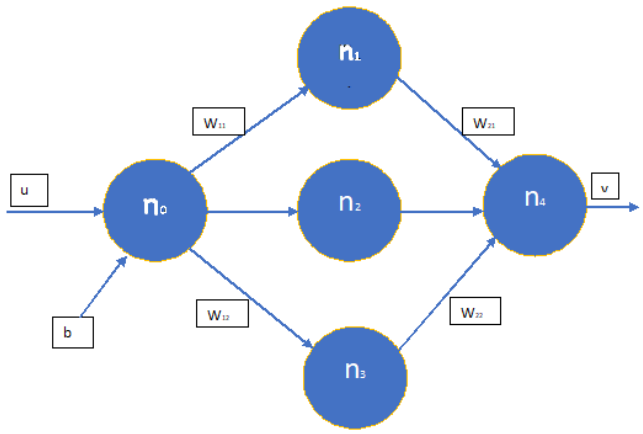

Figure 6.3: **Multilayer Perceptron.** n_0 **forms input layer,** $[n_1, n_2, n_3]$ **forms the hidden layer, and** n_4 **forms the output layer.**

Given an input u of dimension m and a target y, a MLP learns an approximation mapping,

$$f(.) : R^m \rightarrow R^n \tag{6.2}$$

where, m is the dimension of input and n is the dimension of the output.

6.4 OneVsRest Classifier

OneVsRest classifier is a popular approach to solve multi-label classification problems. OneVsRest classifier is a technique which trains N binary classifiers on one particular class at a time, leaving out the rest of the classes. One major disadvan-

tage is that if the number of classes are huge, it takes a proportionally long time period for training and prediction.

Chapter 7

Feature Representation

7.1 Learning Behavioral Cues

In order to analyze the aspects of the object position that are relevant to classify the subject in the given video as pre-TBI or post-TBI, we employed SVM, Kernel SVM and MLP. The parameters considered for the evaluation are position vector (containing respective position coordinates along the the various trajectories windowed over a fixed time duration), distance vector (containing respective distances covered every unit time along the various trajectories windowed over a fixed time duration), pose vector (containing respective object poses or orientations windowed over a fixed time duration) and slope vector (containing respective slopes of object trajectory paths windowed over the fixed time duration). The results of the study are shown in Table 7.1. The accuracy is computed as the ratio of the number of test samples whose labels (i.e., whether it is pre-TBI or post-TBI) are

correctly predicted to the total number of test samples. We found that the distance vector and the slope vector generated from the object trajectory were better features for the classification of the subject as pre-TBI or post-TBI as compared to object positions over the time duration of the trajectory. However, most of the results are random in nature (with accuracy as low as 0.5), and hence we require better features for efficient learning of the trajectory behavior.

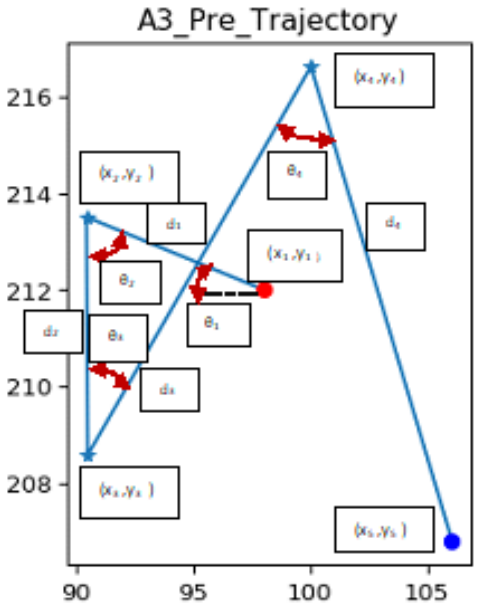

Figure 7.1: **Illustration of trajectory parameters considered, using first five trajectory positions for the pre-TBI case of rodent subject A3. (x_1, y_1) through (x_5, y_5) denote the subject positions at consecutive time instants, θ_1 through θ_4 denote the angle between consecutive paths in the trajectory, and d_1 through d_4 denote the trajectory path lengths at consecutive time instants.**

In order to find the relative nature of the rodent trajectories, the relative trajectory distances (refer Table 7.2) were computed. Given two rodent trajectories,

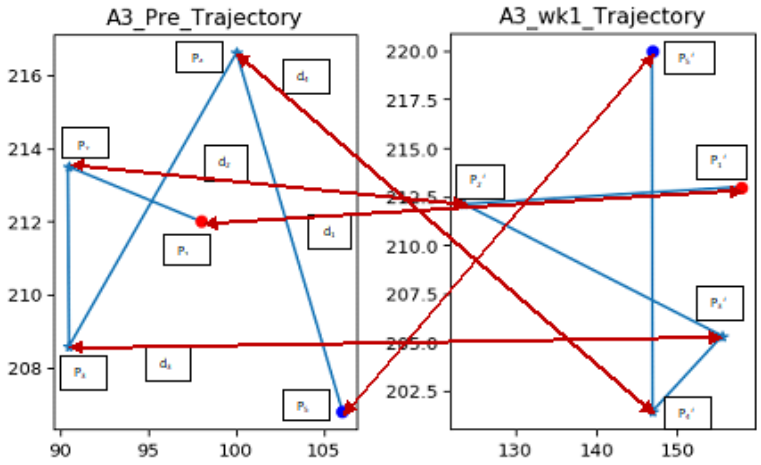

Figure 7.2: **Relative trajectory distance illustrated using first five trajectory positions of pre-TBI and post-TBI cases of rodent subject A3.** P_1 **through** P_5 **and** $P_1^{'}$ **through** $P_5^{'}$ **denote the subject positions at consecutive time instants for the pre-TBI and post-TBI cases, respectively.** d_1 **through** d_4 **denote the relative trajectory Euclidean distance at consecutive time instants.**

Table 7.1: **Classification accuracy using Relative Trajectory Distances.**

Parameter(s) Used	Accuracy		
	Linear SVM	SVM (RBF kernel)	MLP
Position	0.48	0.52	0.52
Distance	0.47	0.58	0.44
Slope	0.53	0.55	0.56
Pose	0.48	0.52	0.48
Position + *Distance*	0.47	0.52	0.45
Position + *Slope*	0.52	0.52	0.52
Position + *Pose*	0.47	0.52	0.52
Distance + *Pose*	0.44	0.52	0.44
Distance + *Slope*	0.47	0.52	0.48
Slope + *Pose*	0.31	0.34	0.56
Position + *Distance* + *Slope* + *Pose*	0.47	0.52	0.45

the relative trajectory distance is computed as the mean of the Euclidean distances between the respective trajectory points of the two trajectories. Figure 7.2 illustrates the idea of relative trajectory distance computation. The distance values shown in Table 7.2 are the respective mean values of the relative trajectory Euclidean distances between the trajectories over a fixed time duration. The mean and standard deviation of the respective distance distributions are shown in Table 7.3. Figure 7.3 illustrates these probability distributions. As can be inferred from Table 7.3 and Figure 7.3, the probability distributions of relative trajectory distances for pre-TBI *vs.* pre-TBI cases, post-TBI *vs.* post-TBI cases and pre-TBI *vs.* post-TBI cases have significant overlap, and hence a direct approach to discern the behaviors defined by these distributions is quite a difficult task.

Table 7.2: **Relative Trajectory Distances (pre-TBI *vs.* post-TBI).**

$Videos \quad Compared$	Mean Euclidean distance
$A3 : pre-TBI \& post-TBI$	86.2
$B2 : pre-TBI \& post-TBI$	97.9
$C1 : pre-TBI \& post-TBI$	85.2
$D1 : pre-TBI \& post-TBI$	83.1
$E2 : pre-TBI \& post-TBI$	94.1
$F2 : pre-TBI \& post-TBI$	86.9

Table 7.3: **Mean and Standard Deviation for the Relative Trajectory Distance values.**

Case	Mean	Standard Deviation
pre-TBI *vs.* pre-TBI	92.24	5.50
post-TBI *vs.* post-TBI	84.52	4.22
pre-TBI *vs.* post-TBI	87.19	4.42

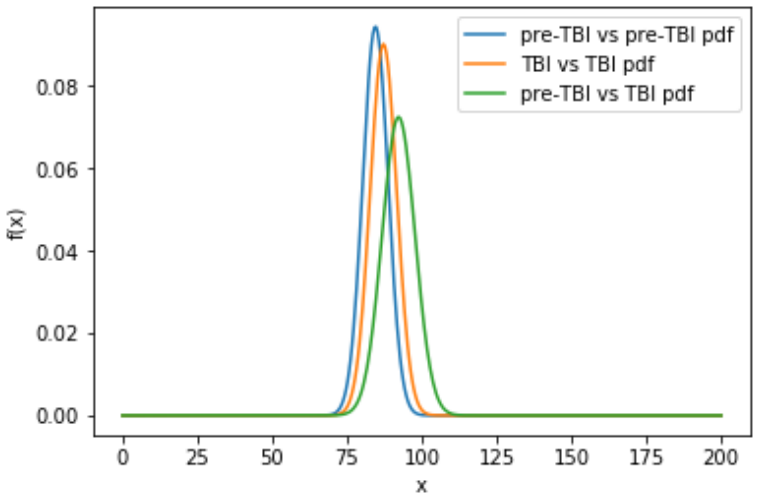

Figure 7.3: **Probability density functions (PDFs) for relative trajectory distances for pre-TBI** *vs.* **pre-TBI, post-TBI** *vs.* **post-TBI and pre-TBI** *vs.* **post-TBI cases.**

7.2 Matching trajectories via Shape Context

7.2.1 Overview of Shape Context

Shape Context [22] is a feature descriptor widely used for object recognition. Shape Context is used for describing shapes, which can be then used for tasks such as computing shape similarities and further generate point correspondences [23] [24]. In order to describe a shape, we start with n points on the contour of the shape. For each point P on the contour, we find the distance (vectors) and the angle to the remaining $n-1$ points on the contour, and compute the resulting histogram. The histogram gives the Shape Context of those respective points. The angle

measured is between the respective local tangents and the distance values used are log-distances. For each point P on the contour, the set of all such vectors is a rich description of the shape of the object localized at the given point P, but is too detailed a descriptor. The distribution of the vectors over all contour points could be considered as a robust, compact and highly discriminative descriptor. For a given point p_i on the contour, the corresponding Shape Context is the coarse histogram of relative coordinates of the remaining $n-1$ points computed as:

$$h_i(k) = \#\{q \neq p_i : (q - p_i) \in \text{bin}(k)\} \tag{7.1}$$

where, q denotes the points on the contour other than the point under consideration, p_i. The bins are considered to be uniform in log-polar space. For a given point p_i on the contour, we consider 12 angle bins and 5 log-distance bins: a total of 60 bins. Figure 7.4 illustrates the Shape Context computation as described in the original paper by Belongie et al. [22].

7.2.2 Match trajectory signatures using Shape Context

In order to check the similarity between the trajectories of the pre-TBI and post-TBI cases, we use Shape Context. We extract the trajectory of the rodent for a fixed time duration (i.e., fixed number of frames), for each pair of source and target videos. The source and target videos could belong to pre-TBI or post-TBI rodent subject. For each of the trajectories, we compute the Shape Context representation which is a histogram representation that implicitly captures the

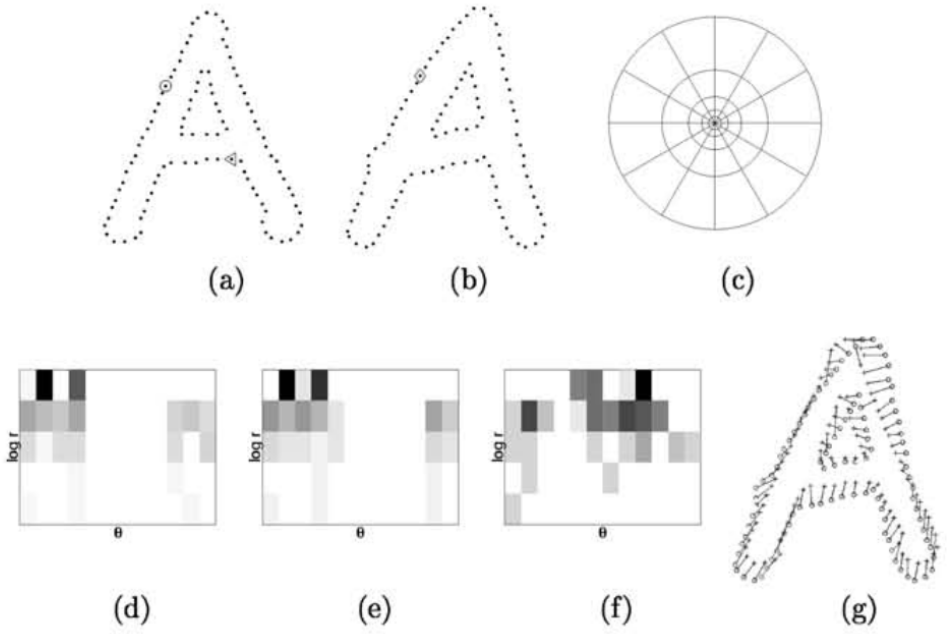

Figure 7.4: **Shape Context computation and matching. (a, b) Sampled edge points of two shapes. (c) Diagram of log-polar histogram bins used in computing the shape contexts. We use 5 bins for log r and 12 bins for θ. (d-f) Example Shape Contexts for reference samples marked by o, $\diamond$, $\triangleleft$ in (a,b). Each Shape Context is a log-polar histogram of the coordinates of the rest of the point set measured using the reference point as the origin (Dark = large value). Note the visual similarity of the Shape Contexts for o and $\diamond$, which were computed for relatively similar points on the two shapes. By contrast, the Shape Context for $\triangleleft$ is quite different. (g) Correspondences found using bipartite matching, with costs defined by the χ^2 distance between histograms. (Figure courtesy: Belongie et al. [22]).**

shape of the trajectory. Then, we compute the Euclidean distance between the respective histograms. The histogram distance gives a measure of how similar or dissimilar the trajectory shapes are. The larger the distance value the more different the relative trajectory behaviors are, and vice versa.

The Shape Context results for the pre-TBI and post-TBI cases of the rodent subject "A3" are shown in Figure 7.5. Tables 7.4 and 7.5 show the evaluation measures based on the Shape Context computation. Table 7.4 shows the evaluation of matching pre-TBI case of rodent subject "A3" with its post-TBI case, and the pre-TBI as well as post-TBI cases of the rest of the 5 rodent subjects used in our experiments. Table 7.5 shows the evaluation of matching post-TBI case of rodent subject "A3" with the pre-TBI as well as post-TBI cases of the rest of the 5 rodent subjects used in our experiments. As described in Belongie et al. [22], the matching correspondences are found using bipartite matching, with costs defined by the χ^2 distance between histograms. Given the set of costs $\{C_{ij}\}$ between all pairs of points i on the first shape and j on the second shape, we want to minimize the total cost of matching, subject to the constraint that the matching be one-to-one. This is an instance of the square assignment (or weighted bipartite matching) problem, which can be solved in $O(N^3)$ time using the Hungarian algorithm [37]. In Belongie et al. [22], the authors use the more efficient algorithm proposed by Jonker et al. [38]. The total cost of matching consists of an affine cost and tangent angle dissimilarity cost. The affine cost is the computational cost involved in computing the affine transformation model during the shape matching process. The tangent angle dissimilarity cost is the measure of dissimilarity of local tangent

angles. The algorithm also computes Shape Context cost and error values. The Shape Context cost is the computational cost incurred in computing the Shape Context representation for the shapes which are matched, and the error values measure the Euclidean distance between the shapes being matched.

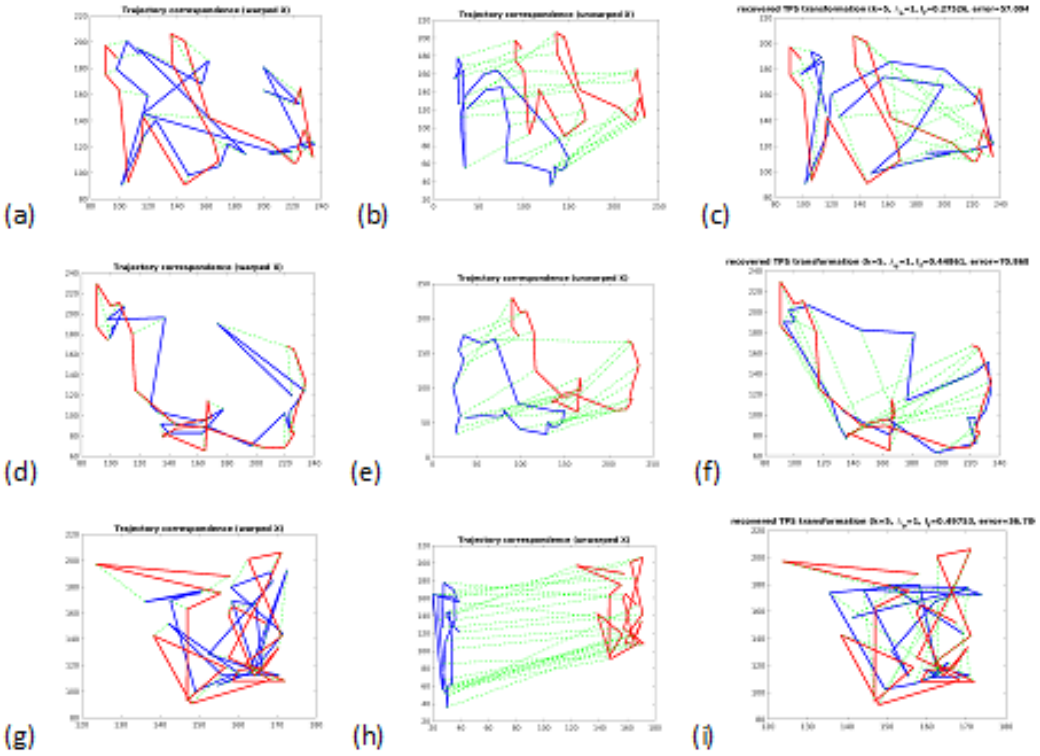

Figure 7.5: **Shape Context Sample Results:** $A3(pre-TBI)$ *vs.* **rest (a-f), and** $A3(post-TBI)$ *vs.* **rest (g-i).**

Table 7.4: **Affine Cost, Shape Context Cost and Error Values for the Shape Context computation for the rodent subject** $A3(pre - TBI)$ *vs.* **rest case.**

Video Reference	Affine Cost	Shape Context Cost	Error
$A3(post - TBI)$	2.9505	0.3077	39.5892
$B2(pre - TBI)$	0.5018	0.3942	76.7967
$B2(post - TBI)$	0.2501	0.3415	70.8682
$C1(pre - TBI)$	4.6099	0.3172	57.0706
$C1(post - TBI)$	0.5374	0.2526	79.9706
$D1(pre - TBI)$	3.9108	0.4160	37.5964
$D1(post - TBI)$	0.3521	0.3563	42.6753
$E2(pre - TBI)$	1.3637	0.2543	80.2924
$E2(post - TBI)$	0.6120	0.2787	65.4966
$F2(pre - TBI)$	0.9153	0.3063	61.8775
$F2(post - TBI)$	1.2227	0.2807	57.0941

Table 7.5: **Affine Cost, Shape Context Cost and Error Values for the Shape Context computation for the rodent subject** $A3(post - TBI)$ *vs.* **rest case.**

Video Reference	Affine Cost	Shape Context Cost	Error
$B2(pre - TBI)$	2.2970	0.4170	69.7543
$B2(post - TBI)$	2.7877	0.2738	56.2740
$C1(pre - TBI)$	2.4631	0.4189	22.7920
$C1(post - TBI)$	1.1328	0.3092	44.7031
$D1(pre - TBI)$	1.5061	0.2982	36.4963
$D1(post - TBI)$	0.6451	0.5376	23.8164
$E2(pre - TBI)$	1.1572	0.2933	62.5499
$E2(post - TBI)$	1.6746	0.2360	44.7109
$F2(pre - TBI)$	2.0638	0.2152	71.6708
$F2(post - TBI)$	2.8664	0.3174	36.7060

7.3 Bag-of-Features

7.3.1 Overview of Bag-of-Features

The Bag-of-Features (BoF) model is a simplified feature representation commonly used in natural language processing (NLP) and information retrieval (IR). In the BoF model for NLP, textual data such a sentence or a document is represented as the bag (or multi-set) of its words while ignoring the grammar or the order of word occurrences, and retaining only the multiplicity of the words. BoF is a widely used descriptor in computer vision [25]. The earliest reference to BoF can be found in the work by Harris [26]. BoF is widely used in IR for document classification tasks where the aim is to find the frequency of occurrences of words in documents, and consequently use it to train a classifier.

Consider an example of two documents: (1) "John likes pancakes. Michelle likes pancakes too.", and (2) "John also likes puddings." Based on the these two text documents, a list is constructed for each document as follows:

$[`John', `likes', `pancakes', `Michelle', `too', `also', `puddings']$

The corresponding bag-of-words representations are as follows:

$BoW_1 = \{`John' : 1, `likes' : 2, `pancakes' : 2, `Michelle' : 1, `too' : 1\}$

$BoW_2 = \{`John' : 1, `also' : 1, `likes' : 1, `puddings' : 1\}$

As the order of the elements is not relevant, $\{`too' : 1, `Michelle' : 1, `pancakes' : 2, `likes' : 2, `John' : 1\}$ is also BoW_1.

BoF model is mainly used as a tool for feature generation. In NLP, BoF model for text is used for computing factors which characterize the text such as term

frequency: the number of times a particular term appears in the document. The lists of term frequencies, however, do not preserve the order in which the words may appear in the documents. This type of feature representation is found to be useful in applications such as email filtering [25]. The idea is to utilize the "term frequency" of words likely to appear in spam emails and use them as a criteria for classifying the given email to be spam or not. Here, two bags of features are used for training the classifier: one containing words which appear mostly in legitimate emails and the other containing words likely to come across only in spam mails. Bayesian spam filtering [27] is a widely used technique which models the email to be comprised for a set of unordered set of words randomly picked from two probability distributions: one representing spam email and the other representing legitimate email. Bayesian spam filter assumes that the email is formed by a pile of words randomly picked from one of the two feature bags, and uses Bayesian probability to predict to which feature bag the given email is likely to belong to.

BoF for images involves 3 main steps: (1) feature detection and representation, (2) image representation, and (3) category classification. Feature detection can be performed using a regular image grid approach [29], an interest point detector [29, 30, 31], random sampling [32] or segmentation-based patches [33]. Feature representation involves detecting the image patches corresponding to the interest points [34, 35], normalizing the patches and computing a feature descriptor [10]. Once we have the feature descriptor, the next stage is to generate the codeword dictionary [31]. The codeword dictionary contains the image patches or the image feature representation (termed as codewords) which can be used as a basis for

generic representation of the images. The next step is the computation of the bag-of-features representation. For this purpose, we find the codewords' frequency corresponding to each image in the training set, and represent each image in the training set as a 'bag' of codeword frequencies. This image representation can be used for training a category classifier. The entire set up can be collectively termed as a bag-of-features classifier. In order for the classifier to be efficient, the features within the bag-of-features descriptor should be generalizable beyond the training set data. For instance, in our experiments, we use the events associated with the rodent movement such as distance, pose, rate of change in distance and pose gradient as the codewords.

7.3.2 Modeling behavior by identifying events

Any behavior episode which is useful for analyzing the effect of TBI on the subjects constitutes an event. In our analysis, we use episodes based on trajectory distance, rate of change in trajectory distance, object pose, and rate of change in the object pose as events, and we try to model the behavior of the subject using a "bag-of-event frequencies" descriptor. The reason we chose these episodes as events, is because these are salient actions which convey information regarding the behavior of the subjects along the trajectory, which can possibly shed light on the extent of the impact TBI had on them.

In our work, we use the "N-grams model" [28] for feature representation. This is because the BoF model is an orderless feature representation. For instance, in the case of the example above, the Bag-of-Words representation of "John likes

pancakes. Michelle likes pancakes too." will not take into consideration the fact that the verb "likes" always follows a noun such as "John". Using a "N-grams model", we can have the feature representation for the first document in the example as ['*John likes*', '*likes pancakes*', '*Michelle likes*', '*pancakes too*']. Thus we can see that the "N-grams model" helps to preserve the spatial relation of the words rather than just binning them. Using the mathematical prefix notation, if N=1 we call the model an uni-gram model, if N=2 we call the model a bi-gram model and so on. We use a bi-gram model for feature representation of events. In our experiments, for instance, the distance traversed along trajectory in unit time could constitute an event for a uni-gram model whereas two consecutive angular distances taken by the rodent subject could constitute an event for a bi-gram model.

Through our experiments, we observed that directly matching the behavior signatures is not helpful since the movement of the rodent is not restricted to any fixed trajectory, and it is free to move as it wishes. Therefore, there is no point in directly matching the behavior signatures as it conveys little reliable information.

The direct matching of trajectories was attempted using Shape Context as shown in Section 7.2.2. As we can see from Figure 5.2 and Figure 5.3, and the results shown in Figure 7.5, the nature of the trajectory varies widely across the videos and hence a direct matching of the trajectory makes little sense.

Figure 7.6 shows the trajectory pairs of pre-TBI and post-TBI cases of 3 rodent subjects for the first 50 frames ($\sim$ 50 seconds). As we can observe from the pairs in the figure, the trajectory for the subject does not follow any reliable, interpretable

path. For the pre-TBI and post-TBI cases of the same subject, the nature of the trajectory is very diverse and there are no available benchmark tracking cues for comparison. In order for a reliable direct comparison of the trajectories traversed by the subjects to make sense, the trajectory should be more sophisticated and should be the same for all, so that we have a benchmark trajectory for comparison with the trajectory traced by the subject. In such a case, given a fixed trajectory, one would be able to analyze how the subjects, prior to TBI and after, would behave along the fixed trajectory, and thus, one could reliably draw a comparison of their respective behaviors.

In our proposed approach, instead of comparing the trajectories which are unconstrained and arbitrary, we identify a set of repeating, uniquely identifiable and salient motion cues which we term as 'events'. Our approach models the occurrence and variation of these events over time. Figure 7.7 shows how we define the events; we use the rodent's pose as the "event" here. We consider the distance traveled by the subject across time, the subject's pose across time, the rate of change in distance traveled across time (velocity gradient or acceleration) and the rate of change in pose across time (pose gradient) as salient motion cues for modeling the behavior of the subject over time. The distance traveled by the subject per unit time is a indicative measure of the alacrity of the subject, pre-TBI and post-TBI. The pose of the subject indicates a measure of disorientation in the subject's behavior; the greater the number of poses a subject assumes in a fixed time duration the more disoriented its behavior is and vice versa. The rate of change in distance traversed over fixed time periods is a measure of mental

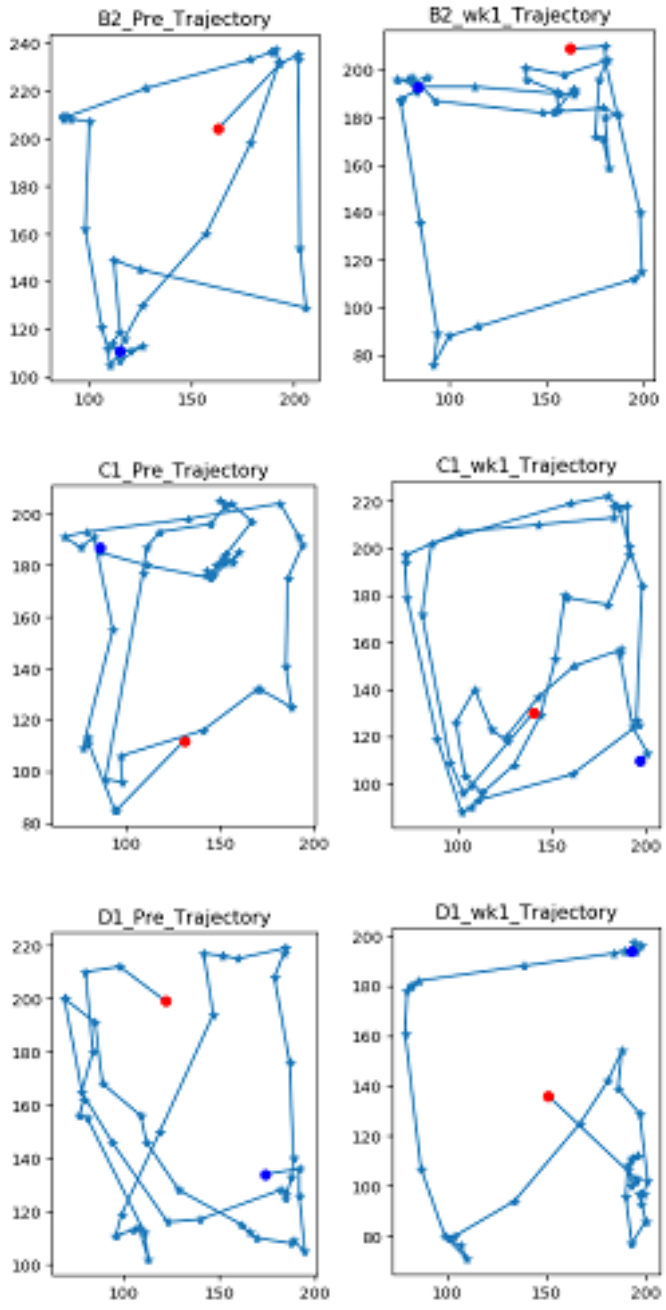

Figure 7.6: **Trajectory Pairs (Left: pre-TBI, Right: post-TBI) of 3 subjects.**

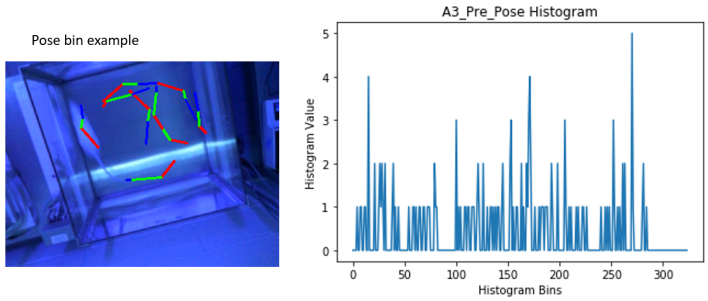

Figure 7.7: **Left: A Bag-of-Feature example using rodent pose, Right: The Bag-of-Feature histogram for the subject, using pose as the event for feature representation.**

vagueness; the subject is not certain about how fast to travel along the direction of its motion. The rate of change of pose is a clear indication of the subject's mental confusion regarding which direction to look towards or follow. However, the process of directly binning the occurrence of events makes us lose the temporal aspect of occurrence of the events. Since behavioral analysis is a time-dependent process, we need to model the feature representation as a function of time. This is achieved by our proposed feature representation using Temporal BoF (T-BoF).

7.3.3 Temporal Bag-of-Features

Temporal Bag-of-Features (T-BoF) models the Bag-of-Features (BoF) representation as a function of time. Figure 7.8 shows the Temporal Bag-of-Features (T-BoF) histograms for the pre-TBI and post-TBI cases, using tracking results and ground truth data, and Tables 7.6 through 7.8 show the corresponding histogram

(Euclidean) distances between the Temporal Bag-of-Features (T-BoF) representations.

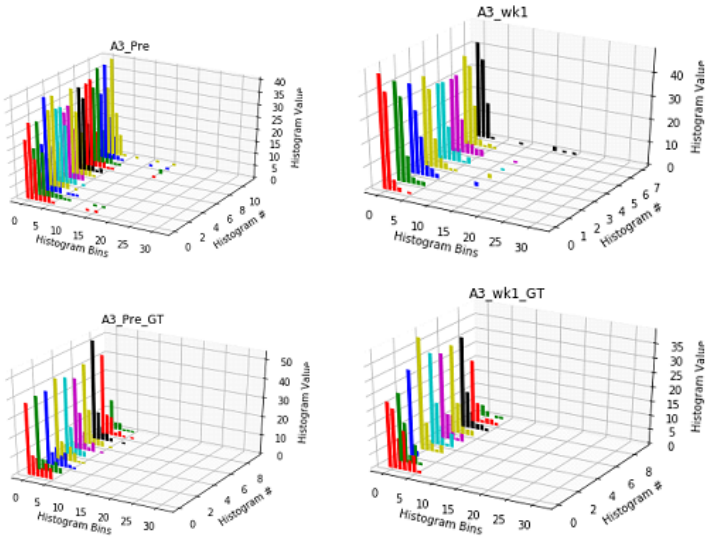

Figure 7.8: **Temporal BoF histograms (based on episode distance) for the same rodent subject A3 (First row: pre-TBI and post-TBI (Tracking results), Second row: pre-TBI and post-TBI (Ground Truth)).**

In the earlier representations such as the relative trajectory distance measures and Shape Context representation, we cannot employ any machine learning models for successfully classifying the obtained feature representation as belonging to either pre-TBI or post-TBI case, owing to three obvious reasons: (1) the feature representation is not an ubiquitous one, (2) there is not enough training data, and (3) there are no reliable cues to discern the nature of one set of data from the other. As we can see from the results obtained in Table 7.1, training any machine learning algorithm requires more data with reliable cues. In Table 7.1, Support Vector Machines (SVM) and Multilayer Perceptron (MLP) were used as initial

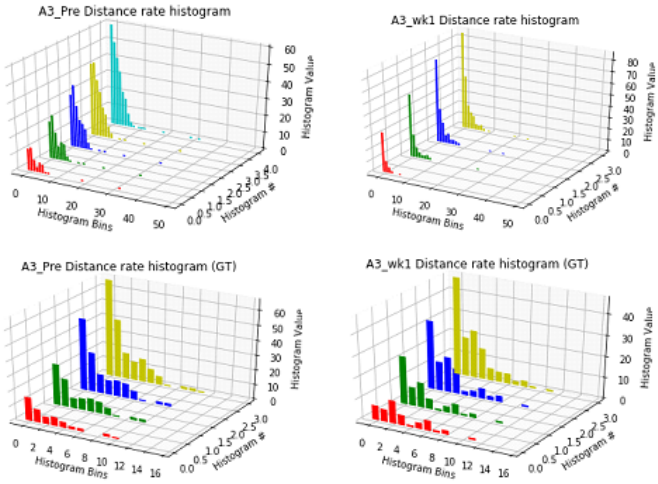

Figure 7.9: **Temporal BoF histograms (based on rate of change in episode distance) for the same rodent subject A3 (First row: pre-TBI and post-TBI (Tracking results), Second row: pre-TBI and post-TBI (Ground Truth)).**

prototypes for testing the efficacy of machine learning algorithms in learning the feature representations for the data. As we can see from the results shown in Table 7.1, the machine learning algorithms fail to identify the features as belonging to either pre-TBI or post-TBI cases.

The use of BoF approach for obtaining the feature representation can provide us with more reliable cues. In case of pose binning, we use a bi-gram model for binning the events. We also impose a criterion that the consecutive pose angles should add up to an angle threshold and the distance traveled along the arcs forming the pose angles are individually above a distance threshold. In case an

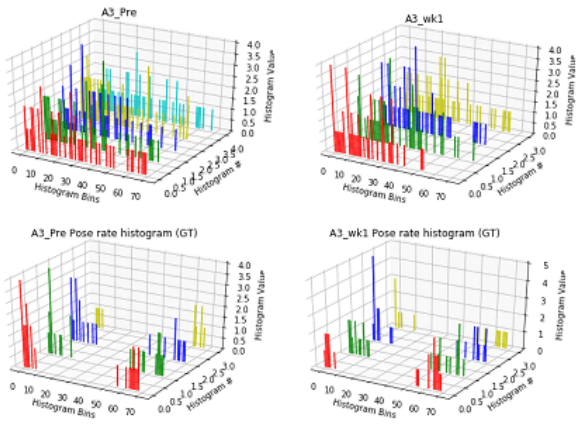

Figure 7.10: **Temporal BoF histograms (based on rate of change in pose) for the same rodent subject A3 (First row: pre-TBI and post-TBI (Tracking results), Second row: pre-TBI and post-TBI (Ground Truth)).**

event satisfies the above threshold conditions, we assign it to the corresponding bin. In our experiments, we used an angle threshold of 50° and a distance threshold of 10 pixels. We chose the bin resolution to be 20° and thus have 18 bins (for an angular coverage of 360°). Since we have a pair of poses constituting an event, it leaves us with two set of 18 bins each and a total of 324 bin combinations and hence 324 histogram bins. In the case of rate of change in distance, we use a bin resolution of 5 pixels and in the case of rate of change in pose, we use a bin resolution of 5°.

Temporal histogram is the modeling of the histograms as a function of time. The histograms are computed over fixed time duration of the given videos. Here, we compute histogram corresponding to video chunks having 100 frames ($\sim$ 100 seconds) each. Since previous research has shown that the activity of the pre-TBI and post-TBI subject can be efficiently studied only for the first 10 to 12 minutes, we use only the activity within initial 12 minutes for the temporal histogram generation.

Our approach tries to reason temporally by maintaining a record of the occurrence and variation of the salient motion cues which constitute the events being monitored over time. These salient motion cues model the behavior of the subject over time.

For analysis, we compared each of the histograms of a particular video with the rest of its histograms. We also compared each histogram of the pre-TBI case of the subject with the histograms corresponding to the same subject's post-TBI case.

Table 7.6: **Euclidean distances between Temporal BoF descriptors (Trajectory distance episodes) for the rodent subject A3 (pre-TBI** *vs.* **pre-TBI). Both rows and columns correspond to pre-TBI episodes.**

0.000	7.483	10.392	12.247	11.489	7.616	13.565	12.570	13.342	13.191	11.045	13.115
7.483	0.000	9.055	17.321	15.492	9.487	17.263	17.378	19.079	18.547	15.033	17.029
10.392	9.055	0.000	13.856	15.033	11.662	20.445	18.276	17.029	14.900	10.296	11.916
12.247	17.321	13.856	0.000	8.000	11.314	14.283	9.899	5.477	5.831	6.164	6.000
11.489	15.492	15.033	8.000	0.000	7.746	11.747	8.367	11.314	13.115	11.225	11.916
7.616	9.487	11.662	11.314	7.746	0.000	10.000	9.695	13.266	14.491	11.576	13.191
13.565	17.263	20.445	14.283	11.747	10.000	0.000	6.325	12.728	16.432	16.248	17.146
12.570	17.378	18.276	9.899	8.367	9.695	6.325	0.000	9.165	12.884	11.916	12.490
13.342	19.079	17.029	5.477	11.314	13.266	12.728	9.165	0.000	4.690	8.602	8.367
13.191	18.547	14.900	5.831	13.115	14.491	16.432	12.884	4.690	0.000	6.928	6.633
11.045	15.033	10.296	6.164	11.225	11.576	16.248	11.916	8.602	6.928	0.000	2.449
13.115	17.029	11.916	6.000	11.916	13.191	17.146	12.490	8.367	6.633	2.449	0.000

Table 7.7: **Euclidean distances between Temporal BoF descriptors (Trajectory distance episodes) for the rodent subject A3 (post-TBI *vs.* post-TBI). Both rows and columns correspond to post-TBI episodes.**

0.000	10.770	22.045	16.062	21.954	22.539	17.944	15.100
10.770	0.000	12.083	6.481	12.410	12.961	7.616	6.325
22.045	12.083	0.000	7.483	9.592	10.100	8.944	11.225
16.062	6.481	7.483	0.000	7.348	8.124	6.633	8.124
21.954	12.410	9.592	7.348	0.000	2.449	9.274	12.728
22.539	12.961	10.100	8.124	2.449	0.000	9.487	13.342
17.944	7.616	8.944	6.633	9.274	9.487	0.000	5.477
15.100	6.325	11.225	8.124	12.728	13.342	5.477	0.000

Table 7.8: **Euclidean distances between Temporal BoF descriptors (Trajectory distance episodes) for the rodent subject A3 (pre-TBI *vs.* post-TBI). Rows correspond to pre-TBI episodes and columns correspond to post-TBI episodes.**

32.527	23.238	16.852	18.000	12.410	11.314	19.235	23.281
38.497	29.563	22.672	24.000	18.221	17.607	25.456	29.360
35.861	27.532	23.664	22.891	16.912	16.432	23.495	27.386
24.249	14.765	12.570	10.770	6.325	6.481	9.899	13.928
30.100	19.698	12.247	14.967	10.488	10.583	13.342	17.493
32.833	23.152	15.166	17.607	12.806	12.806	18.439	22.361
28.390	19.235	9.487	13.856	13.115	13.342	15.875	18.815
26.382	16.613	8.718	11.662	10.677	10.392	11.662	15.427
20.199	10.954	10.100	6.481	4.899	5.099	8.124	11.402
21.448	13.342	14.283	10.000	6.633	6.481	11.225	14.071
27.055	18.655	16.912	14.765	10.677	10.000	14.629	18.330
25.846	17.720	17.029	14.283	10.863	10.392	13.784	17.205

Chapter 8

Results and Evaluation

A combination of the LDOF algorithm and Kalman filter was employed for tracking the movement behavior of different rodent subjects in pre-TBI as well as post-TBI conditions across the given videos. In order to find a criterion that is useful for classification of a given subject as belonging to pre-TBI or post-TBI condition, we considered direct matching of the trajectories of the subjects. Figure 8.1 shows the trajectory pairs (pre-TBI and post-TBI conditions) of 3 subjects. As can be seen from the figure, the subjects were not given any fixed trajectory to follow. We employed distance-based methods such as using relative trajectory (Euclidean) distance to see if it can provide any cues to discern a pre-TBI rodent's behavior from its own behavior post-TBI (refer Table 8.1). As can be inferred from the table, there is a small relative distance between the trajectories taken by the same subject(s) pre-TBI and post-TBI. Therefore, we tried to employ the traditional machine learning algorithms in order to learn and classify a given trajectory behavior

as belonging to pre-TBI or post-TBI, but found that even state-of-the-art machine learning algorithms for classification such as Support Vector Machines (SVM) and Multilayer Perceptron (MLP) fail to learn anything significant from the data accumulated by the trajectory, such as the trajectory distance traveled over fixed time durations, trajectory velocity and acceleration over fixed time durations, rodent's poses over fixed time durations and trajectory slopes over fixed time durations. Upon examining the distribution of trajectories closely, we found that the distribution of the trajectories for pre-TBI and post-TBI cases were severely overlapped, thereby making it hard for any machine learning algorithm to interpret. Figure 8.2 shows the mean, variance and standard deviation of the distributions of the pre-TBI and post-TBI trajectories. We can clearly see that the distributions overlap significantly. This fact is backed by the t-SNE distribution of the pre-TBI and post-TBI trajectories as shown in Figure 8.2.

Table 8.1: **Relative Trajectory Distances (pre-TBI** *vs.* **post-TBI) for 6 rodent subjects.**

$Videos \quad Compared$	Mean Euclidean distance
$A3(pre-TBI)\&A3(post-TBI)$	86.2
$B2(pre-TBI)\&B2(post-TBI)$	97.9
$C1(pre-TBI)\&C1(post-TBI)$	85.2
$D1(pre-TBI)\&D1(post-TBI)$	83.1
$E2(pre-TBI)\&E2(post-TBI)$	94.1
$F2(pre-TBI)\&F2(post-TBI)$	86.9

Since the machine learning methods failed to find any classification criteria owing to the relatively random nature of the trajectories taken, we employed the widely popular and robust shape matching algorithm known as Shape Context

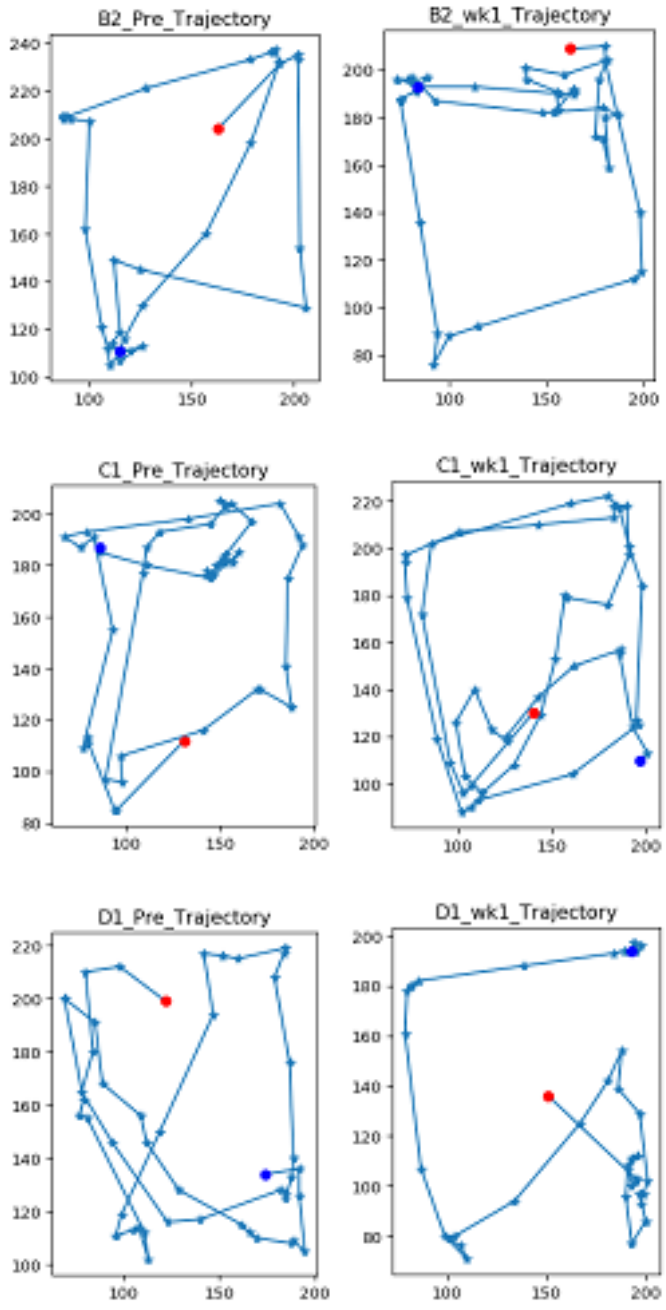

Figure 8.1: **Trajectory Pairs (Left: pre-TBI, Right: post-TBI) of 3 subjects.**

Table 8.2: **Mean and Standard Deviation for the Relative Trajectory Distance cases.**

Case	Mean	Standard Deviation
pre-TBI *vs.* pre-TBI	92.24	5.50
post-TBI *vs.* post-TBI	84.52	4.22
pre-TBI *vs.* post-TBI	87.19	4.42

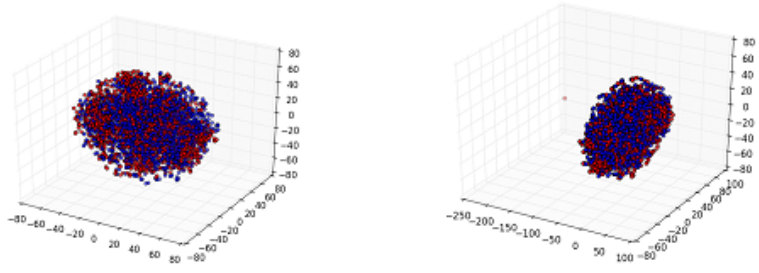

Figure 8.2: **tSNE distributions of (i) position coordinates, distance vectors and slope vectors of estimated trajectory (pre-TBI and post-TBI) and (ii) with pose included.**

which tries to find a relative histogram representation (known as Shape Context representation) of the trajectories and tries to find a possible match between them by translating, rotating, scaling and wrapping trajectories while trying to find a Thin Plate Spline (TPS) transformation [39] which can represent a correspondence between the trajectories. However, from the results of Shape Context employed to match the trajectories for the initial 20 seconds of the video (we used a smaller number of points to check the efficiency of the algorithm) as shown in Figure 8.3, we can see that the error in finding a correspondence mapping between the trajectories is very high. This clearly proves that the direct matching of the trajectories is not useful for our task.

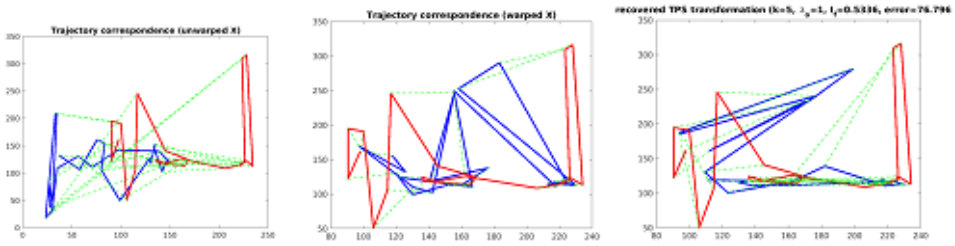

Figure 8.3: **Shape Context matching for subject A3 (pre-TBI and post-TBI).**

Since the direct matching of the pre-TBI and post-TBI trajectories made little sense, we moved on to the idea of employing a feature representation for uniquely discerning a pre-TBI subject from a post-TBI subject. For this, we made use of a widely used feature representation technique called "Bag-of-Features" (BoF). In order to generate the BoF representation, we used the most relevant factors that could possibly shed light on the behavior of the trajectory irrespective of the tra-

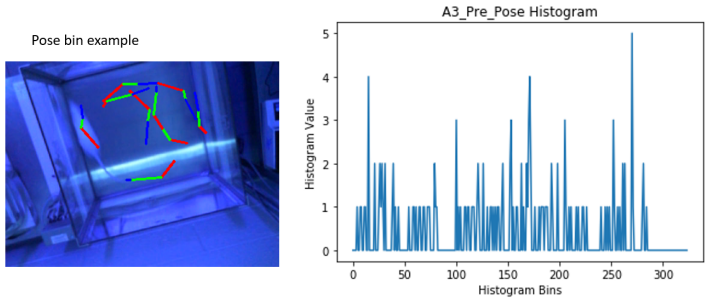

Figure 8.4: **Left: A Bag-of-Feature example using rodent pose, Right: The Bag-of-Feature histogram for the subject, using pose as the event for feature representation.**

jectory the rodent subject follows, and treated them as the "events" constituting the feature representation. We regarded the trajectory distance traveled and the rodent's pose and the rate of change of both as the "events" for generating the BoF representation. The idea is, given a trajectory behavior, we model the given trajectory as a combination of the feature "bags" of these "events". The left image of Figure 8.4 shows a Bag-of-Feature bin for a particular pose of the rodent. The collection of poses indicates the number of occurrences of the encountered pose "event" across the whole trajectory, and hence this forms the "feature bag" for the given pose.

However, we could still observe that a straight-forward BoF is not enough to capture the subtleties involved in the rodent's behavior over the entire trajectory, which might be relevant for identifying the subject to be a pre-TBI or post-TBI subject, since directly binning the events into the bag-of-feature "bags" would

make us lose the temporal aspects of the trajectory behavior. Hence, we tried to model the BoF as a function of time. In Figure 8.5, we use trajectory distance across time as the "event" with a bin resolution of 10 pixels. In Figure 8.6, we use distance rate (or acceleration) across time as the "event" with a bin resolution of 5 pixels. Figure 8.7 shows the BoF histograms as a function of time, using change of pose across time as the "event", and with a bin resolution of 20°. Figure 8.8 shows the BoF histograms as a function of time, using rate of change of pose across time as the "event", and with a bin resolution of 5°. In Figure 8.6, we use distance rate (or acceleration) across time as the "event" with a bin resolution of 5 pixels. As seen from the figures, we were able to significantly model the behavior of the subject in terms of the temporal occurrences of the "events" under consideration. It can be observed that there is some noticeable degree of discernibility between the BoF representation of a subject in its pre-TBI and post-TBI conditions. The results shown in Table 8.3 shows that the presence of 'Temporal Bag-of-Features histogram' of trajectory distance in the feature representation increases the performance of the machine learning algorithm. The combination of 'Temporal Bag-of-Features histograms' representation of trajectory distance and 'Temporal Bag-of-Features histograms' representation of pose, and the combination of 'Temporal Bag-of-Features histograms' of trajectory distance and 'Temporal Bag-of-Features histograms' of pose rate fetch an accuracy of about 0.8 . Thus, we can conclude that, with the help of more reliable data, we can generate much more robust feature representations which can increase the performance even higher.

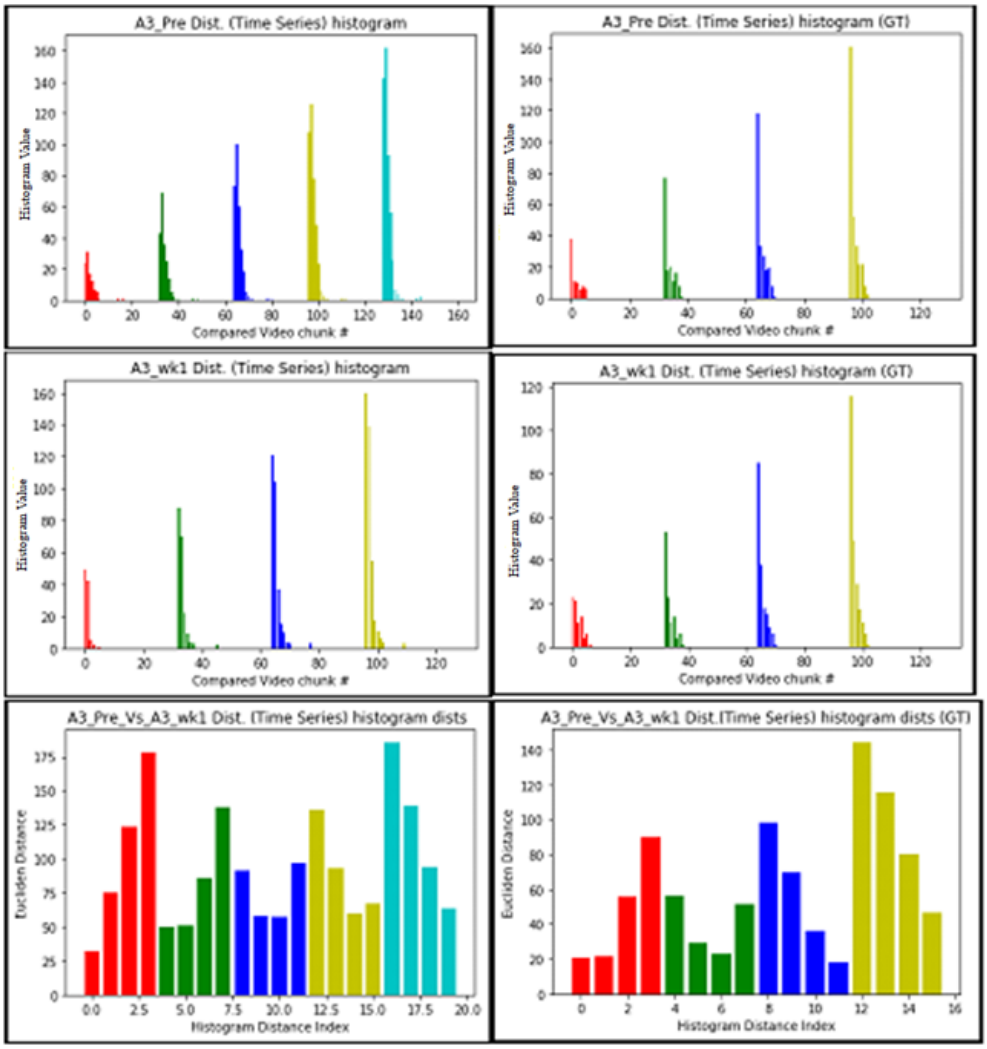

Figure 8.5: **BoF Trajectory Distance histograms for subject A3 (Time Series) (a) pre-TBI (Tracking), (b) pre-TBI (GT), (c) post-TBI (Tracking), (d) post-TBI (GT), (e) pre-TBI** *vs.* **post-TBI (Tracking) and (f) pre-TBI** *vs.* **post-TBI (GT).**

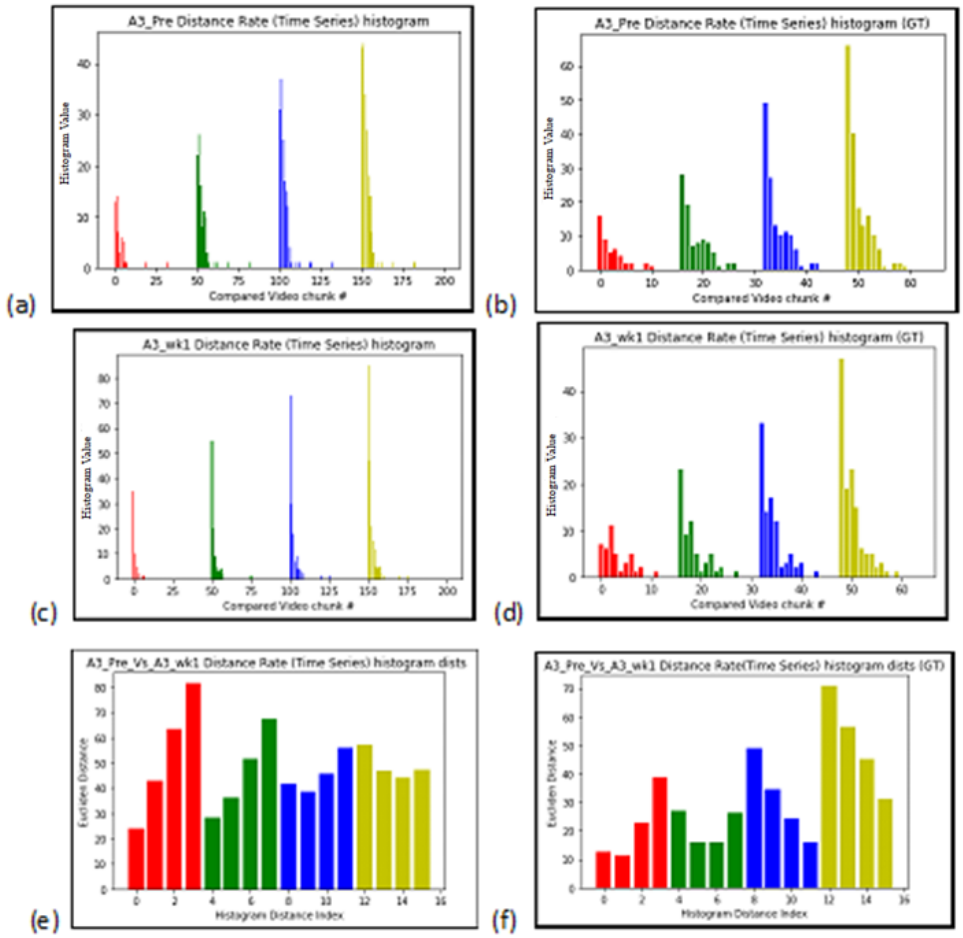

Figure 8.6: **BoF Distance rate (acceleration) histograms for subject A3 (Time Series) (a) pre-TBI (Tracking) (b) pre-TBI (GT), (c) post-TBI (Tracking) (d) post-TBI (GT),(e) pre-TBI *vs.* post-TBI (Tracking) and (f) pre-TBI *vs.* post-TBI (GT).**

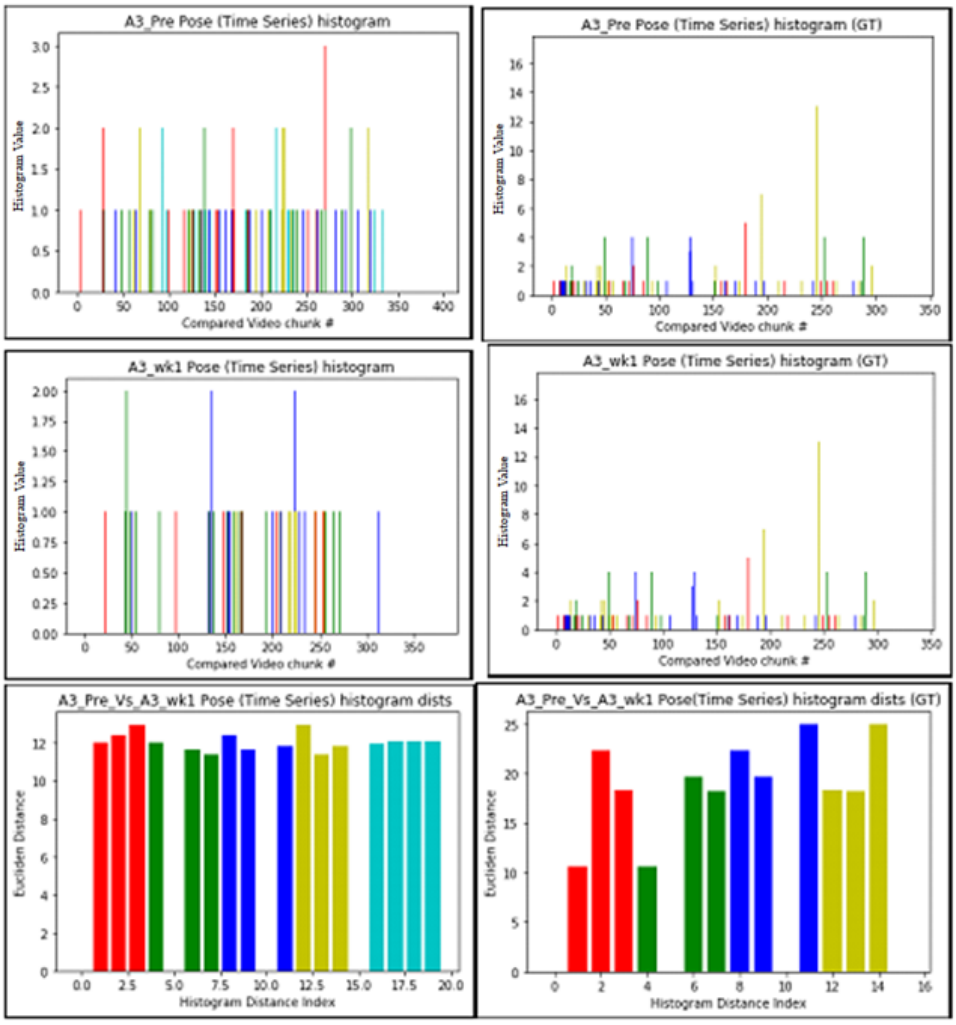

Figure 8.7: **BoF Pose histograms for subject A3 (Time Series) (a) pre-TBI (Tracking), (b) pre-TBI (GT), (c) post-TBI (Tracking), (d) post-TBI (GT), (e) pre-TBI *vs.* post-TBI (Tracking) and (f) pre-TBI *vs.* post-TBI (GT).**

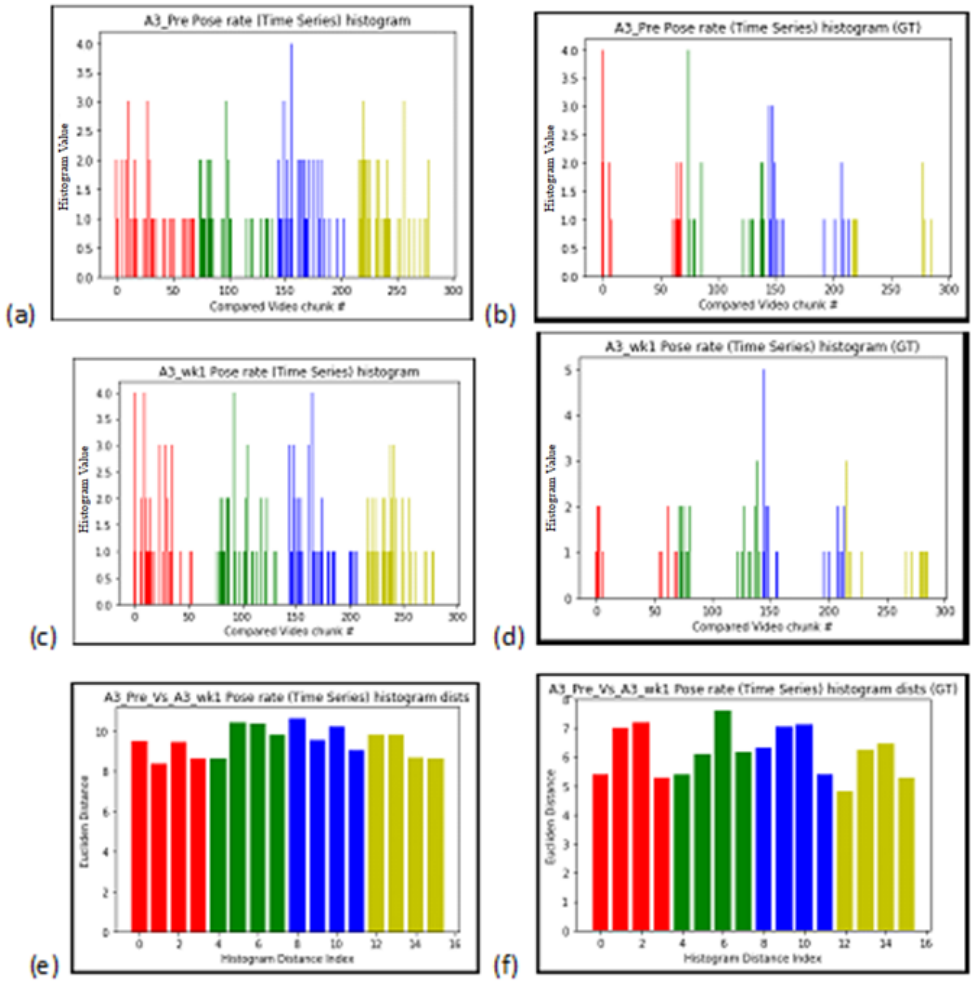

Figure 8.8: **BoF Pose rate histograms for subject A3 (Time Series) (a) pre-TBI (Tracking), (b) pre-TBI (GT), (c) post-TBI (Tracking), (d) post-TBI (GT), (e) pre-TBI** *vs.* **post-TBI (Tracking) and (f) pre-TBI** *vs.* **post-TBI (GT).**

Table 8.3: **Classification accuracy using Relative Trajectory Distance (D: Trajectory distance, DR: Distance rate, P: Pose distance, PR: Pose rate.)**

Parameter(s) Used	Accuracy			
	SVM(Linear)	SVM(RBF kernel)	MLP	OneVsRest
D	0.5	0.5	**0.75**	0.5
DR	0.5	0.5	0.25	0.25
P	0.5	0.5	0.25	0.25
PR	0.25	0.25	0.5	0.25
$D + DR$	0.5	**0.75**	0.25	0.5
$P + PR$	0.25	0.5	0.5	0.25
$D + P$	0.5	**0.75**	**0.75**	0.25
$D + PR$	0.5	**0.85**	**0.75**	0.25
$DR + P$	0.25	0.25	0.5	0.25
$DR + PR$	0.25	0.25	0.5	0.25
$D + DR + P$	0.5	0.5	0.5	0.5
$D + DR + PR$	0.5	0.5	0.25	0.5
$D + P + PR$	0.5	0.5	0.5	0.25
$DR + P + PR$	0.25	0.5	0.25	0.25
$D + DR + P + PR$	0.5	0.5	0.5	0.5

Chapter 9

Conclusion

In this thesis, we have proposed a framework that classifies the behavior of a regular rodent *vs.* rodent with induced traumatic brain injury, in a maze (glass box). The system combines traditional computer vision techniques, that computes optical flow that estimates the motion of pixels between adjacent frames of a video, followed by the use of Kalman filter based filtering to smooth the noisy predictions of the optical flow method, alongside modeling the state (or position of the rodent) at each time step across multiple video frames, enabling the system to predict the current and future positions of the rodent at each frame.

We use a traditional Kalman filter based tracking system to generate trajectories capturing the motion of the rodent at different time steps. The trajectories are then used for modeling the behavior of the rodent before and after TBI. We use machine learning classifiers to learn and classify the videos as belonging to one of the two categories - pre-TBI or post-TBI. The experimental results for

the proposed method clearly demonstrates the ability of the method to effectively model the behavior. The Bag-of-Features representation of trajectory distance, the Bag-of-Features representation of rodent subject's pose, and the Bag-of-Features representation of the rate of change of rodent subject's pose, prove to be reliable cues for representing the rodent behavior. Further increase in reliable data for extending the feature representation is bound to increase the robustness of the proposed approach.

9.1 Future Work

We intend to extend the proposed method to use more effective deep learning based techniques that have been recently shown to yield state-of-the-art performances across many image processing and computer vision problems, to learn a better feature representation and build a better tracking method. We also want to explore the appearance features to design more sophisticated loss functions that model non-linear deformations of the rodent's body.

In addition, most ambiguities that arise in the trajectory can be attributed to the simplistic nature of the maze used. More complex mazes that guide the rodents to follow a specific action can tell more about the subtle differences in the motor and sensory functions of the rodent.

Furthermore, more comprehensive data like the one described above combined with increase in size of dataset can provide us more informative cues to explore learning based methods.

Bibliography

[1] K. Fukunaga ,and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition ", *Journal in IEEE Transactions on Information Theory*,Volume 21 Issue 1, January 1975,pp. 32-40

[2] C. A. Taylor, J. M. Bell, and M. J. Breiding, "Traumatic Brain Injury-Related Emergency Department Visits, Hospitalizations, and Deaths-United States, 2007 and 2013", *Surveillance Summaries*/ March 17, 2017 / 66(9);116

[3] G. R. Bradski, "Computer vision face tracking for use in a perceptual user interface.", *Intel Technology Journal*, 2nd Quarter, 1998

[4] S. Birchfield, "Elliptical head tracking using intensity gradients and color histograms.", *In Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on*, pp. 232-237. IEEE, 1998.

[5] H. Schweitzer, J. W. Bell, and F. Wu, "Very fast template matching.", *In European Conference on Computer Vision*, pp. 358-372. Springer, Berlin, Heidelberg, 2002.

[6] P. Fieguth, and D. Terzopoulos, "Color-based tracking of heads and other mobile objects at video frame rates.", *In Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pp. 21-27. IEEE, 1997.

[7] H. P. Moravec, "Visual mapping by a robot rover.", *In Proceedings of the 6th international joint conference on Artificial intelligence*, Volume 1, pp. 598-600. Morgan Kaufmann Publishers Inc., 1979.

[8] C. Harris, and M. Stephens, "A combined corner and edge detector.", *In Alvey vision conference*, vol. 15, no. 50, pp. 10-5244. 1988.

[9] J. Shi and C. Tomasi, Good Features to Track, *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, 21-23 June 1994, pp. 593-600.

[10] D. G. Lowe, "Distinctive image features from scale-invariant keypoints.", *International journal of computer vision* 60, no. 2 (2004): 91-110.

[11] B.K.P. Horn, and B. G. Schunck, "Determining optical flow.", *Artificial intelligence* 17, no. 1-3 (1981):pp. 185-203.

[12] B. D.Lucas, and T. Kanade, "An iterative image registration technique with an application to stereo vision.", *In International Joint Conference on Artificial Intelligence* (1981): pp. 674-679.

[13] M. J. Black, and P. Anandan, "The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields.", *Computer vision and image understanding* 63, no. 1 (1996):pp. 75-104.

[14] R. Szeliski, and J. Coughlan, "Spline-based image registration." , *International Journal of Computer Vision* 22, no. 3 (1997):pp. 199-218.

[15] T. Brox, C. Bregler, and J. Malik, "Large displacement optical flow.", *In Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 41-48. IEEE, 2009.

[16] Y. Zheng," Trajectory data mining: an overview", *ACM Transactions on Intelligent Systems and Technology (TIST)*, 6(3),(2015): 29.

[17] Y. Zhao, S.Shang, Y.Wang, B.Zheng, Q. V. H. Nguyen, and K. Zheng," Rest: A reference-based framework for spatio-temporal trajectory compression", *In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*,(2018, July):pp. 2797-2806.

[18] P. Yin,M.Ye,W. C.Lee, and Z.Li (2014, May)," Mining GPS data for trajectory recommendation", *In Pacific-Asia Conference on Knowledge Discovery and Data Mining Springer, Cham.*: pp. 50-61.

[19] S.Yu, Y.Kaneko, E. Bae, C. E.Stahl, Y. Wang, H. van Loveren, ..., and C. V. Borlongan,"Severity of controlled cortical impact traumatic brain injury in rats and mice dictates degree of behavioral deficits", *Brain research*,1287, (2009): pp. 157-163.

[20] P. M.Washington, P. A.Forcelli, T. Wilkins, D. N.Zapple, M.Parsadanian, and M. P. Burns, "The effect of injury severity on behavior: a phenotypic study of cognitive and emotional deficits after mild, moderate, and severe controlled cortical impact injury in mice.",*Journal of Neurotrauma*29(13)(2012).:pp. 2283-2296.

[21] O.V.S.Sarma, M.Betancur, R.Pidaparti, and L. Karumbaiah," Lesion Volume Estimation from TBI-MRI.",*Progress in advanced computing and intelligent engineering : proceedings of ICACIE 2016. International Conference on Advanced Computing and Intelligent Engineering (2016 : Bhubaneswar, India)*,563,pp. 197-207.

[22] S. Belongie, J. Malik, and J. Puzicha, "Shape context: A new descriptor for shape matching and object recognition.", *In Advances in neural information processing systems*,2001, pp. 831-837.

[23] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts.", *IEEE Transactions on Pattern Analysis And Machine Intelligence*, April 2002.

[24] S. Belongie, G. Mori, and J. Malik, "Matching with shape contexts.", *In Statistics and Analysis of Shapes*,Birkhuser Boston, 2006, pp. 81-105

[25] J. Sivic, and A. Zisserman, "Efficient visual search of videos cast as text retrieval.", *IEEE transactions on pattern analysis and machine intelligence* 31, no. 4 (2009): 591-606.

[26] Z. S. Harris, "Distributional structure.", *Word* 10, no. 2-3 (1954): 146-162.

[27] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz, "A Bayesian approach to filtering junk e-mail.", *In Learning for Text Categorization: Papers from the 1998 workshop*, vol. 62, pp. 98-105. 1998.

[28] J. J. Hull, and S. N. Srihari, "Experiments in text recognition with binary n-gram and viterbi algorithms.", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5 (1982): 520-530.

[29] L. Fei-Fei, and P. Perona, "A bayesian hierarchical model for learning natural scene categories.", *In Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 2, pp. 524-531. IEEE, 2005.

[30] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints.", *In Workshop on statistical learning in computer vision, ECCV*, vol. 1, no. 1-22, pp. 1-2. 2004.

[31] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman, "Discovering objects and their location in images.", *In Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 1, pp. 370-377. IEEE, 2005.

[32] S. Ullman, M. Vidal-Naquet, and E. Sali, "Visual features of intermediate complexity and their use in classification.", *Nature neuroscience* 5, no. 7 (2002): 682.

[33] K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. M. Blei, and M. I. Jordan, "Matching words and pictures.", *Journal of machine learning research* 3, no. Feb (2003): 1107-1135.

[34] K. Mikolajczyk, and C. Schmid, "Comparison of affine-invariant local detectors and descriptors.", *In Signal Processing Conference, 2004 12th European*, pp. 1729-1732. IEEE, 2004.

[35] A. Zisserman, F. Schaffalitzky, and J. Sivic, "Object retrieval.", *U.S. Patent Application 10/820,671, filed October 13, 2005.*

[36] D. G.Lowe, "Object recognition from local scale-invariant features." In Computer vision, 1999., *The proceedings of the seventh IEEE international conference on*, vol. 2, pp. 1150-1157. Ieee, 1999.

[37] H. W.Kuhn, "The Hungarian method for the assignment problem.", *Naval research logistics quarterly* 2, no. 12 (1955): 83-97.

[38] R. Jonker, and A. Volgenant, "A shortest augmenting path algorithm for dense and sparse linear assignment problems.", *Computing* 38, no. 4 (1987): 325-340.

[39] F. Bookstein, "Principal warps: Thin-plate splines and the decomposition of deformations.", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(6):567-585, 1989.

[40] V. Jean-Philippe, K. Tsuda, and B. Schlkopf, "A primer on kernel methods.", *Kernel methods in computational biology* 47 (2004): 35-70.