AN ABSTRACT OF THE THESIS OF

Vikedo Terhuja for the degree of Master of Science in Computer Science presented on December 3, 2015.

Title: Automatic Detection of Possessions and Shots from Raw Basketball Videos.

Abstract approved: ____________________________

Alan P. Fern

Sports analytics is rapidly evolving today through the use of computer vision systems that automatically extract huge amount of information inherently present in multimedia data without much human assistance. This information can facilitate a better understanding of patterns and strategies in various sports. However, for non-professional teams, due to expense and large variations in the videos, there are no reliable systems for automatic extraction of game statistics and information. In this thesis, we consider two problems in basketball sports analytics with the goal of being robust to wide differences in video footage and being completely automatic. First, we consider the problem of parsing a game into possessions and inferring which team has possession at any time. This information provides basic statistics and the ability to easily navigate a game in terms of possessions. Second, we consider detecting shots, which allows for shot count statistics to be automatically generated. Our experiments across a wide variety of basketball video show that the approaches are accurate and robust to large differences in video type.
Automatic Detection of Possessions and Shots from Raw Basketball Videos.

by

Vikedo Terhuja

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented December 3, 2015
Commencement June 2016
Master of Science thesis of Vikedo Terhuja presented on December 3, 2015.

APPROVED:

__________________________________________________________________________

Major Professor, representing Computer Science

__________________________________________________________________________

Director of the School of Electrical Engineering and Computer Science

__________________________________________________________________________

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

__________________________________________________________________________

Vikedo Terhuja, Author
ACKNOWLEDGEMENTS

Firstly, I would like to thank my adviser Dr. Alan Fern for giving me the opportunity to be a part of The Digital Scout Project at Oregon State University. I express my sincere gratitude to him for his valuable advises, guidance and most importantly for instilling in me the ability to approach problems from different perspectives. I would also like to thank Dr. Sinisa Todorovic for his insightful suggestions and ideas. A very special thanks to Dr. Fuxin Li and Dr. Harry Yeh for agreeing to be in my committee. I also wish to thank Sheng Chen, Behrooz Mahasseni, David Stuve and Anirban Roy for their assistance and advises during my research work.

Finally I wish to thank my mom and dad for their love and constant support. My brothers and sisters, my best friend and my spiritual brother - Sentitemsu Imsong for their constant motivation and encouragement. Jeff Harris, Teresa Harris, Sai Sree and friends in Corvallis for always motivating me and making my adventure at Oregon State University fun, fruitful and memorable.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>2 Related Work</td>
<td>5</td>
</tr>
<tr>
<td>2.1 Computer Vision Technology in Basketball</td>
<td>5</td>
</tr>
<tr>
<td>3 Description of Data</td>
<td>8</td>
</tr>
<tr>
<td>3.1 Dataset Details:</td>
<td>10</td>
</tr>
<tr>
<td>4 Problems</td>
<td>12</td>
</tr>
<tr>
<td>4.1 Possession Detection</td>
<td>12</td>
</tr>
<tr>
<td>4.2 Shots Detection</td>
<td>16</td>
</tr>
<tr>
<td>5 Possession Detection</td>
<td>18</td>
</tr>
<tr>
<td>5.1 Overview of Proposed Method</td>
<td>18</td>
</tr>
<tr>
<td>5.2 Reference Matching</td>
<td>18</td>
</tr>
<tr>
<td>5.2.1 SIFT Features</td>
<td>20</td>
</tr>
<tr>
<td>5.2.2 k-Nearest Neighbors</td>
<td>24</td>
</tr>
<tr>
<td>5.3 Segment Detection</td>
<td>25</td>
</tr>
<tr>
<td>5.3.1 Match Difference</td>
<td>25</td>
</tr>
<tr>
<td>5.3.2 Segment Indexing</td>
<td>26</td>
</tr>
<tr>
<td>5.4 Automatic Reference Frame Selector</td>
<td>29</td>
</tr>
<tr>
<td>5.4.1 Camera State Extraction</td>
<td>30</td>
</tr>
<tr>
<td>5.4.2 Reference Selector</td>
<td>31</td>
</tr>
<tr>
<td>5.5 Results</td>
<td>33</td>
</tr>
<tr>
<td>5.5.1 Evaluation of Automatic Reference Selector</td>
<td>33</td>
</tr>
<tr>
<td>5.5.2 Evaluation of Possession Detection</td>
<td>33</td>
</tr>
<tr>
<td>5.6 Discussions</td>
<td>34</td>
</tr>
<tr>
<td>5.6.1 Alternate Method</td>
<td>35</td>
</tr>
<tr>
<td>6 Shots Detection</td>
<td>37</td>
</tr>
<tr>
<td>6.1 Overview of Proposed Method</td>
<td>37</td>
</tr>
<tr>
<td>6.2 Tracking</td>
<td>38</td>
</tr>
<tr>
<td>6.2.1 Structured Output (Struck) Tracker</td>
<td>39</td>
</tr>
<tr>
<td>6.2.2 Haar-like Features</td>
<td>41</td>
</tr>
</tbody>
</table>
# TABLE OF CONTENTS (Continued)

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.2.3 Backboard Tracking</td>
<td>42</td>
</tr>
<tr>
<td>6.3 Ball Entry Detection</td>
<td>45</td>
</tr>
<tr>
<td>6.3.1 Dense Optical Flow</td>
<td>45</td>
</tr>
<tr>
<td>6.3.2 Grid Optical Flow</td>
<td>46</td>
</tr>
<tr>
<td>6.3.3 Active Cells</td>
<td>46</td>
</tr>
<tr>
<td>6.3.4 Noise Elimination</td>
<td>48</td>
</tr>
<tr>
<td>6.3.5 Adaptive Threshold</td>
<td>50</td>
</tr>
<tr>
<td>6.4 Results</td>
<td>51</td>
</tr>
<tr>
<td>6.4.1 Evaluation of Tracker</td>
<td>52</td>
</tr>
<tr>
<td>6.4.2 Evaluation of Ball Entry Detection</td>
<td>52</td>
</tr>
<tr>
<td>6.5 Discussions</td>
<td>54</td>
</tr>
<tr>
<td>6.5.1 Alternate Method:</td>
<td>55</td>
</tr>
<tr>
<td>7 Conclusion</td>
<td>56</td>
</tr>
<tr>
<td>7.1 Challenges</td>
<td>56</td>
</tr>
<tr>
<td>7.2 Future Work</td>
<td>57</td>
</tr>
<tr>
<td>7.2.1 Improvement of Shots Detection</td>
<td>57</td>
</tr>
<tr>
<td>7.2.2 Shots Identification</td>
<td>57</td>
</tr>
<tr>
<td>Bibliography</td>
<td>57</td>
</tr>
<tr>
<td>Figure</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>1.1 Reviewing decisions in (a) cricket and (b) tennis using Computer Vision technology.</td>
<td>2</td>
</tr>
<tr>
<td>1.2 Possession of the ball and action on the right side of the court.</td>
<td>3</td>
</tr>
<tr>
<td>1.3 Frames capturing motion of the ball towards the rim from a shot.</td>
<td>4</td>
</tr>
<tr>
<td>2.1 SportVU: Real-time tracking of multiple players on the court.</td>
<td>6</td>
</tr>
<tr>
<td>3.1 Standard-Angle-Videos.</td>
<td>9</td>
</tr>
<tr>
<td>3.2 Wide-Angle-Videos.</td>
<td>9</td>
</tr>
<tr>
<td>3.3 Narrow-Angle-Videos.</td>
<td>9</td>
</tr>
<tr>
<td>3.4 Unaligned-Angle-Videos.</td>
<td>10</td>
</tr>
<tr>
<td>4.1 Visualization of optical flow vectors: Motion of players (a) With camera motion and (b) Without camera motion.</td>
<td>13</td>
</tr>
<tr>
<td>4.2 Average optical flow magnitude over a sequence of frames.</td>
<td>14</td>
</tr>
<tr>
<td>4.3 Overlay visualization of a frame (orange borders) on panoramic view of the court.</td>
<td>14</td>
</tr>
<tr>
<td>4.4 Visualization of optical flow vectors (a) w/o camera motion elimination (b) w/ camera motion elimination during transition.</td>
<td>15</td>
</tr>
<tr>
<td>4.5 Average optical flow magnitude over a sequence of frames.</td>
<td>15</td>
</tr>
<tr>
<td>4.6 (a) Left, (b) Middle and (c) Right sides of the court. Camera points in the direction of the action in the basketball court.</td>
<td>16</td>
</tr>
<tr>
<td>4.7 Trajectory of ball from a shot.</td>
<td>17</td>
</tr>
<tr>
<td>5.1 Flow chart of Possession Detection.</td>
<td>19</td>
</tr>
<tr>
<td>5.2 (a) Left Reference frame vs Left Frame. (b) Left Reference Frame vs Right Frame. The reference frame is with yellow borders. The number of matching points on Top is more than Bottom.</td>
<td>20</td>
</tr>
</tbody>
</table>
LIST OF FIGURES (Continued)

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3</td>
<td>Plot of #matches of each frame with Left and Right references.</td>
<td>21</td>
</tr>
<tr>
<td>5.4</td>
<td>Construction of Difference-of-Gaussian at levels of octaves.</td>
<td>22</td>
</tr>
<tr>
<td>5.5</td>
<td>Detection of Maxima and Minima of a sample point.</td>
<td>22</td>
</tr>
<tr>
<td>5.6</td>
<td>Detection of Maxima and Minima of a sample point.</td>
<td>23</td>
</tr>
<tr>
<td>5.7</td>
<td>Plot of the number of matches with Left and Right references for a sequence of frames.</td>
<td>25</td>
</tr>
<tr>
<td>5.8</td>
<td>Plot of the difference of Left Matches and Right Matches, $Diff_{LR}$, for a sequence of frames.</td>
<td>26</td>
</tr>
<tr>
<td>5.9</td>
<td>Raw signal ($Diff_{LR}$) and smoothed signal ($SDiff_{LR}$).</td>
<td>27</td>
</tr>
<tr>
<td>5.10</td>
<td>(a) Smoothed signal ($SDiff_{LR}$) and (b) locally normalized signal ($nSDiff_{LR}$).</td>
<td>27</td>
</tr>
<tr>
<td>5.11</td>
<td>Forward threshold ($Th_f$) and Backward Threshold ($Th_b$) to select the end and start interval of a segment.</td>
<td>29</td>
</tr>
<tr>
<td>5.12</td>
<td>Four possible camera states: (a) Zoom, (b) Moving Right, (c) Still, (d) Moving Left.</td>
<td>30</td>
</tr>
<tr>
<td>5.13</td>
<td>Representative frames of five automatically selected candidates.</td>
<td>32</td>
</tr>
<tr>
<td>5.14</td>
<td>The three signals (red, blue, yellow) represents the number of matches of each frame with Left, Right and Mid reference frames. Out-of-play frames are characterized by low values for all three signals.</td>
<td>36</td>
</tr>
<tr>
<td>6.1</td>
<td>Flow chart of Shots Detection.</td>
<td>38</td>
</tr>
<tr>
<td>6.2</td>
<td>Four types of rectangle features within the target: (a) Two-rectangle feature (horizontal), (b) Two-rectangle feature (vertical), (c) Three-rectangle feature (horizontal), (d) Four-rectangle feature.</td>
<td>41</td>
</tr>
<tr>
<td>6.3</td>
<td>The value of the integral image at 1 is the sum of all pixels in rectangle A, the value at location 2 is A+B, at location 3 is A+C and at location 4 is A+B+C+D.</td>
<td>42</td>
</tr>
<tr>
<td>Figure</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>6.4</td>
<td>Manual initialization of tracker slightly over and beyond the backboard: (a) On the Left backboard include extra region on the right, (b) on the Right backboard include extra region on the left as shown with the yellow arrows.</td>
<td>43</td>
</tr>
<tr>
<td>6.5</td>
<td>Visualization of optical flow vectors on a sequence of backboard frames when ball enters the backboard.</td>
<td>46</td>
</tr>
<tr>
<td>6.6</td>
<td>Four overlapping cells of the grid.</td>
<td>47</td>
</tr>
<tr>
<td>6.7</td>
<td>Active cell (green box) of a sequence of frame of the backboard with the ball.</td>
<td>47</td>
</tr>
<tr>
<td>6.8</td>
<td>Active cell (green box) of a sequence of frames of backboard without the ball.</td>
<td>47</td>
</tr>
<tr>
<td>6.9</td>
<td>Failure case of active cells not tracking the ball in some frames.</td>
<td>48</td>
</tr>
<tr>
<td>6.10</td>
<td>(a) Magnitude of active cells with noise (b) Magnitude of active cells after noise cancellation.</td>
<td>49</td>
</tr>
<tr>
<td>6.11</td>
<td>(a) Results of tracker (a) without re-initialization (b) with re-initialization.</td>
<td>53</td>
</tr>
</tbody>
</table>
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Summary of the dataset.</td>
<td>11</td>
</tr>
<tr>
<td>5.1</td>
<td>Result of Reference frames selector: Manual vs Automatic.</td>
<td>34</td>
</tr>
<tr>
<td>5.2</td>
<td>Results of Ball Possession detection.</td>
<td>34</td>
</tr>
<tr>
<td>5.3</td>
<td>Result of comparison of k-mean vs proposed method.</td>
<td>36</td>
</tr>
<tr>
<td>6.1</td>
<td>Results of Shot Detection with 2-Clusters Heuristic threshold.</td>
<td>54</td>
</tr>
<tr>
<td>6.2</td>
<td>Results of Shot Detection with Adjacency Heuristic threshold.</td>
<td>54</td>
</tr>
</tbody>
</table>
Chapter 1: Introduction

The abundance of video cameras such as in phones, tablets and handy-cams and the ease of operating them has increased the collection of multimedia information by significant amount over the last decades. Videos of different sporting events are of special interest because sports analysts, scouts, managers, coaches and even sports person watch the game videos multiple times in order to improve, strategize or collect statistics of the game. However in a full length of a video not all events are interesting for example in a Basketball game when players are transitioning from one side of the court to the other there is not much actions of interest or in case of American Football the moments before the start of a play (snap of the ball) is not interesting as it may consist of substitutions or the camera is panning or coaching to the spectators or the coaching staffs etc. Employing human to perform the task of identifying interesting events and understanding of strategies and formations is time consuming, inefficient and prone to human errors. These manual and mundane activities can be accelerated and automated by the use of smart systems that can automatically detect events of interest and understand structures and patterns in sports.

In recent years with the advancement in Computer Vision technology, extensive research efforts have been devoted for content-based analysis on sports video due to its high commercial significance. Computer Vision is becoming more matured in academic research and various industries are finding wide range of its application to solve problems in areas of medicines, military, semiconductor manufacturing, sports, photography and other diagnostic fields. Computer Vision is a field in Artificial Intelligence that involves techniques of processing, extracting, analyzing and understanding contents in multimedia information, which are high dimensional data, in order to transform the data to simplified numerical information that can be used to make decisions and infer conclusions. The most fundamental goal of Computer Vision is to attempt to model the human eye and the brain i.e. the human vision. Like the human eye that can see an object and send that information to the brain to recognize what the object is and decide on a reaction to the information, Computer Vision systems aim to achieve the same
mechanism. Although the Computer Vision is still a long way behind the capabilities of the human vision it is making steady progress in areas of feature extraction, pattern recognition and image classification.

Computer Vision technology has highly revolutionized some sports especially when an in-game review of official decision is needed to override and ambiguous decision. In cricket, an appeal for a Leg-Before-Wicket (LBW) the trajectory of a ball can be estimated by analyzing the ball before hitting the batsman’s pad and with the interpolated information determine whether the ball is actually hitting the wicket. Even in soccer determining whether the ball crossed the goal line and the total distance traversed by a player can be determined with high accuracy. This high demand of sophisticated analyzing systems in the sports community to take sports to a higher level and to evaluate team or individual players performance has encourage and motivated a lot of research community to take up this challenge of improving sports analytics. Analyzing sports involves understanding of the sports, building mathematical models, analyzing high dimensional data and computer vision technologies to understand and analyze the patterns and semantics in the games of any sports.

Figure 1.1: Reviewing decisions in (a) cricket and (b) tennis using Computer Vision technology.

In this thesis, the main focus is in the application of Computer Vision techniques in the sports of basketball. Basketball is a team sport where the teams compete by shooting the ball into the opponent teams basket. Basketball coaches, scouts and analysts use the videos of recorded games in order to strategize against opponents strong plays, analyze the team or an individual players performance etc. Besides being time consuming to
analyze the videos manually, humans cannot focus on movements of all the players at once and can miss out interesting events when focusing on other activities. There is strong demand to apply Computer Vision to perform the analysis task in this sport due to the availability of massive amount of video footage.

![Image](image1.png)

**Figure 1.2: Possession of the ball and action on the right side of the court.**

One of the primary steps of analyzing a basketball video is to know which team has the possession of the ball and the periods when the teams have the ball possession as in Figure 1.2. Basketball videos are generally shot from a position outside the court orthogonal to the mid-court line. The camera pans to the left or right side of the court and slightly zoom in during free throws or focus on players after a basket is made. The movement of the players combined with camera motion poses a difficult challenge in determining which team has the ball. Moreover it is a very difficult problem to track the ball due to occlusion, light intensity and lack of distinguishing contrasts of the ball with the spectators in the background. Once we can determine which team has the possession of the ball this information can be used to analyze the actions of a selected team. A metric to evaluate a performance of a team is the attempts of scoring points by shooting the ball into the opponents basket or the points given away to the opponent team. The successful shooting percentage of one of the best shooters (Stephen Curry of Golden State Warriors) in NBA basketball game is less than 50%. The overall percentage of the team may be lower. Shooting percentage may also vary at different locations of the court.
These statistics are very interesting and the ability to detect a shot automatically has high commercial value as huge data-set of basketball videos can be parsed to extract the time frame when shots were taken which further can be extended to distinguish between shots actually made and shots missed.

The main goal of the thesis is to develop efficient and robust methods for automatic detection of ball possession and detection of shots by analyzing raw videos of high school and college basketball games. The automation of these operations can be incorporated and built into smart systems that can facilitate coaches, analysts and scouts in their tasks of scouting, data analytics, analyzing critical information in games and player’s performances. Unlike professional NBA (National Basketball Association) broadcast videos raw (non-professional) videos are quite different: there is no uniformity of the angle from which the videos are shot, the presence of sudden movement and shaky videos caused by hand operated cameras and the lack of smooth flow of the camera makes it more challenging. Our methodologies provides a new perspective of solving these problems and demonstrate the ability to get accurate results and open the possibilities of future research in the field of intelligent sports analyzing systems especially for raw videos.

![Figure 1.3: Frames capturing motion of the ball towards the rim from a shot.](image)

The thesis is organized as follows: In Chapter 2 we discuss the related work on using Computer Vision in sports video analysis. Chapter 3 describes the type of basketball videos on which our approaches have been experimented. Chapter 4 overviews the problems in video analysis that our approaches are trying to solve and in Chapter 5 and 6 we give a detailed description of our proposed methods for detection of ball possession and shots detection, report the result of the experiments and offer discussion towards the result. In Chapter 7 we conclude by highlighting the challenges and providing suggestions for future research in using Computer Vision to build smart systems for analyzing sports videos.
Chapter 2: Related Work

2.1 Computer Vision Technology in Basketball

In this chapter we present an overview of Computer Vision based applications and research in team sports particularly in basketball. The popularity of basketball around the world and the revenue it generates especially in USA where professional, college and high school basketball is an integral part of people’s lives has generated huge commercial interest for many industries over the years. Applications ranging from automatic game statistics collection, game result prediction, content indexing and segmentation, tracking single or multiple players, pattern recognition and semantic event summarizations are some areas the research communities are focusing today in sports video analysis.

The application of Computer Vision technology in sports has been going on for over a decade providing in-depth statistics, better viewing experience, improving referee decisions and presenting a structured approach to understand the dynamics of sports. In basketball the most advanced commercial computer vision system to date is the StatsLLC - SportVU [4] that is based on a technology from Israeli Military. It is a six cameras vision system mounted at strategically symmetric locations in a basketball arena that can record data points of everything that is in motion on the basketball court with its synchronized cameras. It detects and simultaneously tracks all players, the referees and the ball throughout the game at a rate of 25 times a second. The automated data collection system of SportVU can keep track of dribbles, passes, screens, shots, rebounds, speed and distance traversed by players, etc. In the NBA 2013-14 season SportVU was installed in all the NBA arenas and NBA released SportVU tracking data to the public which sparked an interest and attracted research communities for advanced research in basketball. Figure 2.1 is a snapshot of tracking multiple players by SportVU system. The trajectories of the players and the ball from the tracking data of SportVU has been used as the dataset in many advanced research in basketball. Using these data games can be passed into offense, defense, timeout using a probabilistic model and extract
temporal semantic description of activities to compare with manually defined template of activities in the database [1]. Disney Research research analyses the offense and defense dynamics by observing the spatiotemporal changes in team’s formation and computing the percentage of shots made under pressure and shots made when the player is open [2]. Research on coordinated human activity using the trajectories of all players on the court including the trajectory of the ball for activity recognition and performance evaluation [3] is shown to be effective using the SportVU data.

Figure 2.1: SportVU: Real-time tracking of multiple players on the court.

The drawback of SportVU is that the system is expensive and is available only to the 30 professional NBA teams. The system costs around $100,000 per season. Recently three college basketball teams procured this system for better assessment of players, evaluate of team defensive and offensive plays and utilize players and shape the team coordination. Teams with this added arsenal will naturally perform better than the teams without technological assistance. However there are still scores of college and high school basketball teams that do not have the luxury of procuring the SportVU system and still rely on videos captured with semi-professional cameras for evaluation of team’s performance and other statistics. The availability of huge dataset of such broadcast basketball videos provides a hot area for research in sports video analysis using computer vision. There is a separate research community devoted to research on broadcast and raw basketball videos. Analyzing raw videos is very challenging due to lack of uniformity in the videos, inability to extract 3D location of players and ball accurately from the
2D video frames and mostly due to abrupt camera motion from inexperienced camera handlers. In spite of these challenges the ability to extract information from the videos such as the trajectories of players by tracking single or multiple players, coordinated dynamics of players, ball detection and tracking and detecting shots and keeping track of scores can provide statistical data and enhance advanced metrics for coaches and scouts.

Some of the research work done with broadcast basketball videos includes parsing of the videos into segments of zoom, fade, pan, zoom, etc using HMM (Hidden Markov Model) [15], detection of player-possession by tracking both ball and players [31], panoramic viewing experience of the game by automatic panorama generation of the court and overlay frames on the panorama [6, 7, 8, 5]. From the panorama view camera motion which adds noise in the data are eliminated effectively. Collective motion of players provide information about the type of activity and can be used to detect events as demonstrated in football and basketball videos in [10, 11, 12]. Detecting the play-field i.e., the region where the play is actually taking place without the audience in the background is an important sub-problem which is beneficial for higher analysis. Tekalp et.al [18, 17] has demonstrated a robust detection of the region of the court using a dominant color region detection algorithm. In the area of tracking researchers have focused on efficient and high performance single player and multi-player tracker [19, 35, 36, 37] as well as tracking the ball. Trajectory-based tracker has been applied in many sports to track the ball [31] quite effectively by estimating its motion and trajectory. The trajectory information of the players and the ball are used for high level analysis similar to the research done on SportVU trajectories. The velocity of ball and the shooting angle can also be estimated using the ball trajectory [31]. Besides tracking, detection of events such as shots is also an important research problem. Shot identification and scoring event identification using Dynamic Bayesian Network (DBN) and Bayesian Belief Network (BBN) with low level features has been shown to be effective [32, 33]. Shot type and shoot position extraction has also been done on broadcast videos by tracking and estimating the 2D trajectory of the ball and the 3D location of the player [34]. There are some of the current application of computer vision in basketball.
Chapter 3: Description of Data

This chapter describes the dataset on which our methodologies have been experimented. Our dataset consists of raw basketball videos of college and high school games. Raw basketball videos unlike professional NBA videos broadcasted in sports channels do not have uniformity in video quality as they are not recorded by professionals and there are no set of guidelines for shooting a basketball game. Another characteristic of raw videos is the absence of replays and slow motions shots. The variation in quality and the angle from which the videos are shot, sudden jerk on camera due to the use of hand held cameras instead of cameras mounted on stationary tripods, the lack of smooth panning and zooming and the lack of good coverage of the basketball court presents a big challenge to formulate robust and accurate approaches to solve the problems of ball possession and shots detection. Our methodologies are experimented on nine raw basketball video of high school and college basketball games with varying lengths. We categorize our dataset into four types based on the camera angle and position during the recording:

1. **Standard-Angle-Video**: Standard angle videos are those with similar look and feel as those of professional NBA or College broadcast games. However this type of videos in our raw dataset has sudden camera movements and shaky panning of the camera. There is no zoom in or zoom out in this type unlike professional videos where there is zoom in and out especially during free throws or after a player made a shot the camera is zoomed and focused on the player. In this type of videos about 50% of the court is always visible as shown in Figure 3.1.

2. **Wide-Angle-Video**: Wide angle videos are those that are recorded from a high location with a wide view of the court. These videos are characterized by smooth panning of the camera and very mild camera jerks. There are no zoom-ins or zoom-outs and due to the wide angle the ball is usually tiny and very visible even to the human eye to discern whether a shot is made or missed. In this type of videos about 75% of the court is always visible as shown in Figure 3.2.
3. **Narrow-Angle-Video**: This type of videos are recorded from similar position as the standard videos but the angle is narrower than the standard. There is slight zoom in and zoom out especially during free throws. Abrupt camera movements, shaky camera panning and partial cover of the backboard are some of the challenges in this type of videos. About 35-40% of the court is always visible as shown in Figure 3.3.
4. **Unaligned-Angle-Video:** The position of the camera is usually at an elevation outside the court and aligned to the center line of the basketball court. But in this type the video is recorded from a location closer to one of the sides of the court. The viewing angle is similar to the standard angle however due to the camera position the two sides of the court are not symmetric and depending on the direction the camera is facing the percentage of coverage of the court may vary as in Figure 3.4, when the camera is on the Right only 30% of the court is visible and on the Left around 70% is visible.

![Figure 3.4: Unaligned-Angle-Videos.](image)

### 3.1 Dataset Details:

Our dataset consists of 2 standard-angle type videos, 2 wide-angle type videos, 3 narrow-angle type videos and 2 unaligned-angle type videos. We have not considered any NBA broadcast videos although they are of the Standard-Angle-Video type because they have the additional characteristics of zoom in and out on players that scored the basket and slow motion replays of players making shots.
<table>
<thead>
<tr>
<th>ID</th>
<th>Video Type</th>
<th>Length(mins)</th>
<th>FPS</th>
<th>Size</th>
<th>Total Frames</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Narrow</td>
<td>67:48</td>
<td>29.97</td>
<td>1280x720</td>
<td>121941</td>
</tr>
<tr>
<td>3</td>
<td>Narrow</td>
<td>63:05</td>
<td>29.97</td>
<td>1280x720</td>
<td>113445</td>
</tr>
<tr>
<td>6</td>
<td>Unaligned</td>
<td>48:57</td>
<td>29.91</td>
<td>1280x720</td>
<td>88023</td>
</tr>
<tr>
<td>8</td>
<td>Unaligned</td>
<td>12:03</td>
<td>29.97</td>
<td>720x420</td>
<td>21670</td>
</tr>
<tr>
<td>9</td>
<td>Wide</td>
<td>106:09</td>
<td>30</td>
<td>1280x720</td>
<td>191163</td>
</tr>
<tr>
<td>10</td>
<td>Standard</td>
<td>73:29</td>
<td>29</td>
<td>1280x720</td>
<td>132257</td>
</tr>
<tr>
<td>11</td>
<td>Wide</td>
<td>88:29</td>
<td>29</td>
<td>1280x720</td>
<td>160090</td>
</tr>
<tr>
<td>12</td>
<td>Standard</td>
<td>53:15</td>
<td>30</td>
<td>1280x720</td>
<td>95497</td>
</tr>
<tr>
<td>13</td>
<td>Narrow</td>
<td>74:42</td>
<td>30</td>
<td>1280x720</td>
<td>134368</td>
</tr>
</tbody>
</table>

Table 3.1: Summary of the dataset.
Chapter 4: Problems

In this chapter we describe in detail the problems we are addressing in the thesis. Two areas where computer vision can facilitate the analyzing of videos are automatic detection of events of interest and collection of statistics such as the number of attempts a team made to shoot the ball into the opponent’s basket. The initial groundwork required before any advanced analysis of a team’s defense or offense plays and quantify its performance is to know which team has the possession of the ball and which side of the court is the action taking place. This prior knowledge about the team and the side of court can be used to enhance further analysis such as shots detection as we have information of what to look for in that part of the court.

4.1 Possession Detection

The general purpose of ball possession detection is to know which team or player has the possession of the ball at a particular instance of time. The player which is closest to the ball can be assumed to have the possession of the ball. However it is not that simple because the ball is not always visible throughout the length of the video due to occlusion or going out of the frame and during a pass or a shot, when the ball is in the air, the player closest to the ball does not necessarily have the possession of the ball. Even if it can be taken for granted that the ball can be tracked successfully we have another problem of tracking the players. Players of the two teams have contrasting jerseys but tracking human itself is a difficult problem because of the non-rigidity of the form of human in action and the difficulty is amplified as we would need to track multiple players on the court. All these factors make ball possession detection a difficult problem. However it is an interesting problem because it serves as a foundation to all other advanced analysis. Its utility is also found in indexing videos to help in the ease of navigation to events of interest in a game.

Since the possession of the ball can belong to either of the two teams, we redefine ball possession based on when the ball is either on the left side or the right side of the
Figure 4.1: Visualization of optical flow vectors: Motion of players (a) With camera motion and (b) Without camera motion.

court. For example: When the ball is in the left side of the court the team on the right has the possession and is attempting to shoot the ball into the left basket and vice versa. There are instances when the ball is stolen by the left team from the right team on the left side of the court and technically the ball is no longer in the possession of the right team however until the ball is taken to the right side of the court it is considered that the possession is unchanged. This loose definition does not have adverse effect on the notion of ball possession because in most cases after a steal the next action the players perform is move to the opposite side of the court. The problem is now reformulated as finding if a frame belongs to the left or right court. Two obvious approaches are:

1. **Using Camera Motion**: The panning of camera during the game is a good indication of the possession change and it is clearly observable that the camera motion is highest during transition from one side of the court to the other. This motion in camera can be detected using Optical Flow [30] however it is not a reliable metric to determine the direction of the camera due to the noise caused by unsteady panning and shaky videos. Figure 4.1 is a visualization of the magnitude and direction of optical flow vectors between two consecutive frames. One depicts the case with camera motion and the other without camera motion at the Right side of the court. In the plot of average optical flow magnitude of a sequence of frames in Figure 4.2, there is peak in the magnitude during transition but it is inconsistent as there are other peaks cause by sudden camera motions.
2. **Using Players Motion**: Another indication of possession change is the aggregate movement of players at the sides of the court which is generally greater than the aggregate movement during transition if we eliminate the camera motion. A technique to eliminate camera motion is to create panorama of the court and overlay frames over the panorama \([6, 7, 8]\) as shown in Figure 4.3. Although the camera motion is eliminated as seen in the Figure 4.4, this approach of using the total optical flow is not a good metric to determine the sides due to noise from light intensity especially at the corners of the overlay frames as seen in the plot of aggregate optical flow magnitude over a sequence of frames in Figure 4.5.

Our goal is to automatically detect ball possession i.e., to detect when the ball is in the left side or the right side of the court. But as mentioned earlier due to the difficulty
Figure 4.4: Visualization of optical flow vectors (a) w/o camera motion elimination (b) w/ camera motion elimination during transition.

Figure 4.5: Average optical flow magnitude over a sequence of frames.

in tracking the ball especially throughout the length of a basketball game we approach this problem from a different perspective: since we know that the camera follows the action in the court and each side has distinctive features as seen in Figure 4.6 we make use of these observations in determining the possession of the ball which will be discussed in detail in the following chapters.
Figure 4.6: (a) Left, (b) Middle and (c) Right sides of the court. Camera points in the direction of the action in the basketball court.

4.2 Shots Detection

A shot in a basketball game is the action taken by a player in his/her attempt to shoot the ball into the opponent’s basket to score points. It is different from the action taken by a player to pass the ball to other players although in both cases the ball is projected from the hand of the player into the air and towards the destination. The key distinction between the two is the destination of the ball. In other words we need to know where the ball is heading to determine whether an action is a shot. This would require tracking the ball but as discussed in Chapter 3 and the above section, our dataset consists of four types of videos and in each type the ball appears different in size, contrast and color. In some cases the ball is not clearly visible even to the naked human eye. It is best visible in the Narrow-Angle type videos because of the closeup shot of the court but the Narrow-Angle videos do not have a good cover of the court and hence there are instances when the ball moves outside the frame of the video and it cannot be tracked. These challenges makes shots detection a difficult and interesting problem. Shots detection has a lot of significance in automatic collection of game statistics as based on this statistics the strength and weakness of a team or the opponent team can be evaluated.

The goal is to automatically detect shots events in a video given the information of ball possession and the side of the court the action is taking place. Since we are interested only when the ball is moving towards the basket as seen in Figure 4.7 our approach monitors the ball that are heading into the vicinity of the rim and the backboard and considers them as valid shots which will be discussed further in the following chapters.
Figure 4.7: Trajectory of ball from a shot.
Chapter 5: Possession Detection

In this chapter we will discuss our robust method to automatically detect ball possession for all types of raw videos. As discussed in the last chapter the challenges of tracking the ball and the players throughout the length of the video is hard and too computationally expensive and unrealistic to solve the problem of possession detection. We instead use the knowledge that the camera is facing the region with activity on the basketball court and that the camera will repeatedly pan left and right with occasional zoom-in and zoom-out to capture the actions at the ends of the court. Our method based on these observation is computationally efficient, robust and accurate.

5.1 Overview of Proposed Method

The camera pans to a position on the left side where it can capture most of the left court and similarly on the right side of the court. We observe this repetitive pattern of panning left and right in all videos. The basic common objects on both sides of the court are the lines on the court, backboard, rim and in some case the logo. Although symmetrical there are distinctive visual properties on both sides. We reformulate the problem into a matching problem. Thus our method makes use of this observation by choosing two reference images/frames: left side and right side of the court and performs a matching of all the frames in the video with the references to determine if a frame belongs to the right or left side of the court. In Figure 5.1 the flow chart give an overview of the approach for possession detection.

5.2 Reference Matching

Pattern matching is a recurrent problem in Computer Vision that is solved with variety of techniques. Two images are said to match if the key-points in the images match. There are many techniques of extracting key points from images depending on the application. A key-point in an image is one that is distinctive and stable under any lighting condition,
scaling or rotation. Each key point has a numerical representation of any dimension known as the feature vector. The intuition of matching a frame to both the two reference frames is that a frame which is similar to one of the reference frames will have more matching key-points than the other reference frame. For example: A left frame will have more matching points with the left reference than the right reference. In addition we not only want key-points to match but we want to capture the spatial relationship of the key-points during matching. Our method uses SIFT features [5] and 2-nn (2-Nearest Neighbors) matching algorithm to address the matching problem.

As observed in Figure 5.2, the number of matches of a left frame with the Left reference is greater than the number of matches of a right frame with the Right reference. In our method each frame is matched with Left and Right reference and the difference in the number of the matches is used to determine whether a frame is on the Left or the Right side of the court. When the camera or the play is on the sides of the court, the number of matches with one of the reference is greater than the number of matches with the other reference. During transition both the number of matches are small as shown
Figure 5.2: (a) Left Reference frame vs Left Frame. (b) Left Reference Frame vs Right Frame. The reference frame is with yellow borders. The number of matching points on Top is more than Bottom.

in Figure 5.3 where the number of matches with Left and Right references are close to zero.

5.2.1 SIFT Features

1. **SIFT Keypoint**: SIFT stands for Scale Invariant Feature Transform as it transforms information in image into scale-invariant coordinates related to local features [5]. The feature extraction first detects keypoints by using a cascade filtering approach to search for stable features across all possible scales known as scale space. Gaussian function is used as the scale-space kernel. The scale space of an image is defined as a function $L(x, y, \sigma)$ produced by convolution of Gaussian, $G(x, y, \sigma)$,
Figure 5.3: Plot of #matches of each frame with Left and Right references.

with input image $I(x, y)$:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y),$$

where * is the convolution operation in x and y and

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2}e^{-(x^2+y^2)/2\sigma^2}$$

The stable keypoint locations are detected in space scale using space-scale extrema in the difference-of-Gaussian function convolved with the image, $DoG(x, y, \sigma)$, which can efficiently computed by the difference of two nearby scales separated by constant factor k:

$$DoG(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y)$$

$$= L(x, y, k\sigma) - L(x, y, \sigma)$$

The difference-of-Gaussian function provides a close approximation to the scale-normalized Laplacian of Gaussian, $\sigma^2 \Delta^2 G$ where $\sigma^2$ factor is required for true scale invariance. Figure 5.4 shows the approach to construction of $DoG(x, y, \sigma)$. The image is incrementally convolved with Gaussians to produce images separated by a constant factor k in scale space, where scale space is a collection of images.
obtained by progressively smoothing the input image. This process is performed on a number of octaves where each octave smoothing kernel is double the size of the previous octaves and in each octaves on a number of levels which is $k$ factor scale of the previous level.

Figure 5.4: Construction of Difference-of-Gaussian at levels of octaves.

Figure 5.5: Detection of Maxima and Minima of a sample point.

The maxima and minima of the difference-of-Gaussian image are detected by comparing each sample point with its eight neighbors in the current image and the nine neighbors in the scale above and below as in Figure 5.5. The sample point is a keypoint if it is larger or smaller than the 26 neighbors. Keypoints detected are further refined by filtering out those that are unstable either because they are near
image edge or on image structures with poor contrast.

2. **SIFT Descriptors**: SIFT feature is a 128 dimension numerical representation of SIFT keypoint also known as the SIFT descriptor. This descriptor is a 3-D spatial histogram of the image gradients representing the appearance of a keypoint. The gradient at each pixel is regarded as a sample of a 3-D feature vector formed by the pixel location and the gradient orientation using the scale of the keypoint to select the level of Gaussian blur for the image. The keypoint descriptor is shown in Figure 5.6. A Gaussian weighting function with $\sigma$ equal to half of the width of the descriptor window is used to assign weight to each sample point for the purpose of avoiding sudden change in the descriptor with small changes in position of the window and also to give less emphasis to gradient that are far from the center of the descriptor. The orientation of the gradient of the samples of each region in the 4x4 regions are categorized into 8 directional bins. The arrows in the figure on each region corresponds to the direction of gradient and the length of the arrows corresponds magnitude of the histogram. The 4x4 array of histograms with 8 orientations bins in each form the 4x4x8=128 element feature vector for each keypoint.

![Image gradients and Keypoint descriptor](image.png)

**Figure 5.6**: Detection of Maxima and Minima of a sample point.

The feature vector is normalized to unit length to reduce the effect of illumination change. The influence of large gradient magnitudes due to non-linear illumination changes caused by camera saturation of illumination changes on 3D surfaces are reduced by thresholding the values in the unit feature vector to be no larger than 0.2 and then normalizing to unit length. These modifications makes the feature vector to be invariant to illumination, orientation and scaling.
5.2.2 k-Nearest Neighbors

The best candidate match for each keypoint of source image is found by identifying its nearest neighbor in the set of keypoints from the reference image where the nearest neighbor is the keypoint with minimum Euclidean distance for the invariant feature vector. However many features will not have correct match due to noise in the image and it would be useful to discard those features. Comparing the distance of the closest neighbor to that of the second-closest neighbor is found to be an effective measure for correct match. This measure performs well because correct matches need to have the closest neighbor significantly closer than the closest incorrect match to achieve reliable matching. The second-closest match provides an estimate of the density of false matches within the feature space. The ratio of closest neighbor to second-closest neighbor of each keypoint for correct matches is empirically found to be around 0.5 with high probability. It is determined experimentally that the correct match have a probability density function centered at a much lower ratio than that for incorrect matches. For finding the exact nearest neighbors in high dimensions k-d trees are useful data structures but it provides no speedup in computation over exhaustive search. Hence an approximate algorithm called Best-Bin-First (BBF) algorithm [20] is use to find nearest neighbors.

1. **k-d Tree**: A k-d tree is a hierarchical tree structure built by partitioning the data recursively along the dimension with the maximum variance. In each iteration the variance in each dimension is calculated and the data is split into two parts on the dimension with the maximum variance. The mean or median of the values in that dimension can be used as the splitting threshold.

2. **Best-Bin-First (BBF)**: Best-Bin-First (BBF) algorithm is an approximate algorithm that returns the closest neighbor with high probability [20]. The BBF algorithm uses a modified search ordering for the k-d tree algorithm so that bins in feature space are searched in the order of their closest distance from the query location. This search order uses heap-based priority queue for determination of the search order. In the reference matching after checking the first 25% of nearest-neighbor candidates the search is stopped to get an approximate closest neighbor [5]. This heuristic works well because only the matches in which the nearest neighbor is less than 0.8 times the distance of the second-nearest neighbor are considered.
5.3 Segment Detection

The subset of frames from the start frame to the end frame (inclusive) of a ball possession is defined as a segment of the set of the all frames of the video. In a full length video of a basketball game there are a number of segments. Our method detect the segments through post-processing the difference in the number of matches of frames with Left and Right reference frames.

5.3.1 Match Difference

In possession detection each frame is matched with the Left and Right reference frames and the difference between the number of feature matches is the data/signal which is post processed to detect the start and end frames of segments as shown in the flow chart in Figure 5.1. The number of features matches of a left frame with Left reference is greater than the number of feature matches with right reference and vice versa as show in Figure 5.7.

\[ \text{Diff}_{LR} = M_L(t) - M_R(t) \]

![Figure 5.7: Plot of the number of matches with Left and Right references for a sequence of frames.](image)

Let \( M_L(t) \) and \( M_R(t) \) be the number of SIFT feature matches of frame \( t \) with Left Reference and Right Reference respectively. The difference in the number of matches is denoted by \( \text{Diff}_{LR} \).
The corresponding $Diff_{LR}$ of Figure 5.7 is shown in Figure 5.8. In the frames where the transition from one side of the court occurs the value of $Diff_{LR} = 0$ as in the transition frames the matches, which are mostly incorrect matches, are equally low for both Left and Right references. It is determined experimentally that performing matching on just two frames per second i.e., using a frame interval of 15 instead of processing every single frame still yields accurate segment results and decreases the computation time.

![Figure 5.8: Plot of the difference of Left Matches and Right Matches, $Diff_{LR}$, for a sequence of frames.](image)

**5.3.2 Segment Indexing**

1. **Smoothing & Normalization:** As depicted in the flow chart in Figure 5.1, the signal ($Diff_{LR}$) values are first smoothed over an interval of 5 signals preceding it and 5 signals following it to eliminate fluctuation in the number of feature matches which are caused due to sudden camera jerk or panning to capture a player at the corner of the frame. Smoothing provides the effect of an ideal smooth camera panning to the left and right sides of the basketball court.

$$SDiff_{LR}(k) = \frac{1}{(2w + 1)} \sum_{i=k-w}^{k+w} Diff_{LR}(i)$$

where $w$ is the half size of averaging window.

The raw signal and the corresponding smoothed signal after smoothing are shown
Figure 5.9: Raw signal ($\text{Diff}_{LR}$) and smoothed signal ($\text{SDiff}_{LR}$).

Figure 5.10: (a) Smoothed signal ($\text{SDiff}_{LR}$) and (b) locally normalized signal ($n\text{SDiff}_{LR}$).

in Figure 5.9. The Left/Right segments are detected from the smoothed signal by
first performing a local normalization on the detected segments.

2. **Zero-Crossing Detector:** In the smoothed signal \( SDiff_{LR} \), the instances when the signal value change from positive to negative and vice versa corresponds to the frames that captured the middle of the basketball court. In other words it is the point of transition from one side of the court to the other side of the court. The Zero-Crossing Detector algorithm detects the instances when the signal crosses zero value and collects all possible segment candidates. Further the candidates are validated and the signal are normalized based on local extrema to determine the start and end frame of left and right segments in a video of a basketball game. The normalized signal is shown in figure 5.10. The algorithm is described below:

<table>
<thead>
<tr>
<th>Algorithm: Zero-Crossing Segment Detector</th>
</tr>
</thead>
</table>

**Input:** \( M_L(t) \) and \( M_R(t) \) for \( t=1 \) to \( n \) (number of feature matches of each of the \( n \) frames with Left and Right references)

(a) Compute \( Diff_{LR} = \text{Difference}(M_L(t) \text{ and } M_R(t), n) \).

(b) Compute \( SDiff_{LR} = \text{Smooth}(Diff_{LR}) \).

(c) for \( t = 1 \) to \( n \) in \( SDiff_{LR}[1 : n] \)
   
i. Find \( t \) when \( SDiff_{LR}(t) \) changes from +ve to -ve and vice versa.
   
ii. Record start \( t_s \) and end \( t_e \) frame, save segment as candidate.

iii. Record maximum \( d_{max} \) and minimum \( d_{min} \) value of \( SDiff_{LR}(t) \) in the segment.

(d) Validate all segment candidates: Trim candidates with \( |d_{max}| \) and \( |d_{min}| \) below the average \( |d_{avg}| \) of all candidates.

(e) for each segment in candidate segments
   
i. if LEFT segment then normalize \( SDiff_{LR} \) between 0 and 1 wrt \( |d_{max}| \).
   
ii. if RIGHT segment then normalize \( SDiff_{LR} \) between 0 and 1 wrt \( |d_{min}| \).

(f) Set the start and end frame of each segment using a forward threshold \( Th_f \) and backward threshold \( Th_b \) to select the number of frames after and before two transition frames.

**Output:** Start and End frames of all Left/Right segments in the video.
3. **Forward-Backward Threshold:** In a segment the frame with the maximum number of matches is one that is most similar to the reference frame. Since the reference frame is one that best captures the end of the court with the backboard and the rim, high number of matches means that the camera focus is at the end of the court. As mentioned in the previous section the zero-crossings corresponds to the camera in motion during the transition between sides. Hence the frames between two consecutive transition points forms a segment. The start of the segment is few frames ahead of one transition frame and the end is few frames before the next transition frame. We make use of two thresholds as shown in Figure 5.11: Forward threshold $Th_f$ and Backward threshold $Th_b$ to select the start and end of the interval of the segments. The range of both the thresholds are $[0,1]$. In our experiment the value of $Th_f = 0.4$ and $Th_b = 0.3$ is used for all types of games.

Figure 5.11: Forward threshold ($Th_f$) and Backward Threshold ($Th_b$) to select the end and start interval of a segment.

5.4 **Automatic Reference Frame Selector**

The robustness and accuracy of possession detection depends highly on selecting good reference frames that captures most part of the Left and Right side of the basketball court. In this section an approach to automatically select Left and Right reference frames from any type of videos is discussed. We again use the pattern of camera motion which is determined by tracking SIFT features between frames to measure the relative
displacement. There are four possible states of a camera: (a) Zoom-in/Zoom-out (b) Moving Right (c) Still (d) Moving Left as shown in Figure 5.13.

Figure 5.12: Four possible camera states: (a) Zoom, (b) Moving Right, (c) Still, (d) Moving Left.

5.4.1 Camera State Extraction

The state of the camera at each frame is determined from the SIFT based optical flow vectors. Optical Flow vectors are computed by tracking SIFT features between two frames. The frame interval between two frames is set to 60 frames for all types of videos i.e., we are observing the displacement of features in a time span of approximately 2 seconds. We estimate the state of the camera using the median direction ($dir_{med}$) and magnitude ($mag_{med}$) of optical flow vectors. Zoom-in/Zoom-out state of the camera is ignored as its occurrence is relatively infrequent compared to other states of the camera. The camera state, $CameraState(t)$ at frame $t$ is estimated based on the following definition:
\[
\text{CameraState}(t) = \begin{cases} 
\text{STILL}, & \text{mag}_{med}(t) < \text{mag}_{threshold} \\
\text{MOVING RIGHT}, & -\pi/2 < \text{dir}_{med}(t) < \pi/2 \\
\text{MOVING LEFT}, & \pi/2 \leq \text{dir}_{med}(t) < 3\pi/4
\end{cases}
\]

Once the camera state is determined for each frame we make use of the pattern of camera motion as an indicator to select candidates frames for Left and Right references.

5.4.2 Reference Selector

In the automatic reference selector a number of potential candidate frames which corresponds to an acceptable sequence of camera motion pattern are first chosen. From these candidates the best candidate for the Left and Right side of the court are chosen as the references. Let \( M_R \), \( M_L \) and \( M_S \) denote 'Moving Right', 'Moving Left' and 'Still' state of the camera on a frame respectively. The pattern of camera motion that we are interested and will be used for selecting Left and Right reference are of the following form:

\[
\text{Candidate Pattern} = \begin{cases} 
(M_R^+ \rightarrow M_S^i \rightarrow M_L^j) \Rightarrow M_S = \text{Right} \\
(M_L^+ \rightarrow M_S^i \rightarrow M_R^j) \Rightarrow M_S = \text{Left} \\
(M_L \rightarrow M_R^+ \rightarrow M_S^i \rightarrow M_R^j) \Rightarrow M_S = \text{Left} \\
(M_R \rightarrow M_L^+ \rightarrow M_S^i \rightarrow M_L^j) \Rightarrow M_S = \text{Right}
\end{cases}
\]

where \(+\) denotes at least one frame and \(2 \leq i \leq 30, j \geq 2\).

The number of frames with 'Still' camera state is limited so that the reference frames are not selected from frames which have free throws. Free throw frames are not desired reference frames because it contains zoomed-in or zoomed-out view of the court sides. From the four types of valid candidate camera motion patterns defined above, the type of the 'Still' frames can be assigned as Left side or Right side. For instance if the camera pans right and after some 'Still' frames the camera pans left it can be inferred that the camera was focused to the Right side of the court when there was no camera motion and then started to pan left as the possession changed. The details of the algorithm is described below:
**Algorithm: Reference Selector**

**Input:** Video file, n: number of candidate patterns

1. while(candidates ≤ n)
   (a) Compute sparse optical flow between frame(t) and frame(t+60).
   (b) Assign camera state for frame
   (c) Find candidate patterns and add to candidates.
   (d) Assign intervals of $M_S$ frames as Right/Left

2. Extract Representative frames from Left and Right candidates by picking the frame with the highest number of feature matches with other representative frames of the same type. Further the representative

**Output:** Left and Right reference frames.

The algorithm is basically finding a sequence of states that matches a set of regular expressions. From each candidate patterns a representative frame is selected. The representative frame is one that has the maximum number of matches with other frames in its candidate pattern. Further the best representative frame for each side of the court is selected by considering all the representative frames of each candidate patterns. Figure 5.14 shows an example with five candidate patterns and their representative frames.

![Figure 5.13: Representative frames of five automatically selected candidates.](image-url)
5.5 Results

In this section we present the result of our methods for automatic ball possessions detection and the evaluation of the automatic reference frame selector. The results are presented in detail in the tables below. The evaluation of detection of ball possessions is done using precision and recall metrics and for the automatic reference frame selector we used the similarity percentage between manual and automatic reference as the metric.

\[
\text{Precision} = \frac{tp}{tp + fp} \\
\text{Recall} = \frac{tp}{tp + fn}
\]

where \(tp\) is true positive, \(fp\) is false positive and \(fn\) is false negative.

\[
\text{Similarity} = \frac{segs_m \cap segs_a}{segs_m \cup segs_a}
\]

where \(segs_m\) are segments extracted with manual references and \(segs_a\) are segments extracted with automatic references.

5.5.1 Evaluation of Automatic Reference Selector

The results of Automatic Reference Selector is evaluated by comparing the results of Possession detection using manually selected references and automatically selected references. The metric used for evaluation is the similarity (Intersection-over-Union (IOU) overlap percentage) of segments extracted using manual and automatic references. The segments \(segs_a\) and \(segs_m\) correspond to the ball possessions detected using automatic and manual reference respectively. Both the segments generated are very similar as seen in the high overlapping percentage of for different video types as shown in from Table 5.2.

5.5.2 Evaluation of Possession Detection

Our possession detection approach is applied to all types of videos using the automatically extracted reference frames as described in Section 5.1.6. The results are evaluated using
Table 5.1: Result of Reference frames selector: Manual vs Automatic.

<table>
<thead>
<tr>
<th>ID</th>
<th>Video Type</th>
<th>#segments (manual)</th>
<th>#segments (auto)</th>
<th>Similarity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Narrow</td>
<td>130</td>
<td>133</td>
<td>94.08%</td>
</tr>
<tr>
<td>3</td>
<td>Narrow</td>
<td>129</td>
<td>132</td>
<td>92.41%</td>
</tr>
<tr>
<td>6</td>
<td>Unaligned</td>
<td>100</td>
<td>103</td>
<td>91.83%</td>
</tr>
<tr>
<td>8</td>
<td>Unaligned</td>
<td>29</td>
<td>28</td>
<td>93.54%</td>
</tr>
<tr>
<td>9</td>
<td>Wide</td>
<td>205</td>
<td>210</td>
<td>98.01%</td>
</tr>
</tbody>
</table>

Table 5.2: Results of Ball Possession detection.

<table>
<thead>
<tr>
<th>ID</th>
<th>#segments(GT)</th>
<th>#segments</th>
<th>TP</th>
<th>FP</th>
<th>FN</th>
<th>Precision</th>
<th>Recall</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>124</td>
<td>124</td>
<td>119</td>
<td>5</td>
<td>5</td>
<td>0.959</td>
<td>0.959</td>
<td>Narrow</td>
</tr>
<tr>
<td>3</td>
<td>121</td>
<td>125</td>
<td>118</td>
<td>7</td>
<td>3</td>
<td>0.944</td>
<td>0.975</td>
<td>Narrow</td>
</tr>
<tr>
<td>6</td>
<td>115</td>
<td>109</td>
<td>106</td>
<td>3</td>
<td>9</td>
<td>0.972</td>
<td>0.929</td>
<td>Unaligned</td>
</tr>
<tr>
<td>8</td>
<td>27</td>
<td>29</td>
<td>26</td>
<td>3</td>
<td>1</td>
<td>0.896</td>
<td>0.962</td>
<td>Unaligned</td>
</tr>
<tr>
<td>9</td>
<td>217</td>
<td>219</td>
<td>214</td>
<td>5</td>
<td>3</td>
<td>0.977</td>
<td>0.986</td>
<td>Wide</td>
</tr>
<tr>
<td>10</td>
<td>176</td>
<td>179</td>
<td>175</td>
<td>4</td>
<td>1</td>
<td>0.972</td>
<td>0.994</td>
<td>Standard</td>
</tr>
<tr>
<td>11</td>
<td>181</td>
<td>182</td>
<td>176</td>
<td>6</td>
<td>5</td>
<td>0.967</td>
<td>0.972</td>
<td>Wide</td>
</tr>
<tr>
<td>12</td>
<td>156</td>
<td>150</td>
<td>149</td>
<td>1</td>
<td>7</td>
<td>0.993</td>
<td>0.955</td>
<td>Standard</td>
</tr>
<tr>
<td>13</td>
<td>187</td>
<td>193</td>
<td>170</td>
<td>23</td>
<td>17</td>
<td>0.880</td>
<td>0.909</td>
<td>Narrow</td>
</tr>
</tbody>
</table>

5.6 Discussions

In the above sections we presented the experimental results of our algorithms to address the problem of detecting ball possessions. The algorithms successfully detected ball possessions of the teams through matching of frames with reference frames. As mentioned in Chapter 2, to the best of our knowledge segmenting and indexing the full length raw videos of basketball games has not been addressed in any previous research. For this reason our method cannot be evaluated through comparison with the results of previous research results and hence its proof of effectiveness is based only on its accuracy compared
with the ground truth. A similar indexing problem to detect zoom, fade, replay, etc in professional basketball (NBA) videos was addressed using HMM (Hidden Markov Models) [14, 15]. There are other methods that are focused on region detection using dominant color [18] and direct court detection [19] which can be employed to detect the sides of the court. However these methods requires fine tuning for different types of videos and learning the parameters of HMM from some video type does not generalize enough for accurate result. PageRank algorithm [21] combined with robust color histogram descriptor for segments [17] could be used for automatic reference frame selection but due to symmetry of the court the result of the algorithm is not reliable. We want a generalized solution that works for any video types and our method provides a robust and accurate solution as observed from the result tables and is also computationally efficient because at most four frames are processed be second to compute the matching difference.

5.6.1 Alternate Method

5.6.1.1 Segmentation with k-means Clustering

In this section we describe an unsupervised approach to detect three distinct regions of the basketball court using k-means clustering [9]. For this we incorporate an additional reference frame: the mid-court reference frame. In this approach each frame is matched with three reference frames: Left, Mid and Right and the number of matches with each reference frames forms the three signals as shown in Figure 5.12. It is clear from the plot that when the frame is a mid frame the number of matches of the frame with the mid reference is relatively higher than matches with either left or right references. For non-play frames such as frames which are on commentators, spectator or the coach and the bench the value of all the three signals are low. We normalize each signal and vectorize into a 3-dimension vector and cluster into three clusters using k-means. This method labels the Left, Right and Mid segments fairly well however the start and end
of a segment is not well defined and inconsistent for different types of videos.

\[
S_0(t) = \frac{M_L}{M_L + M_R + M_M} \\
S_1(t) = \frac{M_R}{M_L + M_R + M_M} \\
S_2(t) = \frac{M_M}{M_L + M_R + M_M}
\]

where \(M_L, M_R, M_M\) are the number of matches with Left, Right and Mid references respectively and \(S_i\) is the \(i^{th}\) feature of the feature vector \(S\) for the sample at frame \(t\).

![Figure 5.14: The three signals (red, blue, yellow) represents the number of matches of each frame with Left, Right and Mid reference frames. Out-of-play frames are characterized by low values for all three signals.](image)

A small experiment to compare the result of this method with our original proposed method shows that the two methods have quite similar results. In table 5.3, we show the percentage of overlap between the results.

<table>
<thead>
<tr>
<th>Compare</th>
<th>#LEFT</th>
<th>#RIGHT</th>
<th>MID</th>
<th>#Total Frames</th>
<th># Overlap</th>
</tr>
</thead>
<tbody>
<tr>
<td>k-mean vs proposed</td>
<td>2634</td>
<td>2785</td>
<td>612</td>
<td>6367</td>
<td>94.72%</td>
</tr>
</tbody>
</table>

Table 5.3: Result of comparison of k-mean vs proposed method.
Chapter 6: Shots Detection

In this chapter we will discuss our method of detecting shots in basketball videos in detail. Since we are dealing in videos with high variations as describe in Chapter 3 it is a very challenging task to formulate a generalized approach to detect shots. In addition using multi-player tracker and tracking the ball is hard because of non-rigid human structure, camera motion, lighting conditions and occlusion. It is also very computationally expensive for the purpose of shot detection. Our method uses the fact that in all shots the ball moves towards the backboard and backboard is a rigid and very stable object model for tracking. Thus our method is based on tracking the backboard and monitoring it to detect ball entry which are qualified as shots.

6.1 Overview of Proposed Method

The main idea of our method is to track the backboard and detect the ball entry in the region of the backboard. It is relatively easy to track backboards because the both the backboards on the court have a simple and rigid appearance models. The tracker does not necessarily have to be online as the appearance of the target will not change drastically although there may be change in illumination due to flashes from the audiences’ cameras. A desirable tracker is one that is online and real-time as the backboards on both sides will have to be tracked on a large percentage of frames of a given video. The overview of our Shot Detection method is as shown in the flow chart in Figure 6.1. We use the information of ball possession from the earlier method described in Chapter 5 to index the start and end frame of both the Left and Right segments of a given video. We use two trackers one for each backboard on the left and right side of the court. The trackers are first initialized manually to the targets i.e., the left and right backboard. The tracker in our method is a kernalized structured output support vector machine (SVM) [24, 23] which is learned online and provide adaptive tracking. The features for the target are Haar-like features. The region tracked with high confidence by the trackers are clipped and post processed for detection of ball entries. The frames on which the ball entries
are detected on the backboard corresponds to the detection of shot events in a game. In the next sections the various components of our shot detection method are explained in detail.

Figure 6.1: Flow chart of Shots Detection.

6.2 Tracking

Object tracking is an important computer vision problem that many research groups have been improving and advancing. Tracking remains a challenging problem due to appearance change caused by non-rigidity of target object, illumination, motion, occlusion, etc. An effective appearance model is the key to the success of tracking algorithms. Most tracking algorithm often update models with samples from observations from recent frames to adapt to the changes in model in following frames. Learning a robust
model depends on the sufficiency of proper training data otherwise it encounters the drift problems. An approach of tracking namely tracking-by-detection has become very popular recently. This paradigm treats the tracking problem as a detection task applied over time [23]. The model, basically a classifier, is learned using positive and negative samples around the target area and the region with the maximum classifier confidence corresponds to the best estimate of object location. There are a number of state-of-the-art real-time trackers that performs well for different purposes [22]. For the purpose of backboard tracking we evaluated the real-time Compressive Tracker [25] and Stuck Tracker [24] and chose the latter in our method of shot detection.

6.2.1 Structured Output (Struck) Tracker

Struck tracker is a support vector machine (SVM) based online tracker that runs in real-time. This tracker performs better than other trackers and more robust for backboard tracking on various video types. The approach in Struck Tracker treats the tracking as a structured output prediction problem in which the task is to predict the change in target location in the next frames. The framework of this tracker integrates the learning and tracking by making use of SVMs due to its good generalization ability and robustness to noise and object representation through the use of kernels [24]. This tracker does not require any offline labeled data. During the online learning the framework controls the sample selection and the relationships between samples such as relative similarity are taken into account. In addition this tracker uses a budgeting approach to constrain the number of support vectors and gain computational efficiency. The framework of Struck tracker is explained further in detail in the following sections.

1. Structured Output SVM

The framework of this tracker learns a prediction function \( f : X \rightarrow Y \) to directly estimate the object transformation between frames instead of learning a classifier. Thus the output space is the space of all transformations \( Y \) rather than binary \( \pm 1 \). The function \( f \) is learned in a structured output SVM framework which introduces the discriminant function \( F : X \times Y \rightarrow \mathbb{R} \) used for prediction according to

\[
y_t = f(x_t|^{p_t-1}) = \arg \max_{y \in Y} F(x_t|^{p_t-1}, y). \tag{6.1}
\]
where \( x_{t}^{p_{t-1}} \) is the set of features extracted at \( t \)th frame from within the bounding box positioned at \( p \) in \((t-1)\)th frame, \( y \) is the desired transformation and \( y_{t} \) is the transformation of the object in \( t \)th frame. Using the discriminant function \( F \) the label \( y \) is incorporated into the learning algorithm and to update the prediction function a new labeled pair \((x_{t}^{p_{t}}, y^{0})\) relative to the new tracker location is supplied.

The function \( F \) measures the compatibility between \((x, y)\) and gives a high score for well matched pairs. \( F \) is restricted to be of the form \( F(x, y) = \langle w, \Phi(x, y) \rangle \) where \( \Phi(x, y) \) is a joint kernel map and \( F \) is be learned in a large-margin framework from a set of example pairs \((x_{1}, y_{1}), ..., (x_{n}, y_{n})\) by minimizing the following convex objective function:

\[
\min_{w} \frac{1}{2} ||w||^{2} + C \sum_{i=1}^{n} \xi_{i} \\
\text{s.t} \quad \forall i : \xi_{i} \geq 0 \\
\forall i, \forall y \neq y_{i} : \langle w, \delta \Phi_{i}(y) \rangle \geq \Delta(y_{i}, y) - \xi_{i}
\]  

\((6.2)\)

where \( \delta \Phi_{i}(y) = \Phi(x_{i}, y_{i}) - \Phi(x_{i}, y) \) and \( \Delta \) is a loss function that is defined as in Eqn. 6.3 such that \( \Delta(y, \overline{y}) = 0 \) iff \( \overline{y} = y \) and \( s^{p_{t}}_{p_{t}}(y, \overline{y}) \) is the intersection-over-union overlap percentage.

\[
\Delta(y, \overline{y}) = 1 - s^{p_{t}}_{p_{t}}(y, \overline{y})
\]  

\((6.3)\)

2. **Online Optimization**

In online setting Eqn. 6.2 is optimized using standard Lagrangian duality techniques. The dual form is as below:

\[
\max_{\alpha} \sum_{i,y \neq y_{i}} \Delta(y, y_{i}) \alpha_{i}^{y} - \frac{1}{2} \sum_{i,y \neq y_{i}} \sum_{j,y \neq y_{j}} \alpha_{i}^{y} \alpha_{j}^{\overline{y}} \langle \delta \Phi_{i}(y), \delta \Phi_{j}(\overline{y}) \rangle \\
\text{s.t} \quad \forall i, \forall y \neq y_{i} : \alpha_{i}^{y} \geq 0 \\
\forall i : \sum_{y \neq y_{i}} \alpha_{i}^{y} \leq C
\]  

\((6.4)\)
The discriminant function is expressed as $F(x, y) = \sum_{i \neq y} \alpha_i \langle \delta \Phi_i(y), \Phi(x, y) \rangle$.

The dual in Eqn. 6.4 is simplified by re-parameterizing and solved using the SMO (Sequential Minimal Optimization) algorithm as described in [24].

3. Budget

The support vectors increases overtime and evaluating $F(x, y)$ requires evaluating the inner products (or kernel function) between $(x, y)$ and each support vector which means both space and time complexity grows linearly. Hence the framework incorporates a fix budget of support vectors in order to keep the tracker online and real-time. Each time the budget is exceeded the support vector which results in the smallest change in the weight vector $w$ is removed. This budgeting of support vectors improves the efficiency without significant impact on the performance.

6.2.2 Haar-like Features

Haar-like features are features obtained from integral image representation of an image [26]. These features are reminiscent to Haar basis functions and more specifically the composed of three types of rectangle features: two-rectangle, three-rectangle and four-rectangle features as shown in Figure 6.2. The value of two-rectangle feature is the difference between the sum of the pixels within the rectangular region, the value of three-rectangle feature is the difference between the sum of pixels in the two outside region and the sum of pixels in the center region and the value of four-rectangle is the difference between the sum of pixels between the diagonally aligned pairs of rectangles.

![Figure 6.2: Four types of rectangle features within the target.](image-url)
angle features can be computed very efficiently using the integral image representation. The integral image at location \((x, y)\) inclusive is:

\[
        ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y')
\]  

where \(ii(x, y)\) is the integral image and \(i(x, y)\) is the original image. The integral image is computed efficiently using Dynamic Programming whose recurrence are as follows:

\[
        s(x, y) = s(x, y - 1) + i(x, y) \\
        ii(x, y) = ii(x - 1, y) + s(x, y)
\]

where \(s(x, y)\) is the cumulative sum along the row and \(s(x, -1) = 0\) and \(ii(-1, 0) = 0\). Using the integral image the rectangle features are computed efficiently as in Figure 6.3 using at least four array references. Rectangle features are sensitive in capturing the presence of edges, bars and other simple simple structures. It provides a rich image representation of rigid targets which suits very well to represent backboard target for tracking.

Figure 6.3: The value of the integral image at 1 is the sum of all pixels in rectangle A, the value at location 2 is A+B, at location 3 is A+C and at location 4 is A+B+C+D.

6.2.3 Backboard Tracking

The output of the Ball Possession detection provides the start and end frame index of each segment. When the ball is on one end of the court there is the possibility that a shot is attempted by the team with ball to put the ball into the basket. There can be
instances when a team makes multiple attempts or shots in one possession and in the case of a free throw there can be additionally at least one shot or at most three shots. Hence to detect shots the backboard must be tracked from the start frame to the end frame of each segment. The Haar-like features, as described in the section above, of 192 dimensions and Gaussian kernel are used for representing the target object. Since the backboards have distinct edges and simple vertical and horizontal lines these features are good representation of the backboard. Two important components of tracking is initializing the tracker to the target and re-initializing the tracker periodically or when the tracker goes astray. In the following section we discuss how we initialize and re-initialize the tracker on the backboard.

1. Initialization

As shown in the flow chart in Figure 6.1, we use two trackers: one for the left backboard and one for the right backboard which are initialized manually at the start from the user interface. The trackers are initialized to the target object (backboards) on the reference frames which are extracted automatically as explained in Section 5.1.6. Using the information from the manually initialized tracker, the tracker can be initialized automatically in the rest of the segments.

Figure 6.4: Manual initialization of tracker slightly over and beyond the backboard: (a) On the Left backboard include extra region on the right, (b) on the Right backboard include extra region on the left as shown with the yellow arrows.
(a) **Setting the Target:** The goal is to detect every shot that comes to the vicinity of the backboard so the target for the tracker must enclose the entire backboard. However there are some valid shots that do not really enter the backboard region such as shots that hits the rim and bounce out. For such cases the target must include some extra region beyond the backboard region as shown in Figure 6.4 in order to detect such shots. From our experiments we found out that additional region with width equal to the size of the diameter of the rim is adequate to detect shots that hit the rim.

(b) **Auto Initialization:** In each segment the tracker is initialized automatically by matching the start frame with the correct reference frame and calculating the perspective transformation of reference frame to the start frame. This transformation is then applied to the original tracker box coordinates on the reference frame to obtain the tracker box coordinates on the start frame of the segment.

\[
T_t = f_m(F_r, F_t) \\
P_t = P_r \circ T_t
\]

where \(f_m\) is a function that matches the reference frame \(F_r\) with \(t\)th frame \(F_t\) and computes the transformation \(T_f\). The location of the tracker box is the composition of the location of the box in reference frame \(P_r\) and \(T_t\). For matching we used OpenCV’s SURF features [27] and interface to FLANN (Fast Library for Approximate Nearest Neighbors) library [29] for Flann-based-matcher. [28]. When the tracker box coordinates fall outside the region of the frame i.e., the backboard is not fully in the frame, the initialization method will move on to the next frames until it finds a frame where the box coordinates are within the boundaries of the frame.

2. **Re-initialization**

Stuck tracker can successfully track the backboards in the segments of different video types. However there are instances when the camera moves away and the backboard is no longer in the frame or only a portion of the backboard is in the frame where as the rest falls outside the boundaries of the frame. In such cases it is important to stop tracking and find another subsequent frame to initialize the tracker using Eqn. 6.5. Determining when to stop tracking so that incorrect
samples are not used in updating the object model is critical for online tracking. In our method, the radial samples of windows/boxes around the last position of the tracker box are used to estimate the transformation of the object model in the current frame. The tracker is re-initialized when the best score among the samples has a confidence score $\leq 50\%$. This threshold on the confidence score works well and triggers re-initialization when the bounding box on the target is outside the boundary of the frame and when the backboard is hazy due to camera motion. It is not necessary to track backboard when there is high camera motion or when only a portion of the backboard is within the frame because in most cases when shots are taken the camera is still and focused to the backboard.

Upon successful tracking of the backboard in each frame, an image of the size of the bounding box of the tracker is clipped from the frame and stored in a temporary repository for further post-processing to detect the events of ball entry into the backboard.

6.3 Ball Entry Detection

Ideally when there is no camera motion the image clips of the tracked backboard from a segment is composed of stable images of the backboard which may differ from each other only in illumination. From these frames the goal is to detect the event when the ball enters the backboard region. There can be more than one entry of the ball into the backboard resulting from multiple attempts and free throws. Hence it is important to distinguish between two entry events of the ball in the segment so that the two are not considered as one ball entry event. Our method utilizes the fact that when the ball enters the backboard there is motion in certain regions of the backboard image caused by the movement of the ball. In order to localize the ball using the motion information we extract the dense optical flow from consecutive backboard clips and overlay a grid of overlapping cells to pinpoint the cell with the ball as described in the following sections.

6.3.1 Dense Optical Flow

Dense Optical flow is a technique to estimate the motion or displacement of each pixel between two frames [30]. The estimation is done spatiotemporally using polynomial expansion to approximate some neighborhood of each pixel with a quadratic polynomial.
The output of dense optical flow is the approximate displacement and direction of the motion of each pixel from frame $t$ to frame $(t+1)$ which are also known as the optical flow vectors. This matrix with the optical flow vectors will be known as the optical flow image $I_{of}$. Figure 6.5 is a visualization of the optical flow vectors on a sequence of frames.

Figure 6.5: Visualization of optical flow vectors on a sequence of backboard frames when ball enters the backboard.

### 6.3.2 Grid Optical Flow

In order to localize the ball in the optical flow image we divide the image into $8 \times 8$ grid of overlapping cells of width $w_c$ and height $h_c$. The size of the cells are some percentage of the size of the image (tracker box). Since we want a cell to completely enclose the ball, if present, and to localize it we found by experiment that Eqn. 6.6 is an appropriate cell size.

$$w_c = w_i \times 0.20$$
$$h_c = h_i \times 0.25$$

(6.7)

where $w_i$ and $h_i$ are the width and the height of $I_{of}$. Each cell has a magnitude denoted by $Cell_{(x,y)}$, which is the sum of the magnitude of all optical flow vectors within the cell. The sum of the magnitude of optical flow vectors within each cell are computed efficiently using an integral image representation, $II_{of}$, of the optical flow image, $I_{of}$ similar to the Eqn. 6.5.

$$II_{of}(x,y) = \sum_{x' \leq x, y' \leq y} I_{of}(x',y')$$

$$Cell_{(x,y)} = II_{of}(x,y)$$

### 6.3.3 Active Cells

An active cell, $Cell_{act}(t)$, of frame $t$ is the cell with the maximum magnitude. During a ball entry the region of the backboard where the ball is in motion will have the maximum
optical flow. The cell with the maximum magnitude corresponds to the cell with the ball in an ideal case as seen in Figure 6.7 where the active cells of the sequence of frames are exactly on the ball. In other words the active cells track the ball within the backboard region. Furthermore the active cells during ball entry has the property of being adjacent to each other and overlap the subsequent active cell by some degree.

$$Cell_{act}(t) = \max_{\forall x, \forall y} Cell_{(x,y)}(t)$$

Figure 6.7: Active cell (green box) of a sequence of frame of the backboard with the ball.

Figure 6.8: Active cell (green box) of a sequence of frames of backboard without the ball.

When there is no ball in the backboard region the active cells in a sequence of backboard frames are scattered all over the region of the backboard as shown in Figure 6.8. However the active cell does not always track the ball as shown in Figure 6.8. In the middle frame when the ball ascended to the maximum height and then began to descend the ball is relatively motionless for few frames and the active cells are not tracking the ball. Another case in the same sequence when the active cell failed to track the ball is
at the end when the net is disturbed by players hand. The magnitude of the cell in the net region became the active cell instead of the cell over the ball.

Figure 6.9: Failure case of active cells not tracking the ball in some frames.

It is observed that that one property that differentiates a sequence of active cells without the ball and with the ball is the magnitude of the active cells. The active cells with the ball has naturally higher magnitude but as seen in Figure 6.9 the magnitude of the active cells are not consistently high through out the sequence. Another property is the overlapping active cells in consecutive frames in the presence of the ball on the backboard region.

6.3.4 Noise Elimination

The dataset consists of only unprofessional videos which lacks smooth panning and cause the frames to appear blurry and increase the magnitude of the active cell. The noise in the magnitude of the active cells can be due to other factors such as:

1. Blurry frames of the backboard due to sudden camera movement.

2. Jittery backboard clips caused by the tracker’s prediction of the location of the box in the next frame.

3. Towards the end of each segment, when the camera starts to move to the other side of the court the camera motion adds blurriness to the backboard image.

These three factors increase the magnitude of the active cells by the addition of noise. To eliminate the noise we update the magnitude of an active cell by taking the median
magnitude of all cells and subtracting the median from the active cell’s magnitude.

\[ Cell_{act}(t) = Cell_{act}(t) - \text{median}_{x,y} Cell_{(x,y)}(t) \]

The intuition is that when there is camera motion all the cells will have equally high magnitude hence subtracting the median from the active cell’s magnitude will result in a low magnitude of the active cell. The active cell’s magnitude is much larger than all other cells in case of a valid ball entry. Figure 6.10 shows the elimination of noise by this method.

Figure 6.10: (a) Magnitude of active cells with noise (b) Magnitude of active cells after noise cancellation.
6.3.5 Adaptive Threshold

Our method of detecting the event of ball entry into the backboard utilizes the two properties mentioned in the last section, namely, (a) Adjacent/Overlapping Active cells and Active Cells with high magnitude. Since we want the shot detection to be robust to wide variety of videos it is necessary that we have an adaptive thresholding technique to discriminate between sequence of active cells with the ball and without the ball to detect the event of ball entry into the backboard which corresponds to detection of the shots taken by the players.

1. **Adjacency Heuristic**

In this naive method an adaptive threshold $th_f$ is selected as defined below. In a sequence of active cells if there is a subset of overlapping consecutive active cells, $S$, with magnitude $\geq th_f$ and $|S| \geq m$ where $m$ is another threshold which is the expected size of a subset that corresponds to a valid ball entry. The value of $m = 5$ is used in the experiment.

$$th_f = \frac{1}{k} \sum_{i=0}^{k} SortedCell_{act}(k)$$

where $SortedCell_{act}$ is the list of active cells $Cell_{act}$ sorted in descending order and $k = size(SortedCell) \times 0.15$ i.e., the top 15% of active cells.

$$BallEntry(S) = \begin{cases} True, & \forall t \in S : Cell_{act}(t).magnitude \geq th_f, |S| \geq m \\ False, & otherwise \end{cases}$$

2. **2-Cluster Heuristic**

In the adaptive method a semi-supervised approach is applied to cluster the subsets of frames into two categories: Shot and Non-Shot. The sequence of active cells are divided into samples of size $w$ where samples $s_i$ and $s_{i+1}$ have half of their active cells in common. Each sample or subset of frames is represented by two features:
(a) Sum of the magnitude of active cells in the subset.

\[ s_i[0] = \sum_{k=i-30}^{i} Cell_{act}(k) \]

(b) Sum of temporal gradient of active cells in the subset.

\[ s_i[1] = \sum_{k=i-30}^{i} |Cell_{act}(k + 1) - Cell_{act}(k - 1)| \]

The intuition being that a subset of frames which has the ball (i.e., a valid shot) will have high aggregate magnitude of active cells and since the ball is moving it will have a high aggregate gradient as compared to the subset of frames which do not have the ball. The samples are clustered using k-means cluster into two clusters and the following criteria determines if a there are valid shots in the sequence. The criteria for a cluster to be valid shot cluster is as follows:

\[ \frac{\max_{mag}}{\forall s.magnitude} \]

\[ diff_{AB} = \frac{(|mean_A.magnitude - mean_B.magnitude|)}{max_{mag}} \]

\[ \text{ShotCluster} = \begin{cases} 
Cluster_A, & \text{if } size_B \geq 2 \times size_A, diff_{AB} \geq 0.5 \\
Cluster_B, & \text{if } size_A \geq 2 \times size_B, diff_{AB} \geq 0.5 \\
\text{None,} & \text{otherwise}
\end{cases} \]

It is semi-supervised because we are setting that the 'Shot' cluster and 'Non-Shot' cluster sizes are twice the size of each other and the absolute difference of the magnitude of the mean of the clusters are half the maximum magnitude of a sample. If a valid 'Shot' cluster exists then the samples within that cluster if they are not overlapping are the detected shots.

6.4 Results

In this section we show the details of the result of our Shot Detection method. The ground truth of the tracker are obtained by matching each frame with the reference
to calculate the transformation and applying that transformation to the tracker box position to get the new position as mentioned in section 6.2.3.1. The ground truth for shots are manually annotated.

6.4.1 Evaluation of Tracker

The backboard is a very stable target model for tracking hence the tracker performance and accuracy is high. The tracker with and without re-initialization are compared on few selected segments with high variations from a video. The metric used for evaluation is the percentage of overlap (intersection-over-union) of the tracker box with the ground truth box. It is observed that the tracker’s accuracy increased further with re-initialization. The tracker is reinitialized when the confidence score of the tracker is $\leq 0.5$. Figure 6.11 shows the result of the tracker. With re-initialization the tracker in all the frames of the segments have an overlap $\geq 70\%$ with the ground truth of the backboard tracker.

6.4.2 Evaluation of Ball Entry Detection

The annotation of the ground truth for shot detection is done manually. For some shots in some of the games the backboard was partially out of the frame. Although such shots are marked as valid shots in the ground truth, in the shot detection method such shots could not be detected as our tracker was not designed to track partially visible backboards.

1. 2-Cluster Heuristic Result:

The 2-Cluster Heuristic threshold result table in 6.2 shows that this adaptive threshold performed fairly well for some video types. However it is not robust enough to detect shots in all video types. The threshold performed effectively for Narrow-angle videos and Standard-angle videos with smooth panning. For other types of videos a large number of shots could not be detected. This method is based on the assumption that the smaller cluster is the 'Shot’ cluster which might be a wrong assumption because in the case of a shot from a fast break play the clusters might be of the same size or it may be the case that larger cluster is the ‘Shot’ cluster.
Figure 6.11: (a) Results of tracker (a) without re-initialization (b) with re-initialization.

2. Adjacency Heuristic Result:

This adaptive threshold technique is experimented only on three videos. As shown in table 6.1, for Narrow-angle video that takes a close-up shot of the game the method worked fairly well however as the camera angle became wider the accuracy decreased. In the standard-angle video a large number of shots could not be detected in this threshold. For this reason the method was not experimented on videos shot with wider camera angles.
<table>
<thead>
<tr>
<th>ID</th>
<th>GT</th>
<th>#Shots(GT)</th>
<th>#Shots</th>
<th>TP</th>
<th>FP</th>
<th>FN</th>
<th>Precision</th>
<th>Recall</th>
<th>#Out</th>
<th>Type</th>
<th>Time(mins)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>141</td>
<td>130</td>
<td>123</td>
<td>7</td>
<td>18</td>
<td>18</td>
<td>0.95</td>
<td>0.87</td>
<td>5</td>
<td>Narrow</td>
<td>~65</td>
</tr>
<tr>
<td>10</td>
<td>191</td>
<td>121</td>
<td>106</td>
<td>15</td>
<td>70</td>
<td>70</td>
<td>0.88</td>
<td>0.55</td>
<td>2</td>
<td>Standard</td>
<td>~117</td>
</tr>
<tr>
<td>12</td>
<td>155</td>
<td>111</td>
<td>108</td>
<td>3</td>
<td>47</td>
<td>47</td>
<td>0.97</td>
<td>0.70</td>
<td>3</td>
<td>Standard</td>
<td>~77</td>
</tr>
</tbody>
</table>

Table 6.2: Results of Shot Detection with Adjacency Heuristic threshold.

6.5 Discussions

In this chapter we presented the proposed method for Shots Detection and the experimental results to show the effectiveness of the method. There are previous work on detection of ball and shots based on estimation of the trajectory of the ball [31] however their approach has been on just over few hundred frames of one particular video type. Other work also involves tracking of the basketball and detection of the hoop to identify shots using DBN (Dynamic Bayesian Network) [32]. Huang et.al [33] tracks the ball, players and the basket and identify shots using BBN (Bayesian Belief Network). Tien et.al [34] tracks the ball and estimate the 3D position of the ball trajectory and the hoop to detect shots. However these approaches were experimented only on very few videos. The variation of the videos in our dataset demands a robust solution to the shots detection problem.

Our method is novel in the sense that instead of tracking the ball to detect a shot we track the backboard and detect the events of ball entry for shots detection. As mentioned in the previous section the shots detection consists of two parts: Backboard tracking and
ball entry detection. Due to the rigid nature of the target, the backboard is successfully tracked for all video types. However the ball entry detection is successful only on certain video type and it indicates that the robustness of ball entry detection depends highly on the adaptive threshold. Future improvements can be made on the adaptive threshold method to increase the accuracy and robustness of our shots detection method.

6.5.1 Alternate Method:

6.5.1.1 2-Cluster + Adjacency Heuristic

A possible method to improve the robustness of shots detection is using a modification of both the 2-Cluster and Adjacency heuristic to choose the adaptive threshold. The criteria for choosing the 'Shot' cluster being the cluster with higher norm of the mean of the clusters. From the 'Shot' cluster a finer search for samples with valid shots can be performed by using the adjacency heuristic in other words the if a sample in the 'Shot' cluster has adjacent active cells then a valid shot is present in that sample. This heuristic is based on the observation that since the cluster with valid shot samples cannot have a lower magnitude of active cells and gradient than a cluster without valid shot samples, the presence of adjacent active cells in a sample of the 'Shot' cluster will validate that the sample to be a valid shot sample.
Chapter 7: Conclusion

Computer vision systems in sports is revolutionizing the way people watch sports. From just being a means of entertainment in the past, sports has now grown into a business in itself which not only impacts sportsperson but is highly influential in the socio-economic fabric of a modern society. Today sports statistics is driving sports and its business. An additional information gained about the team or the opponent can be an indispensable arsenal in improving the overall performance of the team. Here is where sports depend on technology to automatically extract game statistics and other useful information. In this thesis we attempted to address the problems of ball possessions and shots detection in basketball videos from unprofessional teams whose solutions have high commercial values to sport analytics industry and also open doors to more advanced level of game analysis. We have shown that our proposed method for possession detection is robust, computationally efficient and accurate. The shots detection method is effective only on few video types and has potential to be improved for robustness and higher accuracy.

7.1 Challenges

The solution for ball possessions detection proposed in this thesis is very accurate as verified from the experimental results and it can be considered as a proven method. A robust solution for shots detection still remains a challenge. We have addressed successfully a part of this problem by tracking the backboard for all types of video with high performance and accuracy but the challenge of ball entry detection still needs to be addressed. This problem can be considered as an anomaly detection problem. The normal condition or model is when there is no ball in the vicinity of the backboard. The entry of the ball into the backboard region is an anomaly from the normal condition. This challenge is compounded due to the presence of huge variation in video types taken by semi-professional camera operators.
7.2 Future Work

The application of computer vision technologies in sports has a huge scope as it is still in its early state. Based on the solutions proposed in this thesis a number of research problems can be formulated specially in the sports of basketball to further apply the use of technology in viewing basketball as well as extracting game statistics, patterns and strategies.

7.2.1 Improvement of Shots Detection

The first possible research direction is in the improvement of Shots Detection particularly the detection of ball entry event on the backboard. In our method the active cells of each frame could track the ball relatively well when the ball is in motion. This method used just the magnitude information of the optical flow vectors within the cell to qualify a cell as the active cell. It is evident from the visualization in Figure 6.5 that the direction of the optical flow vectors within a cell can also give some useful indication on the direction the moving ball and also the entry of the ball. In addition our method chose only one active cell however choosing the top K cells as active cells from a frame may possibly add more dimension to the data make inference. The limitations in our approach is possibly the opposite to the 'curse of dimensionality' in that the information to distinguish a ball entry from noise was not enough.

7.2.2 Shots Identification

Another extension of research based on this thesis is the problem of identification of shots. We have already demonstrated that tracking the backboard can be done successfully for all video types. By initiating a tracker for the ball or using a smarter active cell tracker, the ball can be tracked in the backboard. With the trajectory of the ball and the information of the location of the basket, shots made and shots missed can be determined. Shot type such as long or short shot can also be predicted from the ball trajectory.
Bibliography


[31] Bodhisattwa Chakraborty and Sukadve Meher. *A Trajectory-Based Ball Detection and Tracking System with Applications to Shooting Angle and Velocity Estimation in Basketball Videos*.


