DrifTracker: an iOS app to improve your drift performance

Erika Scaltrito and Sabrina Vinco
Sport Tech Course, University of Trento

December 24, 2024

Contents

1 Introduction

Drifting is a motorsport discipline that challenges drivers to maintain control during high-speed slides, emphasizing precision, coordination, and skill. As the sport has gained popularity in recent years, quantitative performance measurement tools are exclusively available for high-level competitive events, leaving drivers unable to measure their progress during training sessions. This gap has inspired the development of DrifTracker, a solution designed to combine sensor data analysis with stress monitoring.

1.1 Objectives

The primary goal of this project is to develop a native iOS application that helps drivers to analyze their performance during training sessions, focusing on key metrics like drift angle and speed. The app also integrates biometric data, specifically Heart Rate Variability (HRV), to understand how stress affects performance [7]. By offering a user-friendly interface, the app aims also to provide drivers with useful insights to improve their skills.

1.2 Competitor Analysis

To better understand DrifTracker's position in the market, a comparison was made with the only competitor in Italy, Hydra Data Analysis [4]. Hydra is a tool designed specifically for high-level competitions, used in the Drift Italian Championship, and provides features tailored to judges and organizers. However, its use is limited to competition environments, making it inaccessible to individual drivers for personal training purposes.
In contrast, DrifTracker focuses on accessibility and independence. Table 1 highlights the key differences between the two tools, showing how our solution fills the gap left by competition-oriented systems like Hydra.

Feature	**Hydra Data Analysis**	**DrifTracker**
Target Audience	Competition-oriented	Designed for drivers
Accessibility	Limited	High
Specialization	High (for judges)	Versatile (for training)
Convenience	Low	High
Independence	No	Yes

Table 1: Comparison Between Hydra Data Analysis and DrifTracker

2 Project Planning

2.1 Activities and Timeline

The project was structured into specific activities, each corresponding to a phase of development and carried out following a well-defined timeline:

1. **Requirement Definition (Week 1):** This phase involved defining the functionalities, creating mockups, and selecting the technological stack necessary for the project.
2. **Learning Swift and Sensor Analysis (Week 2–5):** In these two parallel phases, were studied the Apple Developer Documentation [5] and libraries to gain a deep understanding of Swift programming, while simultaneously were analyzed the characteristics of various IMU sensors.
3. **Application Development (Week 5–9):** This phase involved building the user interface and integrating the app with sensors for data collection. Initially, was

explored the connectivity with watchOS to retrieve HRV data but later a strategic change was made: use directly the iPhone's Health app, which ensured broader compatibility with different smartwatches.

4. **Testing and Data Collection (Week 10-11):** The app was tested during real-world training sessions to validate its functionality, while driver data was collected for analysis.
5. **Data Analysis (Week 11-12):** Insights were extracted from the collected data, focusing on the impact of stress on performance and the accuracy of the calculated metrics.

2.2 Milestones

The project was structured around three key milestones:

- **M1: October 6th**
  Completion of the requirement definition phase, including identifying user needs, defining functionalities, and finalizing the technological stack.
- **M2: October 27th**
  Completion of learning Swift and sensor analysis phase to start the application development.
- **M3: December 1st (Adjusted to December 6th)**
  Initially planned for the 1st of December, this milestone marked the completion of the app's main features. Due to an underestimation of development time, the milestone was achieved on the 6th of December.

3 Sensor Analysis

A critical aspect of DrifTracker is the accurate calculation of drift angles, which relies on gyroscope data. However, raw sensor data is often affected by noise and inaccuracies, especially during the dynamic conditions of drifting. To evaluate the reliability of the iPhone sensor, a comparison was performed against the IMU Nicla Sense ME sensor [2] and the GoPro sensor [3].

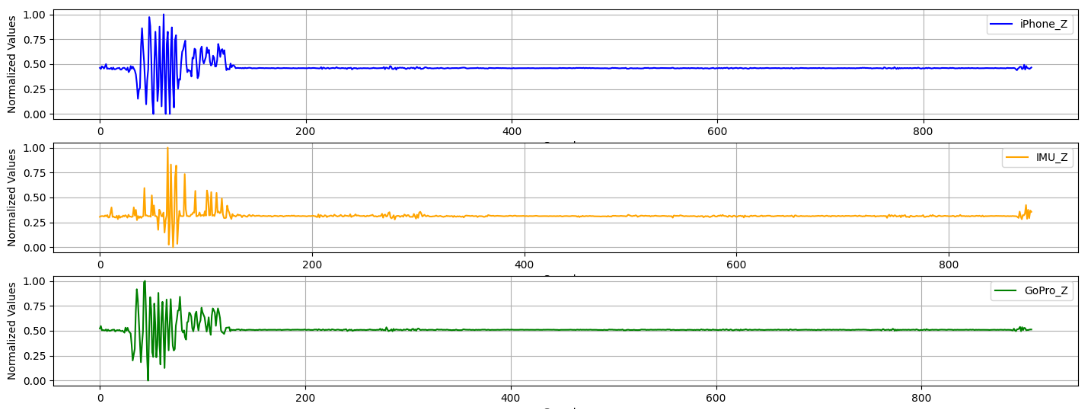


Figure 1: Comparison of Z-Axis Readings

The results showed that the iPhone sensor produces readings comparable in accuracy to the external sensors, validating its use for this application. This decision simplifies the app's design by avoiding the need for additional hardware.

To effectively address issues related to sensor noise and drift error [1], the app uses the Kalman Filter.

3.0.1 Kalman Filter

The Kalman Filter operates by combining measurements from multiple sensors to produce a more accurate estimate of the desired state, in this case the drift angle. It uses a recursive process that updates estimates as new data becomes available. It works iteratevly as follows:

- **Prediction Step:** Based on the system's model, the filter predicts the drift angle.
- **Update Step:** The filter refines the predicted angle using the current sensor measurements: $\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$. Where K_k is the Kalman gain, z_k is the current measurement and H is the measurement matrix.

4 Application Developement

The implementation of **DrifTracker** follows a modular architecture, adhering to the principle of *separation of concerns*, ensuring that each component of the application is dedicated to a specific functionality.

4.1 Core Data

Starting from the lowest level of the application — data persistence — user data is stored directly on their device. This decision was made to simplify the initial version of the application, as it represents a practical and manageable solution for the early stages of development. To achieve this, *Core Data*, Apple's persistence framework, had been chosen because to its maturity and stability, its support for advanced features, and its compatibility with iOS versions prior to 17. This framework efficiently handles the application's data structure, facilitating both the storage and retrieval of information, while offering greater flexibility for potential future requirements.
The data model is organized into entities specifically designed to manage reference sessions (*ReferenceSession*) and drift sessions (*DriftSession*). It also includes entities for telemetry samples (*ReferenceEntry* and *DriftEntry*) and stress-related data (*StressEntry*), all of which are linked to these sessions.

4.2 Managers

To handle business logic and backend operations, the application primarily relies on two main managers, which enable the gathering and processing of data from the relevant sensors. The **GeneralManager**, based on Apple's *CoreLocation* and *CoreMotion* frameworks, handles GPS and gyroscope data, recording sessions and telemetry data, and computing drift and performance metrics in real time. It stores session data in Core Data and incorporates a Kalman filter to enhance motion data accuracy, using a pre-existing implementation that it was adapted for the project needs [6].
To compute the drift angle, the application records a reference session with multiple samples, each containing latitude, longitude, and gyroscope Z data. During the drift session, the current sample is matched to the closest reference sample using the Haversine distance function. Subsequently, the drift angle is calculated as:

$$Drift\ Angle = gyroZ_{ref} - gyroZ_{drift} - offset$$

where the offset is the difference between the gyroscope Z value at the end of the reference session and the gyroscope Z value at the start of the drift session.
The **HealthManager** retrieves HRV data from Apple HealthKit, in order to monitor stress levels and save this data for each drift session.

4.3 User Interface (UI) Components

The user interface of DrifTracker has been designed to prioritize simplicity and user-friendliness, while offering all the functionalities needed for an optimal user experience.
The core of the app is the main dashboard, called **TrackerView**. This dashboard provides users with real-time telemetry data: drift angles, speed, and a visualization of their reference path, integrated with a map. This view utilizes the backend functionalities provided by the GeneralManager to update performance metrics in real time, giving accurate and quick feedback during training sessions.
A significant component of the interface is the reference management system. Users can record and manage reference sessions, which are essential for calculating accurate drift angles. Through the **ReferenceView**, users can select, rename, delete, or search for specific reference sessions, making it easy to set a baseline trajectory for comparisons.
The app also includes a **HistoryView**, allowing users to review the details of their past training sessions, grouped by day. This feature not only lets users explore their performance data but also enables them to rename sessions, export data as CSV files, and delete or search for specific sessions. Additionally, the stress analysis feature is integrated into this view, presenting a visual representation of HRV trends for each session to help users understand the impact of stress on their performance.
Last but not least, to ensure compliance with privacy regulations, the app includes a dedicated **ConsentView**. Upon first use, the app prompts users to grant permissions for location tracking and sensor access. From the consent tab, users can enable or disable these permissions and adjust their preferences by navigating to the phone's settings.

5 Testing and Data Collection

5.1 Methodology

The testing process focused on verifying the accuracy of sensors for measuring drift angles and speed, ensuring the reliability of biometric data, and evaluating the interface's usability during real training sessions.
Initial functionality tests were performed during development, using walking simulations to mimic drift behavior and test angle and speed calculations.
Field tests followed, with the iPhone mounted on the car and Apple Watches used to measure HRV, involving both a professional and a beginner driver during an entire afternoon of training session. During this time, the drivers recorded a straight reference, which was used as a baseline. They then performed a simple "parking" exercise, aiming to finish in the parking spot with an angle of 90° with respect to the starting line. This approach validated the app's performance across varying skill levels while gathering data for further analysis.

5.2 Feedback and Improvements

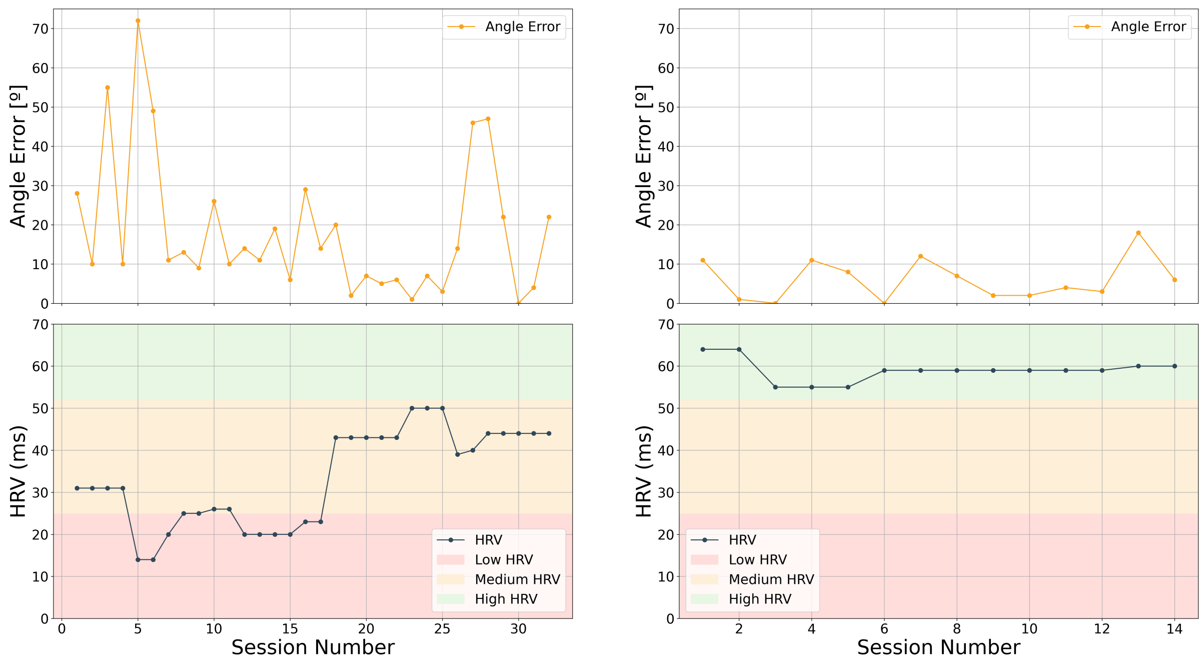
During testing, several issues were identified and resolved to improve the app's accuracy and usability:

- An offset was introduced to correct drift angle calculations caused by differences in the gyroscope's orientation between the end of the reference session and the start of the current drift session.

- A modulo function was added to ensure angles remained within 0°–360° during continuous rotations.
- It was observed that the phone must remain fixed in a horizontal position in the car, as a vertical placement changes the gyroscope's axis, affecting calculations that relied on the Z-axes.

6 Data Analysis

Using the collected data, an analysis was conducted to extract meaningful insights by comparing the performance of the two drivers: the professional and the beginner.



(a) Beginner (b) Pro

Figure 2: Performance comparison between beginner and professional driver across the sessions. The top plots illustrate angle error trends, representing the difference among the angle achieved in the exercise and the target angle of 90°, while the bottom plots show HRV levels.

The charts in Fig.2 provide a clear comparison between the beginner (2a) and professional driver (2b) in terms of performance and stress levels, as reflected by HRV. For the beginner, the data shows a strong correlation between high stress levels (low HRV values) and inconsistent performance, characterized by significant variability and larger drift angle errors. Over the sessions, as stress levels decreased, the beginner's performance improved, leading to more consistent results with reduced angle errors. In contrast, the professional driver maintained steady performance throughout the sessions, with minimal drift angle errors and consistently low stress levels, as indicated by stable HRV values. This highlights the professional's ability to maintain control and accuracy under similar conditions, likely due to greater experience and familiarity with the activity.

Metric	Beginner	Pro
Angle Error STD (σ)	17.61°	5.38°
Angle Error Mean	18.5°	6.07°
HRV STD (σ)	11.45 ms	2.77 ms
HRV Mean	32.25 ms	59 ms

Table 2: Comparison of Key Metrics Between Beginner and Pro Drivers

The Table 2 further emphasizes these differences quantitatively: the professional driver exhibits a much lower angle error standard deviation and higher mean HRV compared to the beginner, showcasing a clear distinction in skill level and stress management.

7 Future Directions

DrifTracker has proven to be a valuable tool for drift training, offering drivers an easily accessible solution by allowing them to monitor their performances directly using their own phone.
Although the current version meets the primary objectives, there is room for improvement and expansion to make the app even more versatile and user-friendly:

- Improve the app's stability and usability to ensure higher reliability during training sessions.
- Integrate smartwatch functionality, allowing users to control sessions directly from their wrist for greater convenience and accessibility.
- Introduce real-time communication features, enabling external spotters to monitor a driver's performance remotely and provide immediate feedback.
- Develop centralized cloud storage for securely storing session data and enabling advanced analytics and performance predictions.
- Incorporate a dashcam feature to visually record sessions, providing an additional layer of data for performance review and analysis.

References

[1] Y. Akcayir and Y. Ozkazanc. "Gyroscope drift estimation analysis in land navigation systems". In: *IEEE Conference on Control Applications* (2003).

[2] Arduino. *Nicla Sense ME.* Accessed: 2024-12-23. 2024. URL: `https://docs.arduino.cc/hardware/nicla-sense-me/`.

[3] GoPro. *Open GoPro Developer Documentation.* Accessed: 2024-12-23. 2024. URL: `https://gopro.github.io/OpenGoPro/`.

[4] *Hydra Data Analysis.* Accessed: 2024-12-23. 2024. URL: `https://www.hydradataanalysis.com/UseCases/DriftingTelemetry`.

[5] Apple Inc. *Apple Developer Documentation.* Accessed: 2024-12-23. 2024. URL: `https://developer.apple.com/documentation/`.

[6] *Kalman Filter Implementation in Swift.* Accessed: 2024-12-23. 2024. URL: `https://github.com/wearereasonablepeople/KalmanFilter`.

[7] Hye-Geum Kim et al. "Stress and Heart Rate Variability: A Meta-Analysis and Review of the Literature". In: *Psychiatry Investigation* (2018).