Soccer Data Cleaning and Visualization

A Technical Report submitted to the
Department of Informatics and Networked Systems
at the School of Computing and Information
University of Pittsburgh
Pittsburgh, Pennsylvania

Insert Author Name
Spring 2023

Project Team Members
Daniel Atlantis
Thomas Brusilovsky
Anna Heltz
Peter Magasiny
Zach Palmer

Signature Daniel Antantis Date 04/06/2023
Daniel Antantis

Signature Thomas Brusilovsky Date 04/06/2023
Thomas Brusilovsky

Signature Anna Heltz Date 04/06/2023
Anna Heltz

Signature Peter Masaginy Date 04/06/2023
Peter Masaginy

Signature Zach Palmer Date 04/06/2023
Zach Palmer

Advisor: Dr. Ahmed Ibrahim, Department of Informatics and Networked Systems

Table of Contents

Abstract

Our project facilitates analysis of player performance by coaches and sports scientists of the University of Pittsburgh Men's Soccer Team through two dynamic web applications that enable easy and intuitive data cleaning, transformation, and aggregation, and offer in-depth visualization, exploration, and analysis of said data respectively. Our initial goals for the project involved creating a series of data pipelines and functions that would serve as the backend for an interactive web application that the user would use directly to enter relevant input to tailor figures and results as they desired. Ultimately, we were to create a seamless product that imported data in to perform necessary processing, then used that processed data to support visualization and analysis of relevant variables related to player performance, including velocity, acceleration, and distance traveled, among others. Within the vein of this analysis, we also were asked to prioritize functionality that offered further insight into the high intensity actions, or sprint bursts, performed by players throughout games to provide specialized analysis which could then be used to optimize player training and recovery between games.

To accomplish these goals we utilized a number of tools and technologies, including the power of the R programming language, which is developed specifically for statistical analysis, and the RStudio integrated development environment (IDE) to simplify project development and integration of the various scripts necessary to run the web applications. Additionally, we made use of the Shiny package for R to facilitate development and deployment of the frontend components of the project, and Microsoft Excel to manage the relational database tables that store the player data from individual games as well as locational data for each of the fields these games are played on. With these tools and technologies at our disposal, we built a pipeline of R functions that handle rotation, rescaling, and recentering of player positional data to populate the backend of our applications. With this backend in place, we then populated visuals for display in the frontend and created an intuitive user interactive to collectively offer the user an easy and customizable experience with which they can isolate the players and data they desire for further

investigation. Ultimately, our product combines the functionality overviewed above to provide an auxiliary tool to the coaches and sports scientists of the University of Pittsburgh Men's Soccer Team that hopes to aid in optimizing player performance, training and recovery.

List of Figures

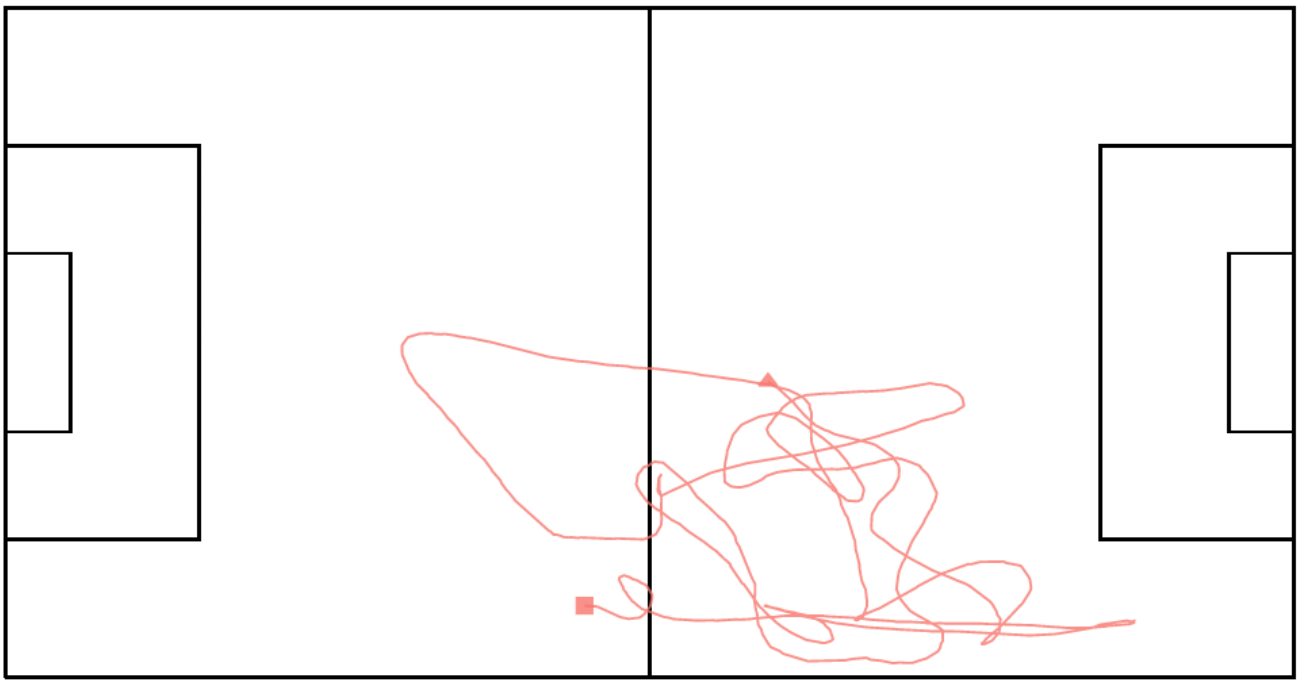


Figure 1. Player Path

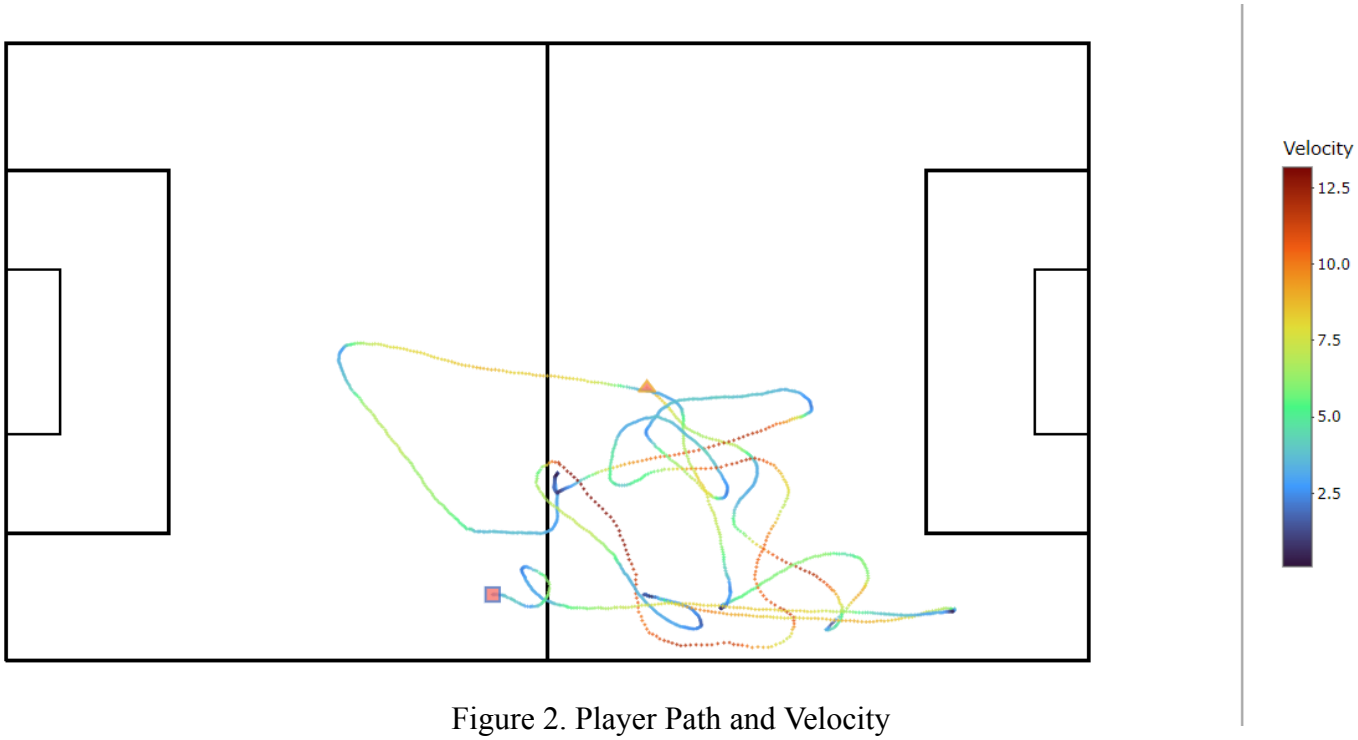


Figure 2. Player Path and Velocity

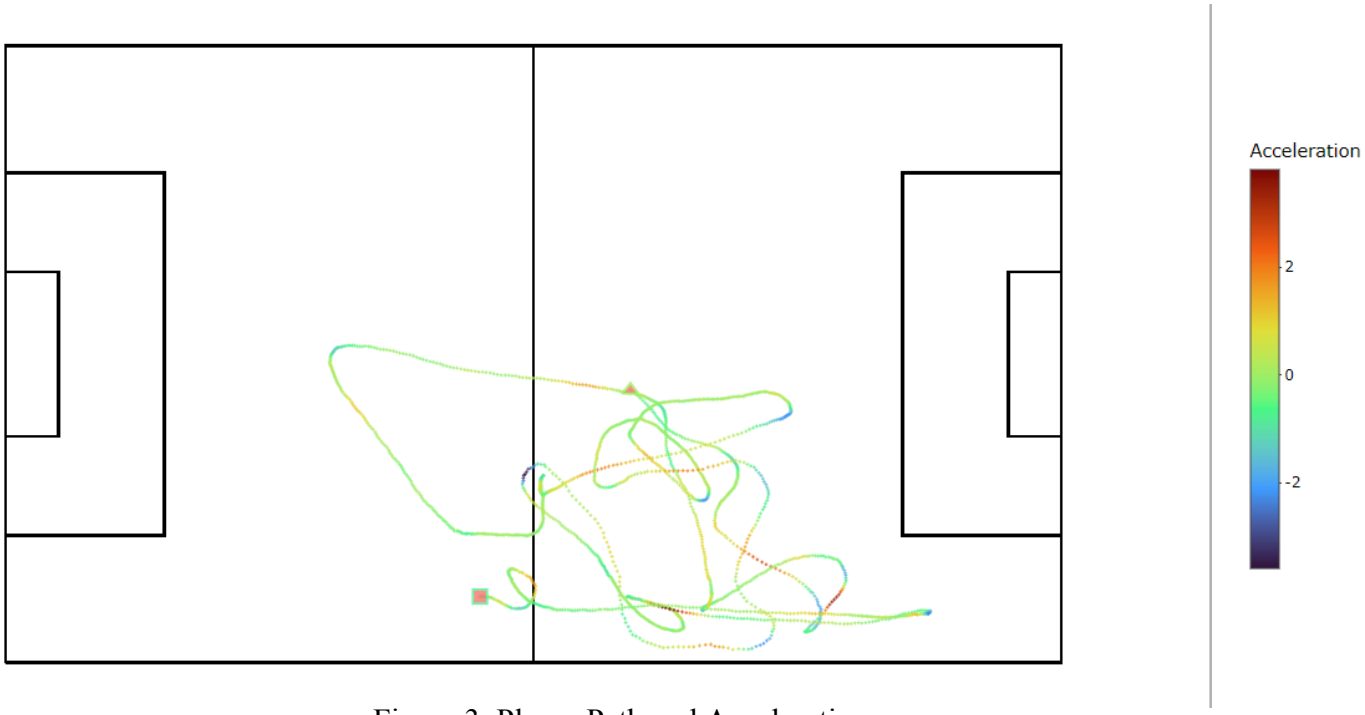


Figure 3. Player Path and Acceleration

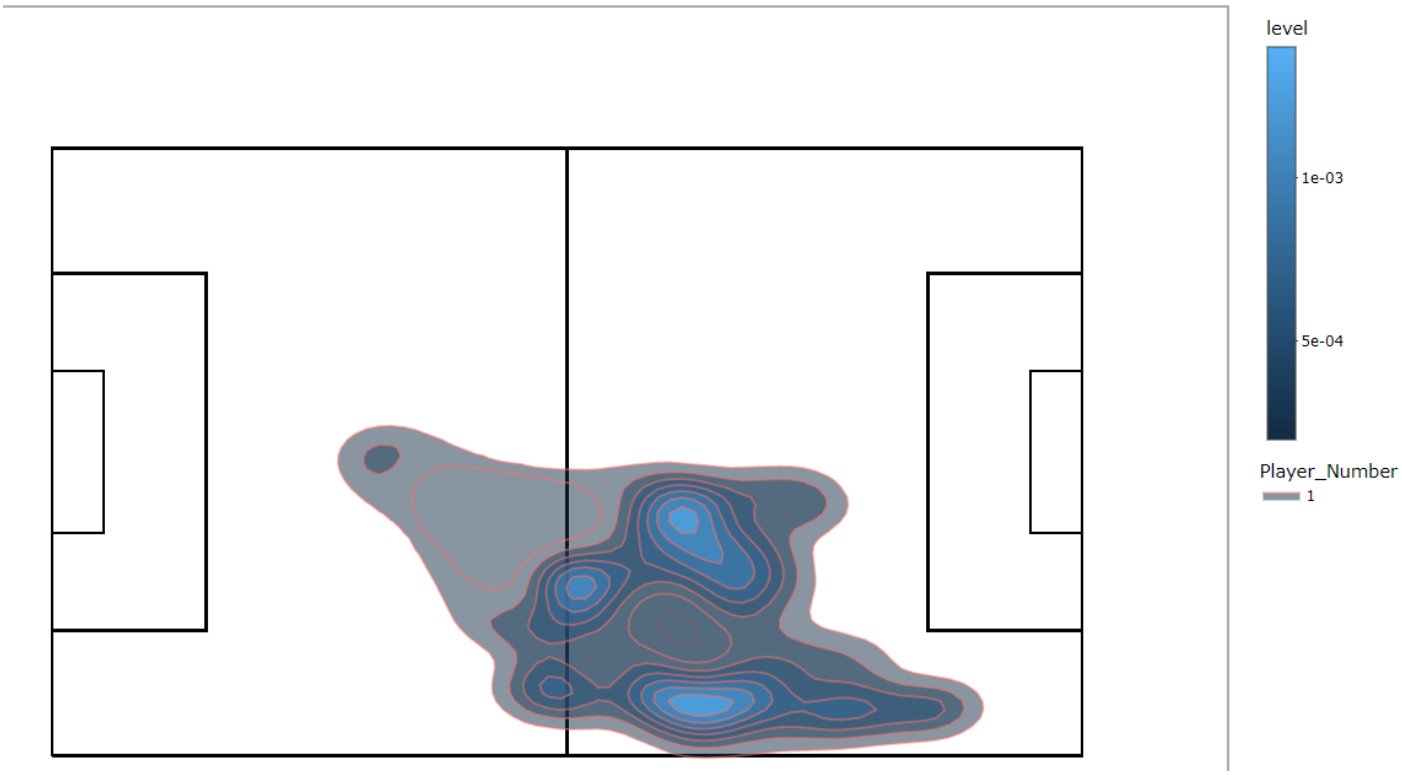


Figure 4. Player Positional Density

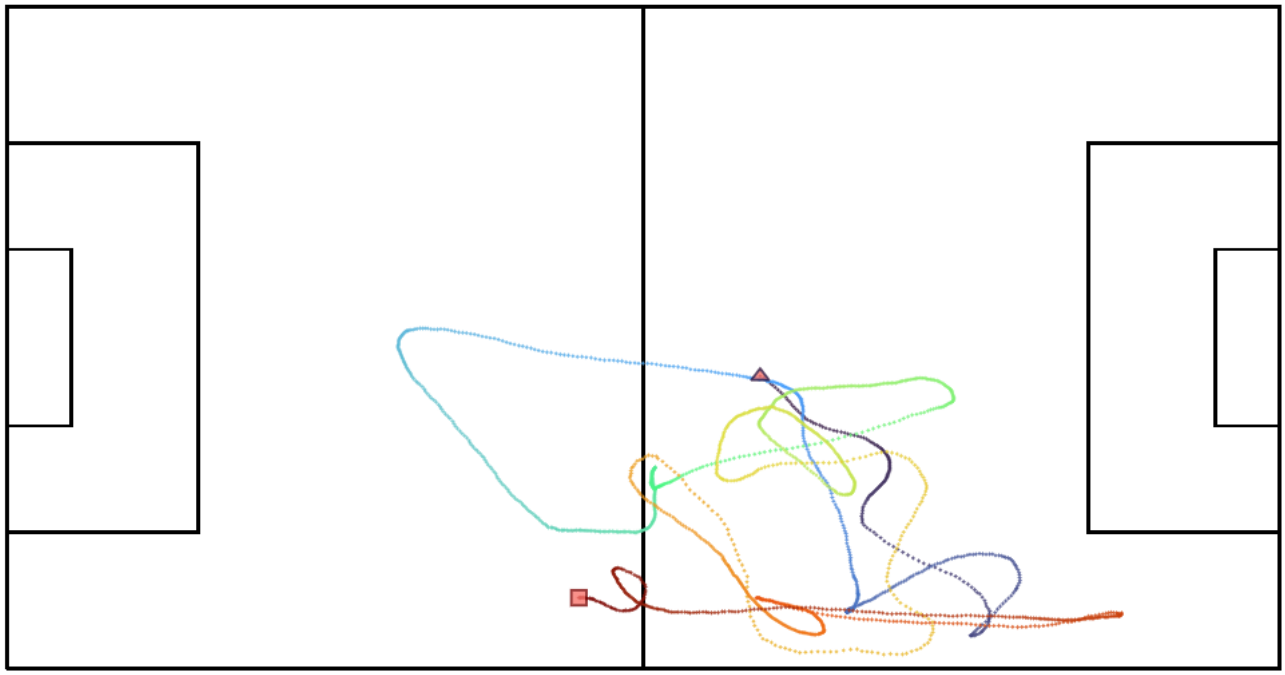


Figure 5. Player Path and Game Time

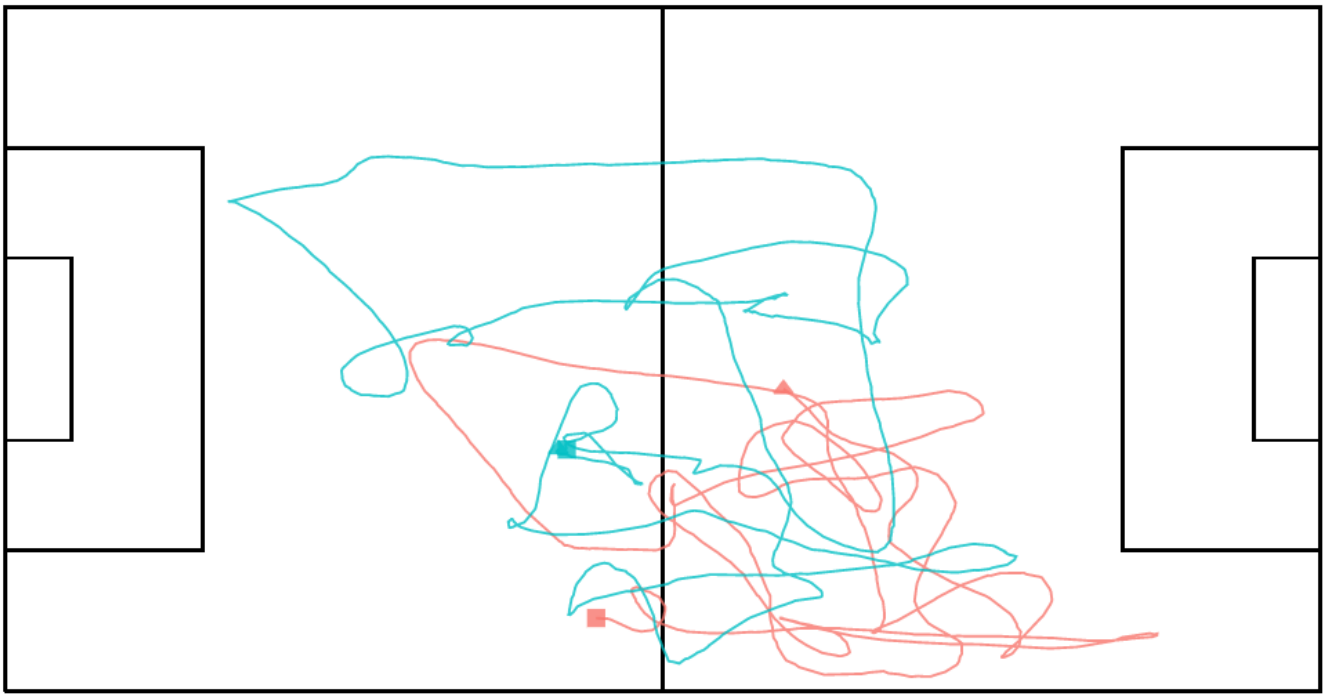


Figure 6. Multiple Player Paths

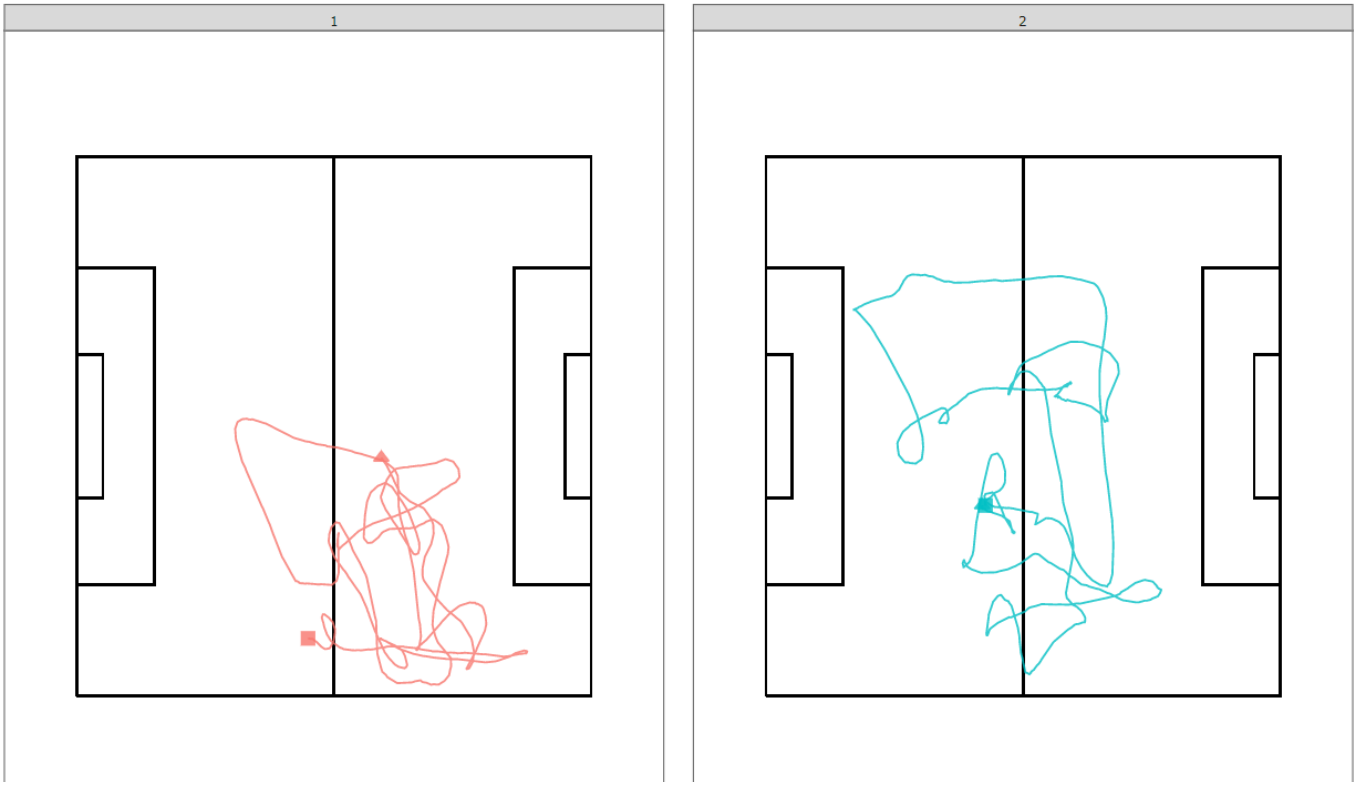


Figure 7. Multiple Player Paths with Faceting

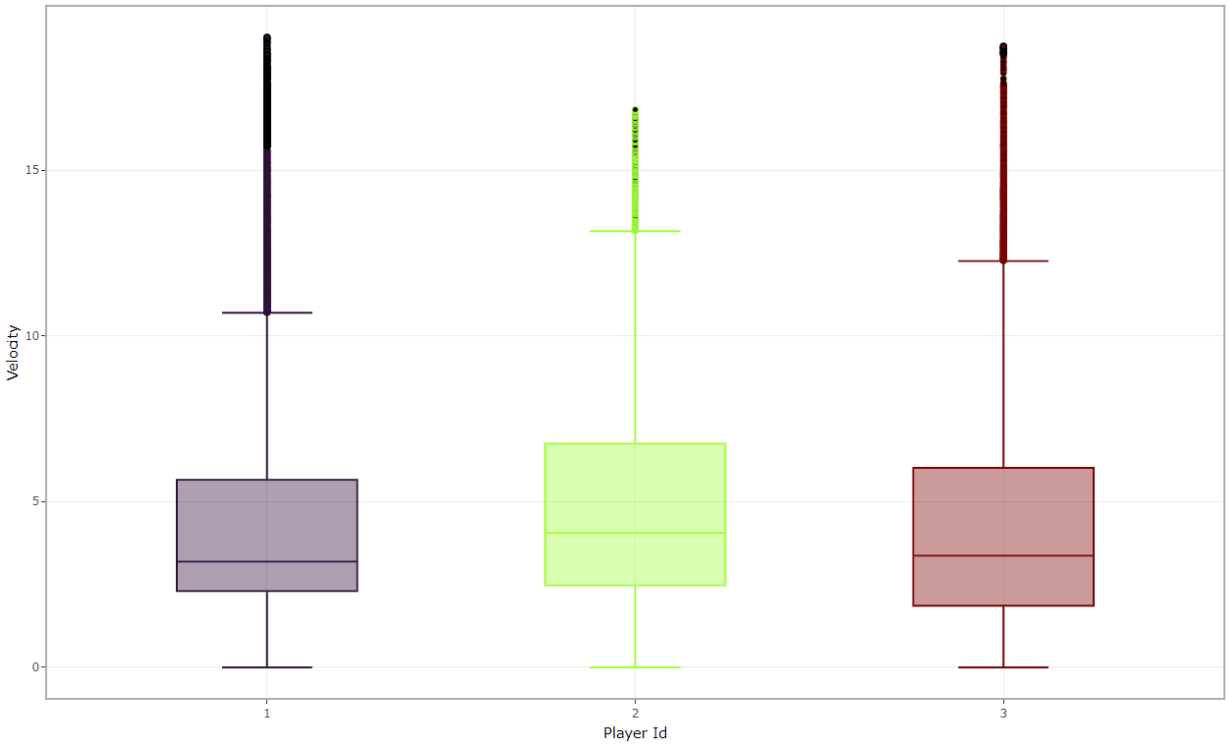


Figure 8. Box Plots of Player Velocity

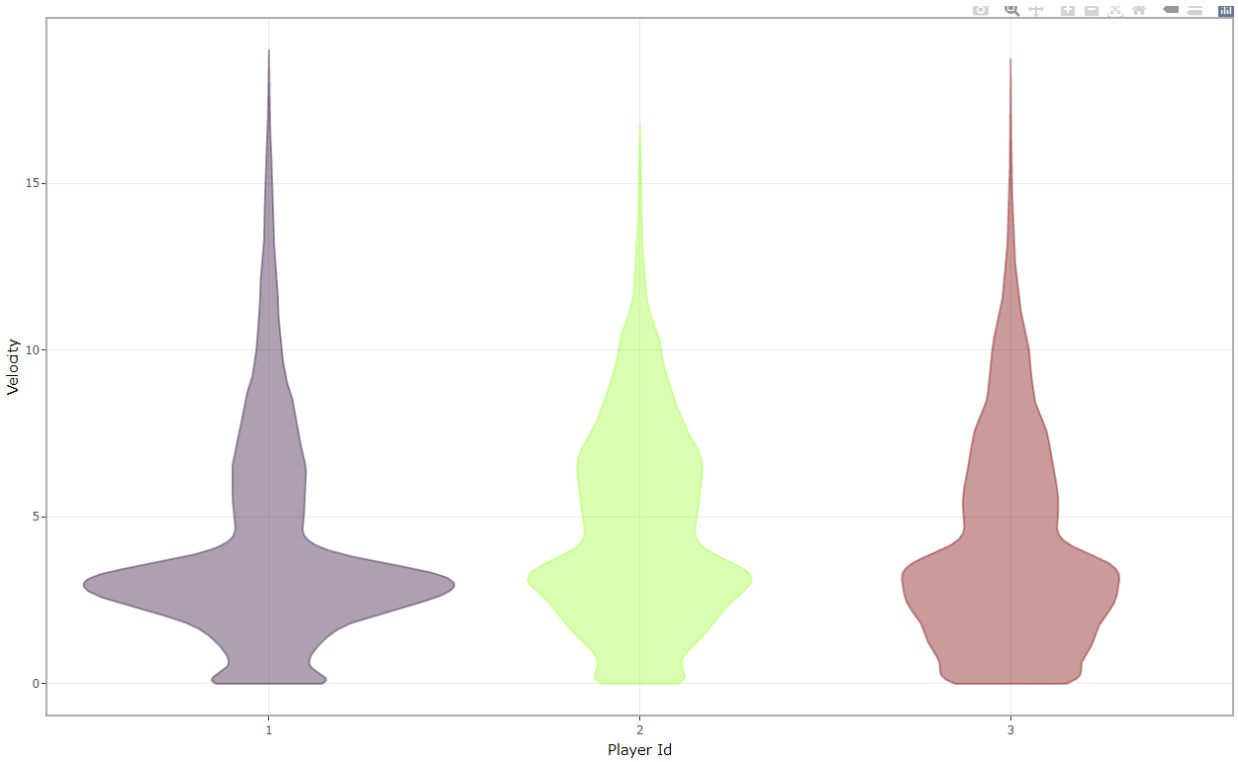


Figure 9. Violin Plot of Player Velocity

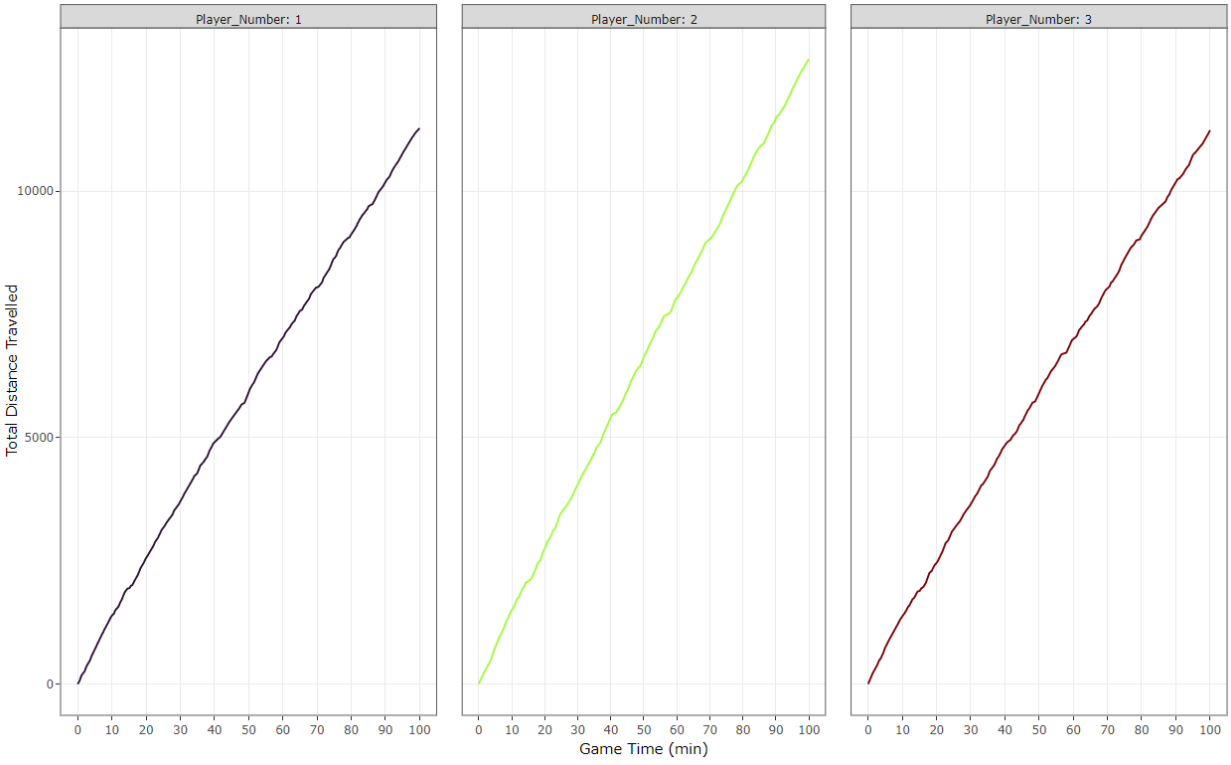


Figure 10. Total Distance Traveled per Player

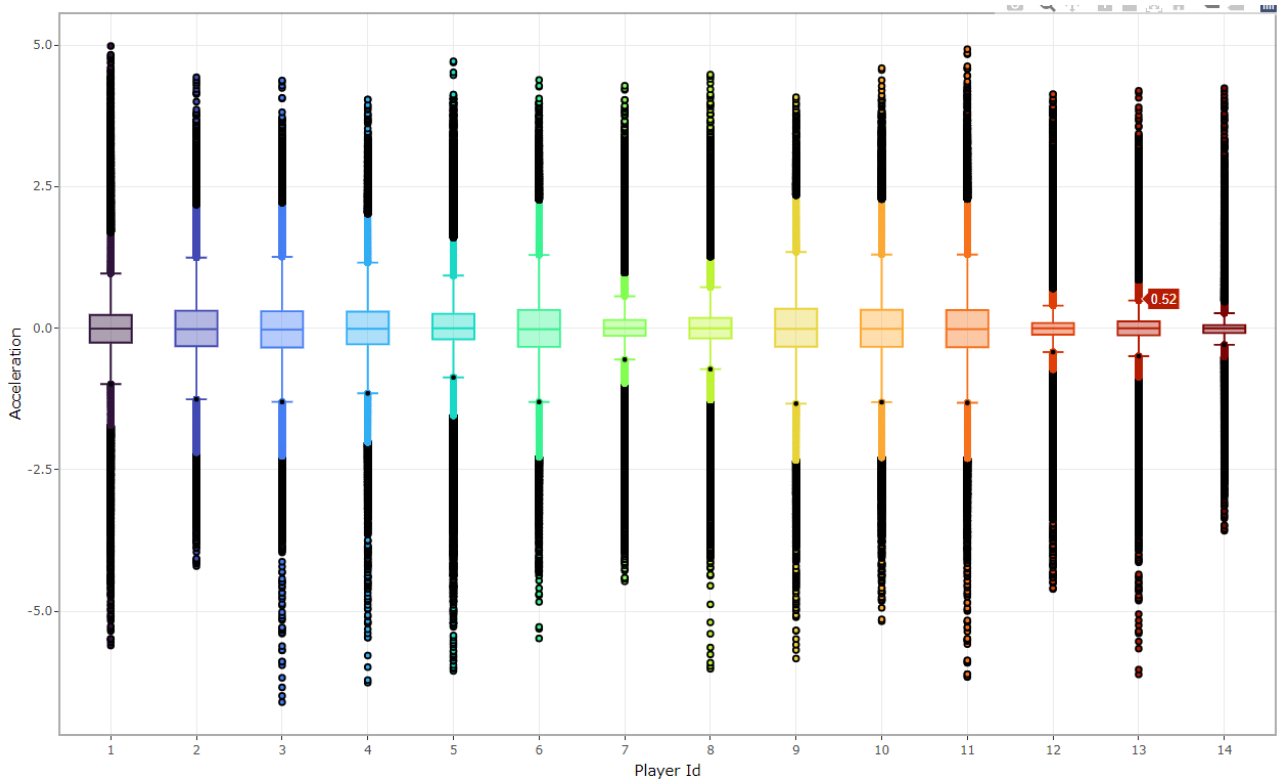


Figure 11. Box Plots of Player Acceleration

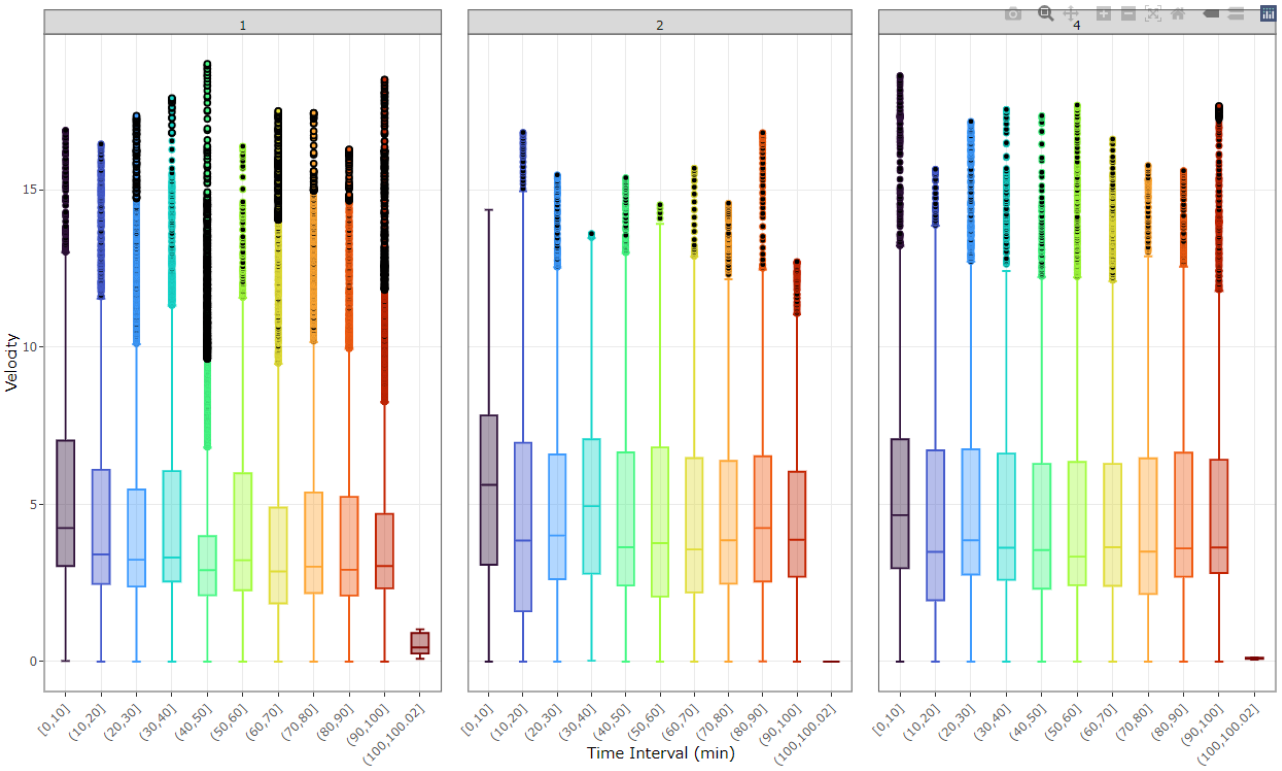


Figure 12. Box Plots of Player Velocity Over Different Time Intervals

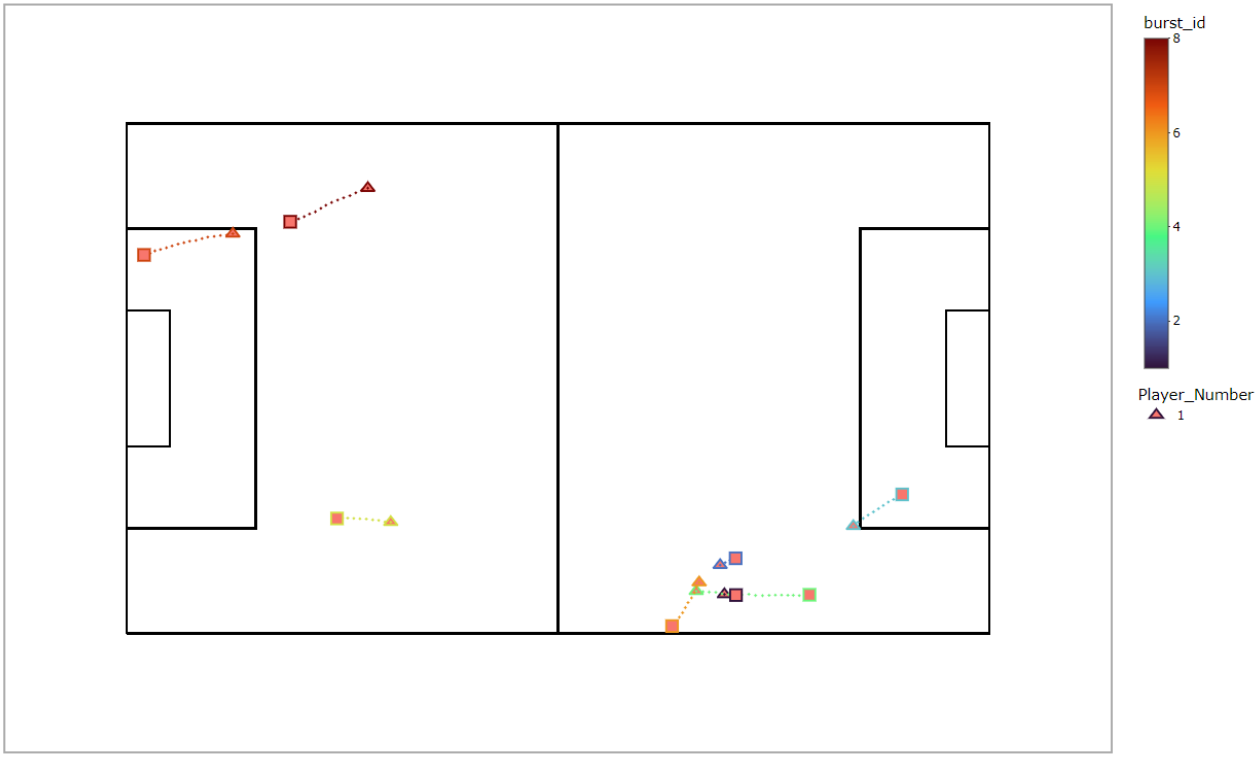


Figure 13. Sprint Burst Paths

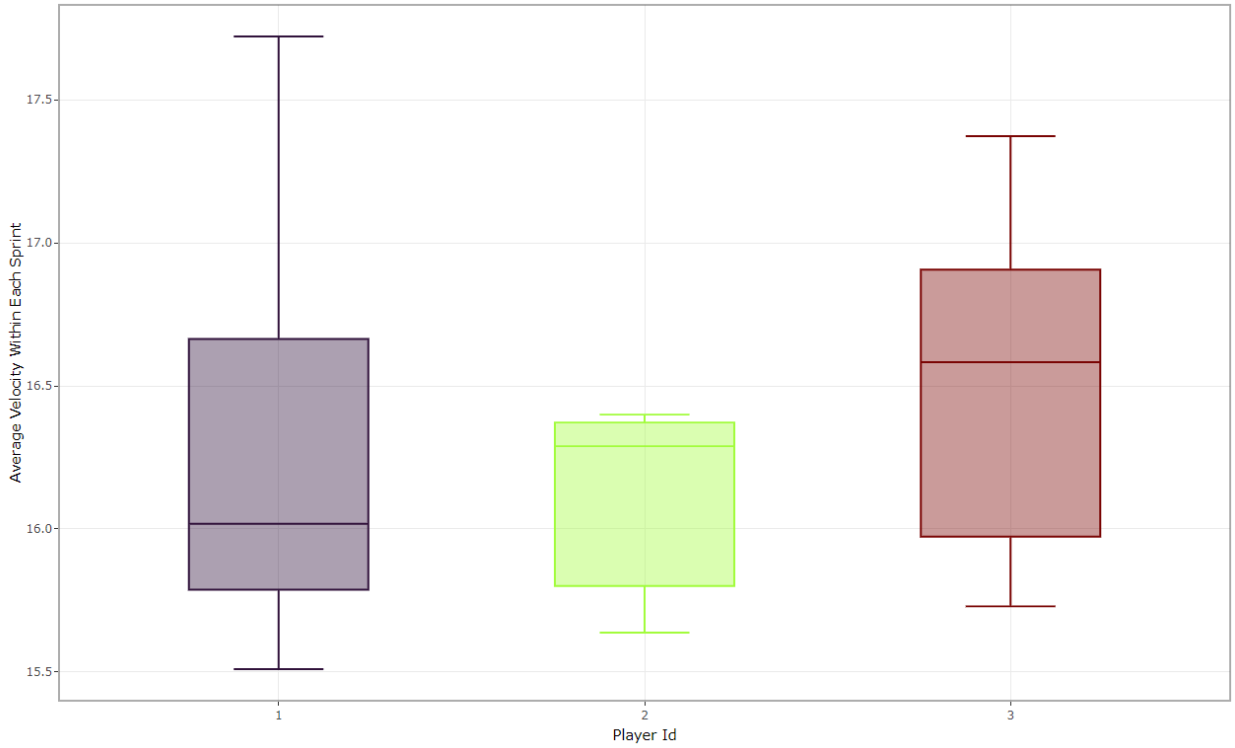


Figure 14. Box Plots of Average Velocity Within Sprint Bursts

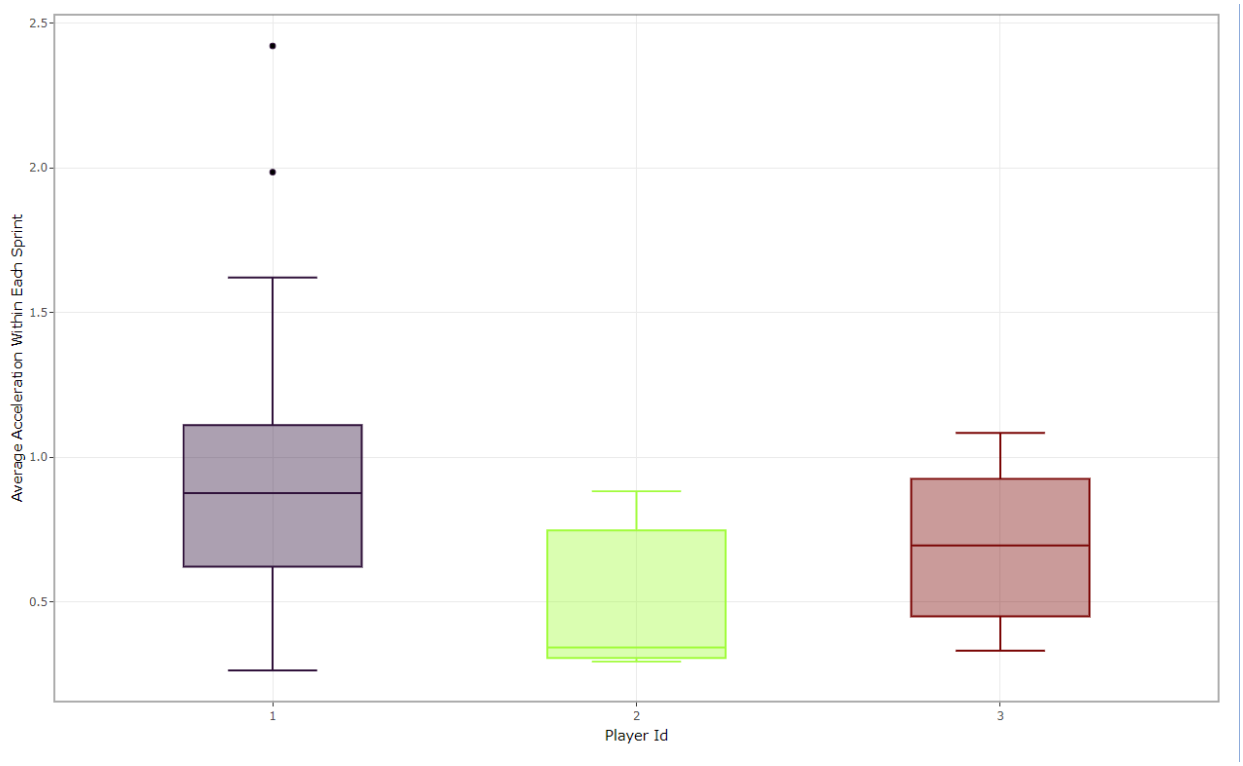


Figure 15. Box Plots of Average Acceleration Within Sprint Bursts

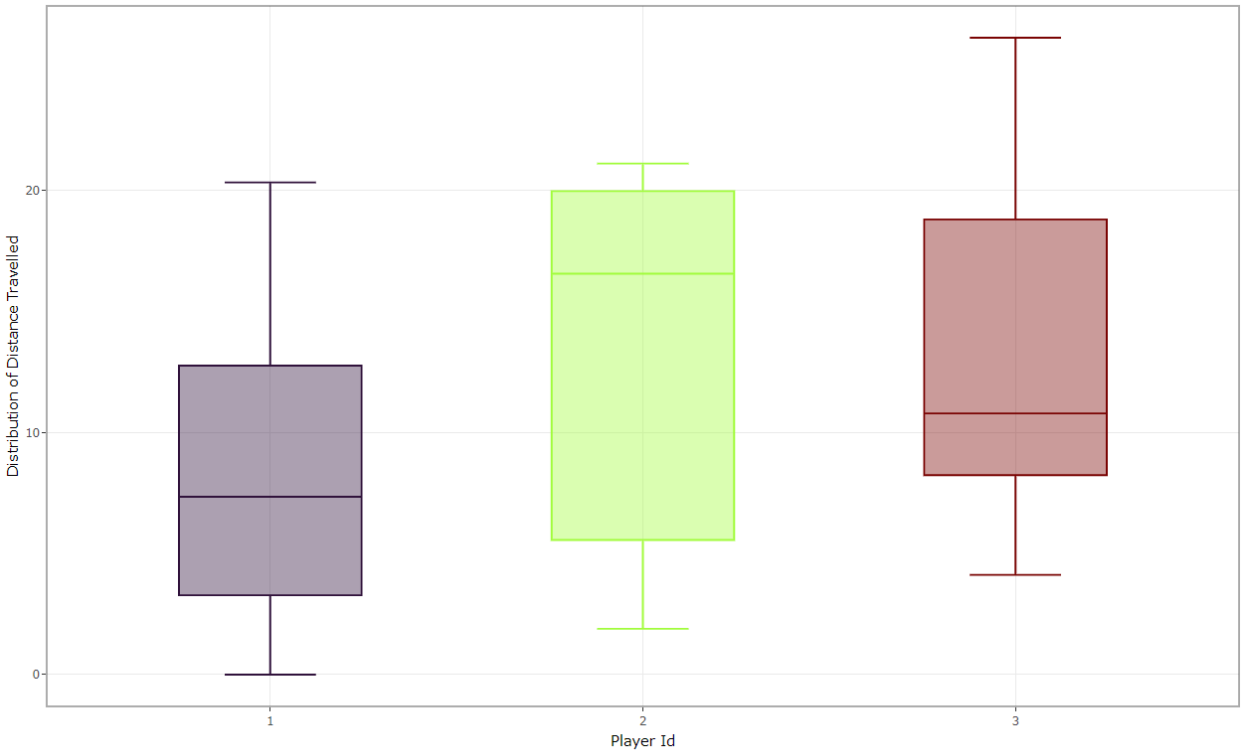


Figure 16. Box Plots of Distance Traveled Within Sprint Bursts

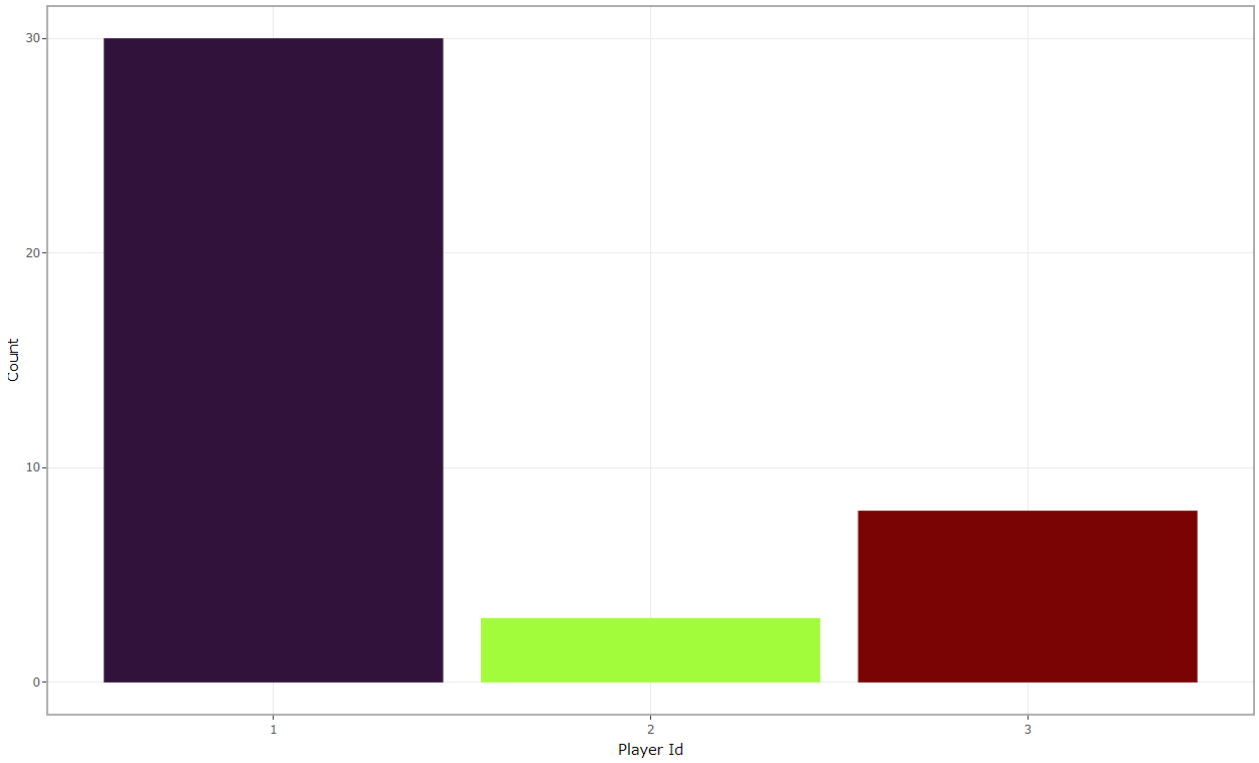


Figure 17. Bar Graph of the Number of Sprint Bursts per Player

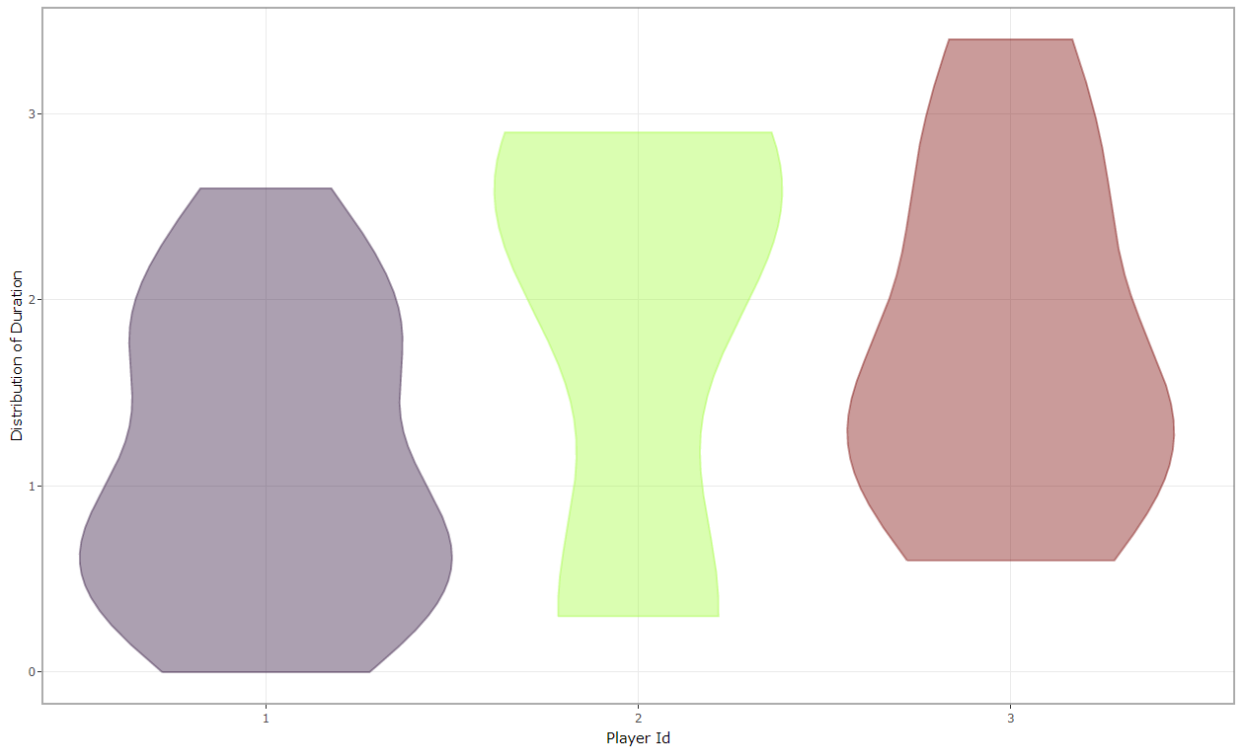


Figure 18. Violin Plot of Sprint Burst Durations

1. Introduction

Soccer matches can involve up to 14 players per match and each player can travel a span of miles every match. Our client was interested in visualizing/analyzing: the position of the players, their velocity and acceleration, and sprint bursts. We created an interactive dashboard that imports game files that have been cleaned and rotated to appear horizontally. Then overlays player's positions on a virtual soccer field with multiple different ways to compare player data among varying time intervals.

With the help of this dashboard, team staff can visualize areas of the game where players are sprinting faster (exerting more energy) and can more accurately quantify player fatigue at the end of every game. The dashboard is completely customizable. Meaning the user can pick any section of the game, and any aspect to analyze. Whether it be velocity, acceleration, distance traveled, or sprint bursts. Each aspect can be analyzed proficiently.

1.1 Problem Statement

Our client was Dr. Yurko. Dr. Yurko runs the Data Science Program through the School of Computing of Information at the University of Pittsburgh. Dr. Yurko teamed up with Dr. Felix Prossel. Who is the Director of Sports Science at the University of Pittsburgh. Together they came up with a problem they wanted us, the Data Science Capstone team to solve. The problem being that there was no way to clean, analyze, and visualize data being gathered by Catapult vests worn by the men's soccer team at the University of Pittsburgh. These vests grab each player's position, velocity, acceleration and more. But with no way to visualize and analyze easily, this data was not of much use. So, this is what we needed to solve.

Before our work, Dr. Prossel would have to manually run many filters on a single player's file to try and find data from a specific play that a coach wanted to see. This took much time, and even with this, there was no way to visualize the data, this just showed the players velocity at the given time the coach wanted. Now, with our applications, each game will be visualized horizontally, creating unity between games since you can see them the same way. Each game does not have to be filtered manually, you can filter with slider buttons and then click a button to download. A game can be analyzed in as little as 5 minutes, depending on if the game went into overtime. This means that there will be a lot more files for the overtime play.

The only problem that our applications face is that there is no way yet to compare a player's data between games. You can only visualize one game at a time.

1.2 Contributions

We accomplished everything that our client asked of us. Our tasks were to import data into an app and perform necessary processing for visualization and analysis. To visualize with color-coding proportional to velocity, acceleration, or distance traveled, analyze runs, or sprint bursts, made by players throughout each game. We created an app that solves the problem of preparing the data to be visualized and analyzed. We created another application that takes the outputted cleaned and prepared data from the first application and is able to visualize and analyze it in many different ways. The rest of this thesis is organized in four sections. In Section 2, our related work is presented. Section 3 shows our approach to address the aforementioned problem stated, our applications, as well as the system design. The results of our work are discussed in Section IV. Finally, Section VI concludes this thesis.

2. Related Work

There are no other systems that are publicly available that exist to visualize and analyze data from catapult vests. Our system is specific to soccer games. There are no systems to visualize data on a soccer field and allow hover functionality. Since there were no other systems in place, we had to create our own unique application to rotate and clean data. Then use this cleaned data in a different application to visualize and analyze the data in a sleek and efficient manner. This system is designed specifically for soccer fields and for data that is gathered from catapult vests. Therefore, this system is perfect for our client and the problem they wanted to solve.

3. System Design

3.1 System Requirements

Importance of Gathering System Requirements

Gathering system requirements is important as, by including the client in discussions of the goals and functionality of the product, the development team can make sure that the image or idea of the product that the client has is aligned with their own. By ensuring that both groups are on the same page regarding the product's development, we can help prevent conflicts, misunderstandings, or disagreements from arising later on in the development lifecycle. Finally, through gathering requirements, the development team is able to gain a better grasp on the client's wishes and goals for the product, so they do not waste time developing features or functionalities that are not important to the client.

List of Initial System Requirements - From Submission at Start of Semester

Minimum Requirements

General Use:

- As a USER, I should be able to navigate between all the sections of the application without complications
- As a USER, I should be able to easily interact with the GUI without issue
- As a USER, I should be able to interact with the GUI and receive feedback within a reasonable period of time (the application is fairly responsive)

Data Import/Selection (first page):

- As a USER, I should be able to select which game(s) I want to analyze
- As a USER, I should be able to select which athlete(s) I want to analyze
- As a USER, I should be able to easily select the game(s) and athlete(s) I want from the established file structure (shown in the GUI) without having to select every file individually

Data Filtering/Visualization (second page):

- As a USER, I should be able to select which variable I want to visualize (e.g. path, distance, velocity, . . .)
- As a USER, I should be able to filter the game data to only display/analyze specific time intervals and/or game states
- As a USER, I should be able to tag specific plays/sequences and/or time intervals for future analysis (e.g. “good pass”, “great defensive sequence”, . . .)
    - As a USER I should be able to view data and statistics about the database of tagged moments/sequences

Desired/Optional System Requirements

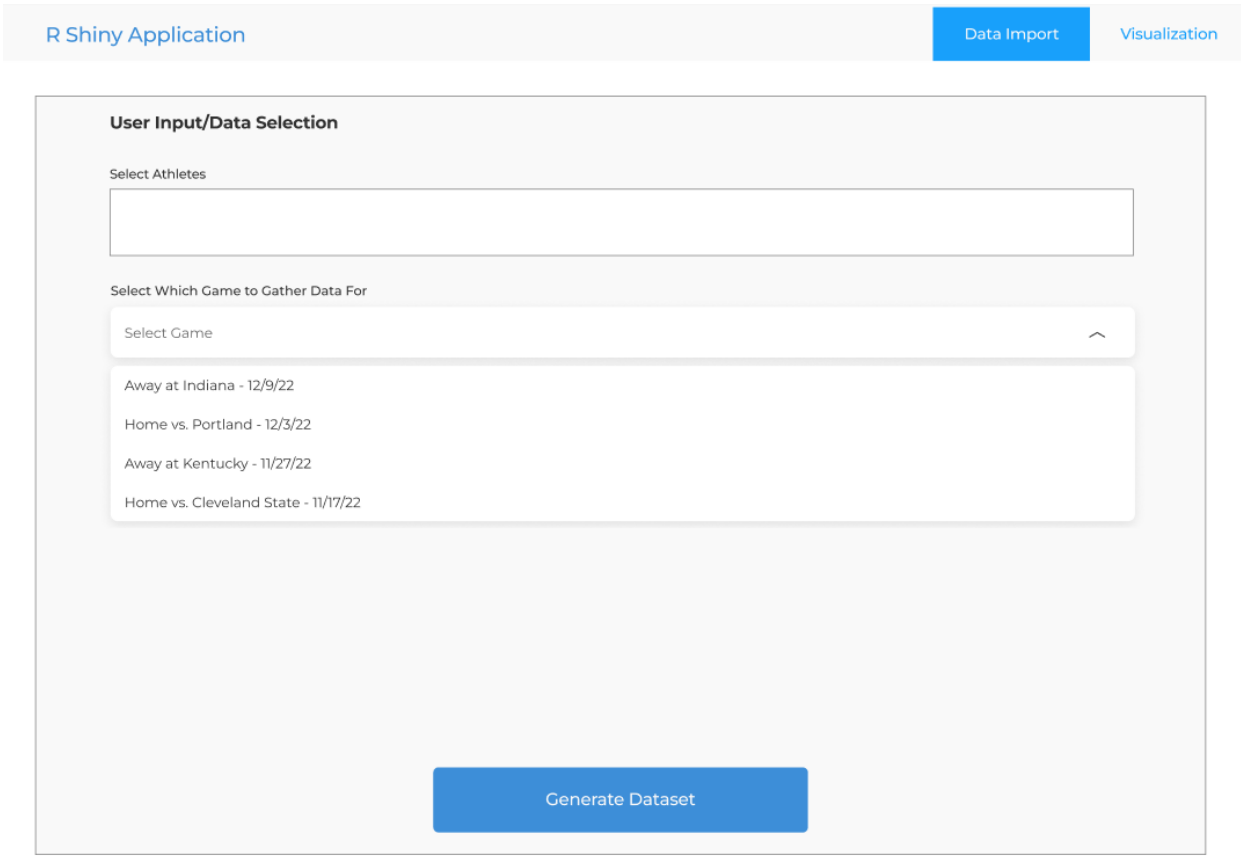
Hopeful Future/Optional Functionality:

- As a USER, I should be able to filter the players paths by a variable (e.g. velocity, acceleration, . . .) to track/identify unique sprints
    - As a USER, I should then be able to analyze these unique sprints to identify values such as the average velocity per sprint, sprints per time interval, . . .
- As a USER, I should be able to add new game data to the database (enables scalability/future use)
    - Dr. Yurko - document the overall process to rotate the data points and create a dataset/dictionary of the corners for all of the main ACC opponents

3.2 Wireframes

Wireframes are essential for providing an initial visualization and idea of what the system or application will look like. This early concept of the system is incredibly beneficial to have, as it helps to clarify the connections between the product's functionality and architecture and the user interface or information that is displayed to the user. Additionally, they allow us to demonstrate the control flow or progression of the application and revise the visual layout of the UI elements to best accomplish a particular purpose or functionality and achieve the highest usability or best user experience.

Initial Wireframes - From Submission at the Start of the Semester



R Shiny Application

Data Import | Visualization

User Input/Visualization Options

Start Point (Filter by Time)

End Point (Filter by Time)

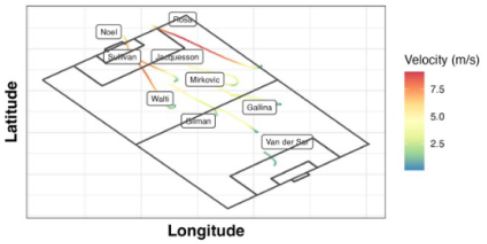
Variable for Visualization

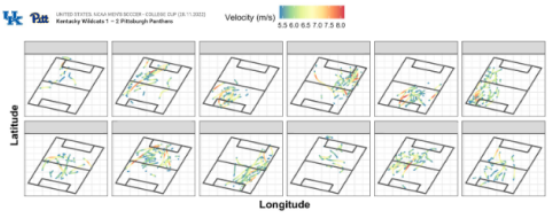
Select Variable

- Distance
- Velocity
- Acceleration
- Heart Rate

Produce Visualization

Visualization Results/Output





3.3 Sample Code

Sample Function 1: Rotating Each Player's Data File to be Horizontal

```
### define a function which rotates the player 2D projected coordinates
### based on the field the game is located at
rotate_player_2d <- function(players_df, a_field)
{

  if(a_field$clock_type == 'counter clockwise' ){
    #rotate counter clockwise
    players_df <- players_df %>%
      mutate(rotate_x = x * cos( a_field$rotate_radian ) - y * sin( a_field$rotate_radian ),
             rotate_y = x * sin( a_field$rotate_radian ) + y * cos( a_field$rotate_radian ))

  }
  else {
    # rotate clockwise
    players_df <- players_df %>%
      mutate(rotate_x = x * cos( a_field$rotate_radian ) + y * sin( a_field$rotate_radian ),
             rotate_y = -x * sin( a_field$rotate_radian ) + y * cos( a_field$rotate_radian ))
  }

  # subtract out the rotated center point
  players_df <- players_df %>%
    mutate(new_x = rotate_x - a_field$rotated_mid_x,
           new_y = rotate_y - a_field$rotated_mid_y) %>%
    mutate(Lat = new_x,
           Long = new_y)
}
```

This function takes in inputs of a dataframe and field's rotation information. The data frame holds a single player's data for a given half of a given game. The rotation information holds the angle we need to rotate the data, the corner of the field we need to rotate around and the center of the field we need to center the data around.

Sample Function 2: Getting Sprint Bursts given a Velocity Threshold

```
grab_sprint_bursts = function(df, velocity_threshold) {
  # filter the dataframe to only include times where the player is running at a speed above the velocity threshold
  sprint_burst_df = df %>%
    filter(Velocity > velocity_threshold) %>%
    group_by(p_id) %>%
    mutate(lag_seconds = lag(Seconds)) %>%
    mutate(diff_seconds = round(Seconds - lag_seconds, 2)) %>%
    mutate(check_seq = ifelse(diff_seconds > 0.1,
                              1,
                              0)) %>%
    mutate(burst_id0 = check_seq) %>%
    mutate(burst_id0 = ifelse(is.na(burst_id0), 1, burst_id0)) %>%
    mutate(burst_id = cumsum(burst_id0)) %>%
    ungroup()

  return(sprint_burst_df)
}
```

This function takes two inputs, a data frame and a velocity threshold. The data frame holds all of the player's data and the velocity threshold is an integer. This function finds all of the times that each player is running greater than this given velocity.

Sample Function 3: Creating a Data Frame That Holds All Player's Data in One Dataframe

```
the_big_file <- bind_rows(file_list)
# editing the seconds depending on which half of the game it is
half_1 = the_big_file %>% filter(Half == 1)
half_2 = the_big_file %>% filter(Half == 2)
half_3 = the_big_file %>% filter(Half == 3)
the_big_file = the_big_file %>%
  select(Seconds, Velocity, Acceleration, Odometer, Lat, Long, Half, p_id) %>%
  mutate(Seconds = ifelse(Half == 2, Seconds + max(half_1$Seconds),
                          ifelse(Half == 3, Seconds + max(half_2$Seconds) + max(half_1$Seconds),
                                 ifelse(Half == 4, Seconds + max(half_3$Seconds) + max(half_2$Seconds) + max(half_1$Seconds), Seconds))))
Seconds = c()
Velocity = c()
Acceleration = c()
Odometer = c()
Lat = c()
Long = c()
Half = c()
p_id = c()

df <- data.frame(Seconds,Velocity,Acceleration,Odometer,Lat,Long,Half,p_id)

# fix the odometer for each player's data
for(player in unique(the_big_file$p_id)) {
  half_1 = the_big_file %>% filter(Half == 1) %>%
    filter(p_id == player)
  half_2 = the_big_file %>% filter(Half == 2) %>%
    filter(p_id == player)
  half_3 = the_big_file %>% filter(Half == 3) %>%
    filter(p_id == player)

  # fix odometer reading so total distance ran is cumulative for whole game
  updated_odometer = the_big_file %>%
    filter(p_id == player) %>%
    mutate(Odometer = case_when(Half == 2 ~ Odometer + max(half_1$Odometer),
                                Half == 3 ~ Odometer + max(half_2$Odometer) + max(half_1$Odometer),
                                Half == 4 ~ Odometer + max(half_3$Odometer) + max(half_2$Odometer) + max(half_1$Odometer),
                                Half == 1 ~ Odometer))
  df = rbind(df, updated_odometer)
}
```

As you can see, the first variable is called the_big_file. This file contains all of the rotated player files that have been row-binded to create a very long file that is in tidy data format. This new file

contains a column to identify which row corresponds to each player and another column to identify which half. This is so we can easily graph data from a specified game in the visualization app with one file as opposed to multiple files for each player. After we create this big file, we make some changes to the data. We edit the seconds so that the seconds are continuous throughout the game instead of restarting back to zero at the start of a new half, since each half is its own file. We also edit the odometer column so that a player's distance traveled is cumulative throughout the whole game instead of restarting at zero like the seconds.

3.4 Installation Instructions

Prior to downloading all of the files needed to run the program, users must first download and install R and RStudio on their computer. All of the application files can be downloaded from the github repository by pulling from the “Pre-Disaster” branch, after the files are stored locally on your computer no further work needs to be done moving files and the user is ready to open RStudio.

When importing new data into the system, open the R Shiny Data App. This can be done by opening the ui.R, server.R, or global.R files located in the “R Shiny Data App” folder in RStudio and running them. Prior to running the files, a few packages may need to be installed in RStudio (this will only need to be done once) and these packages can be found at the top of the global.R file. Once the application is open, a welcome page will be displayed with a more detailed step-by-step instruction manual on how to use the data application. The process for opening the main application is the same as the data application. Open up one of ui.R, server.R,

or global.R located in the “RShiny App” folder in RStudio. Then ensure that all needed libraries are downloaded and press run to open the application.

4. Results

The system solves all of the problems that the team was tasked with which included: rotating player coordinates to be visualized on a horizontal field, creating an interactive application that allows users to customize plots for analysis, creating a system for future game data to be easily integrated into the system, calculating sprint bursts based on velocity and duration. The result is two applications which divide the data management and data visualization so that each application can function independently of one another for maximum efficiency. The data management application handles the importing of new game files and the storage of the files in the database (managed by a relational database schema across numerous CSV files) and is updated upon every new game entry. The calculation of sprint bursts is only executed on demand, in order to allow the user to specify the minimum threshold for velocity and distance of each sprint burst.

5. Conclusions

Our conclusions are that our system can rotate and clean data from catapult vests worn during soccer practice or a game. We concluded that in order to rotate a game to be horizontal, we needed to account for the curvature of the Earth by projecting the data onto a two-dimensional surface as well as rotating the points by the angle of the field. We also concluded that when we cleaned the data we had to ignore data that was taken from locations

that are off the field. Once we cleaned and rotated the data, we could visualize and analyze data proficiently from a player or a team during a game or practice.

6. Future Work

Future work for our system would be to create another application to compare data between games. Currently our application can only visualize and analyze data from one single game at a time. If there was another application to visualize data from multiple games, this could be used to track a players and/or teams performance throughout a season and/or over the years.