DELIVERED PERFORMANCE

Simulator latency.

Stage-by-stage measurement of physical-input-to-motion latency through the DDX Precision pipeline, from button contact to accelerometer-detected platform motion.

> Latency is perhaps the single most important factor in whether a simulated vehicle controls like its real-world counterpart. Racing-grade simulation demands motion latency well below 10 ms: a car at the limit is balanced on the edge of stability, acutely sensitive to every input, and added lag - detectable from just 12 ms - pushes corrective inputs toward overcorrection, oscillation, and loss of control. Larger delays, and especially timing mismatches between channels, break the connection between input (wheel, pedal, button) and output (display, control loading, motion cueing), producing conflicting cues and simulator sickness.

VERSION 3.0 / OCTOBER 30, 2026
DRIVEN DYNAMIX
drivendynamix.com

Latency is perhaps the single most important factor in whether a simulated vehicle controls like its real-world counterpart. Racing-grade simulation demands motion latency well below 10 ms: a car at the limit is balanced on the edge of stability, acutely sensitive to every input, and added lag - detectable from just 12 ms - pushes corrective inputs toward overcorrection, oscillation, and loss of control. Larger delays, and especially timing mismatches between channels, break the connection between input (wheel, pedal, button) and output (display, control loading, motion cueing), producing conflicting cues and simulator sickness.

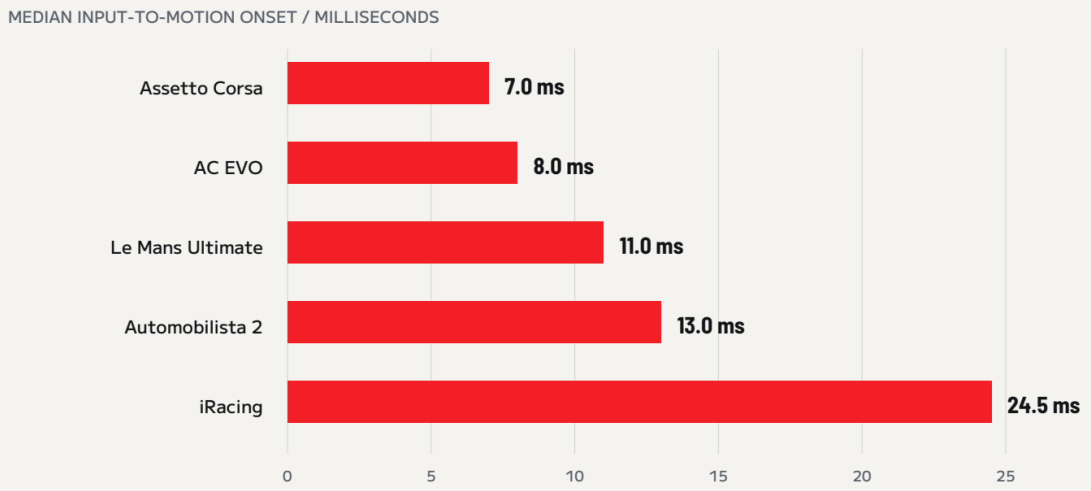
Yet latency is often marketed as a single "too good to be true" number (0 ms? 1 ms?) with no explanation of what was measured: input-to-motion, telemetry-to-motion, command-to-motion, servo response, or just the loop period of a sub-component? Is the input a square wave (infinite acceleration), or typical shaped motion?

In anonymized DDX reference evidence, observed physical-input-to-platform-motion onset was approximately 40 ms, 80-120 ms, and 200 ms across three selected non-DDX motion systems. Separately, another system used a 40 Hz command cadence—25 ms between commands before downstream filtering or actuation—and one professional vendor published a figure greater than 30 ms without a sufficiently defined boundary. These are heterogeneous market reference points, not a controlled head-to-head ranking. They are far more believable than a single-digit marketing figure: USB input, simulation, filtering, motion cueing, and electromechanical response time add up quickly if not controlled.

Summary of results

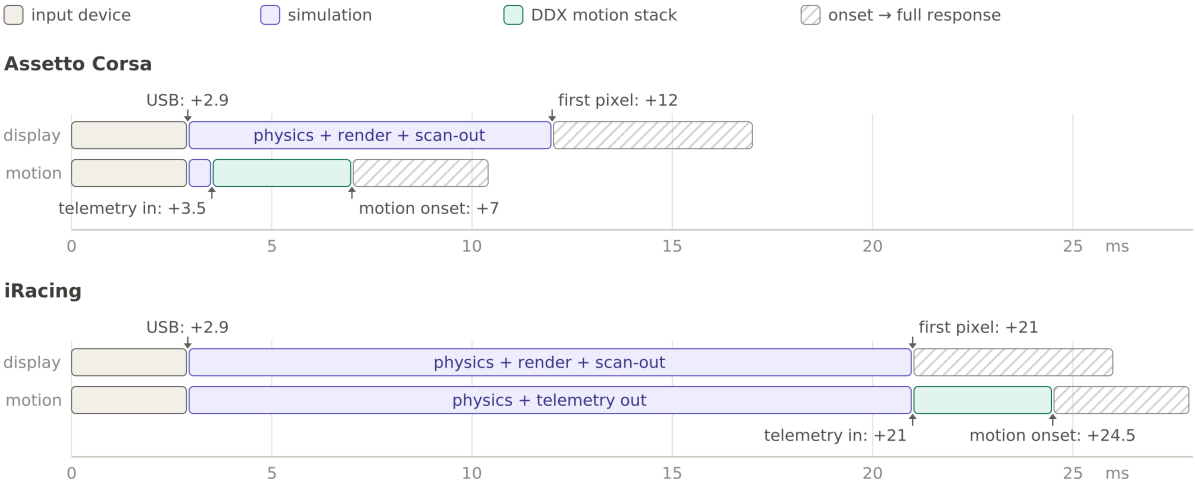
- Across five tested racing simulations on DDX Precision, median physical-input-to-platform-motion onset ranged from 7.0 to 24.5 ms across 95 trials. Per-title medians were 7.0 ms in Assetto Corsa, 8.0 ms in AC EVO, 11.0 ms in Le Mans Ultimate, 13.0 ms in Automobilista 2, and 24.5 ms in iRacing.
- DDX Precision and DDX Prime operate within an approximately 2.5-3.5 ms telemetry-to-motion latency envelope, measured from raw simulation telemetry arriving at Dynamo to accelerometer-detected physical motion onset. On the characterized production DDX Precision path, onset measured approximately 3.45 ms and peak acceleration occurred at approximately 6.85 ms.
- Measured at the median in iRacing: input, physics, and telemetry take about 21 ms, and the DDX software, hardware, electrical, and mechanical stack adds roughly 3.5 ms on top of that, for a total of 24.5 ms from physical input to motion onset. Conventional interpolation smoothing would add approximately 17 ms more - 42 ms in total.
- On a tested USB wheel-rim input, physical button-contact to outbound polled USB status-packet latency measured 0.4-4.3 ms across 21 trials, averaging 2.7 ms with a 2.9 ms median.

- Across four tested racing simulations with reliable UI timing, average physical-input-to-first-pixel latency ranged from 11 to 23 ms over 40 trials. In Assetto Corsa, the platform was already moving after a median 7 ms - before the display produced its first changed pixel (measured at an average 11 ms).



Median physical-input-to-motion onset by simulation on DDX Precision: 95 trials, button contact to first accelerometer-detected platform acceleration.

This document measures the actual latency of each stage in the DDX Precision's pipeline.

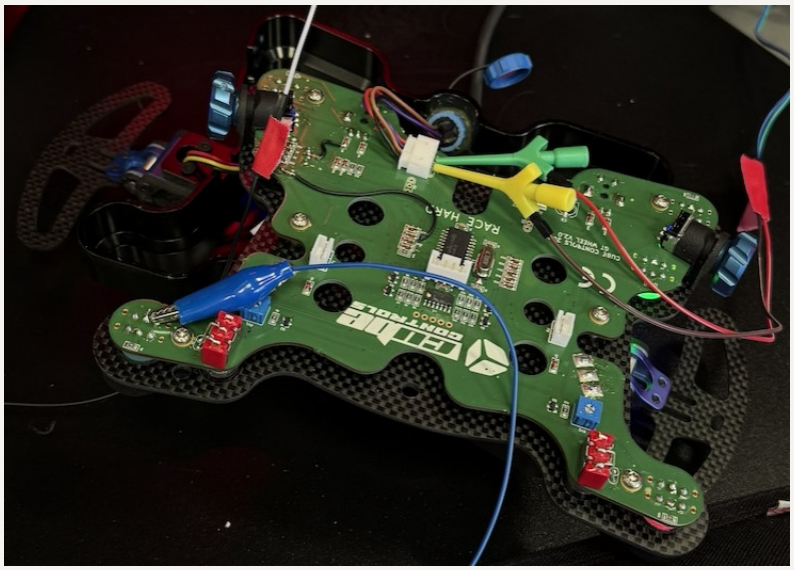


Median timelines for Assetto Corsa and iRacing, from physical input to display and to motion: the input device, the simulation, the DDX motion stack, and the onset-to-full-response tail.

We'll explore each of the bracketed time spans in more detail below.

Physical Input to USB

A simulator's physical inputs are our first stop. From the moment a button (or pedal, or wheel) is pressed, the device's microcontroller can have a mix of polling, debounce, and filtering delays, on top of the regular cadence interval of USB communication.



Input latency test rig: oscilloscope probes on the button contact and the USB data pins.

To measure the time from physical button press to USB packet, we attached oscilloscope probes to the button contact and USB data pins on a Cube Controls GT v2 wheel rim.

On a tested USB wheel-rim input, physical button-contact to outbound polled USB status-packet latency measured 0.4-4.3 ms across 21 trials, averaging 2.7 ms with a 2.9 ms median.

Input to Display

Display latency - in this case, measuring the total latency from physical input to visual response. Not directly related to motion cueing, but important in professional simulators for aligning the various outputs to arrive at similar times (display, audio, wheel loading, and motion cueing). It also serves as a useful baseline to compare motion cueing latency against, given that the input to simulation to render to display pipeline is a well-understood optimization area from the gaming industry for reaction time.

To measure the time from physical input to display output, we modified our input latency test rig to include an LED wired to the button contacts and a 960 Hz camera (GoPro MISSION 1 Pro).



Display latency test setup: LED on the button contacts, 960 Hz camera on the panel.

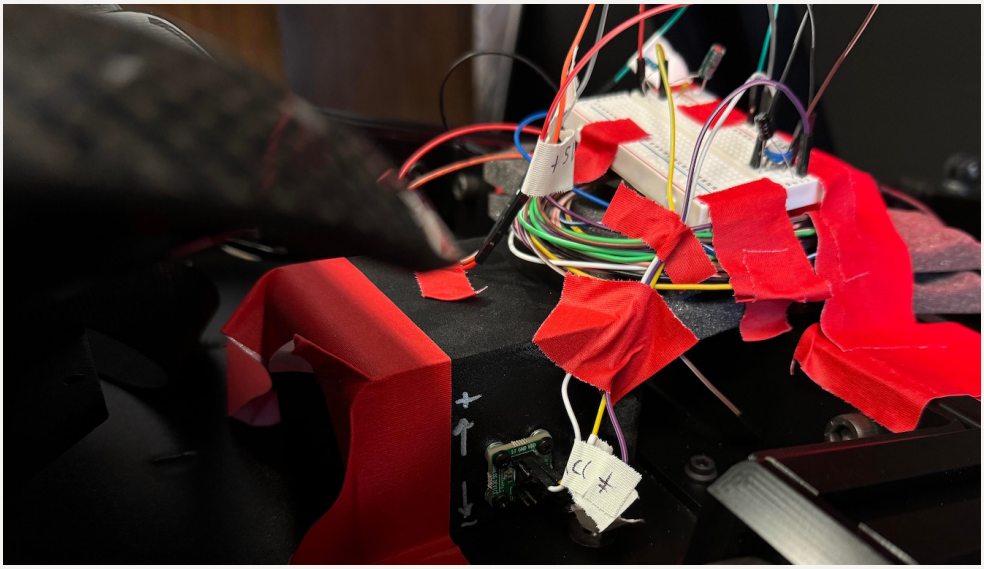
The display used in this test is a Samsung S49CG950EN, a 240 Hz VA-type panel. We measured a pixel transition time of 2 - 5 ms. Timing below is measured from physical input trigger (initial LED illumination) to initial pixel transition onset (first visible change in pixel value). Input is mapped to brake, clutch, or gear upshift, depending on the simulation; latency is time until simulation UI reflects the input. Delayed inputs (such as shifting out of neutral in some titles) were discarded; v-sync and pre-rendered frames were disabled.

Simulation	Samples	Display Latency Avg (ms)	Display Latency Min (ms)	Display Latency Max (ms)	Approximate Framerate (Hz)
Assetto Corsa	5	11	8	14	1400
Automobilista 2	12	12	8	15	800
iRacing	8	23	16	31	700
Le Mans Ultimate	15	19	12	29	600
AC EVO	21	--	26	61	250

* AC EVO UI renders at 40 ms (or every 10 frames) cadence, making measurements unreliable.

Across four tested racing simulations with reliable UI timing, average physical-input-to-first-pixel latency ranged from 11 to 23 ms over 40 trials. A fifth title, AC EVO, was excluded from the average range because its coarse UI cadence made onset timing unreliable.

Input to Motion

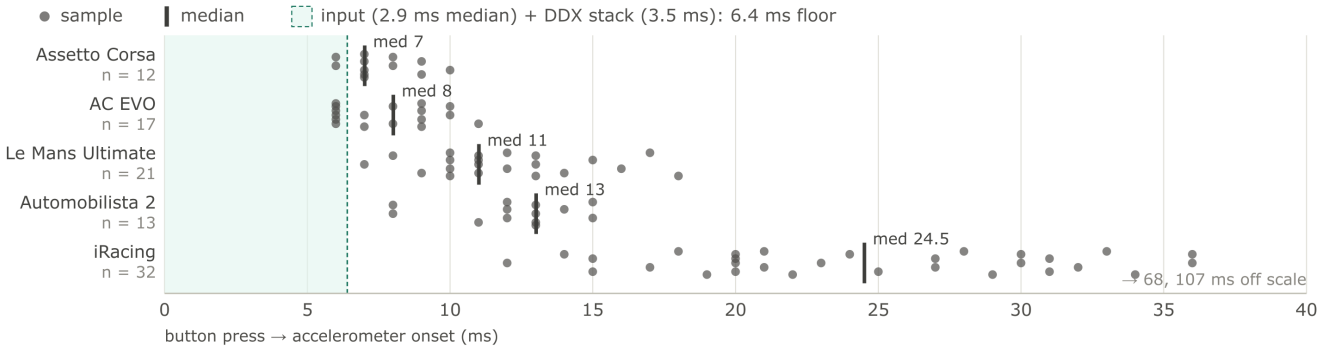


Analog Devices ADXL1002 accelerometer test rig, on the same oscilloscope as the button probe.

To measure the time from physical input to motion cue output, we modify our test setup to include a high-speed analog accelerometer, hooked up to the same oscilloscope as our button probe trigger. Time is measured from button press to accelerometer onset.

Simulation	Samples	Average (ms)	Median (ms)	Min (ms)	Max (ms)
Assetto Corsa	12	7.6	7	6	10
AC EVO	17	7.8	8	6	11
Automobilista 2	13	12.2	13	8	15
iRacing	32	28.3	24.5	12	107
Le Mans Ultimate	21	12.0	11	7	18

* 360 Hz telemetry disabled. iRacing's 360 Hz mode is 6 subsamples at 60 Hz, adding approximately 14 ms buffering delay.



Every input-to-motion trial, by title. The shaded band is the floor set by the input device and the DDX stack: 2.9 ms plus 3.5 ms.

Across five tested racing simulations on DDX Precision, median physical-input-to-platform-motion onset ranged from 7.0 to 24.5 ms across 95 trials.

Every sample sits above a 6.4 ms floor - the input device's 2.9 ms median plus the DDX stack's 3.5 ms. Nothing below that line is available to any motion system on this input path, and the spread above it belongs to the simulation.

Command to Motion

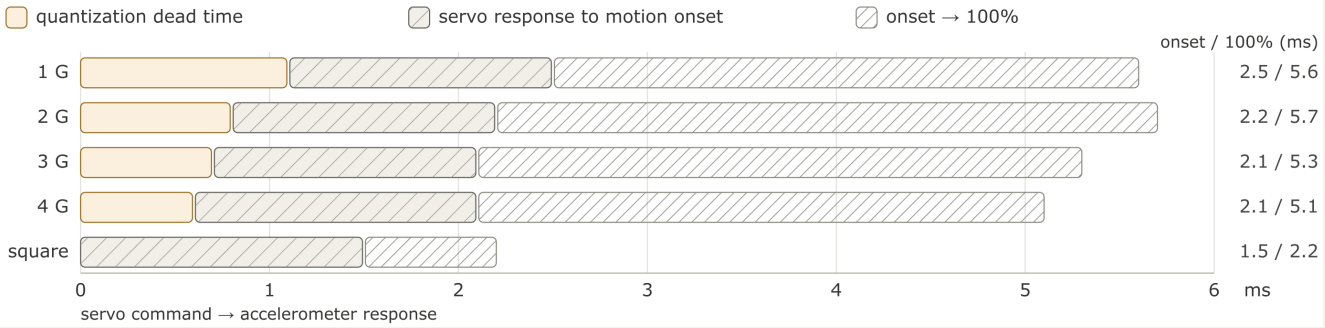
Next, we measure the responsiveness of our servo motor and drive to commands from our motion controller. DDX simulators use industrial-grade servos from US manufacturer Teknic, enabling a substantially tighter control loop, faster response time, and higher bandwidth than is seen on typical consumer motion simulators.

As measured on our servo drive's integrated scope, the time from command received to full torque output is between 1.2 - 2.4 ms; between one and two iterations of the scope's sampling rate. We should expect to see our results fall in this 1.2 to 2.4 ms range when sampled via our oscilloscope and accelerometer harness.

The time from our motion controller emitting a position command to acceleration onset is measured using a GPIO trigger from the microcontroller, emitted on position target update, versus response on our analog accelerometer.

For all inputs except the square wave (infinite acceleration), there is an initial dead time due to our command resolution on DDX Precision rounding the first moments of the move shape to zero. This effect is predictable and can be subtracted out to determine the command response latency. DDX Prime uses an upgraded servo package with significantly finer command resolution.

Input (G)	uC Command to Motion Onset (ms)	uC Command to 100% Acceleration (ms)	Onset: Quantization Dead Time (ms)	Onset Residual (ms)	100% Residual (ms)
1	2.5	5.6	1.1	1.4	4.5
2	2.2	5.7	0.8	1.4	4.9
3	2.1	5.3	0.7	1.4	4.6
4	2.1	5.1	0.6	1.5	4.5
Square Wave	1.5	2.2	0.0	1.5	2.2



Commanded motion against accelerometer response, by input shape: quantization dead time, servo response to motion onset, and onset to full acceleration.

After adjusting for command quantization dead time, we see response onset in the 1.4 to 1.5 ms range. 100% response is at 2.2 ms (for step-response square wave input) and 4.5 to 4.9 ms for shaped acceleration inputs.

DDX Dynamo – Software Stack

Let's take a closer look at the software powering DDX Precision - DDX Dynamo. DDX Dynamo sits between the simulation's output telemetry and the physical motion stack. Its job is to translate the simulation's physics into motion and haptic cues faithfully, then convert those into motion commands, while introducing as little processing delay as possible.

Raw telemetry from consumer-grade simulations comes in at coarse and variable cadence, ranging from a consistent 60 Hz in iRacing, to a variable and jittered 200 - 400 Hz in Automobilista 2 (sampled from their internal 720 Hz). DDX Dynamo applies low-latency filtering to these raw telemetry frames - cadence fixing, outlier rejection, interpolation / extrapolation / prediction - to produce a reliable 1 kHz telemetry stream.

Excess filtering time, unreliable motion command cadence, and low command rate are traps in commonly used consumer motion cueing software. To mask these issues, motion controllers must use additional smoothing and relaxed servo control loops. These, in turn, hurt response time (motion lag), position authority (soft or uncontrolled motion), and haptic bandwidth (lost high-frequency details).

DDX's tightly-integrated, low-overhead motion stack avoids these issues - minimal filtering, minimal delay.

For timing measurements, we use high performance counters on the PC side (QPC). On our motion controller, we use clock cycle counts, and GPIO signals plus oscilloscope when we need to correlate to the outside world.

From raw telemetry received, to motion pose transmission to our microcontroller, we see a minimum time of 0.3 ms for our PC-side processing stack. Thread scheduling and telemetry arrival cadence can result in higher numbers, so 0.3 ms should be considered our filtering and cueing overhead. For intermediate 1 kHz filtered telemetry samples, the number is lower and more tightly bounded: 0.15 - 0.25 ms.

Our PC to motion controller pose transmission time is approximately 0.75 ms, of which 0.2 - 0.3 ms is driver stack and OS overhead to initiate the transfer. The remaining 0.5 ms is transmission time over the wire - an area of future improvement (packet compression and faster comms frequency).

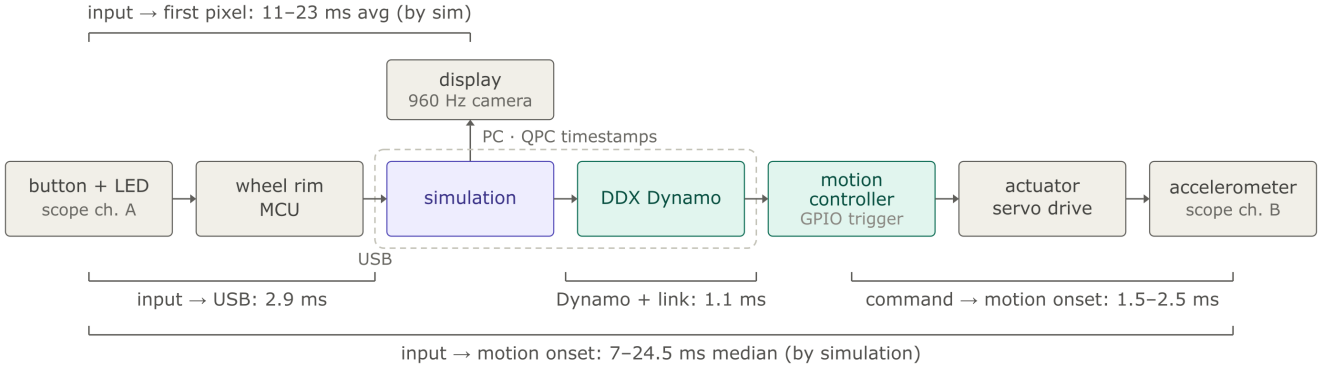
From motion controller pose received to motion onset: 2.4 ms. To peak acceleration, 5.8 ms. This matches our command to motion numbers above - there is no significant delay between receiving the motion pose from the PC, to orchestrating motion on the servos.

Measured, not claimed

On production DDX Precision hardware, the DDX-controlled path from raw simulation telemetry to the first detected platform acceleration totals approximately 3.45 ms: approximately 0.3 ms for Dynamo filtering and cue construction, approximately 0.75 ms for PC-to-controller communication, and 2.4 ms to motion onset.

End to end, median input-to-motion onset was 7 ms in Assetto Corsa, 8 ms in AC EVO, 11 ms in Le Mans Ultimate, 13 ms in Automobilista 2, and 24.5 ms in iRacing (with its 60 Hz telemetry). The input device contributed a median 2.9 ms and the DDX-controlled stages approximately 3.45 ms. These results sit well below the anonymized non-DDX reference points above. In Assetto Corsa, the platform was already moving after a median 7 ms - before the display produced its first changed pixel (measured at an average 11 ms).

This performance comes from the complete control stack, not a single fast component or a datasheet specification. DDX Dynamo converts irregular simulation telemetry into a clean 1 kHz stream, constructs motion cues with minimal filtering delay, and delivers commands at high cadence to our real-time motion controller. That reliable command stream allows us to run industrial-grade servos with tight, aggressive tuning. Many consumer systems must mask lower or irregular command rates with additional smoothing and relaxed servo loops. The result may look smooth, but it starts late, follows loosely, and suppresses the subtle and high-frequency information the driver needs. DDX can track demanding motion immediately and accurately because the software, communications, firmware, drives, and actuators were engineered as one real-time system.



Where each published latency figure starts and stops: button contact to USB, Dynamo and the controller link, command to motion onset, and the whole chain from input to motion.

This is also why an advertised "0 ms" or "1 ms" latency figure means very little without a defined measurement boundary. A software loop period, transmission time, servo specification, or response to an idealized step is not physical-input-to-platform latency. The figures in this report come from direct measurements of the actual pipeline, including button contacts, USB traffic, PC processing, controller communication, and accelerometer-detected platform motion. When evaluating any latency claim, ask three questions.

Where does the timer start? Where does it stop? How was physical motion detected? If those answers are missing, the number is marketing - not performance data.

The DDX stack is already fast enough that the simulation itself becomes the long pole.

DDX PreActiv attacks that remaining delay by predicting vehicle state across the telemetry cadence gap, instead of waiting for the next simulation frame. It turns simulation latency from an external limitation into another problem our software can solve.

"Software-Defined Motion", already published at drivendynamix.com/blogs/news/software-defined-motion, describes the stack this report measures. Two companion reports are to follow. "Motion Bandwidth" measures how much of a commanded cue survives the trip to the driver, and at which frequencies. "PreActiv User Study" measures what cue alignment and latency reduction are worth to drivers. To feel the difference now, book a demo at drivendynamix.com.

Study note: USB latency is physical contact transition to outbound USB status packet on one tested device, and does not generalize to all input hardware. Input-to-motion is button contact to first accelerometer-detected platform acceleration on one DDX Precision configuration; excitation amplitude, axis, frame rate and telemetry cadence were title-dependent, so this is an observed range rather than a controlled ranking of game engines. The 2.5-3.5 ms telemetry-to-motion envelope is motion-subsystem latency, not physical-input-to-motion latency, whole-simulator perceptual latency, or PreActiv effective latency. The 42 ms figure describes a conventional interpolated pipeline, not DDX's measured performance. Non-DDX motion systems remain anonymized: that evidence is heterogeneous and is not a controlled head-to-head comparison.

The sub-10 ms figure follows the published recommendation of Melzi & Previati (2025, J. Vib. Control) - the most sensitive of 28 drivers on a moving-base simulator discriminated a 12 ms increment of added mechanical latency, and the authors conclude racing applications warrant mechanical latency well below 10 ms - plus converging evidence. The smallest added latency tested in an unstable second-order control task, 14 ms, degraded performance before participants consciously perceived any delay, with effects growing as the system sensitivity to input rose (Martens et al. 2018, Qual. & User Exp.); a car at the tire limit is exactly such a system. Delay consumes stability margin in proportion to control bandwidth (McRuer & Jex 1967), and delayed steering destabilizes a coupled driver-vehicle model at speed (Giacintucci et al. 2024, IFAC-PapersOnLine). At larger delays and cross-channel timing mismatch: 170 - 340 ms visual/motion delays increased steering reversals and sickness (Frank, Casali & Wierwille 1988, Human Factors); reducing apparent display latency from approximately 78 to approximately 18 ms significantly reduced simulator sickness (Buker et al. 2012, Human Factors); and false or missing motion cues provoke disproportionately more sickness than merely scaled ones (Kolff et al. 2025, Transp. Res. Part F) - quantifying sensory-conflict theory (Reason & Brand 1975, Motion Sickness).

Source revision: ddx-kb@eb209e5fa7b4d33d389d96b8aac9032878e2570d
CLM-0024 / CLM-0025 / CLM-0026 / CLM-0027 / CLM-0028 / CLM-0031 / CLM-0034 / CLM-0127