Paper Evaluation: GPU Computing

1. Paper Title, Authors, and Affiliations

Title: *GPU Computing*

Authors: John D. Owens, Mike Houston, David Luebke, Simon Green, John E. Stone, and James C. Phillips

Affiliations: University of California, Davis; Stanford University; NVIDIA Corporation, the University of Illinois at Urbana Champaign

2. Main Contribution

The main contribution of this paper is to clearly explain how GPUs have evolved from specialized hardware for graphics, that mainly serve rendering into programmable processors that can be used for general purpose computing across many fields. The authors show that GPUs are designed around massive parallelism and high throughput rather than low latency, which makes them especially effective for workloads that process large amounts of data at once. By introducing the GPU computing models and demonstrating successful applications in areas such as game physics and molecular biology, the paper argues that GPUs are not just graphics devices but an important platform for high performance computing.

3. Outline of the Major Topics

The paper begins by comparing GPUs with CPUs and highlighting key differences in architecture, especially the GPU's higher arithmetic throughput and memory bandwidth. It then explains the stream computing model, where many data elements are processed in parallel using the same program. After that, the authors describe the evolution of GPU hardware, including programmable shaders, unified architectures, and the shift toward general purpose programmability. The paper also reviews software tools such as CUDA and other frameworks that make GPU programming easier. Finally, several case studies are presented, including Havok FX for game physics and scientific applications like Folding@Home and molecular dynamics simulations.

4. One Thing I Liked

One part I found especially interesting was the discussion of the gap between processor speed and human perception. The authors explain that computers operate in nanoseconds while humans perceive changes in milliseconds, which means small delays usually do not matter visually. This helps explain why GPUs are designed to focus on throughput rather

than latency. I found it interesting how human perception and experience in the real world could affect hardware design.

5. What I Did Not Like

A minor issue is that the paper introduces many different GPU programming models and tools in a relatively short space. Although the authors do a good job explaining the transition from earlier systems like Brook to more modern approaches such as CUDA, the number of frameworks discussed at once can still feel a bit dense for new readers. It requires extra effort to keep track of how these systems relate to each other. A brief summary or comparison table might make this progression even clearer.

6. Questions for the Authors

1. Are there specific types of problems that still do not benefit much from GPUs, and how should developers recognize these cases in advance?
2. Even though tools like CUDA make GPU programming easier than before, it can still be hard to learn. How accessible is GPU computing for people without a graphics background? Could this become an obstacle to using GPUs more widely in other fields? Are there possibilities to improve?