Performance Evaluation of Terapixel Rendering in Cloud (Super) computing

Introduction

Terapixel images are images composed of 1 trillion pixels and can be used to present information to stakeholders intuitively, however, it requires a lot of computing power. So, it is advantageous to analyze the data collected from cloud supercomputing infrastructure to optimize the process and reduce the cost incurred.
The challenge that was addressed in this problem was, how massive scale supercomputer resources will be delivered for the computation of terapixel visualization of Newcastle Upon Tyne.
Our task here is to analyze the data collected from the application checkpoint and system metric output from the production of a terapixel image and improve the cost-efficiency.

Business Understanding

Which event types are taking most of the execution time?
By analyzing dominant events we can manage resources efficiently and hence reduce the cost of operation.

How GPU temperature and Performance are related?
This will allow us to switch resources efficiently, "GPU throttling" mainly happens because of inefficient heat management. So, if we observe some GPUs starting to throttle because of an increase in temperature we can switch them with GPUs of the same throughput but operating at lower temperatures.

Is there any impact of increased power draw on render time?
With this, we can quantify whether the increase in power draw has any impact on render time. If yes, then how much improvement are we getting? If not, what can be done to reduce the power consumption.

Is there any performance difference in GPU cards?
If different GPUs are performing at different throughputs, we may well be able to allocate high-end GPUs for more computation-intensive tasks.

Is there a specific pattern in which tiles/pixels are consuming more or less power?
If we can figure out if some specific regions on the terapixel image are taking more compute power, we can allocate flagship GPUs for those specific pixels and mid to low-end GPUs for the rest of the pixels. This will improve the rendering of the image and resource management.

Data Understanding

application-checkpoint.csv: This file contains details of what type of events are happening at what timestamp and their hostname, taskId, jobId.

- **timestamp:** The exact time at which the event started or stopped.
- **hostname:** There are 1024 unique hostnames, each GPU can be considered as a host.
- **eventName:** There are 5 events, Tiling, Render, Saving Config, Uploading, and TotalRender.
    - **TotalRender:** It is the complete task.
    - **Render:** It is when tiles of the image are being rendered.
    - **Tiling:** It is when the post-processing of rendered tile is done.
    - **Saving Config:** Each task has some specific configurations, which are being saved.
    - **Uploading:** Output of post-processed tiles are uploaded to Azure Blob Storage.
- **eventType:** START or STOP, this shows whether the event is getting started or being terminated.
- **jobId:** There are 3 jobIds for Azure batch jobs. For level 12, level 8, and level 4.
- **taskId:** Id of the Azure batch task. There's a unique ID for each set of all 5 eventNames along with their start and stop timestamps. For example, if there are n rows in the application-checkpoint.csv there will be n/10 unique taskIds. 5 events and their start and stop rows = 5*2 = 10.

gpu.csv: This file consists of GPU conditions at a particular instance of time.

- **gpuSerial:** There are 1024 GPUs and each has a unique serial number.
- **gpuUUID:** The unique system id assigned by the Azure system to the GPU unit.
- **powerDrawWatt:** Power drawn by the GPU in Watt.
- **gpuTempC:** Temperature of the GPU in Celsius.
- **gpuUtilPerc:** Percentage of cores of GPU utilized.
- **gpuMemUtilPerc:** Percentage of memory of GPU utilized.

task-x-y.csv: This file has x and y coordinates of the image tiles being rendered.

- **x:** x coordinates of the image tiles.
- **y:** y coordinates of the image tiles.
- **level:** The visualization created is a zoomable "google maps style" map. In total, we create 12 levels. Level 1 is zoomed right out and level 12 is zoomed right in. You will only see levels 4, 8, and 12 in the data as the intermediate levels are derived in the tiling process.

Data Processing

Merging 3 raw data files into one.

- First left join was performed on task-x-y.csv with application-checkpoint.csv on 'jobId' and 'taskId'.

- Then inner join on merged data from the previous step and gpu.csv using 'timestamp'.

Calculation of time duration required by each event.

- Extracted only time component from the timestamp column, as whole data was collected on the same date i.e., 2018-11-08.
- Now grouped final merged data (from the previous step) on eventName and eventType.
- Made different data frames for each event with their start and stop times separately. Hence, now we have 10 separate data frames. 5x2=10; 5: events(*TotalRender, Render, Tiling, Saving Config, Uploading*), 2: eventType(*Start and Stop*)
- Merged start and stop data frames of respective events on taskId. Now we have 5 data frames, 1 for each event.
- Subtracted start time from stop time and stored it in a separate column named "duration".
- Finally, stacked all the 5 event's data frames into 1.

Important Assumption(s) and Consideration(s)

- In the market when we look for which GPU or processor is better than the other, to measure their performance we look into their peak power draw. The more power is drawn, the more will be the performance. This fact is taken into consideration while comparing the performance of the GPU with its other conditions like GPU temperature, GPU memory utilization, GPU utilization.
- We have merged the Application checkpoint and GPU data based on timestamp which can not be done in all the cases however, for the analysis we are carrying out, we are taking the average of most of the features so that we can smoothen the data and don’t let the errors affect our results.

Data Analysis

Size of each data frame:

- Application Checkpoint data has 660400 rows and 6 columns.
- GPU has 1543681 rows and 8 columns.
- task-x-y has 65793 rows and 5 columns.
- Merged Data has 86633 rows and 16 columns.

Important statistics of GPU data:

	powerDrawWatt	gpuTempC	gpuUtilPerc	gpuMemUtilPerc
count	1543681	1543681	1543681	1543681
mean	89.18	40	63.05	33.41
std	39.75	3.8	41.44	23
min	22.50	26	0	0
25%	49.99	38	0	0
50%	96.59	40	89	43
75%	121.34	42	92	51
max	197.01	55	100	83

Fig 1: Pair Plot of GPU Conditions

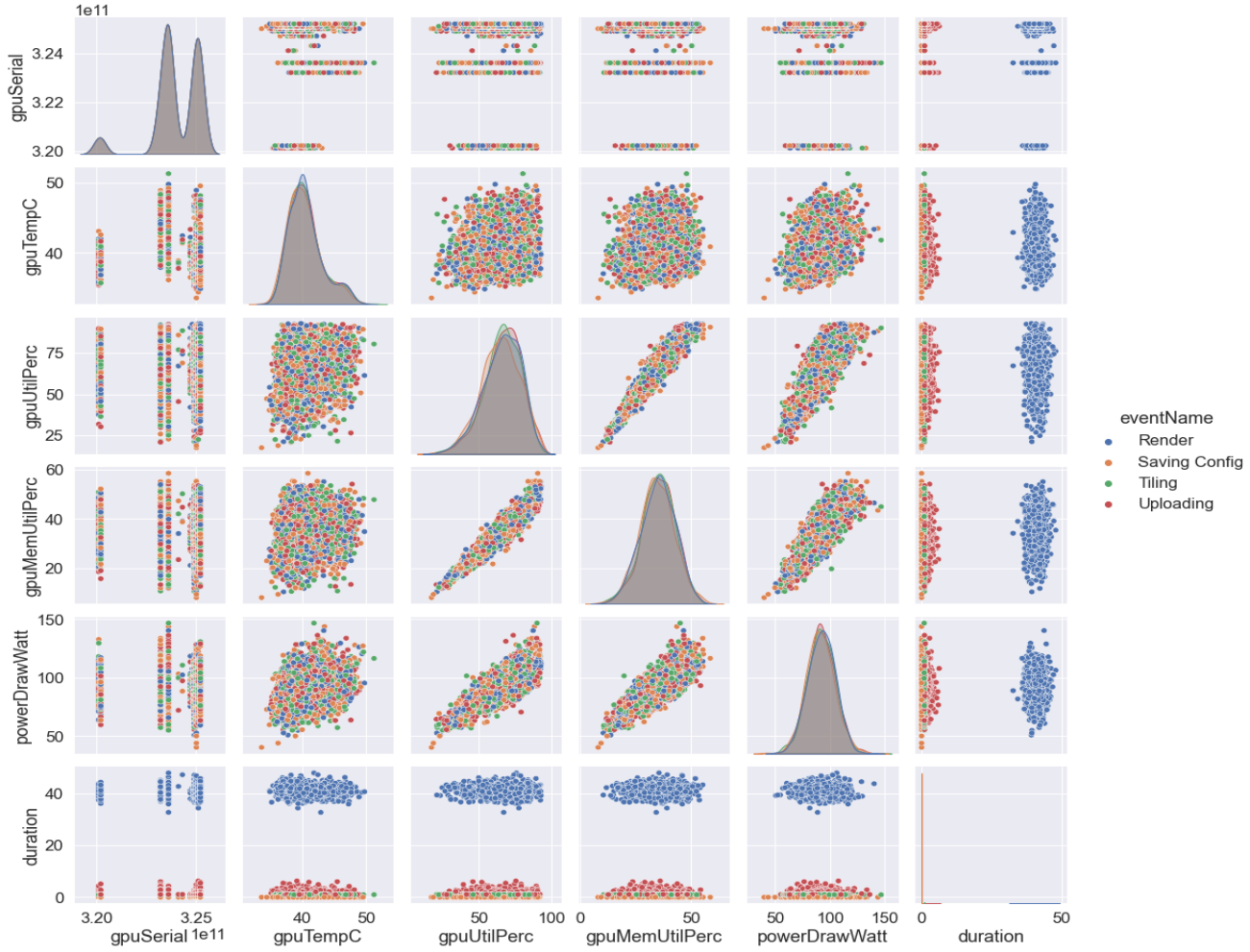

Fig 1: Pair Plot of GPU Conditions.

From Fig 1, we can get the following insights:

- The duration of the Render event is significantly more than other events.
- GPU temperature does not have any specific relationship with Power draw, GPU memory utilization, GPU Utilisation percentage.
- GPU Utilisation percentage is very highly positively correlated to power draw and GPU memory utilization.
- GPU memory utilization has a positive high correlation with power drawn by the GPU.

Events that are dominating the runtime.

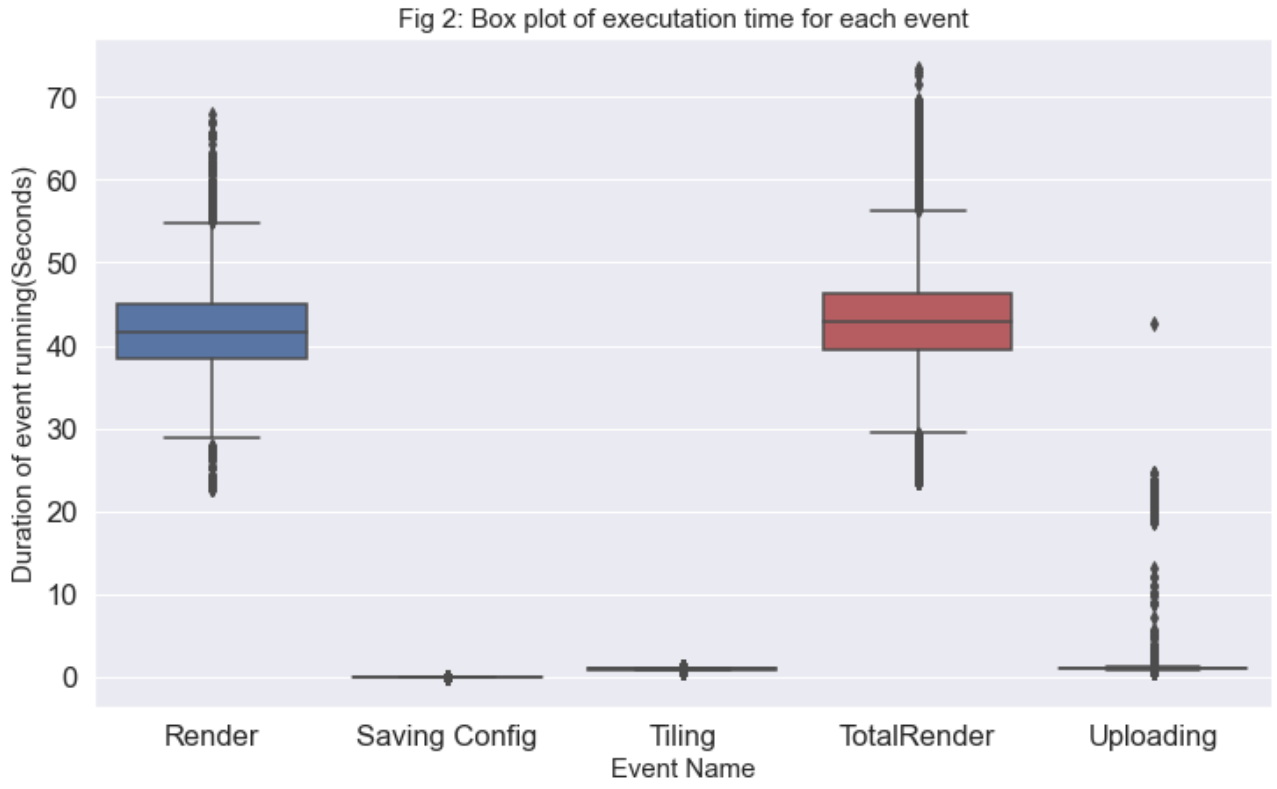

Fig 2: Box plot of execution time for each event.

Event Name	Total Time Taken
Tiling	17.986
Saving Config	0.046
Render	761.960
TotalRender	787.769
Uploading	25.755

From Fig 2 and the table above, we can observe the following key points:

- Event Render is taking most of the processing time as we can see it is almost equivalent to event TotalRender, however, we have a considerable amount of outliers in both Render and TotalRender.
- On average Rendering takes 42 seconds and the interquartile range is from 38 to 45 approx.
- Saving Config, Tiling, and Uploading doesn't take much computation time.
- Uploading event exhibits a significant number of outliers.

The interplay between performance and temperature.

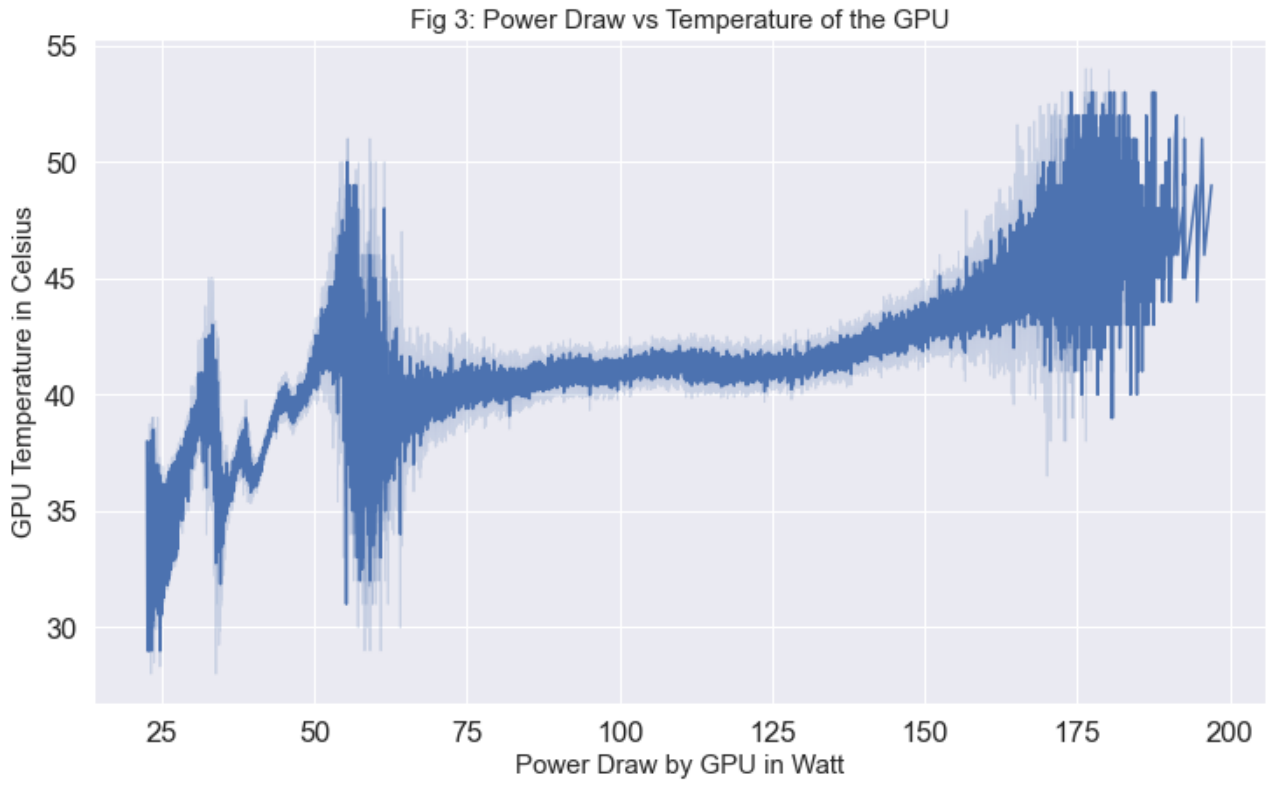

Fig 3: Power Draw vs Temperature of the GPU.

From Fig 3, we can perceive the following pointers:

- With the increase in power draw(Performance) there is an increase in temperature as well.
- We can also observe that at around 60 watts we have huge fluctuations in the temperatures, so we can say that some low-performing GPUs are reaching higher temperatures even at lower power draw.
- Even though temperatures are increasing with the increase in performance it is not going absurdly high. Hence, in general, we can say that cooling infrastructure is good. However, this doesn't tell the full story so, we will further

look into variation in GPU utilization and GPU memory utilization as opposed to power draw.

To support the findings from Fig 3 we also plotted Fig 4 and below are findings:

- With the increase in power draw (performance) we can see both GPU utilization and GPU memory utilization is also increasing which makes sense because to increase the performance of GPU and its components' utilization will increase. Hence we can say that with the increase in power draw there is an increase in performance.

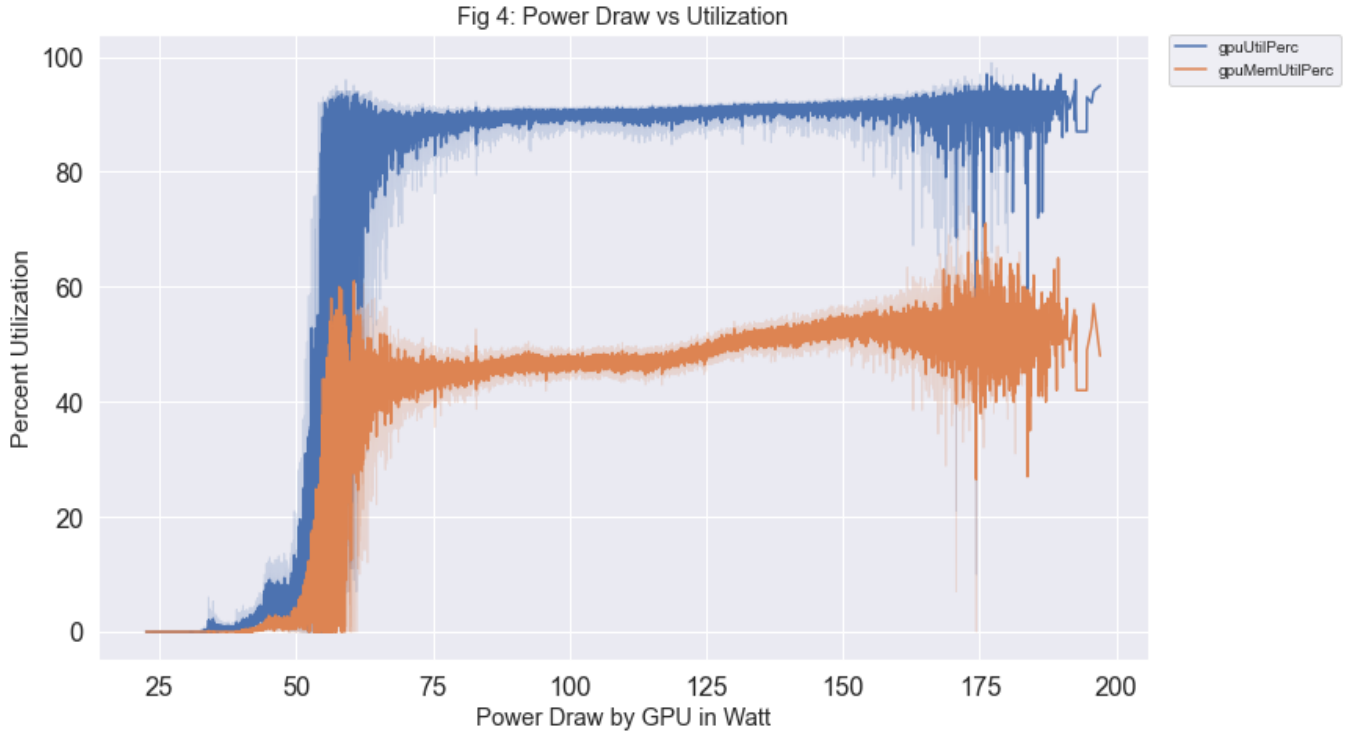

Fig 4: Power Draw vs Utilisation

- However, performance gain after around 60 watts is very marginal, furthermore, at higher power draw (160+ watts) we can see drop/fluctuation in GPU utilization as well as GPU memory utilization which is a clear sign of GPU throttling.
- Also, in the previous plot (Fig 3) we observed at around 60 watts there is a huge fluctuation in temperatures, so, it may not entirely be because of weaker GPUs but also because of the temperature hindering the clock speeds. We will further investigate this by plotting power draw against GPU serial numbers to see whether some GPUs are more powerful than others or not.

Performance of GPUs based on their serial number.

gpuSerial	powerDrawWatt
325117171574	80.510313
325017017790	80.742673
325017018552	81.087075
325017049295	81.226755

Table: Low Performing GPUs

gpuSerial	powerDrawWatt
323617042596	106.247462
323617042596	101.974324
323217056368	101.549633
325017049041	99.057575

Table: High Performing GPUs

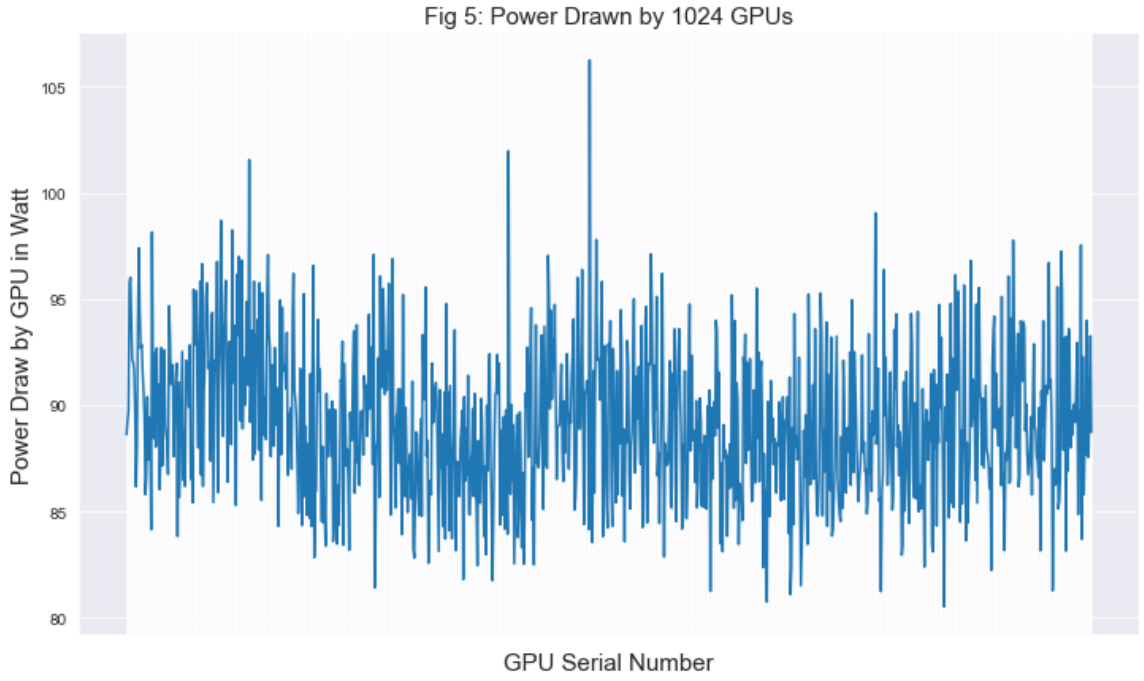

Fig 5: Power Drawn by 1024 GPUs.

From Fig 5 and two tables, we can take out the following points:

- Although there are few high-performing GPUs, there isn't any set pattern of their performance based on their serial number.
- From the above two tables, we can see some of the serial numbers of high-performing as well as low-performing GPUs.
- On average GPU with serial number **323617042596** tends to perform better than others.

The interplay between render time and power draw.

From Fig 6, 7, and 8 we can infer the following key points:

- From fig 6: there isn't any improvement in render time when power drawn by the GPUs increases.

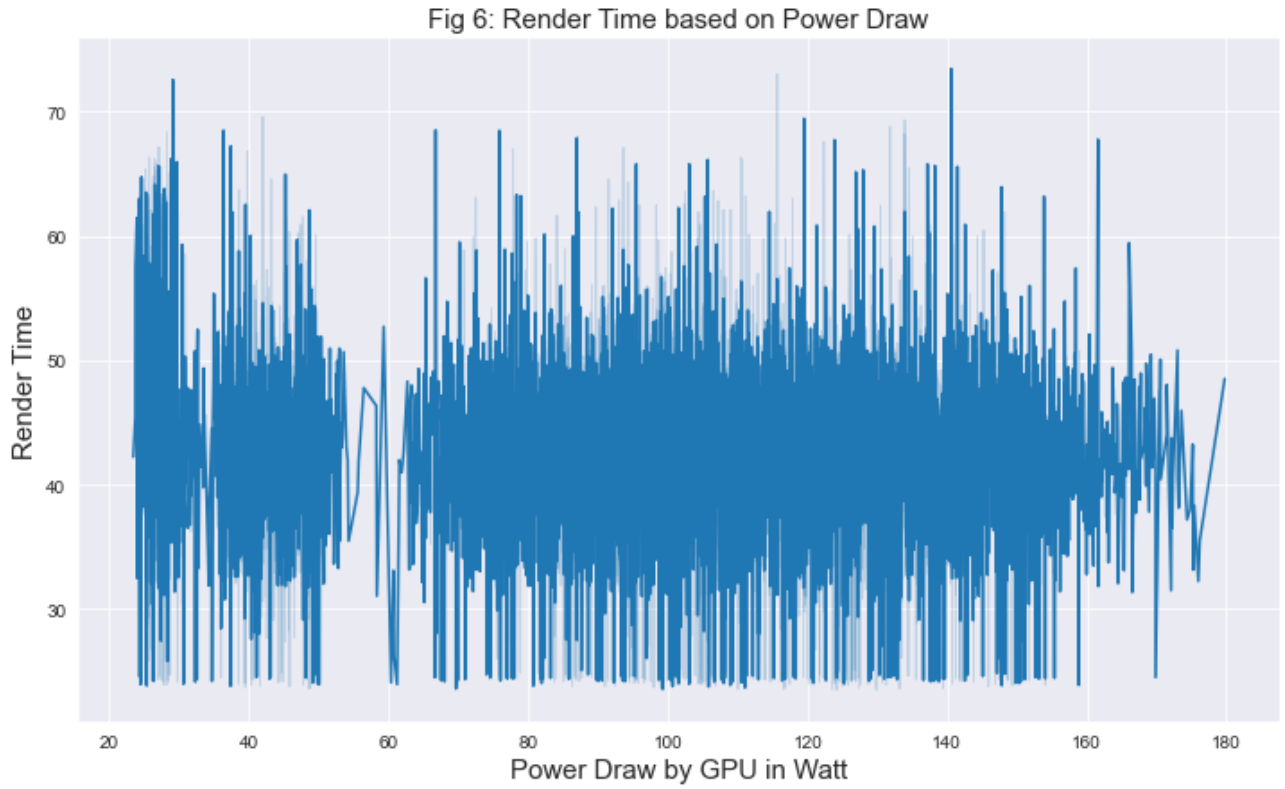

Fig 6: Render Time based on Power Draw.

- From fig 7: As the temperature is increasing we can see power draw is increasing and render time is decreasing hence, we can say for certain that cooling infrastructure is good and heating or increase in temperature does not affect the performance much.
- From fig 8: render time is decreasing drastically when the temperature is approaching 35 degrees celsius and then it is fairly consistent afterwards and not improving much.

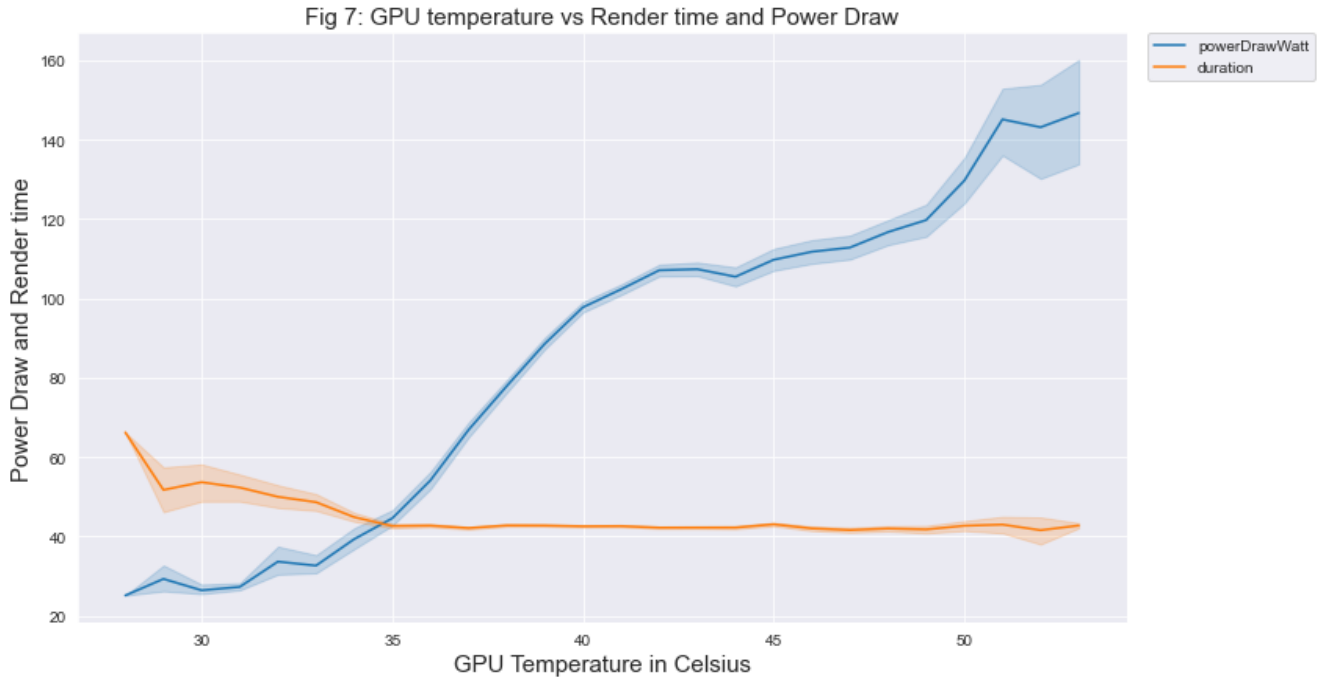

Fig 7: GPU temperature vs Render time and Power Drawn.

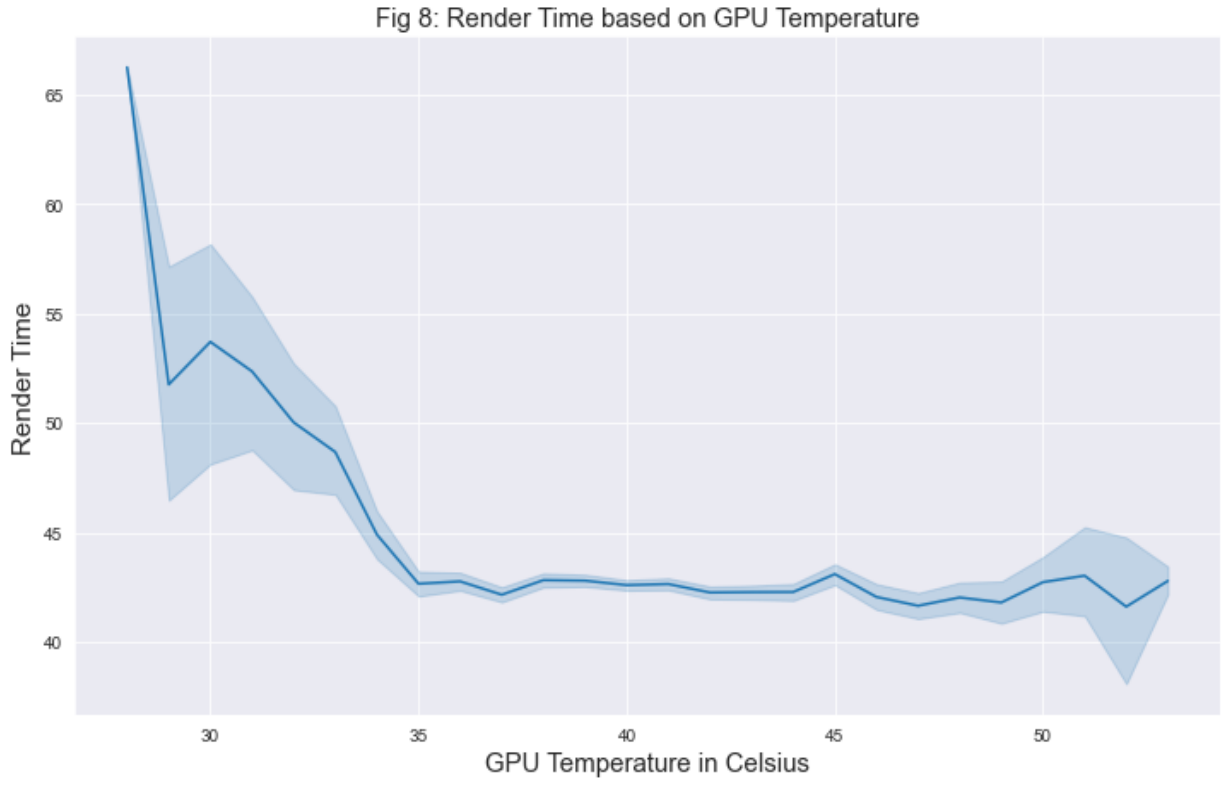

Fig 8: Render Time based on GPU Temperature.

- Overall, there is a significant change in the power draw but not much improvement in the render time whereas, with the increase in temperature we can see some improvement in render time.

Variation in computation requirements for particular tiles.

From Fig 9, we can observe the following pointer(s):

- There isn't much set pattern of tiles taking more or less time to render. However, we can see darker/greenish(less rendering time) pixels near legend and USB and brighter pixels on the left edge and near St James' Park which makes sense because the area near USB doesn't have much detailing whereas the area near St James' Park has many buildings and streets which is taking time to render.

Fig 9: Heatmap of Newcastle Upon Tyne

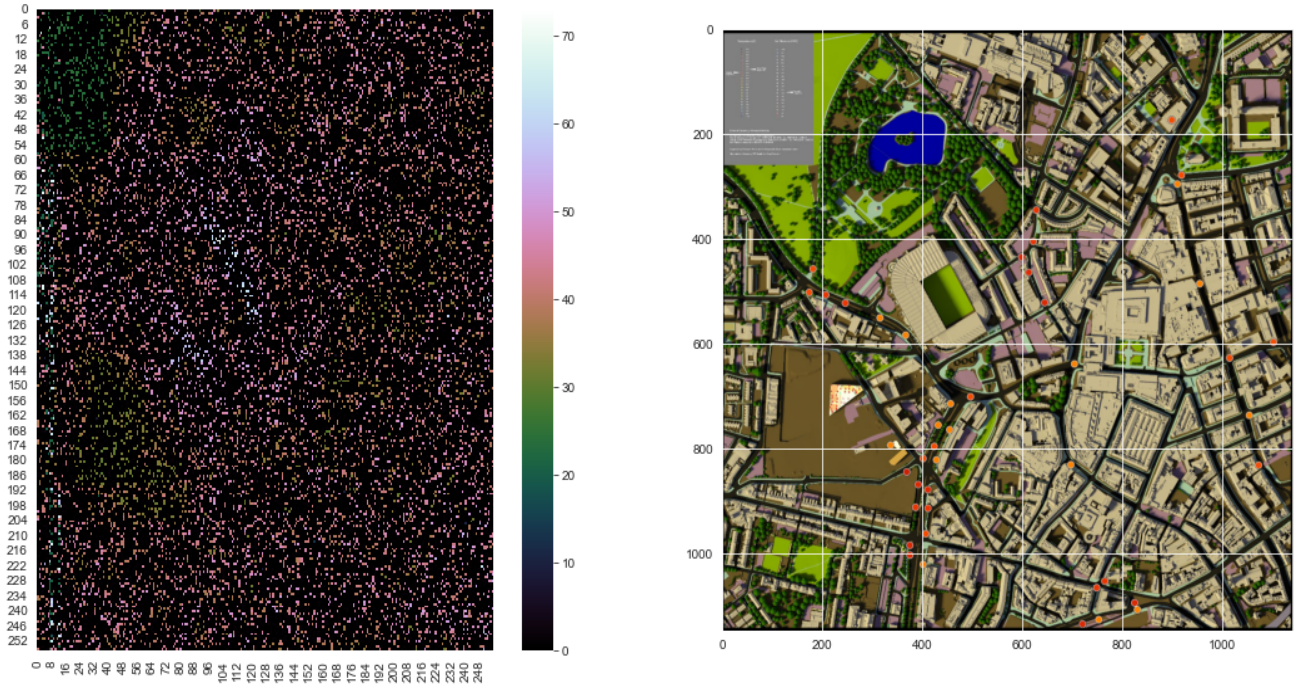

Fig 9: Heatmap of the Newcastle Upon Tyne.

Conclusions and Future Scope

- Most of the computation time is being used by Rendering events.
- With the increase in power draw(performance) GPU temperature is also increasing as more and more GPU and its components will be put to work.
- There isn't much improvement in the render time when power draw is increased however, with the increase in GPU temperature we observed improvement in render time. This can further be analyzed as to what are the reasons that are not letting render time improve with power draw whereas it is improving with the increase in GPU temperature.
- There are 1024 GPUs and there is a very slight difference in their performance and very few are significantly better than others.

- Tiles associated with streets and buildings(crowded areas) are taking more time to render as compared to tiles with empty spaces. So, the allocation of those high-performing GPUs in crowded areas will be more efficient.

References

- Video to Demonstrate the application of terapixel visualization
- Visualizing Urban IoT Using Cloud Supercomputing by Nicolas Holliman, Manu Antony, Stephen Dowsland & Mark Turner
- Data Smoothing Technique (rafalab.github.io)
- Professor Nick Holliman, Manu Antony, Stephen Dowsland, Professor Philip James, Mark Turner, "Petascale Cloud Supercomputing for Terapixel Visualisation of a Digital Twin" in arXiv, Online Publication 2019.