IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

Method to Capture Tearing and Stuttering in a GPU Driven Application

Yogesh Dasgaonkar[1], K K Joshi[2]

Department of Computer Engineering, Veermata Jijabai Technological Institute (VJTI), Mumbai, Maharashtra, India[1,2]

Abstract: The manufacturing of a display system associated with a GPU system involves a mathematically driven algorithm and is based on several use cases that determine this algorithm to be designed in the correct way so as to avoid the various display related issues like stuttering, flickering, ghosting etc. This paper explains the various issues related to display technologies and provides a mechanism to calculate tearing and stutter datamathematically rather than observing them only through visual means. When we are dealing with the variable frames per second (fps) output of the GPU processed data to be outputted on the screen in real time , the type of activity that may be required to challenge occurs when the frames per second changes in a real time game graphics engine. There are displays suited to gaming, design work, and ultra-high resolution detail, but mixing priorities results in compromise. Fast response time, colorfully vibrant, or high-DPI—pick any two, because there's still no monitor that quite hits every bullet point. Even if a monitor hits all the bullet points it cannot always guarantee that it will be free from the issues associated with a display system. There exist a number of issues with these systems like stuttering, flickering,ghosting, micro stuttering and tearing which has a very high impact on the output and gameplay of the game played on that GPU system. These common display output issues will hereafter be referred to as "display system issues".Visual Technology, graphic intensive applications and the display market itself (which is estimated to be worth 169.17 Billion USD by 2022) will be required to be able to incorporate means to avoid stutters and tearing.[5]

Keywords: stuttering, flickering , ghosting , micro stuttering and tearing.

I. INTRODUCTION

The graphics display resolution comprises of mainlythe width, height, and color depth and refresh rate dimensions of an electronic visual display device, such as a computer monitor, in pixels. There may be a large number of custom resolutions supported by the system, but certain combinations of width, height and aspect ratio are standardized and typically given a marketing name that defines the type of the panel. Increasing the display resolution in a display of the same size means that display content will appear more sharp and crisp.

The popular screen displays available currently are:

1. Standard Definition (SD) 720 × 480
2. qHD 960 x 540 pixels
3. HD 1280 × 720
4. Full HD 1920 × 1080
5. 2K 2048 × 1152
6. Quad HD (QHD) 2560 × 1440
7. Quad Full HD (QFHD)/Ultra HD 3840 × 2160
8. 4K 4096×2160

The effects of display related issues can also be seen in Visual reality systems and they cause even more eye strain compared to large screen display systems. With mobile phones being used on large scale and bigger screen panel being manufactured nowadays for gaming the display issues have popped with them as well as the displays keep on getting better and the games keep getting CPU/GPU intensive. Also the need of the algorithm for handling these display issues should take care to provide more power saving or not hamper the currently applicable power savings on the mobile device. Bureau of Energy Efficiency Schedule [1] for Laptops/Notebooks specifies the energy labeling requirements for Desktops, Integrated Desktops and notebook/ laptop computers, manufactured, imported, or sold in India for household/office and similar use. The product sold by the vendors shall meet all the requirements specified in this schedule if they are to qualify for being energy efficient. This Standard was prepared on the basis of Energy Star specification for computers developed by US Environment Protection Agency. Alternatively for the BIS standard for laptops and notebooks this standard follows the version 6.11 - Energy Star program requirements Product specification for computers systems.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

II. LITERATURE SURVEY

ISSUES RELATED TO DISPLAY SYSTEM

1. TEARING

Screen tearing is a can be observed as an incomplete draw of previous frame onto a new present frame which cause the image production on screen to appear to have been cut into proportional to difference of frame rates) .

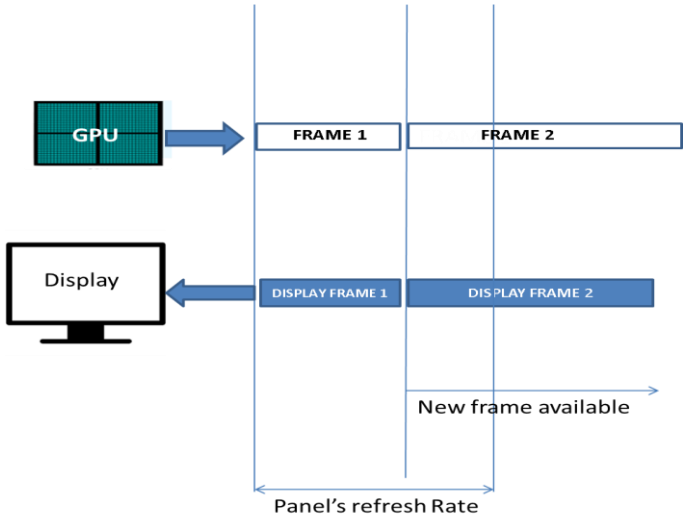

Fig 1 : If we display a frame whenever its available without accounting for the variable fps conditions of the application we are bound to see tearing.

It can be classified into two categories depending on how the issue is observed or reproduced. Classification is important because based on the application use case we need to take care of elimination of tearing in the display algorithm. This classification is based on the occurrence of the tear line on the screen.

1.1 Types of screen tearing:

1.1.1 Momentary tear line This effect occurs when the frames generated by the real time graphics application to the display device is not in sync with the display's refresh rate. This can occur due to miss matching refresh rates, hence in this case we observe that the tearing line moves as the phase difference changes (with speed proportional to difference of frame rates)

1.1.2 Fixed tear line

In fixed tear line the refresh rates are same but the synchronization point for them is faulty, in which case the tear line is at a fixed location that corresponds to the phase difference. In a video, fixed tear line screen tearing creates a stacked up lines like a staircaseon the edges of objects (such as a wall or a tree) as they failed to line up correctly.

III. PROPOSED METHODOLOGY FOR CAPTURING TEARING

1.2 Algorithm design to capture tearing

So from the above we see that tearing leads to incomplete display of an frame on the screen which overlaps with the next frame, so the algorithm to capture tearing must take into account or capture the time at which a new frame is completed its scan vs the time at which the previous frame was completely shown on the screen .If the scan out time of the next available frame is less than the time at which the complete image is shown on the screen then it is a frame tear.

1.2.1 Challenges to the tear detection algorithm and experiments conducted on simulated tearing environment

While it may seem simple to capture the scan out time from the previous , it may not be possible to record the exact time at which the frame was completely rendered on the screen .This is because of the manufacturing of the panel by different vendors and the type of technology used (OLED , LED-Backlit.etc). Even if were able to record there would be some delay associated with logging this time.

Hence, we conducted an experiment to find out the time at which a scan out completed for each frame in a simulated tearing environment. A plot was made of the scan out time (Y-axis) vs frame no. (X-axis) as seen in Fig 1.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

Scan out time is calculated as: Time at which the frame (Fi) is placed in the frame buffer minus time at which frame (Fi+1) was placed in the buffer. So if there is a tear we observed that the scan out time during the tear was very low which when divided by 1 and compared with the panel's current refresh rate indicates that the fps exceeds the current refresh rate. This gives us an indication that tearing has occurred.

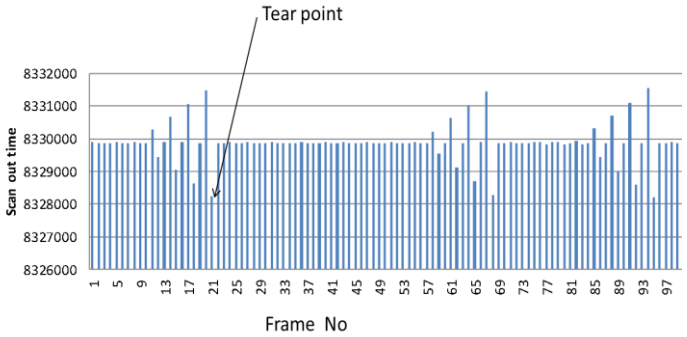

Fig 2: Depicts the scanout time of the frames for a 120 Hz refresh rate panel

1.3Techniques to avoid tearing

1.3.1 Double Buffering

The concept of double buffering uses the concept of buffer flips on every frame output cycle. The advantages of this technique are that we have the complete frame available for the next display output cycle.

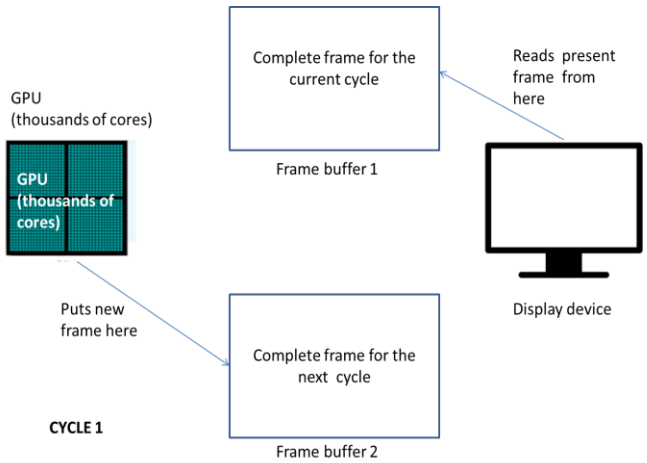

Fig 3: Depicts use of double buffering used to avoid tearing. This is system state before the buffer flip occurs.

One of the frame buffer stores the complete frame from the next cycle whereas one of them stores the complete frame for the current cycle. In the next cycle buffer flip happens and the GPU and display device point to the opposite buffer from the previous cycle.

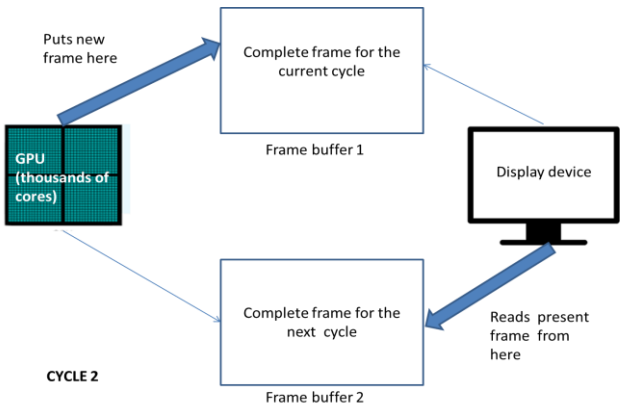

Fig 4: System state after the buffer flip has occurred.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

1.3.2Triple Buffering

The disadvantage of double buffering is that the algorithm has to wait for the buffer allocated to display in the current cycle to have completed outputting data to the display port , only then can it flip the buffer to make it available for the GPU to copy the data for the next cycle.

Hence it is advantageous to have another buffer to improve the performance.

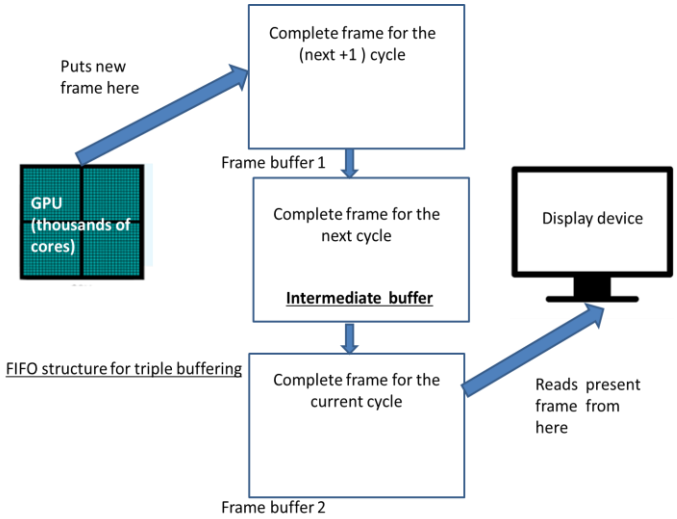

Fig 5: Use of triple buffering provides improved performance over double buffering.

1.3.3 Quad and Hex Buffering

This can be used in Visual Reality or 3D vision systems where the generated for the left and right eye are not exactly same and hence the display algorithm may accommodate using double or triple buffers for each eye, thereby using a total of four or six buffers respectively.

1.3.4 Vertical sync (V-Sync)

V-sync is a freely available frame synchronization mechanism wherein the display device will output the frame only after it has been completely made available by the GPU.

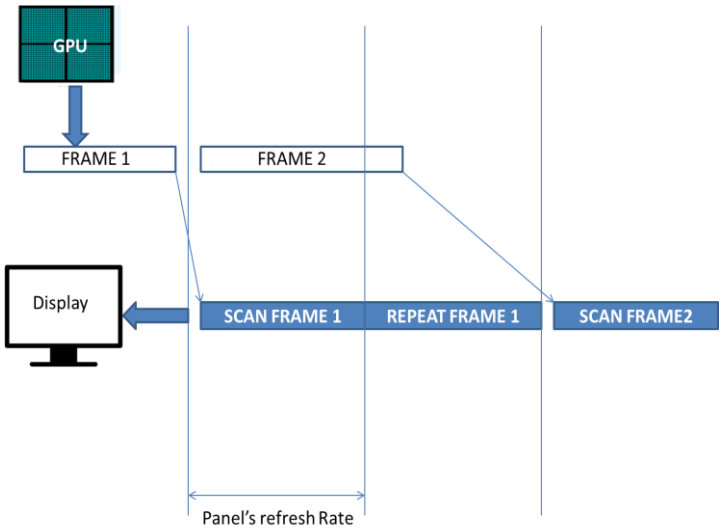

Fig 6: Working of V-sync in removal of tearing but nevertheless introduces lag.

1.3.5AMD's Free sync

Free sync is an AMD technology that extends the refresh rate of the panel based on the frame data generated by the GPU. This means Free-Sync works opposite to what V-Sync works which means it alters the rate at which a panel can show the frames rather that altering the GPU output to match with the panel's default refresh rate.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

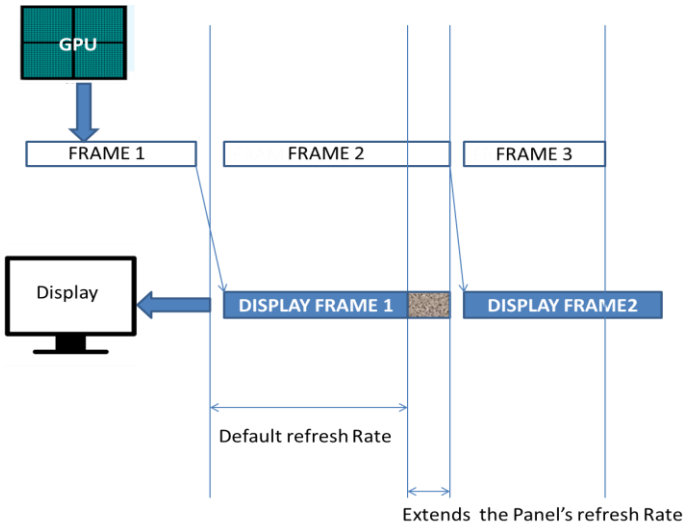

Fig 7: Working of other technologies which display a frame whenever its available without waiting for next refresh cycle.

1.3.5 NVidia's Gsync

Like free sync, g-sync also alters the panel's refresh rate rather that altering the GPU frame rendering rate. The difference between the free sync and g-sync is the method by which they alter the refresh rate.

2. STUTTER

Shuttering is a phenomenon seen in display technology due to irregularities in the frame rendering by the GPU which causes the frame scan out to be delayed/repeated resulting in to visual lag appearing on the screen. In an intensive rendering application like a 3d game where the frame rate is not constant , any rendering activity that takes more time from the GPU may cause the next frame to be delayed on to the screen because we had not accommodated for the changes in the frame rate and the panel kept scanning at a fixed rate.

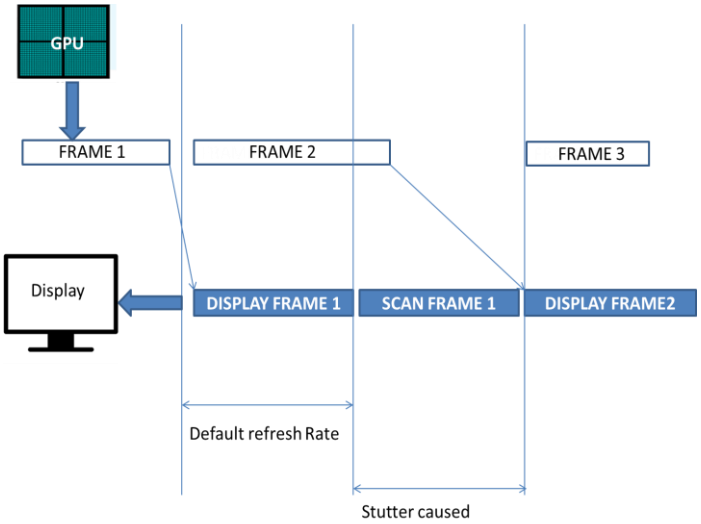

Fig 8: Depicts the cause of stutter as we are waiting for the panels refresh cycle to complete after which we can display the new frame.

We can describe the five components of the described system as follows:

2.1: Types of stutter

2.1.1 Single GPU stutter:

In a single configuration, we may see stutter whenever the frame scan out time is greater than a particular thresh hold value defined by the stutter detection algorithm.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

2.1.2 Multi –GPU stutter

Stutter issues are more prominent in multi GPU configurations wherein each frame is rendered by the allocated GPU in parallel and then synchronized in order to achieve smooth visual experience. This helps exploit parallelism as the GPU intensive processing is distributed between the GPU's.
If the allocation, processing and synchronization were to happen correctly then there would be no stutter observed.

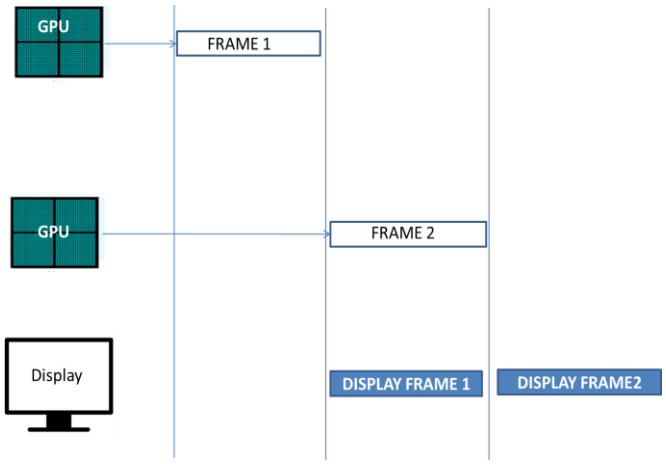

Fig 9: Stutter free output of frames generating from multiple GPUs can be achieved through correct synchronization.

However, not always do we see this kind of synchronized behavior occurring for all application. There could be several factors as listed below which determine this ideally synchronized behavior:
1CPU may be a bottle neck if it is not able to do the allocation of the frames for the GPU in time as it is currently processing other CPU intensive tasks already which will cause delay to the GPU in getting the request serviced and making it available to the frame buffer in time.
2. If either of the multi GPU configurations may be a bottle next then the frame buffer will receive the frame with delay and this will cause the output to stutter.

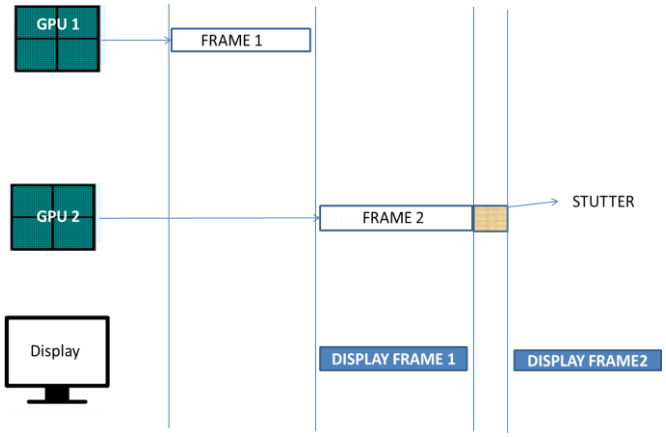

Fig 10: Here GPU2 is a bottleneck and it cannot provide the frames in time to match with the synchronization timings and hence introduces a stutter.

3.The game engine , multi GPU bugs related to the application may cause stutter as it would not be able to decide which of the GPU tasks need splitting up and which do not require.

IV. PROPOSED METHODOLOGY FOR CAPTURING STUTTERING

2.2 Algorithm design to capture stutters

Capturing stutters for single and multi GPU configurations is similar and it consists of tracking down the scan out time for the frames. Threshold is fixed to find out the frame is completed its scan vs the time at which the previous frame was completely shown on the screen .If the scan out time of the next available frame is less than the time at which the complete image is shown on the screen then it is a frame tear.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

2.2.1 Challenges to the stutter detection algorithm and experiments conducted

It is necessary to set the correct threshold in order to detect stutters in the given application. Keeping the value to small would log scan out times which are not stutters or in the acceptable range, as stutters. Keeping the threshold too big will log no/fewer stutters.

2.2.1.2 Finding the ideal threshold for a given use case

Below experiments were conducted to find the threshold for different use cases:

2.2.1.2.1 Fixed 50 fps video

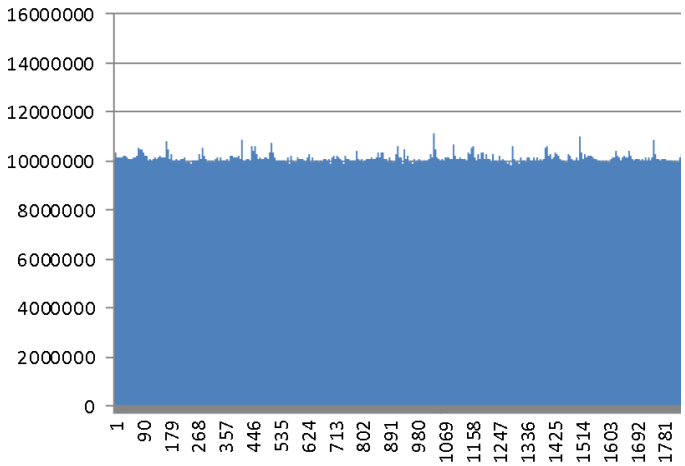

Fig 11a: Plot of frame no vs scan out time for 50 fps fixed video

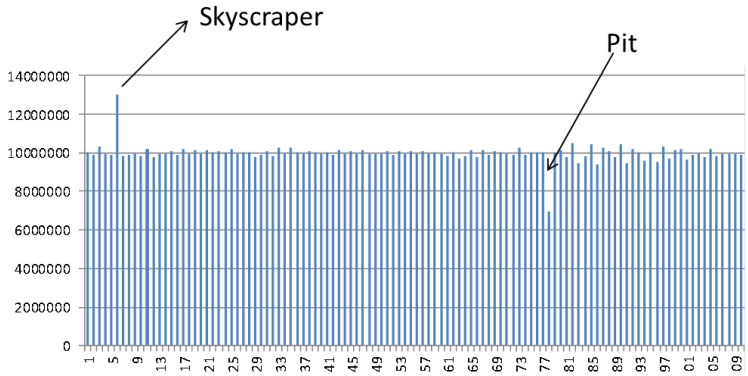

Fig 11b: Plot of frame no vs scan out time for 50 fps fixed video. We could observe stutters at specific points and they have been labelled as skyscraper (values greater than a threshold than the next and previous frame scan out times) and pits (values less than a threshold than the next and previous frame scan out times)

2.2.1.2.2 Variable fps (steady change): application pendulum

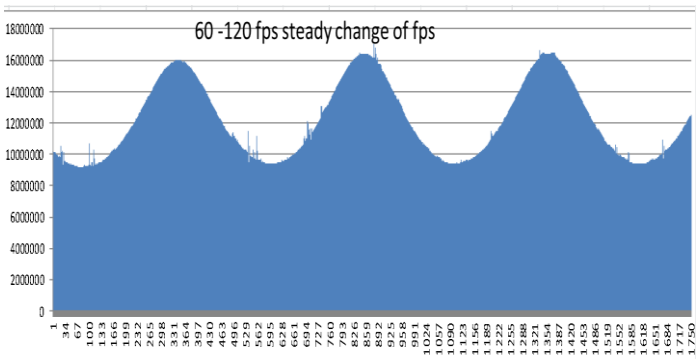

Fig 12a: Experiments were conducted on a stutter simulated environment with a steadily changing frame rate application and plot of the frame vs scan out time showed a regular wave pattern.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

Fig 12b: Calibrating stutter threshold:Zooming into the individual values and using breakpoints in code at stutter occurrence, it was found that setting the threshold at 20% was able to detect most of the actual visual stutters.

2.2.1.2.3 Variable fps (abrupt change):3DGame: Dota 2

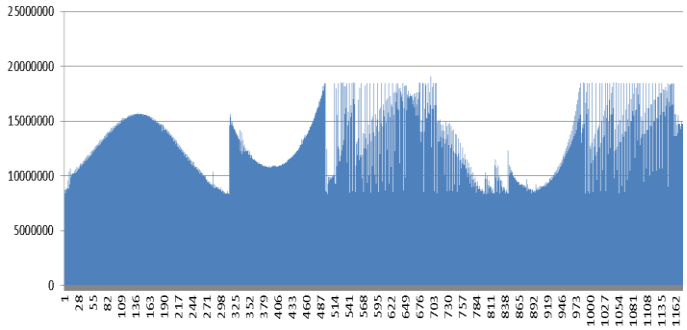

Fig 12b: We see that for the variable fps scenario the graph pattern is not regular so using skyscraper and pits concepts was not beneficial, so in such cases we need the application itself to maintain a fps counter so that we can compare the current fps with the scan out time. If difference between them beyond the desired threshold we cab mark them as stutter.

V. CONCLUSION

Most of the time it becomes difficult for the naked eye to capture stutters and tearing and it may differ from one person to another persistence of vision to detect them when they occur .For some it may appear smooth while for some there may have been a stutter or tear. Nevertheless, this paper tries to find out a method to capture stutters and tearing in gpu driven application mathematically. Itstarts by experimenting with the frame scan out times and finds the difference between consecutive frames. Then it experiments with various threshold values to come out with anideal threshold that would capture observable stutters based on the type of application use case.

REFERENCES

[1] https://beeindia.gov.in/sites/default/files/ctools/shcedule14com.pdf
[2] https://developer.nvidia.com/sites/default/files/akamai/gameworks/CN/Stuttering_Analysis_EN.pdf
[3] Design of graphics processing framework on FPGAS G Ramanathan; B Pradeep Kumar; C M Ananda; E P Jayakumar IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Year: 2016Pages: 387 - 391
[4] Research of an Intelligent Auto-Controlling System for LCD Screen Flicker Ying-Yun Chen; Ko-Wen Jwo; Rong-Seng Chang Journal of Display Technology Year: 2016, Volume: 12, Issue: 6 Pages: 557 - 561, DOI: 10.1109/JDT.2015.2506783
[5] http://www.prnewswire.com/news-releases/display-market-worth-16917-billion-usd-by-2022-613685423.html
[6] A Review of the Literature on Light Flicker: Ergonomics, Biological Attributes, Potential Health Effects, and Methods in Which Some LED Lighting May Introduce Flicker Arnold Wilkins;Brad Lehman, (2/26/10)-IEEE Standard P1789