

/technology

GPU: brute processing power usable for more than just images

The GPU (Graphics Processing Unit) in a modern PC is a monster number cruncher. When producing a smooth flow of images, the GPU performs more calculations per second than the processor (CPU). But in practice, using GPUs for other applications is not so simple.

Driven by developments in the 3D gaming market, very fast graphics processors have been developed. In the first graphics applications, the PC's CPU calculated the image to be reproduced pixel by pixel. The image was then sent to the video card pixel by pixel, which in turn reproduced only the pixels. This is a highly processing-intensive task, certainly if this example needs to be performed for a 3D game at 30 frames per second with a resolution of 1920 x 1200.

In the nineties, ever more powerful video cards came onto the consumer market and the video card acquired its own processor, the GPU. Instead of directing the video card pixel by pixel, objects and settings are delivered allowing this separate processor to work independently alongside the CPU.

Standardisation

In 3D-gaming, a consumer market, very high volumes of these processors are sold. Because the producers of the GPUs and the games are different companies, the need arose for standardisation of the interfaces (APIs). The most important 3D interface standards are DirectX/Direct3D and OpenGL. What it comes down to in practice is that the GPU producers strictly follow the specified functionality of these 3D-interfaces (APIs). ATI (now AMD) and NVIDIA are well-known producers of GPUs.

The core of 3D games is the depiction of a virtual 3D world on the screen from the perspective of a (virtual) player, at a speed of for example 30 frames per second. This is achieved by performing a

whole series of operations in a fixed order. This process is called a rendering pipeline.

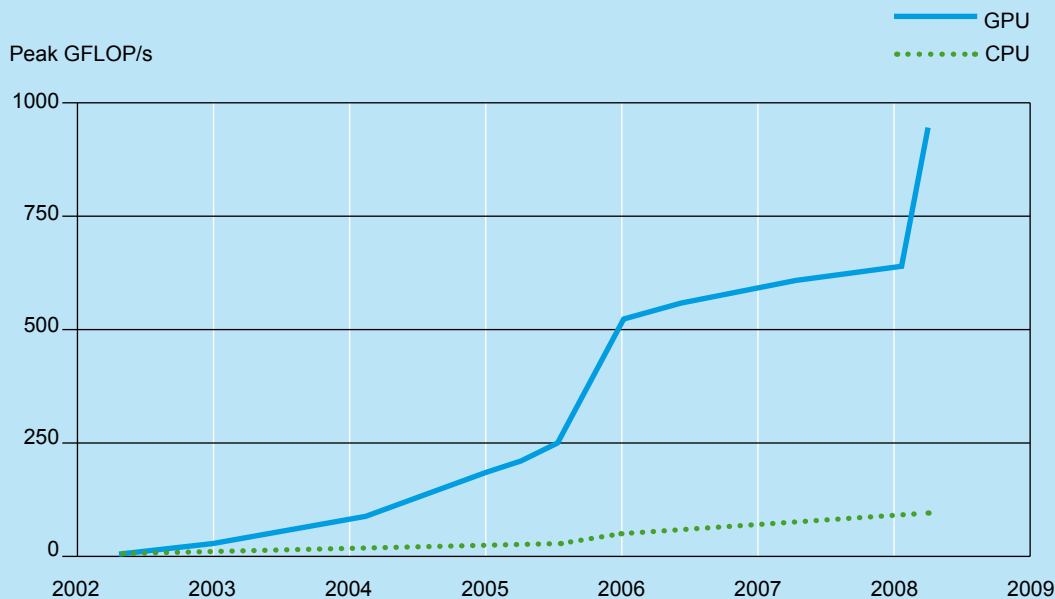
History

As video cards became ever more powerful, an ever increasing proportion of the rendering pipeline was executed in hardware. The processing steps in the rendering pipeline are the so-called shaders. For example, a vertex shader adds particular 3D effects to objects, a geometry shader generates new objects based on previously defined objects and a pixel shader calculates the colour of a pixel. An important drawback was that by defining specific shaders in hardware, the calculating power of each shader was also fixed. Yet different games place different demands on the distribution of processing power between the different shaders.

Relatively recently, as part of the DirectX-standard, a Unified Shader Model has been specified. This is a processor model equipped with an instruction set that is suited to the functionality of all the different types of shaders. The advantage of this concept is that the full processing power can be deployed and is programmable at each step of the rendering pipeline. This new type of graphics processor also offers the first practical prospect of using the GPU for different processing tasks.

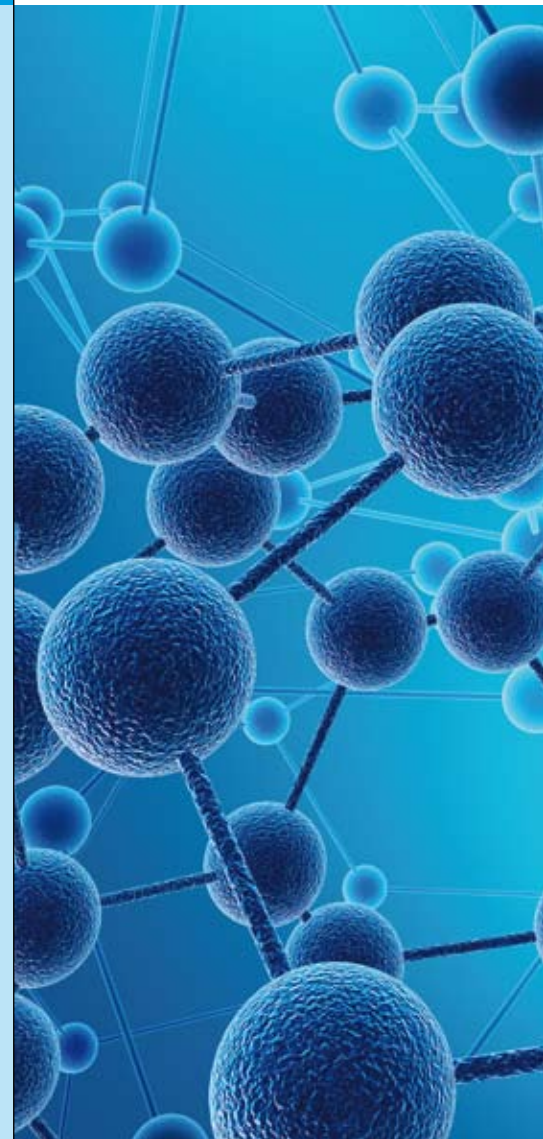
Processing power difficult to harness

The modern GPU has enormous processing power. It can handle up to 30 times as many floating point calculations as a regular CPU (see figure). That sounds great: so much more processing power for broadly the same price. Anyone would want that. But if something sounds too good to be true, it often is. There are plenty of ifs and buts attached to 'alternative' uses of a GPU, other than as a graphics card in a PC. The arithmetical problem needs to suit the architecture of a GPU, as does the floating point precision, and in practical terms it is not possible to combine a GPU with hardware architecture that is different to a PC's.



Dream scenario: GPU versus CPU

Optimum deployment of a GPU for image processing can yield the above curve (with many more MIPS for the GPU than the CPU). Beyond the specific graphics application, a GPU is limited. Running non-graphics applications on a GPU would show a different picture: the performance collapses to below the CPU-curve.



Deeper into the technology of the GPU

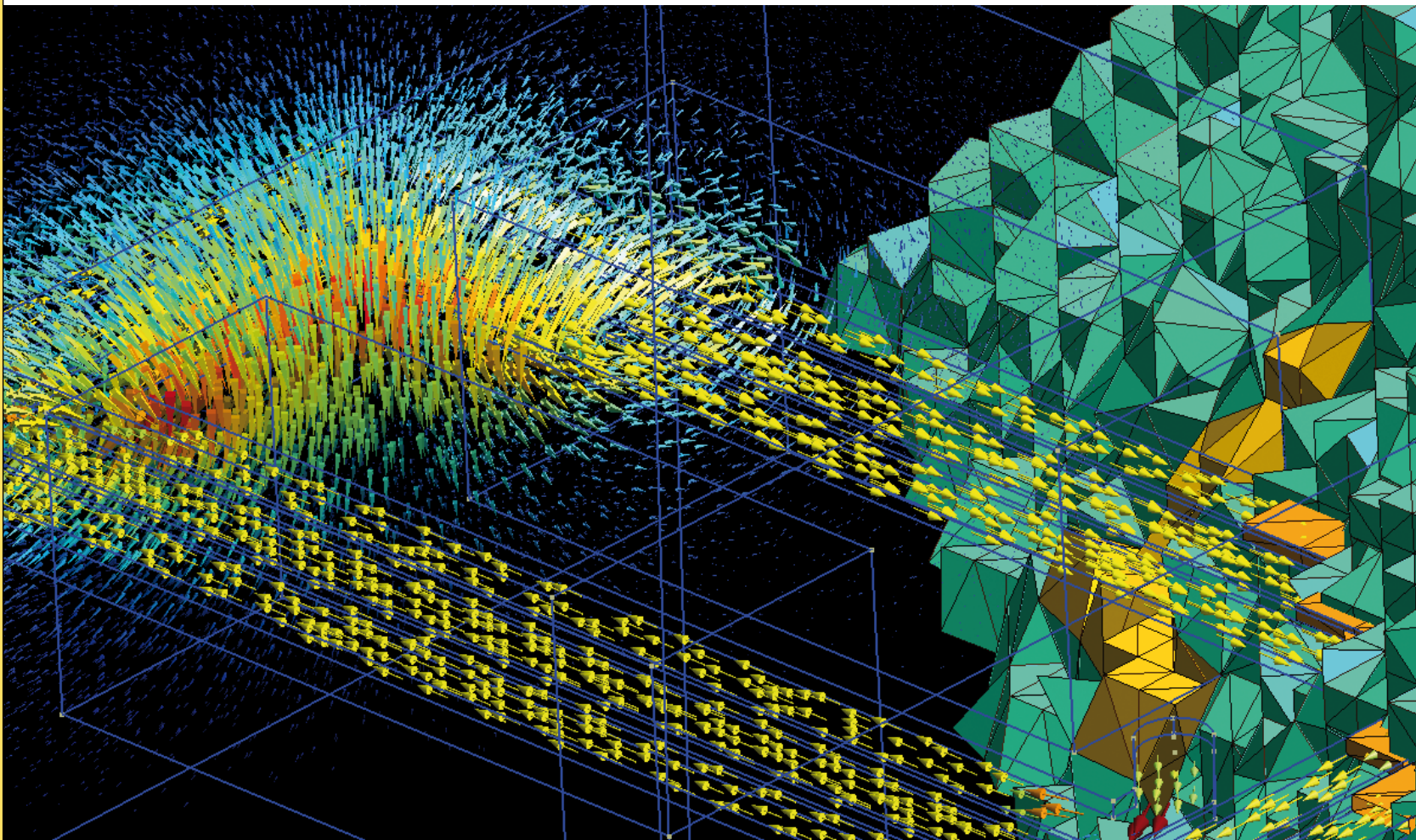
So should we simply conclude that while the images a GPU generates get better every year, it will never be more than a picture machine? Things are not quite so black and white. In certain cases and under strict conditions, a GPU could serve as an alternative for other big processing tasks beyond graphical work. To understand this, we need to delve a little deeper into the technology. A GPU is a parallel processor: it carries out a very large number of parallel calculations. This is reflected, for example, in the large memory bus widths, up to 256 bits. By way of comparison: the CPU relatively recently went from 32 to 64 bits. So a GPU handles a lot more data at a time. But the design is intended for one-way traffic: a chunk of data comes in with elementary image information; it makes some calculations for each pixel, then it pushes the results out the other end in the direction of the screen.

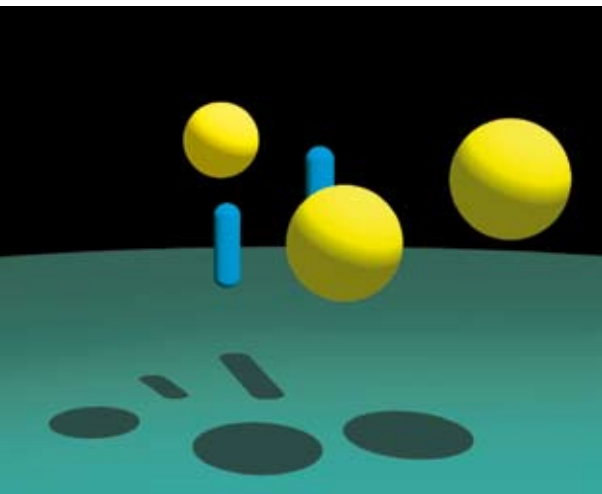
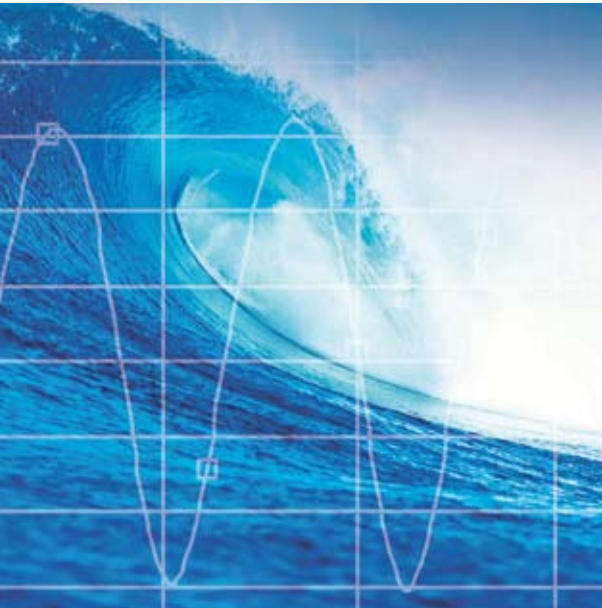
A GPU is bad at making choices, such as going through typical “if ..., then ..., else ...” software constructions. That slows it down. The point of a GPU is that it is designed to do the same for all parallel paths. This is called single instruction, multiple data (SIMD). There is an instruction decoder and a whole group of execution units. As soon as a choice comes along, it can only apply to one execution unit and the others cannot do anything effective. An ordinary processor is much better at solving if-then-else problems. It can smartly pick its way between a series of conditions and dependencies and save itself a lot of processing, like a skier

continually making choices and steering to find the fastest way down a run. A GPU is more like a rowing boat with, say, eight rowers all trying to reach the same goal. If each rower were told to row a different course, the only way to achieve it would be to row eight different courses separately, each time with just one of the rowers rowing. So a GPU is good at parallel processing of arithmetical calculations according to fixed formulas.

Directing calculations using API

If a GPU is to be used, it will require special instructions, which can differ for every brand of GPU. The standard APIs are not usable. It is therefore essential that access to the GPUs is standardised, in order that the required functions can be provided at a reasonably abstract level. At the moment, the two most important APIs are CUDA from NVIDIA and the brand-independent OpenCL (Open Computing Language). OpenCL was initially defined by Apple Inc., who then turned it into an open standard via the Kronos Group. By using these APIs, the complex details of a GPU can be managed, allowing the programmer to focus on the problem at hand. Thanks to standardisation programmes (e.g. Matlab), libraries will also become available with GPU support for frequent problems, meaning that people will not need to reinvent the wheel each time.





Using the processing power of the GPU differently

As indicated above, the APIs cited make it possible to successfully carry out suitable arithmetical problems, without output to a screen, faster on a GPU than on a CPU. Examples would include matrix calculations and Fourier transformations, for example in spatial problems such as 3D simulations of flows or electromagnetic fields. These are phenomena which may be analysed by dividing the world into small boxes or cubes to solve a physical equation (also known as the finite element method). Solving such equations requires a very large number of 'simple' calculating steps applied to a large data set, but also other large-scale calculations (particularly floating point) and image generation for the professional and scientific domains. For these kinds of applications, NVIDIA has developed a professional GPU known as Tesla. This allows a researcher to have a desktop which can compete with a supercomputer in terms of processing power for a relatively modest sum.

>>no "if..., then..., else..."

GPU in embedded systems?

For professional technical applications, a GPU is only usable in its natural habitat: on a graphics card, built into a PC, with standard software. Even where GPU chips are available separately, they are difficult to fit into embedded systems. The problem needs to fit with the architecture of the chip, the chip needs to fit into the system and the business model needs to be right. For instance, being designed for graphics applications GPUs have a short lifespan and as a result are difficult to source over the long term. The professional embedded world is virtually always better off with FPGAs, which will remain available for long enough. The choice of FPGAs is enormous: for every conceivable problem there is a specific FPGA somewhere. Not only that, they are highly flexible, because the developer himself can choose the balance between parallel and serial, between data load and processing power. In some cases not as many separate calculations are required, but they need to be performed on 10Gb data streams, such as in a high resolution camera. This generates a bulk of data that first needs to be processed before a PC can do anything with it. A GPU might be able to do the job, but for processing in the embedded domain Technolution always opts to process the data stream using an FPGA – precisely because of the issues discussed above.

The future of the GPU

One of the biggest limitations in parallel processing today is memory throughput speed. Larrabee is the code name of a development at Intel which marries an X86 CPU with a GPU to create a hybrid GPGPU (General Purpose GPU). In this case, that means genuine CPU cores with SIMD-support and a conventional cache. In the future, integrating a CPU and a Larrabee chip would allow both types of processors to be placed onto a single silicon wafer, giving very short and hence very fast connections. This would provide a lot of flexibility and enhanced performance for a broader range of processing-intensive applications. Intel's performance claims are being treated with scepticism by existing GPU specialists. But if this combination chip succeeds, it could herald the end of the separate graphics card and so bring entirely new applications for the GPU into view.