CAN you Feel it?
An Immersive and Tangible Car Simulation Experience

Carlos Gabriel Pereira[1], **David Ohara**[1], **Amadeo Neto**[1], **Fatima Nunes** [2], **João Marcelo Teixeira**[1,2]

[1] Voxar Labs, Universidade Federal de Pernambuco (UFPE)
Recife, Brazil

[2]EACH, Universidade de São Paulo (USP)
São Paulo, Brazil

Abstract. *We present a Work-in-Progress (WiP) Virtual Reality (VR) system that turns a real vehicle into an input controller for its own 3D twin. Using a commodity On-Board Diagnostics II (OBD-II) Bluetooth adapter attached to a BYD Dolphin Plus, we extract real-time steering wheel angle, longitudinal acceleration, and brake status and forward them to a Meta Quest 3 headset. In the current prototype, a PC acts as a bridge between the OBD-II link and the Head Mounted Display (HMD); the VR application, implemented in Unity, renders a full-scale digital model registered to the real car via spatial anchors, allowing users to physically touch the real vehicle while seeing and interacting with its virtual counterpart. We report feasibility results of Bluetooth protocol inspection for the BYD Dolphin Plus, low-latency telemetry forwarding, and anchor stability indoors. We outline the remaining steps to remove the PC bridge and connect the OBD-II adapter directly to the Quest 3, enabling standalone operation. Our findings indicate that real-vehicle telemetry can drive a collocated VR twin with low end-to-end latency, supporting the broader vision of using the physical car as a tangible interface in training and demonstration scenarios.*

Keywords: Virtual Reality, OBD-II, Spatial Anchors, Digital Twin

1. Introduction

Training and demonstration scenarios for automotive safety, eco-driving, and Human Machine Interaction (HMI) studies benefit from VR's controllability while retaining real-world tangibility [Borges et al. 2025], since they can enhance visuo-tactile experience and immersion feeling by accurately pairing real and virtual elements in an environment that the user can feel and control the vehicle. We noticed that few studies have developed a similar approach in the vehicular context. In this study, we investigate whether a commodity OBD-II Bluetooth adapter can feed real-time vehicle telemetry to a Meta Quest 3 such that a collocated 3D twin mirrors the physical car. By spatially aligning the twin to the real vehicle, users can touch what they see, receiving authentic haptic feedback from the physical chassis while interacting with a simulated environment, which can be used to simulate driving under safer and accident-free (no real collisions and obstacles) and low-cost (reduced car parts wearing and less energy consuming) conditions, provided by the virtual environment interaction, where the user's real world location is kept.

Our paper reports main intermediate results as a protocol inspection and decoding of the BYD Dolphin Plus OBD-II streams, a bridge pipeline (vehicle $\rightarrow$ PC $\rightarrow$ Quest 3) for rapid iteration, and a preliminary latency and anchor-stability observations, where the final target is a *PC-free* setup (OBD-II $\leftrightarrow$ Quest 3), using the real car as a controller for the simulated car.

2. Related Work

Research on immersive driving simulation has compared HMDs with fixed-display setups in controlled tasks, generally finding comparable performance and realism with mixed effects on sickness depending on the scenario and visual demands [Blissing et al. 2022, Himmels et al. 2023]. In parallel, VR-based vehicular digital twins have been used to study human–vehicle interaction and HMI strategies in safety-critical scenes, linking real-world layouts to interactive, immersive experiments [Serrano et al. 2023].

For maintaining geometric registration between a virtual replica and its physical counterpart, platform spatial-anchoring APIs expose persistent, world-locked frames that can be shared across sessions and devices; best-practice guidance emphasizes anchor placement, re-localization, and drift monitoring [Meta 2024, Microsoft 2025]. Beyond platform documentation, empirical work on drift tolerance and registration accuracy informs design choices such as session length, periodic re-checks, and use of fiducials when needed [Benz et al. 2019, Scargill et al. 2021].

Furthermore, a real-virtual car pairing study was conducted [Weiss and Gerdes 2023], where the authors use high-precision onboard sensors (GPS/INS) to feed a reference dynamics model, which scales the vehicle's real motion to provide visual, vestibular, and haptic feedback in VR. In contrast, our approach directly uses ELM327 data from the car's ECU (acceleration, brake, and steering) to drive the VR simulation without complex dynamics modeling. Moreover, AR DriveSim [Gabbard et al. 2019], an immersive and low-cost driving simulator that integrates a real vehicle cab, projection-based immersion, physiological monitoring, and force-feedback steering to study the impact of augmented reality head-up display (AR HUD) navigation graphics on driver performance, attention, and preferences, concluding that conformal AR graphics are not inherently superior and may increase cognitive workload in certain contexts. However, we explore the feasibility of real-vehicle integration with virtual reality for training and demonstration scenarios, highlighting low-latency telemetry forwarding, perceptually stable spatial anchoring, and tangible interaction with the physical car as key contributions.

Finally, commodity OBD-II remains a practical telemetry source for prototyping real-time VR/AR applications around vehicles. While standardized PIDs cover core powertrain signals, EVs often rely on manufacturer-specific data identifiers (DIDs) accessed via UDS service 0x22 [CSS Electronics nd], and Bluetooth-linked adapters such as the ELM327 [ELM Electronics nd] are commonly used in research prototypes. Our system follows this trend, combining OBD-II telemetry with spatial anchors on a standalone HMD to realize a collocated, full-scale car twin.

3. System Overview

Figure 1 presents the current system communication pipeline. In the current setup, an ELM327 device retrieves telemetry data from the BYD device and transmits it via Bluetooth to a PC, which relays it to the Quest 3 via an HTTP server. In the target pipeline, the HTTP Server will be removed to allow data to be transmitted directly to the Quest 3 over Bluetooth. The Unity app then renders a 1:1 car model using spatial anchors.

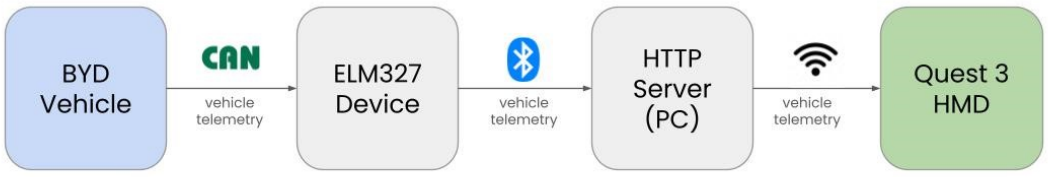

Figure 1. Current communication pipeline. In the target pipeline, the HTTP server is removed to streamline communication.

3.1. Hardware

- **Vehicle:** BYD Dolphin Plus (EV).
- **OBD-II adapter:** ELM327-class Bluetooth adapter (BLE/Classic).
- **HMD:** Meta Quest 3.
- **PC bridge:** Laptop with Bluetooth and Wi-Fi to forward telemetry.

3.2. Signals

The ELM327 is a common OBD-II adapter that translates vehicle diagnostics into a serial interface, controlled through AT commands (its configuration language, e.g., for protocol selection). Standard sensor values are usually accessed via PIDs (Parameter IDs), while many manufacturer-specific signals require DIDs (Data Identifiers, accessed with service 0x22). Each module in the car, such as the engine or brake controller, is an ECU (Electronic Control Unit) that responds at its own address. In the final setup, the Quest 3 receives the data from JSON via HTTP.

We used an ELM327 adapter connected to the car and the Car Scanner app to check if steering angle, accelerator, and brake pedal values could be read via Bluetooth. Since the app successfully retrieved them, we attempted to access the same data from a PC using Python. The BYD, however, does not expose these signals through standard OBD-II PIDs but via Diagnostic IDs (DIDs) using service 0x22.

To map them, we scanned ECU addresses (0x000–0x7FF, 11-bit) and identified which responded. For each responding ECU, we iterated over possible DIDs (0x0000–0xFFFF). We logged those that returned data and monitored which ones varied when pressing pedals or turning the steering wheel. This way, accelerator and brake variables were identified. The steering angle signal was discovered separately by sniffing communication between Car Scanner and the ELM327, using a Python script.

The recovered DIDs were initially polled by a Python service on a PC, which parsed responses and forwarded compact JSON packets $\langle$s, a, b$\rangle$ (steering weel, accelerator, brake) to the Quest 3. In the final setup, the PC bridge will be removed, with the Quest 3 handling Bluetooth communication, DID requests, and on-device parsing, reducing latency while preserving the exact signal set.

We target three real-time channels minimal for a convincing twin:

1. Steering wheel: **ECU 783, DID 0003**, range $\approx -540^\circ$ to 540° (degrees).
2. Acceleration: **ECU 7E2, DID 000B**, range 0% to 100% (dimensionless).
3. Brake: **ECU 7E2, DID 000C**, range 0% to 100% (dimensionless).

3.3. Software Stack

The bridge service PC opens a Bluetooth link to the OBD-II adapter, issues initialization commands (e.g., `ATZ`, echo off, protocol auto), polls target DIDs at ~ 8.4Hz (mix of standard and vendor-specific requests), parses responses, and forwards compact messages via HTTPS to the Quest 3 over Wi-Fi.

The Unity app consumes telemetry packets, updates the car model (steering wheel angle, throttle/brake visual cues), and maintains a persistent spatial anchor to align the 3D twin with the physical car. A calibration routine binds the twin's coordinate frame to the anchor using fiducials or manual pose adjustment.

Telemetry is serialized as a compact JSON:

$$\langle s, a, b \rangle$$

where s is the steering wheel angle, a is the longitudinal acceleration, and b is the brake proportion captured. Messages are requested by the Unity application every frame update.

3.4. Real-Virtual Scale Correlation

Before the spatial anchor alignment, we used Room Scan Exporter app, that access Quest 3's Assisted Space Setup system, to scan the real car and generate its 3D mesh in its actual size, which we can adjust the virtual model scale match the real one [ZoyncTech 2025].

3.5. Spatial Alignment

Using Quest 3's video passthrough, the spatial anchors overlays real world placements with virtual elements that are moved with device's controllers, which can difficult visibility at placing bigger objects, like the car, as its size overlaps the real car reference. To better perform the real-virtual alignment, at first, we manually place only the virtual car's steering wheel, a smaller part, to monitor the anchor drift by measuring re-projection errors of the reference points on the real steering wheel. Next, with both virtual and real objects being aligned, we render the complete car's 3D model after the placed steering wheel. After the virtual car's calibration and placement, its position is saved, so it can be loaded between sessions, as its real world placement is defined. The spatial anchor and the virtual car can be seen in Figure 2.

4. Preliminary Results

We successfully issued initialization and periodic DID requests to the BYD Dolphin Plus via a commodity adapter. Steering wheel angle, accelerometer and break statuses readings were stable. Although we have not yet performed quantitative drift measurements, qualitative observation indicates that the virtual–real registration remains perceptually stable during indoor sessions under steady lighting (as shown in Figure 2). In practice, users can place their hands on the virtual steering wheel and interior controls while touching the corresponding physical, reporting a convincing "touch-through" alignment with no noticeable slip or drift over several minutes; brief recalibration is only needed after large user motions around the vehicle. It is important to note that spatial anchors and vehicle-to-app communication are not yet integrated in the current prototype. The communication pipeline was tested using a laptop running the Unity Engine in Android development

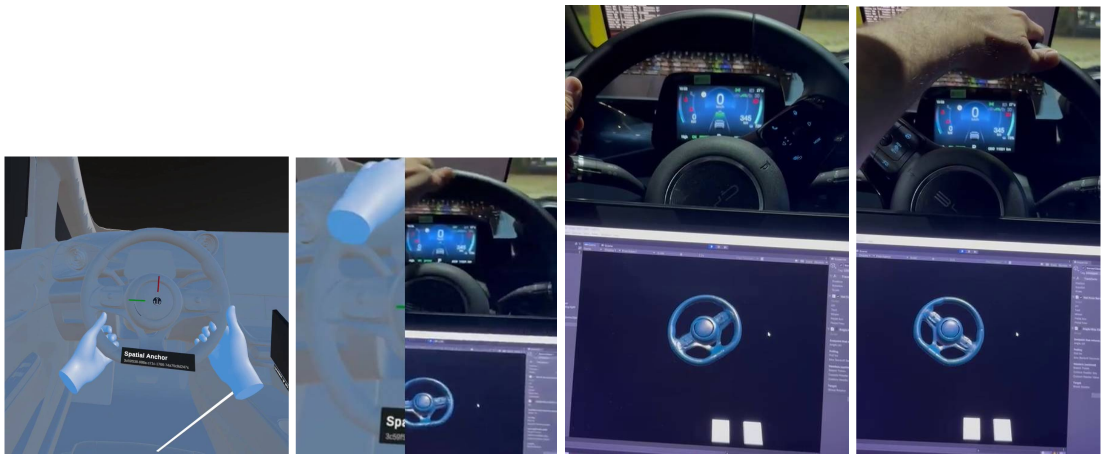

Figure 2. Spatial anchor placement test inside the car, and Complete test capturing information from car to Unity application running in real time.

mode, while spatial anchors were tested separately on a standalone Quest 3 build. Additionally, Bluetooth packet request tests were performed between the ELM327 adapter and the laptop, yielding a maximum/optimal request rate of 8.4 Hz, with about 96% of valid packets (invalid packets correspond to losses or inconsistent data). This results in an effective rate of approximately **8Hz** across the three sensors, i.e., about **2.66Hz** update rate per sensor.

5. Next Steps

This ongoing research has the following next steps:

- **Direct Quest 3 OBD-II.** Implement BLE stack and ELM327 session on-device; remove PC bridge.
- **Robust alignment.** Combine the developed system with the virtual car to show real time responsive feedback at steering wheel and pedals usage.
- **User study.** Compare presence, perceived responsiveness, and task performance with/without real-car tangibility.
- **Integration.** Integrate the spatial anchor system with the communication pipeline between the vehicle and the HMD.

All experiments were conducted on a private property with the car parked with engine-off configuration. No on-road use is planned for this solution, the car will remain stationary.

6. Conclusion

In this paper, we demonstrated a WiP system that couples a real car and its VR twin through commodity OBD-II telemetry and spatial anchors on a Meta Quest 3. Through our system, we demonstrated the feasibility of integrating a real vehicle with a VR application, enabling changes from the real vehicle to be reflected on its digital replica, albeit through two separate experiments. Early results indicate low latency and stable registration in indoor scenarios. Ongoing work targets a fully standalone HMD pipeline, without the need for the HTTP Server, and broader signal coverage to support training and demonstration use cases.

References

Benz, T. M., Riedl, B., and Chuang, L. L. (2019). Projection displays induce less simulator sickness than head-mounted displays in a real vehicle driving simulator. In *Proceedings of the 11th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, pages 379–387.

Blissing, B., Bruzelius, F., and Eriksson, O. (2022). The effects on driving behavior when using a head-mounted display in a dynamic driving simulator. *ACM Transactions on Applied Perception (TAP)*, 19(1):1–18.

Borges, L. F. M. R., Miranda, A. S., Ferreira, J. V. L., Lazarini, T. P., Neto, A. D. A., Novais, M. G., Schimidt, M. Q., Andreão, R. V., and Mestria, M. (2025). Virtual reality system for inspection and training: A case study in wheel-loader operations. *Journal on Interactive Systems*, 16(1):288–301.

CSS Electronics (n.d.). Uds protocol tutorial - unified diagnostic services. `https://www.csselectronics.com/pages/uds-protocol-tutorial-unified-diagnostic-services`. Accessed: 2025-08-01.

ELM Electronics (n.d.). Elm327 datasheet. `https://cdn.sparkfun.com/assets/learn_tutorials/8/3/ELM327DS.pdf`. Accessed: 2025-08-01.

Gabbard, J. L., Smith, M., Tanous, K., Kim, H., and Jonas, B. (2019). Ar drivesim: An immersive driving simulator for augmented reality head-up display research. *Frontiers in Robotics and AI*, Volume 6 - 2019.

Himmels, C., Andreev, V., Syed, A. A., Lindner, J., Denk, F., and Riener, A. (2023). Are head-mounted displays really not suitable for driving simulation? a comparison with a screen-based simulator. In *2023 IEEE intelligent vehicles symposium (IV)*, pages 1–6. IEEE.

Meta (2024). Spatial anchors overview (unity). `https://developers.meta.com/horizon/documentation/unity/unity-spatial-anchors-overview/`. Accessed 2025-08-01.

Microsoft (2025). Spatial anchors—mixed reality design guidelines. `https://learn.microsoft.com/windows/mixed-reality/design/spatial-anchors`. Accessed 2025-08-01.

Scargill, T., Chen, J., and Gorlatova, M. (2021). Here to stay: Measuring hologram stability in markerless smartphone augmented reality.

Serrano, S. M., Izquierdo, R., Daza, I. G., Sotelo, M. A., and Llorca, D. F. (2023). Digital twin in virtual reality for human-vehicle interactions in the context of autonomous driving. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 590–595. IEEE.

Weiss, E. and Gerdes, J. C. (2023). High speed emulation in a vehicle-in-the-loop driving simulator. *IEEE Transactions on Intelligent Vehicles*, 8(2):1826–1836.

ZoyncTech (2025). Room scan exporter for meta quest 3. `http://zoynctech.itch.io/room-scan-exporter-for-meta-quest-3`. Accessed: 2025-08-01.