INF522 - Report
Interactive F1 Telemetry Dashboard

by

Roxane Chen

GITHUB LINK

Décembre 2023

Sommaire

1 Introduction

This project aims to implement an interactive telemetry dashboard for F1 races. The goal is to provide a comprehensive tool for users to gain insights into the intricate details of past F1 races' telemetry data. This report will delve into the technical aspects and implementation strategies used to build this dashboard.

Annually, Formula 1 commences a new season composed of around twenty races, occurring in different cities. Each race features a grid of 10 teams, each with two drivers. Points are awarded to the top 10 finishers, contributing to both the driver and constructor championships.

As a casual watcher of F1 for several years, I have always been really curious about how engineers on the grid analyze the performances of the car and how different parameters influence the car's performance. So I saw this project as an opportunity to finally take the time to dive deeper into this subject.

2 Dataset

I was first going to do my project on the Kaggle Formula 1 World Championship (1950 - 2023) dataset but I realized that the dataset wasn't detailed enough for what I wanted to do. I then found the **FastF1 python API**. This API gives access to F1 lap timing, car telemetry, tire data, weather data, the event schedule, and session results. To be able to use that information in my web application without any backend work, I compiled manually the data into CSV files. This is how I chose to organize the files :

For each season : **race_schedule.csv**

And for each race of the season I have a :

— **session_laps_info.csv** : that gives general information on each laps of each driver on the grid such as the lap time, track status, compound, tyre life etc.

— **session_result.csv**

— **circuit_trace.csv** : that gives the X,Y coordinates that are then used to draw the map

— a folder **telemetry** : with for each driver a csv file with all the telemetry data (throttle, brake, speed, ngear, drs, rpm) during the whole race

A jupyter notebook detailing the data-compiling process was added to the project submission.

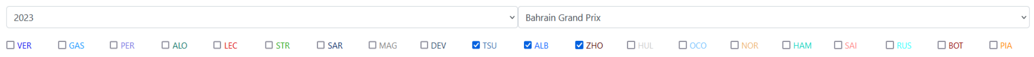
Warning : For the submission to not be 8Go, I only added to the file the data for 2 races (the 2023 Bahrain race and the 2023 Saudi Arabian race) which should be enough to get an overview of this dashboard's capacities.

3 Visualisation

The dashboard was implemented using mainly the d3.js library. Using this library made it easier to build interactive and customizable visualization.

3.1 General overview of the dashboard

The user needs to first choose a season and then a race. The corresponding files are then loaded and processed which takes some time.
After choosing the race, a list of all the drivers who participated in the race will appear and the user can then choose which driver they want to compare. Each driver's color is associated with his teams color (red for Ferrari, orange for McLaren etc) The user can for example choose to compare the two drivers of a constructor as they race in the same car the telemetry data are essential to understand the difference in performance between the two.



☐ VER ☐ GAS ☐ PER ☐ ALO ☐ LEC ☐ STR ☐ SAR ☐ MAG ☐ DEV ☑ TSU ☑ ALB ☑ ZHO ☐ HUL ☐ OCO ☐ NOR ☐ HAM ☐ SAI ☐ RUS ☐ BOT ☐ PIA

FIGURE 1 – Race and driver selection

The dashboard can then be divided into two parts :

- General Information on the race selected
- Precise telemetry data lap per lap

3.2 Race Overview Section

This first section gives general information on the race selected with a lap time chart and a result table.

3.2.1 Lap Time Chart

On the right, there is a **lap time chart**. This chart allows the user to have a general overview of the race at a glance. Indeed, the user can see the laps that are unusually slower which often means that the driver made a pit stop during this lap or that there was a safety car. It also shows if the driver didn't finish a race, at which lap he stopped.

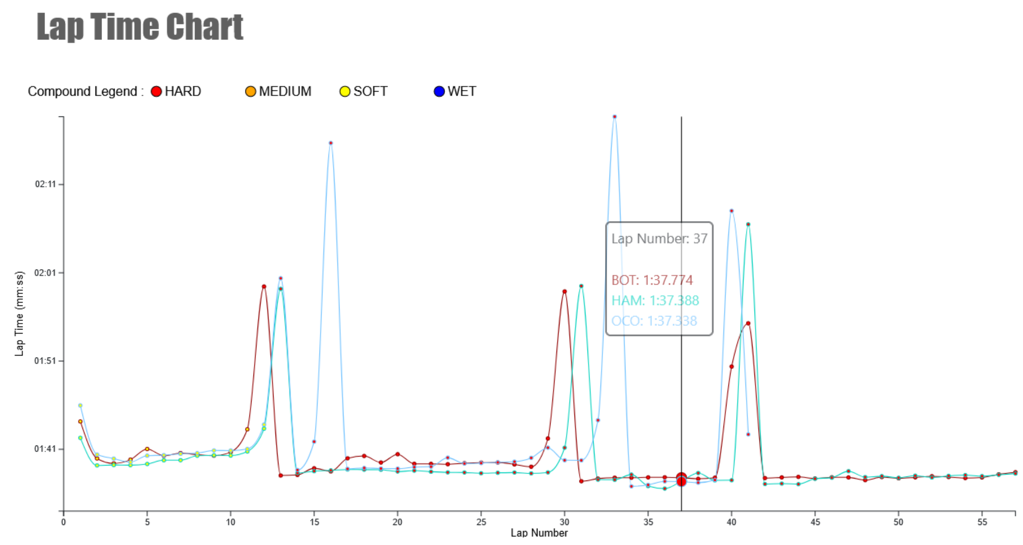


FIGURE 2 – Lap Time Chart

Design Choice & Implementation

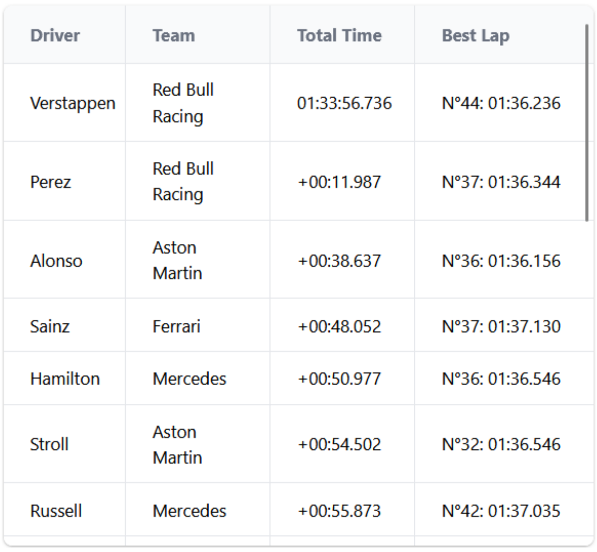
If the user selects too many drivers the chart becomes quickly really difficult to read as the lap time tends to be quite close, so I added a tooltip, which makes a line and a text legend appear and move with the mouse. This way, the user can easily navigate and compare the lap times. To increase the visibility I also enlarged the data points selected.
Another information that can be useful is the tyre compound used by the driver during the lap, to make this information appear, I changed the fill of the data point circle according to the compound (red for hard, orange for medium, yellow for soft and blue for wet). This way the user can easily see the tyre strategy for each driver.

3.2.2 General Ranking

On the left of the lap time chart, the user can find all the information regarding general statistics. There is a simple table that summarizes the total race time for each driver, their fastest lap time, and the corresponding lap number. In Formula 1, the crucial information is not the overall race time but rather the time difference between each driver. To convey this, I present, as it is usually the case, the complete race time solely for the race winner, while for other drivers, I show the time gap to the leader if the gap is less than a full lap. If the driver didn't finish the race I display, instead of the time, the reason why they did not finish. I had a hard time deciding what data to display because so many data were available. For example, I didn't deem it necessary to add any information on the number of points won and the driver or constructor standing after each race because it didn't bring anything to understand the race better.

Design Choice & Implementation

For the implementation, I chose to use the grid.js library as I was planning to only do a simple grid display without any interaction. It was easier to use this library than to build a grid from scratch.



Driver	Team	Total Time	Best Lap
Verstappen	Red Bull Racing	01:33:56.736	N°44: 01:36.236
Perez	Red Bull Racing	+00:11.987	N°37: 01:36.344
Alonso	Aston Martin	+00:38.637	N°36: 01:36.156
Sainz	Ferrari	+00:48.052	N°37: 01:37.130
Hamilton	Mercedes	+00:50.977	N°36: 01:36.546
Stroll	Aston Martin	+00:54.502	N°32: 01:36.546
Russell	Mercedes	+00:55.873	N°42: 01:37.035

Figure 3 – Driver Ranking and Time

3.3 Telemetry Analysis Section

For each race, each of the 20 drivers made around 50 laps, and the race lasted for around two hours. So it was essential to think about the best way to present the telemetry data. For example, it was not possible to present the telemetry data for the whole race all at once in a chart. It would have been impossible to understand. Instead, I decided to do the telemetry analysis lap-per-lap (ex. Compare the first lap of Gasly and Ocon).
Using the lap time chart introduced in the previous section, the user can click and select a lap to analyze.

This section is also made up of two sub-sections : a circuit display and a telemetry charts stack.

3.3.1 Circuit display

To enhance the user's comprehension of the telemetry data, I decided to develop a circuit display. This visual representation allows users to observe telemetry data in relation to the car's position on the circuit. The circuit display serves as a valuable tool for illustrating how telemetry parameters influence the behavior of the car throughout different sections of the track.

Design choice and implementation
The Fast F1 API doesn't actually offer the trace of each circuit but this issue is easily countered by using the position data of a driver during a single lap run. To create the circuit display, I use a SVG element and the D3.js library. Given the X, and Y positions of a driver during a lap, I established scales for both the vertical and horizontal axes, mapping the data points onto the corresponding dimensions of the SVG canvas. The resulting circuit is drawn as a continuous line connecting these data points, forming a visual representation of the car's trajectory during a race. The generated SVG element with an accompanying path serves as a foundational component for further integrating and visualizing driver position at different moments of the race. The drivers are simply represented using a color circle.

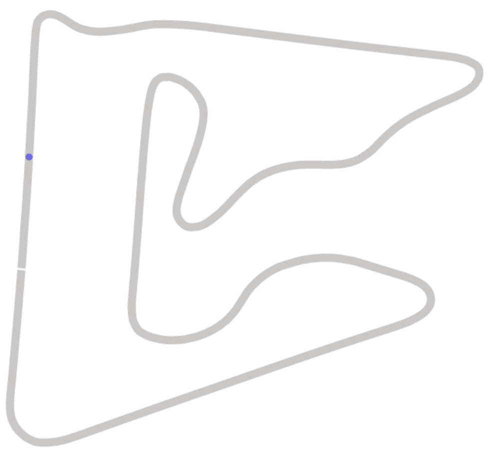


FIGURE 4 – 2023 Bahrain Race Circuit Display

3.3.2 Telemetry Charts

For this project I used the telemetry data made available by the FastF1 API which are :

— **Speed** in km/h

— **Throttle percentage** : it captures the position of the throttle pedal. This data is especially crucial when taking a corner. Precise throttle control can make a split-second differences that are crucial in racing.

— **Brake Pressure**

— **DRS** (Drag Reduction System) : a driver-controlled device used to aid overtaking and improve wheel-to-wheel racing.

— **RPM** (Revolutions Per Minute) : measure of how fast the engine is spinning ie how much power it makes.

— **Gear Number**

Design Choice and Implementation

I had the choice to either plot the telemetry against the distance covered since the beginning of the race or the time since the beginning of the lap. I decided to go with the latter because I added a real-time playback option and also because it gives better insight into how the telemetry parameters strategy impacts the lap time. The first option makes more sense to compare how each driver takes the corner (for example when do they start braking ?). As for the lap time chart, because of the number of charts, the charts are difficult to read so I added a tooltip to help the user read the six charts at once. Also, if the user selects more than 4 drivers to compare, the text legends of the different charts start to overlap which makes it hard to read : so I added the option to remove the legend with a checkbox.

The charts were implemented using D3.js, I chose to use this library instead of chart.js for example because it was easier to add interactive features such as the one I will detail in the next section.

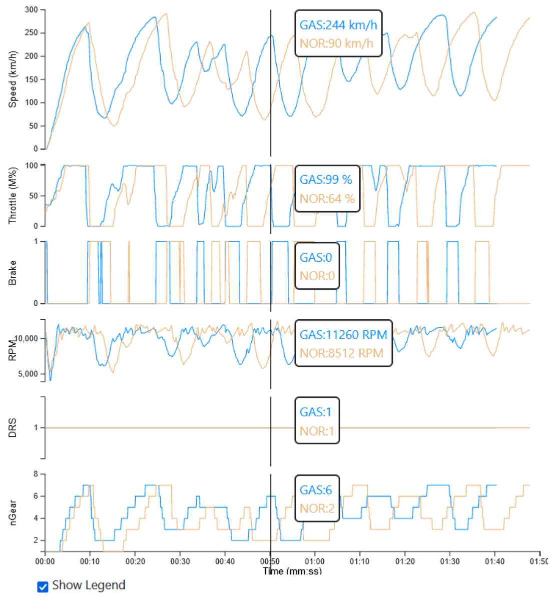


FIGURE 5 – Telemetry Charts

3.3.3 Interaction between the two

These two elements were implemented to interact, there are two ways they can interact :

— **Manually** : when the user hovers and moves their mouse over the charts, on the circuit the circles representing the drivers selected also move. This way the user can understand where the car is on the circuit rather than just looking at charts and data that are difficult to understand with just the time as a reference. Through the circuit display, users gain insights into the driver's actions during a race. For instance, it becomes evident when the driver starts braking before approaching a corner. The display also facilitates the observation that local speed minima align with corners, showcasing a clear correlation. Notably, sharper corners necessitate lower speeds, as well as earlier braking by the driver. These straightforward observations are made more accessible and comprehensible through the visual representation provided by the circuit display.

— **Pseudo real-time replay** : another option I added, was a pseudo-real-time playback. When the user clicks on the "Start Replay" button located at the bottom of the circuit display, both the circle on the map and the charts' tooltip lines dynamically move. This functionality enables users to visualize the car's speed, observe deceleration during cornering, and also simply offer an alternative way to replay a lap. This interactive playback feature encourages users to take the time to look into the telemetry data throughout a whole lap.
Implementation I was able to implement this feature because the API has the option to generate the telemetry data at a fixed given sampling rate (10Hz). I then use an Interval to send the telemetry data every 10ms to move the drivers' position on the circuit and the line on the charts.

4 Conclusion and Possible Future Work

This project successfully achieved its objective of being an interactive dashboard, enhancing users' comprehension of a specific race and telemetry data. However, there is still a lot of room for improvement. A notable improvement would be to be able to use the FastF1 Python API with the web interface. It would involve some backend development that I am not familiar with right now and I didn't have enough time to get into it as it is not the goal of this course.

Another feature I could add in the future would be introducing the capability to compare any lap of any driver together (to be able to compare two different laps of the same driver for example). This feature would empower users to conduct detailed comparisons and analyses across various laps. This could further refine the tool's clarity and versatility.