Visualization Methods for Sports Data Collection and Analysis

DISSERTATION

Submitted in Partial Fulfillment of

the Requirements for

the Degree of

DOCTOR OF PHILOSOPHY (Computer Science)

at the

NEW YORK UNIVERSITY
TANDON SCHOOL OF ENGINEERING

by

Jorge Henrique Piazentin Ono

September 2021

Visualization Methods for Sports Data Collection and Analysis

DISSERTATION

Submitted in Partial Fulfillment of

the Requirements for

the Degree of

DOCTOR OF PHILOSOPHY (Computer Science)

at the

NEW YORK UNIVERSITY
TANDON SCHOOL OF ENGINEERING

by

Jorge Henrique Piazentin Ono

September 2021

Approved:

_____

Department Chair Signature

_____

Date

University ID: N13689494
Net ID: jpo286

Approved by the Guidance Committee:

Major: Computer Science

Cláudio T. Silva
Professor of Computer Science
New York University

Date

Enrico Bertini
Associate Professor of Computer Science
New York University

Date

Jean-Daniel Fekete
Professor of Computer Science
Université Paris-Saclay, CNRS, Inria & LISN

Date

Juliana Freire
Professor of Computer Science
New York University

Date

Vita

Jorge Henrique Piazentin Ono was born in the city of Bauru, Brazil, in September 1991. He has a B.Sc. in Computer Science from the State University of Sao Paulo (2012) and an M.Sc. in Computer Science and Computational Mathematics from the University of Sao Paulo (2015). While completing his master studies, he worked as a researcher at the VICG lab, where he developed visualization tools for music information retrieval and cover song identification. He started his Ph.D. in September 2015, working in various areas, including interactive data labeling, explainable machine learning, and sports analytics. During his Ph.D., he was a visiting researcher at NYU Paris and an intern at AT&T and Facebook. He has received several awards, including best paper honorable mention awards from Eurovis (2018) and ACM CHI (2019), the NYU Provost's Global Research Initiatives Fellowship (2018), and the NYU Tandon Pearl Brownstein Doctoral Research Award (2021).

Acknowledgements

I am grateful to my mother and family members, who have always believed in me and were always rooting for my success.

I would like to thank my advisor, Cláudio Silva, for the support, guidance, and exchange of ideas throughout my Ph.D. studies. Thank you for allowing me to work on challenging projects and constantly pushing me to bring my research to a higher level. I would also like to thank the members of my Ph.D. committee, Enrico Bertini, Jean-Daniel Fekete, and Juliana Freire (my unofficial co-advisor), for their expertise, ideas, and feedback.

My research would not be possible without some amazing collaborations I have had throughout the years. I would like to thank Carlos Dietrich, Dan Cervone, and Marcos Lage, for introducing me to Sports Analytics. Peter Xenopoulos, João Rulff, Yurii Piadyk, Arvi Gjoka, and Justin Chiang for their inspiring passion for sports. Jun Yuan and Gromit Chan for the discussions on visualization and XAI. Justin Salamon for our collaboration on sports sound analysis. Sonia Castelo, Roque Lopez, Raoni Lourenco, Yamuna Krishnamurthy, Remi Rampin, Bowen Yu, Cristian Felix, Josua Krause, Iddo Drori, Kyunghyun Cho, and Ray Hong, for our work on the D3M project. Gustavo Nonato, for his guidance throughout my Master's studies and for encouraging me to pursue my Ph.D. in the United States. My first research advisors, Marco Caldeira and Antonio Sementille, who introduced me to graphics and visualization.

I would also like to thank the funding agencies that supported the work presented here: National Science Foundation (NSF awards CNS-1229185, CCF-1533564, CNS-1544753, CNS-1730396, and CNS-1828576), DARPA D3M Program, MLB Advanced Media, NASA, Moore-Sloan Data Science Environment, Labex DigiCosme (France), CNPq (Brazil) and FAPERJ (Brazil). Any opinions, findings, and conclusions or recommendations expressed in this thesis are those of the author and do not necessarily reflect the views of the funding agencies.

Finally, I thank all my friends. Aecio, Aline, Chris, Dani, Fernando, João, Joe, Juliana, Laura, Neel, Raoni, Roque, Sonia, Tiago, Vini, Vivian, and Yamuna, thank you for being my family in NYC. Luke and Kyle, thanks for being my support from the north. My oldest friends, Bruno, Camilla, and Mateus, thank you for

the laughs, stories, and shared experiences since high school. I am grateful to all members of the VIDA group for making grad school such a fun and exciting journey. Thank you for the coffee breaks, parties, and discussions. Special thanks to the NYU admin team, Ann Borray, Kari Schwartz, Eve Henderson, and Susana Garcia.

Jorge Henrique Piazentin Ono
September 2021

To my family and friends.

ABSTRACT

Visualization Methods for
Sports Data Collection and Analysis

by

Jorge Henrique Piazentin Ono

Advisor: Prof. Cláudio T. Silva, Ph.D.

Submitted in Partial Fulfillment of the Requirements for
the Degree of Doctor of Philosophy (Computer Science)

September 2021

With the recent advancements in tracking sensors, major league team sports can now generate a large volume of spatiotemporal data for analysis. Spatiotemporal data has revolutionized sports analytics, enabling experts and fans to compute statistics on demand, evaluate players' performance, improve strategies and even prevent injuries. However, several challenges arise because of the data complexity and volume, including the high cost of acquisition, storage, and exploration. For example, collecting trajectories for a single season of Major League Baseball costs millions of dollars and takes more than one terabyte of storage space.

My thesis aims to investigate the use of visual analytics systems to address the challenges surrounding spatiotemporal sports data. We present our results on visualization and visual analytics methods that facilitate the trajectory data acquisition, the exploration and analysis of baseball datasets, and the understanding

of the spatiotemporal evolution of a play.

First, we address the challenge of organizing and exploring large baseball trajectory datasets. Baseball has led the development of tracking technology in sports, and currently, each season of the game generates terabytes of data. We present our work on building StatCast Dashboard, a visualization and analytics infrastructure to support the study of baseball tracking data. We demonstrate the tool's usefulness by describing a use case on the exploration of the 2015 MLB season game dataset.

Second, we tackle the spatiotemporal analysis of plays. In sports, Play Diagrams are the standard way to represent game information. However, there are situations where these diagrams may be hard to understand, such as when several actions are packed in a particular region of the field or when the order of the actions affects the outcome of the play. To address this limitation in the baseball game, we present Baseball Timeline, a visualization that can convey the temporal and the spatial evolution of plays using a 2D diagram. We generalize this approach to other sports in our follow-up work, TrackRuler, a sports-agnostic chart that encodes trajectories using a ruler metaphor.

Next, we investigate the contextual analysis of baseball plays. Traditional sports analytics systems use statistics to compare and rank players. However, these systems do not take into consideration the context of the play. More specifically, they can make unfair comparisons, as context heavily influences the player's decision-making. For example, a batter may run slower when making a sacrifice bunt, and faster during a grounder. While sports commentators are proficient in comparing plays, an amateur viewer would have trouble analyzing plays in such a detailed manner. We propose GameCast, a baseball commentary system that uses play clustering and natural language generation to craft a contextual narrative while allowing for a fair exploration of the game. We validate our tool with use cases motivated by fans and sports journalists.

Finally, we improve upon trajectory data acquisition methods for sports. The sports data tracking systems available today are based on specialized hardware. While effective, implementing and maintaining these systems pose several challenges, including the high cost and the need for close human monitoring. Manual annotation is an alternative to automatic tracking systems. However, they can put too much

burden on the user. We propose HistoryTracker, a methodology that facilitates the creation of tracking data for baseball games using a vast collection of historical data to warm-start the annotation process. We evaluate our tool with the help of baseball experts and show that it helps users produce tracking data in a fast and reliable way.

We demonstrate our methods with real use cases and domain expert evaluations. Our contributions focus on the analysis of baseball games, but we also present discussions on extending the work to other sports.

Contents

List of Figures

List of Tables

Chapter 1

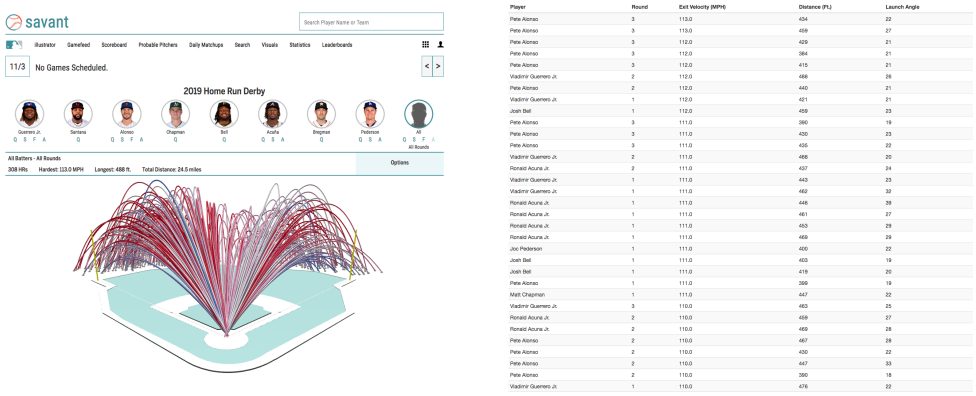
Introduction

Analytics play a significant role in today's team sports, where statistics and computational tools are frequently used to evaluate players and explore games [67]. Baseball has led the data-driven movement, and currently, vast amounts of baseball statistics, video, and tracking data are available for analysis [27]. This growing volume of information opens up new opportunities for coaches, analysts, and fans to engage in a data-driven way to better understand sports, explore player performance, prevent injuries, and design new playing strategies [2, 18, 22, 27, 99]. However, given the complexity, volume, and multi-modality of sports data, several challenges arise, including effectively collecting tracking data, analyzing and visualizing plays, and exploring extensive game repositories.

In this dissertation, we are interested in tackling four of these challenges regarding trajectory data: 1) Improving the organization and navigation of large sports data repositories; 2) Designing compelling visualizations for sports trajectory data, 3) Designing tools that allow the contextual analysis of sports plays and 4) Reducing the cost of trajectory data collection. We will present four visual analytics systems that address these challenges with a focus on the baseball game. We also discuss strategies to adapt these techniques to other sports.

1.1 Motivation

Trajectory data has revolutionized how professional sports teams inspect games, enabling their analysis at an unprecedented level of detail. Since 2015, Major



(a) Hit trajectory (b) Statistics table

Figure 1.1: MLB's Baseball Savant system showing (a) ball trajectory and (b) ball statistics during the 2019 Home Run Derby [52].

League Baseball (MLB) has been collecting player and ball positions using MLB's proprietary tracking system, StatCast [41]. This system enabled the computation of statistics on demand for professional analysis and interactive content creation for fans. For example, Figure 1.1 shows a screenshot of Baseball Savant [52], an online system that enables fans to explore baseball statistics and historical tracking data. Batter and ball statistics for the 2019 Home Run Derby are shown in a graphical and tabular format.

Such data opens up new and exciting opportunities for sports analytics, enabling experts and fans to learn more about the sport, develop and test hypotheses, and improve the game. However, given the complexity and diversity of the data, several challenges arise. We discuss some of these challenges below.

Challenges

1. **Data Organization and Navigation.** Major League Baseball (MLB) has a long history of capturing detailed, high-quality data from its games, leading to a tremendous surge in sports analytics research in recent years. In 2015, `MLB.com` released StatCast, a spatiotemporal data tracking system that captures player and ball locations, as well as semantically meaningful game events and statistics. The collected data takes more than one terabyte of disk space per season. Managing and understanding this extensive collection of information

is challenging, particularly for stakeholders with little programming knowledge. Chapter 3 shows how Visual Analytics can help in this process.

2. **Spatiotemporal trajectory visualization.** In sports, Play Diagrams are the standard way to represent play information. While widely used by coaches, managers, journalists, and fans in general, there are situations where these diagrams may be hard to understand, for example, when several actions are packed in a particular region of the field or when there are just too many actions to be transformed in a clear depiction of the play. The time and the relationship among the players' actions through time are critical in depicting complex plays. However, this information is not readily available on Play Diagrams. To address this issue, Chapter 4 presents a novel visualization that can convey both the spatial and temporal aspects of baseball plays. We present a generalization of this method to other sports in Chapter 5.

3. **Play Contextual Analysis.** Analyzing plays in the context that they happen is an important skill in sports analytics, as context heavily influences player decision-making. For example, in baseball, the speed of the batter en-route to first base is much different for sacrifice bunts than it is for sharp infield grounders. While sports commentators are used to this task, an amateur viewer would have trouble analyzing plays in such a detailed manner. Chapter 6 introduces a visual analytics system that can automatically generate context-aware commentaries that highlight the interesting aspects of a play.

4. **Trajectory Data Collection.** The sports data tracking systems available today are based on specialized hardware (high-definition cameras, speed radars, RFID) to detect and track targets on the field. While effective, implementing and maintaining these systems poses many challenges, including high cost and the need for close human monitoring. As a result, this type of data is restricted to large teams and professional leagues. Chapter 7 introduces a new manual tracking system that produces accurate tracking data at a low cost and reduces the human annotation effort.

1.2 Contributions

In this dissertation, we describe five contributions that address the challenges described in Section 1.1. The main contributions can be summarized as follows.

1. **StatCast DashBoard** (Chapter 3). To address challenge 1, we present a visualization and analytics infrastructure to help query and facilitate the analysis of large volumes of baseball tracking data [41]. Our goal is to go beyond descriptive statistics of individual plays, allowing analysts to study diverse collections of games and game events. Our system enables the exploration of the data through a simple yet powerful querying interface and a set of flexible and interactive visualization tools.

2. **Baseball Timeline** (Chapter 4). To address challenge 2, we present a study on how player movement and actions may be depicted on a 2D timeline diagram [62]. The work focuses on baseball plays, a sport where diagrams are heavily used to summarize players' actions. We propose a new and straightforward approach inspired by Marey's graphical train schedule, representing spatiotemporal information in the form of a timeline.

3. **TrackRuler** (Chapter 5). The visualization presented in Chapter 4 takes advantage of the limited movement choices that players have in a Baseball game. To enable the spatiotemporal exploration of other team sports (challenge 2), we propose a novel sports-agnostic trajectory visualization.

4. **GameCast** (Chapter 6). To address challenge 3, we present GameCast, a visual analytics system for visualizing and comparing baseball plays that automatically generate sports commentaries using context-aware statistics. GameCast guides the exploration of baseball plays based on the interestingness of the commentaries and statistics computed.

5. **HistoryTracker** (Chapter 7). To address challenge 4, we introduce HistoryTracker [68], a novel manual tracking system that facilitates the creation of tracking data for baseball games. We reduce the human effort and annotation time by *warm-start*ing the annotation process using a vast collection of

historical data. We show that HistoryTracker helps users to produce accurate data in a fast and reliable way.

1.3 Organization

The remaining of this thesis is organized as follows. First, Chapter 2 discusses the related work on sports analytics and visualization. Next, Chapter 3 introduces StatCast Dashboard, a system that enables the interactive exploration of an entire collection of baseball plays. Chapter 4 presents Baseball Timeline, a visualization of baseball trajectory data that conveys spatial and temporal information about single plays. Chapter 5 describes TrackRuler, an extension of Baseball Timeline to the visualization of trajectory data in general team sports. Chapter 6 presents GameCast, a system that allows the automatic generation of sports commentaries and the identification of interesting plays. Chapter 7 introduces HistoryTracker, a novel sports annotation framework that uses previously acquired knowledge to minimize user interaction. Finally, Chapter 8 concludes the dissertation, highlighting potential future works.

Chapter 2

Related Work

The area of sports analytics has exploded in the last few years. Now, most team sports use game statistics and tracking data to evaluate players, discover playing patterns, and develop new strategies. In the following, we describe the related work on *sports tracking* (Sec 2.1), *sports retrieval* (Sec 2.2) and *sports visualization* (Sec 2.3).

2.1 Sports Tracking

Tracking data is commonly used in various sports applications, both for entertainment purposes and for expert analysis. In the United States, some of the major examples are Major League Baseball (MLB), National Football League (NFL), and National Basketball Association (NBA). Since 2015, MLB has been using its tracking infrastructure, MLB StatCast, to augment its broadcasting videos and generate new content to the public [41, 86]. NFL and NBA also deploy tracking technologies to augment their broadcastings and compute statistics for fans [29, 59]. Sports teams and leagues frequently use tracking data to analyze and improve player performance and game strategies. More recently, major leagues are starting to make this data public in order to engage fans and allow them to do their own analyses [52].

Automated methods generate most of the sports tracking data produced by mainstream media. Commercial systems, such as Pitch F/X [30], ChyronHego TRACAB [21], and STATS Sport VU [80] are used at every game from major

league sports teams, producing huge amounts of data for analysis. These modern tracking systems use specialized sensors, like high-definition cameras, speed radars, or RFID technology, to collect movement data at a high precision and sampling rate [27, 74]. For a review on automatic tracking methodologies, please refer to the surveys by Santiago et al. [74] and Kamble et al. [38].

Tracking systems produce a valuable stream of data for analysis by sports teams. However, implementing and maintaining these systems pose three major difficulties. 1) They may be expensive: Major League Baseball's Statcast, for example, was an investment of tens of millions of dollars [86]. Such costs are not a problem for big sports teams and leagues, but they may be prohibitive for small entities or amateur players. 2) The quality of the tracking data may be affected by several factors [27, 64]: changes in lighting, camera position, occlusion, and small objects can result in missing or noisy data, and 3) these systems cannot produce tracking data for historical plays, which commentators and analysts often reference during their analyses. However, if the game happened before the tracking system was implemented, it is impossible to compare the plays quantitatively.

While professional sports leagues have shifted towards automated methods, they are very protective of their data, only sharing small aggregated statistics with the public. Therefore, manual tracking is still used when the data is not readily available, e.g., academic research and amateur teams [64, 67]. Before developing automatic tracking systems, experts had to manually perform the annotation of players and ball position manually [74]. For example, Spencer et al. [78] hand-annotated hockey players' movement and speed throughout multiple games to analyze how player performance changes during a tournament. Bogdanis et al. [12] hand-annotated basketball games in order to compare the effects of training programs on players. The annotation was made offline, using video footage of the game and training sessions, and the experts had to collect and annotate both player trajectories and actions: e.g., dribbles and offensive/defensive moves.

Crowdsourcing has also been used to generate sports data [64, 83, 88, 89]. Crawling Twitter streams enables the extraction of game highlights, where hashtag peaks might indicate the most exciting moments in the game [83, 88]. While this technique does not produce tracking data, highlights are a valuable data source that can be gathered from a publicly available platform. Vondrick et al. [89]

investigated the use of crowdsourcing interfaces to annotate basketball videos. The authors divided the work of labeling video data into micro-tasks that many human annotators could complete and showed that combining the multiple users' output resulted in more accurate tracking data. Perin et al. [64] followed the same principles but extended this approach to enable the real-time annotation of games. Each person is asked to annotate either one player or one event in their system, and averaging multiple annotations resulted in high data accuracy. While micro-tasks made the annotation process easier, it has the downside of requiring a large number of users to produce a single play annotation.

Hand annotating sports is a time-consuming and challenging task. Meanwhile, automatic sports tracking systems are expensive and may contain errors. Chapter 7 presents HistoryTracker, a system that facilitates the manual tracking of Baseball plays by using a warm-start approach. Our methodology enables users to quickly annotate sports games by initializing the annotation process with a vast collection of historical tracking data.

2.2 Sports Information Retrieval

Another research area closely related to play annotation is called sports information retrieval. With massive amounts of sports tracking data being generated every year by automated systems, it has become harder to organize and search this data.

Many methods have been developed to retrieve particular games based different types of queries, for example text-based [32, 101] and sketch-based [77]. Early work in sports information retrieval focused on retrieving games based on textual queries. Fleischman et al. [32] developed a language model for baseball game retrieval built on top of closed captions. Their system enabled users to query specific game events as long as the commentator described them. The automatic classification of video footage can also be used to enable quick retrieval of games. Zhou et al. [101] proposed a basketball video classification system based on decision trees and used this system to retrieve games based on a set of textual constraints, such as player position (right field, left field), scores, and types of offense/defense.

One of the most advanced play retrieval methods is based on sketches: Sha et

al. [77] designed a system for querying basketball games with hand-drawn sketches of the top view of the field. While this method was very successful in retrieving similar trajectories, it has the downside of essentially asking a user to draw the play's entire trajectory by hand.

The challenges surrounding the query and organization of sports data are prevalent in Baseball, where the size of the data can exceed one Terabyte per season [93]. In Chapter 3, we present StatCast Dashboard, a visualization and analytics infrastructure that helps the study of baseball tracking data. Our goal is to go beyond single-play statistics, allowing the study of a collection of games and their trends. StatCast Dashboard allows the exploration of the data through a simple yet powerful querying interface and a set of flexible, interactive visualization tools. Chapter 6 introduces GameCast, a system for the exploration of live Baseball games that identifies game highlights for the user, taking into consideration the context of the play.

2.3 Sports Visualization

A vast collection of works show how information visualization techniques can be used to inspect sports data in more detail. Some examples include tennis [70], baseball [27], basketball [34, 77, 84], soccer [65, 81, 82], hokey [69] and rugby [19, 20]. While each of those works are adapted to better illustrate their respective sports, their main focus is on clearly conveying the trajectories, or metrics computed from trajectories, to the user.

The use of diagrams is widespread in sports to represent strategy, formation, or complex player actions. The diagrams can be roughly grouped in terms of the type of information that they portray: *event visualizations*, *single play trajectory*, *aggregated spatial data*, and *aggregated statistics data*.

Event visualizations present a high-level description of play or game highlights (or *events*) in order to contextualize and explain the main developments in the game. They are usually associated with a temporal representation, where events are sorted on time and provide a view of the action flow through the play. Event visualizations have been used in soccer [92, 97], tennis [90, 94, 95] and Pommerman [1]. Wongsuphasawat et al. [92] proposed Outflow, a system that uses Sankey

diagrams in order to portray the evolution of soccer events during a game season (i.e., wins, losses, and draws). The tool aggregates event sequences to form pathways connected with outcomes to model how alternative chains of events may lead to different results. Xie et al. [97] developed Passvizor, a system for studying pass behavior in soccer. The system uses a topic modeling approach to provide a high-level abstraction of passing behaviors over time.

Tennis and table tennis events have been studied from different perspectives as well. Wu et al. [95] developed a system for the analysis of tennis plays from three viewpoints: time-oriented, statistical and tactical. The system allows users to analyze stroke patterns and detect tactics using a timeline. Wang et al. [90] proposed Tac-Simur, a system that enables users to visualize tactics in table tennis games using a small multiples representation of hit events. The system builds an internal representation of games using Markov models and enables users to do what-if analysis. More recently, Wu et al. [94] created a system that models each rally in the game as a sequence of hits, which are graphically represented using custom glyphs in a time series visualization. A steerable sequential pattern mining algorithm is integrated into the system in order to enable the identification of tactical patterns and playing styles.

Visualization has also been used to understand e-sports events. Agarwal et al. proposed Bombalytics [1], a timeline visualization that represents events in Pommerman, a multiplayer bomb laying game. Bombalytics enables the summarization of games using game statistics and the exploration of individual games in detail using the timeline view, which shows player events and interactions between players, for example, bombs laid, explosions, kills, and power-ups.

Single play trajectory visualization is a popular approach for the visualization of actions in sports. The movements of the targets (players and the ball) are depicted as Play Diagrams, i.e., top-view mappings of the field, where movements are represented as lines in the chart. We can find examples of these charts in basketball [84], baseball [27, 41], soccer [65, 82], badminton [98] and tennis [70, 71]. Sport-specific information can also be encoded in the chart; Theron et al. added player speed information using colors and events, and information about team strategies, such as dribbling and defensive traps, using icons. Dietrich et al. [27] added annotations to the tracking data to highlight baseball statistics like player

top speed and ball velocity. Stein et al. [81] created an icon-based representation for soccer plays, where movement and interaction among players are represented in an adapted Play Diagram. The authors also developed a system that shows the actual video footage as a diagram overlay to give even more context to the visualization [82]. Polk et al. [71] used a small multiples representation of tennis plays to convey the ball's position over time. The visualization also enables the querying and sorting of plays based on statistics to facilitate pattern discovery. Ye et al. [98] designed a 3D virtual reality environment to enable the analysis of badminton games. The system presents the trajectory data using animations in a 3D field and takes advantage of the human's peripheral vision to present game statistics.

Aggregated spatial visualizations are used in sports to show trends in the position of players or ball over a large period of time. In baseball, for example, it can be used to identify trends in the ball landing position per pitch type [27, 41, 56]. In basketball, it can be used to show the regions of the court where players score more points [34]. This data is traditionally visualized using heat maps or scatter plots contextualized with a top view of the field of play in the background. Pileggi et al. [69], for example, proposed a radial heat map to encode the proportion of shots taken by the distance to the goal in hockey games. Cross et al. [25] modeled players' spatial batting ability and presented their models using heatmaps. This approach naturally highlights the areas in the strike zone where the batters had good performances. Losada et al. [45] also used classic and radial heatmaps to analyze offense and defense in basketball games. The movements of the targets can also be aggregated in the visualization. Sacha et al. [72] proposed a visualization that combines soccer player trajectories to summarize the overall movement during the play. The authors presented an approach that simplifies and cluster trajectories, thus reducing the visual cluttering.

Statistics visualizations are also a ubiquitous tool in sports. Cox et al. [23] proposed Sportsvis, an interactive system that show statistics about baseball games. In Sportsvis, bar charts and treemaps represent baseball statistics, such as the number of home runs and on-base percentage per game. The visualization of statistics is also studied in the context of soccer analytics. Albinsson et al. [2] proposed a system that used linked bar charts and scatter plots to enable the interactive

querying and filtering of soccer statistical data. Perin et al. [66] introduced an augmented tabular representation for soccer statistics, which highlighted changes in the order of the rows to show how a team progressed over time. Cervone et al. [17] proposed a new metric to evaluate basketball players and their ball pass events. In the paper, the authors used small multiples of Play Diagrams to convey the game's temporal evolution and a time graph to show how the proposed metric changed over time. Sports statistics have also been used to guide the exploration of sports videos using faceted search. Matejka et al. [48] proposed VideoLens, a system for the exploration of extensive collections of videos and associated metadata. The system was used in the context of Baseball Plays, where users could interact with event timelines to query the video collection.

All the visualizations mentioned above are effective at conveying static information about the sport they represent. However, they are not focused on the *temporal* aspect of the data. In sports, the knowledge of how the players and the ball move during a play or drill is necessary to reveal the participants' underlying strategy. The knowledge of *when* interactions among players took place, however, is critical for the understanding of complex situations. For example, in baseball, knowing that both the runner and the baseman arrived in a given base does not give enough information for inferring if the runner is safe or not. Chapter 4 presents Baseball Timeline, a visualization that enables the spatiotemporal analysis on baseball trajectories. We show a generalization of this method to other sports in Chapter 5.

Chapter 3

StatCast Dashboard: Interactive Exploration of Spatiotemporal Baseball Data

3.1 Introduction

Although many sports now use statistics and videos to analyze and improve gameplay, baseball has led the charge throughout the history of sports analytics. With the advent of new technologies that can track every player and the ball across the entire field, it is now possible to bring the understanding of this game to another level. In 2015, `MLB.com` released StatCast, a system that uses player and ball location, combined with semantically meaningful game events to capture games with unprecedented detail. The StatCast Metrics Engine [27] is one of the key components of the system, which uses discrete locations across time to reconstruct entire baseball games, enabling the computation of player statistics on demand, such as "route efficiency" or "lead distance", which allow for more detailed and accurate analyses of player and team performance.

The StatCast infrastructure is a complex system that involves new ballpark sensors, sophisticated software, and a completely new workflow to capture baseball games at a high level of spatiotemporal detail. Games are stored as collections of actions from the beginning to end of the play, called a *gameplay*; other ancillary data, including videos and metrics generated during each play, are also stored. This

wealth of new data is just starting to be used for analysis.

In this chapter, we present our work on building a visualization and analytics infrastructure to help in the study of the StatCast baseball tracking data. Our goal is to go beyond single-play statistics to allow the study of a collection of games. Our system allows the exploration of the data through a simple yet powerful querying interface and a set of flexible, interactive visualization tools. The results described in this chapter have been published in [41].

The rest of this chapter is organized as follows. Sections 3.2 and 3.3 provide an overview of the Baseball game and the StatCast data. Section 3.4 presents StatCast Dashboard, our visual analytics system for the study of StatCast baseball tracking data. Section 3.5 presents an use case that shows the power and flexibility of our system. Finally, we discuss our results in Section 3.6.

3.2 Baseball Overview

Baseball is a bat-and-ball game that is played on a field shaped like a diamond. The field has a set of four bases placed at the corners of a ninety-foot square at the bottom of the diamond. The bases are labeled in counter-clockwise order starting at the bottom as home (or home plate), first, second, and third. The area right above the square is called *infield*, while the area above the dirt is called *outfield*. During the game, the teams alternate between the nine defensive and the four offensive roles. The defensive roles are the pitcher (P), the catcher (C), the basemen (1B, 2B and 3B), the short stop (SS) and the outfielders to the left (LF), center (CF) and right (RF). As the offensive roles, there are the batter (B) and zero to three runners on bases (R@1, R@2 and R@3). Figure 3.1 shows the field and the players average positions. The runners are not shown in the picture for conciseness, but their starting positions are next to the first, second and third bases.

Baseball is a very structured sport. It consists of nine innings, each of which are divided into two halves with teams taking turns on attack and defense. In general, a play starts when the pitcher makes the first movement and finishes when the ball returns to the pitcher's glove or goes out of play. Every player has a fixed initial position, and the set of actions they perform is relatively limited. Players in the offensive role try to touch all four bases in anti-clockwise order (1st, 2nd, 3rd and

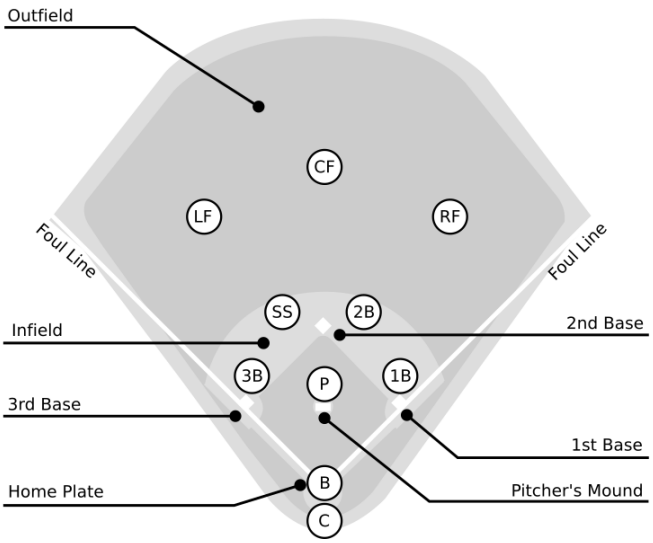


Figure 3.1: Baseball field of play and player positions: pitcher (P), batter (B), catcher (C), infielders (1B, 2B, 3B and SS) and outfielders (LF, CF and RF).

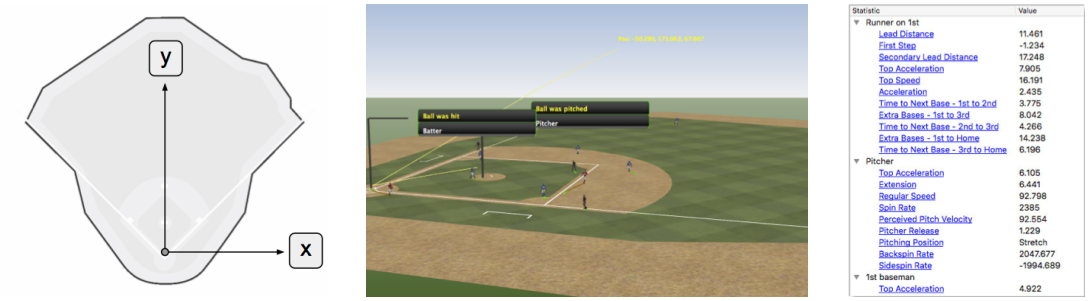
home plate). Meanwhile, players in the defensive role try to catch the ball and eliminate the attackers.

3.3 StatCast Overview

The StatCast project is an effort to capture all actions performed in the play field and process them to generate interesting content for the public. The project comprises both the hardware necessary to capture the players and ball behavior as they move over the field (cameras, radars, etc.) and the software layer required to gather and process this information. This section we will focus on the characteristics of the tracking data (the data captured from the players and the ball) and on the processing of that data for the computation of metrics inform the public about the performances players and teams.

3.3.1 StatCast Player and Ball Tracking

For each play, there are detailed measurements of each players' location, of the ball's position and of the ball-related events (pitch, catches, releases, etc.). These measurements are processed to generate player- and team-level metrics, like,



(a) (b) (c)

Figure 3.2: StatCast Overview. Coordinate system (a); Player and ball tracks and game events (b); partial list of metrics computed by the system (c).

for example, all metrics related to the pitch: pitch release time, speed, back-spin, side-spin, extension, and so on. Figure 3.2 shows an example play (b) and a list with a subset of its associated metrics (c).

StatCast works by optically-tracking player location in the field as (x, y, z) measurements at high frame rate (nearly 30 Hz), which gives a detailed description on what the players do on the field. For each player, there is a sequence of positions recorded for the duration of the play. In the StatCast coordinate system, $(0, 0, 0)$ is at home plate where the batter is, the y-axis points to pitcher's mound, and the x-axis points to the right, going parallel to a line between the 3rd and the 1st bases (see Figure 3.2a for an illustration).

The player tracking system used is ChyronHego Tracab. This sophisticated system uses three pairs of stereo cameras to track in real-time all the players in the field. The system "compresses" all the information about a player to a two-dimensional point. The process of taking complex human motion in three dimensions and turning into a single two-dimensional point eliminates a good amount of fine details, which obviously limits the kind of movement analysis that can be performed on the output. For instance, consider a runner trying to reach a base by sliding. What (x, y) position should be reported by the system during this movement? Will the player's location ever truly overlap with the geometric position of the base? Answering these questions depends on the exact algorithm used for such a conversion.

One way to achieve this is described by Borg [13] in the context of tracking

soccer players. Borg's algorithm uses the pairs of stereo images to compute a set of three-dimensional voxels (i.e., a volume element) that belong to each player as they move. This computation involves finding corresponding pixels on the different images and using stereo disparity to recover the depth information of the voxels. These three-dimensional data points are then projected in the field to compute a two-dimensional center of mass of the voxels, which are used as an approximation for the center of mass of the player. Note that player movement affects their "location" even if they are not actually moving at all; imagine a player swinging his arms — the voxels detected by the system will change over time, and their projected location will change as well. One way to think of the tracking algorithm is that it is *integrating* a complex motion into a simpler signal. A detailed analysis of the accuracy and the limits of this system is beyond the scope of this chapter, but the knowledge of the measurement process characteristics is crucial to the design of the metrics computation and analysis.

The ball moves much faster than the players, so it requires a higher sampling rate. The system uses Trackman radar technology to track the ball. Using a radar solution has a number of advantages over optical tracking, most notably the higher temporal resolution. This enables the computation of nearly exact timing of significant game events, for instance, when the ball leaves the pitcher's hand, or when it is caught by a fielder. The ball is also optically tracked, and in cases where the radar measurements are unavailable (e.g., a rolling ball on the field is hard to track with the radar), the optically tracked positions are used.

3.3.2 StatCast Metrics Engine

The StatCast Metrics Engine uses discrete positions across time to reconstruct entire baseball games, enabling the computation of new player metrics. The system that is currently deployed is a highly refined version of the one described in [27]. The StatCast Metrics Engine works on top of data about player and ball location, and semantically-meaningful game events. The baseball game is taken as a continuous stream of data and events, including pitches, catches, throws, and player movements. In general, each play starts when the pitcher goes into his windup and finishes when the ball returns to the pitcher's glove or goes out of play (e.g., a home run or foul ball). Then, plays can be divided into three parts: the pitch (i.e., actions

from the windup to the moment the ball is in the front of home plate), the hit (i.e., actions from the moment the ball is hit to the moment it is fielded), and the field. Understanding the state of the ball is critical to determining what part of the play we might be in.

Besides the positioning data, a stream of "game events" is key to being able to reconstruct the gameplay. Game events are the high-level events associated with the ball while it is in play—the moments when a specific player hits, obtains possession, or relinquishes possession of the ball. The data stream is composed of tuples that contain a time stamp, game event, and player id. A minimum set of important game events include: "ball was pitched", "ball was caught", "ball was released", "ball was hit", "end of play", "pick off released", and "ball was deflected".

At a high level, the StatCast Metrics Engine works by filtering the location data merging the game events, and eventually reconstructing a state machine that represents the game. It computes a wide set of metrics for players grouped in different categories, for instance, there are baserunning metrics (e.g., player acceleration, speed, and lead distance), fielding metrics (e.g., arm strength, pop time, route efficiency), and pitching metrics (e.g., extension, release time, spin rate, perceived velocity). We note that the StatCast Metrics Engine is a surprisingly sophisticated piece of software. One of the most complex parts of the system are the data filtering operations. The system performs a substantial amount of error checking, which is used in other parts of StatCast, including informing human operators when parts of the system might be malfunctioning or need user input.

3.3.3 Building The Analysis Database

In order to build a unified database that enables interactive visualization and exploration of the gameplay tracking data, we merge the StatCast data with the MLB.com *Game Metadata Directory*. The Game Metadata Directory provides metadata information about the baseball games such as the date and time they were played, their scoreboards, players appearances, among others. The data is publicly accessible through a web API.

The Dashboard Database is a document-oriented database that enables fast querying over the derived gameplay statistics and the tracking dataset. Also, in order to facilitate the querying process, we developed a simple keyword-based query

system. The Dashboard database is composed of five different document collections, described below.

In the *games* collection, each game is represented by one document and in order to allow for fast queries, we use different game properties as indexes. More specifically, we chose to have the game unique identifier, the team names, and the game date as indexes since we were interested in filtering games by dates and involved teams or analyzing the data produced during a specific match. We also have corresponding data collections for *players* and *lineups*. Since baseball analysis is often focused on pitcher-batter match ups and we are interested in studying optimized indexes for pitchers, batters and fielders.

The last and largest part of our analysis database is based on the output of the StatCast Metrics Engine on each gameplay. The StatCast Metrics Engine outputs a clean and filtered dataset that contains all of the positioning data for the players, the ball, and related game events and metrics in a normalized format. We saved this set of *tracking* information and related *metrics* in our database as two separate document collections.

The StatCast tracking data is a large dataset. Each of MLB's thirty teams plays 162 games during the regular season and a typical game has between three hundred and four hundred plays, so the total number of plays recorded during the season is in the order of 700,000 plays that occupy 1.5 terabytes on disk.

3.4 StatCast Dashboard

The purpose of the StatCast Dashboard is to make it easier to explore and analyze the StatCast data. The complexity and richness of the data will certainly require different visual user interfaces depending on the target audience and applications, but at this point, our goal was to build a system that enables initial studies of the data. More precisely, we decided to design a web-based system that allows for an easy way to query, filter, and explore a large collection of gameplays, the related metrics computed by the StatCast Metrics Engine, and the tracking data. Here we list the specific system requirements:

R1: The user should be able to query gameplays based on a flexible selection mechanism, without the need to knowledge of programming or database technology.

For example, the system should allow the user to select gameplays played by Michael Pineda from August 2015 to October 2015 or gameplays from games between Mets and Yankees that Juan Lagares played as a fielder.

R2: The user should be able to spatially visualize individual gameplays or groups of gameplays. The visualization of individual gameplays is important since it gives the user the ability to study the behavior of players and the ball. The visualization of groups of gameplays provides a global understanding of the patterns and can help the study of game strategies.

R3: The user should be able to interactively filter the queried gameplays based on the metrics computed by the StatCast Metrics Engine. That is especially useful when the user is interested in performing comparisons of the type: which right fielders had the highest *Top Speed* values or which pitchers had an *Extension* in a range of interest.

R4: The user should be able to draw over the field to interactively define regions that in turn directly filter the queried gameplays. That is, the system should enable direct interaction with the tracking data. For example, the user can find gameplays where the center fielder ran to their left to catch the ball.

R5: Finally, the user should be able to export the data, that is, the metrics and the tracking information of the gameplays that were selected. This functionality allows the user to continue the analysis of the identified gameplays using other tools like R or Python.

To address the previous requirements, we designed a visual user interface for the StatCast Dashboard composed of three core components: the Keywords Manager, the Statistics Viewer and the Gameplay Viewer. The current version of the visual interface of the StatCast Dashboard is shown in Figure 3.3. In the following we will give a description of the main system interactions available through this interface, the strategies used to efficiently implement the available interactions and the interface components associated with each interaction.

3.4.1 Querying Gameplays

To facilitate the access to the database and address R1, we developed a keyword based querying strategy, that allows for users without programming knowledge to use the Dashboard and select gameplays from StatCast. The keyword query system

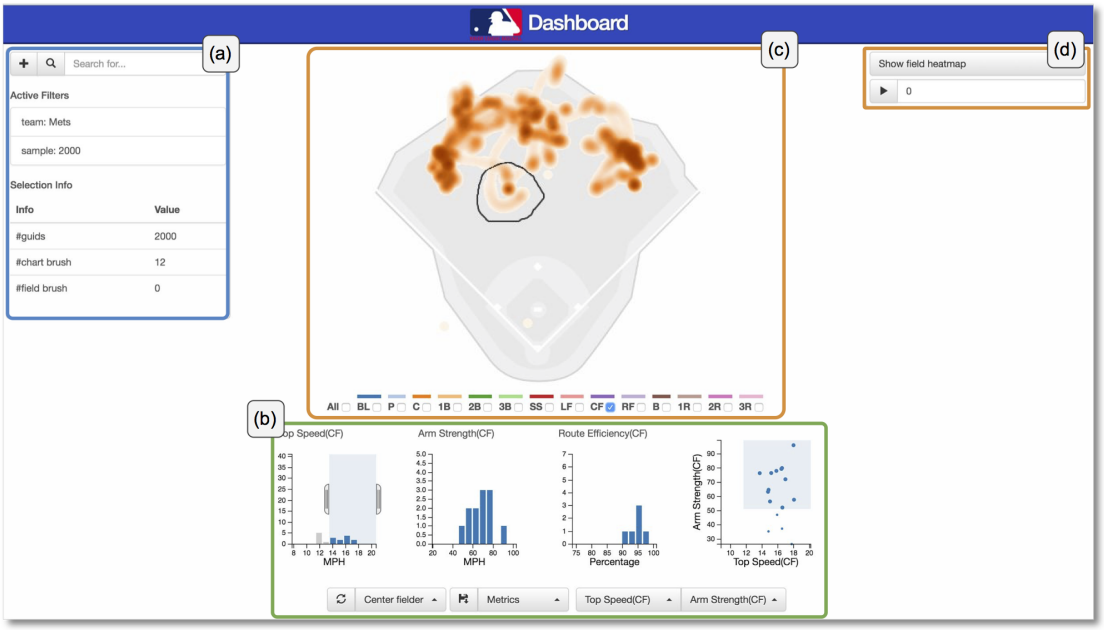


Figure 3.3: The StatCast Dashboard visual interface is composed of three core components, the Keywords Manager **(a)**, the Statistics Viewer **(b)** and the Gameplay Viewer **(c-d)**, that can be used by the user to query, filter, explore and export gameplays from the StatCast dataset.

is based on a set of available keywords and a list of active keywords instances. To create a new keyword instance, the user needs to write using the syntax `keyword:value1[,value2,...,valueN]`. Currently, the Dashboard supports ten different keywords: `team`, `vs`, `date`, `pitcher`, `batter`, `fielder`, `game_id`, `play_id`, `limit`, and `sample`.

The `team` keyword can be used to query gameplay data that involves a given team. For example, if one wants to load the gameplays played by the Boston Red Sox (both batting and fielding) he needs to add the keyword instance `team:Red Sox` to the keyword list. The `vs` keyword can be used to find gameplays of a user defined match. If one wants to analyze gameplays of games between New York Yankees and New York Mets, he needs to create the keyword instance `vs:Yankees, Mets`. We observe that the order of the teams is not important in this example. Similarly, the user can specify queries with the dates, the pitchers, the batters, the fielders, the game and gameplay identifiers of interest.

The `limit` and `sample` keywords are used to reduce the amount of data allowed

to be returned from a query. The first limits the current query to the first n returned gameplays and the second samples the query result uniformly to get n distributed gameplays. We also allow the use of wildcards and regular expressions. For example, one can write `pitcher:.*Pineda` to query Michael Pineda gameplays. For convenience, we use Python syntax for the regular expressions.

Keywords Manager. The Keywords Manager (shown in Figure 3.3a) is a widget that is dynamically updated whenever the user adds or removes a keyword from the actual list of active keywords used to describe a query. The user can add a new keyword to the list writing it in the text input and pressing the *add button*, the button with the plus sign icon shown in Figure 3.3a, or pressing return in the keyboard. To remove a keyword from the list the user simply needs to click on the undesired item. Once the list of keywords to be composed is done, the user can query the database using the *query button*, the button with the magnifier icon shown in Figure 3.3a. The keyword based queries in the list are them composed using an AND logical operator. For example, if the keywords manager has two active items, `team:Padres`, `batter:.*Amarista`, the query will return all gameplays that were played by San Diego Padres and has Alexi Amarista as the batter. The Keywords manager also contains an information panel that shows the number of gameplays returned by the executed query and selected using the metrics and the spatial filters that will be described next.

3.4.2 Filtering Gameplays

Once a query is performed, in order to allow the user to filter and explore the returned data, addressing R3 and R4, we developed two components of the StatCast Dashboard user interface: the Metrics Viewer and the Gameplay Viewer. Also, to address the requirements described by R2, the Gameplay Viewer supports two operation modes.

Metrics Viewer.. The Metrics Viewer (shown in Figure 3.3b and highlighted on Figure 3.4) is composed of a set of histograms and a scatter plot. Each histogram shows the distribution of a particular metric from the currently loaded set of gameplays. Since the metrics computed in a baseball game are closely related to the player position, we also provide a category selector that allows for the user

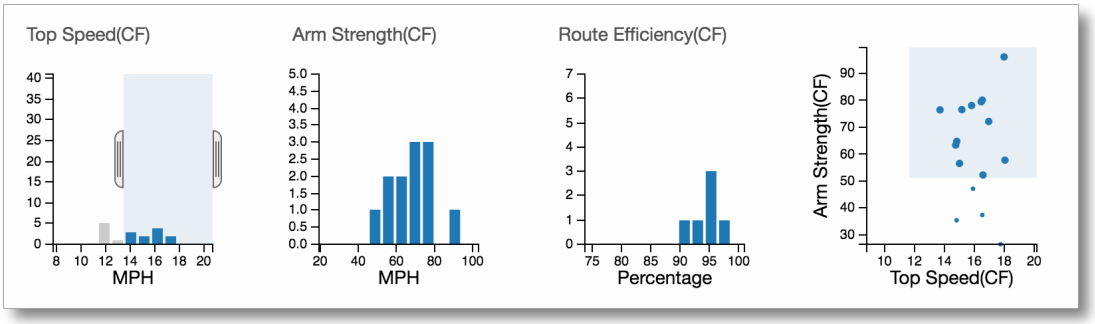
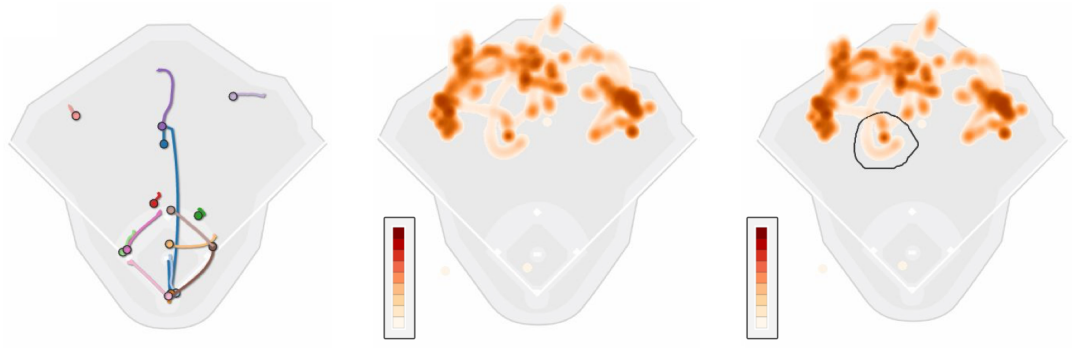


Figure 3.4: The Metrics Viewer is composed of a set of histograms and a scatter plot. The histograms shows the distribution of the available metrics on the gameplays returned by the query. The scatter plot shows the correlation between two of the available metrics. Both the histograms and the scatter plot can be brushed to filter the gameplays (light blue regions).

to change the set of histograms currently displayed. We also provide selectors to allow the user to change the axes of the scatter plot. Using the charts, the user can filter the loaded gameplays by brushing the desired ranges of each statistics that he is interested in. For example, if the user selects the gameplays where a fielder had a speed between 14 and 20 miles per hour, he has to click and drag the mouse over the interval on the *top speed* histogram of the fielder category (left chart, Figure 3.4). Also, the user can brush the scatter plot to filter gameplays (right chart, Figure 3.4).

After filtering the gameplays based on the desired statistics, the user can press the *load tracking data* button, the one with the arrows icon shown in Figure 3.3b, to load the tracking data of the filtered gameplays on the Gameplay Viewer.

Gameplay Viewer. The Gameplay Viewer (shown in Figure 3.3c) can be used both to visualize the tracking data and to filter the gameplays currently loaded on the field based on a user-defined spatial criteria. The widget presents a top view of the baseball field that is used as render context to display the gameplay tracking data. The tracking data can be visualized over the field using two different modes. The individual gameplay visualization mode shows the paths stored on the StatCast tracking dataset using poly lines (see Figure 3.5a). The individual gameplay can be animated to show the play evolution over time. The group gameplay visualization mode uses a heat map to show multiple gameplays' tracking information at the

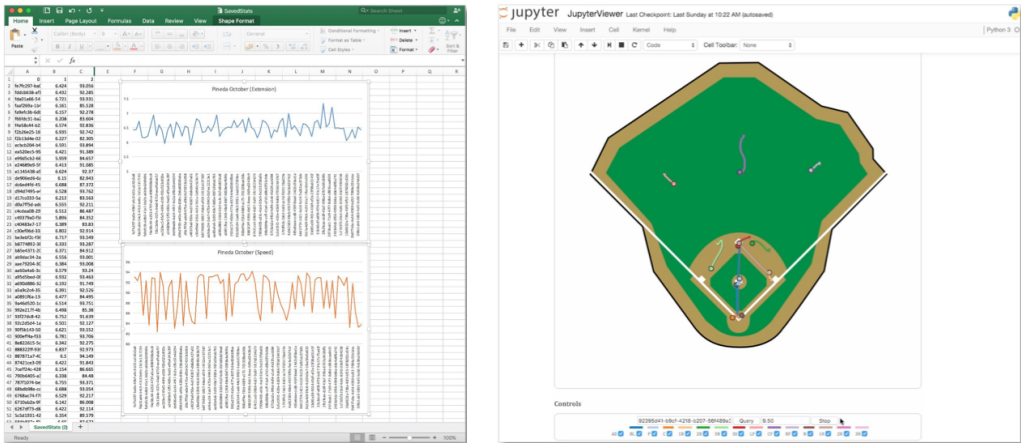


(a) (b) (c)

Figure 3.5: The Gameplay Viewer can be used to visualize the tracking data or filtering the currently selected gameplays using a spatial criteria. **(a)** Shows the individual gameplay visualization mode that can be used to review the behaviour of the players and the ball (shown in blue) through the rendering of their paths. **(b)** Shows the multiple gameplays visualization mode that can be used to understand global patterns on the positioning of the center field using a heat map. **(c)** Shows the user-defined spatial selection criteria represented by the region in black.

same time (see Figure 3.5b-c). The player positions to be considered on both visualization modes can be chosen using the positions selector right below the field diamond (see Figure 3.3c). For example, if one wants to visualize the fielder's movement on a set of gameplays, he needs to mark the left, center and right fielders check-boxes.

Using the field, the user can also define polygons of interest to filter the currently loaded tracking data. In order to do that, the user just needs to click and drag the mouse over the field to draw the desired geometry. Once the geometry is defined, all tracking points that fall inside the polygon are identified and their associated gameplays are stored. Since the number of tracking points on a set of gameplays is typically huge, the selection of tracking points inside the user defined polygon is optimized by the use of a kd-tree, a generalization of a binary search tree that stores points in k-dimensional space [73]. Every time a set of gameplays is loaded on the Gameplay Viewer, each tracking point present in the gameplay is added to the kd-tree. Once the kd-tree is built and the user defines the interest region, the bounding box of the region is computed and used to select the tracking points that can potentially be inside the region and the final set of points is selected performing



(a) (b)

Figure 3.6: Detailed analysis of data. (a) The dataset queried and filtered using the StatCast Dashboard can be exported in the `csv` format and used to perform analytics in external software. (b) The widgets of the StatCast Dashboard interface can be used inside Jupyter.

a point inside polygon geometric test.

3.4.3 Detailed Analysis of Data

We realize the need to drill down into the data for the purposes of performing detailed statistical analysis. We kept this in mind as we designed the StatCast Dashboard, and in particular we decided to enable flexible data paths for analytics using other tools. To achieve that, the metrics and the tracking information of gameplays selected using the StatCast Dashboard visual interface can be downloaded in `csv` format to be exported to other systems. The left hand side image on Figure 3.6 shows the pitch metrics from Michael Pineda in October loaded on Microsoft Excel. Also, the widgets we implemented in the StatCast Dashboard can all be embedded in Jupyter, which supports R and python for further analysis. This enables detailed analysis of all the aspects of the data, including manipulations of the wealth of numerical data as well as the use of graphical widgets for display and data selection and filtering. The right hand side image on Figure 3.6 shows the tracking data of a gameplay loaded on Jupyter using the StatCast Dashboard Gameplay Viewer. Because the Dashboard data is hosted using MongoDB, the

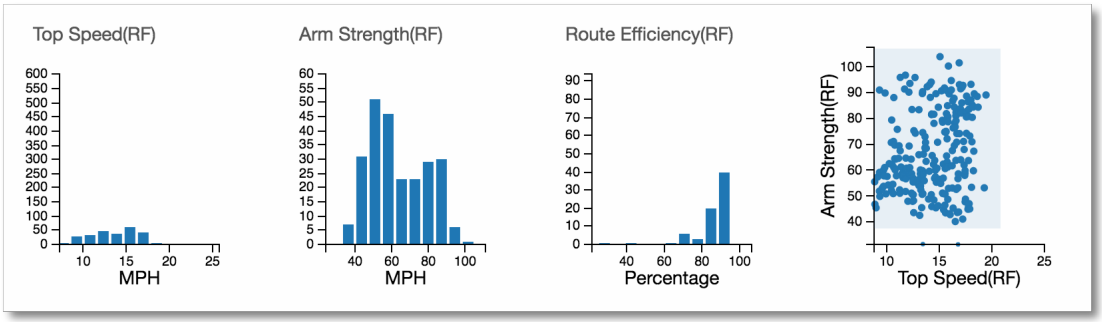
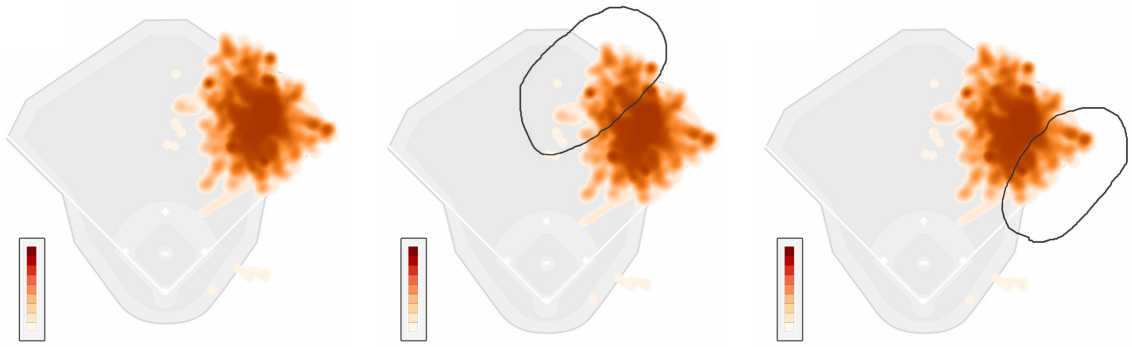


Figure 3.7: Example of use case, we want to find all plays during the season in which Bryce Harper fielded the ball. We can do this by selecting all plays for which is arm strength and top running speed are non-null.

MongoClient function from the `pymongo` module can access all the separate data collections, including the games collection, the tracking data, and the metrics collections. By using the Dashboard to collect the file names of the relevant plays, Python code can parse out data and create aggregate statistics such as averages or outliers of the data. Data can be processed by efficient Python libraries such as e.g., `matplotlib`, a rich plotting library can be used for graphing the data; or `scikit-learn`, which can be used to perform higher level machine learning functions such as clustering the data. This flexibility was used to produce the use case presented in Section 3.5.

3.5 Example Use Case

This section provides an example of how a quantitative researcher could use the dashboard to explore, query, and export data for statistical analysis. An exciting new application of the StatCast data is the ability to more precisely quantify and assess players' defensive abilities. For decades, baseball's defensive metrics depended entirely on coarse game event data, such as the number of balls a player caught (or didn't catch), the number of fielding errors he committed, and the number of baserunners he threw out. In recent years, defensive metrics have evolved to include subjective assessments from gameplay analysts, as well as batted ball trajectory information. StatCast, by tracking all nine fielders and the ball at high spatiotemporal resolution, provides data for novel defensive analyses based on



(a) (b) (c)

Figure 3.8: Refining the query on plays involving Bryce Harper using the Gameplay Viewer. Here, we split Harper's plays based on whether he moves to his left or to his right. The results of these sub-queries can be exported and analyzed for spatial variation in Harper's play characteristics.

positioning, reaction time, speed and route efficiency, and arm strength.

For example, we can analyze the positioning of Washington Nationals' right fielder Bryce Harper. Using the dashboard, we first find all plays involving Harper by querying and using a player filter for the right fielder. Metrics for the right fielder, such as top speed and arm strength, are only computed when that player is involved in the play, which in the case of Harper would require him to field the ball during the play. Thus, using the Statistics Viewer, we can select plays where metrics such as arm strength and top speed are non-null (see Figure 3.7). This produces a sample of all plays during the season in which Bryce Harper fielded the ball.

The Gameplay Viewer can be used to further refine this query and study variation in fielding metrics as a function of space. This variation can reveal inefficiencies in where a fielder is usually positioned—for instance, if a player is slower running to his right than to his left (which is possible, as players need to lead with their non-dominant arm for maximum range when attempting a catch), then he should be placed so that he runs right less frequently than left. For Bryce Harper, we see that for 107 plays, he ran to his left, whereas for 113 plays he ran right (see Figure 3.8). For each spatial query, we can export a `csv` containing the available metrics for each play in the query results, which enables myriad downstream statistical analyses.

With Harper's data, we see nearly equal maximum top speeds of 19.45 mph and 19.29 mph for moving right and moving left, respectively. Likewise, Harper shows equal top arm strength (101.5 mph) from both regions of the field. Combined with the nearly even split (113 to 107) or plays in each direction, we see no evidence that Harper is systematically sub-optimally positioned in the outfield, though the analysis performed was quite cursory.

3.6 Final Considerations

StatCast is a first-of-a-kind system that uses novel sensors, state-of-the-art game reconstruction software, and technology that glues everything in an end-to-end pipeline to create a "season library" of unprecedented detail.

This chapter described our first attempts at building an analytics and visualization stack to complement StatCast. We report on a typical analytics use case in which a quantitative researcher can use the system to explore, query, and export the data to perform statistical analysis on a particular player.

The aggregated play information presented in StatCast Dashboard is helpful for identifying global patterns in the data. However, it does not allow for the detailed exploration of individual plays. While the individual play animations in the Gameplay Viewer are useful, this approach becomes harder to understand in longer and more complex plays. In the next chapter, we introduce Baseball Timeline, a visualization that enables the understanding of single play trajectory and event data in more detail.

Chapter 4

Baseball Timeline: Spatiotemporal Visualization of Baseball Plays

4.1 Introduction

The use of Play Diagrams is one of the easiest ways to explain rules, strategies and actions in sports. They are widely used in sports like basketball, baseball, football and soccer, as a clear way to represent and convey play information. The diagram is basically a top-view mapping of the actions of the targets (the players and the ball) during a certain period of time. The language and the symbols used are usually well understood among managers, journalists and fans. Although efficient, these charts were created to convey just spatial information. When several players move to a small region in the field, the visualization becomes cluttered and users cannot read the movements of the players clearly. The same happens when the diagram depicts a long play, where too many actions are packed into a single chart. In baseball, for example, all actions and movements usually happen around the bases, and many actions might happen in a single play. Figure 4.1 (left) shows an example of a diagram of a complex play, where several actions are cluttered around the third base.

The Major League Baseball (MLB) Statcast project [53] frequently shares Play Diagrams with fans in their twitter account. One example is shown in Figure 4.2 [1]. The diagram depicts the trajectory of the players and the ball during the play,

[1] `https://twitter.com/_dadler/status/893907355255156737`

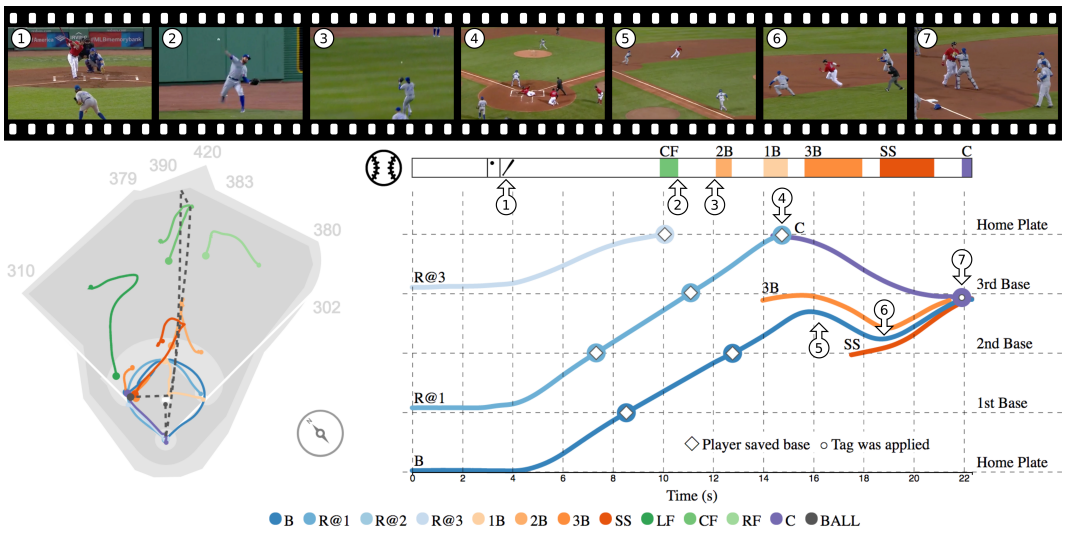


Figure 4.1: Visualization of a Toronto Blue Jays vs Boston Red Socks play. (1) Batter hits the ball. (2-3) Center fielder throws ball to second baseman. (4) Runner reaches home. (5) Rundown, batter stranded between two bases. (6) Batter changes direction. (7) Batter is tagged out.

but it cannot encode *when* each action took place. It is not possible to know if the runner reached third base, for example, or if he was tagged by the third baseman. What Figure 4.2 illustrates is that the time, and especially the relationship among the players through time, is critical for the understanding of complex situations. Since time is just implicitly represented on such diagrams, we have a limited canvas to express the information contained in the play.

The representation of complex plays usually makes use of additional techniques to overcome such limitations. Techniques like using color mappings for overlapping objects, decomposing the player actions, focusing (or zooming in) the complex parts of the play, are commonly seen on sport websites or other artistic depictions of plays. The use of animated diagrams, for example, is a technique that let the user follow the targets as they move through the play and offers a clear representation of time. An animated diagram, however, may result in adding complexity to the visualization. If the trajectories of the targets are kept in the diagram as the players move, the resulting diagram will still contain the characteristic overlapping of the static diagrams, and if they are not, they lack the elements that help the user to compare and analyze the dynamics in the play.

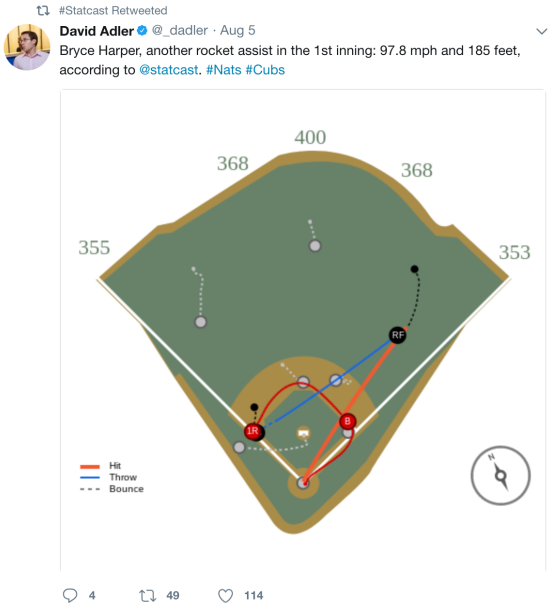


Figure 4.2: An example of the way the MLB Statcast [53] project makes use of Play Diagrams. Each play containing interesting features, like the speed of a throw, for example, is published as a diagram and a brief description of what set it apart.

In this chapter, we present Baseball Timeline, a new way to visualize baseball plays with a focus on the temporal aspect of the game. We used a requirement driven approach in order to design a visualization that is able to clearly and succinctly convey players actions and game events throughout time. We take advantage of the way the players move over the field in order to build a visual representation that encodes both position and time in a clear two-dimensional chart. Our encoding is inspired by Marey's Graph, a successful visualization that was originally proposed for the presentation of train schedules. Our technique has the advantage of providing baseball analysts with an overview of the play while enabling them to identify interesting temporal events in the chart, for example, "The 1st baseman arrived at 1st base before the batter". Figure 4.1 (right) shows a baseball play using the timeline visualization, with the X axis representing time, and the Y axis, the position in respect to the bases. In our expert interviews, we show that the use of Baseball Timeline together with a Play Diagram improves the user understanding of the events in the game, even for complex and long plays. The results described in this chapter have been published in [62].

Our work includes the following contributions:

- We present Baseball Timeline, a visualization driven approach to understanding the spatiotemporal evolution of baseball plays. We show how our design can be combined with the Play Diagram in order to create more meaningful play depictions.
- A study showing how Baseball Timeline can be used to analyze a set of six baseball plays. We highlight the insights that can be extracted from a timeline representation and how they improve the reader's understanding of the plays.
- An interview with four domain experts, who helped us validate and improve our design.

This chapter is structured as follows: Section 4.2 reviews relevant works in the field of spatiotemporal visualization. Section 4.3 presents an overview of the data and the requirements described by domain experts. Section 4.4 presents the Baseball Timeline and the design iterations we went through in order to fulfill our requirements. Section 4.5 presents use cases of our visualization applied to interesting baseball plays, and how it can be used to gain insight into the data. Section 4.6 presents the feedback we received from baseball experts on our visualization and how we improved our design based on their comments. Finally, Section 4.7 presents our conclusions, the limitations of our design and future work.

4.2 Spatiotemporal Visualization Review

The visualization of spatiotemporal data is a challenging task. The data is frequently hard to analyze in full detail, and some type of simplification has to be performed in order to display it [5, 9]. Many approaches have been used to represent both space and time in a graph, including (but not limited to) small multiples, space-time cube, clustering and time graphs. In this section, we briefly review each of these approaches. We refer the reader to Bach et al. [9] for a more detailed survey of spatiotemporal visualizations

Small multiples [85] is a popular way of displaying spatiotemporal information in charts [5]. In this approach, representative times (or *samples*) are chosen from

the data. The samples are then shown as a set of juxtaposed charts, which allows an easy comparison between different samples. The downside of small multiples is that comparing situations where quantitative differences are not very prominent is a hard task [5]. Moreover, scalability with respect to time is also a concern [57].

The space-time cube [36] is a three-dimensional representation of spatiotemporal data that places the spatial data on two dimensions, and the time in the remaining dimension. This approach was explored to represent movement data as well [39]. Although visually interesting, it might be not appropriate for identifying position of multiple elements over time, and often requires interaction, such as cube rotation, for users to make sense of the data [5]. In a recent survey of visual analytic techniques for movement, Andrienko et al. [5] suggest that the use of two displays, one mapping for spatial information, and another one for the time information, is more effective for the inspection of spatiotemporal data. Lukasczyk et al. [46] used this approach to visualize hotspot events over time: the spatial distribution was shown using kernel density estimates, while the temporal distribution was presented with an adaptation of Reeb graphs for time series.

Data clustering can be also used to simplify the spatiotemporal data before it is displayed. Andrienko et al. [6] developed an approach that groups similar spatial configurations together in order to guide the visual analysis. In their system, the trajectory data is presented using multiple views, i.e. map small multiples and time series, with clusters annotations encoded using color.

Marey's train schedule [47] is perhaps the biggest inspiration for our work. The visualization portrayed the train schedule for Paris to Lyon in the end of the nineteenth century, by using a line chart with the horizontal axis representing time, and the vertical axis representing the train stations. The stations on the chart were separated proportionally to the actual distance between them, therefore it was possible to identify train velocity based on line angle and train crossings based on line crossings [85].

4.3 Data and Domain Requirements

In 2015, the MLB Advanced Media team unveiled the Statcast project [53], a system designed for the tracking of the players, the ball and a series of high-level

game events with an unprecedented level of detail. The StatCast goal was to capture all actions performed on the playing field and process them to generate interesting content for the public. The project consists of both the hardware necessary for the tracking of targets as they move over the field (an optical and a doppler radar-based tracking system) and the software layer required to gather and process this information. The available data is extremely rich, enabling a deeper understanding of baseball games and making it possible for analysts to present interesting plays to fans in a high level of detail.

The baseball data is inherently visual, its availability was immediately followed by the research of interesting ways to visualize it. The visualization of the baseball data usually relies on mappings to convey the tracking information. Spray charts, or scatter plots that show ball landing position with respect to the ballpark, are frequently used to identify batting trends in players and teams. Meanwhile, Play Diagrams, or top-view mappings of the trajectory of the targets, are used to depict interesting situations or even entire plays. When analysts are concerned with the temporal information in the game, for example, identifying who was the first player to reach a base, they usually have to go back to the actual video footage of the play, or an animation of the tracking data. This approach to temporal analysis of games has three drawbacks: first, watching a play takes time, and the analyst might need to replay segments in order to identify the event of interest. Second, it relies on the user's memory to store the progression of the play, and they might need to rewind the video to review game events. Third, it requires the use of an electronic device in order to visualize the play, which is not ideal given that there is a lot of baseball information in static media being generated even today. Therefore, there is a clear need towards better visualization mechanisms to explore spatiotemporal information in baseball plays.

Based on these observations and on feedback we obtained from domain experts, we have compiled a list of requirements for a baseball play visualization that enables a detailed analysis of game events and player movements:

R1 Represent the entire play in a concise and clean manner: the visualization should present the information succinctly and without clutter. Moreover, it should not require much training to be understood.

R2 Identify players' actions, as well as relate those actions with the context in the play: the visualization should represent how players move in the baseball field throughout time, as well as how they interact with other players.

R3 Highlight relevant events in the play and enable the ordering of events by time: users should be able to easily see relevant events in the chart. For example, saving a base, catching the ball, or tagging a player must be easily identifiable actions.

R4 Do not require interactive content: the visualization should not require interaction or animation in order to convey game information. This requirement makes the chart easily shareable both in social and static media (books, newspapers, magazines, and brochures).

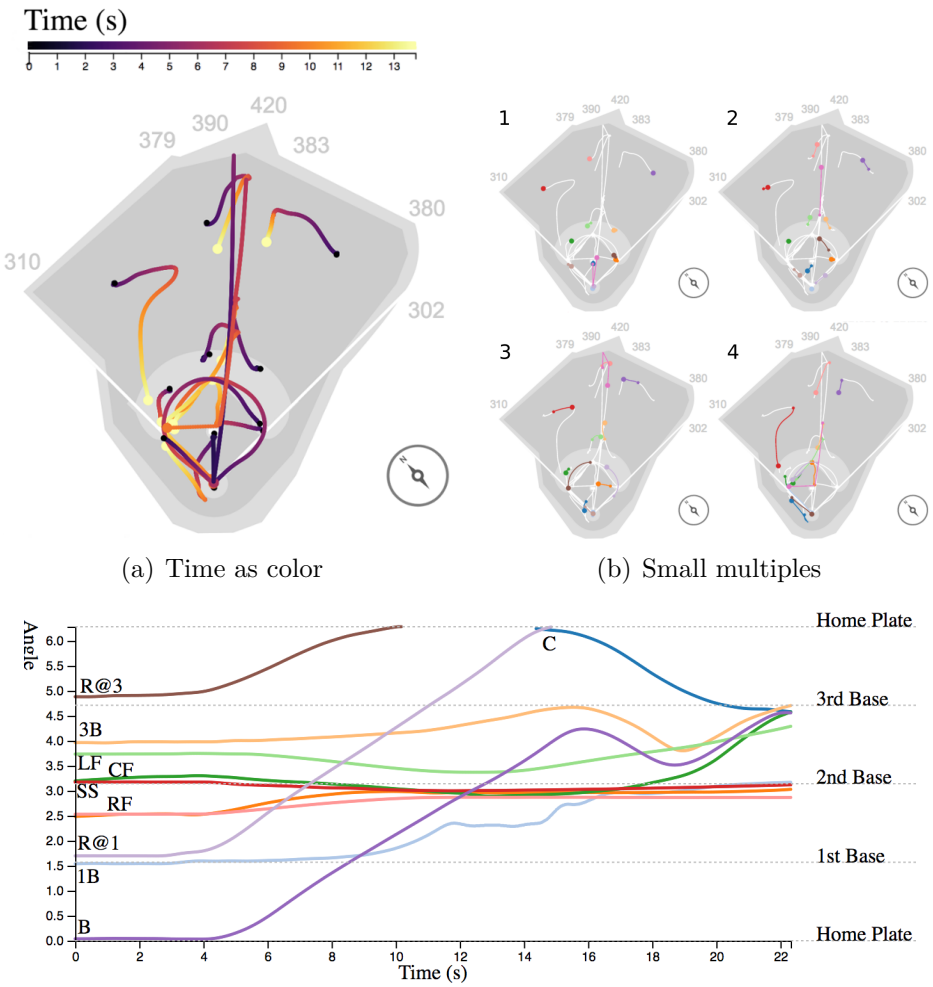
4.4 Spatiotemporal Visualization of Baseball Plays

4.4.1 Design considerations

The development of the Baseball Timeline was a iterative process, where we tested several well-known approaches for the spatiotemporal data visualization. This section presents some of these previous designs.

We started by focusing on small and incremental changes to the traditional Play Diagrams. The initial attempts were around time-based color mappings of the data and the automatic selection of interesting events for the creation of small multiples. Figure 4.3(a) shows one example of the time-as-color approach. The resulting chart is not effective at presenting the movement of players over time. As discussed by Munzner [57], color is not an expressive visual channel to encode a continuous variable. Moreover, this visualization still had problems with cluttering, especially near the third base, and did not highlight game events. In summary, the time-color encoding does not meet the requirements R1, R2 and R3.

The second approach was focused on selecting interesting game events that would delimit the elements of a small multiple visualization. Figure 4.3(b) shows an example of this approach. The visual encoding had two problems: (1) as discussed by Andrienko et al. [5], it is hard to identify small movement changes in the chart,



(a) Time as color

(b) Small multiples

(c) Bases as landmarks

Figure 4.3: Design attempts that did not meet our requirements.

and (2) it is not possible to find the exact position of a player in time, given that player position is aggregated into each element. Small multiples solve the problem with clutter (R1), since it represents a small time frame in each chart. However, one could not read player position and game events from the chart, violating requirements R2 and R3.

The initial attempts were not successful in conveying spatiotemporal information of baseball plays. We then moved to an approach where the play is taken as a time series, inspired by the work of Marey [47]. Figure 4.3(c) shows the first result of the "Bases as landmarks" approach. Time is shown on the horizontal axis and spatial

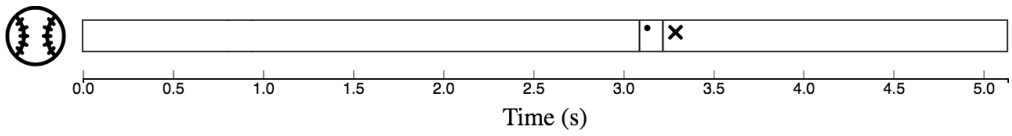
data (the position of the targets) on the vertical axis. Since most of the actions in baseball games occur close to the bases, we then took the bases as landmarks, and positioned players on the vertical axis according to how close they were to the bases. More specifically, player position was encoded as angle, with home plate being 0 radians and second base being π radians. Although this chart encodes the movements of the infielders, one could not read outfielder positions directly from it. As seen in Figure 4.3(c), fielders LF, CF and RF are all cluttered on second base (around π radians), violating requirement R1. Events were not present on the chart either, violating R3.

The "bases as landmarks" approach, although with limited success, is capable of representing the movements of the targets through time. The next session describes how we improved this approach in order to satisfy requirements R1-4.

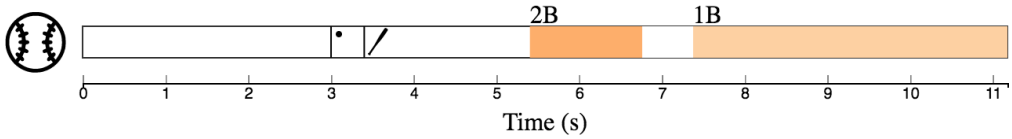
4.4.2 Baseball Timeline

In order to fulfill the requirements given by the domain experts, we designed Baseball Timeline (BT), a visualization that enables the analysis of baseball plays by providing information regarding both temporal and spatial aspects of the tracking data. Figure 4.1 shows an instance of BT. The proposed visualization consists of three views, (1) the *Play Diagram*, (2) the *Ball Status*, and (3) the *Play Timeline*. Every player and the ball are associated with a color, which is the same on all three views. We have colored the batter and runners in shades of blue, basemen in shades of orange, fielders in green and catcher in purple.

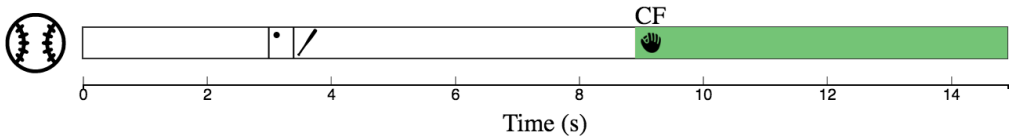
The **Play Diagram** is well-known in the baseball community. It consists of a top-view mapping of the field, with the movements of the targets encoded as polylines. This view provides a spatial summary of the play to the user and was chosen in accordance to requirements R1 and R4. The starting position of every player and ball is encoded as a small circle, and the ending position of their trajectory is encoded with a big circle. The ball trajectory is shown with a dotted line in order to differentiate it from the players trajectory. Our dataset contains metadata about the game, including at which stadium the game took place. In case a custom image of the stadium is available, it is shown in the play diagram, together with a compass that shows the stadium orientation. Otherwise, a default image is displayed. Figure 4.1 shows an example of a play in the Fenway Park, a



(a) 07/05/2015 - Los Angeles Angels @ Texas Rangers: Strike. The ball was pitched, but the batter did not hit it with the bat.



(b) 04/14/2017 - Chicago White Sox @ Minnesota Twins: Ball was batted, retrieved by the second baseman and thrown to the first baseman.



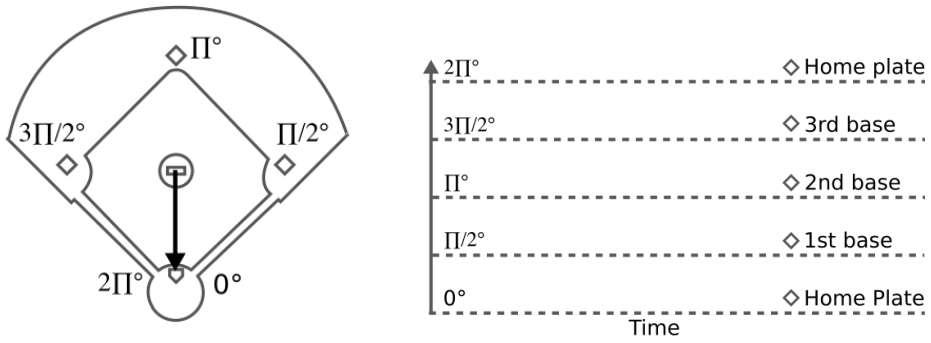
(c) 05/13/2016 - Oakland Athletics @ Tampa Bay Rays: Ball caught in the air by the center fielder.

Figure 4.4: Ball Status: this view shows the status of the ball throughout the play, from the moment it is pitched until the end of the play.

stadium in Boston, Massachusetts.

The **Ball Status** shows the temporal evolution of the ball possession during the play, from the moment the ball is pitched to the end of the play. It encodes all events related to the ball and makes clear, at any time, who has the possession of the ball. The horizontal axis represents the time, and the color of the bar, the player with the possession of the ball. Events are represented as icons on the timeline; the pitch is represented by a ball, the batting is represented by a bat, the moment the ball was caught (or gloved) as a mitt and a strike as an "x". This view fulfills requirements R2 and R4. Figure 4.4 shows three possible configurations for the Ball Status view: (1) 4.4(a) shows a strike, (2) 4.4(b) shows a batted ball and (3) 4.4(c) shows a ball caught in midair.

The **Play Timeline** shows the movements of the players in the vicinity of the bases. It consists in a two-dimensional line chart, with the horizontal axis encoding time, and the vertical axis, the mapping of the trajectories of the players. The chart is constrained to a region of interest around the area delimited by the bases, since this area encloses most of the complex interactions among baseball players.



(a) Infield (b) Timeline view

Figure 4.5: Mapping of the player position to angle. (a) Infield with angle annotations and reference vector from Pitcher's Mound to Home Plate (b) Representation of player position in the Y axis.

The position of the player is encoded as the angle with respect to the vector defined between the pitcher mound and the home plate. Figure 4.5 illustrates how this mapping is performed. In 4.5(a), we show the infield and the four bases, with their respective angles. In 4.5(b), we show how the angle is mapped in a line, with the bases annotated on the right. The home plate is represented by 0 radians, the 1st base, by $\pi/2$ radians, 2nd base by π radians and 3rd base by $3\pi/2$ radians. We encode the home plate again using 2π radians in order to present the full path of the runners through the bases without discontinuities. This encoding enables users to visualize the movement of the runners as they go over bases, as well as the strategy of the defense players, as they try to reach the runner or a base. Visual clutter is avoided by representing only players in the vicinity of the bases. However, even if all players were to be in this region, the chart would still be readable: according to Munzner [57] (Chapter 12), superimposed line charts can be understood up to a few dozen lines shown simultaneously. The two most relevant events in baseball are shown using special marks: when a runner saves a base (reaches a base safely), we encode this action by a small diamond. When a runner is tagged, i.e. touched by a defensive player that is holding the ball, we encode this action by a small circle. These events are present in Figure 4.1.

Baseball is characterized by a set of interesting plays, and among them is the Grand Slam. A Grand Slam is a home run hit with all bases loaded (all bases

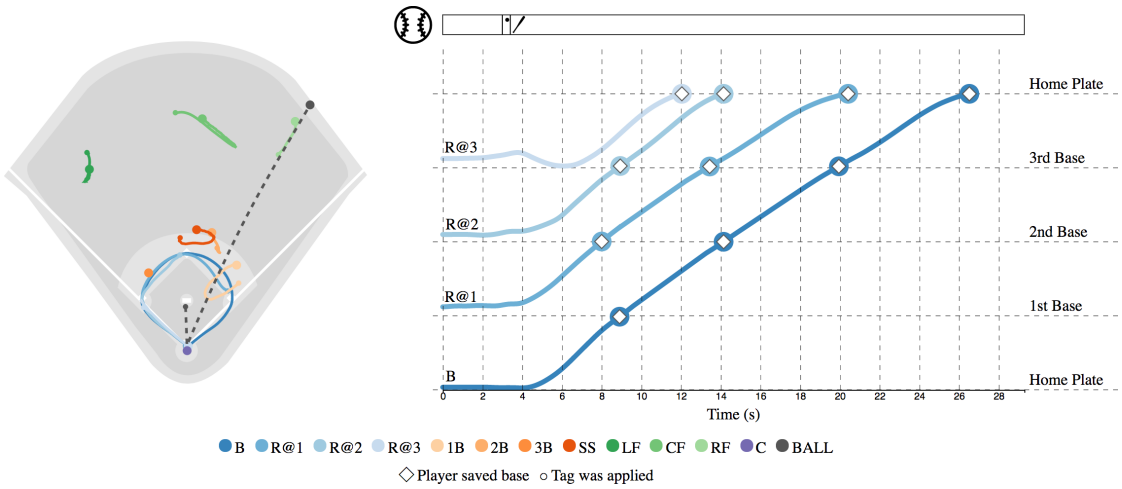


Figure 4.6: 06/03/2017 - Los Angeles Dodgers @ Milwaukee Brewers: Travis Shaw hits a Grand Slam

occupied by runners), which grants four runs to the offensive team. Figure 4.6 shows a grand slam from the bottom half of the 7th inning of the Los Angeles Dodgers and Milwaukee Brewers game (June 3rd, 2017). In this play, batter Travis Shaw hits a grand slam to right center field and runners Eric Sogard, Domingo Santana and Jesus Aguilar score. Even by taking into consideration that this might be considered a simple play, the Play Timeline highlights some interesting aspects, like the reaction time of each runner, the way they move back and forth while waiting for the landing of the batted ball, and the speeds of the runners in relation to each other. This is the type of temporal data the Play Timeline was created to highlight.

Implementation Details

Our visualization was implemented as a client-server web application. The client is written in Javascript with the library D3.js, and is responsible for rendering the Baseball Timeline chart. The server is written in Python, and is responsible for querying plays from the MLB server using the Statcast JSON API, preprocessing the data (tracking, events, as well as game metadata) and sending it to the client through a REST API. This infrastructure is ready to be deployed in order to present plays to fans and experts alike.

4.5 Analysis of Baseball Plays

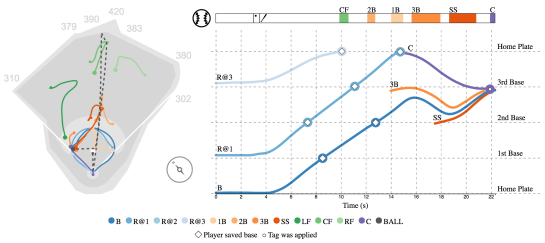
In this section, we present six baseball plays and show how they are represented in the Baseball Timeline approach. The plays are shown in Figure 4.7, in the same way they were presented during the interviews with the baseball experts, which are detailed in the next session.

[**Play 1**] The first play we present is from the bottom of the first inning of the Toronto Blue Jays versus Boston Red Sox game in April 15, 2016. The play is shown in Figure 4.1 with annotations and image footage of the game, and without annotations on Figure 4.7(a). In this play, batter Travis Shaw hits a fly ball to center fielder Kevin Pillar and safely reaches 2nd base. Runners David Ortiz and Hanley Ramirez reach home plate and score. However, Travis Shaw is stranded between 2nd and 3rd bases, and at 22 seconds in the play, is tagged out by catcher Josh Thole. This is a complex play, and can be hardly understood directly from the Play Diagram. However, all the actions become clear in the timeline view. The batter path contains a sinuous line, as he moves back and forth 3rd base, while he is surrounded by the defensive players.

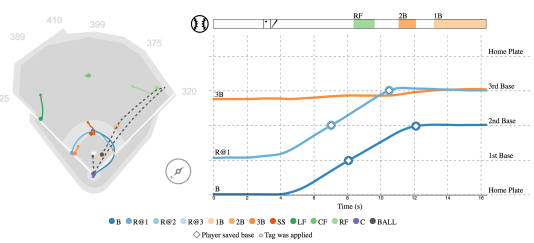
[**Play 2**] Top of the eighth inning of the St. Louis Cardinals versus Pittsburgh Pirates game (April 06, 2016). Batter Greg Garcia hits the ball in the direction of the right fielder Gregory Polanco. The fielder grabs the ball and throws to second baseman, who then assists the first baseman. Greg Garcia reaches second base safely, and the runner at first base, Jeremy Hazelbaker, reaches third base safely. Figure 4.7(b) shows how this information is conveyed by the timeline view. The actions around the 3rd base, especially, are clearly depicted through the time.

[**Play 3**] Top of the eighth inning of the Arizona Diamondbacks versus Kansas City Royals game, September 30, 2017. In this play, batter Rey Fuentes grounds out to first baseman Cheslor Cuthbert (Figure 4.7(c)). The batter is the third out, so the movement of the runner at 3rd, also depicted on the chart, does not contribute to the play. This is highlighted by the lack of the glyph at the end of the trajectory of the runner.

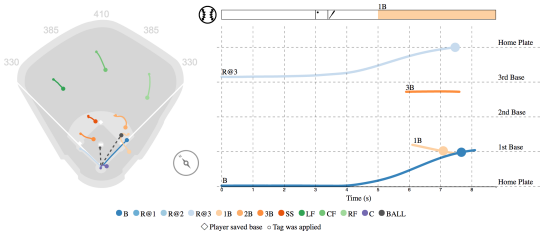
[**Play 4**] In the fourth play, on the other hand, the batter reaches first safe (top of the fourth inning of the Detroit Tigers and Chicago Cubs game, in August 18, 2015). The batter Tyler Collins hits a ground ball to second baseman Chris



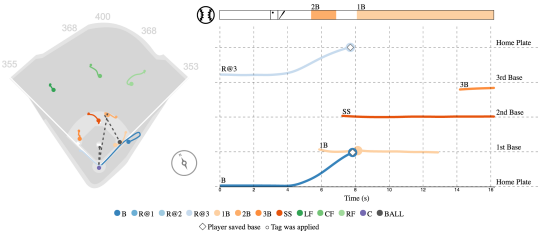
(a) 04/15/2016 - Toronto Blue Jays @ Boston Red Sox: runners David Ortiz and Hanley Ramirez score. Batter Travis Shaw is tagged out.



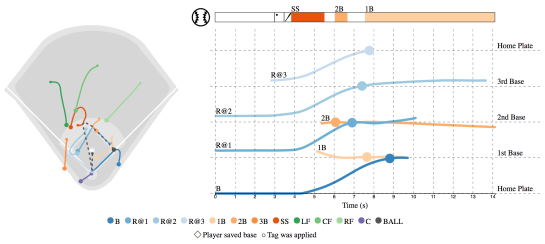
(b) 04/06/2016 - St. Louis Cardinals @ Pittsburgh Pirates: batter Greg Garcia doubles. Runner Jeremy Hazelbaker reaches third base.



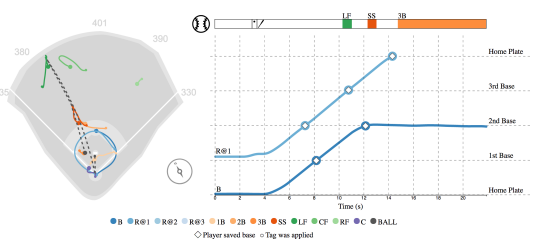
(c) 09/30/2017 - Arizona Diamondbacks @ Kansas City Royals: batter Rey Fuentes is out.



(d) 08/18/2015 - Detroit Tigers @ Chicago Cubs: batter Tyler Collins reaches first base just before first baseman with the ball.



(e) 04/15/2015 - Tampa Bay Rays @ Toronto Blue Jays: double play. Batter Rene Rivera and runner Kevin Kiermaier are out.



(f) 06/30/2015 - Washington Nationals @ Atlanta Braves: batter Danny Espinosa doubles. Runner Denard Span scores a run.

Figure 4.7: Six baseball plays used in the expert study represented by Baseball Timeline.

Coghlan, who throws it to first baseman Anthony Rizzo. The actions around the first base are clearly depicted in the timeline view, while they can be hardly understood in the Play Diagram. This play is particularly interesting, since the order of events completely changes its outcome. More specifically, the order at which players reach the base determines whether the offensive team scores or not.

[**Play 5**] The fifth example is a 6-4-3 (double play), when the defense team makes two outs during the same play (Figure 4.7(e)). This play is from the top of the fourth inning of the Tampa Bay Rays versus Toronto Blue Jays game, in April 15, 2015. The batter Rene Rivera hits a ground ball. Shortstop Jose Reyes takes the ball, pass it to second baseman Devon Travis, who passes it to first baseman Justin Smoak. Rivera is out at first and runner Kevin Kiermaier is out at second base. The timeline view shows the exact moment at which the runner and batter are out. With the timeline, we see that both the batter and the runner arrived at their bases approximately one second too late.

[**Play 6**] Finally, our last play is from the top first inning of the Washington Nationals versus Atlanta Braves game, in June 30, 2015. This play is relatively simple, with batter Danny Espinosa hitting the ball all the way to the outfield and making a double, and runner Denard Span scoring a run (Figure 4.7(f)). However, the timeline visualization can expand the reader's knowledge about the play even in simple cases: using the ball status view, we notice that Span had plenty of time to score a run and reach home plate. The third baseman only got the ball after the runner reached home, and by that time it was too late to tag him.

This section described six distinct baseball plays using Baseball Timeline. Our examples show that BT is a powerful tool to explain baseball plays without the need of animation. In the next section, we describe the feedback we received from four baseball experts on our visualization.

4.6 Domain Expert Interviews

We conducted interviews with four domain experts, two sabermetricians and two expert-fans, in order to evaluate and gain feedback on our visualization. The two sabermertricians, S1 and S2, are knowledgeable in data science and have published peer reviewed papers in the field of baseball analytics. The two expert-fans, F1 and

F2, understand the sport deeply, and have been following it for more than 10 years. These four domain experts were chosen because we wanted to see how well our visualization conveys play information to professional experts and expert-fans alike.

The interviews occurred as follows: firstly, we made sure the expert was familiar with the Play Diagram. Next, we presented our visualization and clarified any questions that they had. We showed them one play using the Play Diagram, followed by Baseball Timeline. Then, we began the qualitative evaluation of the tool. In order to perform the interview, we showed them the six baseball plays discussed in Section 4.5. For each play, we first showed the traditional Play Diagram and asked the expert to describe it. Next, we showed them Baseball Timeline, and asked them if they could identify any features that they did not notice before. We played the video footage of the play, and asked the expert if their conclusions were correct. After all six plays were analyzed, we asked the experts for feedback on our visualization. In order to steer the discussion, we asked them the following questions:

1. Did you have any difficulties learning to read the visualization? Why?
2. Did the timeline help you to better understand the play? Why?
3. Which parts of the chart do you think can be improved?
4. Can you think of any information that is not shown in the chart, but is relevant for play understanding?

4.6.1 Describing Plays

Our subjects described the six plays presented in Section 4.5 using the traditional Play Diagram followed by Baseball Timeline. Both experts and expert-fans made a few incorrect descriptions using the Play Diagram, but *described all plays correctly using Baseball Timeline*. Table 4.1 shows the correct and incorrect descriptions that our experts made using solely the Play Diagrams. We briefly describe the mistakes they made, and what caused them.

[**Play 1**] None of the participants were able to correctly describe Play 1 (Figure 4.7(a)) based on the Play Diagram. Due to clutter, S1 and F2 could not see how

many runners there were on the bases. S2 and F1 could see the runners, but they thought that the play was a triple (batter reaching third base). After seeing the timeline and ball status view, all participants described the play correctly.

[**Play 2**] All participants described Play 2 correctly with the Play Diagram. Play 2 (Figure 4.7(b)) had a very uncluttered Play Diagram. Moreover, there were no defense players protecting bases or tagging runners, therefore it was clear for the experts that both batter and runner at first were safe at their bases. The participants used Baseball Timeline to confirm their predictions.

[**Play 3**] S1, S2 and F1 described Play 3 (Figure 4.7(c)) correctly, stating that they knew the batter was out because all the players were moving out of the field when the play was over. F2, however, did not notice this fact, and could not describe the play, saying that he was not sure if the batter saved first base or not. After seeing the timeline and ball status views, F2 was able to correctly describe the plays.

[**Play 4**] None of the participants could describe Play 4 (Figure 4.7(d)) using the Play Diagram, particularly because they were not sure who arrived at base first: the batter or the first baseman. After seeing Baseball Timeline, all the experts described this play correctly.

[**Play 5**] S1, F1 and F2 described Play 5 (Figure 4.7(e)) correctly. S2, however, described Play 5 as being a ground out to second baseman after reading the Play Diagram. He realized that he was wrong after reading the timeline visualization, stating that the batter was actually safe at first base.

[**Play 6**] All participants described Play 6 (Figure 4.7(f)) correctly. Similarly to Play 2, the Play Diagram was uncluttered and it was visible that batter and runner at first were safe at second base and home plate respectively.

With this experiment, we noticed that the timeline representation is able to help readers to better understand and describe baseball plays, especially in cases where the Play Diagram was too cluttered or did not provide enough temporal information about the game events. All subjects, both experts and expert-fans, were able to correctly describe plays after seeing the Baseball Timeline representation.

	Play 1	Play 2	Play 3	Play 4	Play 5	Play 6
S1	✗	✓	✓	✗	✓	✓
S2	✗	✓	✓	✗	✗	✓
F1	✗	✓	✓	✗	✓	✓
F2	✗	✓	✗	✗	✓	✓

Table 4.1: Play descriptions *without* using Baseball Timeline (✓- correct description, ✗ - incorrect description). *With* Baseball Timeline, S1, S2, F1 and F2 described every play correctly.

4.6.2 Expert Feedback

Overall, the experts liked Baseball Timeline and did not have any difficulty understanding or reading it. All subjects stated that the timeline chart helped them to better understand the plays. S1 and F1 appreciated the fact that they could see interesting properties of the play with the timeline approach, such as player speed and reaction time based on the slope of the line. F1 also made positive comments regarding the ball status view, saying that determining the player in possession of the ball was hard using the Play Diagram, but this information was easy to see in the ball status chart. S1, S2 and F2 mentioned that the Play Diagram and the timeline complement each other, because while the first described spatial information in detail, the second showed baserunner and ball events which are hard to read on the traditional diagram.

We also received feedback on how to improve our visualization. Originally, we used the d3 color scheme "category20" to encode player positions. However, that resulted in similar player positions having vastly distinct colors. For instance, while the second baseman was represented by orange, the third baseman was represented by green. S2 and F1 suggested that we used similar colors for similar players, therefore we changed the color scheme to d3 "category20c", and that resulted in a more consistent color encoding: batter and runners are represented by shades of blue, basemen are encoded by shades of orange, and outfielders, by shades of green.

The second suggestion we received was regarding the choice of players to be shown in the timeline visualization. Originally, these players were chosen based on a threshold of the distance to the bases in the field. More specifically, we chose only players that moved within 10 feet of the square with corners at the 1st, 2nd, 3rd

bases and the home plate. However, S1 and S2 suggested that even if the player moved close to the square, he might not do anything useful in the play. Therefore, in our prototype, we kept the distance threshold approach, but we also enabled the user to manually show or hide players as they see fit in order to create a better chart. This feature helped to reduce clutter in the visualization and made it more easily readable. Note that this interaction does not violate requirement R4: once the analyst is satisfied with his Play Timeline, the chart is static and therefore it can be printed and distributed in static media.

4.7 Final Considerations

This chapter presented Baseball Timeline, a visualization that represents how baseball plays evolve through time. Using BT, we were able to translate complex baseball plays into a two-dimensional graph that is easy to understand and conveys both the temporal and the spatial aspects of the game simultaneously. We showed six use cases of our technique and described how it helps users to understand the evolution of the play throughout time. We also performed interviews with both sabermetricians and baseball fans, who were able to read and describe the plays quickly with our visualization. While the Play Diagram is the industry standard for visualizing baseball plays, we realized that it is not the most effective way for doing so. More specifically, we have shown cases where experts were not able to describe plays using the Play Diagram, but were successful in doing so with Baseball Timeline. The source code for this visualization is available at `https://github.com/jorgehpo/BaseballTimeline`.

Baseball Timeline was tailored to show the spatiotemporal evolution of plays near the bases. However, it cannot represent the movement of players in the outfield, as the mapping of angle to position is not invertible. More specifically, distinct positions in the outfield may correspond to the same position in the timeline view, which results in visual clutter. Another limitation of Baseball Timeline is that it relies on Statcast data, which may contain missing information about the game. One example of missing information are errors in passes between players, for example, when the right fielder throws the ball to the first baseman, but he misses. Such information is not available in the Statcast description of the game;

therefore we cannot display it in our visualization currently. Should more events be available in the future, we can display them using special marks in the Play Timeline, similarly to what is done with the events "player reached a base" and "player was tagged". Since the number of events in a baseball play is limited, and they are spread through time, we do not expect visual clutter to be an issue.

This chapter focused on the description of baseball plays, emphasizing the temporal events that happen during the game. We took advantage of the runners' movement patterns and used a visual encoding similar to Marey's Graph to convey spatiotemporal information about the play. The following chapter extends this work and discusses a sports agnostic visualization for sports trajectory data.

Chapter 5

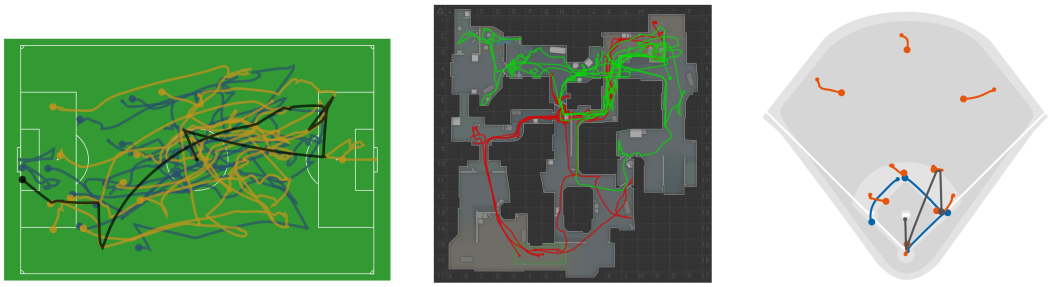
TrackRuler: Sports-Agnostic Visualization of Trajectory Data

5.1 Introduction

With advancements in tracking technology, every major league sport can acquire dense spatiotemporal data from its games. Highly accurate measurements of players and ball locations empower experts and fans alike to analyze sports with unprecedented detail. For example, they can now compute statistics on demand, evaluate player performance, improve strategies, and even prevent injuries [2, 18, 22, 27, 99]. Data Visualization is an essential tool for exploratory trajectory analysis, leading to several recent developments.

Tracking data are often visualized with Play Diagrams - an overhead view of the playing field mapping player movements to a static graph. While well-understood by sports experts and fans, Play Diagrams have two limitations: (1) they are hard to read when player movements overlap, and (2) they cannot convey temporal aspects of the underlying trajectories, which are essential for a complete understanding of the play [62]. Figure 5.1 illustrates Play Diagrams across three sports: (a) soccer, (b) Counter-Strike and (c) baseball, with especially pronounced clutter in (a) and (b). Temporal dependencies between events cannot be assessed via these diagrams, further impeding play understanding.

To address these limitations, other visualization strategies such as animation [27, 98], small multiples [17, 71], and time diagrams [62, 96] have been proposed.



(a) Soccer (24 seconds). This play contains both attack and defense from both teams, Orange and Blue, resulting in players changing direction and a cluttered diagram.

(b) Counter-Strike (1 minute and 28 seconds). It is not clear what team reached the bomb site first, Terrorists or Counter Terrorists?

(c) Baseball (9 seconds). It is not clear if the batter is safe at base. In other words, who arrived at base first, the batter or the first baseman?

Figure 5.1: Play Diagram illustrations for soccer, Counter-Strike, and baseball highlighting how their temporal sequentiality (the time and order in which events happen) are critical to play understanding.

Animated diagrams offer a clear representation of time and let users follow the targets as they move through the play. However, keeping track of multiple player movements across time makes their use cognitively burdensome. Further, animations do not allow for compact static play representations. Small multiples visualize trajectories at different time frames but lose information in the process of temporal aggregation. Finally, time diagrams compress planar 2D data into a one-dimensional line-chart representation with time as the x-axis. While time diagrams fully preserve temporal aspects, spatial information is lost. Further, time diagrams visualize sports-specific metrics such as the distance to the bases in baseball [62], and cannot be easily transferred across sports.

This chapter presents TrackRuler, a framework for visualizing sports tracking data that maintains spatial and temporal viewpoints. TrackRuler is developed alongside experts in three domains (soccer, baseball, and Counter-Strike) to better address their needs and provide real-world validation of our design. TrackRuler combines the advantages of two visualization approaches (small multiples and time diagrams) to enable rapid temporal play understanding. Shared time and spatial axes alongside cross-filtering interactions facilitate the association of temporal and spatial dimensions. We use a ruler metaphor to roughly measure the time passed between player actions using the time axis. Figure 5.2 presents TrackRuler

analyzing a soccer play, where spatial information is visualized in detail using small multiples (B), and temporal information is conveyed via the time diagram (C). The shared axis between the two views (dashed line in C3) allows for the association of a spatial event (e.g., the ball reaching a location) and the time at which this happened (e.g., at 18 seconds). An event view aligned with the time axis encodes additional play information (D). This view is relevant because sports trajectory data often accompanies an event dataset, which can augment spatial information with player and game events. For example, in (D), the ball possession of a soccer play is shown.

Contributions. Our contributions are three-fold:

1. We develop novel methodologies for analyzing team sports trajectories that enable the visualization of both spatial and temporal aspects of the play without assuming the sport represented.

2. We propose TrackRuler, a tool for spatiotemporal visualization that can be used as a standalone chart and shared amongst stakeholders, published online alongside an open-source Python library enabling convenient usage and adoption by the community.

3. We present three use cases that show the analytical power of TrackRuler and the types of insights that can be gathered by visualizing trajectory data in detail: we better understand game outcomes, identify players' mistakes, and analyze competing strategies.

The rest of the chapter is organized as follows. Section 5.2 describes the sports data and the requirements described by the domain experts. Section 5.3 describes our original designs and failed attempts. Section 5.4 presents our visualization methodology the design choices we made to implement it in TrackRuler. We present use cases and insights in Section 5.5 and the expert feedback in Section 5.6. We discuss our results and limitations in Section 5.7 and present our conclusions in Section 5.8.

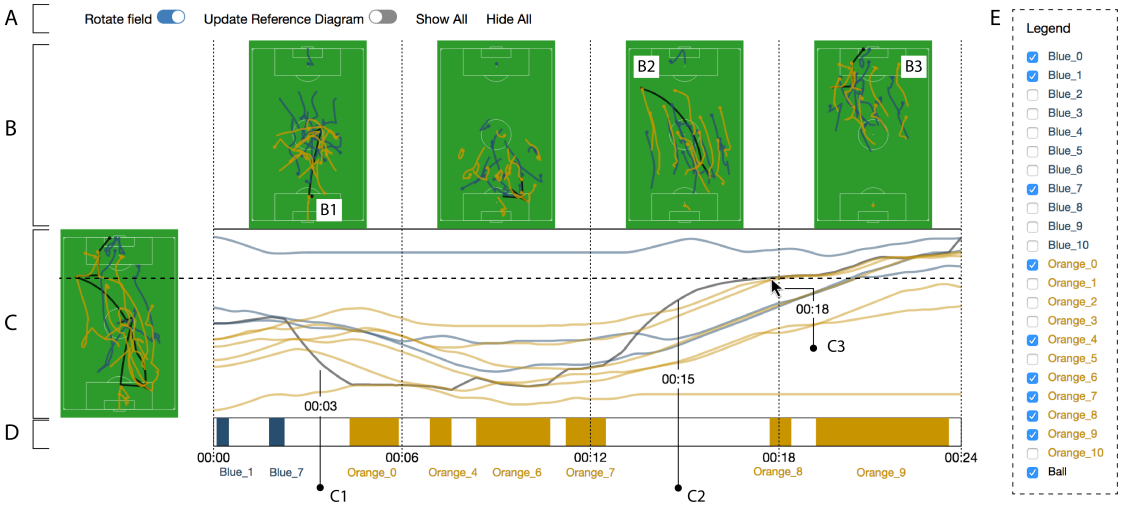


Figure 5.2: TrackRuler visualization of a soccer play. A) Visualization menu. B) Play Small Multiples, showing time slices of the play aligned with the time axis. C) Projected Timeline, showing the projected Y axis over time. D) Event View, containing ball possession events. E) Legend, with the option to hide players from the timeline (checkbox). The Blue Team starts the play in the attack, but makes a bad pass and the Orange goalkeeper takes the ball (B1, C1). The ball is passed between the Orange Team players and kicked across the field (B2, C2). The Orange team attempts a shot but misses the goal (B3). A horizontal guideline is shown when the user moves the mouse over the Projected Timeline. This dashed line is aligned with the reference Play Diagram (C3).

5.2 Data and Domain Requirements

Game data is customarily split into plays, or short segments of a few minutes, containing an initial setup, a development, and a conclusion. For example, a baseball play starts with the pitch and ends with the ball reaching the pitcher's glove or out of play. A soccer play starts with ball possession and ends with a goal or the ball out of play. In CSGO, the game is structured as a series of rounds. The rounds are recorded as a series of "frames", which describe the state of the game at every second.

Sports trajectory data often consists of complex and high dimensional time series describing game element positions and events across time. In this chapter, we analyze data from three sports: baseball, soccer, and Counter-Strike Global Offensive (CSGO), a popular esport. For baseball, we use ten plays manually annotated from MLB game videos using HistoryTracker [68]. We use the Google Research Football reinforcement learning environment [40] to acquire 11 soccer plays. This environment allows us to simulate a soccer game between two trained agents. For CSGO, we use the game parser from Xenopoulos et al. [96], and we parse the July 13th, 2020 match between the teams forZe and Endpoint, which is a publicly available game. This match contains 20 plays. Each data set contains the 2D coordinates of players on the playing surface. Additionally, for baseball and soccer, the ball location is also available. Events, such as hits or throws in baseball, passes in possessions in soccer, or kills or bomb plants in CSGO, are also included.

Throughout the development of TrackRuler, we have worked closely with five sports experts and expert fans in three domains: soccer (S1 and S2), Counter-Strike (C1, C2), and baseball (B1). S1 and S2 are expert fans: they are amateur players that have followed soccer for more than twenty years and played the sport in college. C1 is a CSGO expert that has followed competitive Counter-Strike for seven years and played non-competitively for longer. He is a data scientist that is currently building a commercial app for quick retrieval of CSGO plays. C2 is an expert fan that has played non-competitive for more than ten years and has followed competitive games for the past two years. B1 is a researcher that has five years of experience with baseball data analysis. He works with sports tracking and video data, particularly on how to extract knowledge from baseball video footage. All

experts have programming knowledge and have used our prototypes in the Jupyter Notebook environment. The experts were recruited by email.

All experts were interested in looking at sports trajectories to analyze them in detail. S1 and S2 said they used soccer video footage to understand what happened in previous plays. They complained that the video footage did not always present the entire field, sometimes focusing on a particular player or groups of players. C1 and C2 frequently watch plays from a first-person perspective (the Counter-Strike system can record games and replay them afterward as animation from one player's perspective). They mentioned that Counter-Strike players could take many actions, such as eliminating other players and planing bombs. They would like to see such events as well. Finally, B1 analyzes baseball plays both from videos and tracking data using Play Diagrams. Based on our observations and discussions with domain experts, we have gathered a list of requirements for a general purpose sports trajectory visualization.

[**R1**] Be sports-agnostic: while there is a plethora of sports visualization available, they are all tailored for a particular sport. Having a shared representation across sports can facilitate the sharing of insights and improve visualization literacy in sports analytics.

[**R2**] Describe the entire play (S1, S2): our visualization should provide a summary of the entire play, from start to end. We strive to follow the visualization mantra, providing an overview first, interaction, and details on demand.

[**R3**] Represent the game elements' trajectories across time (S1, S2, C1, C2, B1): the reader should be able to identify the players and ball positions at any time.

[**R4**] Highlight relevant events (C1, C2): sports tracking data often contains events, player actions that are not directly encoded in the trajectory data but affect the game in some way. We want this information to be directly encoded in the visualization.

5.3 Design Considerations

TrackRuler was designed in an iterative process, where we tested four well-known spatiotemporal visualizations: animated Play Diagram, space-time cube, small multiples, and time diagram. In this section, we present these designs, their advantages, and limitations. Figure 5.3 shows our previous designs. All examples use the same Counter-Strike round to facilitate the comparison. The previous designs have focused on the representation of trajectory data. The event data [R4] can be directly encoded using a glyph timeline (Figure 5.4), similarly to [62].

Animated Play Diagram. As a baseline for our trajectory visualization system, we implemented an animated Play Diagram that lets users scrub through the trajectory over time (Figure 5.3(A)). Animated Play Diagrams can show a single point in time, or keep a trace of the previous positions. However, both approaches have limitations. When using the single point in time approach, the diagram completely relies on human memory to analyze the dynamics of the play. On the other hand, if the trajectories of the targets are kept in the diagram as the players move, the resulting diagram will still contain the characteristic overlapping of the static diagrams [62]. Our implementation lets the user choose the trace amount, i.e., how long to keep the previous positions in the visualization. In our discussions with the sports experts, we have noticed that as the plays grew longer and more complex, understanding them using animated Play Diagrams required a lot more effort and interaction (scrubbing through time).

Space-Time Cube. The space-time cube [39] is a popular spatiotemporal visualization. The visualization consists of a 3D line chart that encodes spatial data on the first two dimensions and time on the third, encoding the entire spatiotemporal data without any aggregation or data transformation. Figure 5.3(B) shows an instance of this approach. Our sports experts had trouble using this visualization, particularly for finding the player position in a particular time instant. Interactions such as rotation and zoom helped in this task, but they required significant effort from the user.

Small Multiples is a visualization that can maintain the actual layout of the data space while encoding temporal information. Our implementation used Play Diagrams of equally spaced time intervals, which were aligned on top of a

time axis to facilitate the understanding of the temporal evolution of the play (Figure 5.3(C)). The user could control the level of aggregation (length of the time interval). Our experts found that while small multiples could effectively summarize the play's temporal dynamics, fine-grained temporal information was lost due to the aggregation.

Time Diagram. In order to address the concerns about the temporal aggregation in the small multiple views, we directed our attention to time diagrams. Instead of using the distance to a reference point like in [24, 62], we decided on plotting a single axis of the trajectory data against time. Our approach had two views: a play diagram and a time diagram (Figure 5.3(D)). Both views had a shared Y axis, which facilitated the reading of the spatial evolution of the plays. Our experts liked using this visualization in short plays but noted that the time diagram became confusing in longer plays.

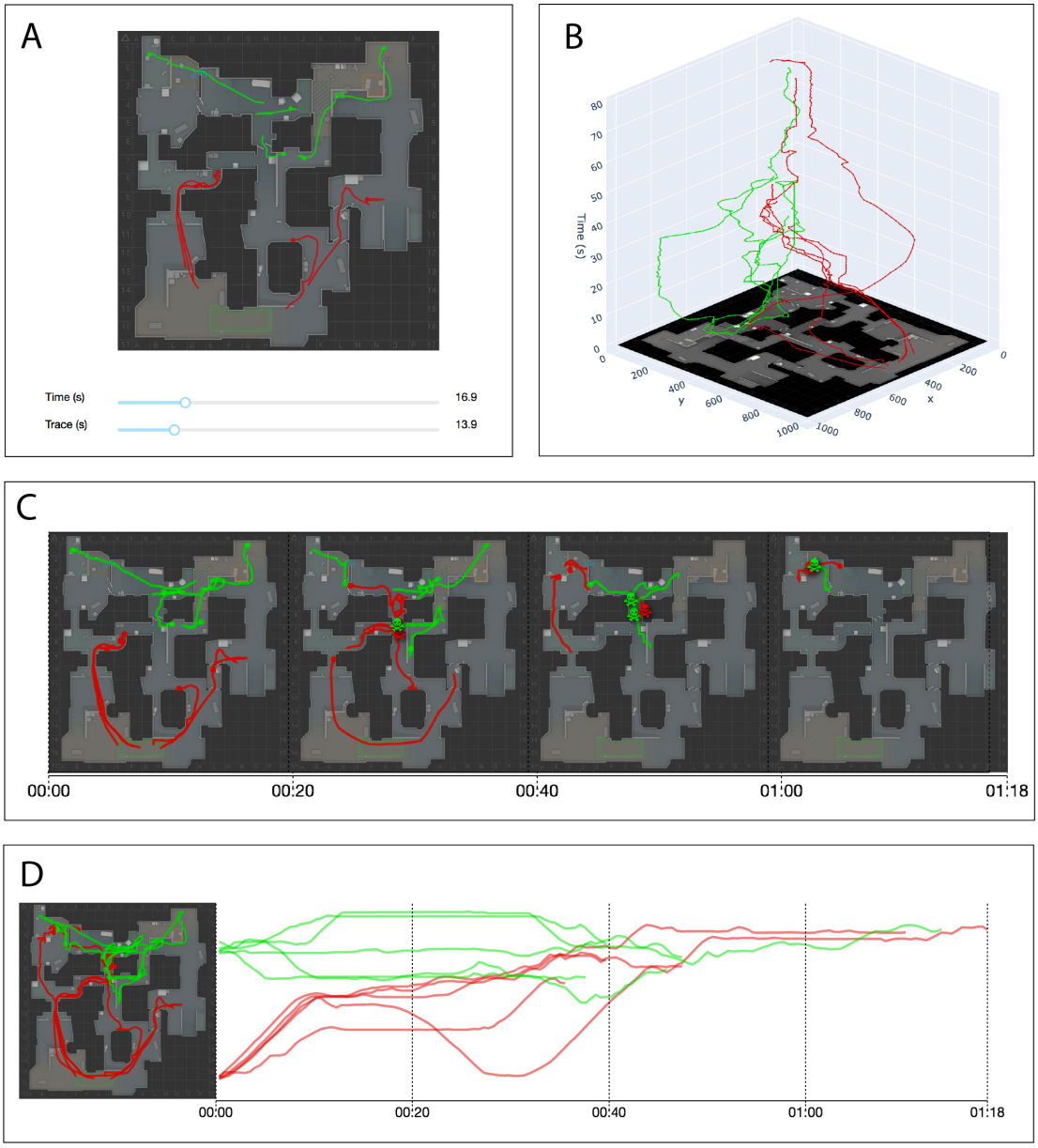


Figure 5.3: Previous designs of the spatiotemporal visualization. A) Animated Play Diagram. B) Space-Time Cube. C) Small Multiples. D) Time Diagram

5.4 TrackRuler

In order to fulfill the requirements described in Section 5.2, we designed TrackRuler, a sports-agnostic spatiotemporal visualization for team sports. Figure 5.2 shows an example of the visualization for a soccer play. TrackRuler consists of three views aligned in a time axis: the Play Small Multiples, the Projected Timeline, and the Event View. The Play Small Multiples (B) show the trajectories of the players in time slices delimited by the temporal axis. Each time slice contains 5 seconds of play, except for the last slice, containing 4 seconds. The Projected Timeline (C) shows a Y axis projection of the trajectories over a time diagram. A reference Play Diagram is shown on the left to guide the exploration. When the user hovers the mouse over the Projected Timeline, a horizontal dashed line extends to the reference play diagram to facilitate the analysis (C1). The Event View (D) shows relevant events for the play. In soccer, those events are ball possessions and pass between players (represented as interval bands). The user can control system parameters in the Visualization Menu (A). Finally, the Legend (E) allows users to hide or show players using a checkbox. We discuss this soccer play in detail in Section 5.5.

The Event View represents game events using icons, similarly to the encoding in Baseball Timeline [62]. However, in TrackRuler, we generalize the types of events that can be used in the visualization: events can be of two types: *instant events* or *interval events*. *Instant events* have an associated icon, the time of the event, a description, and the associated game element (player or ball). *Interval events* are represented using bars and have an associated color, the time and duration of the event, a description, and an associated game element. When users hover the mouse over the event in the Event View, a textual description of the event is shown using a tooltip. Figure 5.4 shows examples of Event View instances for three sports: baseball, soccer, and Counter-Strike. The Event View can accept custom image glyphs and event descriptions, making it customizable to any sport.

The Play Small Multiples view shows Play Diagrams of equally spaced time segments of the play. This approach has several advantages: it is a standard encoding of the trajectories to the reader; reduces clutter in the visualization, since only part of the trajectory is represented; and summarizes complex movement

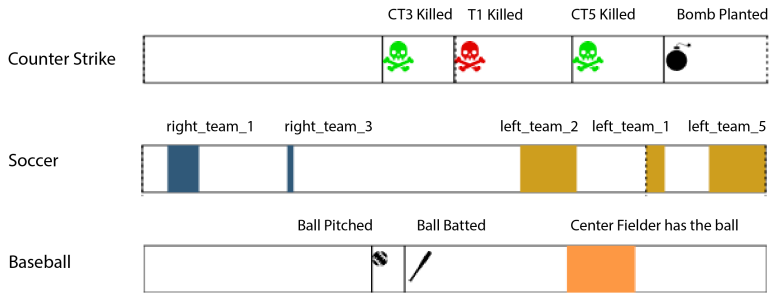


Figure 5.4: Event View showing example events for games of Counter-Strike, Soccer and Baseball. Events can be instant (kills in Counter-Strike, ball batted in baseball) or intervals (ball possessions in soccer and baseball).

patterns in easy-to-digest intervals. However, it has two drawbacks: first, while small multiples make it easy to identify movement trends (e.g., players are moving up), visualizing individual movement patterns becomes more challenging because the element trajectory is not aligned across small multiples. Second, users cannot identify the exact moment a particular event happened in the small multiples. Instead, they can only see the associated time interval for the trajectory slice. To mitigate the first limitation, we implemented the highlight interaction. When the user clicks on a game element trajectory, we highlight this game element and gray out the others, facilitating the reading of the trajectory lines. To address the second challenge, we propose the use of the Projected Timeline.

The Projected Timeline is a time diagram that presents positional information of game elements across the Y axis of the field. The timeline has two views: a Play Diagram and a Time Diagram, both sharing a common Y axis. When the user hovers the mouse over the Time Diagram, the associated Play Diagram time slice is shown as a reference. The Projected Timeline helps the user to identify the exact time of an element's movement. To demonstrate the power of this visual encoding, we use a synthetic example: Figure 5.5 shows the Projected Timeline of a semi-circular path. Using this view, we can understand the dynamics of the trajectory; for example, we see that the trajectory is twice as fast on the 2^{nd} quadrant as it is on the 1^{st}. In Figure 5.2(C1), we can see the exact moment the ball crosses the field. Furthermore, we can see movement trends: the Blue Team is on the attack on the first 5 seconds of the play (players are moving down), but they go to the defense afterward (move up to try to recover the ball).

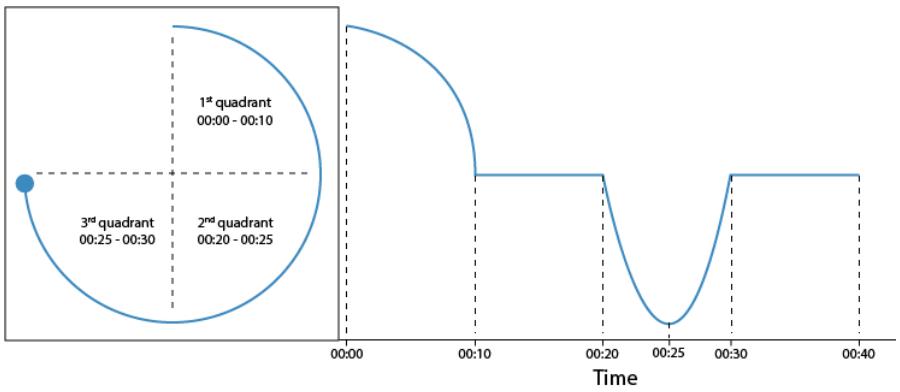


Figure 5.5: Projected Timeline of an artificial trajectory following a semi-circle. Using the timeline, we can quickly identify the temporal progression of the trajectory. 0-10s: 1^{st} quadrant. 10-20s: stopped moving. 20-25s: 2^{nd} quadrant. 25-30s: 3^{rd} quadrant. 30-40s: stopped moving.

The TrackRuler visualization satisfies all of our requirements. Because the visualization components work for generic time-series data, it is sports-agnostic [R1]. TrackRuler presents the data's spatial, temporal, and event dimensions for the entire game [R2] across time [R3]. The game events, paramount for describing game context, are highlighted in the Event View [R4].

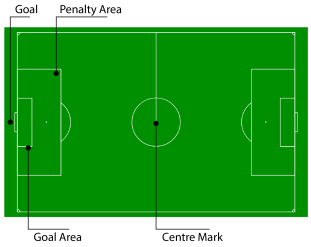
5.4.1 Interactions

Several interactions are available in TrackRuler:

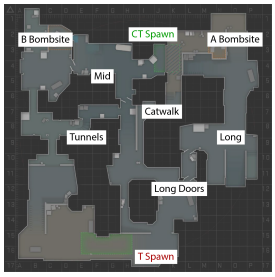
Orientation. The user can control the orientation of the play diagrams (Figure 5.2(D)). This interaction is useful for adjusting the Projected Timeline so that the movement we want to investigate is most discernible in the Y axis. For example, if most movements happen on the X axis, rotating the chart by 90 degrees will reduce clutter and improve the understanding of the projected view.

Time splits. The user can control the size of the time splits in the Play Small Multiples. This interaction controls how summarized or detailed the visualization will be. It can also be interpreted as a zoom operation in the different regions of the trajectory.

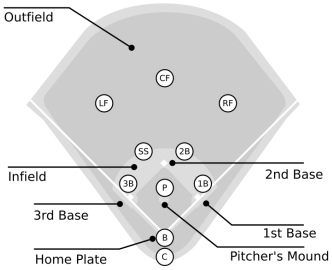
Reference Guideline. When the user hovers the mouse over the Projected Timeline, a horizontal dashed line extending until the reference Play Diagram is shown (Figure 5.2(C1)). This feature facilitates the reading of the Projected



(a) Schema of a soccer field. The positions of the Goal, Goal Area, Penalty Area and Centre Mark are annotated on the left side of the field.



(b) Annotated area locations in the *dust2* Counter-Strike map. CT spawn in green, T spawn in red.



(c) Schema of the baseball playing field. We annotated the positions of the four bases and the starting positions of the players.

Figure 5.6: Annotated playing fields for soccer, Counter-Strike and baseball.

Timeline, as the Y axis is shared by both play diagram and timeline views. The user can choose if the Reference Diagram will be updated based on the mouse position using the "Update Reference Diagram" toggle button (Figure 5.2(A)). If this option is active, the Reference Diagram will show the small multiple corresponding to the mouse position (Figure 5.8(E)).

Element Highlight and Removal. The user can click on a game element (players and ball) to highlight it. When an element is highlighted, all the other elements are grayed out (Figure 5.7 (A)). This interaction facilitates the understanding of individual trajectories. The user can also remove elements from the Projected Timeline by unchecking the element name in the chart legend (Figure 5.2(E)).

5.4.2 Implementation Details

TrackRuler was implemented in Javascript with the libraries React [33] and D3 [14]. Because the sports experts were data scientists familiar with the Python programming language, TrackRuler can be shared as a Python library for the Jupyter environment [37]. The library also enables users to export a self-contained HTML visualization that can be shared with stakeholders.

Currently, TrackRuler supports three sports: soccer, Counter-Strike, and baseball. However, because our encodings do not assume the data source, our visualization can be directly extended to other domains. The plotting function *plot_trackruler*

takes as input the background field (SVG or PNG), the field dimensions (width and height), color mapping (the color of the game elements), the trajectory data, and a list of events in the JSON format.

5.5 Use Cases

We asked the sports experts (Section 5.2) to investigate play trajectory datasets using TrackRuler. They reported their insights to us via videoconferencing and a shared computer screen. All experts explored TrackRuler visualizations in two stages: first, they looked at the Small Multiples and Event View to get an overview of the entire play. They found intriguing temporal regions in the diagram and investigated them in more detail using the Projected Timeline. In this section, we present three of these analyses.

5.5.1 Finding Mistakes in a Soccer Play

Soccer is a ball game played by two teams of 11 players each. The game is played in a rectangular field with two opposite goals. The rules of soccer are straightforward. Players are allowed to move freely in the field. Teams score points (goals) by kicking the ball inside the opposite team's goal. By the end of the game, the team with the most points win.

Each team has a goalkeeper, a player whose task is to protect the team's goal. Players cannot touch the ball with their hands or arms, except for the goalkeeper, who is allowed to do so when inside the Penalty Area. Another important region is the Goal Area, where the goalkeeper can free-kick (kick the ball to restart a play). Figure 5.6(a) shows a schema of the soccer field, with the main areas annotated. For more details about the game, please see [31].

Figure 5.1(a) shows a soccer play synthesized by the Google Research Football reinforcement learning environment [40]. If we look at the ball (black trajectory line), we see that it moved towards both left and right goals (both teams played offense and defense). Because players switched directions, the Play Diagram becomes cluttered and hard to read.

In order to better understand what happened in this play, we analyze it with

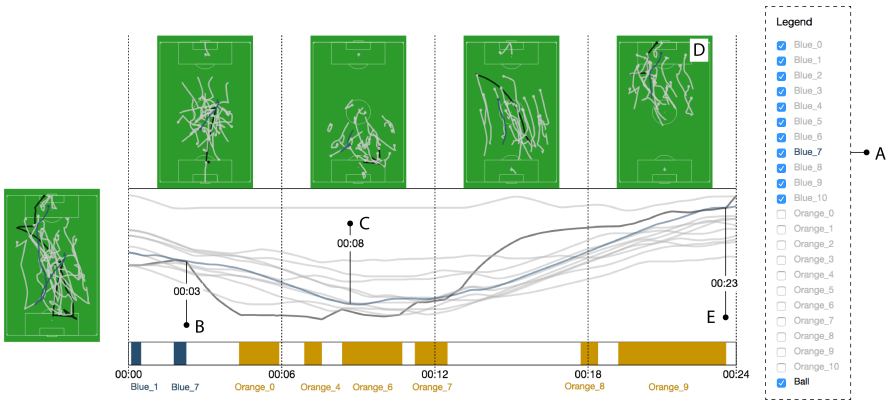


Figure 5.7: Analysis of the same soccer play from the perspective of the Blue team (defense). (A) Hiding Orange team and highlighting Blue_7. (B) Player loses the ball. (C) Player tries to recover the ball near the Penalty Area. (D) Player runs to the opposite side of the field.

TrackRuler (Figure 5.2). We hide the players that did not possess the ball using the Legend view. We can identify several exciting insights for this play: the Blue Team starts in the offense, but Blue_7 makes a mistake in the ball pass and loses it to the Orange goalkeeper (Orange_0) (B1). We can see the temporal evolution of this pass in (C1). The Orange team takes control of the ball, and a series of passes happen between 6 and 12 seconds of the play. Orange_7 kicks the ball across the field (C2), and the ball reaches the receiving player (Orange_8) at 18 seconds (B2, C3). Finally, Orange_9 attempts a goal but misses.

We can also analyze this play from the perspective of the Blue team (Figure 5.7). We hide the Orange team using the Legend view, highlighting the Blue_7 and the ball (A). Something interesting happens here: we have seen that that Blue_7 makes a bad pass (B) and loses the ball to the Orange goalkeeper. However, we see that this player made a big effort to recover the ball. He runs to the Penalty Area (C) to try to take the ball from Orange_4, but fails and runs to the opposite side of the field (D). At the end of the play, Blue_7 was the defensive player closest to the ball (E). The expert inferred that this player was trying to make up for losing the ball at the beginning of the play and put more effort into trying to recover it.

5.5.2 Analyzing Strategies in Counter-Strike

Counter-Strike is a popular esport, where two teams of five players each fight to complete various objectives. The two teams play on a pre-selected *map*, which is a virtual world. One team is assigned to the Terrorist (T) side, and the other to the Counter-Terrorist (CT). The two teams switch sides after 15 rounds, and the first team to 16 rounds won wins the map. The T side can win the round by planting a bomb and having it explode, or by eliminating all CT players. The CT side can win by defusing the bomb or preventing the T side from planting the bomb, either through running out the round time (roughly two minutes) or eliminating all T players. For our case study, we use the popular map *dust2*. We provide an annotated version of *dust2* in Figure 5.6(b), which contains labels of named areas of the map.

Counter-Strike as a game often has periods of a round where players may not move at all. Although players may not be moving, they perform valuable functions as they watch over vast swaths of the map or hide from opponents to draw them out of their position. This behavior is tough to spot in a play diagram, as we lose any sense of time. Furthermore, trajectories can easily overlap, obscuring crucial movements. Again, using Figure 5.1(b), we lose track of the T side trajectories around the mid and catwalk areas of the map.

Figure 5.8 shows the TrackRuler visualization for the aforementioned round. We are interested in analyzing the movement of the T side, so we deselect the CT players. We see that four T players head towards tunnels and then start to move towards mid. While doing so, one T player stays near long doors. After this initial movement by the T side, we see that they all stay stationary (A). This is an intriguing play, as teams typically spread out across the map to gain information on where the CT players are defending.

It is clear what the T side does after the 1-minute mark. The T player `MiGHTYMAX` pushes mid and gets eliminated by a CT player. The other T players begin to push the catwalk, and `srrl` is eliminated (B). With the remaining three players, the T side eventually makes it to the *A Bombsite* and plants the bomb, which we can infer not only from their movement but also from the event timeline (C).

If a CT team attempts to defuse a planted bomb, they first must *retake* the site from the T side. The retake is an essential maneuver in Counter-Strike, and

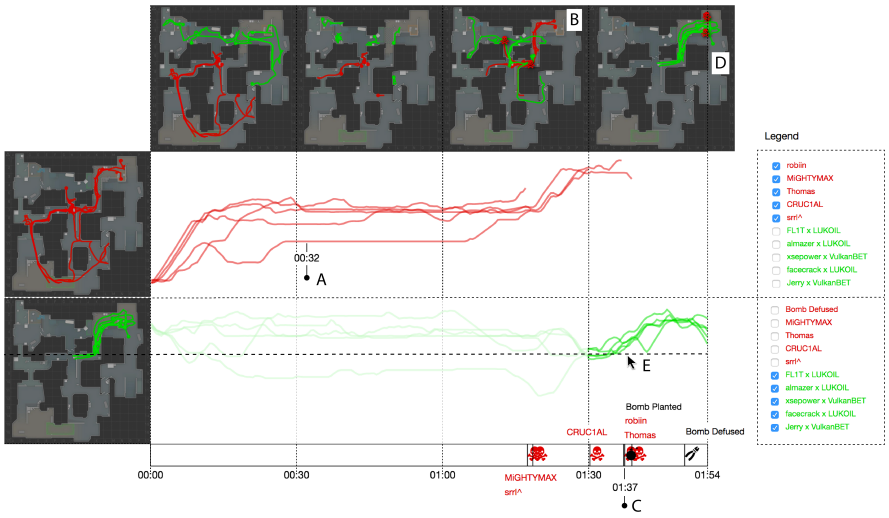


Figure 5.8: TrackRuler representation of a Counter-Strike match between the teams forZe and Endpoint. (A) T players remain stationary. (B) T players begin to push catwalk. (C) T Players reach *A Bombsite* and plant a bomb. (D) CT players start to retake the A bombsite. (E) CT movement, from the catwalk to bombsite. The Reference Diagram is updated to only show the time segment from 1:30 - 1:54.

is subject to analysis from experts. Oftentimes, teams may attempt to retake a bombsite from multiple directions to overload the defending T side. However, from the projected timeline, we see that all CT players start to retake the A bombsite from the same part of the map: catwalk (D, E). This makes sense, as all of the CT players are closest to the catwalk. First, these CT players eliminate the T player `CRUC1AL`, whose death location implies that he was defending the catwalk from the bombsite. Next, a few seconds later, the rest of the T players are eliminated. The CTs then go to defuse the bomb, since all T players were eliminated.

5.5.3 Understanding the Game Outcome in Baseball

Baseball is a bat-and-ball game played on a diamond field with four bases. Two teams of 9 players each take turns between batting (offense) and fielding (defense) positions. The batters have to hit the ball out of the reach of the fielding team and run by the four bases to score a point (run). Meanwhile, the fielding team has to catch the ball and eliminate the attackers (by reaching a base with the ball) before they are able to save bases and score runs [54].

Every offensive player starts at the batting position. A pitcher throws the ball at the batter, who needs to hit it with a bat before running the bases. Figure 5.6(c) shows a schema of the baseball playing field, with the position of the bases and the starting positions of the players. Offensive: Batter (B), and defensive: Pitcher (P), Catcher (C), three Basemen (1B, 2B, 3B), three Fielders (LF, CF, RF), and ShortStop (SS). Batters that reached a base safely become runners (not shown), denoted by R@1, R@2, and R@3. The temporal evolution of baseball plays has a significant impact on its outcome. If a player from the batting team reaches a base before a player of the fielding team, he is safe. If the opposite happens, he is out.

Figure 5.1(c) shows the Play Diagram of a play by the Texas Rangers versus the Colorado Rockies on August 9, 2019. There are runners on the first and second bases. We can see that they reached the second and third bases safely. However, it is not possible to see what happened to the batter. The ball is batted, then caught by the second basemen and passed to the first basemen. During the play, both the first basemen and batter go to the first base, but we do not know who arrived first.

Figure 5.9 shows the TrackRuler visualization for this play. We are interested in analyzing the outcome of the batter. Therefore, we hid all the players that do not interact with the batter: catcher (C), pitcher (P), shortstop (SS), third baseman (3B), fielders (LF, CF, RF), and the other two runners (R@1, R@2).

The resulting chart makes the outcome of the play very clear. The user moves the mouse close to the play diagram and aligns the horizontal guideline with the first base (A). Using the Projected Timeline, we find that the first baseman arrived at first base at 6s (B) and received the ball at 7s (C). Meanwhile, the batter only reached first base at 8s (D); therefore, he is out. Using the Play Small Multiples, we can see the approximate time of runners reaching the third (E) and second (F) bases.

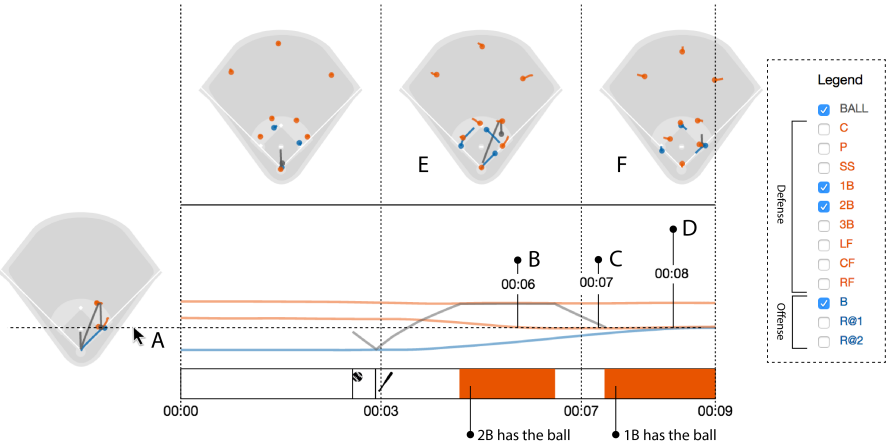


Figure 5.9: TrackRuler representation of a Texas Rangers versus the Colorado Rockies play. A) Horizontal guideline showing the Y position of first base. B) first baseman arrived at first base. C) received the ball. D) batter reached first base after the first baseman, and therefore is out. The Play Small Multiples show us the outcome of the other two runners, who have reached second and third bases safely (E, F).

5.6 Expert Feedback

We collected feedback from the five sports experts described in Section 5.2, namely soccer (S1, S2), CSGO (C1, C2), and baseball (B1) experts. We gathered the feedback in one-on-one interviews.

We received very positive feedback from the experts. All participants learned to read TrackRuler in a few minutes and were excited to explore their play trajectory datasets with it.

S1 and S2 liked to use the Projected Timeline to analyze soccer plays. Using the timeline, they could quickly identify when teams were on the attack or defense (attack players move towards the opposite team goal, whereas defense players would move back to their own goal). S1 preferred to look at the movement trends of all players at the same time. He liked to select players to highlight them while keeping the other players grayed out in the background. Meanwhile, S2 said he liked to hide groups of players to declutter the graph. In particular, he analyzed soccer plays one team at a time.

Both C1 and C2 enjoyed the overview provided by the Play Small Multiples and the Event View. The experts could understand entire CSGO plays by looking at

them. C1 mentioned he liked using the Projected Timeline to see players changing directions, which CSGO experts call “fake”. A player pretends to go in one way in a fake, but changes direction once the opposing team spots them. C2 used the Projected Timeline to analyze how synchronized the teams are. In other words, are they moving at the same pace? He also liked that this view could represent CSGO movement patterns very clearly, such as pushes (advancing to another location) and camping (staying in a fixed location), which were hard to see in Play Diagrams.

B1 appreciated that the entire baseball play could be represented at the same time in TrackRuler. He also enjoyed using the system interactions, such as rotating the field and highlighting players, to explore the play in detail. B1 mentioned that the player movement in baseball is structured, and he was interested to see how our visualization would behave in less structured sports. We showed him the soccer and CSGO visualizations, and he was particularly impressed with the visualization of soccer plays, noting attack and defense patterns immediately.

5.7 Discussions and Limitations

TrackRuler presents spatiotemporal trajectory information using two linked views: the Play Small Multiples and the Projected Timeline. Each component is suited for a different type of analysis, but shared axes can facilitate context switch between them. The Play Small Multiples focuses on representing the spatial evolution of a play, and the temporal information is shown by the alignment of the small multiples and the time axis. Meanwhile, the Projected Timeline focuses on portraying the play’s temporal dynamics. This chart is aligned with a reference Play Diagram and portrays the Y position of the players over time. This view has the disadvantage of having half of the spatial information discarded. We alleviate this issue by introducing interactions that allow the user to rotate the field by 90 degrees (representing the temporal evolution of the players’ X position) and to highlight players so that their movement becomes more evident in the Play Diagram. Another limitation is that clutter might occur depending on the number of players in the field and their respective positions. In order to reduce clutter and facilitate the data analysis, we allow users to hide players from this visualization.

In order to demonstrate the analytical power of our visualization, we have used

three sports trajectory datasets: synthesized soccer plays, Counter-Strike trajectory logs, and hand-annotated baseball plays. We limited our analysis to these sports because trajectory data is an expensive resource [68], and very few spatiotemporal sports datasets are openly available today. The baseball and the Counter-Strike datasets contain real plays. The soccer data, however, is synthetically produced by a reinforcement learning algorithm. We note that the plays are generated in an advanced physics-based 3D simulator [40], and our experts mentioned they looked realistic. As future work, we would like to test our visualization in other sports domains. Once more spatiotemporal sports datasets become available, TrackRuler can be directly applied. We also note that our encodings are not limited to sports. We would also like to investigate other spatiotemporal dataset domains, including biology and urban sciences.

The sports trajectories investigated by our experts were less than 3 minutes long. We anticipate that for longer trajectories, the user will have to set a longer time window in the Play Small Multiples view. Furthermore, depending on the user's monitor size, they will have to scroll the visualization horizontally to explore the entire play. In future work, we would like to investigate these issues and visual encodings that can address them.

5.8 Final Considerations

This chapter presented TrackRuler, a methodology for visualizing spatiotemporal sports data that can represent game trajectories and events in detail. TrackRuler does not make any assumptions on the trajectory data and can be customized to any sport. We guided our design on interviews with sports experts, who described their workflows and provided us with feedback in each iteration of the system design. We have described use cases on three domains, soccer, Counter-Strike, and baseball, showcasing the generalizability of our work. Our use cases demonstrate how TrackRuler can help analysts understand the play evolution, identify playing strategies, find player mistakes and better understand the game outcomes.

TrackRuler allows the exploration of one gameplay at a time. However, it is customary in sports to contextualize the play based on historical data. For example, noticing that a baseball runner is the fastest player in the season is a valuable insight

for sports fans. In the next chapter, we present a tool for the contextual analysis of baseball plays that can automatically generate commentaries and insights to the users.

Chapter 6

GameCast: Context-Aware Sports Analytics

6.1 Introduction

Sports events are generating increasing amounts of data, from broadcast video to low-level tracking data [10]. For example, a professional baseball game generates almost a full terabyte of data [93]. As stadia and arenas are being retrofitted with data capture devices, sports leagues and teams exploit the new data by making it accessible to teams and consumers. While there has been a rising focus on how data can transform team operations, such as analytical methods to value players and predict performance, data is also revolutionizing sports media by driving unique modes of fan engagement. For example, Major League Baseball (MLB) has started to produce successful, stats-focused media broadcasts [15, 16]. Statistics-forward media development aligns well with the interests of sports fans, who are heavy consumers of statistical content [8].

Despite the growing interest in statistics-focused sports media content, there are significant analytical and visual challenges when dealing with sports data. First, statistical sports content is often non-contextual. Context is essential for comparing sports plays, as play context heavily influences player decision-making. For example, in baseball, the speed of the batter en-route to first base is much different for sacrifice bunts than it is for sharp infield grounders, as the players will almost certainly exert different amounts of effort when running. If two players

see drastically different distributions of play contexts, the interpretation of the statistics they accrue on those plays should be different. Accordingly, this nuance of capturing context is crucial when comparing statistics across a corpus of plays and should be accounted for in sports visualization systems.

Visually presenting sports data is challenging due to competing interests among the various stakeholders. Zhi et al. [100], indicate that while fans seek to identify particular plays, sports journalists tend to explore the "why" and "how" of those plays. Furthermore, when it comes to statistical information, Zhi et al. suggest that journalists typically use the data facts to serve as leads for exciting storylines, whereas fans typically consume the raw numerical content. Due to these competing interests, existing systems find it difficult to reconcile both the narrative and exploratory visualization requirements of each user groups' interests. One emerging area that serves both sets of users is natural language generation and processing. For basketball, Metoyer et al. [50] detail a methodology which links text and visualization by analyzing text to generate relevant graphics for narrative visualization. Chen et al. [18] suggest that users look for supporting descriptions for the context of the visualizations they see. Natural language generation has proven helpful in interpreting a broad range of visualizations, such as in the Voder system introduced by Srinivasan et al. [79]. In sports visualization, natural language descriptions can also provide data facts around plays that satisfy fans' needs to consume interesting statistical content while also allowing sports journalists to use the generated textual commentary to explore possible stories.

This chapter presents GameCast, a system that automatically generates interactive commentary from live feed baseball data. Our system is directed towards fans who seek a more data-focused game viewing experience. We expand upon previous sports narrative work, which focused on providing retrospective summary visualizations of games. Crucial to our system is a unique clustering methodology that contextualizes plays, so statistics are more fairly compared.

6.2 Play Clustering Review

Clustering sports plays is helpful for sports visualization, as it allows users to explore similar plays in an interpretable context. At a high level, drawing from the

history of baseball games, baseball plays can be clustered using simple language of symbols, such as "=" to denote a double, "5-3" to designate an out where the third baseman threw to the first baseman for an out, or Ƹ, a backwards K, to indicate a strikeout looking, which is a strikeout where the batter does not swing [55]. While clustering plays based on how they would be scored is easy and provides for a quick clustering method, we discard valuable information, such as trajectories or derived play statistics, that can provide more granularity for play clusters.

In soccer and basketball, trajectories are a common data source used to cluster plays. Sacha et al. [72] present a system that allows for visual exploratory analysis of a large trajectory dataset by allowing for several levels of abstraction. Sha et al. [76] present *Chalkboarding*, a system where one can query for similar soccer plays based on player trajectories. Di et al. [26] extend this work by proposing a pairwise learning to rank approach that improves search rankings based on user click behavior. Sha et al. [77] present an interactive system where a user can draw on the broadcast video and query for similar plays. While the above trajectory-based methods have been mostly applied to sports such as soccer or basketball, there have been no attempts at clustering baseball plays beyond those widely accepted in-game scoring. One potential downside to using trajectories is that fundamentally similar plays can exhibit significant variations in trajectories, as shown in Figure 6.1.

6.3 GameCast Play Clustering

To power our clustering method, we rely on a large corpus of plays collected by Statcast. Each game contains three sets of data: (1) the game JSON file, (2) the game trajectories, and (3) the game video. In the game JSON file, which is updated throughout a game, we have the game's metadata, such as the teams playing, venue information, and score, along with a list of gameplays. Each play is given a unique ID within the context of a game. Additionally, depending on how each play transpires, a specific set of statistics are maintained. For example, in a play where the second baseman throws the ball, statistics like throw speed or release time are calculated for the second baseman, but not for plays where the second baseman does not make a throw. Over the past few years, as Statcast has been introduced into broadcasts, many of the advanced statistics that Statcast

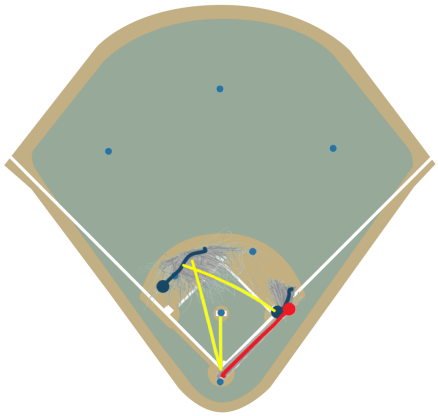


Figure 6.1: Trajectory cluster: Trajectories can vary drastically within a single type of play. In this play, the shortstop throws to the first baseman for the out (known as a "6-3" play). On the left side of the field, we can see significant variability in the trajectories for the shortstop across over 5,000 similar plays.

maintains have become ubiquitous among baseball fans, such as exit velocity or launch angle. We use games from the 2017 and 2018 seasons to power our system, totaling 2,466 games and 187,021 plays.

While one can use the trajectory-based methods described in Section 6.2 to cluster baseball plays, trajectories themselves carry a high degree of variability. For example, in Figure 6.1, we see a 6-3 play, the name for a play where the shortstop threw to the first baseman for an out. In particular, we observe a significant amount of variability in the shortstop's movement. Furthermore, trajectory-based clustering methods can impose high computational costs, mainly when the space of candidate plays is large. It is unclear how to cluster trajectories in such a large and noisy space in an interpretable, computationally efficient manner. At the same time, we could cluster plays using the events in a play, such as which players throw the ball or are tagged out. However, we discard information about other players whose actions may not have been recorded yet still are valuable to understanding the play. To address these issues, we propose to cluster plays based on the statistics that Statcast calculates. In doing so, we create interpretable clusters of plays at a negligible computational cost.

The statistics computed from plays provide a balance between the low-level tracking information and the high-level play descriptions. The Statcast system

Statistic	P	C	1B	2B	3B	SS	LF	CF	RF	B	R1	R2	R3
Sprint Speed	13.3	10.7	9.2	15.0	14.4		12.1	13.7	13.7	13.6	13.1	14.2	18.1
Pitch Velocity	82.4												
Hit Distance										267.7			
Arm Strength			53.6					90.1					
Distance Covered								62.9					
Home to First										6.1			
Home to Second													
Home to Third													
First to Second													
First to Third													
First to Home													
Second to Third													
Second to Home													
Third to Home													9.3
Pop Time													
Launch Angle										43.1			
Exit Velocity										91			
Hit Type										Fly			
Throw Distance			45.23		33.52			183					
Extension	6.1												

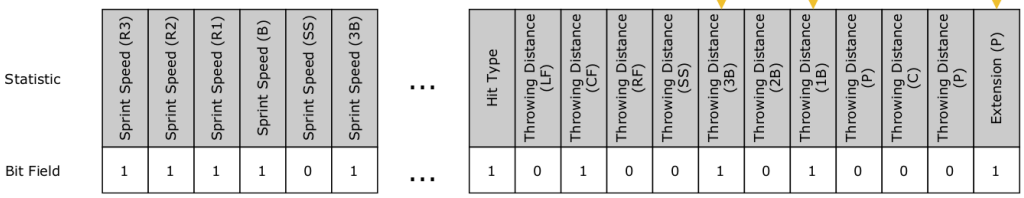


Figure 6.2: The statistics computed from a given play, viewed as a table (top) and the conversion of the table to a bit field (bottom). If we consider that the statistics cover most of the aspects of the behavior of the players, the shaded cells would work as a signature of the type of the play, or of the play context, even without their actual values. These statistics say what happened, and how it happened, and thus capture the context of a play.

determines which statistics are relevant to a given play and computes them when appropriate. For example, "Top Speed" will only be computed if a player ran and "Throwing Distance" will only be computed when the ball is thrown. The set of computed statistics for a play then encodes information about the geometry (the way players moved) and the events (runs and throws). Thus, the context of the play will be captured by the set of statistics itself. In other words, knowing which statistics were computed for a play is enough to understand what happened in the play. Other plays, for which the same statistics were computed, could be considered similar to this one since the same actions were performed.

We then propose a method to cluster plays by their recorded statistics. The goal is to associate a unique identifier to each distinct set of collected statistics for

every play. We start by taking the table of statistics computed for a given play. The table is converted to a bitfield where each cell corresponds to a bit (Figure 6.2). The bit field is then passed through a hash function to generate a cluster id, which uniquely identifies the type of play where these sets of statistics are observed. We discard plays with errors, such as missing statistics, from consideration.

Plays are then clustered based on their cluster id. Clustering plays into coarse groups to facilitate visual analytics applications has proven effective in sports like basketball, as it provides for quick play lookup [76, 77]. While the bit field itself may be hard to read, and the cluster ids themselves are meaningless, once the plays are clustered, the information contained in each cluster becomes clear. One benefit to our play clustering method is that it can be performed in an online fashion. Another benefit to our method is that bit fields also lend themselves to efficient inter-cluster similarity calculation through the use of a measure such as the Jaccard similarity coefficient, defined below for two sets, A and B.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} \tag{6.1}$$

For example, take two clusters, defined by their bit fields: $C_1 = 0011001$ and $C_2 = 1111001$. If we consider each bit that is equal to 1 as an element of each set, we would see $J(C_1, C_2) = \frac{3}{5}$. We could also use Hamming distance, which is defined as the number of bit flips to obtain equal strings. Using C_1 and C_2, we would see a Hamming distance of 2. Intra-cluster similarity may be achieved by techniques such as using simple Euclidean distance on the raw play statistics or trajectory-based methods described in Section 6.2.

6.4 The GameCast System

Based on informal domain expert interviews, we collected the following requirements for a baseball visualization system:

R1 Allow for both a game summary and quick play selection. Users should easily be allowed to select events and plays, and the degree of interest or importance of a play should be apparent.

R2 Enable cluster exploration so that users can understand the context with which play statistics occur.

R3 Provide textual descriptions of the play using play context and StatCast statistics.

R4 Support video replay, where the video is synchronized with the other views. Metoyer et al. [50] find that fans frequently use video playback to focus on particular plays, and Perin et al. [65] suggest that synchronization between video and the system interface is essential for live systems.

We present the interface of GameCast in Figure 6.3. GameCast consists of a collection of five linked visualizations. Our interface includes a play selection widget, two play diagrams (Baseball Timeline [62]), video playback and textual commentary.

6.4.1 Inning and Play Selection

Easily filtering and selecting important plays is crucial to any exploratory or narrative visualization for baseball, as a game has on average 75 plays. As Zhi et al. [100] note, fans focus attention and exploration on particular plays when navigating recaps. At the same time, sports journalists usually seek plays they consider as "key" events. In baseball, this usually means a close play or a scoring play. Therefore, GameCast users of all types should easily and quickly be able to select events and plays that are interesting to satisfy **R1**. Since baseball naturally has a hierarchy consisting of innings and half-innings, which is familiar to almost all fans, we developed our play selection tool with this in mind. Figure 6.4 shows a standard table used to summarize baseball games, often dubbed the *box score*, along with our inning-play selection table.

Our inning-play table provides a hierarchical view of the baseball game that is familiar to baseball fans. Innings are split into top and bottom halves, where the visiting team bats in the top half and the home team in the bottom half. Each half-inning contains numerous at-bats, which are synonymous with plays. Under each at-bat, there are multiple metadata, such as markers indicated scoring or rare plays. The metadata we include for each play encompasses a subset of plays

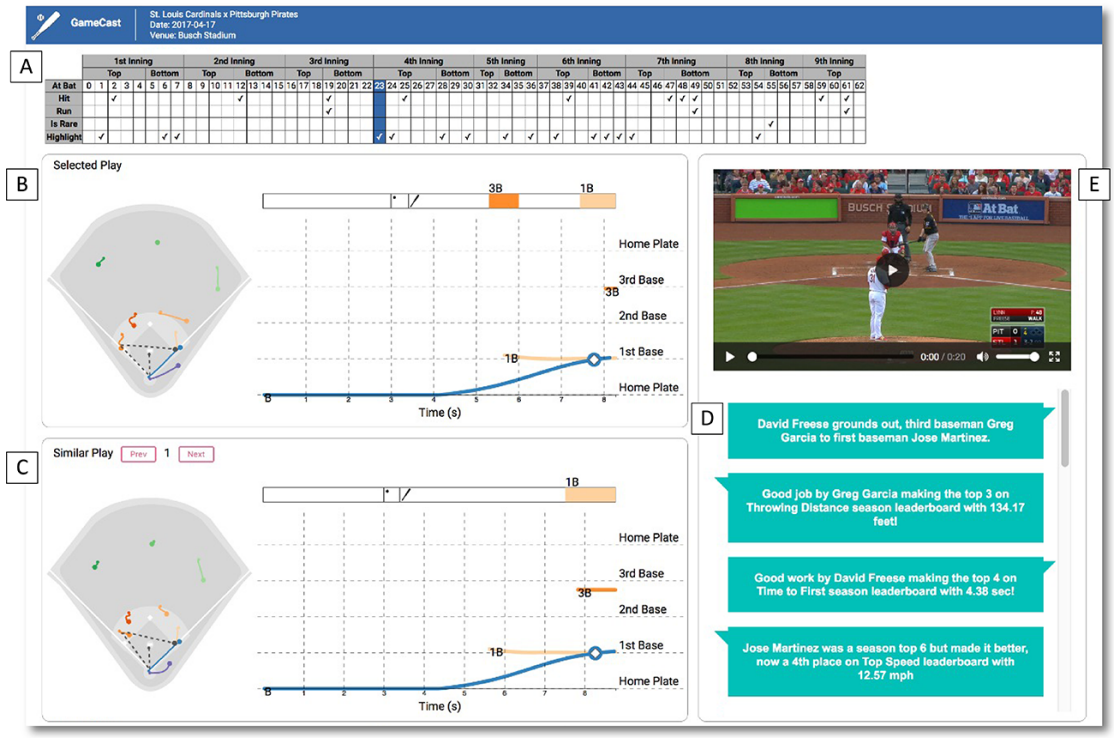


Figure 6.3: GameCast interface showing a game between the St. Louis Cardinals and Pittsburgh Pirates. **A** shows the inning and play selector, where "interestingness" is shown via associated markers below the play, such as flags for rare or scoring plays. **B** is the selected play's corresponding play diagram and baseball timeline. Play diagrams show player trajectories and baseball timelines show events, such as ball releases or catches. In **C**, users can cycle through the various other inter-cluster plays through "previous" and "next" buttons to see their associated play diagram and baseball timeline. **D** is the generated contextual commentary for the play. **E** is the associated play video feed.

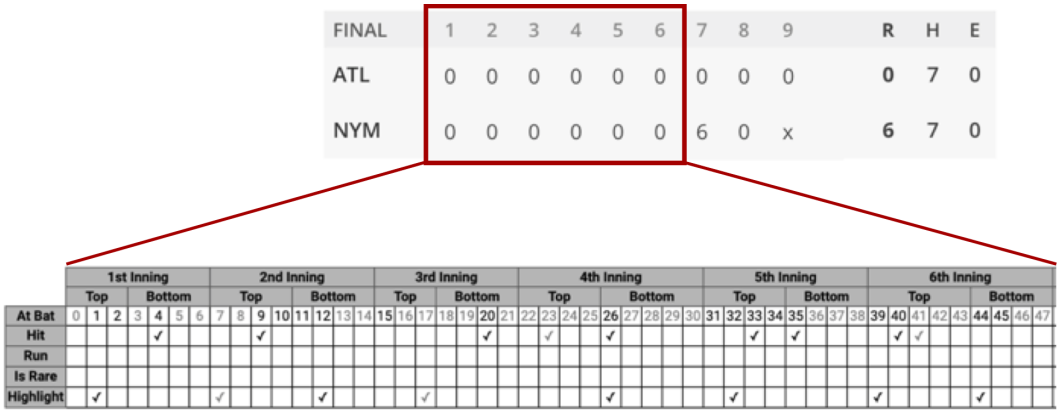


Figure 6.4: Inning-play selection view: Traditional box score (top) is typically sparse, due to the nature of baseball, while GameCast's inning-play selection view (bottom) provides more detail to differentiate plays, which allows emphasis on the ones that interest the user. While a box score would suggest the first six innings were uneventful, as no runs were scored, GameCast's play selection tool reveals interesting plays for users to dig deeper. In fact, $17/47 = 36\%$ of plays in the first six innings provided either a highlight or hit.

generally thought of as interesting. After consulting with domain experts, we define an interesting play as one that falls into one of the following categories: (1) a scoring play, (2) a play with generated textual data facts, (3) a "rare" play, (4) a hit. We define a rare event using a threshold for cluster size. For GameCast, a play is considered rare if it belongs to a cluster of fewer than 30 other plays. From our inning-play table view, users can not only immediately identify interesting groups of plays across a variety of different criteria and easily select them to populate the other views, thus fulfilling **R1**.

One of the benefits of our design is that we overcome the sparsity of box scores, as the number in each team-inning cell corresponds to several runs scored in that half-inning. Due to the nature of baseball, it is quite common for an inning to result in zero runs scored for either team. However, a lack of runs scored does not imply that the inning was less interesting than when runs did occur. For example, an inning may contain multiple hits or rare plays without any scoring plays. GameCast's inning-play selector better elucidates interesting stretches of the game than a traditional box score while remaining familiar to the vast majority of baseball fans.

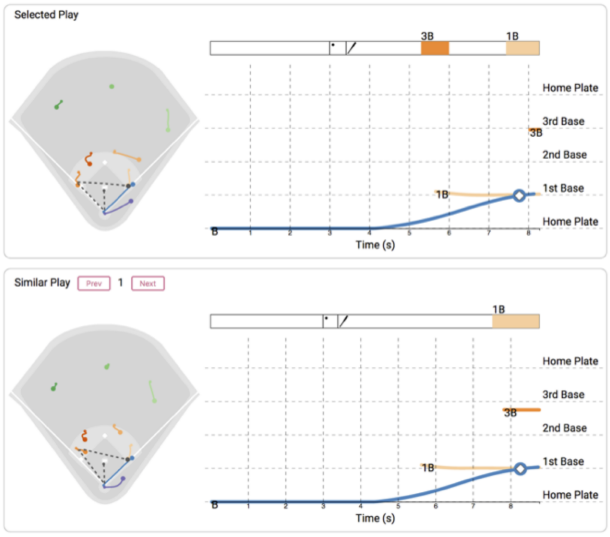


Figure 6.5: Comparing the selected play with a similar play. Jose Martinez grounds out to third baseman David Freese, who throws to first baseman Josh Bell for the out. Users are able to better visualize this class of plays – grounders to third basemen – by exploring the corresponding cluster of plays using "Previous" and "Next" buttons.

6.4.2 Understanding Play Context

Understanding a player's trajectory may assist in understanding their actions. For example, if a fielder moves directly towards a base, we may infer that he is tagging a player out or anticipating a catch. One way to visualize player trajectories statically is through *play diagrams*. A play diagram is a standard way of mapping spatial baseball data, where we have a top-view of the playing field with objects' trajectories as lines. We map player trajectories using solid lines and the ball trajectory using a dotted line. The small circle on a player's trajectory is his starting point, where the big circle is the player's position at the end of the play. Since baseball fields are not standardized across stadiums, we use the appropriate field shape for each stadium in our data set.

Player actions, along with their trajectories, define a play's context. While trajectories may give hints on player actions, they do not give the full and necessary detail of a play. To help visualize player movements and their events, we utilize Baseball Timelines [62] positioned to the right of the play diagrams. Baseball timelines summarize the actions of the players by displaying both spatial and

temporal aspects of plays. Since actions, such as throwing the ball, generate a specific set up play statistics (e.g., throw speed) for certain players, baseball timelines shed light on the context determining a play's assigned cluster. At the top of the baseball timeline chart is a series of events, such as ball release, hits, which player had the ball, when, and for how long. Baseball timelines can be especially useful for plays where the play diagram is cluttered due to too many overlapping trajectories, such as plays with runners on base. Both the play diagrams and timelines are synchronized with the play video footage. More specifically, the play diagram shows the players' positions up to the current video frame being played, and a vertical bar encodes this information in the timeline. These visual encodings satisfy **R4**.

The use of play diagrams and baseball timelines is a significant departure from MLB's Gameday system, which focuses on the batter-pitcher duel. One of the downsides is that many events considered exciting during a baseball game, such as a hit, a double play, or a tough out, are only described textually in Gameday. Play diagrams and baseball timelines provide more color to these impactful events by characterizing player movement and the actions they take. Furthermore, we allow a play to be easily compared against those in its cluster. Users can pull similar plays to the selected one using the "Previous" and "Next" buttons, satisfying **R2**.

6.4.3 Play Commentary

Chen et al. [18] found that textual narrations complemented their basketball narrative visualization system well, as the game and play views lacked supporting descriptions. Since GameCast is a visual aid to baseball gamecasting and the overall fan experience, providing game commentary is essential. As most play-by-play commentary revolves around statistics, our clustering method based on a play's recorded Statcast statistics fits well [44].

Our commentary system utilizes 14 templates, 4 for new, exciting results and 10 for historical highlights about the play category, to generate insights from cluster-specific data facts. The unbalanced number of messages is a consequence of the baseball dynamics, where only an average of 16 new and exciting results on statistics are observed in a game. An example of an exciting result follows the line of "*Great job by Austin Romine making the top 5 on Top Speed season*

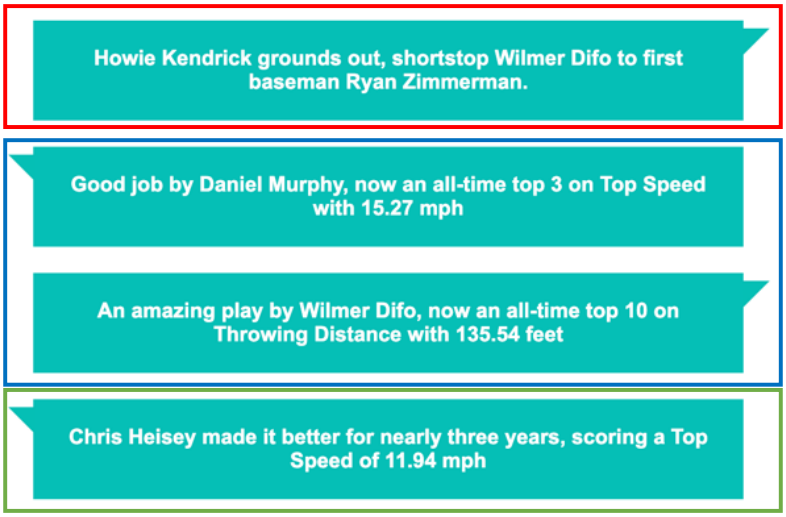


Figure 6.6: A collection of commentaries including the play description (red), exciting results (blue) and historical highlight (green). Although a shortstop to first baseman out is one of the most common outs in baseball, this particular play contained impressive defensive performances from Daniel Murphy running to cover the first baseman, and Wilmer Difo for a long throw, which are missed in traditional game recaps.

leaderboard with 12.3 mph!", while a historical highlight looks like "*Eduardo Nunez holds the top Throwing Speed as a shortstop with a 103.87 mph for three years*". The commentary is structured as a back and forth between two commentators, much like how commentating works on broadcast.

The number of messages generated for each play is dependent on the number of statistics observed in the play, but historical highlights will always be part of the output. The messages result from comparing the players' performance and the rankings of the cluster statistics. Thus, given these constraints, we satisfy **R3**. MLB maintains a hand-annotated textual description for each play called a "stringer". In our system, we list the *stringer* play description before any commentary. We outline the distinction between the stringer play description and different types of generated commentary in Figure 6.6.

4th Inning								5th Inning					6th Inning						
Top					Bottom			Top		Bottom			Top				Bottom		
23	24	25	26	27	28	29	30	31	32	34	35	36	37	38	39	40	41	42	43
		✓													✓				
✓	✓				✓		✓			✓		✓		✓			✓	✓	✓

Figure 6.7: Selecting a play in the Inning-play selection view. Although no runs were scored in these three innings, and there were only two hits, over half of the plays generated commentary with interesting data facts.

6.5 Case Studies

Here, we present three use cases inspired by interviews with professional teams and baseball fans, each of which highlighting how the different components of GameCast aids in the user experience. Specifically, we focus on a game between the Pittsburgh Pirates (away) and the St. Louis Cardinals (home) on April 17, 2017.

6.5.0.1 Play Selection

Often, baseball game recaps distill information into an aggregated form. For example, box scores aggregate runs by an inning and are often accompanied by tables containing tabulated batter performance for the corresponding game. Most of the focus on game recap revolves around run-scoring events, as this is shown directly in box scores, and usually, the first column of player statistics displays how many runs the player made. However, we know that run-scoring events are not the only ones of interest. Fans want to find exciting plays quickly, and journalists work on tight timetables to produce game recaps.

From Section 6.4.1, we know that baseball summary statistics tend to be sparse due to the nature of the sport. In box scores, this can be deceptive if a user wants to identify exciting plays. For example, in Figure 6.7, we show how although the Pittsburgh - St. Louis box score displays no runs scored in the 4th, fifth and sixth innings, it contains a stretch of the game with the most interesting events, as defined in Section 6.4.1.

By all conventional thought, after consulting with our domain experts, the innings displayed in Figure 6.7 would be considered uneventful as only two hits and zero runs were recorded. However, these innings contained 11 out of the 15

total generated commentaries for the game. Interestingly, we see that on the two plays that were hits, GameCast generated no commentary, meaning besides the hit, there were no intriguing statistical performances to report.

6.5.1 Exploring Play Clusters

One of the benefits of GameCast over traditional systems is the ability to explore the large space of plays similar to the selected play. In doing so, a user can better frame the current play against other similar plays and compare and contrast to find unique characteristics. Sports journalists may find this helpful in contextualizing their narrative, and fans may enjoy this feature to explore relevant statistics. Additionally, our play exploration feature is necessary, considering that our play clustering algorithm produces 3,612 total clusters of plays. Since we have 187,021 plays, this gives an average cluster size of about 52 plays. However, cluster size distribution is positively skewed, meaning the median of cluster size, which is 28, is lower than the mean. Thus, most of the clusters are small enough for users to explore.

Let us consider the play at the top of Figure 6.8, where we see a standard play: a groundout from third to first (a "5-3" out) with no runners on base. If one were to view the play diagram and baseball timeline, the play would appear as a typical "5-3" out. However, a user can quickly view the play diagrams and baseball timelines for other plays in the same cluster through the cluster exploration feature. In doing so, it becomes apparent that there are a few remarkable characteristics of the play. For starters, the play diagram clarifies that the third baseman made the throw right on the boundary of the field from behind the third base. Additionally, the third baseman's trajectory indicates that this likely was a backhand grab, which is considered a difficult play in baseball. Finally, the first baseman makes a long recovery run, as he is considerably farther from the first base than in similar plays. These observations are confirmed within the game video.

There may be some instances where play diagrams are too cluttered to read. Typically, this may happen with plays with many infield trajectories, i.e., when there are runners on base. Ono et al. [62] highlight these situations as the motivating factor behind baseball timelines. Because these plays typically have runners on base, they are often high leverage plays since there is a significantly higher chance

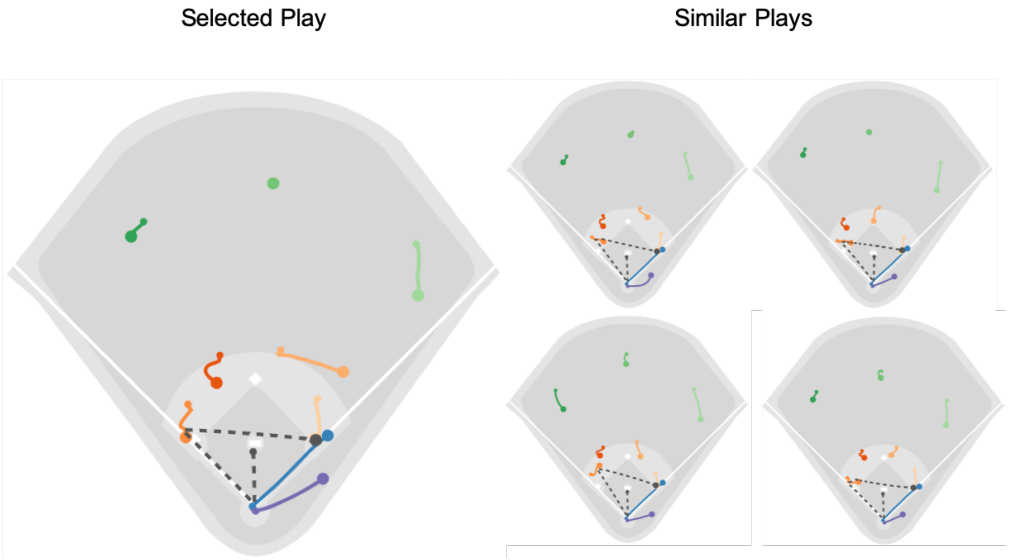


Figure 6.8: Play Diagrams of the selected play and similar plays. David Freese grounds out, third baseman Greg Garcia to first baseman Jose Martinez. Similar plays indicate the broad class of plays, which are grounders to third basemen with bases empty.

of a run occurring with runners on base than not. In Figure 6.9, we see a play with runners on base, where the third baseman opts to throw the runner from the third base out at the home plate rather than throw to first base (known as a fielder's choice). On the associated play diagram, there are many overlapping lines, which makes it hard to interpret. From the baseball timeline, we can easily see that the third baseman throws to the catcher for out. In similar plays, we see the third baseman aim primarily for the first baseman, highlighting the unique decision-making of this particular play.

6.5.2 Fair Data Facts from Play Commentary

A play's underlying context influences player's effort, and thus the statistics their performances generate. For example, players may not run as hard on routine plays or throw the ball as fast. Additionally, players may be more adept than others in particular situations. Thus, it is difficult, and sometimes inappropriate, to broadly compare a player's average performance for a statistic unless that statistic is calculated in similar contexts.

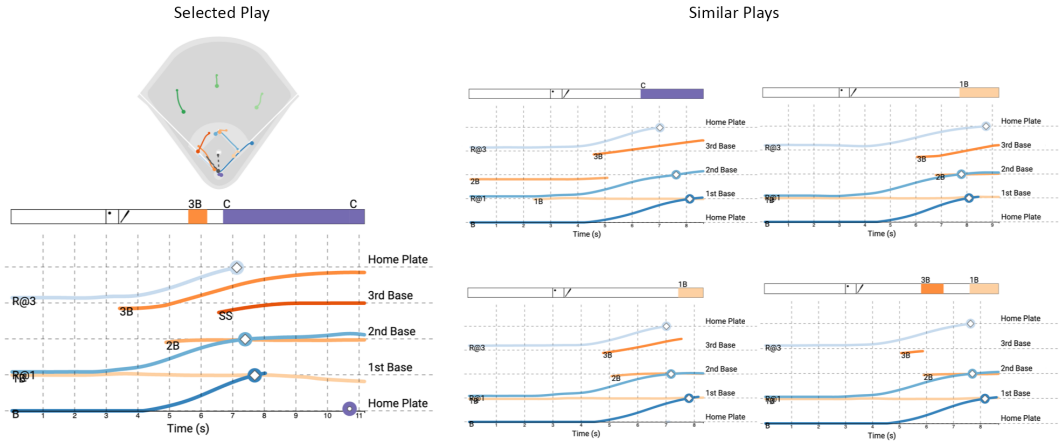


Figure 6.9: Baseball Timeline representation of the selected play and similar plays. New York Yankees vs. Cincinnati Reds (July 25, 2017). The third baseman takes a fielder's choice and decides to throw out the runner from third base at home plate. The play's play diagram (top left) makes this hard to see, as many lines overlap. We show the play's baseball timeline (bottom left) and similar plays (right), highlighting the third baseman's unique decision-making on this play.

While the play diagrams and baseball timelines help users understand the play context, the generated commentary helps users understand the consequences of that context. In doing so, the generated commentary provides interesting data facts. Using the same play as indicated in Figure 6.8, we show the associated commentary in Figure 6.10. The generated data facts confirm the analysis in Section 6.5.1. We see that third baseball broke into the top three of throw distances on similar plays, which broadly cover "5-3" outs. Additionally, our observation of the first baseman's speed is confirmed, as he beat his previous best top speed on this play. Finally, although hard to acknowledge from a play diagram or baseball timeline, we can see that the batter, David Freese, made an impressive baserunning performance, although he was out. This is an interesting data fact, as Freese was generally considered an older, slower player at the time of this play.

6.5.3 Domain Expert Interviews

To further evaluate GameCast, we conducted interviews with four experts. Two of these experts were analysts at MLB teams (A1, A2), and two were lifelong fans (F1, F2) with a deep understanding of the sport. Each expert had a significant

Figure 6.10: Commentaries generated for the selected play. From the commentary, it is clear that all three main players involved in the play (the batter, the third baseman and the first baseman) performed remarkably on this play. The third baseman completed a long throw, and both the first baseman and batter produced remarkable running statistics such as top speed and time to first. Without this commentary, viewers would only see the play description text (top).

understanding of baseball's recent technological advances and was familiar with Statcast and its generated data. We chose these four experts for their broad familiarity with baseball, its data, and current gamecasting experience.

We conducted the interviews accordingly: first, we ensured the user was familiar with play diagrams and baseball timelines. Then, we introduced GameCast and answered any questions the user had. We allowed the user to explore the game that we reference in Section 6.5. Next, we conducted a qualitative interview, where we asked the following questions:

Q1 Did you find any aspect of GameCast difficult to use? Why?

Q2 Did the play diagrams and baseball timelines help you mentally generate an idea of the similar plays? How?

Q3 Did you find the commentary insightful and relevant? Why?

Q4 What other visuals or data do you wish you had?

Generally, the different components of GameCast were well received by the domain experts and required little training time. As all viewers could understand

the play diagram and inning selection diagram, there were no negative comments from any of the experts on these views, and they appreciated the ease of use of these features. A1 noted that he welcomes a tool to accompany statistics-focused broadcasts, as he found the broadcasts to be too educational to support advanced fans. The users required training to use the baseball timelines, as they had no previous familiarity with this visualization method. This is unsurprising, as baseball timelines are relatively new visualizations. F2, however, reaffirmed the baseball timeline's ability to decode the clutter in the play diagram on plays with large amounts of infield movement, noting it was helpful on plays where there were runners on base. However, as the number of plays in a baseball game with runners on base can be low, these findings indicate that future designs may want to provide the option to view a baseball timeline but default to the more familiar play diagram.

F1 and F2 found the feature to explore similar plays helpful as they were curious to see how the selected play differed from the rest of its cluster. They also found the functionality particularly useful for plays designated as rare. Conversely, A1 and A2 saw limited need for exploring the space of similar plays, as they felt satisfied from inferring context directly from the play diagrams and cycling through a few similar plays. When exploring similar plays, every expert mentioned they focused their attention mainly on the play diagram rather than the baseball timeline. Both user groups noted that the play clusters made sense and felt that they had a good sense of the play cluster after viewing a handful of plays, which gives credence to our play clustering method.

Each expert mentioned that having the raw statistics, such as throw distance or sprint speed, for both the play and its similar plays would be helpful to the exploration and framing of the context. However, each user group found a different utility in having the raw statistics. The fans drew excitement from being able to reference the statistics, particularly for outlier statistical performances. On the other hand, the analysts idealized quickly sorting through plays to estimate the distribution of the statistics. A1 and F2 mentioned they would like to see these statistics displayed instead of the baseball timelines. However, F2 noted that the option to view baseball timelines would be crucial for particularly cluttered play diagrams. These insights provide direction to future visualization efforts, which should likely include the ability for the users to view the raw statistics generated by

Statcast. Given fans' tendencies in sports as a whole, integrating the raw statistics or data into the visual system is likely to hold across all sports.

Users found the generated commentary insightful, and particularly so for new, exciting events. A1 mentioned that the generated commentary helped confirm suspicions on impressive performances identified from the video. F1 and F2 noted that while the historical statistical commentary was interesting, it could be contextualized better to see the complete statistics leaderboard. A2 even suggested that the commentary should only focus on new, exciting statistical achievements. The experts found the coupling with video to be particularly important for new and exciting commentary, as it acted as a reminder to check the game video to understand the achievement further. All experts agreed that the language used in the commentary could be improved, mentioning it sounded "monotone" or "robotic".

Concerning **Q4**, A2 suggested that the information in the play selector be given more granularity. For example, each cell could contain a visual encoding for the hit type in the hit row, such as grounder, fly ball, or line drive. For the rare play row, the rarity of a play could be represented by the size of the play's cluster. Cluster size could be encoded either through the number itself or visually using a gradient.

6.6 Final Considerations

This chapter introduced GameCast, a live baseball commentary system that uses play clustering and natural language generation to craft a contextual game narrative while allowing for fair exploration of a game's plays. GameCast, and its associated play clustering technique, are built upon thousands of games of StatCast data, which provide low-level event and player trajectory data for every play. GameCast implements a hierarchical view of plays, play diagrams, and baseball timelines, along with contextual data facts generated as game commentaries. We evaluate GameCast across three different use cases, each highlighting a particular functionality of the system, along with expert interviews, including analysts at professional baseball teams and longtime fans. Our evaluation suggests that GameCast can significantly support the user game viewing experience.

We see several avenues for future work. One of the limitations of GameCast is that it focuses on the play and at-bat level of baseball. However, we see that

other applications, like MLB's Gameday system, focuses on the batter-pitcher level. In future work, a system can blend elements of both, allowing users to explore the at-bat at an even deeper level. In line with the work on automated textual narrative generation for baseball, from Lee et al. [44] and Allen et al. [4], along with the user feedback on the generated commentary, we believe future work should investigate the use of advanced textual generation systems which rely on artificial intelligence techniques. Currently, many text generation techniques for sports rely on templates, which are often considered dull. However, Wiseman et al. [91] suggest that template-based approaches exceed the performance of existing neural models. Thus, more work is needed to develop text generation models that provide a higher variance in the generated text structure. Additionally, since we see that GameCast and its underlying play clustering algorithm are easily extendable to other sports, future work can implement similar game commentary systems in other domains.

The power of GameCast comes from the vast amounts of data used to contextualize plays. However, we remark that acquiring such data is still a challenging and expensive task. In the next chapter, we present a manual annotation system to facilitate the collection of trajectory data in sports.

Chapter 7

HistoryTracker: Minimizing Human Interactions in Baseball Game Annotation

7.1 Introduction

Sports analytics changed the way sports are played, planned and watched. Furthermore, the demand for precise, accurate and consistent data is higher than it ever was. While teams and sport organizations rely on multiple sources of data, such as smart watches, heart rate monitors and sensing textiles [61, 63], tracking data produced by specialized tracking systems may be considered the primary source of data in professional sports. Modern tracking systems make use of specialized sensors, such as high-definition cameras, speed radars or RFID technology, in order to collect movement data with precise measumerements and high sampling rates [27, 74]. Some examples of commercial tracking technologies are Pitch F/X and ChyronHego for baseball [21, 30], and STATS Sport VU for soccer, basketball and American football [80].

Tracking systems produce a valuable stream of data for analysis by sports teams. However, implementing and maintaining these systems pose three major difficulties. 1) They are expensive: Major League Baseball's Statcast, for example, was an investment of tens of millions of dollars [86]. Such cost may not be a problem for professional sports teams and leagues, but they are likely unattainable for smaller

organizations or amateurs. 2) The quality of the tracking data is often affected by multiple hard-to-control factors [7, 27, 41, 64], including changes in lighting, camera position in relation to the field, occlusion and small objects, all of which can result in missing or noisy data, and 3) these systems cannot be used to produce tracking data for historical plays. At the same time, commentators and analysts often reference older games during their analysis. However, if the game happened before the tracking system was implemented, it is not possible to quantitatively compare the plays.

Adding manual annotation is a promising direction to address these issues, and a number of studies have explored how human annotators can be used to create reliable sports data from scratch. Manual annotation can be done by a single annotator [12, 75, 78] or by a collection of annotators through crowdsourcing [64, 89]. While individual manual annotation can be a reliable source of tracking data, it puts a major burden into a single person. Meanwhile, crouwdsourcing systems can split the annotation process into many tasks that can be completed quickly. The downside of this approach is that a large number of volunteers might be necessary to produce reliable data.

In this chapter, we propose a novel methodology for manual tracking of baseball plays that reduces the annotation burden and is more enjoyable to users, in comparison to manual annotation from scratch. Our approach reduces the time needed to produce reliable tracking data by warm-starting the annotation process: instead of annotating trajectories on an empty canvas, users modify existing trajectories to reflect the play they want to annotate. The term "warm-start" is borrowed from machine learning, where it means that the model training started from a better initial point [58]. More specifically, we quickly collect a summary of the play by asking a few easy-to-answer questions, and use this information to recommend a set of similar plays that have already been tracked and can be used as an initial approximation. Our method produces reliable annotations at a lower cost and can be used to annotate historical plays that would be otherwise lost for quantitative analysis. Our user studies show that warm-starting the annotation of baseball plays reduces the time needed to generate the hand-annotated tracking data and has an equivalent performance to manually annotating plays from scratch. The results described in this chapter have been published in [68].

Contributions: Our contributions are three-fold: 1) we present a novel methodology for acquiring tracking data that is more reliable and faster than manual annotation from scratch. 2) we describe HistoryTracker, a system that implements our warm-starting methodology for baseball tracking. 3) we present our quantitative and qualitative results, showing that our method is able to produce reliable annotations in a shorter amount of time, and make the annotation process more enjoyable to the users.

The rest of this chapter is organized as follows. Section 7.2 presents our tracking methodology and the design choices we made to implement it in HistoryTracker. We evaluate our methodology in Section 7.3 and discuss our results in Section 7.4.

7.2 HistoryTracker: Tracking System with Warm-Start

Hand annotating sports from scratch is a difficult and time-consuming task commonly done offline by experts, who have to repeatedly watch recordings of the games in order to produce a good approximation of the players movement [74]. While more recent work has focused in transforming the annotation effort into micro-tasks distributed across a large number of crowdsourced annotators, this approach relies on a massive number of workers to produce reliable tracking data. We propose a methodology to enable quick single-user manual tracking of baseball plays, by introducing a warm-starting step to the annotation process.

Our approach consists of three steps: 1) *Fast play retrieval*: we present a video of the play of interest to the user, and ask them questions that they can quickly answer based on the footage (Figure 7.3A). This information is used to retrieve a collection of similar trajectories from the game corpus dataset (Figure 7.3C). 2) *Automatic tuning*: the user can refine the search by aligning the event icons with the events in the video and performing a temporal query in the data, i.e. a query indexed by the event times. We use the aligned events to automatically tune the retrieved trajectory and make it more similar to the video (Figure 7.3D). 3) *Refinement on demand*: the retrieved trajectory is used to warm-start the manual annotation and the user is asked to manually fix the trajectory where it does not match the video.

Below, we describe this methodology for play annotation.

7.2.1 Play Description and Fast Retrieval

In order to warm-start the annotation process, we search our historical trajectory dataset for plays with a similar structure as the one being annotated. We use a query based approach similar to [101] to retrieve the similar plays. Our approach, however, is not based on video features, but historical tracking data instead. The broadcasting videos we take as input are focused on actions, and show only the players that have an impact on the play outcome. On baseball broadcasting, specifically, these actions usually include players contouring bases, throws, catches, tags, etc. The challenge is then to build a mapping from actions that may be identified on videos to a list of plays. These plays should be similar to the play from where the actions were identified, on both the actions performed and on the movements of the players. In order to implement such a mapping, we need a way to represent baseball plays by the actions that are performed by the players. In baseball, just like most sports, the tracking data of a play is given as a colletion of 2D time series data representising player movement, 3D time series of ball positioning, high-level game events and play metadata (see [27] for details).

The game events are pairs (action, player) that refer to specific actions that give context to the tracking data, like the moment the ball was pitched, hit, caught or thrown by a player, etc. By themselves, the game events offer a high level representation of the play that is close to what is necessary for building the query. This representation only lacks information about the geometry of the play (trajectories of the targets), which would help to narrow the search down to plays where the targets movement resembles what is observed in the video. We then propose an augmented set of events to represent plays, with new events that represent more details of the way the players move on top of the the original set of events as illustrated in Figure 7.1.

Once the play representation is defined, the straightforward approach would be to ask the user about the events that may be seen on the video (Figure 7.3A). The query is then built on questions that guide the user in the process of looking for the events that would lead to similar plays on the database. We have worked with baseball experts in order to select a group of questions that effectively summarize

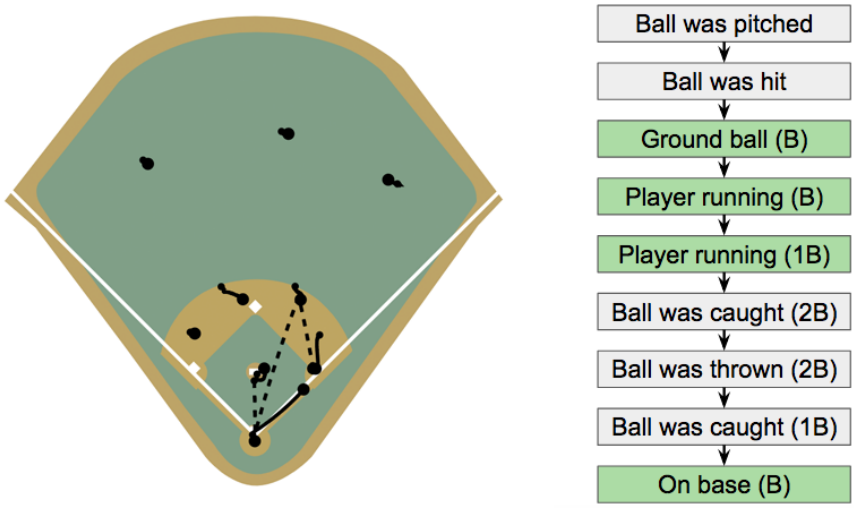


Figure 7.1: An example of a play (left) and the resulting set of events (right). The original set of events, shown in gray, is focused on the representation of the interaction between the players and the ball. We propose the representation of plays by an augmented set of events, shown in green, which encompass information about both the actions and the movements of the players.

baseball plays: 1) Who ran? 2) Who are stealing bases? 3) What are the runners end bases? 5) Who caught the batted ball in flight? 6) Who threw the ball? and 7) What is the hit type? The questions are ordered by the impact on the overall trajectory data, allowing for a trajectory approximation to be generated as early as possible in the process. We accomplish this by first asking questions directly related to the play outcome (i.e. number of runs), and leaving play detail questions to the end. The set of events is then converted to a play index where each pair (event, target) is associated to a bit sequence, as illustrated in Figure 7.2.

This approach leads to the clustering of plays by similarity, given by the way the augmented set of events was designed. Since the augmented set of events contain information about both the actions and the geometry of the play, each cluster contains plays that are similar in both actions and geometry. The events and the clusters of plays were designed to accommodate small differences in the play geometry, in a trade-off between the amount of information that will be requested from the user for the query and the usefulness of the plays returned by the query. Empirically, the first play returned by the system is a good approximation of the actions and movements observed in the video. If the user chooses to inspect other

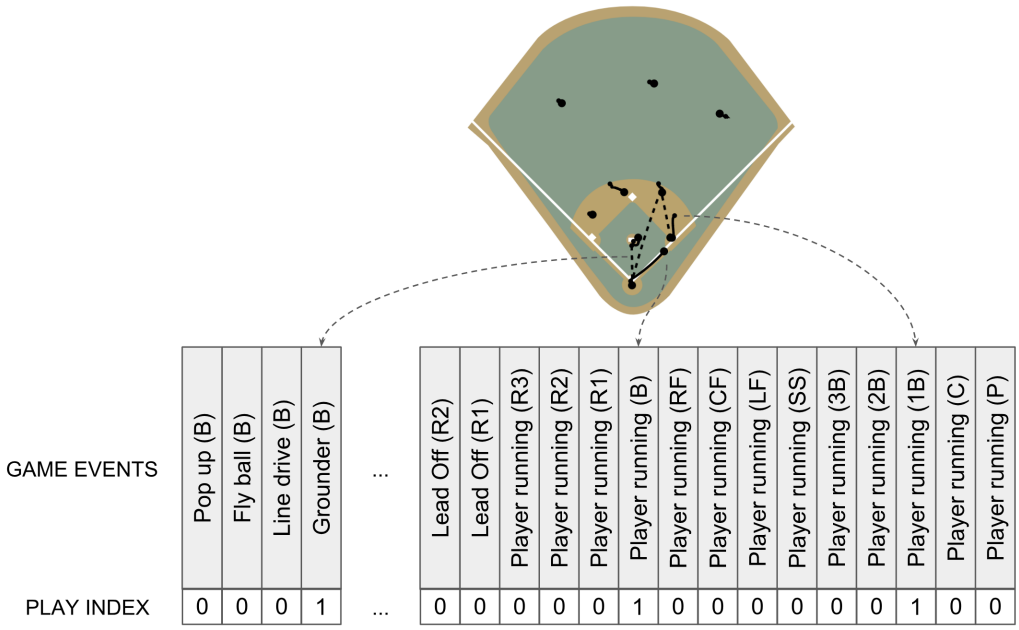


Figure 7.2: The events of a play are converted to a index, where each bit is associated to a pair (event, player). The index is the key to retrieve similar plays, where the similarity is given by how much information about the play is captured by this group of events.

plays in searching for a better one, the variability among them reduces the number of plays to be inspected.

The user query might result in an index for which there are no exact cluster matches in the database. In order to retrieve the most similar cluster to the user query, we select the cluster with the largest number of bits in common with the query. We also allow the users to increase the importance of some of the questions. For example, if the user wants to make sure only the selected players ran during the play, he can increase the weight of the question "Who ran?". Let n be the number of questions, Q be the query bits, W be the bits' weights, X be a cluster in the database, and $\mathbb{1}$ be the indicator function, the similarity between Q and X is given by:

$$S = \sum_{i=1}^{n} W_i \times \mathbb{1}(Q_i = X_i)$$

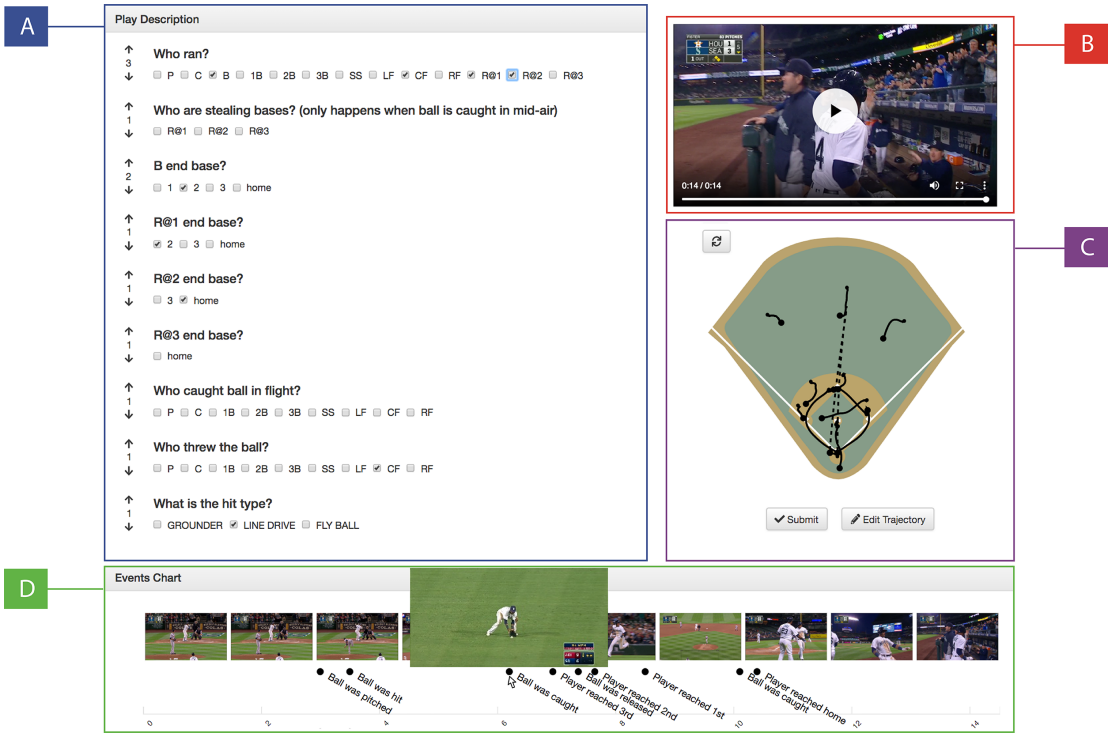


Figure 7.3: HistoryTracker system. A: Users can create a description of the play based on simple questions. B) A video of the play to be annotated. C) Trajectories are recommended based on the play description provided by the user. D) Events can be used to create a more fine grained query of the play.

7.2.2 Automatic Trajectory Tuning Based on Play Events

After a description of the play is collected using our query interface (Figure 7.3A), our system recommends a cluster of trajectories that respects the specified event constraints. A random trajectory within this cluster is displayed to the user, as shown in Figure 7.3C. If this trajectory does not represent correctly the play, the user has three options: 1) change the weight of the questions in Play Description, in order to retrieve a better cluster for the play. 2) Click the switch button (top left corner of Figure 7.3C) to select another random play from the cluster. 3) Use the Events Chart (Figure 7.3D) to query this cluster based on event times.

In the Events Chart view, the main game events are displayed. In order to align the events of the trajectory data with the video, we use the *sound of the baseball hit in the video*. If the video contains a batting event (bat hits ball), we can detect the precise moment of the batting event in the audio signal and we can use this

information to align the event data with the video content. To achieve this, we treat the problem as an audio onset detection problem under the assumption that the batting event corresponds to the strongest onset (impulsive sound) in the audio signal. We use the superflux algorithm for onset detection [11] as implemented in the librosa audio processing library [49] to compute an onset strength envelope representing the strength of onsets in the signal at every moment in time. For the analysis we use a window size of 512 samples and a hop size of 256 samples, where the sampling rate of the audio signal is 44,100 Hz, leaving all other parameters at their default values. To evaluate the approach, we manually annotated a validation set of 311 audio recordings with the timestamp of the batting event, and compared the output of our detection method to the annotations, where we consider the output to be correct if it is within 100 ms of the annotated value. Applying the approach the audio recordings achieved an accuracy of 94.5%, which we deem appropriate for our application. If the video does not have a batting event, we let the user perform the event alignment manually.

The user can drag and drop game events across the time axis and query for a play that respects the time at which these events happened. Once the user starts dragging an event, an image with the current video frame will be positioned over the user's mouse, enabling him to identify exactly when this event happened in the play. For example, if the user wants to specify the time at which the ball was caught and search for this event in the cluster, he should drag the event "Ball was Caught" in the Events Chart so that it aligns with the player action in the video (Figure 7.3D).

Our system adapts the retrieved trajectory so that it respects the event "Ball was pitched" in the Events Chart. In order to do so, we shift the retrieved trajectory so that the pitched event matches the one specified in the trajectory view. This action is a simple trajectory preprocessing step, but it allows us to quickly align the begin-of-play on the retrieved trajectory with the begin-of-play in the video.

By querying the data with the Events Chart, the user can obtain a better initial trajectory to warm-start their annotations. After this step is completed, the user has two options: If they are satisfied with the retrieved trajectory and think it perfectly matches the play in the video, they can click the submit button to save the new trajectory to the disk. Otherwise, the user can click the "Edit Trajectory"

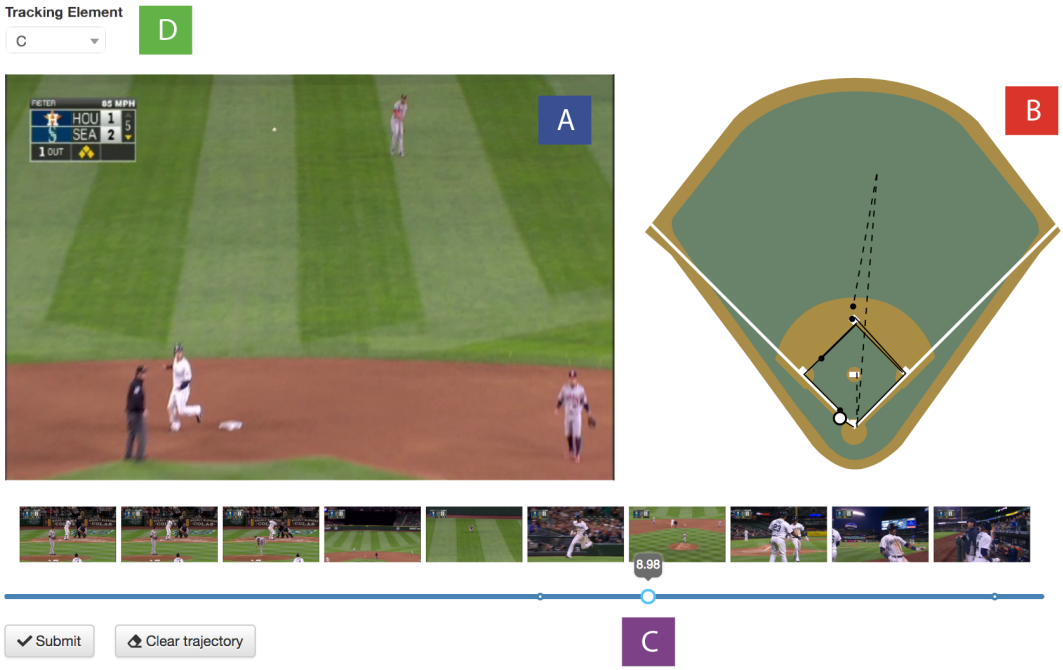


Figure 7.4: Manual tracking system. A) video playback screen. B) Play diagram for position input. C) Video playback slider. D) Tracking element selector.

button and manually change the positions of players or ball that do not reflect the elements in the video.

7.2.3 Refinement on Demand: Manual Annotation

We implemented a hand annotation system that allows users to edit and refine the previously recommended trajectories (Figure 7.4). The system is comprised of four parts: A) the video playback screen; B) the play diagram in which the user annotates the current player position; C) the video playback slider; and D) the tracking element selector.

The trajectory annotation process is straightforward. The user positions the video at a frame of interest (keyframe) using the playback slider (Figure 7.4C), and marks the player position in the field by selecting the same position in the play diagram (Figure 7.4B). Consecutive keyframes are linearly interpolated, generating the tracking data. After the annotation of a player / ball is completed, the user can annotate the next player by selecting it in the Tracking element selector (Figure

7.4D).

If the user determines that the warm-start trajectory for an element is wrong, they can click on the button "Clear Trajectory" to delete the keyframes from the current element trajectory and start the annotation again.

7.3 Evaluation

In order to evaluate our annotation methodology, we compared it to manual tracking with no warm-start, hereby called Baseline. Ten plays were selected to be used in our evaluation: we attempted to maximize the variability of the play configurations in our sample, regarding the number of players, events, and outcomes. Our system was evaluated with 8 users: half of them (type A) annotated the odd plays using HistoryTracker and the even plays using Baseline, while the other half (type B) did the inverse (even plays using HistoryTracker, odd plays using Baseline). In total, 80 play annotations were produced, 40 with HistoryTracker and 40 with the Baseline. Users were recruited through email lists; the only condition for participation was having followed baseball for a minimum of one year. The age of the users varied from 19 to 39, with the majority being in their 20's. Although most were not involved with baseball professionally (most of our users were students and researchers) they all professed to having a deep understanding of baseball. In the screening questionnaire, the users reported having a knowledge of baseball of 8.12 +/- 1.35, on a scale of 1 to 10.

In this section, we analyze the tracking results with respect to the tracking error and the annotation time. We also perform a qualitative analysis of the system and present the results of a likert scale questionnaire we applied to the users after they performed the plays annotation.

7.3.1 Analysis of Plays

In this section, we present the ten plays that were used for the generation of tracking data using HistoryTracker and Baseline. All plays were used for both annotation methods, with each user annotating the same play only once. Figure 7.5 shows the ten plays using Baseball Timeline [62], a spatio-temporal visualization

that represents the position of the players with respect to how close they are to the bases, as well as ball possessions, throws and hits. In this visualization, player position is represented in the Y axis and time, in the X axis.

[**Play 1**] The first play is from the fourth inning of the Philadelphia Phillies versus the Atlanta Braves, June 6, 2017. Batter Maikel Franco grounds the ball softly towards first base. First baseman Matt Adams catches the ball, runs to first base and the batter is out. This is a relatively simple play.

[**Play 2**] This play is from the third inning of the Chicago White Sox vs Detroit Tigers game, June 27, 2015. Alexei Ramirez hits a ground ball, second baseman Ian Kinsler catches it, throws to first baseman Miguel Cabrera and the batter is out.

[**Play 3**] Ninth inning of the Texas Rangers versus the Colorado Rockies, August 9, 2016. There are runners on first and second base. Batter Gerardo Parra grounds the ball out, where it is caught by second baseman Rougned Odor who throws it to first baseman Mitch Moreland to get the batter out. Runners at first and second advance one base.

[**Play 4**] Third inning of New York Yankees versus the Minnesota Twins, July 26, 2015. Baseman Mark Teixeira flies the ball out to right fielder Torii Hunter and reaches first base. The runner on second base reaches third base.

[**Play 5**] Sixth inning of New York Yankees versus Baltimore Orioles, June 13, 2015. Batter Didi Gregorius hits a ground ball. Second baseman Ryan Flaherty catches the ball, throws it to shortstop J. Hardy. The Runner at first is out at 2nd base. Batter saves first base.

[**Play 6**] is from the fourth inning of Chicago Cubs versus the Houston Astros, September 10, 2016. This is a very unique play. With Jose Altuve batting, runner at first Alex Bregman steals second base. Catcher throws the ball the the second baseman.

[**Play 7**] Houston Astros versus New York Yankees, third inning, October 17, 2017. Batter Austin Romine grounds the ball, which is caught by third baseman Alex Bregman who throws it to first baseman Yuli Gurriel. Runner at first Todd Frazier reaches second base.

[**Play 8**] is from the Seattle Mariners versus the New York Yankees, sixth inning, August 27, 2017. Batter Starlin Castro hits a line drive towards the center of the

outfield, where it is caught by center fielder Guillermo Heredia. Batter reaches first base.

[**Play 9**] First inning of the Pittsburgh Pirates versus the Colorado Rockies, September 24, 2015. Batter Starling Marte grounds the ball towards left fielder Rafael Ynoa. Runner at second Jaff Decker reaches home plate. Runner at first Andrew McCutchen reaches second base while the batter reaches first base.

[**Play 10**] The final play is from the seventh inning of the San Diego Padres versus the Baltimore Orioles, June 22, 2016. Batter Yangervis Solarte hits a soft line drive towards right fielder Mark Trumbo, whose fielding error allows second baseman Matt Kemp to reach home plate. The batter reaches 1st base.

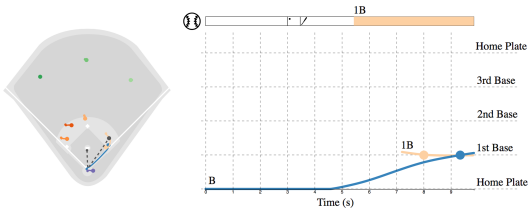
7.3.2 Quantitative Analysis

In this section, we compare the quality and annotation time of the tracking data produced by HistoryTracker and the tracking data acquired with the Baseline manual annotation tool. In order to compare the quality of the annotations, we computed the average Euclidean distance of the annotations to the ground truth produced by MLB Statcast, by averaging the distances between the annotated position and the ground truth position at every sampled time step. Denoting the set of play elements (players and ball) as E and the set of times over which the positions are sampled as T, we have the error of the annotated trajectory $\hat{x}$ with respect to ground truth x as:

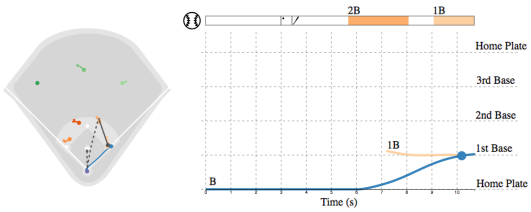
$$Error = \frac{\sum_{e \in E} \sum_{t \in T} ||x_t^e - \hat{x}_t^e||}{|E| \times |T|}$$

The comparison between error results of HistoryTracker and Baseline are shown in Figure 7.6(a). HistoryTracker performs significantly better than the Baseline, exhibiting about 20% lower median error, with the same amount of spread. Furthermore, about 40% of the tracking data generated by History Tracker has a lower error than the lowest error generated by the Baseline.

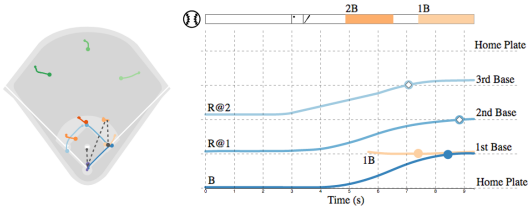
We also compared the play annotation time of HistoryTracker and Baseline. Figure 7.6(b) shows the time taken to complete the annotations of every play with both tools. The median time taken to annotate using the HistoryTracker was about 1.5 minutes less than with the Baseline. Overall, our user study indicates that



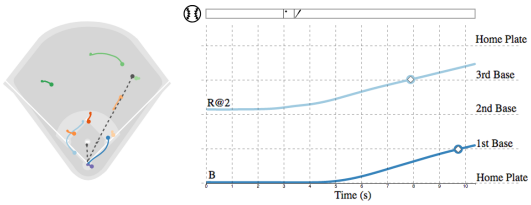
(a) [Play 1] Ground ball to first baseman.



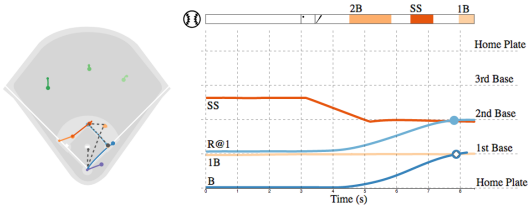
(b) [Play 2] Ground ball to second and first basemen.



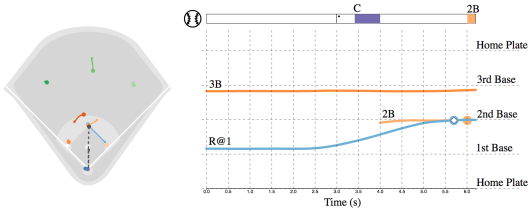
(c) [Play 3] Runners reach second and third bases.



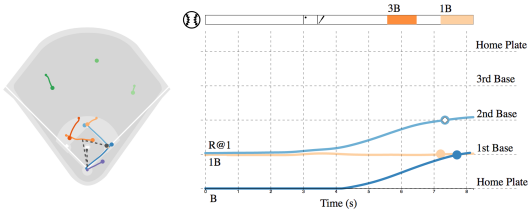
(d) [Play 4] Batter and runner at second advance one base.



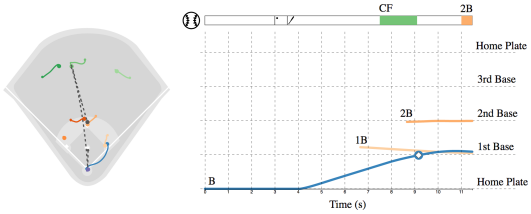
(e) [Play 5] Runner at first out. Batter advances one base.



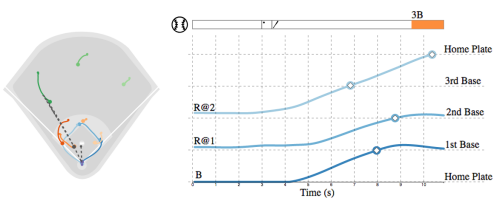
(f) [Play 6] Runner at first steals second base.



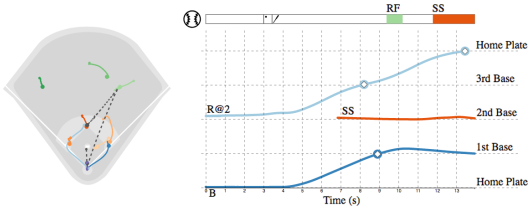
(g) [Play 7] Batter out. Runner at first steals second.



(h) [Play 8] Line drive to center fielder.

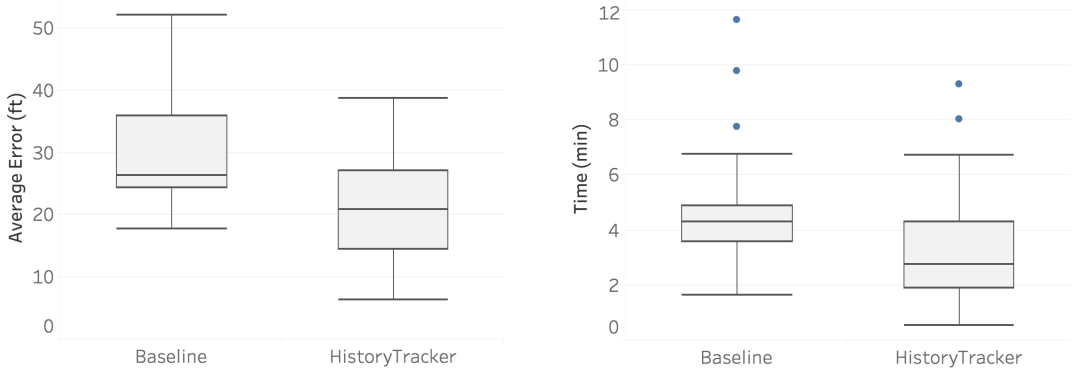


(i) [Play 9] Three players advance bases. Runner at second scores a run.



(j) [Play 10] Batter advances to first base. Runner at second scores a run.

Figure 7.5: Graphical representation of the evaluation plays using Baseball Timeline [62].



(a) Error comparison between HistoryTracker and manual tracking from scratch (Baseline).

(b) Time comparison between HistoryTracker and manual tracking from scratch (Baseline).

Figure 7.6: Quantitative Analysis of the HistoryTracker system.

	Strong. Disagree	Disagree	Neutral	Agree	Strong. Agree
1) I feel confident about the tracking data I produced using HistoryTracker.				6 (75%)	2 (25%)
2) I feel confident about the tracking data I produced using Baseline.			2 (25%)	5 (62.5%)	1 (12.5%)
3) I feel like HistoryTracker allowed me to annotate baseball plays in a faster manner, in comparison to Baseline.				3 (37.5%)	5 (62.5%)
4) I feel like HistoryTracker is easy to understand and use.			1 (12.5%)	3 (37.5%)	4 (50%)
5) I feel like Baseline is easy to understand and use.			2 (25%)	3 (37.5%)	3 (37.5%)

Table 7.1: User perception of the system, using a 5 point likert scale.

warm-starting play annotations make users more efficient and accurate.

7.3.3 User Feedback

After the users performed the play annotations with the Baseline and HistoryTracker, they were asked to fill a likert scale questionnaire, which contained statements regarding their perception of both systems. The users could rank the statements from 1 (strongly disagree) to 5 (strongly agree). The statements and the answers from all the users are presented in Table 7.1.

Our first two statements evaluated how confident users were about their anno-

tations, regarding both HistoryTracker and Baseline. We can see that the users are equally confident on using both systems. However, the third statement shows that all users perceived HistoryTracker to produce the annotations in a shorter amount of time, compared to Baseline. Regarding ease of use, the fourth and fifth statements show that the users thought both systems were equally easy to understand and use. Therefore, from a perceptual standpoint, using HistoryTracker makes the annotation process faster, with no loss in the perception of difficulty or quality of annotations.

Overall, the HistoryTracker was well received by our users. Two annotators noted that HistoryTracker offered a good initial approximation of the plays. Another user mentioned that they spent less time annotating common movements, such as pitching and the batter running to first base, because a lot of the work was filled in already.

7.4 Final Considerations

It is widely accepted that sports tracking data has been revolutionizing sports analytics with its unprecedented level of detail. Instead of relying on derived statistics, experts can use that data to "reconstruct reality" and create their own statistics or analysis without prior constraints [27]. Moreover, tracking data can be used for training "simulation engines", that can predict game developments and enable new hypothesis to be tested [75]. Unfortunately, this data can be expensive to acquire, either requiring multi-million dollar investments in infrastructure and services (e.g., MLBAM Statcast or the NFL's Zebra tracking system), or systems that use manual annotators, which tends to be tedious and require many passes to generate high-quality data.

In this chapter, we proposed to use knowledge already acquired to lower the cost of future data acquisition. This is intuitively a very simple idea. We present HistoryTracker, a tool that takes broadcast video from baseball games, and with much lower level of user input, is able to generate high-quality tracking data. The system automates many of the tedious tasks by leveraging information retrieval techniques on a corpus of previously acquired tracking data. We presented a tool tailored for baseball, but we believe HistoryTracker could be extended for other

domains. Extending our tool for other sports, such as soccer and basketball, is straightforward: we only need a set of events to describe plays and some historical tracking data. One can imagine applying the warm-starting procedure to non-sports domains as well. For example, we can use historical information to help annotating semantic image segmentation datasets [3]. Pixel-wise image annotation is a time consuming task, so it would greatly benefit from our warm-starting methodology. As future work, we would like to investigate how to use historical data to initialize image annotation tasks.

Furthermore, we believe systems such as ours can be used in novel applications, for instance, annotating historical video collections can potentially be used for generating statistics for comparing how players performance changes over time; or the system can be used for college or high-school video collections, enabling parents (or coaches) to track the performance of players as they mature.

There are many opportunities for improvements. An obvious extension of HistoryTracker would be to make it into a crowdsourcing tool that could potentially be used during live events. Among the challenges, we would need to research the best way to integrate multiple people's input, including potentially providing an intelligent interface that would update as others make edits to the play. Supporting multiple sports is also another obvious extension. We would also like to explore introducing more intelligence into the system as to further simplify the role of the user.

Chapter 8

Conclusions and Future Work

This dissertation presented five main contributions to the interactive collection, organization, and analysis of sports datasets. The first contribution was StatCast Dashboard [41], a visual analytics system for the exploration of baseball trajectory and statistics data. With our tool, sabermetricians and statisticians could query and visualize extensive collections of baseball plays using a simple graphical interface that allowed them to compare different teams, players, and playing strategies. The second contribution was Baseball Timeline [62], a visualization for baseball play trajectories that took into consideration both the spatial and temporal aspects of the data. Our timeline representation enabled users to have a more accurate and thorough understanding of the evolution of the play. The third contribution was TrackRuler, a generalization of Baseball Timeline to sports trajectory data. TrackRuler makes no assumption on the movement of the players and ball. Therefore, it can represent trajectories in any team sport. We demonstrated this visualization with data from soccer, Counter-Strike and baseball games. The fourth contribution was GameCast, a tool for the visualization of live games that automatically generates sports rankings, highlights, and commentaries for the user. While traditional sports ranking systems relied solely on statistics to compare players, our approach considered the context of the play, resulting in a more fair comparison. Finally, the fifth contribution was HistoryTracker [68], a system that facilitated the manual annotation of trajectory data from sports videos. Because manual annotation is a costly and time-consuming process, we proposed a methodology that took advantage of historical tracking data to provide the user with an approximation of the play and speed up the game annotation.

Research in sports analytics is still in its early stages, and many challenges remain, including how to support the inference and what-if analysis on trajectory data, how to facilitate and improve data collection and annotation, and how to integrate the sports analytics tools into the stakeholders' workflow. Next, we present some of these challenges and highlight exciting directions of future research in visual analytics for sports.

What-if analysis. Recent ML models are able synthesize play trajectories from scratch [42, 43, 99]. By changing the initial play configurations (player positions and stats), one can perform complex what-if analyses on games, for example, "what would happen if the ball went 5 ft further?". While these models can generate realistic trajectories, their results are stochastic by nature, and they do not provide any information about the distribution of the possible game outcomes. An interesting direction is investigating how visual analytics can help users explore the space of outcomes in sports simulations. Moreover, by enabling users to change initial configurations interactively, we can facilitate the creation of what-if scenarios and provide a more thorough analysis of the game.

Improving data collection. In this thesis, we have proposed a methodology to speed up the manual annotation of sports trajectories. While HistoryTracker produces high-quality tracking data in a short amount of time, there are many avenues to improve the work, including crowdsourcing, semi-automated annotation, and the support for multi-camera annotation. *Crowdsourcing* has been used in a variety of sports data annotation tasks with successful results [64, 83, 88, 89]: not only does it reduce the individual effort for annotation, but also it increases the data quality by averaging out the errors. HistoryTracker currently does not support crowdsourcing, but adding this feature would allow us to produce better tracking data in a shorter amount of time. *Semi-automated* annotation methods use machine learning, image processing, and retrieval techniques to reduce the human annotation effort [28, 60]. We believe HistoryTracker can be improved by using some of these ideas. For example, we can use image processing to identify key events in the video (e.g., ball pitch or player reaching a base). Moreover, we can use the historical tracking data to train a model that completes the trajectories as soon as the user starts drawing them on the screen, similarly to handwriting completion models [35]. Finally, sports games are usually recorded from multiple

points of view. We can take advantage of the replicated video footage to allow users to perform *multi-camera annotation* [51, 60, 87]. Since sports videos normally focus on a small region of the field, allowing the user to see the same play from multiple perspectives will result in a more comprehensive understanding of the game events and player actions. For multi-camera annotation to work properly, some challenges need to be addressed, including video synchronization and context switching between different cameras.

Adoption by domain experts. The visual analytics systems proposed in this dissertation are built as standalone tools that stakeholders can use in the browser. When a more thorough analysis is needed, data can be exported to a standard format, such as CSV or JSON [41, 68], and imported into the analytics tool of choice, such as Jupyter Notebooks or Excel. Although analysts can incorporate this approach into their workflows, the frequent context switching can hamper the data exploration. An exciting research direction is to develop strategies to incorporate our systems into the analysts' workflow. For example, most baseball experts we have collaborated with were familiar with Python and the Jupyter environment. As future work, we would like to investigate whether interactive Jupyter visualizations [37] can improve the usage and adoption of our tools by domain experts.

Bibliography

[1] S. Agarwal, G. Wallner, and F. Beck. Bombalytics: Visualization of competition and collaboration strategies of players in a bomb laying game. In *Computer Graphics Forum*, volume 39, pages 89–100. Wiley Online Library, 2020.

[2] P.-A. Albinsson and D. Andersson. Extending the Attribute Explorer to Support Professional Team-Sport Analysis. *Information Visualization*, 7(2):163–169, 2008.

[3] Alexander Klaser. LEAR - Image Annotation Tool. `https://lear.inrialpes.fr/people/klaeser/software_image_annotation`, 2010. [Online; accessed 14-Feb-2021].

[4] N. D. Allen, J. R. Templon, P. S. McNally, L. Birnbaum, and K. Hammond. Statsmonkey: A data-driven sports narrative writer. In *2010 AAAI Fall Symposium Series*, 2010.

[5] G. Andrienko, N. Andrienko, P. Bak, D. Keim, and S. Wrobel. *Visual Analytics of Movement*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[6] N. Andrienko, G. Andrienko, H. Stange, T. Liebig, and D. Hecker. Visual Analytics for Understanding Spatial Situations from Episodic Movement Data. *KI - Künstliche Intelligenz*, 26(3):241–251, 2012.

[7] R. Arthur. MLB's hit-tracking tool misses a lot of hits. `https://fivethirtyeight.com/features/mlbs-hit-tracking-tool-misses-a-lot-of-hits/`, 2016. [Online; accessed 14-Feb-2021].

[8] W. Aspray and B. M. Hayes. *Everyday Information: The evolution of information seeking in America*. MIT Press, 2011.

[9] B. Bach, P. Dragicevic, D. Archambault, C. Hurter, and S. Carpendale. A review of temporal data visualizations based on space-time cube operations. In *Eurographics conference on visualization*, 2014.

[10] R. C. Basole and D. Saupe. Sports data visualization [guest editors' introduction]. *IEEE Computer Graphics and Applications*, 36(5):24–26, 2016.

[11] S. Bock and G. Widmer. Maximum Filter Vibrato Suppression for Onset Detection. In *Proc. of the 16th Int. Conf. on Digital Audio Effects (DAFx). Maynooth, Ireland (Sept 2013)*, page 7, 2013.

[12] G. C. Bogdanis, V. Ziagos, M. Anastasiadis, and M. Maridaki. Effects of two different short-term training programs on the physical and technical abilities of adolescent basketball players. *Journal of Science and Medicine in Sport*, 10(2):79–88, 2007.

[13] J. Borg. Detecting and Tracking Players in Football Using Stereo Vision. Master's thesis, Department of Electrical Engineering, Linköping University, Sweden, 2007.

[14] M. Bostock, V. Ogievetsky, and J. Heer. D^3 data-driven documents. *IEEE transactions on visualization and computer graphics*, 17(12):2301–2309, 2011.

[15] M. Brown. ESPN pulls out all the tech stops for MLB AL wild-card game, with Statcast AI on ESPN2. `https://www.forbes.com/sites/maurybrown/2019/10/01/espn-pulls-out-the-technology-stops-for-mlb-al-wild-card-includes-statcast-ai-on-espn2`, 2019. [Online; accessed 14-Feb-2021].

[16] A. Bucholtz. ESPN2's NL wild card game Statcast broadcast drew rave reviews. `https://awfulannouncing.com/espn/espn2s-wild-card-statcast-broadcast-drew-rave-reviews.html`, 2018. [Online; accessed 14-Feb-2021].

[17] D. Cervone, A. D'Amour, L. Bornn, and K. Goldsberry. POINTWISE: Predicting points and valuing decisions in real time with NBA Optical Tracking Data. In *MIT Sloan Sports Analytics Conference*, volume 28, 2014.

[18] W. Chen, T. Lao, J. Xia, X. Huang, B. Zhu, W. Hu, and H. Guan. GameFlow: Narrative Visualization of NBA Basketball Games. *IEEE Transactions on Multimedia*, 18(11):2247–2256, 2016.

[19] D. H. Chung, P. A. Legg, M. L. Parry, R. Bown, I. W. Griffiths, R. S. Laramee, and M. Chen. Glyph sorting: Interactive visualization for multi-dimensional data. *Information Visualization*, 14(1):76–90, 2015.

[20] D. H. S. Chung, M. L. Parry, I. W. Griffiths, R. S. Laramee, R. Bown, P. A. Legg, and M. Chen. Knowledge-assisted ranking: A visual analytic application for sports event data. *IEEE Computer Graphics and Applications*, 36(3):72–82, 2016.

[21] ChyronHego. TRACAB Optical Tracking. `https://chyronhego.com/content/tracab-player-tracking/`, 2016. [Online; accessed 14-Feb-2021].

[22] J. G. Claudino, D. de Oliveira Capanema, T. V. de Souza, J. C. Serrão, A. C. M. Pereira, and G. P. Nassis. Current approaches to the use of artificial intelligence for injury risk assessment and performance prediction in team sports: a systematic review. *Sports medicine-open*, 5(1):28, 2019.

[23] A. Cox and J. Stasko. Sportsvis: Discovering meaning in sports statistics through information visualization. In *Compendium of Symposium on Information Visualization*, pages 114–115. Citeseer, Citeseer, 2006.

[24] T. Crnovrsanin, C. Muelder, C. Correa, and K.-L. Ma. Proximity-based visualization of movement trace data. In *2009 IEEE symposium on visual analytics science and technology*, pages 11–18. IEEE, 2009.

[25] J. Cross and D. Sylvan. Modeling spatial batting ability using a known covariance matrix. *Journal of Quantitative Analysis in Sports*, 11(3):155–167, 2015.

[26] M. Di, D. Klabjan, L. Sha, and P. Lucey. Large-scale adversarial sports play retrieval with learning to rank. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 12(6):1–18, 2018.

[27] C. Dietrich, D. Koop, H. T. Vo, and C. T. Silva. Baseball4d: A tool for baseball game reconstruction & visualization. In *2014 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 23–32. IEEE, 2014.

[28] T. D'Orazio, M. Leo, N. Mosca, P. Spagnolo, and P. L. Mazzeo. A semi-automatic system for ground truth generation of soccer video sequences. In *2009 Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 559–564. IEEE, 2009.

[29] ESPN. Player tracking transforming NBA analytics. `http://www.espn.com/blog/playbook/tech/post/_/id/492/492`, 2012. [Online; accessed 14-Feb-2021].

[30] M. Fast. What the heck is pitchf/x. *The Hardball Times Annual*, 2010:153–158, 2010.

[31] FIFA. *Laws of the Game 2020/21.* The International Football Association Board, Zurich, Switzerland, 1st edition, June 2020.

[32] M. Fleischman and D. Roy. Unsupervised Content-based Indexing of Sports Video. In *Proceedings of the International Workshop on Workshop on Multimedia Information Retrieval*, MIR '07, pages 87–94, New York, NY, USA, 2007. ACM.

[33] C. Gackenheimer. *Introduction to React.* Apress, 2015.

[34] K. Goldsberry. Courtvision: New visual and spatial analytics for the nba. In *MIT Sloan Sports Analytics Conference*. MIT Sloan Sports Analytics Conference, 2012.

[35] A. Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[36] T. Hagerstraand. What About People in Regional Science? *Papers in Regional Science*, 24(1):7–24, 1970.

[37] Jorge Piazentin Ono, Juliana Freire, and Claudio Silva. Interactive Data Visualization in Jupyter Notebooks. *Computing in Science & Engineering*, 23(2):99–106, 2021.

[38] P. R. Kamble, A. G. Keskar, and K. M. Bhurchandi. Ball tracking in sports: a survey. *Artificial Intelligence Review*, 2017.

[39] Kraak, M. The space-time cube revisited from a geovisualization perspective, 2003. OCLC: 81047846.

[40] K. Kurach, A. Raichuk, P. Stańczyk, M. Zajkac, O. Bachem, L. Espeholt, C. Riquelme, D. Vincent, M. Michalski, O. Bousquet, et al. Google research football: A novel reinforcement learning environment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4501–4510, 2020.

[41] M. Lage, J. P. Ono, D. Cervone, J. Chiang, C. Dietrich, and C. T. Silva. StatCast Dashboard: Exploration of Spatiotemporal Baseball Data. *IEEE Computer Graphics and Applications*, 36(5):28–37, 2016.

[42] H. M. Le, P. Carr, Y. Yue, and P. Lucey. Data-Driven Ghosting using Deep Imitation Learning. In *MIT Sloan Sports Analytics Conference*, page 15, 2017.

[43] H. M. Le, Y. Yue, P. Carr, and P. Lucey. Coordinated multi-agent imitation learning. In *International Conference on Machine Learning*, pages 1995–2003. PMLR, 2017.

[44] G. M. Lee, V. Bulitko, and E. Ludvig. Sports commentary recommendation system (scores): Machine learning for automated narrative. In *Eighth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2012.

[45] A. G. Losada, R. Therón, and A. Benito. BKViz: A Basketball Visual Analysis Tool. *IEEE Computer Graphics and Applications*, 36(6):58–68, 2016.

[46] J. Lukasczyk, R. Maciejewski, C. Garth, and H. Hagen. Understanding hotspots: A topological visual analytics approach. In *SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 36. ACM, 2015.

[47] E.-J. Marey. *La méthode graphique dans les sciences expérimentales et principalement en physiologie et en médecine*. G. Masson, 1878.

[48] J. Matejka, T. Grossman, and G. Fitzmaurice. Video lens: rapid playback and exploration of large video collections and associated metadata. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*, pages 541–550, 2014.

[49] B. McFee, C. Raffel, D. Liang, D. Ellis, M. McVicar, E. Battenberg, and O. Nieto. librosa: Audio and Music Signal Analysis in Python. In *Proceedings of the 14th python in science conference*, page 8, 2015.

[50] R. Metoyer, Q. Zhi, B. Janczuk, and W. Scheirer. Coupling Story to Visualization: Using textual analysis as a bridge between data and interpretation. In *23rd International Conference on Intelligent User Interfaces*, pages 503–507, 2018.

[51] G. Miller, S. Fels, A. Al Hajri, M. Ilich, Z. Foley-Fisher, M. Fernandez, and D. Jang. Mediadiver: Viewing and annotating multi-view video. In *CHI'11 Extended Abstracts on Human Factors in Computing Systems*, pages 1141–1146. ACM, 2011.

[52] MLB. Baseball Savant 3D Home Run Derby. `https://baseballsavant.mlb.com/hr_derby`. [Online; accessed 02-Feb-2021].

[53] MLB. Statcast. `http://m.mlb.com/glossary/statcast`, 2015. [Online; accessed 02-Feb-2021].

[54] MLB. Official baseball rules. `http://mlb.mlb.com/mlb/official_info/official_rules/official_rules.jsp`, 2019. [Online; accessed 21-May-2021].

[55] MLB. Official info: Baseball basics: Score. `https://www.mlb.com/official-information/basics/score`, 2019. [Online; accessed 15-Feb-2021].

[56] B. Moon and R. Brath. Bloomberg sports visualization for pitch analysis. In *Workshop on Sports Data Visualization*, 2013.

[57] T. Munzner. *Visualization analysis and design*. CRC press, 2014.

[58] K. P. Murphy. *Machine learning: a probabilistic perspective*. Adaptive computation and machine learning series. MIT Press, Cambridge, MA, 2012.

[59] NFL. Glossary | NFL Next Gen Stats. `https://nextgenstats.nfl.com/glossary`, 2018. [Online; accessed 14-Feb-2021].

[60] J. Nino-Castaneda, A. Frías-Velázquez, N. B. Bo, M. Slembrouck, J. Guan, G. Debard, B. Vanrumste, T. Tuytelaars, and W. Philips. Scalable semi-automatic annotation for multi-camera person tracking. *IEEE Transactions on Image Processing*, 25(5):2259–2274, 2016.

[61] S. Nylander, J. Tholander, F. Mueller, and J. Marshall. HCI and Sports. In *CHI '14 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '14, pages 115–118, New York, NY, USA, 2014. ACM.

[62] J. P. Ono, C. Dietrich, and C. T. Silva. Baseball Timeline: Summarizing Baseball Plays Into a Static Visualization. *Computer Graphics Forum*, 37(3):491–501, 2018.

[63] T. Page. Applications of Wearable Technology in Elite Sports. *Journal on Mobile Applications and Technologies*, 2(1):1–15, 2015.

[64] C. Perin, R. Vuillemot, and J. D. Fekete. Real-Time Crowdsourcing of Detailed Soccer Data. In *What's the score? The 1st Workshop on Sports Data Visualization*, 2013.

[65] C. Perin, R. Vuillemot, and J. D. Fekete. SoccerStories: A Kick-off for Visual Soccer Analysis. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2506–2515, 2013.

[66] C. Perin, R. Vuillemot, and J.-D. Fekete. À table! improving temporal navigation in soccer ranking tables. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 887–896, 2014.

[67] C. Perin, R. Vuillemot, C. D. Stolper, J. T. Stasko, J. Wood, and S. Carpendale. State of the art of sports data visualization. In *Computer Graphics Forum*, volume 37, pages 663–686. Wiley Online Library, 2018.

[68] J. Piazentin Ono, A. Gjoka, J. Salamon, C. Dietrich, and C. T. Silva. Historytracker: Minimizing human interactions in baseball game annotation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1–12, 2019.

[69] H. Pileggi, C. D. Stolper, J. M. Boyle, and J. T. Stasko. SnapShot: Visualization to Propel Ice Hockey Analytics. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2819–2828, 2012.

[70] G. Pingali, A. Opalach, Y. Jean, and I. Carlbom. Visualization of Sports Using Motion Trajectories: Providing Insights into Performance, Style, and Strategy. In *Conference on Visualization*, VIS '01, pages 75–82, Washington, DC, USA, 2001. IEEE Computer Society.

[71] T. Polk, D. Jäckle, J. Häußler, and J. Yang. Courttime: Generating actionable insights into tennis matches using visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):397–406, 2019.

[72] D. Sacha, F. Al-Masoudi, M. Stein, T. Schreck, D. A. Keim, G. Andrienko, and H. Janetzko. Dynamic visual abstraction of soccer movement. In *Computer Graphics Forum*, volume 36, pages 305–315. Wiley Online Library, 2017.

[73] H. Samet. *The design and analysis of spatial data structures*, volume 199. Addison-Wesley Reading, MA, 1990.

[74] C. B. Santiago, A. Sousa, M. L. Estriga, L. P. Reis, and M. Lames. Survey on team tracking techniques applied to sports. In *2010 International Conference on Autonomous and Intelligent Systems, AIS 2010*, pages 1–6, 2010.

[75] T. Seidl, A. Cherukumudi, A. Hartnett, P. Carr, and P. Lucey. Bhostgusters: Realtime Interactive Play Sketching with Synthesized NBA Defenses. *MIT Sloan Sports Analytics Conference*, page 13, 2018.

[76] L. Sha, P. Lucey, Y. Yue, P. Carr, C. Rohlf, and I. Matthews. Chalkboarding: A new spatiotemporal query paradigm for sports play retrieval. In *Proceedings of the 21st International Conference on Intelligent User Interfaces*, pages 336–347, 2016.

[77] L. Sha, P. Lucey, Y. Yue, X. Wei, J. Hobbs, C. Rohlf, and S. Sridharan. Interactive Sports Analytics: An Intelligent Interface for Utilizing Trajectories for Interactive Sports Play Retrieval and Analytics. *ACM Transactions on Computer-Human Interaction*, 25(2):1–32, 2018.

[78] M. Spencer, C. Rechichi, S. Lawrence, B. Dawson, D. Bishop, and C. Goodman. Time-motion analysis of elite field hockey during several games in succession: a tournament scenario. *Journal of Science and Medicine in Sport*, page 10, 2005.

[79] A. Srinivasan, S. M. Drucker, A. Endert, and J. Stasko. Augmenting visualizations with interactive data facts to facilitate interpretation and communication. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):672–681, 2018.

[80] STATS. SportVU Player Tracking | STATS SportVU Tracking Cameras. `http://www.stats.com/sportvu/sportvu-basketball-media`, 2016. [Online; accessed 14-Feb-2021].

[81] M. Stein, H. Janetzko, T. Breitkreutz, D. Seebacher, T. Schreck, M. Grossniklaus, I. D. Couzin, and D. A. Keim. Director's Cut: Analysis and Annotation of Soccer Matches. *IEEE Computer Graphics and Applications*, 36(5):50–60, 2016.

[82] M. Stein, H. Janetzko, A. Lamprecht, T. Breitkreutz, P. Zimmermann, B. Goldlucke, T. Schreck, G. Andrienko, M. Grossniklaus, and D. A. Keim. Bring it to the Pitch: Combining Video and Movement Data to Enhance Team Sport Analysis. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):13–22, 2018.

[83] A. Tang and S. Boring. # epicplay: Crowd-sourcing sports video highlights. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1569–1572. ACM, 2012.

[84] R. Theron and L. Casares. Visual Analysis of Time-Motion in Basketball Games. In *International Symposium on Smart Graphics*, volume 6133, pages 196–207, 2010.

[85] E. R. Tufte. *The visual display of quantitative information*. Graphics Press Cheshire, CT, 2001.

[86] USAToday. Data deluge: MLB rolls out Statcast analytics on Tuesday. `https://www.usatoday.com/story/sports/mlb/2015/04/20/data-deluge-mlb-rolls-out-statcast-analytics-on-tuesday/26097841/`, 2015. [Online; accessed 14-Feb-2021].

[87] Á. Utasi and C. Benedek. A multi-view annotation tool for people detection evaluation. In *Proceedings of the 1st international workshop on visual interfaces for ground truth collection in computer vision applications*, page 3. ACM, 2012.

[88] G. Van Oorschot, M. Van Erp, and C. Dijkshoorn. Automatic extraction of soccer game events from twitter. *Proceedings of the Workhop on Detection, Representation, and Exploitation of Events in the Semantic Web (DeRiVE 2012)*, 902:21–30, 2012.

[89] C. Vondrick, D. Ramanan, and D. Patterson. Efficiently Scaling Up Video Annotation with Crowdsourced Marketplaces. In *European Conference on Computer Vision*, pages 610–623. Springer, 2010.

[90] J. Wang, K. Zhao, D. Deng, A. Cao, X. Xie, Z. Zhou, H. Zhang, and Y. Wu. Tac-simur: Tactic-based simulative visual analytics of table tennis. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):407–417, 2019.

[91] S. Wiseman, S. M. Shieber, and A. M. Rush. Challenges in data-to-document generation. *arXiv preprint arXiv:1707.08052*, 2017.

[92] K. Wongsuphasawat and D. Gotz. Exploring flow, factors, and outcomes of temporal event sequences with the outflow visualization. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2659–2668, 2012.

[93] A. Woodie. Today's Baseball Analytics Make Moneyball Look Like Child's Play. `https://www.datanami.com/2014/10/24/todays-baseball-analytics-make-moneyball-look-like-childs-play/`, 2014.

[94] J. Wu, Z. Guo, Z. Wang, Q. Xu, and Y. Wu. Visual analytics of multivariate event sequence data in racquet sports. In *2020 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 36–47. IEEE, 2020.

[95] Y. Wu, J. Lan, X. Shu, C. Ji, K. Zhao, J. Wang, and H. Zhang. ittvis: Interactive visualization of table tennis data. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):709–718, 2017.

[96] P. Xenopoulos, H. Doraiswamy, and C. Silva. Valuing player actions in counter-strike: Global offensive. *arXiv preprint arXiv:2011.01324*, 2020.

[97] X. Xie, J. Wang, H. Liang, D. Deng, S. Cheng, H. Zhang, W. Chen, and Y. Wu. Passvizor: Toward better understanding of the dynamics of soccer passes. *IEEE Transactions on Visualization and Computer Graphics*, 2020.

[98] S. Ye, Z. Chen, X. Chu, Y. Wang, S. Fu, L. Shen, K. Zhou, and Y. Wu. Shuttlespace: Exploring and analyzing movement trajectory in immersive visualization. *IEEE Transactions on Visualization and Computer Graphics*, 2020.

[99] R. A. Yeh, A. G. Schwing, J. Huang, and K. Murphy. Diverse generation for multi-agent sports games. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4610–4619, 2019.

[100] Q. Zhi, S. Lin, P. Talkad Sukumar, and R. Metoyer. GameViews: Understanding and supporting data-driven sports storytelling. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1–13, 2019.

[101] W. Zhou, A. Vellaikal, and C. Kuo. Rule-based Video Classification System for Basketball Video Indexing. In *Proceedings of the 2000 ACM Workshops on Multimedia*, MULTIMEDIA '00, pages 213–216, New York, NY, USA, 2000. ACM.