BIG DATA AND COMPUTER GRAPHICS

Irene Maria Gironacci

SUMMARY

1. INTRODUCTION

We've entered a data-driven era, in which data are continuously acquired for a variety of purposes. The ability to make timely decisions based on available data is crucial to business success, clinical treatments, cyber and national security, and so on. However, most data have become simply too large and often have too short a lifespan. Almost all fields of study and practice eventually will confront this big-data problem.

2. WHAT IS BIG DATA

Big data is the term for a collection of datasets to large and complex that it becomes difficult to process using on-hand database management tools or traditional data processing applications. The challenges include capture, curation, storage, search, sharing, transfer, analysis and visualization.

3. IMPACT OF BIG DATA ON COMPUTER GRAPHICS

Visualization is the process of representing data graphically and interacting with these representations in order to gain insight into the data. An important goal of visualization technology is to support the exploration and analysis of very large amounts of data.
Big-data analytics and discovery present new research opportunities to the computer graphics and visualization community. Computer graphics has provided a powerful mechanism for creating, manipulating, and interacting with these representations.
Furthermore, a growing number of customers are using GPUs for big data analytics to make better, real-time business decisions or for computing real-time data for computer graphics.

In *"Customizing Computational Methods for Visual Analytics with Big Data "*[1] is shown the interplay between precision and convergence—two aspects that haven't received appropriate consideration in visual analyses so far. The authors propose customizing computational methods to include low-precision computation and iteration-level visualizations to ensure real-time visual analytics for big data.

In *"Visualizing Large, Heterogeneous Data in Hybrid-Reality Environments"* [2], Khairi Reda shows how a new kind of visualization space called *hybrid-reality environments* can achieve scalable visualization of heterogeneous datasets. These environments synergize the capabilities of VR and high-resolution tiled LCD walls, letting users juxtapose 2D and 3D datasets and create hybrid 2D-3D information spaces. The authors introduce two such environments—Cyber-Commons and CAVE2—and some real-world applications.

4. RELATIONS BETWEEN BIG DATA AND COMPUTER GRAPHICS

Sources of big data in computer graphics are basically:

- range scanning
- imagery
- sensor networks
- scientific computing
- light transport
- databases
- web

The definition of "big" always varies with the application, but similar issues arise again and again, such as bandwidth constraints, latency requirements, scalability limits, parallel load balancing, visual fidelity, and output device characteristics. These issues are particularly critical when it is important to build a system that is interactive. Some solutions of these problems are: parallelism, specific algorithms, bigger displays and ad hoc architectures.

4.1 PARALLELISM

In computer graphics, rendering is the process by which an abstract description of a scene is converted to an image. When the scene is complex, or when high-quality images or high frame rates are required, the rendering process becomes computationally demanding. To provide the necessary levels of performance, parallel computing techniques must be brought to bear. In Figure 4.1.1a and Figure 4.1.2b we can see a graphics pipeline: sequence of steps used to create a 3D scene in a parallelism context. An article describing in detail the graphic pipeline is [5].
Recent graphics accelerators outstrip the ability of the host interface to supply parallel graphics with data (e.g. INVIDIA's GeForce256).
Several different types of parallelism can be applied in the rendering process. These include functional parallelism, data parallelism, and temporal parallelism. Let's focus on data parallelism: instead of performing a sequence of rendering functions on a single data stream, it may be preferable to split the data into multiple streams and operate on several items simultaneously by replicating a number of identical rendering units.

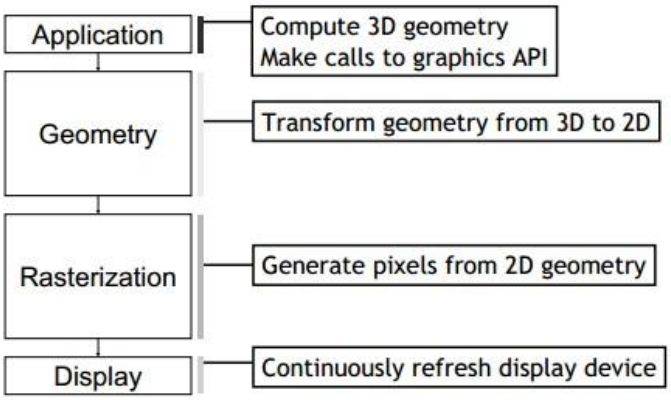

Figure 4.1.1a Graphics pipeline

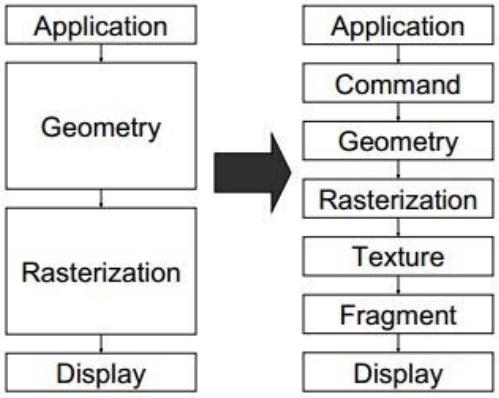

Figure 4.1.1b Graphics pipeline

4.2 ALGORITHMS

As non-shared memory multiple instruction, multiple data systems become more common, it becomes important to develop parallel rendering algorithms for theme. These systems, known as multicomputers, can produce data sets so large that is difficult to visualize the data on conventional graphics systems, especially if the visualization proceeds in tandem with the calculation. Parallel systems must run interactive graphics to allow convenient visualizations of their computations.
Main parallel algorithms used for these purposes are based on: horizontal or vertical strips, rectangular areas, load balancing (between regions, stages or frames), sort-first architecture, and so on.
For example, an algorithm suited for interactive polygon rendering is described in article [3], where the model's image on screen generally has frame-to-frame coherence. This algorithm uses this coherence to perform load-balancing calculations in parallel with the other calculations. The algorithm also uses an optimized version of personalized all-to-all communication, where all processors communicate with all other processors.
An overview of parallel composing techniques on shared memory architectures lie on reference [7].

4.3 VISUALIZATION

The demands for better interactivity and realism in applications such as vehicle simulation, architectural walkthrough, computer-aided design and scientific visualization have continually been driving forces for increasing the graphics performance available from high-end graphics systems. Interactivity implies that the images are drawn in real-time in rapid response to user input. This immediately brings out two requirements from the graphic systems: it must be able to draw images at approximately 30 frames per second (real-time), and it must have low latency (rapid response). Realism implies that the images are rendered from detailed scene descriptions, meaning that the scenes consists of many thousands of graphics primitives. Realism also requires a display system that can show the scenes with a level of detail matching what the eye can see. Providing such detail for a reasonable field of view requires millions of pixels. For furthermore details about high-performance polygon rendering see the reference [6].

4.4 ARCHITECTURES AND SCALABILITY

Interactive graphics applications have long been challenging graphic systems designers by demanding machines that can provide ever increasing polygon rendering performance. Another trend in interactive graphics is the growing use of display devices with pixel counts well beyond what is usually considered "high resolution". If we examine the architectural space of high-performance rendering systems, we discover only one architectural class that promises to deliver high polygon performance with very-high resolution displays: this architectural class is known as "sort-first". The algorithm sort first is described in detail in the article [4].

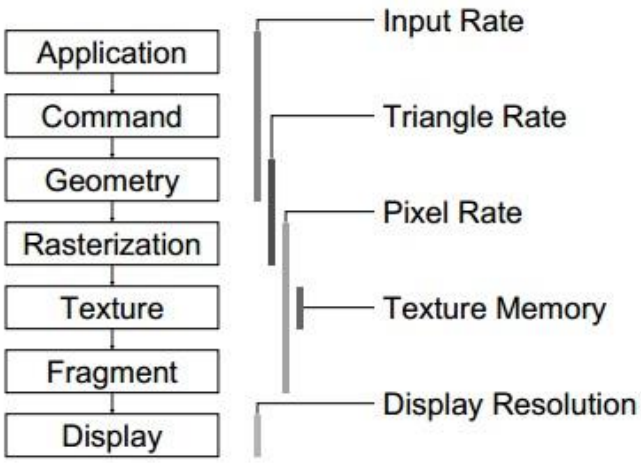

Figure 4.4.1 measuring performance

A fully scalable graphics architecture should provide scalability on the five key metrics : input rate, triangle rate, rasterization rate, texture memory capacity and display bandwidth (Figure 4.1, Figure 4.2).
An example of a parallel hardware architecture for polygon rendering that provides all of this is *Pomegranade.* Pomegranade uses the network to load balance triangle and fragment work independently, to provide a shared texture memory and to provide a scalable display system. Pomegranade operates at 87-99% parallel efficiency with 64 pipelines, for a simulated performance of up to 1.10 billion triangles per second and 21.8 billion pixels per second.

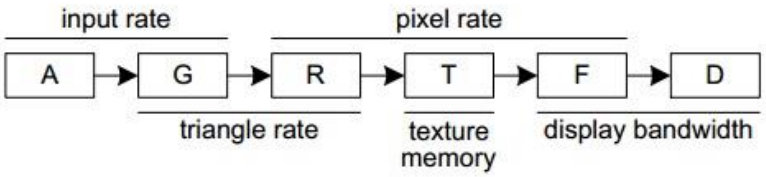

Figure 4.4.2 the serial graphics pipeline of an application (A), a geometry processor (G), a rasterizer (R), a texture processor (T), a fragment processor (F) and a display processor (D).

5. REFERENCES

[1] Jaegul Choo, Haesun Park, "Customizing Computational Methods for Visual Analytics with Big Data", in IEEE Computer Graphics and Applications, by IEEE Computer Society, July/August 2013

[2] K. Reda et all, "Visualizing Large, Heterogeneous Data in Hybrid-Reality Environments," IEEE Computer Graphics and Applications, vol. 33, no. 4, pp. 38-48, July-Aug. 2013

[3] D. A. Ellsworth, A New Algorithm for Interactive Graphics on Multicomputers, Journal IEEE Computer Graphics and Applications archive Volume 14 Issue 4, IEEE Computer Society Press Los Alamitos, CA, USA

[4] C. Mueller, Hierarchical graphics databases in sort-first, PRS '97 Proceedings of the IEEE symposium on Parallel rendering. Pages 49-f, 1997

[5] S. Molnar, M. Cox, D. Ellsworth, H. Fuchs, A sorting classification of parallel rendering, Journal IEEE Computer Graphics and Applications archive, Volume 14 Issue 4, July 1994, Page 23-32

[6] Computer Graphics n.4, August 1988

[7] E. Reinhard, C. Hansen, A comparison of parallel compositing techniques on shared memory architectures, In Proceedings of the Third Eurographics Workshop on Parallel Graphics and Visualisation, 2000