CMPT353

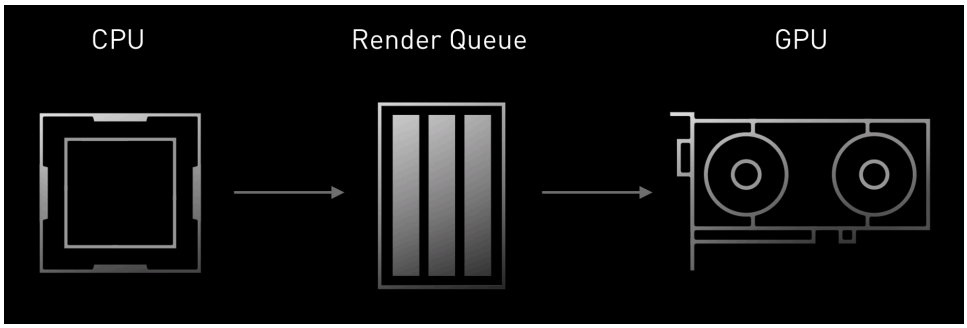
PROJECT REPORT

Peter Soava, pgs2
Caleb Bradley, cba52

What is Nvidia Reflex?

In our modern world, we increasingly interact with video games as a common pastime. More often than not, however, the responsiveness of those games is taken for granted. Development budgets are commonly driven towards impressive presentation rather than immediate feedback, harming user experience but improving visual appeal. To help remedy this, in 2020 Nvidia released Reflex, a latency-reduction technology.

Reflex is a technology designed for 3D applications to facilitate efficient communication between the CPU and GPU, thus reducing input latency. When generating an image the CPU must request the GPU to generate frames. However, if the CPU requests a new frame before the GPU has generated the previous one it is sent to a render queue.



Frame Generation Pipeline (Source: Nvidia)

The major issue with a render queue is that if a frame lingers in the queue too long it can contain data that doesn't match the user's input, thus creating input lag.

Nvidia Reflex aims to cut this delay down by minimizing the render queue as much as possible. However, this poses a couple of questions. Does Reflex reduce latency, and if so, is it a significant enough difference to warrant using it? Ultimately we aimed to answer those questions in this report by answering these sub-questions:

1. **Does running Reflex yield a noteworthy reduction in latency in any game?**
2. **Does Reflex's effectiveness vary between different graphical settings?**
3. **Does Reflex have a similar impact on latency between different games?**

Data Collection (Overall)

To capture the latency data required to answer these questions we used a Nvidia-authored program named FrameView. FrameView produces a .csv file with an immense amount of statistics for each frame an application renders. Although there is plenty more data that FrameView captures, for the sake of keeping this project's scope in check we opted to only analyze PC latency (in ms).

Nvidia FrameView (Desktop and In-Game)

Data Collection (Questions 1 and 3)

Since the render queue mainly gets filled in situations where the CPU is ahead of the GPU we only need to collect data (with Reflex on and off) in two situations:

1. The GPU is at 100% load by working on costly rendering.
2. The GPU is at low load (< 50%) by working on simple rendering.

Situation 1 can be satisfied by tuning a game's graphical options as high as possible and by setting the render resolution to an extremely high value. By doing this the GPU is busy at all times and the CPU can saturate the render queue to its fullest extent.

Conversely, doing the opposite by setting the graphics options as low as possible with an FPS cap will satisfy *situation 2*. In that condition, the GPU is free enough to generate a frame as soon as the CPU sends a request for one.

	GPU STRAIN	
	LOW	HIGH
REFLEX ON	dataset1	dataset3
REFLEX OFF	dataset2	dataset4

Data Collection (Question 2)

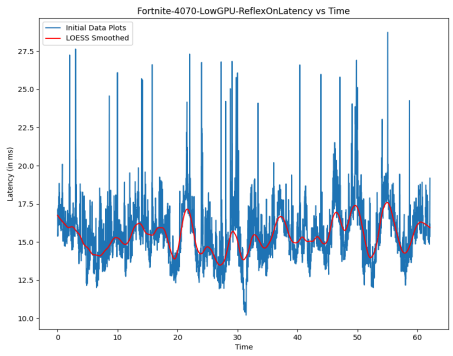
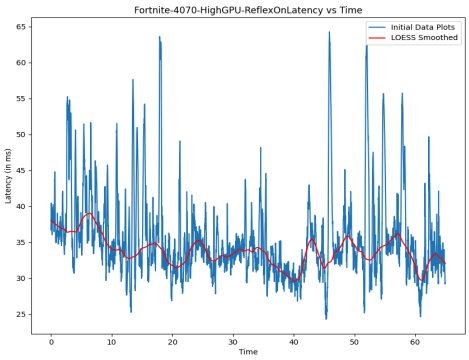
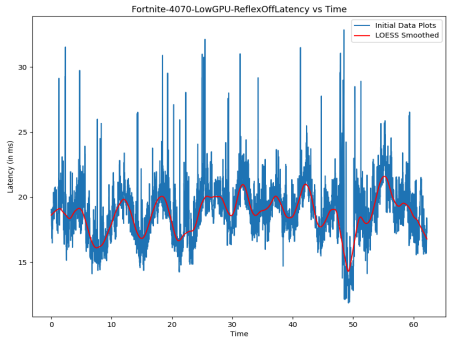
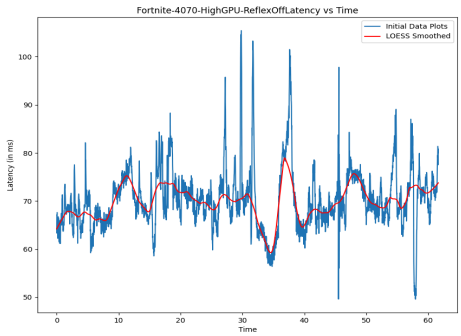
For our tests between multiple graphics settings, the benchmarking methodology was largely the same, but instead of looking for "low and high" GPU load, we searched for different graphics settings. Using Fortnite as the baseline for this test, this is how our data turned out:

	FORTNITE GRAPHICS SETTING					
	LOW	NVIDIA	MEDIUM	HIGH	EPIC	PERSONAL
REFLEX ON	dataset1	dataset3	dataset5	dataset7	dataset9	dataset11
REFLEX OFF	dataset2	dataset4	dataset6	dataset8	dataset10	dataset12

Data Filtering

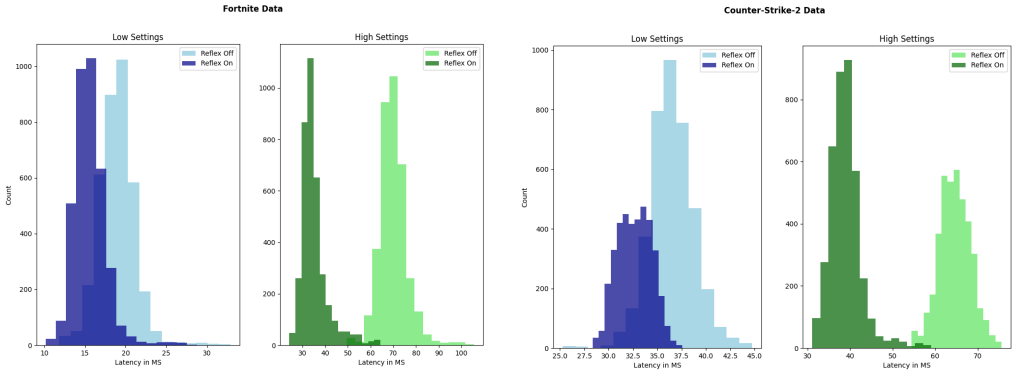
FrameView by default provides an overwhelming wealth of columns of data to sort through. However, given we're only mainly interested in the PC latency and time, we drop all of them except for two columns - 'TimeInSeconds' and 'LatencyInMS'.

Since games give quite variable data, we also did some minor smoothing using the LocalOutlierFactor anomaly detection algorithm to help straighten the latency data from any significant lag spikes. Below are the final graphs on our Fortnite data, with a Loess Smoothed regression line to help make things more readable.

	LOW GPU LOAD	HIGH GPU LOAD
REFLEX ON		
REFLEX OFF		

Data Analysis: Sub-Question 1

This first glimpse of the datasets reassures that there is a measure of difference between when Reflex was on compared to off. This is especially true when looking at when the GPU is under high load.

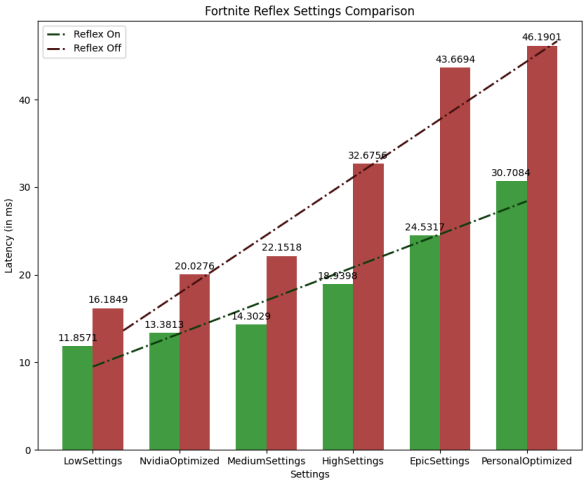


With the null hypothesis for each case being *"Reflex does not have a noteworthy difference on the data,"* we ran a t-test. The corresponding P-values ended up so small that they rounded to zero in almost every case. This statistically confirmed the visual results, which implies reflex does have some effect on the game's latency, with the graphs showing Reflex On is lower than Off.

Therefore, the answer to Sub-Question 1 was: **Yes**, Reflex provides a reduction in latency.

Data Analysis: Sub-Question 2

Our second test analyzed the latency provided by a large variety of different visual settings on a single game. Taking advantage of Fortnite's scalability, we used it as the basis of this test to get as varied data as possible with a single game as a constant.



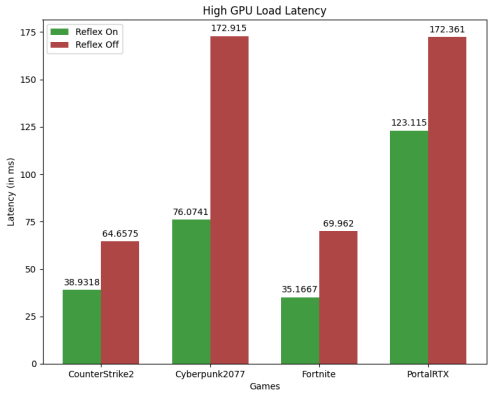
For this test, we gathered and filtered data under six different settings presets. To understand the relationship, we generated a regression line between the average latencies for Reflex On and Off. The slope of the regression line for the Reflex On data is only 3.78, whereas the slope of the Reflex Off data rests at 6.61. This shows that when Reflex is off the amount of latency you gain as you increase graphical settings can be almost as much as 2x compared to Reflex On. Also, since the P-value of both regressions is well below 0.01, we can infer these slopes are confident representations of our data.

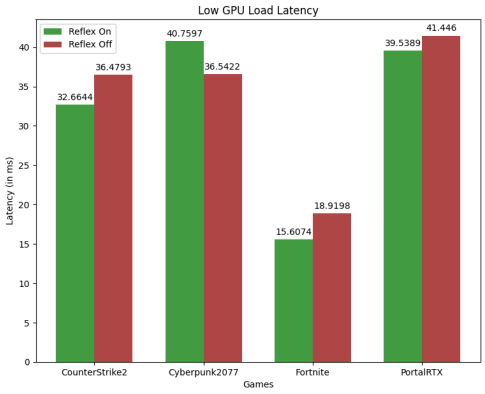
Slope for Reflex_On:	3.78
Reflex_On Regression P-Value:	0.0025
Slope for Reflex_Off:	6.61
Reflex_Off Regression P-Value:	0.0011

Using this analysis we conclude that, **yes**, graphics settings make a huge impact on Reflex.

Data Analysis: Sub-Question 3

The third test determined if Reflex has an effect across different games. We decided to find this by using a chi-squared test as each game can be considered a category.





For High Load situations, we found a P-value of 0.05 from a Chi-Square test. This implies that for each category (games) Reflex does have a measurable difference between on and off. For Low Load situations, we found a P-value of 0.87 which suggests the opposite of little difference between Reflex On and Off.

Low GPU Chi-Square Statistic:	0.71
Low GPU P-value:	**0.87**
High GPU Chi-Square Statistic:	7.68
High GPU P-value:	**0.05**

Given that, our analysis proved interesting: it suggested that regardless of game, Reflex has a similarly low effect in Low GPU usage situations, and a similarly high effect in High GPU situations. That answers our third question as, **yes**, Reflex maintains a similar difference between High and Low loads in different games.

Conclusion

By collecting, refining, and analyzing our data we found substantial evidence that Reflex **is** a very effective technology at reducing latency in certain circumstances. For games that are very easy to render you'll likely see little to no difference between Reflex being enabled. However, for more difficult-to-render games, Reflex can provide a massive reduction in latency.

These improvements are important because they enable more people to maximize the performance of their systems, which can promote system longevity, decrease component cost, and improve the overall gameplay experience. Simply enabling one setting results in a noteworthy benefit without sacrifice.

Limitations

The largest limitation we feel we faced was that our data doesn't take the variability of hardware into account. Although we did collect data between multiple machines, in the end, it proved too large in scope to fit into this project as we already analyzed so much.

Another place we feel would've been nice to optimize a bit more was to do benchmark runs a few more times. Often games have shader caches that cause stuttering unless you've already played through an area. We feel like some of our data may have received some unnecessary stutters due to not having shaders compiled.

Lastly, we had to cut a ton of information to fit our analysis into the limited page count. Computer graphics are very complex, and this final report holds a fraction of the detail of the original draft.

References

https://www.nvidia.com/en-us/geforce/news/reflex-low-latency-platform/
https://www.nvidia.com/en-us/geforce/technologies/reflex/

Project Experience Summary

- **Peter**
    - Took charge in leading the direction of the project by finding the topic, providing thorough background knowledge, and developing the most critical portions of data analysis to maintain cohesive conclusions and keep the project focused.
- **Caleb**
    - Streamlined a complex dataset by providing expertise in data refinement. Eliminated irrelevant components using machine learning, manipulating Pandas data frames, and deploying graphical tools to enhance clarity and set up data for robust analytical conclusions.