Received: 1 October 2021 | Revised: 5 January 2022 | Accepted: 7 January 2022

DOI: 10.1002/jsid.1104

REGULAR CONTRIBUTED PAPER

WILEY

OpenLDAT—A system for the measurement of display latency metrics

Federico Dossena | **Andrea Trentini**

Dipartimento di Informatica, Universitá degli Studi di Milano, Milan, Italy

Correspondence
Andrea Trentini, Dipartimento di Informatica, Universitá degli Studi di Milano, Via Celoria 18, Milan, Italy.
Email: andrea.trentini@unimi.it

Abstract

The OpenLDAT project (short for *Open Latency Display and Analysis Tool*) is a system composed of a self-buildable device and an open-licensed application to measure several display latency metrics. The most interesting metric is *total system latency*: the time between an action happening in the physical world, like a mouse being clicked, and the result being displayed on the screen, such as a muzzle flash from a weapon in a videogame. There is currently no similar device on the market, and this type of measurement is traditionally done manually using a modified mouse and a high speed camera, but OpenLDAT can measure it automatically using a built-in test, or interactively, allowing testing of virtually any game or application, potentially on a separate machine. In addition to system latency, OpenLDAT can also measure more traditional metrics, such as *pixel response times*.

1 | INTRODUCTION

One of the problems that afflict gaming enthusiasts, especially competitive players, is *total system latency*, that is, the delay between an action happening in the physical world, like pressing a mouse button, and the result being visible on the screen.

This is not a new problem, and it has existed since the dawn of real-time computer graphics, but many things have changed over the years: On one side, the introduction of technologies such as high refresh rate displays, VESA Adaptive Sync and driver optimizations for latency-critical applications improved the situation, but on the other side, the tremendous increase in the complexity of graphic pipelines in videogames and the introduction of techniques like *temporal antialiasing*, *checkerboard rendering*, *triple buffering*, mouse smoothing, and desktop compositors made the situation worse to the point that some consider 60 FPS the minimum threshold for playability because of latency.

Many different factors are involved in *total system latency*, from the mouse microcontroller to the single screen pixel, but the main contributors to latency are usually the application, the speed of the hardware, and display used.

OpenLDAT has the following goals:

- Developing a device to measure *total system latency* both automatically and interactively, in the most accurate way possible and allowing comparison between different systems and scenarios
- Allowing the device to be used on a wide range of displays, even when strong interference is present from something like a pulse width modulation (PWM) backlight
- Using the device to provide additional metrics that can be measured with the same sensor, such as pixel response times
- Making the device easy to build, using off-the-shelf, low-cost parts, with a software that doesn't require calibration

- Distributing the software and the device schematics under a free license, to allow users to use, study, modify, and improve the project

1.1 | State of the art

Instruments for measuring the quality of a display's colors such as colorimeters and spectrophotometers have been around for a long time and are readily available starting at around 100$ for the cheapest models, but measuring latencies has almost always been done manually.[1]

Before OpenLDAT, *total system latency* was measured by using a high speed camera and a mouse that has been modified to turn on an LED when the left button is pressed[*]: When the button is pressed, the LED turns on, and after a certain time, the image on the screen will change (for instance, a muzzle shot will be visible if we're using a videogame); the captured footage from multiple runs is then analyzed by a human being to determine the delay between the LED turning on and the image changing on the screen, and that is the *total system latency*. This approach has three main disadvantages: The manual analysis is very time consuming, the temporal resolution of the camera is limited (usually to 1–2 ms unless an expensive camera is used), and since a third party application is involved, it may be difficult to replicate the results.

Around September 2020, Nvidia sent a prototype of a device called Nvidia LDAT,[1, 2] short for *Latency Display Analysis Tool*, to reviewers in the technical press. The Nvidia device was not commercialized,[3] but it was the inspiration for OpenLDAT. While not being a clone of Nvidia's tool, OpenLDAT is a free and open source project that achieves a similar goal: the measurement of latency metrics. It is also worth mentioning that Nvidia did not make their LDAT device available to the general public, they only sent prototypes to a few select members of the tech press, while OpenLDAT is free and open source, and users can even build their own OpenLDAT device if they want to. A comparison table between the two products is provided in Section 4.

The Nvidia LDAT device consists in a small plastic box with an RGB status LED in the front to show device status and clicks, a light sensor on the back, a micro-USB port to connect it to a computer running the proprietary Nvidia LDAT software, a connector for a purposedly modified mouse, a cord to easily attach it to a display, and an audio jack that can be optionally connected to measure audio latency. Nothing is known about the hardware inside the device, but by looking at it, the light sensor appears to be a large photodiode or possibly a charge coupled device (CCD, a type of camera sensor). No teardowns of the device have been performed, so the type of microcontroller and the other electronics inside are unknown.

The Nvidia LDAT software implements a test to measure what Nvidia calls *click-to-photon response*, the time between a mouse button being pressed and a brightness change on the screen. Clicks can be generated by the device itself or they can come from the modified mouse connected at the front of the device. In either case, the test is not fully automatic, as it requires an application, typically a game, to respond to clicks with a flash that is detected by the application.

The software can also measure audio latency in the same way, using the audio jack instead of the light sensor. This feature is absent in the OpenLDAT project as it was deemed unnecessary.

During the development of OpenLDAT, a similar project called DispLagBox[4] was also being developed. Unlike OpenLDAT, DispLagBox is a self-contained device built around a Raspberry Pi and a light sensor: the display being tested is plugged into the Raspberry HDMI port and the software takes care of the rest. Because of this approach, DispLagBox is more focused towards display testing than OpenLDAT since it may compensate for the constant known latencies introduced by the Raspberry Pi, but it has some disadvantages such as being unable to test between different PCs and OSes, being unable to test video games, being significantly more expensive than OpenLDAT and the fact that the Raspberry Pi doesn't support HDMI 2.0 features such as 4 K HDR 144 Hz variable refresh rate displays.

2 | DEVICE AND APPLICATION

OpenLDAT is composed of two main parts: a physical device that can measure light very quickly and an application that uses the device to run a set of tests.

The OpenLDAT device has essentially four functions:

- Sampling a light sensor quickly and regularly
- Generating clicks (automatically or externally depending on the test), pretending to be a mouse to the host PC
- Blinking an LED when clicks are generated, for manual verification using a high speed camera
- Handling communication with the host PC, receiving commands, and sending sensor data and clicks from the virtual mouse

[*] https://youtu.be/m0gILReDQsY

The heart of the device is a Sparkfun Pro Micro 5V/16 MHz: an Arduino-compatible board based on the ATmega 32U4[5] microcontroller, with a micro-USB connector. This microcontroller has a 10-bit analog-to-digital converter (ADC), and thanks to its programmable USB controller, it can be recognized as a mouse (USB HID) as well as a serial device (USB CDC Serial); the first interface is used to send clicks (automatic or manually generated), and the second is used by the application to control the device and receive data samples from it.

The light sensor used is the Everlight ALS-PT19,[6] a phototransistor with fairly low response times (~ 0.1 ms), good linearity, and low cost. Since this is a very small surface-mounted component, this first iteration of the device uses the Adafruit ALS-PT19 breakout board to easily mount it on a printed circuit board (PCB). Spectral response is not particularly relevant for this task, but the sensor's datasheet claims a response similar to that of the human eye plus ultraviolet light.

By manipulating the ADC registers on the microcontroller unit (MCU), it is possible to significantly speed up sampling as long as the input has a low enough impedance; by combining this with buffering techniques, the firmware on the device is able to reach sample rates up to ~ 30 kHz with 10 bits of resolution without a loss of quality.[5] Depending on what the test needs, the application can choose which features to enable on the device: slow or fast sampling, internal or external click generation, sampling of light only or the clicks too, and so on.

The MCU and the sensor are mounted on a custom single-layer PCB that also hosts and LED that shows clicks for validation with a high speed camera, a connector for an external button or a modified mouse, and some resistors that are used to control the gain level of the sensor. Four levels of gain are implemented, as seen in Table 1, and are available to the application.

Figure 1 (left) shows a top view of the OpenLDAT device internals, with the MCU, the LED and the external button connector on top, and the sensor at the bottom.

The prototypes have been mounted inside a round 3D printed case shown in Figure 1 (right). On the bottom of the case, a felt pad prevents accidental scratching of the display being tested, and a thin microscope glass protects the sensor from dust and accidental touches.

TABLE 1 Gain levels

Level	Gain	Brightness (nits)
0	1.000	300-700
1	1.258	250–600
2	2.101	60–300
3	13.883	0–80

2.1 | Application

The OpenLDAT application is the most complex part of this system: It uses the device to run the test and analyzes the collected data to extract various types of information.

The application is designed to be multiplatform and runs on Microsoft Windows, GNU/Linux, and (with some limitations) MacOS. It has been developed using Java SE and OpenGL and has a graphical interface as well as a built-in manual. The screenshot in Figure 2 shows the main screen of the application, with the list of tests on the left and instructions on the right. Some tests also have settings that can be configured by the user, such as duration.

During the test, the user is instructed on where to place the sensor device. This is a massive advantage that OpenLDAT has over the traditional high speed camera method: Since it is almost entirely automated, the user only needs to position the device on the display, and the OpenLDAT software will take care of the rest, ensuring and more reproducible results and making what used to be a tedious manual task a 30-s job. Care should still be taken to eliminate external sources of interference, such as strong lights or electromagnetic fields, and all image enhancement features of the display should also be disabled while the test is running.

At the end of the test, the results are shown in the interface, with the option of exporting them in text form for external analysis. The screenshot in Figure 3 shows the table of *pixel response times* of one of the tested monitors.

OpenLDAT implements tests as follows:

Total system latency (automated test): This test uses the device to automatically generate clicks at regular intervals that are received by the application itself, which generates a white flash in response; the software analyzes the capture to determine the delay between the click being sent and the flash appearing on the screen. The test can simulate various load scenarios that can occur in a videogame. During the test, the status LED blinks to show the clicks being generated, allowing for manual verification using a high speed camera. The algorithm works by measuring the black level and by finding peaks that are significantly above it. Each peak is associated to the click that caused it (there will be more than one peak per flash if PWM is present), and the delay between click and light increase is measured. This method is not yet compliant to the one proposed by SID in International Committee for Display Metrology,[7] but the SID standard

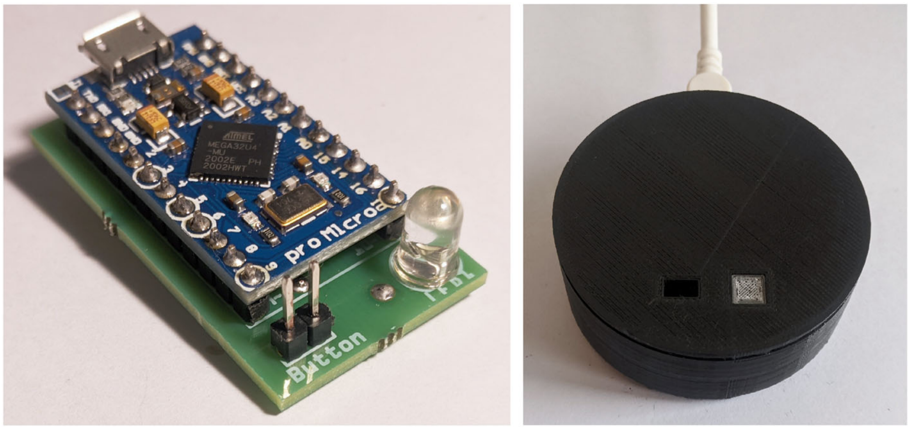
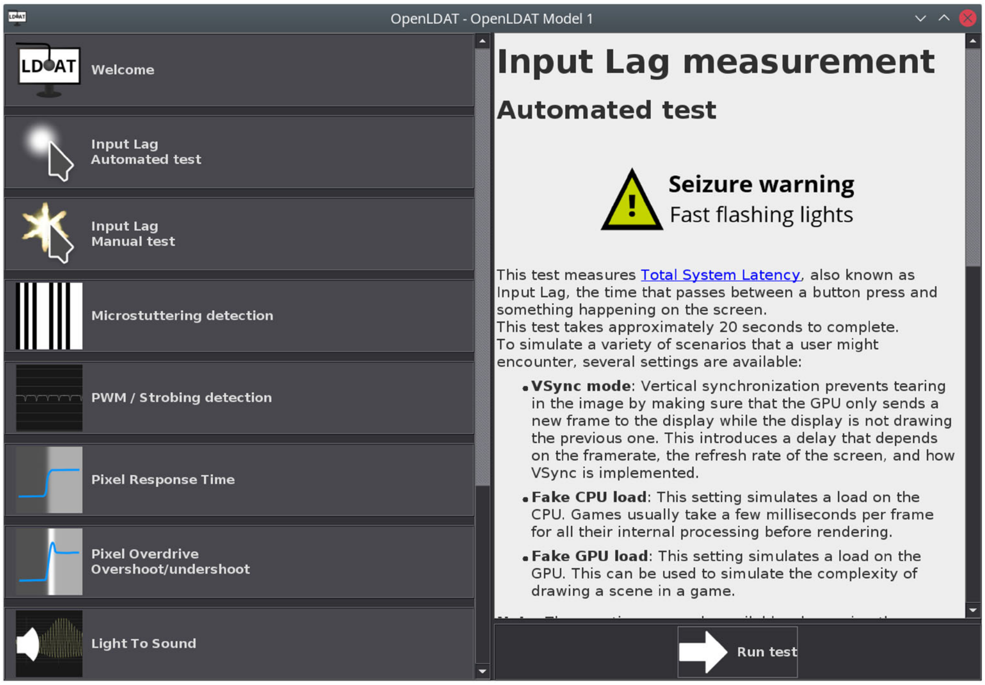


FIGURE 1 The OpenLDAT device



Input Lag measurement

Automated test

Seizure warning
Fast flashing lights

This test measures Total System Latency, also known as Input Lag, the time that passes between a button press and something happening on the screen.
This test takes approximately 20 seconds to complete.
To simulate a variety of scenarios that a user might encounter, several settings are available:

- **VSync mode**: Vertical synchronization prevents tearing in the image by making sure that the GPU only sends a new frame to the display while the display is not drawing the previous one. This introduces a delay that depends on the framerate, the refresh rate of the screen, and how VSync is implemented.
- **Fake CPU load**: This setting simulates a load on the CPU. Games usually take a few milliseconds per frame for all their internal processing before rendering.
- **Fake GPU load**: This setting simulates a load on the GPU. This can be used to simulate the complexity of drawing a scene in a game.

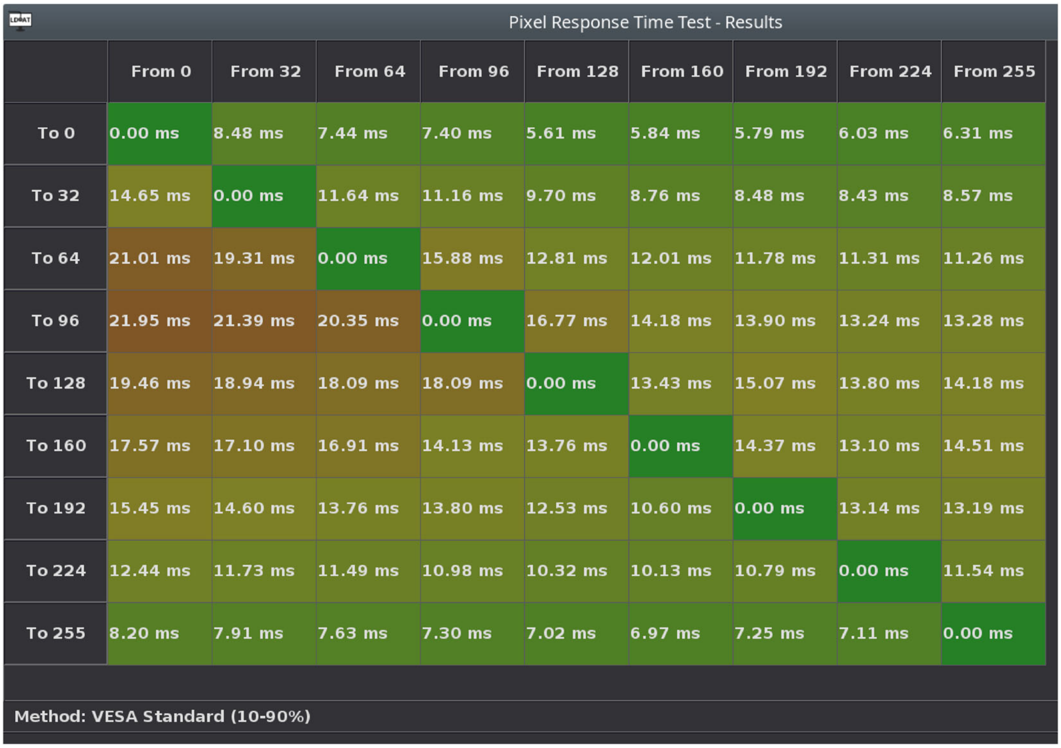
FIGURE 2 Main screen of the OpenLDAT application

will be implemented in the next version of OpenLDAT companion software. The white-flash test is designed to be as simple as possible to render, so that the speed of the GPU doesn't significantly affect the accuracy of the result, with even a low end card being able to reach and exceed 1000 FPS. That being said, tests should still at least be performed on the same OS to have comparable results, since there is no way to compensate for differences between them.

Total system latency (manual test): This test allows measurement of *total system latency* using virtually any application (typically a game), potentially running on a completely separate machine, using a modified mouse that can be connected to both the OpenLDAT device and the target machine, or by using the device itself to generate periodic clicks on the host PC like in the automated test. The user interface allows users to tweak settings and displays the results. The algorithm is essentially identical to the automated test, except that flashes are generated by an external application and the user can set the peak detection threshold.

PWM and noise detection: This test detects the presence of a PWM backlight, as well as other types of noise, and displays the dominant frequency (if present). This test works by displaying a shade of gray, sampling it for a few seconds, and running it through an FFT to detect peaks. If strong peaks are found, a PWM backlight is present, and its frequency is determined by the strongest peak. If no peaks are found but the signal is still noisy, then some other type of noise is present. All tests in the application must be able to handle “holes” in the captured signal caused by PWM/noise.

Pixel response times: This test measures the time that pixels take to transition between many shades of



Pixel Response Time Test - Results

	From 0	From 32	From 64	From 96	From 128	From 160	From 192	From 224	From 255
To 0	0.00 ms	8.48 ms	7.44 ms	7.40 ms	5.61 ms	5.84 ms	5.79 ms	6.03 ms	6.31 ms
To 32	14.65 ms	0.00 ms	11.64 ms	11.16 ms	9.70 ms	8.76 ms	8.48 ms	8.43 ms	8.57 ms
To 64	21.01 ms	19.31 ms	0.00 ms	15.88 ms	12.81 ms	12.01 ms	11.78 ms	11.31 ms	11.26 ms
To 96	21.95 ms	21.39 ms	20.35 ms	0.00 ms	16.77 ms	14.18 ms	13.90 ms	13.24 ms	13.28 ms
To 128	19.46 ms	18.94 ms	18.09 ms	18.09 ms	0.00 ms	13.43 ms	15.07 ms	13.80 ms	14.18 ms
To 160	17.57 ms	17.10 ms	16.91 ms	14.13 ms	13.76 ms	0.00 ms	14.37 ms	13.10 ms	14.51 ms
To 192	15.45 ms	14.60 ms	13.76 ms	13.80 ms	12.53 ms	10.60 ms	0.00 ms	13.14 ms	13.19 ms
To 224	12.44 ms	11.73 ms	11.49 ms	10.98 ms	10.32 ms	10.13 ms	10.79 ms	0.00 ms	11.54 ms
To 255	8.20 ms	7.91 ms	7.63 ms	7.30 ms	7.02 ms	6.97 ms	7.25 ms	7.11 ms	0.00 ms

Method: VESA Standard (10-90%)

FIGURE 3 Pixel response times of an AOC Q2770P

gray. OpenLDAT implements the VESA standard,[8] in other words it measures the time taken to complete the part between 10% and 90% of the transition. Future versions of the OpenLDAT software will implement other measurement standards as well, such as the ones proposed by SID in International Committee for Display Metrology.[7]

This algorithm works in two phases: first, for each shade of gray, it determines the best gain level to use in order to avoid saturation or loss of accuracy; second, for each couple of shades of gray, a transition is sampled, the beginning and end of the transition are determined, and the time difference is calculated. If PWM or noise are present, the gaps in the signal are interpolated; therefore, some degree of accuracy may be lost.

Pixel overdrive: This test measures the error that occurs during pixel transitions between many shades of gray. This algorithm is similar to the *pixel response time* test, but its focus is at the end of the transition. It finds peaks above (if brightness is increasing) or below (if brightness is decreasing) the expected level at the end of the transition and measures it as an absolute percentage (over the entire brightness range of the display) or as a relative percentage (over the brightness range of the transition). If PWM or noise are present, the gaps in the signal are interpolated. This test suffers greatly from the presence of PWM since the peak that the algorithm is trying to find may occur while the backlight is off; it is therefore not recommended for this type of displays.

Microstuttering detection: This test detects the loss or duplication of frames. This may happen for many reasons, from poorly written software to incorrect display settings, for instance, if the input signal has a non-native refresh rate (*overclock*) the display may not be able to process all frames in time. The algorithm works by displaying alternating black and white frames and measuring times between black to white transitions. If frames are dropped or duplicated, there will be a measurable deviations in the timing of these transitions and microstuttering is detected. If PWM or noise are present, a filter is applied to remove it or at least significantly reduce it.

Light to sound: This test allows the user to listen to the signal captured by the light sensor and to see the shape of the signal (similar to an oscilloscope) and detects the dominant frequency (if present). This can be useful for finding sources of interference, such as poorly filtered LED lamps.

3 | MEASUREMENTS

During development, over 20 displays of different types and eras were tested, to make sure that the system would produce accurate measurements under all circumstances.

Before running the tests, the device and the application were tested to ensure an accurate analysis and reproducible results. This was done by manually analyzing the captured signals as well as captures from an oscilloscope and comparing the manually calculated results with the application's output. Tests have shown a good degree of accuracy and consistency across multiple runs. For instance, in the automated *total system latency* test, variance was less than 5%, validated with a high speed camera. More data on this subject will be provided in the future if a proper study on OpenLDAT's accuracy can be made using more sophisticated equipment which was not available at the time of development.

3.1 | Input lag

Several tests have been performed on a variety of display, hardware, and software combinations to determine the impact that it would have on *total system latency*.

The chart in Figure 4 shows how much the display affects the *total system latency*. The data were collected using the automatic test, and all displays were tested with the same hardware and software configuration.

High refresh rate displays that are built for gaming dominate this chart, as one would hope, while TVs sit at the bottom. A more interesting pattern that emerged however is the fact that some displays (like the LG E2360) seem to be able to display the image while they're still receiving it from the computer, similarly to old CRT displays where the incoming signal directly controlled the electron beam without buffering or delays,[†][9, 10] while other displays (like the AOC Q2770P) store an entire frame in a buffer before displaying it, possibly to apply some kind of additional processing.

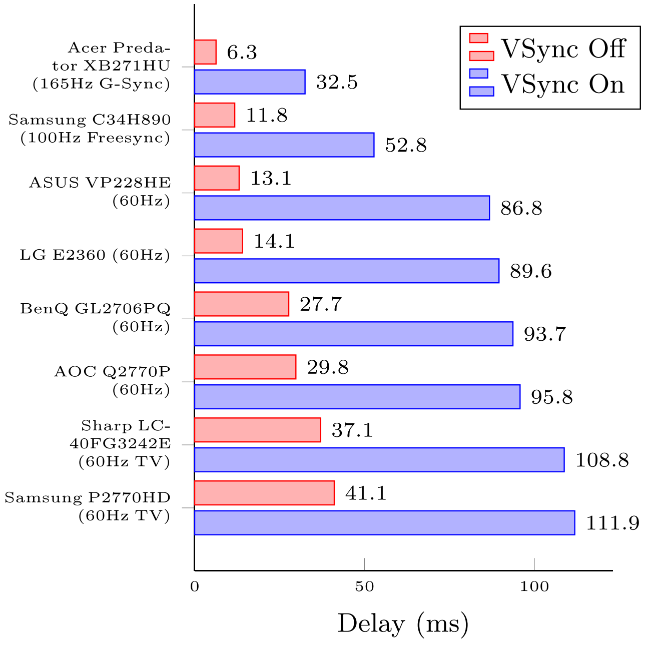


FIGURE 4 Display input lags

The chart in Figure 5 shows the impact that different operating systems, GPUs, and display drivers have on input latency. The data were collected using the automatic test using different hardware and software configurations on the same display (AOC Q2770P).

The Linux + Nvidia combination tops the chart, having the lowest latency both with and without VSync,[‡] at least for OpenGL. This came as a surprise since Nvidia drivers have a bad reputation in the Linux community.[§,¶,#] Another interesting result is the fact that Intel is near the bottom of the chart on both Windows and Linux: This is caused by iGPU being significantly slower than even a low end discrete GPU and therefore running the test at a lower framarate.

Using the interactive input lag test, several applications were also tested to determine the impact that games themselves have on *total system latency*. The test was performed using a single hardware and software configuration (Windows + Nvidia), testing several applications on the same display (AOC Q2770P). Results are shown in Figure 6.

In this scenario, latency is mostly affected by the framerate of the application and how the engine works internally. Please note that OpenLDAT only measures total system latency here, and it has no way to know how much time the game spent doing processing on either the CPU or the GPU; therefore, this metric is more useful as a "gaming" metric rather than a dislpay metric unless those processing times are negligible. All tests in this section were run on the same hardware so that they are comparable.

The lowest latency here was shown by Mass Effect Legendary Edition,[‖] which was unexpected since it is mostly a story based game where latency is not really an issue. The 2007 version of Crysis[**] also showed very good

[†] https://retrocomputing.stackexchange.com/a/17298

[‡] Vertical Synchronization, the GPU buffers are only swapped during the VBlank interval to prevent tearing and limit the framerate to the display refresh rate.

[§] https://www.pcworld.com/article/2911459/why-nvidia-graphics-cards-are-the-worst-for-open-source-but-the-best-, for-linux-gaming.html

[¶] https://wiki.archlinux.org/title/NVIDIA/Troubleshooting

[#] https://www.phoronix.com/forums/forum/linux-graphics-x-org-drivers/nvidia-linux/1149405-linux-with-nvidia- gpu-is-it-really-that-bad

[‖] https://store.steampowered.com/app/1328670/Mass_Effect_Legendary_Edition/

[**] https://www.gog.com/game/crysis

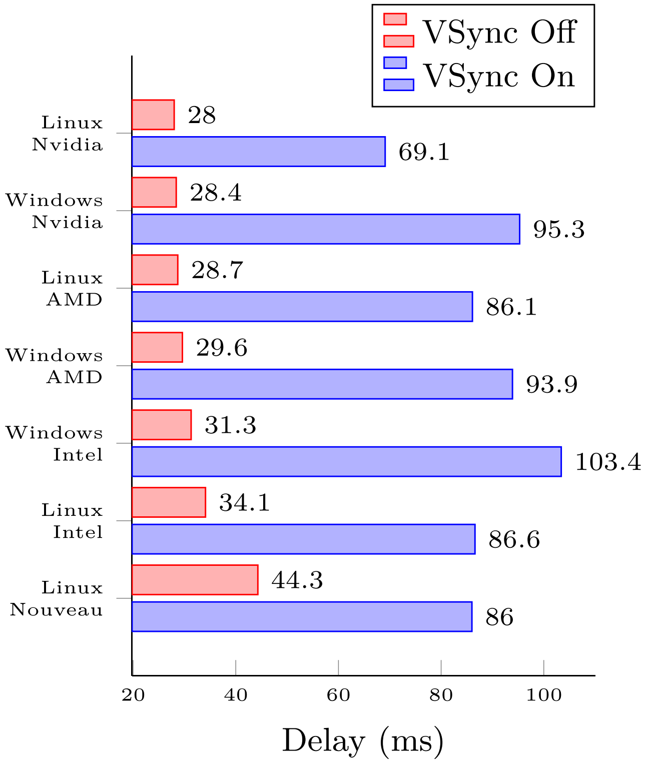


FIGURE 5 Input lags of hardware/software combinations

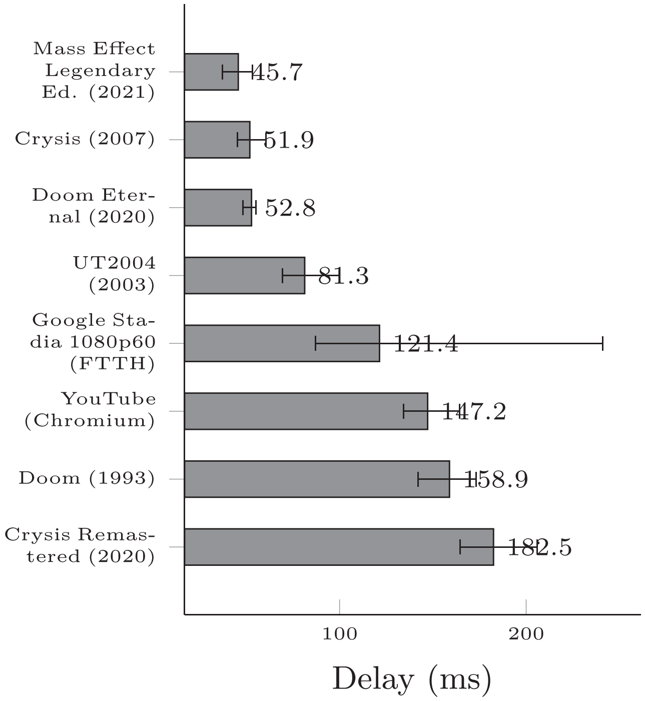


FIGURE 6 Applications input lags

results thanks to its excellent engine (when it came out, most people could not run this game at more than 20–30 FPS, so the game is optimized for these scenarios).

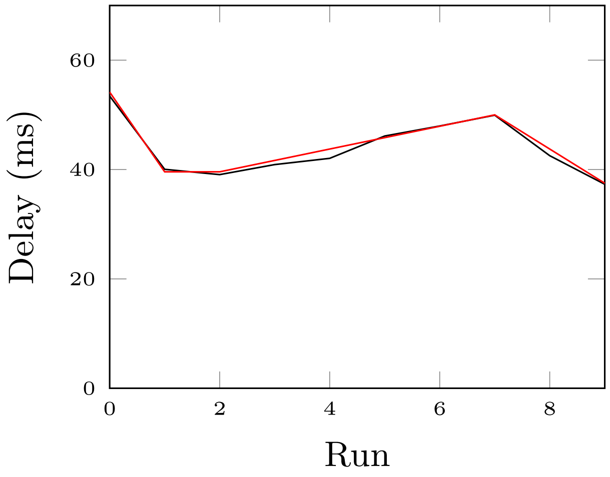


FIGURE 7 Input lag validation

Unreal Tournament 2004[††] showed unexpectedly high latency despite running at over 700 FPS, but it was quickly discovered that the Windows 10 compositor was the cause of this problem as it was forcing the game to run through it instead of using exclusive fullscreen. Google Stadia[‡‡] also shows a remarkable improvement compared to when it was launched, with a borderline playable 120 ms of latency, down from 180 to 300 ms. In the last place, and above what many people would consider playable, there is the 2020 remaster of Crysis[§§], with a very high latency despite running at about 45 FPS. Crysis developers chose to force the game go through the Windows compositor and implemented a long swapchain to favor smoothness over latency; therefore, latency is high unless the game is running at very high framerates.

Finally, the chart in Figure 7 shows the difference between the *total system latency* values measured by OpenLDAT (black line) and those measured manually using the traditional high speed camera approach (red line). The two lines are very close to each other, showing that OpenLDAT is measuring latency correctly.

3.2 | PWM and other types of noise

This test is designed to determine the presence of PWM backlights, and the PWM frequency if present. The test can also detect other types of noise such as black frame insertion and dithering.

Overall, PWM backlight was only found on low-end displays, but the shape of the signal was wildly different across different models. Figure 8 shows an example of

[††] https://www.gog.com/game/unreal_tournament_2004_ece
[‡‡] https://stadia.google.com/
[§§] https://store.steampowered.com/app/1715130/Crysis_Remastered/

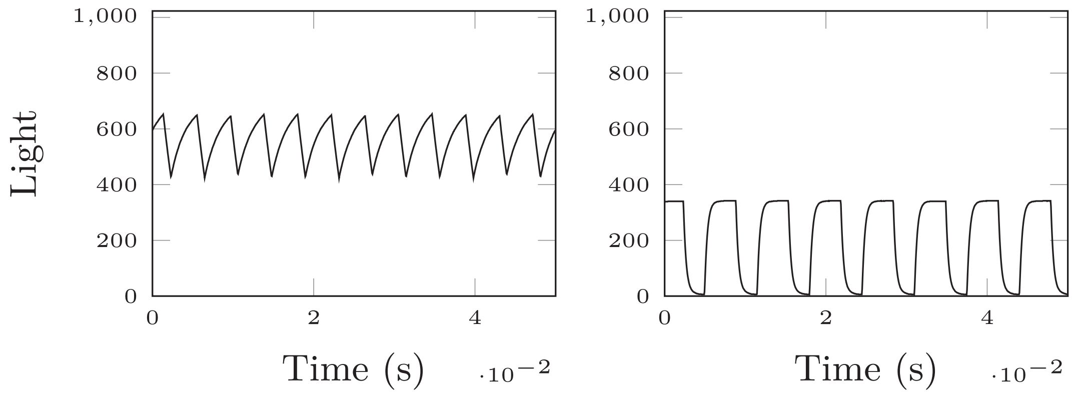


FIGURE 8 Types of PWM backlight

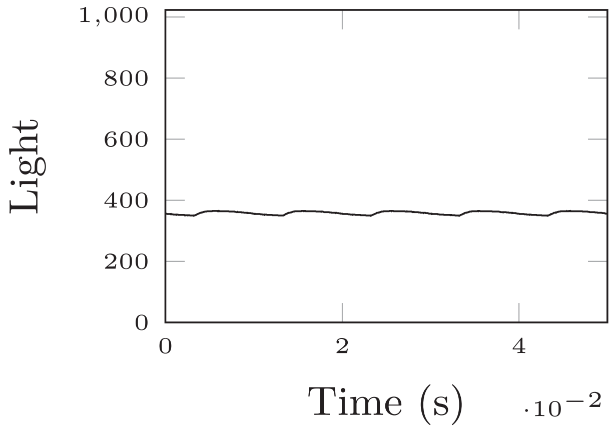


FIGURE 9 Visible pixel refresh cycle

this behavior: The display on the left (LG E2360) looks more like a charge/discharge cycle of a capacitor, while the one on the right (Philips 32PFS4132) looks like more typical PWM.

Another interesting behavior is shown in Figure 9: On some displays (like the Samsung C34H890), the pixel refresh cycle is visible. This is much weaker than the flickering caused by PWM and doesn't cause loss of accuracy in the tests.

3.3 | Pixel response times and overdrive

The chart in Figure 10 shows the pixel respose times of a few of the tested displays with and without overdrive. As a reminder, overdrive is the most common name of the technique of emphasizing transitions by overshooting them slightly to achieve faster transition times. The pixel respose time test generates a table of transition times in which each cell contains the time required to perform the transition between the two corresponding shades of gray (an example has been shown previously in Figure 3). Using this data, the chart in Figure 10 was created by calculating for each display the geometric average of its transition times (the bars) and the range in which all the transitions lie (the horizontal lines).

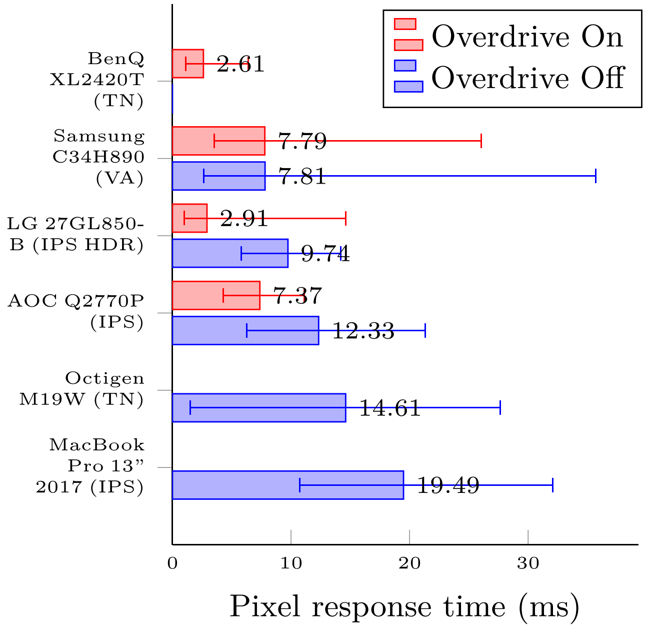


FIGURE 10 Pixel response times

When overdrive was available, it was set to the lowest level required to have at least one transition time match the manufacturer's declared response time (in all cases, this was very close or equal to the maximum value, with a significant loss of image quality).

As expected, gaming displays score higher in this chart, especially TNs. The LG 27GL850-B is very close with overdrive enabled despite having an IPS panel, but this comes the cost of a tremendous loss of image quality as shown in the next test.

Figure 11 shows the transition error committed by the same displays with and without overdrive.

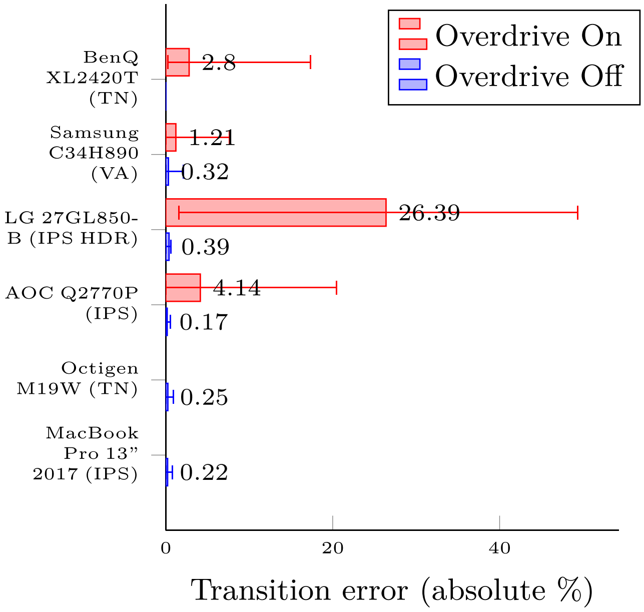


FIGURE 11 Transition errors

With overdrive disabled, all displays have virtually no transition error, but with overdrive, significant errors start to appear, with the worst offender being the LG 27GL850-B. This error creates a visual artifact usually called inverse ghosting, where moving objects on the screen leave a trail of the opposite color. A transition error higher than 1%–2% is noticeable to the user.

The overdrive results may seem low,

since there is no standard on how to measure the transition error

it is possible that OpenLDAT measures it differently from what other researchers define as "overdrive." The authors solicit feedback from other researchers who built OpenLDAT and reproduced these results.

3.4 | Microstuttering

Microstuttering is an irregularity in frame times causing dropped or duplicate frames. This is usually caused by software (for instance, a game loading from disk while playing) or by incorrect display settings (for instance, using the wrong refresh rate).

Out of all tested displays, none has shown microstuttering when running at native refresh rate or when using VESA Adaptive Sync. Some displays like the AOC Q2770P show microstuttering when running at non-native refresh rates; this is presumably due to the internal processing performed by the display, which runs the panel at a constant 60 Hz regardless of the frequency of the input signal, therefore causing dropped or duplicate frames.

TABLE 2 Feature comparison between Nvidia LDAT and OpenLDAT

	Nvidia LDAT	OpenLDAT
Sensor	Photodiode*	ALS-PT19
Gain levels	1	4
Resolution	10 bit*	10 bit
Sample rate	∼ 1000 Hz*	Up to ∼ 30 kHz
LED for validation	Yes	Yes
Click generation	Yes	Yes
External button	Yes	Yes
Audio input	Yes	No
Test: automatic input lag	No	Yes
Test: manual input lag	Yes	Yes
Test: PWM	No	Yes
Test: microstuttering	No	Yes
Test: response times	No	Yes
Test: overdrive	No	Yes
Light to sound	No	Yes
Platforms	Windows	Windows, Linux, MacOS
License	Proprietary	Free
Cost	N/A	∼ 15

4 | CONCLUSIONS

The project has shown satisfactory performance, reaching its goals and providing some interesting results on the way that may be subject of further study. With OpenLDAT, the tedious task of using a modified mouse, a high speed camera, and a game to capture and manually analyze footage to measure input lag is now an automated, standardized, and easy task that virtually anyone can perform with an inexpensive device that can be bought or self-built.

During development and testing, areas of improving emerged, as well as ideas for future iterations of the OpenLDAT project. Since the project is completely *free*[¶¶] and anyone can easily build the device, reproduce our results and improve OpenLDAT, it is likely that some contributions will come from the community.

Possible future developments include improvements to the device to allow the implementation of more tests (especially tests specific to HDR displays) and

[¶¶] https://www.gnu.org/philosophy/free-sw.html improvements to the existing ones, as well as the addition of a colorimeter to also run traditional color accuracy tests.

Table 2 shows a comparison between the features of Nvidia LDAT and OpenLDAT. Since Nvidia didn't provide official specifications for their device and did not make their device commercially available, some of the data provided may be incorrect as it is the result of observation done by the author of this paper, looking at videos and articles about their device. Uncertain values have been marked with "*."

Hardware diagrams, documentation, and all the software of the OpenLDAT project can be found here:

- https://openldat.fdossena.com (website)
- https://github.com/adolfintel/OpenLDAT (repo)

The OpenLDAT project (hardware, firmware, and software) is distributed under the GNU GPL v3 license.[##]

ACKNOWLEDGEMENTS

Open Access Funding provided by Universita degli Studi di Milano within the CRUI-CARE Agreement. [Correction added on May 20, 2022, after first online publication: CRUI funding statement has been added.]

ORCID

Andrea Trentini https://orcid.org/0000-0002-8629-3056

REFERENCES

1. Steve Burke (Gamers Nexus). Automated system latency test methodology (nvidia latency analyzer validation). https://youtu.be/0SZ7ZCac38A; 2020.
2. igor'sLAB. Nvidia ldat latency display analysis tool introduced and tested. https://www.igorslab.de/en/nvidia-ldat-latency-display-analysis-tool-presented-and-tested/; 2020.
3. NVIDIA. Nvidia reviewer toolkit for graphics performance. https://www.nvidia.com/en-us/geforce/news/nvidia-reviewer-toolkit/; 2020.
4. Stadler P, Schmid A, Wimmer R. Displagbox: simple and replicable high-precision measurements of display latency. *Proceedings of the conference on mensch und computer*, MuC '20. Association for Computing Machinery; New York, NY, USA; 2020. p. 105–108. https://doi.org/10.1145/3404983.3410015
5. Microchip. Atmega 32u4 datasheet. https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7766-8-bit-AVR-ATmega16U4-32U4 Datasheet.pdf; 2016.
6. Everlight. Everlight als-pt19 datasheet. https://cdn.sparkfun.com/datasheets/Components/General%20IC/ALS-PT%19-315C-L177-TR_datasheet.pdf; 2012.
7. INTERNATIONAL COMMITTEE FOR DISPLAY METROLOGY. Information display measurements standard; 2021.
8. Video Electronics Standards Association FPDM Task Group. Flat panel display measurements standards v2.0; 2005.
9. Ferraro RF. Programmer's guide to the ega, vga, and super vga cards: Addison-Wesley; 1994.
10. Parekh R. Principles of multimedia: Tata McGraw-Hill; 2006.

AUTHOR BIOGRAPHIES

Federico Dossena Computer Science student at Universitá degli Studi di Milano with a strong interest in Free and Open Source software since 2009, author and developer of several popular projects.

Andrea Trentini is an assistant professor at the Universiá degli Studi di Milano (Dipartimento di Informatica—http://di.unimi.it) where he teaches "Programming," "Embedded Systems," and "Digital Citizenship & Technocivism." He is a strong supporter of the Free Software Movement.

How to cite this article: Dossena F, Trentini A. OpenLDAT—A system for the measurement of display latency metrics. J Soc Inf Display. 2022; 1–10. https://doi.org/10.1002/jsid.1104

[##] https://www.gnu.org/licenses/gpl-3.0.en.html