WPF and .NET line charts performance comparison

September, 2021.

LightningChart® 10.1.1 vs. Competitor (SciChart v6.3.0.13629)

A pick from results

LightningChart .NET is **up to 3000 times faster** than any other charting control in the market.

Foreword

There are only a few truly high-performance charting add-ons for .NET. Based on the manufacturers' claims as well as performance results in official websites, LightningChart and *Competitor* are the top 2 charting libraries.

There is plenty of empty claims that dozens of charting packages are 'high-performance' which is gross misleading of the users. Only real hardware accelerated charts (GPU-accelerated charts) can provide the best rendering performance. So, this test is purely using hardware accelerated charts.

In 2015 *Competitor* made a performance comparison against LightningChart, with false information, which LightningChart team fixed and rectified with fixed performance comparison: https://www.arction.com/lightningchart-and-scichart-benchmark/

It was clear that LightningChart was faster. Today, *Competitor* is still claiming they have the "World's fastest WPF charts", which according to running the Competitor's demos, and suspected and verified by the customers, doesn't seem true at all. So, to clarify this confusion for the users and customers, we were practically forced to make a new comparison to show users a real proof of which chart is *The Fastest In The World*.

This test has been made as a stress test most demanding line charts applications, which are progressing real-time charts, such as medical ECG, EEG, ExG, telemetry, vibration monitoring and instrumentation applications.

LightningChart API is available for **WPF**, **Windows Forms** and **UWP** (Universal Windows Platform). Competitor API is only available for WPF. Therefore, this test is for **WPF charts only**, but LightningChart API performance in WinForms and UWP are nearly identical, if not even better.

For other technologies, charts performance comparisons, performance tests and other proof of performance, please visit LightningChart official website.

LightningChart® SampleDataBlockSeries vs Competitor FastLineRenderableSeries

In LightningChart® .NET v.10.1.1 a new, super-fast, line series was introduced, called **SampleDataBlockSeries**. The data is stored as memory blocks, which disposes old data and appends new data, easier on memory and CPU.

This new series is a replacement for SampleDataSeries, which had a linear memory array. The new series type was especially made for fixed-interval data monitoring, used typically in waveform visualization, e.g., medical monitoring (ECG/EKG, EEG, EMG, ExG), vibration monitoring, telemetric and data logger systems, and the audio engineering industry.

SampleDataBlockSeries rendering scientific data visualization algorithms were designed to take better benefit from GPU computation power, freeing up more CPU resources for other processes and tasks.

According to the Competitor's documentation and examples, the **FastLineRenderableSeries**, with Visual Xccelerator engine with Impossible mode enabled is the most performant way to render progressing line charts with the resampling mode enabled.

Therefore, Visual Xccelerator rendering engine was enabled (which should use DirectX) + *Impossible Mode ON*, and all tips (for optimizing application) from Competitor's developers have been followed. However, we did one exception regarding *ResamplingMode*.

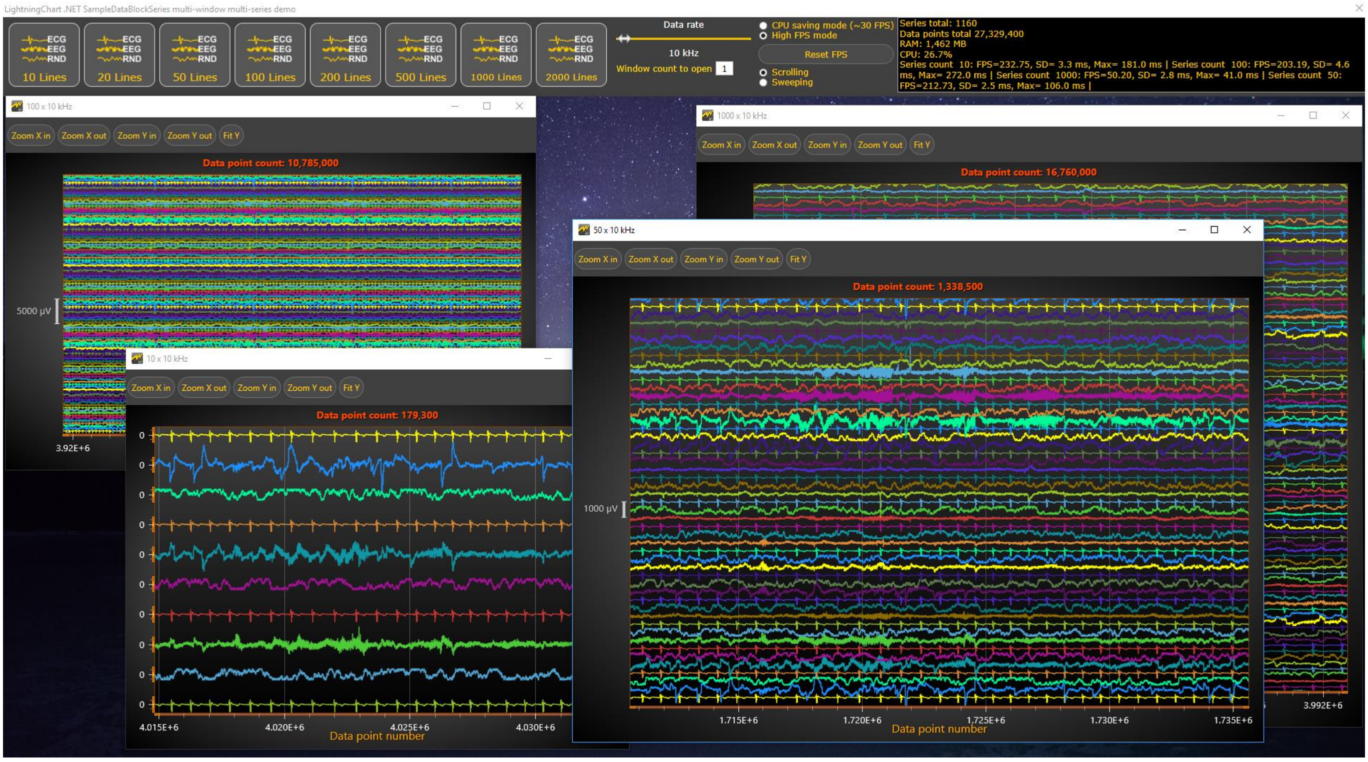
We run all the set of tests with set to *ResamplingMode* 'None' or 'Auto'. The rendering result (*see chart image*) of *ResamplingMode.None* should be comparable to LightningChart rendering (because LightningChart does not do any resampling or down sampling internally but has way more intelligent algorithms to **correctly** optimize the rendering).

What would happen when using resampling?

When using r*esampling* there is a risk of visual defects, which were clearly visible in the tests we executed.

Demanding Test Application

The following test compares these two high-performance line series in practical tests. This scientific data visualization test was performed with a standalone **SampleDataBlockSeries** demo application made by the LightningChart team. *Copyright Arction Ltd, 2021, All rights reserved*.



Data rate 10 kHz

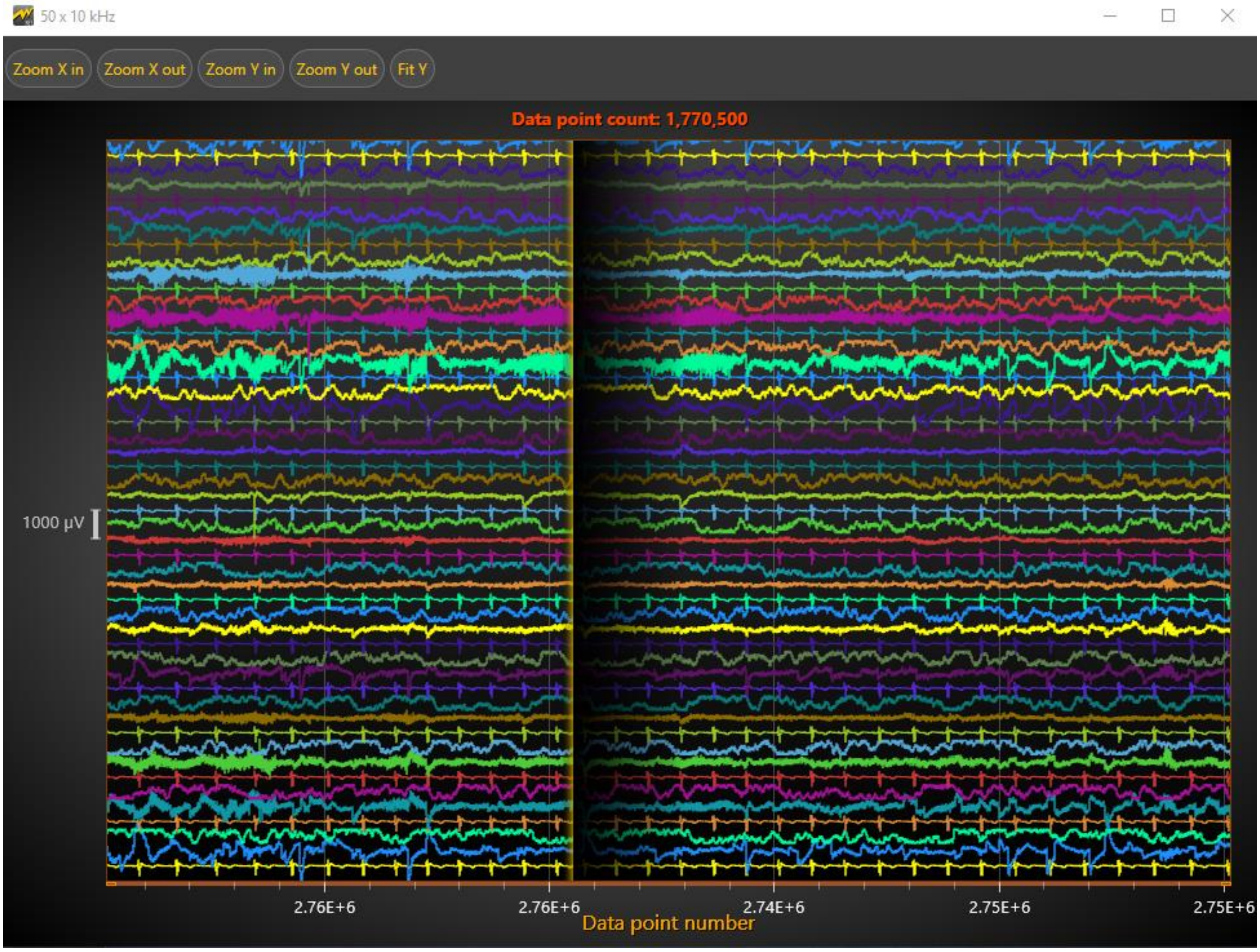
Window count to open 1

```
Series total: 1160
Data points total 27,329,400
RAM: 1,462 MB
CPU: 26.7%
Series count  10: FPS=232.75, SD= 3.3 ms, Max= 181.0 ms | Series count  100: FPS=203.19, SD= 4.6
ms, Max= 272.0 ms | Series count  1000: FPS=50.20, SD= 2.8 ms, Max= 41.0 ms | Series count  50:
FPS=212.73, SD= 2.5 ms, Max= 106.0 ms |
```

The application consists of a header bar and the test application allows opening dozens of chart windows at once. The data rate is adjustable, starting from 1000 Hz (1000 data points/sec) for each series. Each window can be opened with a preferred series count between 10 - 2000.

The data read from a .CSV file consisted of real ECG, EEG, and then random data every 3rd series.

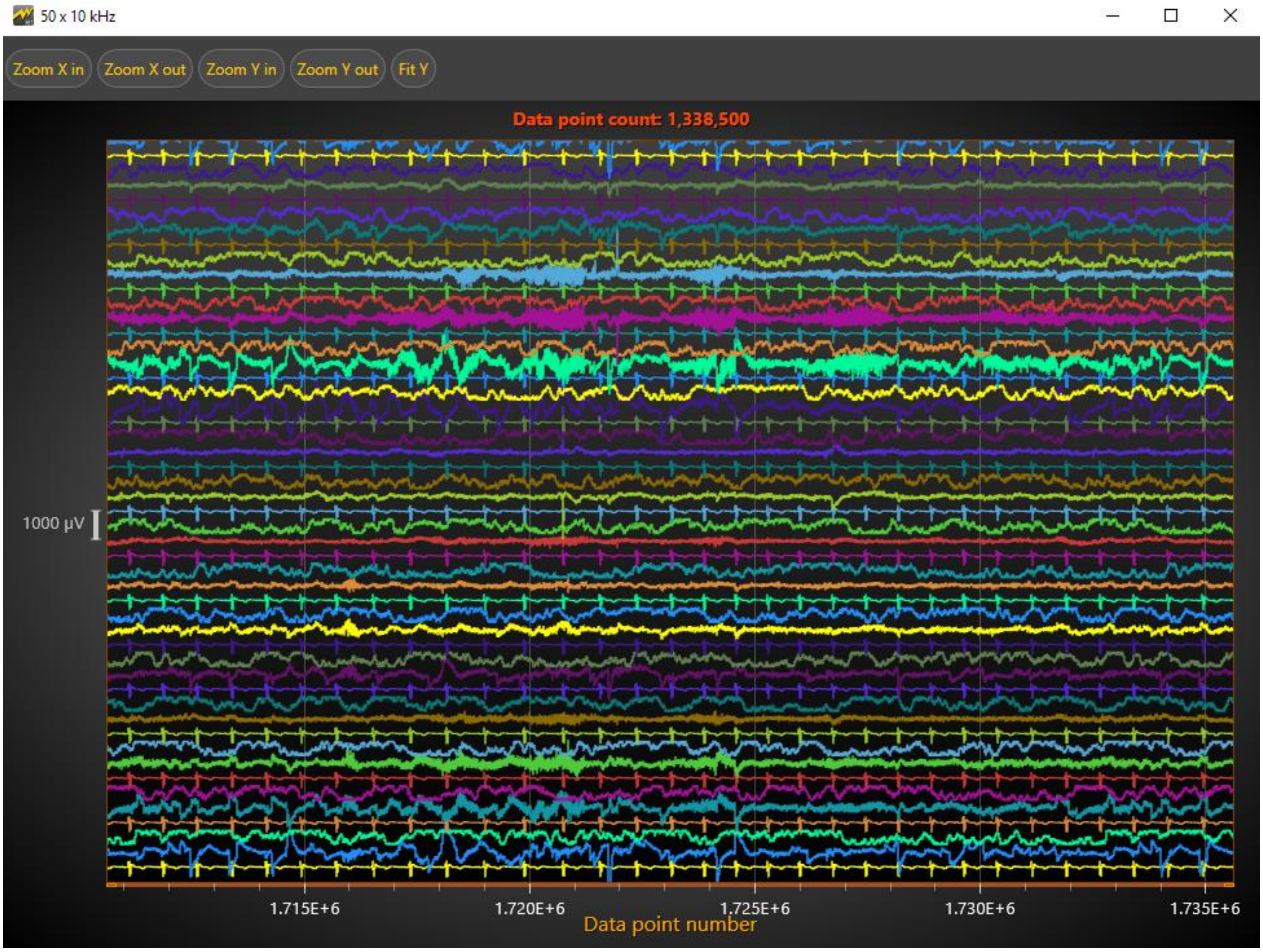
The chart windows can be set to the ‘Sweeping’ real-time mode:



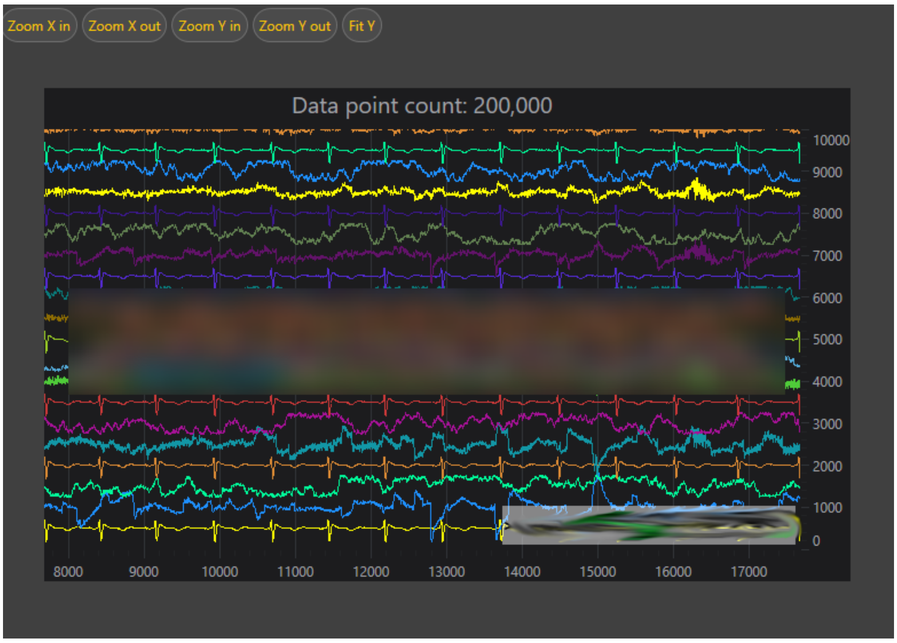
Sweeping mode window

In the following image, the main focus is only based on the **scrolling** mode, as the Competitor does not have a Sweeping mode built-in feature.

The X-axis length was set to hold 10 seconds of data. The line width was set to 1.5 pixels.



Scrolling mode window



Competitor in Scrolling view.

Device Hardware & Operating System Specifications

Mid-Level Desktop PC

AMD Ryzen 5 2600 Six-Core Processor (12 logical CPUs), ~3.8GHz
Memory: 64 GB RAM
GPU: NVIDIA GeForce GTX 1070, 8 GB
Windows 10 64-bit OS

and

Low-end, budget laptop

Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz (logical 8 CPUs), ~1.2GHz
Memory: 20 GB RAM
GPU: NVIDIA GeForce MX110, 2 GB
Windows 10 64-bit OS

Performance Test Legend

The test was performed in two modes:

- CPU-saving mode
- High-FPS mode

Parameters measured or analyzed

- **FPS:** refers to the chart refresh rate. A higher value of frames/sec is better. Smooth scrolling requires stable 30 FPS or more.

- **RAM Consumption:** is measured in MB. A lower value is better.

- **CPU Load:** Measured in %. A lower value is better. Also observe the FPS rate produced with this CPU load. Discussed further in **CPU load / refresh rate** parameter.

- **Standard Deviation (SD)**: Standard deviation of Refresh intervals measured in milliseconds. A lower value is better and it indicates a smoother data-scrolling experience, with less twitching or stuttering.

- **Relative standard deviation (RSD) / coefficient of variation (CV):** A standardized measure of dispersion which is defined as the ratio of the standard deviation to the mean. A lower value is better and indicates a smoother data-scrolling experience with less twitching or stuttering.

- **CPU load / refresh rate:** Demonstrates efficiency, as the CPU load is compared to the number of frames rendered. A lower value is better.

- **Maximum delay in Refreshes.** The delay between refreshes was measured in milliseconds. A lower value is better.

1. Desktop with High FPS mode

In this test the charts were asked to render as fast as possible (*High FPS mode*). Even exceeding the refresh rate of a typical screen (60 Hz).

		LightningChart v10.1.1.4001 -> High FPS mode in all tests						Competitor Xccelerator Engine + Impossible Mode ON + Resample Mode Auto						Competitor Xccelerator Engine + Impossible Mode ON + Resample Mode NONE					
Test	**Total points count**	FPS	CPU Load (%)	RAM (MB)	SD (ms)	Max (ms)	RSD/CV	FPS	CPU Load (%)	RAM (MB)	SD (ms)	Max (ms)	RSD/CV	FPS	CPU Load (%)	RAM (MB)	SD (ms)	Max (ms)	RSD/CV
1 Window x 10 series, 1 kHz Data Rate	100 k	453.48	10.62	267.00	0.67	17.67	0.30	60.02	2.34	271.00	0.57	18.33	0.03	60.05	3.51	336.67	0.50	17.00	0.03
10 Window x 10 series, 1 kHz Data Rate	1 M	134.12	44.44	823.33	5.50	24.67	0.74	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 100 series, 1 kHz Data Rate	1 M	386.10	11.63	280.33	0.63	10.00	0.24	4.88	8.31	211.00	4.83	219.67	0.02	1.34	8.26	229.00	9.20	767.00	0.01
10 Windows x 100 series, 1 kHz Data Rate	10 M	111.36	55.28	1138.67	4.73	26.00	0.53	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 200 series, 1 kHz Data Rate	2 M	260.40	12.97	296.00	0.67	9.67	0.17	2.50	8.37	280.00	7.20	414.33	0.02	0.69	8.44	311.67	9.03	1511.67	0.01
10 Windows x 200 series, 1 kHz Data Rate	20 M	90.02	61.24	1184.33	4.17	27.67	0.38	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 1000 series, 1 kHz Data Rate	10 M	68.67	12.73	411.00	1.53	30.00	0.11	0.53	8.49	491.00	9.20	2011.33	0.00	0.16	8.44	765.67	61.53	7469.33	0.01
5 Windows x 1000 series, 1 kHz Data Rate	50 M	41.35	47.81	1351.33	5.17	55.33	0.21	--	--	--	--	--	--	--	--	--	--	--	--
10 Windows x 1000 series, 1 kHz Data Rate	100 M	23.14	62.18	2344.00	9.30	94.67	0.22	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 10 series, 10 kHz Data Rate	1 M	451.09	9.72	250.67	1.07	17.00	0.48	60.04	3.58	288.33	0.50	17.00	0.03	1.34	8.37	207.67	10.90	794.33	0.01
10 Window x 10 series, 10 kHz Data Rate	10 M	133.00	44.04	907.67	5.60	25.33	0.74	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 100 series, 10 kHz Data Rate	10 M	377.66	11.02	342.33	0.67	13.00	0.25	6.70	7.53	300.33	8.57	167.00	0.06	0.19	8.66	867.33	28.73	7389.00	0.01
10 Windows x 100 series, 10 kHz Data Rate	100 M	103.62	53.34	1779.67	5.20	29.67	0.54	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 200 series, 10 kHz Data Rate	20 M	256.55	11.78	351.67	0.67	14.00	0.17	3.55	9.09	502.33	8.13	311.00	0.03	0.09	8.50	3476.67	6525.90	15160.67	0.57
10 Windows x 200 series, 10 kHz Data Rate	200 M	80.56	55.99	1912.33	4.27	30.33	0.34	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 1000 series, 10 kHz Data Rate	100 M	66.46	13.56	480.00	2.13	47.00	0.14	0.76	9.99	1526.33	13.57	1389.00	0.01	0.03	8.44	7610.67	31196.73	75661.00	0.83
5 Windows x 1000 series, 10 kHz Data Rate	500 M	33.10	40.14	1645.33	7.77	68.67	0.26	--	--	--	--	--	--	--	--	--	--	--	--
10 Windows x 1000 series, 10 kHz Data Rat	1 B	15.37	48.17	3459.00	21.73	153.00	0.33	--	--	--	--	--	--	--	--	--	--	--	--
1 Windows x 1000 series, 100 kHz Data Rat	1 B	49.47	15.52	1172.33	3.87	43.33	0.19	0.46	30.99	13632.00	33.67	2350.00	0.02						--
5 Windows x 1000 series, 100 kHz Data Rat	5B	9.45	31.99	5589.67	46.10	246.67	0.44	--	--	--	--	--	--	--	--	--	--	--	--
1 Windows x 500 series, 500 kHz Data Rate	2.5 B	47.33	15.13	3813.00	1.83	32.00	0.09	0.00	43.71	91367.00	999999.00	999999.00	0.00						--
5 Windows x 500 series, 500 kHz Data Rate	12.5 B	17.74	61.23	23851.00	34.67	176.00	0.62	--	--	--	--	--	--	--	--	--	--	--	--

Green = *Good.* **Yellow** = *satisfactory.* **Red** = *struggling.* ***Bright red*** = *FAIL*

The results proof that LightningChart is capable to render billion data points on a mid-Level desktop PC with an acceptable FPS (at least 30 FPS), with 1000 series all together.

Additionally, 1 million data points can be rendered as fast as 450 FPS. At the same time SD and RSD remains low, which indicates an extremely smooth scrolling. An elevated CPU load for LightningChart is primarily due to the High FPS mode (required by application).

LightningChart's CPU/frame rate is extremely low in comparison to the *Competitor*. As it will be seen in the test for 'CPU saving mode', with a moderate FPS the load of the CPU **gets as low as 2% per window**.

In contrast, the Competitor's performance deteriorates very fast if more than 1 million points are to be rendered on the screen. The Competitor does not tolerate high series counts (more than 10) because the performance declines very fast if the series count increases from 10 to 100.

The Low RCD of the Competitor is meaningless in this context, because 0.1-5 FPS is not what a user would be looking forward to seeing. It is not surprising that without down sampling, the *Competitor* struggles even more.

LightningChart maintains a relatively stable RAM usage through all the tests, but the decrease ration for more demanding tests is an indication of the Competitor failing to manage the memory.

In this test, how many times faster LightningChart is?

High FPS mode in all tests		**LightningChart vs Competitor (Resampling = Auto)**	**LightningChart vs Competitor (Resampling = None)**
Test	**Total points count**	**FPS** LC times Faster than Competitor	**FPS** LC times Faster than Competitor
1 Window x 10 series, 1 kHz Data Rate	100 k	7.6	7.6
10 Window x 10 series, 1 kHz Data Rate	1 M		
1 Window x 100 series, 1 kHz Data Rate	1 M	79.2	287.4
10 Windows x 100 series, 1 kHz Data Rate	10 M		
1 Window x 200 series, 1 kHz Data Rate	2 M	104.0	375.6
10 Windows x 200 series, 1 kHz Data Rate	20 M		
1 Window x 1000 series, 1 kHz Data Rate	10 M	129.6	420.4
5 Windows x 1000 series, 1 kHz Data Rate	50 M		
10 Windows x 1000 series, 1 kHz Data Rate	100 M		
1 Window x 10 series, 10 kHz Data Rate	1 M	7.5	335.8
10 Window x 10 series, 10 kHz Data Rate	10 M		
1 Window x 100 series, 10 kHz Data Rate	10 M	56.3	1987.7
10 Windows x 100 series, 10 kHz Data Rate	100 M		
1 Window x 200 series, 10 kHz Data Rate	20 M	72.3	2960.2
10 Windows x 200 series, 10 kHz Data Rate	200 M		
1 Window x 1000 series, 10 kHz Data Rate	100 M	87.8	2492.3
5 Windows x 1000 series, 10 kHz Data Rate	500 M		
10 Windows x 1000 series, 10 kHz Data Rat	1 B		
1 Windows x 1000 series, 100 kHz Data Rat	1 B	106.8	
5 Windows x 1000 series, 100 kHz Data Rat	5 B		
1 Windows x 500 series, 500 kHz Data Rate	2.5 B		
5 Windows x 500 series, 500 kHz Data Rate	12.5 B		

Green = *LightningChart is faster.* ***Brown*** *= no significant difference,* ***Red*** = Competitor *is faster*

The empty rows are due to the Competitor’s failure to produce any measurable values. As the FPS ration indicates, LightningChart can render about **3000 times faster** (when the Competitor Resampling is disabled), and **130 times faster** when Competitor Resampling is enabled, in an application where points count and 100’s millions or several billions. A higher CPU load of LightningChart is due to the application requesting maximize the FPS rate.

2. Desktop with CPU saving mode

Secondly, here are the results for the set of tests with a Desktop PC. This time, the charts were asked to save CPU and limit to 60 FPS (*CPU-saving mode*).

		LightningChart v10.1.1.4001 -> CPU saving mode in all tests						Competitor Xccelerator Engine + Impossible Mode ON + Resample Mode Auto						Competitor Xccelerator Engine + Impossible Mode ON + Resample Mode NONE					
Test	**Total points count**	FPS	CPU Load (%)	RAM (MB)	SD (ms)	Max (ms)	RSD/CV	FPS	CPU Load (%)	RAM (MB)	SD (ms)	Max (ms)	RSD/CV	FPS	CPU Load (%)	RAM (MB)	SD (ms)	Max (ms)	RSD/CV
1 Window x 10 series, 1 kHz Data Rate	100 k	60.05	2.43	265.33	0.50	17.00	0.03	60.02	2.34	271.00	0.57	18.33	0.03	60.05	3.51	336.67	0.50	17.00	0.03
10 Window x 10 series, 1 kHz Data Rate	1 M	39.48	20.71	805.33	8.07	31.00	0.32	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 100 series, 1 kHz Data Rate	1 M	59.82	1.96	292.33	5.90	32.67	0.35	4.88	8.31	211.00	4.83	219.67	0.02	1.34	8.26	229.00	9.20	767.00	0.01
10 Windows x 100 series, 1 kHz Data Rate	10 M	39.98	22.58	1161.00	8.43	30.33	0.34	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 200 series, 1 kHz Data Rate	2 M	59.84	3.71	308.00	0.53	17.67	0.03	2.50	8.37	280.00	7.20	414.33	0.02	0.69	8.44	311.67	9.03	1511.67	0.01
10 Windows x 200 series, 1 kHz Data Rate	20 M	40.16	31.48	1318.33	8.27	33.67	0.33	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 1000 series, 1 kHz Data Rate	10 M	60.08	11.18	412.67	0.50	16.67	0.03	0.53	8.49	491.00	9.20	2011.33	0.00	0.16	8.44	765.67	61.53	7469.33	0.01
5 Windows x 1000 series, 1 kHz Data Rate	50 M	37.81	40.89	1471.67	8.13	48.33	0.31	--	--	--	--	--	--	--	--	--	--	--	--
10 Windows x 1000 series, 1 kHz Data Rate	100 M	23.26	61.79	2286.00	10.40	91.67	0.24	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 10 series, 10 kHz Data Rate	1 M	60.09	1.99	256.00	0.50	17.33	0.03	60.04	3.58	288.33	0.50	17.00	0.03	1.34	8.37	207.67	10.90	794.33	0.01
10 Window x 10 series, 10 kHz Data Rate	10 M	37.53	17.10	890.33	8.20	33.00	0.31	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 100 series, 10 kHz Data Rate	10 M	60.16	1.91	326.67	0.50	17.00	0.03	6.70	7.53	300.33	8.57	167.00	0.06	0.19	8.66	867.33	28.73	7389.00	0.01
10 Windows x 100 series, 10 kHz Data Rate	100 M	39.50	24.17	1828.00	8.27	34.00	0.33	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 200 series, 10 kHz Data Rate	20 M	60.01	3.23	341.67	0.50	17.00	0.03	3.55	9.09	502.33	8.13	311.00	0.03	0.09	8.50	3476.67	6525.90	15160.67	0.57
10 Windows x 200 series, 10 kHz Data Rate	200 M	39.29	30.77	1964.33	8.20	34.00	0.32	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 1000 series, 10 kHz Data Rate	100 M	60.03	11.82	465.33	0.50	17.00	0.03	0.76	9.99	1526.33	13.57	1389.00	0.01	0.03	8.44	7610.67	31196.73	75661.00	0.83
5 Windows x 1000 series, 10 kHz Data Rate	500 M	32.70	39.26	1843.00	10.47	82.67	0.34	--	--	--	--	--	--	--	--	--	--	--	--
10 Windows x 1000 series, 10 kHz Data Rat	1 B	15.05	48.16	3532.33	15.20	123.33	0.23	--	--	--	--	--	--	--	--	--	--	--	--
1 Windows x 1000 series, 100 kHz Data Rat	1 B	49.51	14.35	1261.00	6.80	39.33	0.34	0.46	30.99	13632.00	33.67	2350.00	0.02						--
5 Windows x 1000 series, 100 kHz Data Rat	5B	9.26	34.25	5713.33	46.70	444.67	0.43	--	--	--	--	--	--	--	--	--	--	--	--
1 Windows x 500 series, 500 kHz Data Rate	2.5 B	52.11	19.22	3474.33	6.70	119.00	0.35	0.00	43.71	91367.00	999999.00	999999.00	0.00						--
5 Windows x 500 series, 500 kHz Data Rate	12.5 B	9.16	70.75	24649.67	50.03	238.33	0.46	--	--	--	--	--	--	--	--	--	--	--	--

Green = *Good.* **Yellow** = *satisfactory.* **Red** = *struggling.* ***Bright red*** = *FAIL*

Similarly, to previous test set (in *High FPS mode*), the results show LightningChart is capable of rendering billions of data points on a mid-Level desktop PC while maintaining an acceptable FPS (30-60 FPS). As much as 2.5 billion points (500 series x 500 kHz sampling x 10 sec) can be rendered at 50+ FPS with a mid-Level desktop PC. At the same time, the CPU load is as low as 2% even for 10 million points.

In contrast, the Competitor's performance deteriorates very fast if more than 1 million points need to be rendered on the screen. The Competitor does not cope with a high series count (more than 10), because performance declines very fast if the series count increases from 10 to 100.

A low RCD of the Competitor is meaningless in this context, because 0.1-5 FPS is not what a user would be looking forward to seeing. It is not surprising that without down sampling, the Competitor struggles even more.

It is efficiently proven that LightningChart is running with a very low CPU, requiring only around **1/80 of CPU** required by the *Competitor*.

In this test, how many times faster LightningChart is?

CPU saving mode in all tests		**LightningChart vs Competitor (Resampling = Auto)**	**LightningChart vs Competitor (Resampling = None)**
Test	**Total points count**	**FPS** LC times Faster than Competitor	**FPS** LC times Faster than Competitor
1 Window x 10 series, 1 kHz Data Rate	100 k	1.0	1.0
10 Window x 10 series, 1 kHz Data Rate	1 M		
1 Window x 100 series, 1 kHz Data Rate	1 M	12.3	44.5
10 Windows x 100 series, 1 kHz Data Rate	10 M		
1 Window x 200 series, 1 kHz Data Rate	2 M	23.9	86.3
10 Windows x 200 series, 1 kHz Data Rate	20 M		
1 Window x 1000 series, 1 kHz Data Rate	10 M	113.4	367.9
5 Windows x 1000 series, 1 kHz Data Rate	50 M		
10 Windows x 1000 series, 1 kHz Data Rate	100 M		
1 Window x 10 series, 10 kHz Data Rate	1 M	1.0	44.7
10 Window x 10 series, 10 kHz Data Rate	10 M		
1 Window x 100 series, 10 kHz Data Rate	10 M	9.0	316.6
10 Windows x 100 series, 10 kHz Data Rate	100 M		
1 Window x 200 series, 10 kHz Data Rate	20 M	16.9	692.4
10 Windows x 200 series, 10 kHz Data Rate	200 M		
1 Window x 1000 series, 10 kHz Data Rate	100 M	79.3	2251.3
5 Windows x 1000 series, 10 kHz Data Rate	500 M		
10 Windows x 1000 series, 10 kHz Data Rat	1 B		
1 Windows x 1000 series, 100 kHz Data Rat	1 B	106.8	
5 Windows x 1000 series, 100 kHz Data Rat	5B		
1 Windows x 500 series, 500 kHz Data Rate	2.5 B		
5 Windows x 500 series, 500 kHz Data Rate	12.5 B		

Green *= LightningChart is faster.* ***Brown*** *= no significant difference,* ***Red*** = Competitor *is faster*

With the application designed to maintain a moderate FPS (30-60), LightningChart easily outperformed the *Competitor* in all the important categories.

While the CPU load and the memory usage are low at any test with more than 1 Million points, the FPS remains at least **2000 times higher** than the Competitor. The empty rows are due to Competitor's failure to produce any measurable values.

3. Laptop in High-FPS mode

Finally, we repeated the tests with a budget laptop (*High FPS mode*). The result table can be found below.

		LightningChart v10.1.1.4001 -> High FPS mode in all tests						Competitor Xccelerator Engine + Impossible Mode ON + Resample Mode Auto						Competitor Xccelerator Engine + Impossible Mode ON + Resample Mode NONE					
Test	**Total points count**	FPS	CPU Load (%)	RAM (MB)	SD (ms)	Max (ms)	RSD/CV	FPS	CPU Load (%)	RAM (MB)	SD (ms)	Max (ms)	RSD/CV	FPS	CPU Load (%)	RAM (MB)	SD (ms)	Max (ms)	RSD/CV
1 Window x 10 series, 1 kHz Data Rate	100 k	111.60	16.60	235.00	1.00	25.00	0.11	60.93	15.17	302.00	4.43	427.33	0.27	18.71	12.87	132.33	15.13	102.67	0.28
10 Window x 10 series, 1 kHz Data Rate	1 M	22.73	71.33	912.33	11.17	116.33	0.25	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 100 series, 1 kHz Data Rate	1 M	96.53	16.67	276.67	1.37	51.33	0.13	6.40	13.23	183.67	38.10	314.67	0.24	2.23	12.50	202.33	163.33	680.33	0.36
10 Windows x 100 series, 1 kHz Data Rate	10 M	19.28	73.23	1172.67	17.70	174.33	0.34	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 200 series, 1 kHz Data Rate	2 M	78.73	17.00	299.67	2.27	32.67	0.18	3.37	12.63	230.00	59.40	460.33	0.20	1.24	12.43	271.33	342.97	1044.00	0.42
10 Windows x 200 series, 1 kHz Data Rate	20 M	13.90	67.70	1368.00	41.50	423.00	0.58	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 1000 series, 1 kHz Data Rate	10 M	28.37	14.87	453.00	4.97	117.67	0.14	0.75	13.07	496.67	343.40	1666.00	0.26	0.24	12.80	781.67	1858.67	5005.33	0.45
5 Windows x 1000 series, 1 kHz Data Rate	50 M	9.73	54.70	1469.33	34.00	424.00	0.33	--	--	--	--	--	--	--	--	--	--	--	--
10 Windows x 1000 series, 1 kHz Data Rate	100 M	3.73	70.93	2782.00	113.00	919.00	0.42	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 10 series, 10 kHz Data Rate	1 M	107.30	16.53	254.33	1.60	102.33	0.17	59.70	14.77	353.67	5.73	617.00	0.34	2.35	12.37	199.67	176.60	564.67	0.42
10 Window x 10 series, 10 kHz Data Rate	10 M	20.23	70.57	981.33	18.97	214.33	0.38	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 100 series, 10 kHz Data Rate	10 M	92.93	17.23	340.00	1.67	39.00	0.15	8.83	13.46	299.33	11.23	177.67	0.10	0.31	12.93	851.00	2127.33	21611.33	0.66
10 Windows x 100 series, 10 kHz Data Rate	100 M	17.30	65.70	1854.67	25.00	267.67	0.43	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 200 series, 10 kHz Data Rate	20 M	74.97	16.03	355.67	2.17	52.67	0.16	4.63	13.63	499.00	21.00	344.67	0.10	0.18	12.53	2168.33	4700.67	13452.00	0.85
10 Windows x 200 series, 10 kHz Data Rate	200 M	13.63	65.60	2048.67	33.63	478.33	0.46	--	--	--	--	--	--	--	--	--	--	--	--
1 Window x 1000 series, 10 kHz Data Rate	100 M	29.50	14.87	518.33	4.97	134.33	0.15	0.95	15.77	1519.67	136.67	1517.00	0.13	0.08	13.38	6336.00	17804.33	51942.33	1.48
5 Windows x 1000 series, 10 kHz Data Rate	500 M	8.87	51.93	1961.33	58.97	817.33	0.52	--	--	--	--	--	--	--	--	--	--	--	--
10 Windows x 1000 series, 10 kHz Data Rat	1 B	2.56	50.66	3703.00	242.40	1391.00	0.62	--	--	--	--	--	--	--	--	--	--	--	--

Green = *Good.* **Yellow** = *satisfactory.* **Red** = *struggling.* ***Bright red*** = *FAIL*

As expected, the budget laptop's performance was lower than compared to a mid-Level desktop PC. However, the overall picture of the performance between LightningChart and Competitor remains the same (as reported above).

Certainly, 1 million points and 10 series seem to be the limit for the Competitor product. In contrast, LightningChart is able to render (at 30 FPS) 100 Million points in 1000 series.

In this test, how many times faster LightningChart is?

High FPS mode in all tests		**LightningChart vs Competitor (Resampling = Auto)**	**LightningChart vs Competitor (Resampling = None)**
High FPS mode in all tests	**Total points count**	**FPS** LC times Faster than Competitor	**FPS** LC times Faster than Competitor
1 Window x 10 series, 1 kHz Data Rate	100 k	1.8	6.0
10 Window x 10 series, 1 kHz Data Rate	1 M		
1 Window x 100 series, 1 kHz Data Rate	1 M	15.1	43.2
10 Windows x 100 series, 1 kHz Data Rate	10 M		
1 Window x 200 series, 1 kHz Data Rate	2 M	23.4	63.7
10 Windows x 200 series, 1 kHz Data Rate	20 M		
1 Window x 1000 series, 1 kHz Data Rate	10 M	38.0	118.2
5 Windows x 1000 series, 1 kHz Data Rate	50 M		
10 Windows x 1000 series, 1 kHz Data Rate	100 M		
1 Window x 10 series, 10 kHz Data Rate	1 M	1.8	45.7
10 Window x 10 series, 10 kHz Data Rate	10 M		
1 Window x 100 series, 10 kHz Data Rate	10 M	10.5	299.8
10 Windows x 100 series, 10 kHz Data Rate	100 M		
1 Window x 200 series, 10 kHz Data Rate	20 M	16.2	416.5
10 Windows x 200 series, 10 kHz Data Rate	200 M		
1 Window x 1000 series, 10 kHz Data Rate	100 M	31.1	354.0
5 Windows x 1000 series, 10 kHz Data Rate	500 M		
10 Windows x 1000 series, 10 kHz Data Rat	1 B		

Green = *LightningChart is faster.* **Brown** = *no significant difference,* **Red** = Competitor *is faster*

LightningChart outperforms Competitor, being up to **400 times faster**, or even more.

Down sampling and misleading performance claims

We verified the Competitor's "billion points" demo, which is a replica of LightningChart's billion points demo. We found out it provides highly misleading information as is purely impossible to make it render with such claimed data rates and refresh rates presented in their video.

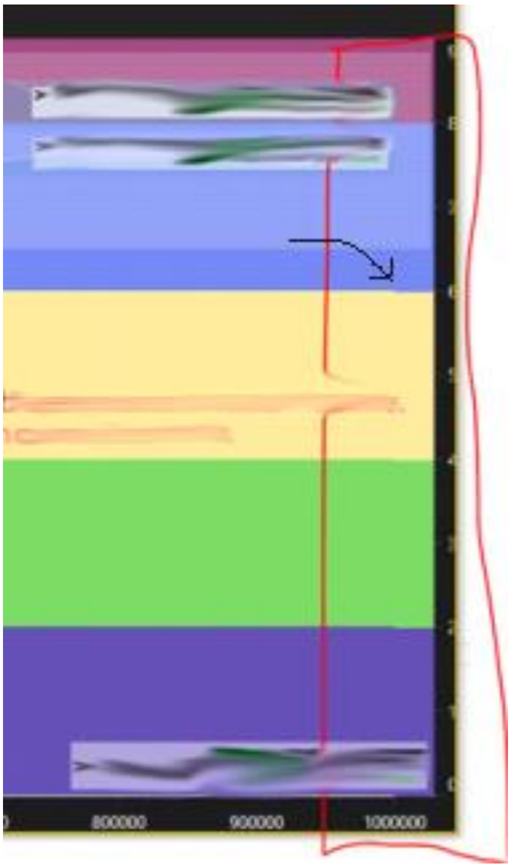
In the "billion points" demo, the Competitor is down sampling/resampling the prefilled dataset. The down sampling / resampling is being used also throughout the entire test. The random data generated, hides the issues produced by down sampling/resampling for users who do not pay attention to this.

In their "billion points" YouTube video, the FPS readings are very far from the truth. The chart does not update with a shown rate. The FPS is being shown high however, the chart does not update.

The marketing information of the *Competitor* is totally misleading, and the performance claims are not achievable.

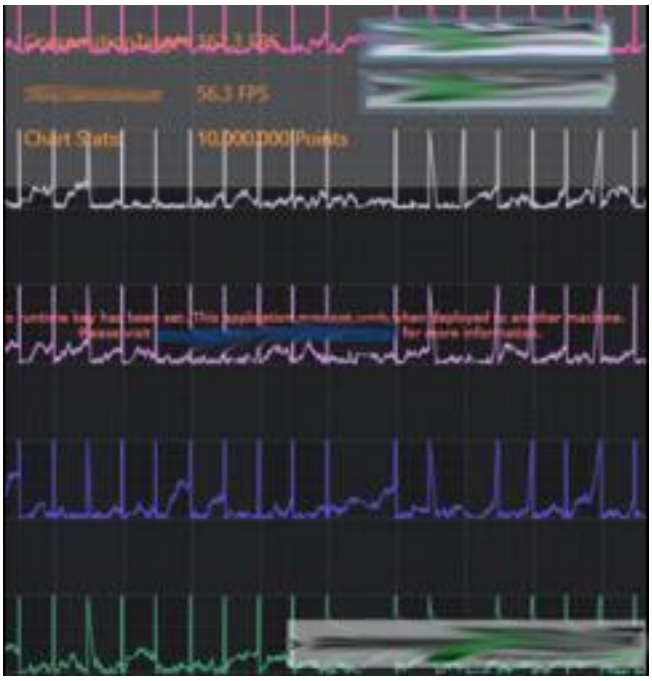
Performance Tricks of the Competitor

Trick 1. Prebuffered data being resampled.



Competitor Resampling is Off. Still data shape switches after prebuffered region.

Trick 2. Down sampling/Resampling

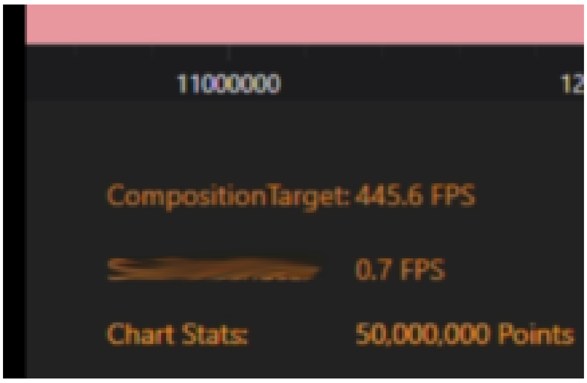


The used data contains peaks at regular intervals. Down sampling/Resampling is very coarse and it is missing peaks and producing diagonal line transitions.

Using down sampling/resampling is a trick to improve performance but it shouldn't be used for any real application!

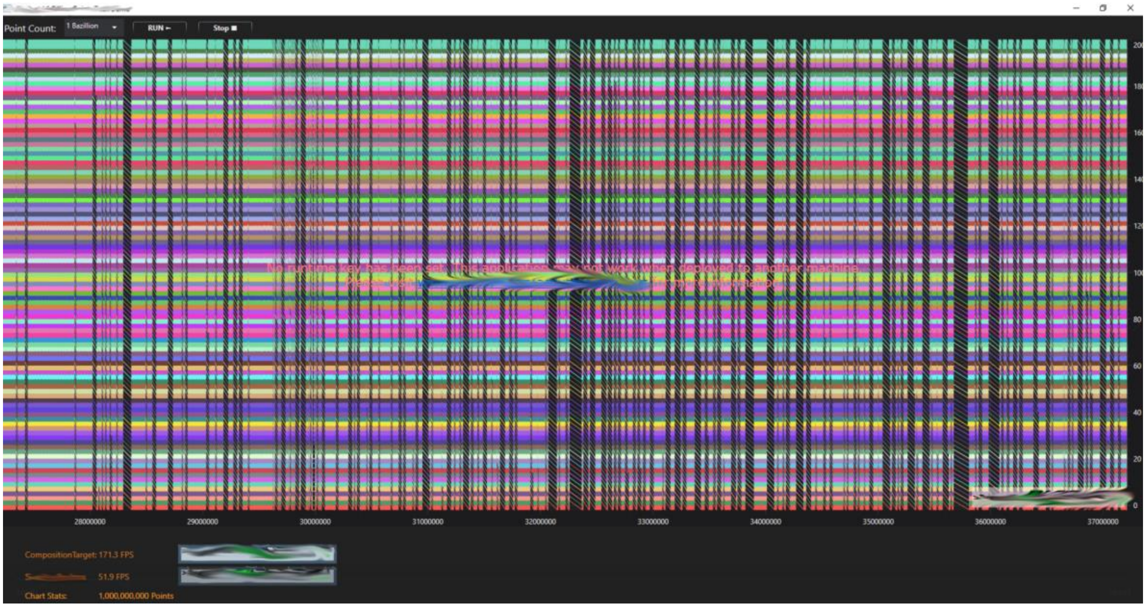
Trick 3. FPS Calculator Lags

The FPS calculator lags behind for some time so it gives a very high FPS but even the chart cannot clearly update with such claimed data rate. Even when pushing the data in a rate the chart cannot update at all, the FPS rate shows high.



When Down sampling/Resampling is disabled, the rendering goes unusable with 10 million data points and 50 million data points, 0.7 FPS is shown, and CompositionTarget shows 445.6 FPS. The chart actually cannot render even with 0.7 FPS, but 0.3 FPS or similar.

Trick 4. Faulty rendering algorithms



Point Count: 1 Bazillion

CompositionTarget: 171.3 FPS

[redacted] 51.9 FPS

Chart Stats: 1,000,000,000 Points

The rendering Competitor uses may save computer resources, but apparently line rendering is not working, and it is skipping data blocks. Resampling is enabled.

(LightningChart team has reported this issue to SciChart).

Conclusion of the Test

The *Competitor* is a fast hardware accelerated library but **LightningChart** being **up to 3000 times faster**, **outperforms the Competitor in all tests.** So, even when the *Competitor* resampling has been enabled, **LightningChart still is 130 times faster**.

LightningChart's **SampleDataBlockSeries** is the optimal scientific data visualization series type to visualize scrolling/sweeping real-time data. Additionally, the **SampleDataBlockSeries** uses less RAM and CPU resources, provides a higher FPS rate, and a significantly better responsiveness to user interactions.

If we had tested with a higher performance computer hardware, the performance difference between LightningChart and Competitor would have been even greater.

Unlike the *Competitor*, **LightningChart's performance is real**, needing no gimmicks or tuning special flags to produce the correct rendering output. The *Competitor* suffers from an erroneous rendering algorithms and disadvantages produced by resampling. The *Competitor* doesn't work in a multi-window solution using several dispatchers, so we couldn't measure and compare performance in that scenario.

*LightningChart® is an extreme and the **World's fastest charting library**, providing simply unmatched performance for serious medical, telemetry, finance, science, and engineering apps. It includes built-in sweeping mode and works perfectly in multi-window and multi-dispatcher environments.*

*LightningChart supports all WPF, UWP and WinForms technologies, and switching from one technology to another is easy, so all customers can ensure the development is possible during a long lifetime of the application. **LightningChart** real-time monitoring systems have been developed since 2008 and is the **pioneer in GPU-accelerated real-time charts**.*