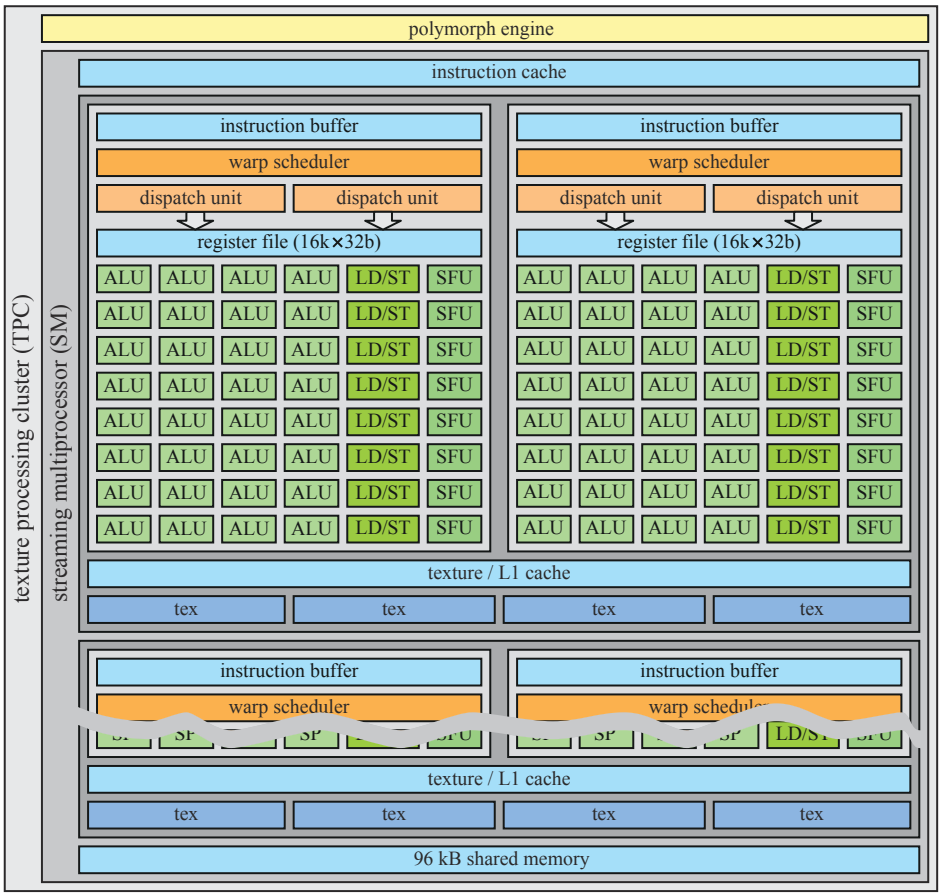

Figure 23.24. The Pascal streaming multiprocessor (SM) has $32 \times 2 \times 2$ unified ALUs and the SM is encapsulated with a polymorph engine, which together form a texture processing cluster (TPC). Note that the top dark gray box has been duplicated just below it, but parts of that duplication have been left out. *(Illustration after NVIDIA white paper [1297].)*

All ALUs in the SM share a single instruction cache, while each SIMT engine has its own instruction buffer with a local set of recently loaded instructions to further increase the instruction cache hit ratio. The warp scheduler is capable of dispatching two warp instructions per clock cycle [1298], e.g., work can be scheduled to both the ALUs and the LD/ST units in the same clock cycle. Note that there are also two L1 caches per SM, each having 24 kB of storage, i.e., 48 kB per SM. The reason to have two L1 caches is likely that a larger L1 cache would need more read and write ports, which increases the complexity of the cache and makes the implementation larger on chip. In addition, there are 8 texture units per SM.

Because shading must be done in 2×2 pixel quads, the warp scheduler finds work of 8 different pixel quads and groups them together for execution in the 32 SIMT lanes [1050]. Since this is a unified ALU design, the warp scheduler can group one of vertices, pixels, primitives, or compute shader work into warps. Note that an SM can handle different types of warps (such as vertices, pixels, and primitives) at the same time. The architecture also has zero overhead for switching out a currently executing warp for a warp that is ready for execution. The details of what warp is next selected for execution on Pascal are not public, but a previous NVIDIA architecture gives us some hints. In the NVIDIA Tesla architecture from 2008 [1050], a *scoreboard* was used to qualify each warp for issue each clock cycle. A scoreboard is a general mechanism that allows for out-of-order execution without conflicts. The warp scheduler chooses among the warps that are ready for execution, e.g., not waiting for a texture load to return, and chooses the one with the highest priority. Warp type, instruction type, and "fairness" are the parameters that are used to select the highest-priority warp.

The SM works in conjunction with the *polymorph engine* (PM). This unit was introduced in its first incarnation in the Fermi chip [1296]. The PM performs several geometry-related tasks, including vertex fetch, tessellation, simultaneous multi-projection, attribute setup, and stream output. The first stage fetches vertices from a global vertex buffer and dispatches warps to SMs for vertex and hull shading. Then follows an optional tessellation stage (Section 17.6), where newly generated (u, v) patch coordinates are dispatched to the SMs for domain shading and, optionally, geometry shading. The third stage handles viewport transform and perspective correction. In addition, an optional simultaneous multi-projection step is executed here, which can be used for efficient VR rendering, for example (Section 21.3.1). Next comes an optional fourth stage, where vertices are streamed out to memory. Finally, the results are forwarded to the relevant raster engines.

A raster engine has three tasks, namely, triangle setup, triangle traversal, and z-culling. Triangle setup fetches vertices, computes edge equations, and performs backface culling. Triangle traversal uses a hierarchical tiled traversal technique to visit the tiles overlapping a triangle. It uses the edge equations to perform tile tests and to perform the inside tests. On Fermi, each rasterizer can process up to 8 pixels per clock cycle [1296]. There are no public numbers for this on Pascal. The z-culling unit handles culling on a per-tile basis using the techniques described in Section 23.7. If a tile is culled, then processing is immediately terminated for that tile. For surviving triangles, per-vertex attributes are converted into plane equations for efficient evaluation in the pixel shader.

A streaming processor coupled with a polymorph engine is called a *texture processing cluster* (TPC). On a higher level, five TPCs are grouped into a *graphics processing cluster* (GPC) that has a single raster engine serving these five TPCs. A GPC can be thought of as a small GPU, and its goal is to provide a balanced set of hardware units for graphics, e.g., vertex, geometry, raster, texture, pixel, and ROP units. As we will see at the end of this section, creating separate functional units allows designers to more easily create a family of GPU chips with a range of capabilities.

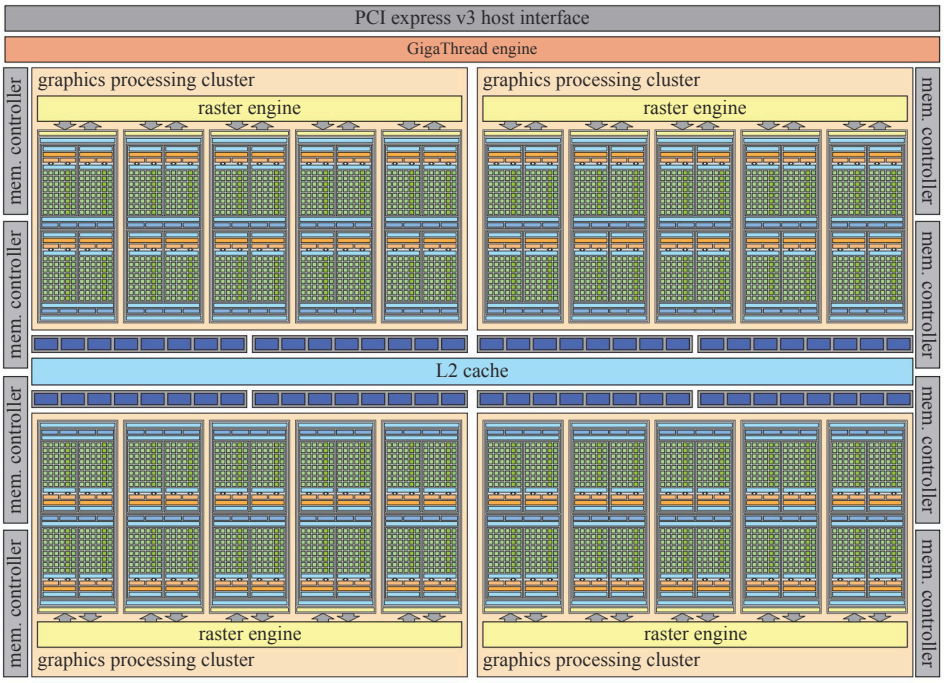

Figure 23.25. The Pascal GPU in its GTX 1080 configuration with 20 SMs, 20 polymorph engines, 4 raster engines, $8 \times 20 = 160$ texture units (with a peak rate of 277.3 Gtexels/s), $256 \times 20 = 5120$ kB worth of register file, and a total of $20 \times 128 = 2560$ unified ALUs. *(Illustration after NVIDIA white paper [1297].)*

At this point, we have most of the building blocks for the GeForce GTX 1080. It consists of four GPCs, and this general setup is shown in Figure 23.25. Notice that there is another level of scheduling here, powered by the GigaThread engine, along with an interface to PCIe v3. The GigaThread engine is a global work distribution engine that schedules blocks of threads to all the GPCs.

The raster operation units are also displayed in Figure 23.25, albeit somewhat hidden. They are located immediately above and below the L2 cache in the middle of the figure. Each blue block is one ROP unit, and there are 8 groups, each with 8 ROPs for a total of 64. The major tasks of the ROP units are to write output to pixels and other buffers, and to perform various operations such as blending. As can be seen on the left and right in the figure, there are a total of eight 32-bit memory controllers, which sums to 256 bits in total. Eight ROP units are tied to a single memory controller and 256 kB of the L2 cache. This gives a total of 2 MB of the L2 cache for the entire chip. Each ROP is tied to a certain memory partition, which means that a ROP handles a certain subset of the pixels in a buffer. The ROP units also handle lossless compression. There are three different compression modes

Figure 23.26. The rendered image is shown on the left, while the compression results are visualized for Maxwell (middle), the architecture before Pascal, and for Pascal (right). The more purple the image is, the higher the success rate of buffer compression. *(Images from NVIDIA white paper [1297].)*

in addition to supporting uncompressed and fast clears [1297]. For $2:1$ compression (e.g., from 256 B to 128 B), a reference color value is stored per tile and the differences are encoded between pixels, where each difference is encoded with fewer bits than its uncompressed form. Then $4:1$ compression is an extension of the $2:1$ mode, but this mode can only be enabled if the differences can be encoded using even fewer bits, and it works for only those tiles with smoothly varying content. There is also an $8:1$ mode, which is a combination of $4:1$ constant color compression of 2×2 pixel blocks with the $2:1$ mode above. The $8:1$ mode has priority over $4:1$, which has priority over $2:1$, i.e., the mode with the highest compression rate that also succeeds at compressing the tile is always used. If all these compression attempts fail, the tile has to be transferred and stored in memory as uncompressed. The efficiency of the Pascal compression system is illustrated in Figure 23.26.

The video memory used is GDDRX5 with a clock rate of 10 GHz. Above we saw that the eight memory controllers provide 256 bits = 32 B in total. This gives a total of 320 GB/s of total peak memory bandwidth, but the many levels of caching combined with the compression techniques give an impression of a higher effective rate.

The base clock frequency of the chip is 1607 MHz, and it can operate in boost mode (1733 MHz) for when there is sufficient power budget. The peak compute capability is

$$\underbrace{2}_{\text{FMA}} \cdot \underbrace{2560}_{\text{num. SPs}} \cdot \underbrace{1733}_{\text{clock freq.}} = 8,872,960 \text{ MFLOPS} \approx 8.9 \text{ TFLOPS}, \tag{23.16}$$

where the 2 comes from the fact that a fused-multiply-and-add often is counted as two floating point operations and we have divided by 10^6 to convert from MFLOPS to TFLOPS. The GTX 1080 Ti has 3584 ALUs, which results in 12.3 TFLOPS.

NVIDIA has developed sort-last fragment architectures for a long time. However, since Maxwell, they also support a new type of rendering called *tiled caching*, which is somewhat between sort-middle and sort-last fragment. This architecture is illustrated in Figure 23.27. The idea is to exploit locality and the L2 cache. Geometry is processed in small enough chunks so that the output can stay in this cache. In addition, the framebuffer stays in L2 as well, as long as the geometry overlapping that tile has not finished pixel shading.

There are four raster engines in Figure 23.25, but as we know the graphics APIs must (in most cases) respect primitive submission order [1598]. The framebuffer is

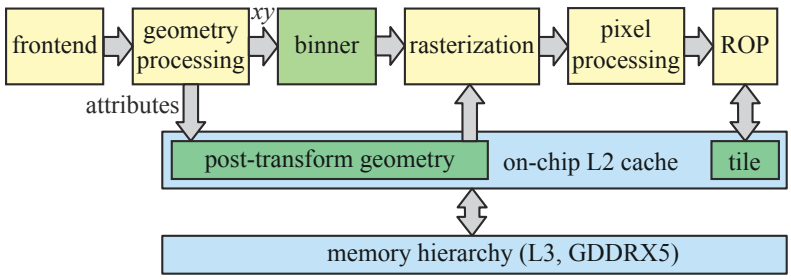

Figure 23.27. Tiled caching introduces a binner that sorts geometry into tiles and lets the transformed geometry stay in the L2 cache. The currently processed tile also stays in L2 until the geometry in that tile for the current chunk is done.

often split into tiles using a generalized checkerboard pattern [1160], and each raster engine "owns" a set of the tiles. The current triangle is sent to each raster engine that has at least one of its tiles overlapping with the triangle, which solves the ordering problem independently for each tile. This makes for better load balancing. There are usually also several FIFO queues in a GPU architecture, which are there to reduce starvation of hardware units. These queues are not shown in our diagrams.

The display controller has 12 bits per color component and has BT.2020 wide color gamut support. It also supports HDMI 2.0b and HDCP 2.2. For video processing, it supports SMPTE 2084, which is a transfer function for high dynamic range video. Venkataraman [1816] describes how NVIDIA architectures from Fermi and after have one or more *copy engines*. These are memory controllers that can perform *direct memory access* (DMA) transfers. A DMA transfer occur between the CPU and the GPU, and such a transfer is typically started on either of these. The starting processing unit can continue doing other computations during the transfer. The copy engines can initiate DMA transfers of data between the CPU and the GPU memory, and they can execute independently of the rest of the GPU. Hence, the GPU can render triangles and perform other functions while information is transferred from the CPU to the GPU or vice versa.

The Pascal architecture can also be configured for non-graphical applications, such as for training neural networks or large-scale data analysis. The Tesla P100 is one such configuration [1298]. Some of the differences from the GTX 1080 include that it uses high-bandwidth memory 2 (HBM2) with 4096 bits for the memory bus, providing a total memory bandwidth of 720 GB/s. In addition, they have native 16-bit floating point support, with up to 2× the performance of 32-bit floating point, and substantially faster double precision processing. The SM configuration is also different, as well as the register file setup [1298].

The GTX 1080 Ti (titanium) is a higher-end configuration. It has 3584 ALUs, a 352-bit memory bus, 484 GB/s in total memory bandwidth, 88 ROPs, and 224 texture units, compared to 2560, 256-bit, 320 GB/s, 64, and 160 for the GTX 1080.

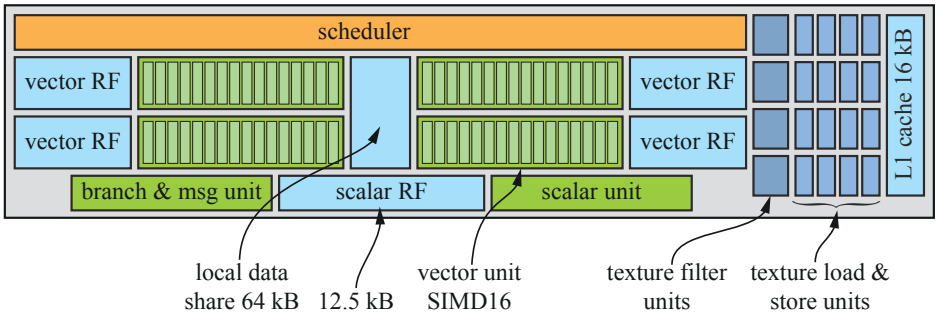

Figure 23.28. The GCN compute unit of the Vega architecture. Each of the vector register files has a 64 kB capacity, while the scalar RF has 12.5 kB, and local data share has 64 kB. Note that there are four units of 16 SIMD lanes (light green), with 32-bit floating point, for compute in each CU. *(Illustration after Mah [1103] and AMD white paper [35].)*

It is configured using six GPCs, i.e., it has six raster engines, compared to four in the GTX 1080. Four of the GPCs are exactly the same as in the GTX 1080, while the remaining two are somewhat smaller with only four TPCs instead of five. The 1080 Ti is built from 12 billion transistors for the chip, while the 1080 uses 7.2 billion. The Pascal architecture is flexible in that it can also scale down. For example, the GTX 1070 is a GTX 1080 minus one GPC, and the GTX 1050 consists of two GPCs, each with three SMs.

23.10.3 Case Study: AMD GCN Vega

The AMD Graphics Core Next (GCN) architecture is used in several AMD graphics card products as well as in the Xbox One and PLAYSTATION 4. Here, we describe general elements of the GCN Vega architecture [35], which is an evolution of the architectures used in these consoles.

A core building block of the GCN architecture is the compute unit (CU), which is illustrated in Figure 23.28. The CU has four SIMD units, each having 16 SIMD lanes, i.e., 16 unified ALUs (using the terminology from Section 23.2). Each SIMD unit executes instructions for 64 threads, which is called a *wavefront*. One single-precision floating point instruction per clock cycle can be issued per SIMD unit. Because the architecture processes a wavefront of 64 threads per SIMD unit, it takes 4 clock cycles before a wavefront has been fully issued [1103]. Note also that a CU can run code from different kernels at the same time. Since each SIMD unit has 16 lanes and one instruction can be issued per clock cycle, the maximum throughput for the entire CU is 4 SIMD units per CU $\times$ 16 SIMD lanes per unit $=$ 64 single-precision FP operations per clock cycle. The CU can also execute twice as many half-precision (16-bit floating point) instructions compared to single-precision FP, which can be useful for cases where less accuracy is needed. This can include machine learning and shader computations, for example. Note that two 16-bit FP values are packed into a single

32-bit FP register. Each SIMD unit has a 64 kB register file, which amounts to $65,536/(4 \cdot 64) = 256$ registers per thread, since a single-precision FP uses 4 bytes and there are 64 threads per wavefront. The ALUs have four hardware pipeline stages [35].

Each CU has an instruction cache (not shown in the figure) that is shared between up to four SIMD units. The relevant instructions are forwarded to a SIMD unit's instruction buffer (IB). Each IB has storage for handling 10 wavefronts, which can be switched in and out of the SIMD unit as needed in order to hide latency. This means that the CU can handle 40 wavefronts. This, in turn, is equivalent to $40 \cdot 64 = 2560$ threads. The CU scheduler in Figure 23.28 can thus handle 2560 threads at a time, and its task is to distribute work to the different units of the CU. Each clock cycle, all wavefronts on the current CU are considered for instruction issues, and up to one instruction can be issued to each execution port. The execution ports of the CU include branches, scalar/vector ALU, scalar/vector memory, local data share, global data share or export, and special instructions [32], that is, each execution port maps roughly to one unit of the CU.

The scalar unit is a 64-bit ALU that is shared between the SIMD units as well. It has its own scalar register file and a scalar data cache (not shown). The scalar RF has 800 32-bit registers per SIMD unit, i.e., $800 \cdot 4 \cdot 4 = 12.5$ kB. Execution is tightly coupled to the wavefronts. Since it takes four clock cycles to fully issue an instruction into a SIMD unit, the scalar unit can serve a particular SIMD unit only every fourth clock cycle. The scalar unit handles control flow, pointer arithmetic, and other computations that can be shared among the threads in a warp. Conditional and unconditional branch instructions are sent from the scalar unit for execution in the branch and message unit. Each SIMD unit has a single 48-bit program counter (PC) that is shared between lanes. This is sufficient, since all of them execute the same instructions. For taken branches, the program counter is updated. Messages that can be sent by this unit include debug messages, special graphics synchronization messages, and CPU interrupts [1121].

The Vega 10 architecture [35] is illustrated in Figure 23.29. The top part includes a graphics command processor, two hardware schedulers (HWSs) and eight asynchronous compute engines (ACEs) [33]. The task of the GPC is to dispatch graphics tasks onto the graphics pipelines and compute engines of the GPU. The HWSs' buffers work in queues that they assign to the ACEs as soon as it is possible. The task of an ACE is to schedule compute tasks onto the compute engines. There are also two DMA engines that can handle copy tasks (not shown in the figure). The GPC, ACEs, and DMA engines can work in parallel and submit work to the GPU, which improves utilization, since tasks can be interleaved from different queues. Work can be dispatched from any queue without waiting for other work to finish, which means that independent tasks can execute on the compute engines simultaneously. The ACEs can synchronize via cache or memory. They can support task graphs together, so that one ACE's task can depend on another ACE's task, or on the graphics pipeline's tasks. It is recommended that smaller compute and copy tasks are interleaved with heavier graphics tasks [33].

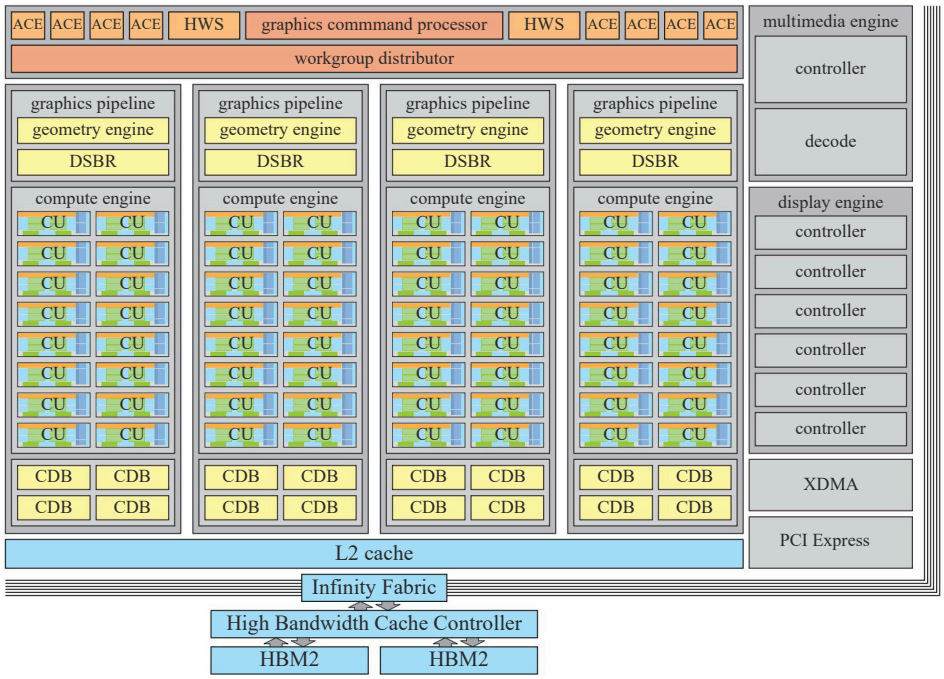

Figure 23.29. A Vega 10 GPU built with 64 CUs. Note that each CU contains the hardware shown in Figure 23.28. *(Illustration after AMD white paper [35].)*

As can be seen in Figure 23.29, there are four graphics pipelines and four compute engines. Each compute engine has 16 CUs, which sums to 64 CUs in total. The graphics pipeline has two blocks, namely a geometry engine and draw-stream binning rasterizer (DSBR). The geometry engine includes a geometry assembler, tessellation unit, and vertex assembler. In addition, a new *primitive shader* is supported. The idea of the primitive shader is to enable more flexible geometry processing and faster culling of primitives [35]. The DSBR combines the advantages of sort-middle and sort-last architectures, which also is the goal of tiled caching (Section 23.10.2). The image is divided into tiles in screen space, and after geometry processing, each primitive is assigned to the tiles they overlap. During rasterization of a tile, all data (e.g., tile buffers) required are kept in the L2 cache, which improves performance. Pixel shading can be deferred automatically until all the geometry in a tile has been processed. Hence, a z-prepass is done under the hood and pixels shaded only once. Deferred shading can be turned on and off; e.g., for transparent geometry it needs to be off.

To handle depth, stencil, and color buffers, the GCN architecture has a building block called the color and depth block (CDB). They handle color, depth, and stencil

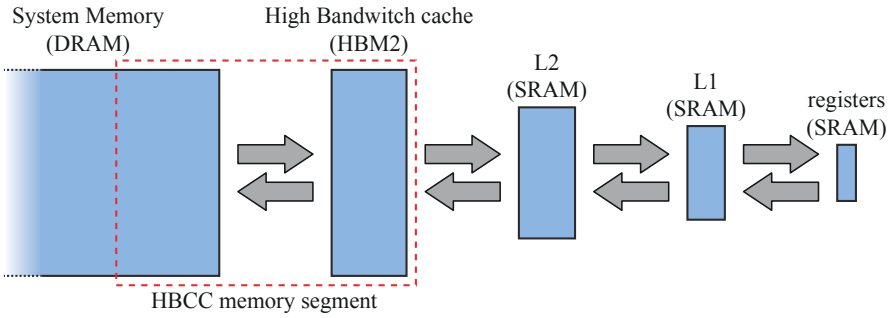

Figure 23.30. The cache hierarchy of the Vega architecture.

read and writes, in addition to color blending. A CDB can compress the color buffer using the general approach described in Section 23.5. A delta compression technique is used where one pixel's color is stored uncompressed per tile, and the rest of the color values are encoded relative to that pixel color [34, 1238]. In order to increase efficiency, the tile size can be dynamically chosen based on access patterns. For a tile originally stored using 256 bytes, the maximum rate is 8 : 1, i.e., compression down to 32 bytes. A compressed color buffer can be used as a texture in a subsequent pass, in which case the texture unit will decompress the compressed tile, which provides further bandwidth savings [1716].

The rasterizers can rasterize up to four primitives per clock cycle. The CDBs connected to the graphics pipeline and compute engine can write 16 pixels per clock cycle. That is, triangles smaller than 16 pixels reduce efficiency. The rasterizer also handles coarse depth testing (HiZ) and hierarchical stencil testing. The buffer for HiZ is called *HTILE* and can be programmed by the developer, e.g., for feeding occlusion information to the GPU.

The cache hierarchy of Vega is shown in Figure 23.30. At the top of the hierarchy (rightmost in the figure), we have registers, followed by the L1 and L2 caches. Then, there is high-bandwidth memory 2 (HBM2), which is located on the graphics card as well, and finally system memory located on the CPU side. A new feature of Vega is the High-Bandwidth Cache Controller (HBCC), shown at the bottom of Figure 23.29. It allows the video memory to behave like a last-level cache. This means that if a memory access is made and the corresponding content is not in the video memory, i.e., HBM2, then the HBCC will automatically fetch the relevant system memory page(s) over the PCIe bus and put it in video memory. Less recently used pages in the video memory may be swapped out as a consequence. The memory pool shared between the HBM2 and system memory is called the HBCC memory segment (HMS). All graphics blocks also access memory through the L2 cache, which differs from previous architectures. The architecture also supports virtual memory (Section 19.10.1).

Note that all the on-chip blocks, e.g., HBCC, XDMA (CrossFire DMA), PCI express, display engines, and multimedia engines, communicate over an interconnect called the *Infinity Fabric* (IF). AMD CPUs can also be connected to the IF. The Infinity Fabric can connect blocks on different chip dies. The IF is also coherent, meaning that all blocks get to see the same view of the content in memory.

The base clock frequency of the chip is 1677 MHz, i.e., the peak compute capability is

$$\underbrace{2}_{\text{FMA}} \cdot \underbrace{4096}_{\text{num SPs}} \cdot \underbrace{1677}_{\text{clock freq.}} = 13,737,984 \text{ MFLOPS} \approx 13.7 \text{ TFLOPS}, \tag{23.17}$$

where the FMA and TFLOPS calculations match those in Equation 23.16. The architecture is flexible and extendable, so many more configurations are expected.

23.11 Ray Tracing Architectures

This section will give a brief introduction to ray tracing hardware. We will not list all recent references on this topic, but rather provide a set of pointers that the reader is encouraged to follow. Research in this field was started by Schmittler et al. [1571] in 2002, where focus was on traversal and intersection, and shading was computed using a fixed-function unit. This work was later followed up by Woop et al. [1905], who presented an architecture with programmable shaders.

The commercial interest in this topic has increased considerably over the past few years. This can be seen in the fact that companies, such as Imagination Technologies [1158], LG Electronics [1256], and Samsung [1013], have presented their own hardware architectures for real-time ray tracing. However, only Imagination Technologies has released a commercial product at the time of writing.

There are several common traits in these architectures. First, they often use a bounding volume hierarchy based on axis-aligned bounding boxes. Second, they tend to reduce hardware complexity by reducing precision in the ray/box intersection tests (Section 22.7). Finally, they use programmable cores to support programmable shading, which is more or less a requirement today. For example, Imagination Technologies extend their traditional chip design by adding a ray tracing unit, which can exploit the shader cores for shading, for example. The ray tracing unit consists of a ray intersection processor and a coherency engine [1158], where the latter gathers rays with similar properties and processes them together to exploit locality for faster ray tracing. The architecture from Imagination Technologies also includes a dedicated unit for building BVHs.

Research in this field continues to explore several areas, including reduced precision for efficient implementation of traversal [1807], compressed representations for BVHs [1045], and energy efficiency [929]. There is undoubtedly more research to be done.

Further Reading and Resources

A great set of resources are the course notes on computer graphics architectures by Akeley and Hanrahan [20] and Hwu and Kirk [793]. The book by Kirk and Hwu [903] is also an excellent resource for information about programming with CUDA on GPUs. The annual *High-Performance Graphics* and *SIGGRAPH* conference proceedings are good sources for presentations on new architectural features. Giesen's trip down the graphics pipeline is a wonderful online resource for anyone wanting to learn more about the details of GPUs [530]. We also refer the interested reader to Hennessy and Patterson's book [715] for detailed information about memory systems. Information on mobile rendering is scattered among many sources. Of note, the book *GPU Pro 5* has seven articles on mobile rendering techniques.

Chapter 24
The Future

"Pretty soon, computers will be fast."
—Billy Zelsnack

"Prediction is difficult, especially of the future."
—Niels Bohr or Yogi Berra

"The best way to predict the future is to create it."
—Alan Kay

There are two parts to the future: you and everything else. This chapter is about both. First, we will make some predictions, a few of which may even come true. More important is the second part, about where you could go next. It is something of an extended *Further Reading and Resources* section, but it also discusses ways to proceed from here—general sources of information, conferences, code, and more. But first, an image: See Figure 24.1.

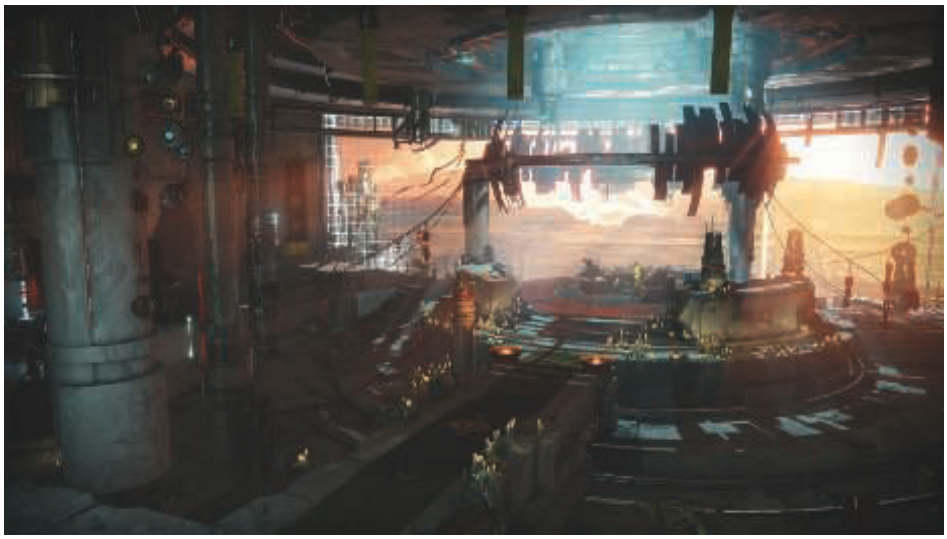

Figure 24.1. A glimpse of one future, through the game *Destiny 2*. *(Image ©2017 Bungie, Inc. all rights reserved.)*

24.1 Everything Else

Graphics helps sell games, and games help sell chips. One of the best features of real-time rendering from a chip-maker's marketing perspective is that graphics eats huge amounts of processing power and other resources. Hardware-related features such as frame rate, resolution, and color depth can also grow to some extent, further increasing the load. A minimum solid frame rate of 90 FPS is the norm for virtual reality applications, and 4k pixel displays are already testing the abilities of graphics systems to keep up [1885].

The complex task of simulating the effect of light in a scene is plenty on its own for absorbing compute power. Adding more objects or lights to a scene is one way in which rendering can clearly become more expensive. The types of objects (both solid and volumetric, such as fog), the way these objects' surfaces are portrayed, and the types of lights used are just some factors where complexity can increase. Many algorithms improve in quality if we can take more samples, evaluate more accurate equations, or simply use more memory. Increasing complexity makes graphics a nearly bottomless pit for processing power to attempt to fill.

To solve performance concerns in the long run, rosy-eyed optimists like to turn to Moore's Law. This observation gives an acceleration rate of 2× every 1.5 years or, more usefully, about 10× speedup every 5 years [1663]. However, processor speed is usually not the bottleneck, and probably will be less so as time goes on. Bandwidth is, as it increases by a factor of 10 every 10 years, not 5 [1332].

Algorithms from the film industry often find their way into real-time rendering, since the fields share the same goal of generating realistic images. Looking at their practices, we see statistics such as that a single frame of the 2016 movie *The Jungle Book* includes millions of hairs in some scenes, with render times of 30 to 40 hours a frame [1960]. While GPUs are purpose-built for real-time rendering and so have a noticeable advantage over CPUs, going from $1/(40 \times 60 \times 60) = 0.00000694$ FPS to 60 FPS is about seven orders of magnitude.

We promised some predictions. "Faster and more flexible" is a simple one to make. As far as GPU architecture goes, one possibility is that the z-buffer triangle rasterization pipeline will continue to rule the roost. All but the simplest games use the GPU for rendering. Even if tomorrow some incredible technique supplanted the current pipeline, one that was a hundred times faster and that consisted of downloading a system patch, it could still take years for the industry to move to this new technology. One catch would be whether the new method could use exactly the same APIs as existing ones. If not, adoption would take a while. A complex game costs tens of millions of dollars or more to develop and takes years to make. The target platforms are chosen early in the process, which informs decisions about everything from the algorithms and shaders used, to the size and complexity of artwork produced. Beyond those factors, the tools needed to work with or produce these elements need to be made and users need to become proficient in their use. The momentum that the current rasterizer pipeline has behind it gives it several years of life, even with a miracle occurring.

Change still happens. In reality, the simple "one rasterizer to rule them all" idea has already begun to fade. Throughout this book we have discussed how the compute shader is able to take on various tasks, proof that rasterization is hardly the only service a GPU can offer. If new techniques are compelling, retooling the workflow will happen, percolating out from game companies to commercial engines and content creation tools.

So, what of the long-term? Dedicated fixed-function GPU hardware for rendering triangles, accessing textures, and blending resulting samples still gives critical boosts to performance. The needs of mobile devices change this equation, as power consumption becomes as much of a factor as raw performance. However, the "fire-and-forget" concept of the basic pipeline, where we send a triangle down the pipeline once and are entirely done with it for that frame, is not the model used in modern rendering engines. The basic pipeline model of transform, scan, shade, and blend has evolved almost beyond recognition. The GPU has become a large cluster of stream-based processors to use as you wish.

APIs and GPUs have coevolved to adapt to this reality. The mantra is "flexibility." Methods are explored by researchers, then implemented on existing hardware by developers, who identify functionality they wish was available. Independent hardware vendors can use these findings and their own research to develop general capabilities, in a virtuous cycle. Optimizing for any single algorithm is a fool's errand. Creating new, flexible ways to access and process data on the GPU is not.

With that in mind, we see ray/object intersection as a general tool with numerous uses. We know that perfectly unbiased sampling using path tracing eventually yields the correct, ground-truth image, to the limits of the scene description. It is the word "eventually" that is the catch. As discussed in Section 11.7, there are currently serious challenges for path tracing as a viable algorithm. The main problem is the sheer number of samples needed to get a result that is not noisy, and that does not twinkle when animated. That said, the purity and simplicity of path tracing make it extremely appealing. Instead of the current state of interactive rendering, where a multitude of specialized techniques are tailored for particular situations, just one algorithm does it all. Film studios have certainly come to realize this, as the past decade has seen them move entirely to ray and path tracing methods. Doing so lets them optimize on just one set of geometric operations for light transport.

Real-time rendering—all rendering for that matter—is ultimately about sampling and filtering. Aside from increasing the efficiency of ray shooting, path tracing can benefit from smarter sampling and filtering. As it is, almost every offline path tracer is biased, regardless of marketing literature [1276]. Reasonable assumptions are made about where to send sample rays, vastly improving performance. The other area where path tracing can benefit is intelligent filtering—literally. Deep learning is currently a white-hot area of research and development, with the initial resurgence of interest due to impressive gains in 2012 when it considerably outpaced hand-tweaked algorithms for image recognition [349]. The use of neural nets for denoising [95, 200, 247] and antialiasing [1534] are fascinating developments. See Figure 24.2. We are already

Figure 24.2. Image reconstruction with a neural net. On the left, a noisy image generated with path tracing. On the right, the image cleaned up using a GPU-accelerated denoiser at interactive rates. *(Image courtesy of NVIDIA Corporation [200], using the Amazon Lumberyard Bistro scene.)*

seeing a large uptick in the number of research papers using neural nets for rendering-related tasks, not to mention modeling and animation.

Dating back to AT&T's *Pixel Machine* in 1987, interactive ray tracing has long been possible for small scenes, low resolutions, few lights, and compositions with only sharp reflections, refractions, and shadows. Microsoft's addition of ray tracing functionality to the DirectX API, called *DXR*, simplifies the process of shooting rays and is likely to inspire hardware vendors to add support for ray intersection. Ray shooting, enhanced with denoising or other filtering, will at first be just another technique for improving rendering quality of various elements, such as shadows or reflections. It will compete with many other algorithms, with each rendering engine making choices based on such factors as speed, quality, and ease of use. See Figure 24.3.

Hierarchical ray shooting as a fundamental operation is not an explicit part of any mainstream commercial GPU as of this writing. We take PowerVR's Wizard GPU [1158] as a good sign, in that a mobile device company is considering hardware support for testing rays against a hierarchical scene description. Newer GPUs with direct support for shooting rays will change the equations of efficiency and could create a virtuous cycle, one where various rendering effects are less customized and specialized. Rasterization for the eye rays and ray tracing or compute shaders for almost everything else is one approach, already being used in various DXR demos [1, 47, 745]. With improved denoising algorithms, faster GPUs for tracing rays, and previous research reapplied as well as new investigations, we expect to soon see the equivalent of a 10× performance improvement.

We expect DXR to be a boon to developers and researchers in other ways. For games, baking systems that cast rays can now be run on the GPU and use similar

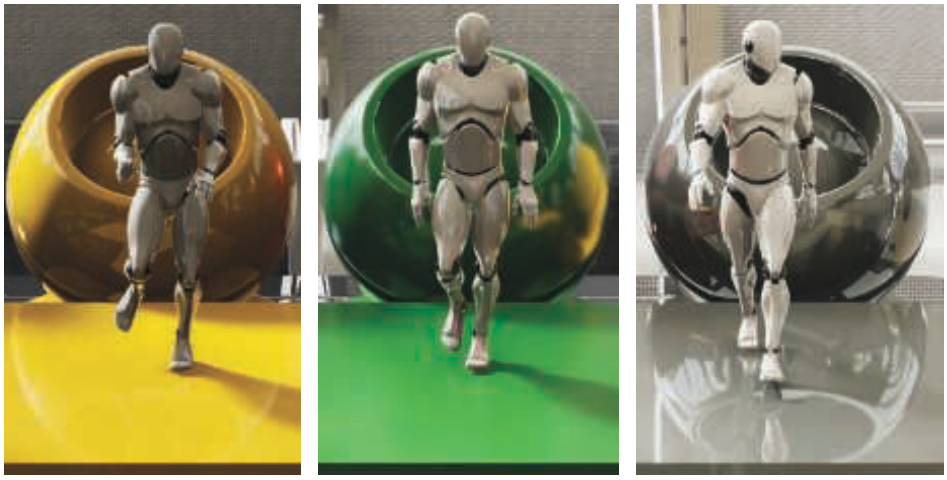

Figure 24.3. These images were rendered at interactive rates with two reflection ray bounces per pixel, a shadow ray for the screen location and both bounces, and two ambient occlusion rays, for a total of seven rays per pixel. Denoising filters were used for shadows and reflections. *(Images courtesy of NVIDIA Corporation.)*

or the same shaders as found in the interactive renderer, with improved performance as a result. Ground-truth images can be more easily generated, making it simpler to test and even auto-tune algorithms. The idea of architectural changes that allow more flexible generation of GPU tasks, e.g., shaders creating shader work, seems a powerful one that will likely have other applications.

There are certainly other fascinating possibilities of how GPUs might evolve. Another idealized view of the world is one in which all matter is voxelized. Such a representation has any number of advantages for light transport and simulation, as discussed in Section 13.10. The large amount of data storage needed, and difficulties with dynamic objects in the scene, make the likelihood of a complete switchover extremely unlikely. Nonetheless, we believe voxels are likely to get more attention, for their use in a wide range of areas, including high-quality volumetric effects, 3D printing, and unconstrained object modification (e.g., *Minecraft*). Certainly a related representation, point clouds, will be part of much more research in the years to come, given the massive amounts of such data generated by self-driving car systems, LIDAR, and other sensors. Signed distance fields (SDFs) are another intriguing scene description method. Similarly to voxels, SDFs enable unconstrained modification of the scene and can accelerate ray tracing as well.

Sometimes, the unique constraints of a given application allow its developers to "break the mold" and use techniques previously considered exotic or infeasible. Games such as Media Molecule's *Dreams* and *Claybook* by Second Order, pictured in Figure 24.4, can give us intriguing glimpses into possible rendering futures where unorthodox algorithms hold sway.

Figure 24.4. *Claybook* is a physics-based puzzle game with a world of clay that can be freely sculpted by users. The clay world is modeled using signed distance fields and rendered with ray tracing, including primary rays as well as ray-traced shadows and AO. Solid and liquid physics are simulated on the GPU. *(Claybook. © 2017 Second Order, Ltd.)*

Virtual and mixed reality deserve a mention. When VR works well, it is breath-taking. Mixed reality has enchanting demos of synthetic content merging with the real world. Everyone wants the lightweight glasses that do both, which is likely to be in the "personal jetpacks, underwater cities" category in the short term. But who knows? Given the huge amount of research and development behind these efforts [1187], there are likely to be some breakthroughs, possibly world-changing ones.

24.2 You

So, while you and your children's children are waiting for The Singularity, what do you do in the meantime? Program, of course: Discover new algorithms, create applications, or do whatever else you enjoy. Decades ago graphics hardware for one machine cost more than a luxury car; now it is built into just about every device with a CPU, and these devices often fit in the palm of your hand. Graphics hacking is inexpensive and mainstream. In this section, we cover various resources we have found to be useful in learning more about the field of real-time rendering.

This book does not exist in a vacuum; it draws upon a huge number of sources of information. If you are interested in a particular algorithm, track down the original publications. Our website has a page of all articles we reference, so you can look there for the link to the resource, if available. Most research articles can be found using

Google Scholar, the author's website, or, if all else fails, ask the author for a copy—almost everyone likes to have their work read and appreciated. If not found for free, services such as the *ACM Digital Library* have a huge number of articles available. If you are a member of SIGGRAPH, you automatically have free access to many of their graphics articles and talks. There are several journals that publish technical articles, such as the *ACM Transactions on Graphics* (which now includes the SIGGRAPH proceedings as an issue), *The Journal of Computer Graphics Techniques* (which is open access), *IEEE Transactions on Visualization and Computer Graphics*, *Computer Graphics Forum*, and *IEEE Computer Graphics and Applications*, to mention a few. Finally, some professional blogs have excellent information, and graphics developers and researchers on Twitter often point out wonderful new resources.

One of the fastest ways to learn and meet others is to attend a conference. Odds are high that another person is doing something you are, or might get, interested in. If money is tight, contact the organizers and ask about volunteer opportunities or scholarships. The SIGGRAPH and SIGGRAPH Asia annual conferences are premier venues for new ideas, but hardly the only ones. Other technical gatherings, such as the Eurographics conference and the Eurographics Symposium on Rendering (EGSR), the Symposium on Interactive 3D Graphics and Games (I3D), and the High Performance Graphics (HPG) forum present and publish a significant amount of material relevant to real-time rendering. There are also developer-specific conferences, such as the well-established Game Developers Conference (GDC). Say hello to strangers when you are waiting in line or at an event. At SIGGRAPH in particular keep an eye out for *birds of a feather* (BOF) gatherings in your areas of interest. Meeting people and exchanging ideas face to face is both rewarding and energizing.

There are a few electronic resources relevant to interactive rendering. Of particular note, the *Graphics Codex* [1188] is a high-quality, purely electronic reference that has the advantage of being continually updated. The site *immersive linear algebra* [1718], created in part by a coauthor of this book, includes interactive demos to aid in learning this topic. Shirley [1628] has an excellent series of short Kindle books on ray tracing. We look forward to more inexpensive and quick-access resources of this sort.

Printed books still have their place. Beyond general texts and field-specific volumes, edited collections of articles include a significant amount of research and development information, many of which we reference in this book. Recent examples are the *GPU Pro* and *GPU Zen* books. Older books such as *Game Programming Gems*, *GPU Gems* (free online), and the *ShaderX* series still have relevant articles—algorithms do not rot. All these books allow game developers to present their methods without having to write a formal conference paper. Such collections also allow academics to discuss technical details about their work that do not fit into a research paper. For a professional developer, an hour saved by reading about some implementation detail found in an article more than pays back the cost of the entire book. If you cannot wait for a book to be delivered, using the "Look Inside" feature on Amazon or searching for the text on Google Books may yield an excerpt to get you started.

When all is said and done, code needs to be written. With the rise of GitHub, Bitbucket, and similar repositories, there is a rich storehouse to draw upon. The hard part is knowing what does not fall under Sturgeon's Law. Products such as the Unreal Engine have made their source open access, and thus an incredible resource. The ACM is now encouraging code to be released for any technical article published. Authors you respect sometimes have their code available. Search around.

One site of particular note is Shadertoy, which often uses ray marching in a pixel shader to show off various techniques. While many programs are first and foremost eye candy, the site has numerous educational demos, all with code visible, and all runnable within your browser. Another source for browser-based demos is the three.js repository and related sites. "Three" is a wrapper around WebGL that encourages experimentation, as just a few lines of code produces a rendering. The ability to publish demos on the web for anyone to run and dissect, just a hyperlink click away, is wonderful for educational uses and for sharing ideas. One of the authors of this book created an introductory graphics course for Udacity based on three.js [645].

We refer you one more time to our website at realtimerendering.com. There you will find many other resources, such as lists of recommended and new books (including a few that are free and of high quality [301, 1729]), as well as pointers to worthwhile blogs, research sites, course presentations, and many other sources of information. Happy hunting!

Our last words of advice are to go and learn and do. The field of real-time computer graphics is continually evolving, and new ideas and features are constantly being invented and integrated. You can be involved. The wide array of techniques employed can seem daunting, but you do not need to implement a laundry list of buzzwords-du-jour to get good results. Cleverly combining a small number of techniques, based on the constraints and visual style of your application, can result in distinctive visuals. Share your results on GitHub, which can also be used to host a blog. Get involved!

One of the best parts of this field is that it reinvents itself every few years. Computer architectures change and improve. What did not work a few years ago may now be worth pursuing. With each new GPU offering comes a different mix of functionality, speed, and memory. What is efficient and what is a bottleneck changes and evolves. Even areas that seem old and well-established are worth revisiting. Creation is said to be a matter of bending, breaking, and blending other ideas, not making something from nothing.

This edition comes 44 years after one of the milestone papers in the field of computer graphics, "A Characterization of Ten Hidden-Surface Algorithms" by Sutherland, Sproull, and Schumacker, published in 1974 [1724]. Their 55-page paper is an incredibly thorough comparison. The algorithm described as "ridiculously expensive," the brute-force technique not even dignified with a researcher's name, and mentioned only in the appendices, is what is now called the z-buffer. In fairness, Sutherland was the adviser of the inventor of the z-buffer, Ed Catmull, whose thesis discussing this concept would be published a few months later [237].

This eleventh hidden-surface technique won out because it was easy to implement in hardware and because memory densities went up and costs went down. The "Ten Algorithms" survey done by Sutherland et al. was perfectly valid for its time. As conditions change, so do the algorithms used. It will be exciting to see what happens in the years to come. How will it feel when we look back on this current era of rendering technology? No one knows, and each person can have a significant effect on the way the future turns out. There is no one future, no course that must occur. You create it.

What do you want to do next? *(CD PROJEKT®, The Witcher® are registered trademarks of CD PROJEKT Capital Group. The Witcher game © CD PROJEKT S.A. Developed by CD PROJEKT S.A. All rights reserved. The Witcher game is based on the prose of Andrzej Sapkowski. All other copyrights and trademarks are the property of their respective owners.)*

Bibliography

[1] Aalto, Tatu, "Experiments with DirectX Raytracing in Remedy's Northlight Engine," *Game Developers Conference*, Mar. 19, 2018. Cited on p. 1044

[2] Aaltonen, Sebastian, "Modern Textureless Deferred Rendering Techniques," *Beyond3D Forum*, Feb. 28, 2016. Cited on p. 906, 907

[3] Abbas, Wasim, "Practical Analytic 2D Signed Distance Field Generation," in *ACM SIGGRAPH 2016 Talks*, article no. 68, July 2016. Cited on p. 677, 678

[4] Abrash, Michael, *Michael Abrash's Graphics Programming Black Book*, Special Edition, The Coriolis Group, Inc., 1997. Cited on p. 823

[5] Abrash, Michael, "Latency—The *sine qua non* of AR and VR," *Ramblings in Valve Time* blog, Dec. 29, 2012. Cited on p. 920, 939

[6] Abrash, Michael, "Raster-Scan Displays: More Than Meets The Eye," *Ramblings in Valve Time* blog, Jan. 28, 2013. Cited on p. 922, 1012

[7] Abrash, Michael, "Down the VR Rabbit Hole: Fixing Judder," *Ramblings in Valve Time* blog, July 26, 2013. Cited on p. 935, 1011

[8] Abrash, Michael, "Oculus Chief Scientist Predicts the Next 5 Years of VR Technology," *Road to VR* website, Nov. 4, 2016. Cited on p. 931, 932

[9] Adams, Ansel, *The Camera*, Little, Brown and Company, 1980. Cited on p. 291

[10] Adams, Ansel, *The Negative*, Little, Brown and Company, 1981. Cited on p. 289, 291

[11] Adams, Ansel, *The Print*, Little, Brown and Company, 1983. Cited on p. 291

[12] Adorjan, Matthias, *OpenSfM: A Collaborative Structure-from-Motion System*, Diploma thesis in Visual Computing, Vienna University of Technology, 2016. Cited on p. 574, 575

[13] Aila, Timo, and Ville Miettinen, "dPVS: An Occlusion Culling System for Massive Dynamic Environments," *IEEE Computer Graphics and Applications*, vol. 24, no. 2, pp. 86–97, Mar. 2004. Cited on p. 666, 821, 839, 850, 879

[14] Aila, Timo, and Samuli Laine, "Alias-Free Shadow Maps," in *Eurographics Symposium on Rendering*, Eurographics Association, pp. 161–166, June 2004. Cited on p. 260

[15] Aila, Timo, and Samuli Laine, "Understanding the Efficiency of Ray Traversal on GPUs," *High Performance Graphics*, June 2009. Cited on p. 511

[16] Aila, Timo, Samuli Laine, and Tero Karras, "Understanding the Efficiency of Ray Traversal on GPUs—Kepler and Fermi Addendum," Technical Report NVR-2012-02, NVIDIA, 2012. Cited on p. 511, 961

[17] Airey, John M., John H. Rohlf, and Frederick P. Brooks Jr., "Towards Image Realism with Interactive Update Rates in Complex Virtual Building Environments," *ACM SIGGRAPH Computer Graphics (Symposium on Interactive 3D Graphics)*, vol. 24, no. 2, pp. 41–50, Mar. 1990. Cited on p. 687, 837

[18] Airey, John M., *Increasing Update Rates in the Building Walkthrough System with Automatic Model-Space Subdivision and Potentially Visible Set Calculations*, PhD thesis, Technical Report TR90-027, Department of Computer Science, University of North Carolina at Chapel Hill, July 1990. Cited on p. 837

[19] Akeley, K., P. Haeberli, and D. Burns, `tomesh.c`, a C-program on the *SGI Developer's Toolbox CD*, 1990. Cited on p. 692

[20] Akeley, Kurt, and Pat Hanrahan, "Real-Time Graphics Architectures," Course CS448A Notes, Stanford University, Fall 2001. Cited on p. 1040

[21] Akenine-Möller, Tomas, "Fast 3D Triangle-Box Overlap Testing," *journal of graphics tools*, vol. 6, no. 1, pp. 29–33, 2001. Cited on p. 974, 975

[22] Akenine-Möller, Tomas, and Jacob Ström, "Graphics for the Masses: A Hardware Rasterization Architecture for Mobile Phones," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 801–808, 2003. Cited on p. 146, 1015, 1027

[23] Akenine-Möller, Tomas, and Ulf Assarsson, "On the Degree of Vertices in a Shadow Volume Silhouette," *journal of graphics tools*, vol. 8, no. 4, pp. 21–24, 2003. Cited on p. 667

[24] Akenine-Möller, T., and T. Aila, "Conservative and Tiled Rasterization Using a Modified Triangle Setup," *journal of graphics tools*, vol. 10, no. 3, pp. 1–8, 2005. Cited on p. 996, 1001

[25] Akenine-Möller, Tomas, and Björn Johnsson, "Performance per What?" *Journal of Computer Graphics Techniques*, vol. 1, no. 18, pp. 37–41, 2012. Cited on p. 790

[26] Akenine-Möller, Tomas, "Some Notes on Graphics Hardware," *Tomas Akenine-Möller* webpage, Nov. 27, 2012. Cited on p. 999, 1000

[27] Akin, Atilla, "Pushing the Limits of Realism of Materials," *Maxwell Render* blog, Nov. 26, 2014. Cited on p. 362, 363

[28] Alexa, Marc, "Recent Advances in Mesh Morphing," *Computer Graphics Forum*, vol. 21, no. 2, pp. 173–197, 2002. Cited on p. 87, 88, 102

[29] Alexa, M., and T. Boubekeur, "Subdivision Shading," *ACM Transactions on Graphics*, vol. 27, no. 5, pp. 142:1–142:3, 2008. Cited on p. 767

[30] Aliaga, Daniel G., and Anselmo Lastra, "Automatic Image Placement to Provide a Guaranteed Frame Rate," in *SIGGRAPH '99: Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, ACM Press/Addison-Wesley Publishing Co., pp. 307–316, Aug. 1999. Cited on p. 561

[31] AMD, "AMD PowerTune Technology," AMD website, 2011. Cited on p. 789

[32] AMD, "AMD Graphics Cores Next (GCN) Architecture," AMD website, 2012. Cited on p. 1036

[33] AMD, "Asynchronous Shaders: Unlocking the Full Potential of the GPU," AMD website, 2015. Cited on p. 1036

[34] AMD, "Radeon: Dissecting the Polaris Architecture," AMD website, 2016. Cited on p. 1038

[35] AMD, "Radeon's Next-Generation Vega Architecture," AMD website, 2017. Cited on p. 1035, 1036, 1037

[36] AMD, GPUOpen, "TressFX," *GitHub* repository, 2017. Cited on p. 642, 644, 647

[37] American Society for Photogrammetry & Remote Sensing, "LAS Specification, Version 1.4—R13," *asprs.org*, July 15, 2013. Cited on p. 573

[38] Anagnostou, Kostas, "How Unreal Renders a Frame," *Interplay of Light* blog, Oct. 24, 2017. Cited on p. 899, 905, 913

[39] Anderson, Eric A., "Building Obduction: Cyan's Custom UE4 Art Tools," *Game Developers Conference*, Mar. 2016. Cited on p. 366

[40] Andersson, Johan, "Terrain Rendering in Frostbite Using Procedural Shader Splatting," *SIGGRAPH Advanced Real-Time Rendering in 3D Graphics and Games course*, Aug. 2007. Cited on p. 43, 175, 218, 877, 878

[41] Andersson, Johan, and Daniel Johansson, "Shadows & Decals: D3D10 Techniques from Frostbite," *Game Developers Conference*, Mar. 2009. Cited on p. 245, 246, 247

[42] Andersson, Johan, "Parallel Graphics in Frostbite—Current & Future," *SIGGRAPH Beyond Programmable Shading course*, Aug. 2009. Cited on p. 893

[43] Andersson, Johan, "DirectX 11 Rendering in *Battlefield 3*," *Game Developers Conference*, Mar. 2011. Cited on p. 147, 888, 890, 893, 896

[44] Andersson, Johan, "Shiny PC Graphics in *Battlefield 3*," *GeForce LAN*, Oct. 2011. Cited on p. 569, 570, 604

[45] Andersson, Johan, "Parallel Futures of a Game Engine," *Intel Dynamic Execution Environment Symposium*, May 2012. Cited on p. 811, 812

[46] Andersson, Johan, "The Rendering Pipeline—Challenges & Next Steps," *SIGGRAPH Open Problems in Real-Time Rendering course*, Aug. 2015. Cited on p. 156, 514, 1014

[47] Andersson, Johan, and Colin Barré-Brisebois, "Shiny Pixels and Beyond: Real-Time Raytracing at SEED," *Game Developers Conference*, Mar. 2018. Cited on p. 1044

[48] Andersson, M., J. Hasselgren, R. Toth, and T. Akenine-Möller, "Adaptive Texture Space Shading for Stochastic Rendering," *Computer Graphics Forum*, vol. 33, no. 2, pp. 341–350, 2014. Cited on p. 910, 911

[49] Andersson, Magnus, *Algorithmic Improvements for Stochastic Rasterization & Depth Buffering*, PhD thesis, Lund University, Oct. 2015. Cited on p. 1015

[50] Andersson, M., J. Hasselgren, and T. Akenine-Möller, "Masked Depth Culling for Graphics Hardware," *ACM Transactions on Graphics*, vol. 34, no. 6, pp. 188:1–188:9, 2015. Cited on p. 849, 1015, 1016

[51] Andreev, Dmitry, "Real-Time Frame Rate Up-Conversion for Video Games," in *ACM SIGGRAPH 2010 Talks*, ACM, article no. 16, July 2010. Cited on p. 537, 542

[52] Andreev, Dmitry, "Anti-Aliasing from a Different Perspective," *Game Developers Conference*, Mar. 2011. Cited on p. 147

[53] Anguelov, Bobby, "DirectX10 Tutorial 10: Shadow Mapping Part 2," *Taking Initiative* blog, May 25, 2011. Cited on p. 249

[54] Annen, Thomas, Jan Kautz, Frédo Durand, and Hans-Peter Seidel, "Spherical Harmonic Gradients for Mid-Range Illumination," in *Proceedings of the Fifteenth Eurographics Conference on Rendering Techniques*, Eurographics Association, pp. 331–336, June 2004. Cited on p. 488

[55] Annen, Thomas, Tom Mertens, Philippe Bekaert, Hans-Peter Seidel, and Jan Kautz, "Convolution Shadow Maps," in *Proceedings of the 18th Eurographics Conference on Rendering Techniques*, Eurographics Association, pp. 51–60, June 2007. Cited on p. 255

[56] Annen, Thomas, Tom Mertens, Hans-Peter Seidel, Eddy Flerackers, and Jan Kautz, "Exponential Shadow Maps," in *Graphics Interface 2008*, Canadian Human-Computer Communications Society, pp. 155–161, May 2008. Cited on p. 256

[57] Annen, Thomas, Zhao Dong, Tom Mertens, Philippe Bekaert, Hans-Peter Seidel, and Jan Kautz, "Real-Time, All-Frequency Shadows in Dynamic Scenes," *ACM Transactions on Graphics*, vol. 27, no. 3, article no. 34, Aug. 2008. Cited on p. 257

[58] Ansari, Marwan Y., "Image Effects with DirectX 9 Pixel Shaders," in Wolfgang Engel, ed., *ShaderX^2: Shader Programming Tips and Tricks with DirectX 9*, pp. 481–518, Wordware, 2004. Cited on p. 521, 665

[59] Answer, James, "Fast and Flexible: Technical Art and Rendering for The Unknown," *Game Developers Conference*, Mar. 2016. Cited on p. 710, 787, 805, 931, 934, 936, 938

[60] Antoine, François, Ryan Brucks, Brian Karis, and Gavin Moran, "The Boy, the Kite and the 100 Square Mile Real-Time Digital Backlot," in *ACM SIGGRAPH 2015 Talks*, ACM, article no. 20, Aug. 2015. Cited on p. 493

[61] Antonio, Franklin, "Faster Line Segment Intersection," in David Kirk, ed., *Graphics Gems III*, pp. 199–202, Academic Press, 1992. Cited on p. 988, 989

[62] Antonov, Michael, "Asynchronous Timewarp Examined," *Oculus Developer Blog*, Mar. 3, 2015. Cited on p. 936, 937

[63] Apodaca, Anthony A., and Larry Gritz, *Advanced RenderMan: Creating CGI for Motion Pictures*, Morgan Kaufmann, 1999. Cited on p. 37, 909

[64] Apodaca, Anthony A., "How PhotoRealistic RenderMan Works," in *Advanced RenderMan: Creating CGI for Motion Pictures*, Morgan Kaufmann, Chapter 6, 1999. Also in *SIGGRAPH Advanced RenderMan 2: To RI_INFINITY and Beyond course*, July 2000. Cited on p. 51

[65] Apple, "ARKit," Apple developer website. Cited on p. 918

[66] Apple, "OpenGL ES Programming Guide for iOS," Apple developer website. Cited on p. 177, 702, 713

[67] de Araújo, B. R., D. S. Lopes, P. Jepp, J. A. Jorge, and B. Wyvill, "A Survey on Implicit Surface Polygonization," *ACM Computing Surveys*, vol. 47, no. 4, pp. 60:1–60:39, 2015. Cited on p. 586, 683, 751, 753, 781, 944

[68] Arge, L., G. S. Brodal, and R. Fagerberg, "Cache-Oblivious Data Structures," in *Handbook of Data Structures*, CRC Press, Chapter 34, 2005. Cited on p. 827

[69] ARM Limited, "ARM®Mali™Application Developer Best Practices, Version 1.0," ARM documentation, Feb. 27, 2017. Cited on p. 48, 798, 1029

[70] Arvo, James, "A Simple Method for Box-Sphere Intersection Testing," in Andrew S. Glassner, ed., *Graphics Gems*, Academic Press, pp. 335–339, 1990. Cited on p. 977, 984

[71] Arvo, James, "Ray Tracing with Meta-Hierarchies," *SIGGRAPH Advanced Topics in Ray Tracing course*, Aug. 1990. Cited on p. 953

[72] Arvo, James, ed., *Graphics Gems II*, Academic Press, 1991. Cited on p. 102, 991

[73] Arvo, James, "The Irradiance Jacobian for Partially Occluded Polyhedral Sources," in *SIGGRAPH '94: Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, ACM, pp. 343–350, July 1994. Cited on p. 379

[74] Arvo, James, "Applications of Irradiance Tensors to the Simulation of non-Lambertian Phenomena," in *SIGGRAPH '95: Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, ACM, pp. 335–342, Aug. 1995. Cited on p. 389, 390

[75] Asanovic, Krste, et al., "The Landscape of Parallel Computing Research: A View from Berkeley," Technical Report No. UCB/EECS-2006-183, EECS Department, University of California, Berkeley, 2006. Cited on p. 806, 815

[76] Ashdown, Ian, *Radiosity: A Programmer's Perspective*, John Wiley & Sons, Inc., 1994. Cited on p. 271, 442

[77] Ashikhmin, Michael, and Peter Shirley, "An Anisotropic Phong Light Reflection Model," Technical Report UUCS-00-014, Computer Science Department, University of Utah, June 2000. Cited on p. 352

[78] Ashikhmin, Michael, Simon Premože, and Peter Shirley, "A Microfacet-Based BRDF Generator," in *SIGGRAPH '00: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, ACM Press/Addison-Wesley Publishing Co., pp. 67–74, July 2000. Cited on p. 328, 335, 357

[79] Ashikhmin, Michael, "Microfacet-Based BRDFs," *SIGGRAPH State of the Art in Modeling and Measuring of Surface Reflection course*, Aug. 2001. Cited on p. 329

[80] Ashikhmin, Michael, Abhijeet Ghosh, "Simple Blurry Reflections with Environment Maps," *journal of graphics tools*, vol. 7, no. 4, pp. 3–8, 2002. Cited on p. 417, 418

[81] Ashikhmin, Michael, and Simon Premože, "Distribution-Based BRDFs," Technical Report, 2007. Cited on p. 357

[82] Asirvatham, Arul, and Hugues Hoppe, "Terrain Rendering Using GPU-Based Geometry Clipmaps," in Matt Pharr, ed., *GPU Gems 2*, Addison-Wesley, pp. 27–45, 2005. Cited on p. 872, 873

[83] Assarsson, Ulf, and Tomas Möller, "Optimized View Frustum Culling Algorithms for Bounding Boxes," *journal of graphics tools*, vol. 5, no. 1, pp. 9–22, 2000. Cited on p. 836, 982, 986

[84] Atanasov, Asen, and Vladimir Koylazov, "A Practical Stochastic Algorithm for Rendering Mirror-Like Flakes," in *ACM SIGGRAPH 2016 Talks*, article no. 67, July 2016. Cited on p. 372

[85] Austin, Michael, "Voxel Surfing," *Game Developers Conference*, Mar. 2016. Cited on p. 586

[86] Bærentzen, J. Andreas, Steen Lund Nielsen, Mikkel Gjøl, and Bent D. Larsen, "Two Methods for Antialiased Wireframe Drawing with Hidden Line Removal," in *SCCG '08 Proceedings of the 24th Spring Conference on Computer Graphics*, ACM, pp. 171–177, Apr. 2008. Cited on p. 673, 675

[87] Baert, J., A. Lagae, and Ph. Dutré, "Out-of-Core Construction of Sparse Voxel Octrees," *Computer Graphics Forum*, vol. 33, no. 6, pp. 220–227, 2014. Cited on p. 579, 582

[88] Bagnell, Dan, "Graphics Tech in Cesium—Vertex Compression," *Cesium* blog, May 18, 2015. Cited on p. 715

[89] Bahar, E., and S. Chakrabarti, "Full-Wave Theory Applied to Computer-Aided Graphics for 3D Objects," *IEEE Computer Graphics and Applications*, vol. 7, no. 7, pp. 46–60, July 1987. Cited on p. 361

[90] Bahnassi, Homam, and Wessam Bahnassi, "Volumetric Clouds and Mega-Particles," in Wolfgang Engel, ed., *ShaderX*^{5}, Charles River Media, pp. 295–302, 2006. Cited on p. 521, 556

[91] Baker, Dan, "Advanced Lighting Techniques," *Meltdown 2005*, July 2005. Cited on p. 369

[92] Baker, Dan, and Yannis Minadakis, "Firaxis' Civilization V: A Case Study in Scalable Game Performance," *Game Developers Conference*, Mar. 2010. Cited on p. 812

[93] Baker, Dan, "Spectacular Specular—LEAN and CLEAN Specular Highlights," *Game Developers Conference*, Mar. 2011. Cited on p. 370

[94] Baker, Dan, "Object Space Lighting," *Game Developers Conference*, Mar. 2016. Cited on p. 911

[95] Bako, Steve, Thijs Vogels, Brian McWilliams, Mark Meyer, Jan Novák, Alex Harvill, Pradeep Sen, Tony DeRose, and Fabrice Rousselle, "Kernel-Predicting Convolutional Networks for Denoising Monte Carlo Renderings," *ACM Transactions on Graphics*, vol. 36, no. 4, article no. 97, 2017. Cited on p. 511, 1043

[96] Baldwin, Doug, and Michael Weber, "Fast Ray-Triangle Intersections by Coordinate Transformation," *Journal of Computer Graphics Techniques*, vol. 5, no. 3, pp. 39–49, 2016. Cited on p. 962

[97] Balestra, C., and P.-K. Engstad, "The Technology of Uncharted: Drake's Fortune," *Game Developers Conference*, Mar. 2008. Cited on p. 893

[98] Banks, David, "Illumination in Diverse Codimensions," in *SIGGRAPH '94: Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, ACM, pp. 327–334, July 1994. Cited on p. 359

[99] Barb, C., "Texture Streaming in *Titanfall 2*," *Game Developers Conference*, Feb.–Mar. 2017. Cited on p. 869

[100] Barber, C. B., D. P. Dobkin, and H. Huhdanpaa, "The Quickhull Algorithm for Convex Hull," Technical Report GCG53, Geometry Center, July 1993. Cited on p. 950

[101] Barequet, G., and G. Elber, "Optimal Bounding Cones of Vectors in Three Dimensions," *Information Processing Letters*, vol. 93, no. 2, pp. 83–89, 2005. Cited on p. 834

[102] Barkans, Anthony C., "Color Recovery: True-Color 8-Bit Interactive Graphics," *IEEE Computer Graphics and Applications*, vol. 17, no. 1, pp. 67–77, Jan./Feb. 1997. Cited on p. 1010

[103] Barkans, Anthony C., "High-Quality Rendering Using the Talisman Architecture," in *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Workshop on Graphics Hardware*, ACM, pp. 79–88, Aug. 1997. Cited on p. 189

[104] Barla, Pascal, Joëlle Thollot, and Lee Markosian, "X-Toon: An Extended Toon Shader," in *Proceedings of the 4th International Symposium on Non-Photorealistic Animation and Rendering*, ACM, pp. 127–132, 2006. Cited on p. 654

[105] Barré-Brisebois, Colin, and Marc Bouchard, "Approximating Translucency for a Fast, Cheap and Convincing Subsurface Scattering Look," *Game Developers Conference*, Feb.–Mar. 2011. Cited on p. 639, 640

[106] Barré-Brisebois, Colin, and Stephen Hill, "Blending in Detail," *Self-Shadow* blog, July 10, 2012. Cited on p. 366, 371, 890

[107] Barré-Brisebois, Colin, "Hexagonal Bokeh Blur Revisited," *ZigguratVertigo's Hideout* blog, Apr. 17, 2017. Cited on p. 531

[108] Barrett, Sean, "Blend Does Not Distribute Over Lerp," *Game Developer*, vol. 11, no. 10, pp. 39–41, Nov. 2004. Cited on p. 160

[109] Barrett, Sean, "Sparse Virtual Textures," *Game Developers Conference*, Mar. 2008. Cited on p. 867

[110] Barringer, R., M. Andersson, and T. Akenine-Möller, "Ray Accelerator: Efficient and Flexible Ray Tracing on a Heterogeneous Architecture," *Computer Graphics Forum*, vol. 36, no. 8, pp. 166–177, 2017. Cited on p. 1006

[111] Bartels, Richard H., John C. Beatty, and Brian A. Barsky, *An Introduction to Splines for use in Computer Graphics and Geometric Modeling*, Morgan Kaufmann, 1987. Cited on p. 732, 734, 749, 754, 756, 781

[112] Barzel, Ronen, ed., *Graphics Tools—The jgt Editors' Choice*, A K Peters, Ltd., 2005. Cited on p. 1058, 1064, 1065, 1084, 1091, 1111, 1115, 1133, 1138, 1143

[113] Batov, Vladimir, "A Quick and Simple Memory Allocator," *Dr. Dobbs's Portal*, Jan. 1, 1998. Cited on p. 793

[114] Baum, Daniel R., Stephen Mann, Kevin P. Smith, and James M. Winget, "Making Radiosity Usable: Automatic Preprocessing and Meshing Techniques for the Generation of Accurate Radiosity Solutions," *Computer Graphics (SIGGRAPH '91 Proceedings)*, vol. 25, no. 4, pp. 51–60, July 1991. Cited on p. 689

[115] Bavoil, Louis, Steven P. Callahan, Aaron Lefohn, João L. D. Comba, and Cláudio T. Silva, "Multi-Fragment Effects on the GPU Using the k-Buffer," in *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games*, ACM, pp. 97–104, Apr.–May 2007. Cited on p. 156, 624, 626

[116] Bavoil, Louis, Steven P. Callahan, and Cláudio T. Silva, "Robust Soft Shadow Mapping with Backprojection and Depth Peeling," *journal of graphics tools*, vol. 13, no. 1, pp. 16–30, 2008. Cited on p. 238, 252

[117] Bavoil, Louis, "Advanced Soft Shadow Mapping Techniques," *Game Developers Conference*, Feb. 2008. Cited on p. 256

[118] Bavoil, Louis, and Kevin Myers, "Order Independent Transparency with Dual Depth Peeling," NVIDIA White Paper, Feb. 2008. Cited on p. 155, 157

[119] Bavoil, Louis, and Miguel Sainz, and Rouslan Dimitrov, "Image-Space Horizon-Based Aambient Occlusion," in *ACM SIGGRAPH 2008 Talks*, ACM, article no. 22, Aug. 2008. Cited on p. 460

[120] Bavoil, Louis, and Jon Jansen, "Particle Shadows and Cache-Efficient Post-Processing," *Game Developers Conference*, Mar. 2013. Cited on p. 570

[121] Bavoil, Louis, and Iain Cantlay, "SetStablePowerState.exe: Disabling GPU Boost on Windows 10 for more deterministic timestamp queries on NVIDIA GPUs," *NVIDIA GameWorks* blog, Sept. 14, 2016. Cited on p. 789

[122] Beacco, A., N. Pelechano, and C. Andújar, "A Survey of Real-Time Crowd Rendering," *Computer Graphics Forum*, vol. 35, no. 8, pp. 32–50, 2016. Cited on p. 563, 566, 567, 587, 798

[123] Bec, Xavier, "Faster Refraction Formula, and Transmission Color Filtering," *Ray Tracing News*, vol. 10, no. 1, Jan. 1997. Cited on p. 627

[124] Beckmann, Petr, and André Spizzichino, *The Scattering of Electromagnetic Waves from Rough Surfaces*, Pergamon Press, 1963. Cited on p. 331, 338

[125] Beeler, Dean, and Anuj Gosalia, "Asynchronous Timewarp on Oculus Rift," *Oculus Developer Blog*, Mar. 25, 2016. Cited on p. 935, 937

[126] Beeler, Dean, Ed Hutchins, and Paul Pedriana, "Asynchronous Spacewarp," *Oculus Developer Blog*, Nov. 10, 2016. Cited on p. 937

[127] Beers, Andrew C., Maneesh Agrawala, and Navin Chaddha, "Rendering from Compressed Textures," in *SIGGRAPH '96: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, ACM, pp. 373–378, Aug. 1996. Cited on p. 192

[128] Behrendt, S., C. Colditz, O. Franzke, J. Kopf, and O. Deussen, "Realistic Real-Time Rendering of Landscapes Using Billboard Clouds," *Computer Graphics Forum*, vol. 24, no. 3, pp. 507–516, 2005. Cited on p. 563

[129] Belcour, Laurent, and Pascal Barla, "A Practical Extension to Microfacet Theory for the Modeling of Varying Iridescence," *ACM Transactions on Graphics (SIGGRAPH 2017)*, vol. 36, no. 4, pp. 65:1–65:14, July 2017. Cited on p. 363

[130] Bénard, Pierre, Adrien Bousseau, and Jöelle Thollot, "State-of-the-Art Report on Temporal Coherence for Stylized Animations," *Computer Graphics Forum*, vol. 30, no. 8, pp. 2367–2386, 2011. Cited on p. 669, 678

[131] Bénard, Pierre, Lu Jingwan, Forrester Cole, Adam Finkelstein, and Jöelle Thollot, "Active Strokes: Coherent Line Stylization for Animated 3D Models," in *Proceedings of the International Symposium on Non-Photorealistic Animation and Rendering*, Eurographics Association, pp. 37–46, 2012. Cited on p. 669

[132] Bénard, Pierre, Aaron Hertzmann, and Michael Kass, "Computing Smooth Surface Contours with Accurate Topology," *ACM Transactions on Graphics*, vol. 33, no. 2, pp. 19:1–19:21, 2014. Cited on p. 656, 667

[133] Benson, David, and Joel Davis, "Octree Textures," *ACM Transactions on Graphics (SIGGRAPH 2002)*, vol. 21, no. 3, pp. 785–790, July 2002. Cited on p. 190

[134] Bentley, Adrian, "*inFAMOUS Second Son* Engine Postmortem," *Game Developers Conference*, Mar. 2014. Cited on p. 54, 490, 871, 884, 904

[135] de Berg, M., M. van Kreveld, M. Overmars, and O. Schwarzkopf, *Computational Geometry—Algorithms and Applications*, Third Edition, Springer-Verlag, 2008. Cited on p. 685, 699, 967

[136] van den Bergen, G., "Efficient Collision Detection of Complex Deformable Models Using AABB Trees," *journal of graphics tools*, vol. 2, no. 4, pp. 1–13, 1997. Also collected in [112]. Cited on p. 821

[137] Berger, Matthew, Andrea Tagliasacchi, Lee M. Seversky, Pierre Alliez, Gaël Guennebaud, Joshua A. Levine, Andrei Sharf, and Claudio T. Silva, "A Survey of Surface Reconstruction from Point Clouds," *Computer Graphics Forum*, vol. 36, no. 1, pp. 301–329, 2017. Cited on p. 573, 683

[138] Beyer, Johanna, Markus Hadwiger, and Hanspeter Pfister, "State-of-the-Art in GPU-Based Large-Scale Volume Visualization," *Computer Graphics Forum*, vol. 34, no. 8, pp. 13–37, 2015. Cited on p. 586

[139] Bezrati, Abdul, "Real-Time Lighting via Light Linked List," *SIGGRAPH Advances in Real-Time Rendering in Games course*, Aug. 2014. Cited on p. 893, 903

[140] Bezrati, Abdul, "Real-Time Lighting via Light Linked List," in Wolfgang Engel, ed., *GPU Pro*^{6}, CRC Press, pp. 183–193, 2015. Cited on p. 893, 903

[141] Bier, Eric A., and Kenneth R. Sloan, Jr., "Two-Part Texture Mapping," *IEEE Computer Graphics and Applications*, vol. 6, no. 9, pp. 40–53, Sept. 1986. Cited on p. 170

[142] Biermann, Henning, Adi Levin, and Denis Zorin, "Piecewise Smooth Subdivision Surface with Normal Control," in *SIGGRAPH '00: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, ACM Press/Addison-Wesley Publishing Co., pp. 113–120, July 2000. Cited on p. 764

[143] Billeter, Markus, Erik Sintorn, and Ulf Assarsson, "Real-Time Multiple Scattering Using Light Propagation Volumes," in *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, ACM, pp. 119–126, 2012. Cited on p. 611

[144] Billeter, Markus, Ola Olsson, and Ulf Assarsson, "Tiled Forward Shading," in Wolfgang Engel, ed., *GPU Pro*^{4}, CRC Press, pp. 99–114, 2013. Cited on p. 895, 896, 914

[145] Billeter, Markus, "Many-Light Rendering on Mobile Hardware," *SIGGRAPH Real-Time Many-Light Management and Shadows with Clustered Shading course*, Aug. 2015. Cited on p. 893, 900, 903, 914

[146] Bilodeau, Bill, "Vertex Shader Tricks: New Ways to Use the Vertex Shader to Improve Performance," *Game Developers Conference*, Mar. 2014. Cited on p. 51, 87, 514, 568, 571, 798

[147] Binstock, Atman, "Optimizing VR Graphics with Late Latching," *Oculus Developer Blog*, Mar. 2, 2015. Cited on p. 938

[148] Bishop, L., D. Eberly, T. Whitted, M. Finch, and M. Shantz, "Designing a PC Game Engine," *IEEE Computer Graphics and Applications*, vol. 18, no. 1, pp. 46–53, Jan./Feb. 1998. Cited on p. 836

[149] Bitterli, Benedikt, *Benedikt Bitterli Rendering Resources*, https://benedikt-bitterli.me/resources, licensed under CC BY 3.0, https://creativecommons.org/licenses/by/3.0. Cited on p. 441, 445, 447, 449, 450

[150] Bittner, Jiří, and Jan Přikryl, "Exact Regional Visibility Using Line Space Partitioning," Technical Report TR-186-2-01-06, Institute of Computer Graphics and Algorithms, Vienna University of Technology, Mar. 2001. Cited on p. 843

[151] Bittner, Jiří, Peter Wonka, and Michael Wimmer, "Visibility Preprocessing for Urban Scenes Using Line Space Subdivision," in *Pacific Graphics 2001*, IEEE Computer Society, pp. 276–284, Oct. 2001. Cited on p. 843

[152] Bittner, Jiří, Oliver Mattausch, Ari Silvennoinen, and Michael Wimmer, "Shadow Caster Culling for Efficient Shadow Mapping," in *Symposium on Interactive 3D Graphics and Games*, ACM, pp. 81–88, 2011. Cited on p. 247

[153] Bjørge, Marius, Sam Martin, Sandeep Kakarlapudi, and Jan-Harald Fredriksen, "Efficient Rendering with Tile Local Storage," in *ACM SIGGRAPH 2014 Talks*, ACM, article no. 51, July 2014. Cited on p. 156

[154] Bjørge, Marius, "Moving Mobile Graphics," *SIGGRAPH Advanced Real-Time Shading course*, July 2016. Cited on p. 247, 265

[155] Bjorke, Kevin, "Image-Based Lighting," in Randima Fernando, ed., *GPU Gems*, Addison-Wesley, pp. 308–321, 2004. Cited on p. 500

[156] Bjorke, Kevin, "High-Quality Filtering," in Randima Fernando, ed., *GPU Gems*, Addison-Wesley, pp. 391–424, 2004. Cited on p. 515, 521

[157] Blasi, Philippe, Bertrand Le Saec, and Christophe Schlick, "A Rendering Algorithm for Discrete Volume Density Objects," *Computer Graphics Forum*, vol. 12, no. 3, pp. 201–210, 1993. Cited on p. 598

[158] Blinn, J. F., and M. E. Newell, "Texture and Reflection in Computer Generated Images," *Communications of the ACM*, vol. 19, no. 10, pp. 542–547, Oct. 1976. Cited on p. 405, 406

[159] Blinn, James F., "Models of Light Reflection for Computer Synthesized Pictures," *ACM Computer Graphics (SIGGRAPH '77 Proceedings)*, vol. 11, no. 2, pp. 192–198, July 1977. Cited on p. 331, 340, 416

[160] Blinn, James, "Simulation of Wrinkled Surfaces," *Computer Graphics (SIGGRAPH '78 Proceedings)*, vol. 12, no. 3, pp. 286–292, Aug. 1978. Cited on p. 209, 765

[161] Blinn, James F., "A Generalization of Algebraic Surface Drawing," *ACM Transactions on Graphics*, vol. 1, no. 3, pp. 235–256, 1982. Cited on p. 751

[162] Blinn, Jim, "Me and My (Fake) Shadow," *IEEE Computer Graphics and Applications*, vol. 8, no. 1, pp. 82–86, Jan. 1988. Also collected in [165]. Cited on p. 225, 227

[163] Blinn, Jim, "Hyperbolic Interpolation," *IEEE Computer Graphics and Applications*, vol. 12, no. 4, pp. 89–94, July 1992. Also collected in [165]. Cited on p. 999

[164] Blinn, Jim, "Image Compositing—Theory," *IEEE Computer Graphics and Applications*, vol. 14, no. 5, pp. 83–87, Sept. 1994. Also collected in [166]. Cited on p. 160

[165] Blinn, Jim, *Jim Blinn's Corner: A Trip Down the Graphics Pipeline*, Morgan Kaufmann, 1996. Cited on p. 27, 832, 1059

[166] Blinn, Jim, *Jim Blinn's Corner: Dirty Pixels*, Morgan Kaufmann, 1998. Cited on p. 165, 1059

[167] Blinn, Jim, "A Ghost in a Snowstorm," *IEEE Computer Graphics and Applications*, vol. 18, no. 1, pp. 79–84, Jan./Feb. 1998. Also collected in [168], Chapter 9. Cited on p. 165

[168] Blinn, Jim, *Jim Blinn's Corner: Notation, Notation, Notation*, Morgan Kaufmann, 2002. Cited on p. 165, 1059

[169] Blinn, Jim, "What Is a Pixel?" *IEEE Computer Graphics and Applications*, vol. 25, no. 5, pp. 82–87, Sept./Oct. 2005. Cited on p. 165, 280

[170] Bloomenthal, Jules, "Edge Inference with Applications to Antialiasing," *Computer Graphics (SIGGRAPH '83 Proceedings)*, vol. 17, no. 3, pp. 157–162, July 1983. Cited on p. 146

[171] Bloomenthal, Jules, "An Implicit Surface Polygonizer," in Paul S. Heckbert, ed., *Graphics Gems IV*, Academic Press, pp. 324–349, 1994. Cited on p. 753

[172] Blow, Jonathan, "Mipmapping, Part 1," *Game Developer*, vol. 8, no. 12, pp. 13–17, Dec. 2001. Cited on p. 184

[173] Blow, Jonathan, "Mipmapping, Part 2," *Game Developer*, vol. 9, no. 1, pp. 16–19, Jan. 2002. Cited on p. 184

[174] Blow, Jonathan, "Happycake Development Notes: Shadows," *Happycake Development Notes* website, Aug. 25, 2004. Cited on p. 242

[175] Blythe, David, "The Direct3D 10 System," *ACM Transactions on Graphics*, vol. 25, no. 3, pp. 724–734, July 2006. Cited on p. 29, 39, 42, 47, 48, 50, 249

[176] Bookout, David, "Programmable Blend with Pixel Shader Ordering," *Intel Developer Zone* blog, Oct. 13, 2015. Cited on p. 52

[177] Born, Max, and Emil Wolf, *Principles of Optics: Electromagnetic Theory of Propagation, Interference and Diffraction of Light*, Seventh Edition, Cambridge University Press, 1999. Cited on p. 373

[178] Borshukov, George, and J. P. Lewis, "Realistic Human Face Rendering for *The Matrix Reloaded*," in *ACM SIGGRAPH 2003 Sketches and Applications*, ACM, July 2003. Cited on p. 635

[179] Borshukov, George, and J. P. Lewis, "Fast Subsurface Scattering," *SIGGRAPH Digital Face Cloning course*, Aug. 2005. Cited on p. 635

[180] Botsch, Mario, Alexander Hornung, Matthias Zwicker, and Leif Kobbelt, "High-Quality Surface Splatting on Today's GPUs," in *Proceedings of the Second Eurographics / IEEE VGTC Symposium on Point-Based Graphics*, Eurographics Association, pp. 17–24, June 2005. Cited on p. 574

[181] Boubekeur, Tamy, Patrick Reuter, and Christophe Schlick, "Scalar Tagged PN Triangles," in *Eurographics 2005 Short Presentations*, Eurographics Association, pp. 17–20, Sept. 2005. Cited on p. 747

[182] Boubekeur, T., and Marc Alexa, "Phong Tessellation," *ACM Transactions on Graphics*, vol. 27, no. 5, pp. 141:1–141:5, 2008. Cited on p. 748

[183] Boulton, Mike, "Static Lighting Tricks in *Halo 4*," *Game Developers Conference*, Mar. 2013. Cited on p. 486

[184] Bouthors, Antoine, Fabrice Neyret, Nelson Max, Eric Bruneton, and Cyril Crassin, "Interactive Multiple Anisotropic Scattering in Clouds," in *Proceedings of the 2008 Symposium on Interactive 3D Graphics and Games*, ACM, pp. 173–182, 2008. Cited on p. 618, 619, 620

[185] Bowles, H., K. Mitchell, B. Sumner, J. Moore, and M. Gross, "Iterative Image Warping," *Computer Graphics Forum*, vol. 31, no. 2, pp. 237–246, 2012. Cited on p. 523

[186] Bowles, H., "Oceans on a Shoestring: Shape Representation, Meshing and Shading," *SIGGRAPH Advances in Real-Time Rendering in Games course*, July 2013. Cited on p. 878

[187] Bowles, Huw, and Beibei Wang, "Sparkly but not too Sparkly! A Stable and Robust Procedural Sparkle Effect," *SIGGRAPH Advances in Real-Time Rendering in Games course*, Aug. 2015. Cited on p. 372

[188] Box, Harry, *Set Lighting Technician's Handbook: Film Lighting Equipment, Practice, and Electrical Distribution*, Fourth Edition, Focal Press, 2010. Cited on p. 435

[189] Boyd, Stephen, and Lieven Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004. Freely downloadable. Cited on p. 946

[190] Brainerd, W., T. Foley, M. Kraemer, H. Moreton, and M. Nießner, "Efficient GPU Rendering of Subdivision Surfaces Using Adaptive Quadtrees," *ACM Transactions on Graphics*, vol. 35, no. 4, pp. 113:1–113:12, 2016. Cited on p. 779, 780

[191] Bratt, I., "The ARM Mali T880 Mobile GPU," *Hot Chips* website, 2015. Cited on p. 1027

[192] Brawley, Zoe, and Natalya Tatarchuk, "Parallax Occlusion Mapping: Self-Shadowing, Perspective-Correct Bump Mapping Using Reverse Height Map Tracing," in Wolfgang Engel, ed., *ShaderX*^{3}, Charles River Media, pp. 135–154, Nov. 2004. Cited on p. 217

[193] Bredow, Rob, "Fur in *Stuart Little*," *SIGGRAPH Advanced RenderMan 2: To RI_INFINITY and Beyond course*, July 2000. Cited on p. 382, 633

[194] Brennan, Chris, "Accurate Environment Mapped Reflections and Refractions by Adjusting for Object Distance," in Wolfgang Engel, ed., *Direct3D ShaderX: Vertex & Pixel Shader Tips and Techniques*, Wordware, pp. 290–294, May 2002. Cited on p. 500

[195] Brennan, Chris, "Diffuse Cube Mapping," in Wolfgang Engel, ed., *Direct3D ShaderX: Vertex & Pixel Shader Tips and Techniques*, Wordware, pp. 287–289, May 2002. Cited on p. 427

[196] Breslav, Simon, Karol Szerszen, Lee Markosian, Pascal Barla, and Joëlle Thollot, "Dynamic 2D Patterns for Shading 3D Scenes," *ACM Transactions on Graphics*, vol. 27, no. 3, pp. 20:1–20:5, 2007. Cited on p. 670

[197] Bridson, Robert, *Fluid Simulation for Computer Graphics*, Second Edition, CRC Press, 2015. Cited on p. 571, 649

[198] Brinck, Waylon, and Andrew Maximov, "The Technical Art of *Uncharted 4*," *SIGGRAPH production session*, July 2016. Cited on p. 290

[199] Brinkmann, Ron, *The Art and Science of Digital Compositing*, Morgan Kaufmann, 1999. Cited on p. 149, 151, 159, 160

[200] Brisebois, Vincent, and Ankit Patel, "Profiling the AI Performance Boost in OptiX 5," *NVIDIA News Center*, July 31, 2017. Cited on p. 511, 1043, 1044

[201] Brown, Alistair, "Visual Effects in *Star Citizen*," *Game Developers Conference*, Mar. 2015. Cited on p. 366

[202] Brown, Gary S., "Shadowing by Non-gaussian Random Surfaces," *IEEE Transactions on Antennas and Propagation*, vol. 28, no. 6, pp. 788–790, 1980. Cited on p. 334

[203] Bruneton, Eric, and Fabrice Neyret, "Precomputed Atmospheric Scattering," *Computer Graphics Forum*, vol. 27, no. 4, pp. 1079–1086, 2008. Cited on p. 614, 615, 616

[204] Bruneton, Eric, Fabrice Neyret, and Nicolas Holzschuch, "Real-Time Realistic Ocean Lighting Using Seamless Transitions from Geometry to BRDF," *Computer Graphics Forum*, vol. 29, no. 2, pp. 487–496, 2010. Cited on p. 372

[205] Bruneton, Eric, and Fabrice Neyret, "A Survey of Non-linear Pre-filtering Methods for Efficient and Accurate Surface Shading," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 2, pp. 242–260, 2012. Cited on p. 372

[206] Buades, Jose María, Jesús Gumbau, and Miguel Chover, "Separable Soft Shadow Mapping," *The Visual Computer*, vol. 32, no. 2, pp. 167–178, Feb. 2016. Cited on p. 252

[207] Buchanan, J. W., and M. C. Sousa, "The Edge Buffer: A Data Structure for Easy Silhouette Rendering," in *Proceedings of the 1st International Symposium on Non-photorealistic Animation and Rendering*, ACM, pp. 39–42, June 2000. Cited on p. 666

[208] Bukowski, Mike, Padraic Hennessy, Brian Osman, and Morgan McGuire, "Scalable High Quality Motion Blur and Ambient Occlusion," *SIGGRAPH Advances in Real-Time Rendering in 3D Graphics and Games course*, Aug. 2012. Cited on p. 540, 542, 543

[209] Bukowski, Mike, Padraic Hennessy, Brian Osman, and Morgan McGuire, "The *Skylanders* SWAP Force Depth-of-Field Shader," in Wolfgang Engel, ed., *GPU Pro*^{4}, CRC Press, pp. 175–184, 2013. Cited on p. 529, 530, 532, 533

[210] Bunnell, Michael, "Dynamic Ambient Occlusion and Indirect Lighting," in Matt Pharr, ed., *GPU Gems 2*, Addison-Wesley, pp. 223–233, 2005. Cited on p. 454, 497

[211] van der Burg, John, "Building an Advanced Particle System," *Gamasutra*, June 2000. Cited on p. 571

[212] Burley, Brent, "Shadow Map Bias Cone and Improved Soft Shadows: Disney Bonus Section," *SIGGRAPH RenderMan for Everyone course*, Aug. 2006. Cited on p. 249, 250

[213] Burley, Brent, and Dylan Lacewell, "Ptex: Per-Face Texture Mapping for Production Rendering," in *Proceedings of the Nineteenth Eurographics Conference on Rendering*, Eurographics Association, pp. 1155–1164, 2008. Cited on p. 191

[214] Burley, Brent, "Physically Based Shading at Disney," *SIGGRAPH Practical Physically Based Shading in Film and Game Production course*, Aug. 2012. Cited on p. 325, 336, 340, 342, 345, 353, 354, 357, 364

[215] Burley, Brent, "Extending the Disney BRDF to a BSDF with Integrated Subsurface Scattering," *SIGGRAPH Physically Based Shading in Theory and Practice course*, Aug. 2015. Cited on p. 354

[216] Burns, Christopher A., Kayvon Fatahalian, and William R. Mark, "A Lazy Object-Space Shading Architecture with Decoupled Sampling," in *Proceedings of the Conference on High-Performance Graphics*, Eurographics Association, pp. 19–28, June 2010. Cited on p. 910

[217] Burns, C. A., and W. A. Hunt, "The Visibility Buffer: A Cache-Friendly Approach to Deferred Shading," *Journal of Computer Graphics Techniques*, vol. 2, no. 2, pp. 55–69, 2013. Cited on p. 905, 906

[218] Cabello, Ricardo, et al., *Three.js source code*, Release r89, Dec. 2017. Cited on p. 41, 50, 115, 189, 201, 407, 485, 552, 628

[219] Cabral, Brian, and Leith (Casey) Leedom, "Imaging Vector Fields Using Line Integral Convolution," in *SIGGRAPH '93: Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, ACM, pp. 263–270, Aug. 1993. Cited on p. 538

[220] Caillaud, Florian, Vincent Vidal, Florent Dupont, and Guillaume Lavoué, "Progressive Compression of Arbitrary Textured Meshes," *Computer Graphics Forum*, vol. 35, no. 7, pp. 475–484, 2016. Cited on p. 709

[221] Calver, Dean, "Vertex Decompression in a Shader," in Wolfgang Engel, ed., *Direct3D ShaderX: Vertex & Pixel Shader Tips and Techniques*, Wordware, pp. 172–187, May 2002. Cited on p. 713

[222] Calver, Dean, "Photo-Realistic Deferred Lighting," *Beyond3D.com* website, July 30, 2003. Cited on p. 883, 884, 886

[223] Calver, Dean, "Accessing and Modifying Topology on the GPU," in Wolfgang Engel, ed., *ShaderX*^{3}, Charles River Media, pp. 5–19, 2004. Cited on p. 703

[224] Calver, Dean, "Deferred Lighting on PS 3.0 with High Dynamic Range," in Wolfgang Engel, ed., *ShaderX*^{3}, Charles River Media, pp. 97–105, 2004. Cited on p. 288

[225] Cantlay, Iain, and Andrei Tatarinov, "From Terrain to Godrays: Better Use of DX11," *Game Developers Conference*, Mar. 2014. Cited on p. 44, 569

[226] Card, Drew, and Jason L. Mitchell, "Non-Photorealistic Rendering with Pixel and Vertex Shaders," in Wolfgang Engel, ed., *Direct3D ShaderX: Vertex & Pixel Shader Tips and Techniques*, Wordware, pp. 319–333, May 2002. Cited on p. 662, 668

[227] Carling, Richard, "Matrix Inversion," in Andrew S. Glassner, ed., *Graphics Gems*, Academic Press, pp. 470–471, 1990. Cited on p. 68

[228] Carmack, John, "Latency Mitigation Strategies," *AltDevBlog*, Feb. 22, 2013. Cited on p. 920, 936, 937

[229] do Carmo, Manfred P., *Differential Geometry of Curves and Surfaces*, Prentice-Hall, Inc., 1976. Cited on p. 81

[230] Carpenter, Loren, "The A-Buffer, an Antialiased Hidden Surface Method," *Computer Graphics (SIGGRAPH '84 Proceedings)*, vol. 18, no. 3, pp. 103–108, July 1984. Cited on p. 155, 626

[231] Carpentier, Giliam, and Kohei Ishiyama, "*Decima*, Advances in Lighting and AA," *SIGGRAPH Advances in Real-Time Rendering in Games course*, Aug. 2017. Cited on p. 146, 148, 386, 805

[232] Carucci, Francesco, "Inside Geometry Instancing," in Matt Pharr, ed., *GPU Gems 2*, Addison-Wesley, pp. 47–67, 2005. Cited on p. 797

[233] Castaño, Ignacio, "Lightmap Parameterization,' *The Witness Blog*, Mar. 30, 2010. Cited on p. 486

[234] Castaño, Ignacio, "Computing Alpha Mipmaps," *The Witness Blog*, Sept. 9, 2010. Cited on p. 204, 206

[235] Castaño, Ignacio, "Shadow Mapping Summary—Part 1,' *The Witness Blog*, Sept. 23, 2013. Cited on p. 249, 250, 265

[236] Catmull, E., and R. Rom, "A Class of Local Interpolating Splines," in R. Barnhill & R. Riesenfeld, eds., *Computer Aided Geometric Design*, Academic Press, pp. 317–326, 1974. Cited on p. 731

[237] Catmull, E., *A Subdivision Algorithm for Computer Display of Curved Surfaces*, PhD thesis, University of Utah, Dec. 1974. Cited on p. 1048

[238] Catmull, Edwin, "Computer Display of Curved Surfaces," in *Proceedings of the IEEE Conference on Computer Graphics, Pattern Recognition and Data Structures*, IEEE Press, pp. 11–17, May 1975. Cited on p. 24

[239] Catmull, E., and J. Clark, "Recursively Generated B-Spline Surfaces on Arbitrary Topological Meshes," *Computer-Aided Design*, vol. 10, no. 6, pp. 350–355, Sept. 1978. Cited on p. 761, 762

[240] Cebenoyan, Cem, "Graphics Pipeline Performance," in Randima Fernando, ed., *GPU Gems*, Addison-Wesley, pp. 473–486, 2004. Cited on p. 787, 802, 815, 853

[241] Cebenoyan, Cem, "Real Virtual Texturing—Taking Advantage of DirectX11.2 Tiled Resources," *Game Developers Conference*, Mar. 2014. Cited on p. 246, 263, 867

[242] Celes, Waldemar, and Frederico Abraham, "Fast and Versatile Texture-Based Wireframe Rendering," *The Visual Computer*, vol. 27, no. 10, pp. 939–948, 2011. Cited on p. 674

[243] Cerezo, Eva, Frederic Pérez, Xavier Pueyo, Francisco J. Seron, and François X. Sillion, "A Survey on Participating Media Rendering Techniques," *The Visual Computer*, vol. 21, no. 5, pp. 303–328, June 2005. Cited on p. 590

[244] *The Cesium Blog*, http://cesiumjs.org/blog/, 2017. Cited on p. 879

[245] Chabert, Charles-Félix, Wan-Chun Ma, Tim Hawkins, Pieter Peers, and Paul Debevec, "Fast Rendering of Realistic Faces with Wavelength Dependent Normal Maps," in *ACM SIGGRAPH 2007 Posters*, ACM, article no. 183, Aug. 2007. Cited on p. 634

[246] Chaikin, G., "An Algorithm for High Speed Curve Generation," *Computer Graphics and Image Processing*, vol. 4, no. 3, pp. 346–349, 1974. Cited on p. 754

[247] Chaitanya, Chakravarty R. Alla, Anton S. Kaplanyan, Christoph Schied, Marco Salvi, Aaron Lefohn, Derek Nowrouzezahrai, and Timo Aila, "Interactive Reconstruction of Monte Carlo Image Sequences Using a Recurrent Denoising Autoencoder," *ACM Transactions on Graphics*, vol. 36, no. 4, article no. 98, pp. 2017. Cited on p. 511, 1043

[248] Chajdas, Matthäus G., Christian Eisenacher, Marc Stamminger, and Sylvain Lefebvre, "Virtual Texture Mapping 101," in Wolfgang Engel, ed., *GPU Pro*, A K Peters, Ltd., pp. 185–195, 2010. Cited on p. 867

[249] Chajdas, Matthäus G., "D3D12 and Vulkan: Lessons Learned," *Game Developers Conference*, Mar. 2016. Cited on p. 40, 806, 814

[250] Chan, Danny, and Bryan Johnston, "Style in Rendering: The History and Technique Behind *Afro Samurai*'s Look," *Game Developers Conference*, Mar. 2009. Cited on p. 652, 658, 664

[251] Chan, Danny, "Real-World Measurements for *Call of Duty: Advanced Warfare*," in *SIGGRAPH Physically Based Shading in Theory and Practice course*, Aug. 2015. Cited on p. 349, 355

[252] Chan, Eric, and Frédo Durand, "Fast Prefiltered Lines," in Matt Pharr, ed., *GPU Gems 2*, Addison-Wesley, pp. 345–359, 2005. Cited on p. 133

[253] Chandrasekhar, Subrahmanyan, *Radiative Transfer*, Oxford University Press, 1950. Cited on p. 352

[254] Chang, Chia-Tche, Bastien Gorissen, and Samuel Melchior, "Fast Oriented Bounding Box Optimization on the Rotation Group $SO(3, \mathbb{R})$," *ACM Transactions on Graphics*, vol. 30, no. 5, pp. 122:1–122:16, Oct. 2011. Cited on p. 951

[255] Chang, Chun-Fa, Gary Bishop, and Anselmo Lastra, "LDI Tree: A Hierarchical Representation for Image-Based Rendering," in *SIGGRAPH '99: Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, ACM Press/Addison-Wesley Publishing Co., pp. 291–298, Aug. 1999. Cited on p. 565

[256] Chen, G. P. Sander, D. Nehab, L. Yang, and L. Hu, "Depth-Presorted Triangle Lists," *ACM Transactions on Graphics*, vol. 31, no. 6, pp. 160:1–160:9, 2016. Cited on p. 831

[257] Chen, Hao, "Lighting and Material of *Halo 3*," *Game Developers Conference*, Mar. 2008. Cited on p. 475

[258] Chen, Hao, and Natalya Tatarchuk, "Lighting Research at Bungie," *SIGGRAPH Advances in Real-Time Rendering in 3D Graphics and Games course*, Aug. 2009. Cited on p. 256, 257, 475

[259] Chen, K., "Adaptive Virtual Texture Rendering in *Far Cry 4*," *Game Developers Conference*, Mar. 2015. Cited on p. 869

[260] Chen, Pei-Ju, Hiroko Awata, Atsuko Matsushita, En-Cheng Yang, and Kentaro Arikawa, "Extreme Spectral Richness in the Eye of the Common Bluebottle Butterfly, Graphium sarpedon," *Frontiers in Ecology and Evolution*, vol. 4, pp.18, Mar. 8, 2016. Cited on p. 272

[261] Chi, Yung-feng, "True-to-Life Real-Time Animation of Shallow Water on Todays GPUs," in Wolfgang Engel, ed., *ShaderX*^{4}, Charles River Media, pp. 467–480, 2005. Cited on p. 602, 626

[262] Chiang, Matt Jen-Yuan, Benedikt Bitterli, Chuck Tappan, and Brent Burley, "A Practical and Controllable Hair and Fur Model for Production Path Tracing," *Computer Graphics Forum (Eurographics 2016)*, vol. 35, no. 2, pp. 275–283, 2016. Cited on p. 643

[263] Chlumský, Viktor, *Shape Decomposition for Multi-channel Distance Fields*, MSc thesis, Department of Theoretical Computer Science, Czech Technical University in Prague, May 2015. Cited on p. 677, 890

[264] Choi, H., "Bifrost—The GPU Architecture for Next Five Billion," *ARM Tech Forum*, June 2016. Cited on p. 1026, 1027

[265] Christensen, Per H., "Point-Based Approximate Color Bleeding," Technical memo, Pixar Animation Studios, 2008. Cited on p. 454

[266] Cichocki, Adam, "Optimized Pixel-Projected Reflections for Planar Reflectors," *SIGGRAPH Advances in Real-Time Rendering in Games course*, Aug. 2017. Cited on p. 509

[267] Cignoni, P., C. Montani, and R. Scopigno, "Triangulating Convex Polygons Having T-Vertices," *journal of graphics tools*, vol. 1, no. 2, pp. 1–4, 1996. Also collected in [112]. Cited on p. 690

[268] Cignoni, Paolo, "On the Computation of Vertex Normals," *Meshlab Stuff* blog, Apr. 10, 2009. Also collected in [112]. Cited on p. 695

[269] Cigolle, Zina H., Sam Donow, Daniel Evangelakos, Michael Mara, Morgan McGuire, and Quirin Meyer, "A Survey of Efficient Representations for Independent Unit Vectors," *Journal of Computer Graphics Techniques*, vol. 3, no. 1, pp. 1–30, 2014. Cited on p. 222, 714, 715

[270] Clarberg, Petrik, and Tomas Akenine-Möller, "Practical Product Importance Sampling for Direct Illumination," *Computer Graphics Forum*, vol. 27, no. 2, pp. 681–690, 2008. Cited on p. 419

[271] Clarberg, P., R. Toth, J. Hasselgren, J. Nilsson, and T. Akenine-Möller, "AMFS: Adaptive Multi-frequency Shading for Future Graphics Processors," *ACM Transactions on Graphics*, vol. 33, no. 4, pp. 141:1–141:12, 2014. Cited on p. 910, 1013

[272] Clark, James H., "Hierarchical Geometric Models for Visible Surface Algorithms," *Communications of the ACM*, vol. 19, no. 10, pp. 547–554, Oct. 1976. Cited on p. 835

[273] Coffin, Christina, "SPU Based Deferred Shading in *Battlefield 3* for Playstation 3," *Game Developers Conference*, Mar. 2011. Cited on p. 898, 904

[274] Cohen, Jonathan D., Marc Olano, and Dinesh Manocha, "Appearance-Preserving Simplification," in *SIGGRAPH '98: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, ACM, pp. 115–122, July 1998. Cited on p. 212

[275] Cohen, Michael F., and John R. Wallace, *Radiosity and Realistic Image Synthesis*, Academic Press Professional, 1993. Cited on p. 442, 483

[276] Cohen-Or, Daniel, Yiorgos Chrysanthou, Frédo Durand, Ned Greene, Vladlen Kulton, and Cláudio T. Silva, *SIGGRAPH Visibility, Problems, Techniques and Applications course*, Aug. 2001. Cited on p.

[277] Cohen-Or, Daniel, Yiorgos Chrysanthou, Cláudio T. Silva, and Frédo Durand, "A Survey of Visibility for Walkthrough Applications," *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, no. 3, pp. 412–431, July–Sept. 2003. Cited on p. 830, 831, 879

[278] Cok, Keith, Roger Corron, Bob Kuehne, and Thomas True, *SIGGRAPH Developing Efficient Graphics Software: The Yin and Yang of Graphics course*, July 2000. Cited on p. 801

[279] Colbert, Mark, and Jaroslav Křivánek, "GPU-Based Importance Sampling," in Hubert Nguyen, ed., *GPU Gems 3*, Addison-Wesley, pp. 459–475, 2007. Cited on p. 419, 423, 503

[280] Colbert, Mark, and Jaroslav Křivánek, "Real-Time Shading with Filtered Importance Sampling," in *ACM SIGGRAPH 2007 Technical Sketches*, ACM, article no. 71, Aug. 2007. Cited on p. 419, 423

[281] Cole, Forrester, Aleksey Golovinskiy, Alex Limpaecher, Heather Stoddart Barros, Adam Finkelstein, Thomas Funkhouser, and Szymon Rusinkiewicz, "Where Do People Draw Lines?" *ACM Transactions on Graphics (SIGGRAPH 2008)*, vol. 27, no. 3, pp. 88:1–88:11, 2008. Cited on p. 656

[282] Cole, Forrester, and Adam Finkelstein, "Two Fast Methods for High-Quality Line Visibility," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 5, pp. 707–717, Sept./Oct. 2010. Cited on p. 668, 675

[283] Collin, D., "Culling the Battlefield," *Game Developers Conference*, Mar. 2011. Cited on p. 837, 840, 849

[284] Conran, Patrick, "SpecVar Maps: Baking Bump Maps into Specular Response," in *ACM SIGGRAPH 2005 Sketches*, ACM, article no. 22, Aug. 2005. Cited on p. 369

[285] Cook, Robert L., and Kenneth E. Torrance, "A Reflectance Model for Computer Graphics," *Computer Graphics (SIGGRAPH '81 Proceedings)*, vol. 15, no. 3, pp. 307–316, July 1981. Cited on p. 314, 326, 331, 338, 343, 446

[286] Cook, Robert L., and Kenneth E. Torrance, "A Reflectance Model for Computer Graphics," *ACM Transactions on Graphics*, vol. 1, no. 1, pp. 7–24, Jan. 1982. Cited on p. 326, 338, 343, 446

[287] Cook, Robert L., "Shade Trees," *Computer Graphics (SIGGRAPH '84 Proceedings)*, vol. 18, no. 3, pp. 223–231, July 1984. Cited on p. 37, 765

[288] Cook, Robert L., "Stochastic Sampling in Computer Graphics," *ACM Transactions on Graphics*, vol. 5, no. 1, pp. 51–72, Jan. 1986. Cited on p. 249

[289] Cook, Robert L., Loren Carpenter, and Edwin Catmull, "The Reyes Image Rendering Architecture," *Computer Graphics (SIGGRAPH '87 Proceedings)*, vol. 21, no. 4, pp. 95–102, July 1987. Cited on p. 26, 774, 908

[290] Cook, Robert L., and Tony DeRose, "Wavelet Noise," *ACM Transactions on Graphics (SIGGRAPH 2005)*, vol. 24, no. 3, pp. 803–811, 2005. Cited on p. 199

[291] Coombes, David, "DX12 Do's and Don'ts, Updated!" *NVIDIA GameWorks* blog, Nov. 12, 2015. Cited on p. 814

[292] Cormen, T. H., C. E. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*, MIT Press, 2009. Cited on p. 820, 829, 835

[293] Courrèges, Adrian, "*GTA V*—Graphics Study," *Adrian Courrèges* blog, Nov. 2, 2015. Cited on p. 525, 535, 901, 913

[294] Courrèges, Adrian, "*DOOM* (2016)—Graphics Study," *Adrian Courrèges* blog, Sept. 9, 2016. Cited on p. 246, 535, 540, 629, 901, 913

[295] Courrèges, Adrian, "Beware of Transparent Pixels," *Adrian Courrèges* blog, May 9, 2017. Cited on p. 160, 208

[296] Cox, Michael, and Pat Hanrahan, "Pixel Merging for Object-Parallel Rendering: A Distributed Snooping Algorithm," in *Proceedings of the 1993 Symposium on Parallel Rendering*, ACM, pp. 49–56, Nov. 1993. Cited on p. 802

[297] Cox, Michael, David Sprague, John Danskin, Rich Ehlers, Brian Hook, Bill Lorensen, and Gary Tarolli, *SIGGRAPH Developing High-Performance Graphics Applications for the PC Platform course*, July 1998. Cited on p. 1023

[298] Cozzi, Patrick, "Picking Using the Depth Buffer," *AGI Blog*, Mar. 5, 2008. Cited on p. 943

[299] Cozzi, Patrick, and Kevin Ring, *3D Engine Design for Virtual Globes*, A K Peters/CRC Press, 2011. Cited on p. 668, 715, 872, 879

[300] Cozzi, P., and D. Bagnell, "A WebGL Globe Rendering Pipeline," in Wolfgang Engel, ed., *GPU Pro*^{4}, CRC Press, pp. 39–48, 2013. Cited on p. 872, 876

[301] Cozzi, Patrick, ed., *WebGL Insights*, CRC Press, 2015. Cited on p. 129, 1048

[302] Cozzi, Patrick, "Cesium 3D Tiles," *GitHub* repository, 2017. Cited on p. 827

[303] Crane, Keenan, Ignacio Llamas, and Sarah Tariq, "Real-Time Simulation and Rendering of 3D Fluids," in Hubert Nguyen, ed., *GPU Gems 3*, Addison-Wesley, pp. 633–675, 2007. Cited on p. 608, 609, 649

[304] Crassin, Cyril, *GigaVoxels: A Voxel-Based Rendering Pipeline For Efficient Exploration Of Large And Detailed Scenes*, PhD thesis, University of Grenoble, July 2011. Cited on p. 494, 579, 584

[305] Crassin, Cyril, Fabrice Neyret, Miguel Sainz, Simon Green, and Elmar Eisemann, "Interactive Indirect Illumination Using Voxel Cone Tracing," *Computer Graphics Forum*, vol. 30, no. 7, pp. 1921–1930, 2011. Cited on p. 455, 467

[306] Crassin, Cyril, and Simon Green, "Octree-Based Sparse Voxelization Using the GPU Hardware Rasterizer," in Patrick Cozzi & Christophe Riccio, eds., *OpenGL Insights*, CRC Press, pp. 303–319, 2012. Cited on p. 582

[307] Crassin, Cyril, "Octree-Based Sparse Voxelization for Real-Time Global Illumination," *NVIDIA GPU Technology Conference*, Feb. 2012. Cited on p. 504, 582

[308] Crassin, Cyril, "Dynamic Sparse Voxel Octrees for Next-Gen Real-Time Rendering," *SIGGRAPH Beyond Programmable Shading course*, Aug. 2012. Cited on p. 579, 584

[309] Crassin, Cyril, Morgan McGuire, Kayvon Fatahalian, and Aaron Lefohn, "Aggregate G-Buffer Anti-Aliasing," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 10, pp. 2215–2228, Oct. 2016. Cited on p. 888

[310] Cripe, Brian, and Thomas Gaskins, "The DirectModel Toolkit: Meeting the 3D Graphics Needs of Technical Applications," *Hewlett-Packard Journal*, pp. 19–27, May 1998. Cited on p. 818

[311] Crow, Franklin C., "Shadow Algorithms for Computer Graphics," *Computer Graphics (SIGGRAPH '77 Proceedings)*, vol. 11, no. 2, pp. 242–248, July 1977. Cited on p. 230

[312] Crow, Franklin C., "Summed-Area Tables for Texture Mapping," *Computer Graphics (SIGGRAPH '84 Proceedings)*, vol. 18, no. 3, pp. 207–212, July 1984. Cited on p. 186

[313] Culler, David E., and Jaswinder Pal Singh, with Anoop Gupta, *Parallel Computer Architecture: A Hardware/Software Approach*, Morgan Kaufmann, 1998. Cited on p. 810

[314] Cunningham, Steve, "3D Viewing and Rotation Using Orthonormal Bases," in Andrew S. Glassner, ed., *Graphics Gems*, Academic Press, pp. 516–521, 1990. Cited on p. 74

[315] Cupisz, Kuba, and Kasper Engelstoft, "Lighting in Unity," *Game Developers Conference*, Mar. 2015. Cited on p. 476, 482, 509

[316] Cupisz, Robert, "Light Probe Interpolation Using Tetrahedral Tessellations," *Game Developers Conference*, Mar. 2012. Cited on p. 489, 490

[317] Curtis, Cassidy, "Loose and Sketchy Animation," in *ACM SIGGRAPH '98 Electronic Art and Animation Catalog*, ACM, p. 145, July 1998. Cited on p. 672

[318] Cychosz, J. M., and W. N. Waggenspack, Jr., "Intersecting a Ray with a Cylinder," in Paul S. Heckbert, ed., *Graphics Gems IV*, Academic Press, pp. 356–365, 1994. Cited on p. 959

[319] Cyrus, M., and J. Beck, "Generalized Two- and Three-Dimensional Clipping," *Computers and Graphics*, vol. 3, pp. 23–28, 1978. Cited on p. 959

[320] Dachsbacher, Carsten, and Marc Stamminger, "Translucent Shadow Maps," in *Proceedings of the 14th Eurographics Workshop on Rendering*, Eurographics Association, pp. 197–201, June 2003. Cited on p. 638, 639

[321] Dachsbacher, Carsten, and Marc Stamminger, "Reflective Shadow Maps," in *Proceedings of the 2005 Symposium on Interactive 3D Graphics and Games*, ACM, pp. 203–231, 2005. Cited on p. 491

[322] Dachsbacher, Carsten, and Marc Stamminger, "Splatting of Indirect Illumination," in *Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games*, ACM, pp. 93–100, 2006. Cited on p. 492

[323] Dachsbacher, C., and N. Tatarchuk, "Prism Parallax Occlusion Mapping with Accurate Silhouette Generation," *Symposium on Interactive 3D Graphics and Games poster*, Apr.–May 2007. Cited on p. 220

[324] Dallaire, Chris, "Binary Triangle Trees for Terrain Tile Index Buffer Generation," *Gamasutra*, Dec. 21, 2006. Cited on p. 876

[325] Dam, Erik B., Martin Koch, and Martin Lillholm, "Quaternions, Interpolation and Animation," Technical Report DIKU-TR-98/5, Department of Computer Science, University of Copenhagen, July 1998. Cited on p. 81

[326] Davies, Jem, "The Bifrost GPU Architecture and the ARM Mali-G71 GPU," *Hot Chips*, Aug. 2016. Cited on p. 1025

[327] Davies, Leigh, "OIT to Volumetric Shadow Mapping, 101 Uses for Raster-Ordered Views Using DirectX 12," *Intel Developer Zone* blog, Mar. 5, 2015. Cited on p. 52, 139, 156

[328] Davies, Leigh, "Rasterizer Order Views 101: A Primer," *Intel Developer Zone* blog, Aug. 5, 2015. Cited on p. 52, 156

[329] Day, Mike, "CSM Scrolling: An Acceleration Technique for the Rendering of Cascaded Shadow Maps," presented by Mike Acton, *SIGGRAPH Advances in Real-Time Rendering in Games course*, Aug. 2012. Cited on p. 245

[330] Day, Mike, "An Efficient and User-Friendly Tone Mapping Operator," *Insomniac R&D Blog*, Sept. 18, 2012. Cited on p. 286

[331] De Smedt, Matthijs, "PC GPU Performance Hot Spots," *NVIDIA GameWorks* blog, Aug. 10, 2016. Cited on p. 790, 792, 795, 814

[332] Debevec, Paul E., "Rendering Synthetic Objects into Real Scenes: Bridging Traditional and Image-Based Graphics with Global Illumination and High Dynamic Range Photography," in *SIGGRAPH '98: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, ACM, pp. 189–198, July 1998. Cited on p. 406

[333] Debevec, Paul, Rod Bogart, Frank Vitz, and Greg Ward, *SIGGRAPH HDRI and Image-Based Lighting course*, July 2003. Cited on p. 435

[334] DeBry, David (grue), Jonathan Gibbs, Devorah DeLeon Petty, and Nate Robins, "Painting and Rendering Textures on Unparameterized Models," *ACM Transactions on Graphics (SIGGRAPH 2002)*, vol. 21, no. 3, pp. 763–768, July 2002. Cited on p. 190

[335] DeCarlo, Doug, Adam Finkelstein, and Szymon Rusinkiewicz, "Interactive Rendering of Suggestive Contours with Temporal Coherence," in *Proceedings of the 3rd International Symposium on Non-Photorealistic Animation and Rendering*, ACM, pp. 15–24, June 2004. Cited on p. 655

[336] Decaudin, Philippe, "Cartoon-Looking Rendering of 3D-Scenes," Technical Report INRIA 2919, Université de Technologie de Compiègne, France, June 1996. Cited on p. 661, 664

[337] Decaudin, Philippe, and Fabrice Neyret, "Volumetric Billboards," *Computer Graphics Forum*, vol. 28, no. 8, pp. 2079–2089, 2009. Cited on p. 564

[338] Décoret, Xavier, Frédo Durand, François Sillion, and Julie Dorsey, "Billboard Clouds for Extreme Model Simplification," *ACM Transactions on Graphics (SIGGRAPH 2003)*, vol. 22, no. 3, pp. 689–696, 2003. Cited on p. 563

[339] Deering, M., S. Winnder, B. Schediwy, C. Duff, and N. Hunt, "The Triangle Processor and Normal Vector Shader: A VLSI System for High Performance Graphics," *Computer Graphics (SIGGRAPH '88 Proceedings)*, vol. 22, no. 4, pp. 21–30, Aug. 1988. Cited on p. 883

[340] Deering, Michael, "Geometry Compression," in *SIGGRAPH '95: Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, ACM, pp. 13–20, Aug. 1995. Cited on p. 700

[341] Delalandre, Cyril, Pascal Gautron, Jean-Eudes Marvie, and Guillaume François, "Transmittance Function Mapping," *Symposium on Interactive 3D Graphics and Games*, 2011. Cited on p. 570, 612, 620

[342] Delva, Michael, Julien Hamaide, and Ramses Ladlani, "Semantic Based Shader Generation Using Shader Shaker," in Wolfgang Engel, ed., *GPU Pro*^{6}, CRC Press, pp. 505–520, 2015. Cited on p. 128

[343] Demers, Joe, "Depth of Field: A Survey of Techniques," in Randima Fernando, ed., *GPU Gems*, Addison-Wesley, pp. 375–390, 2004. Cited on p. 531

[344] Demoreuille, Pete, "Optimizing the Unreal Engine 4 Renderer for VR," *Oculus Developer Blog*, May 25, 2016. Cited on p. 900, 934

[345] d'Eon, Eugene, and David Luebke, "Advanced Techniques for Realistic Real-Time Skin Rendering," in Hubert Nguyen, ed., *GPU Gems 3*, Addison-Wesley, pp. 293–347, 2007. Cited on p. 635, 636, 639

[346] d'Eon, Eugene, Guillaume François, Martin Hill, Joe Letteri, and Jean-Mary Aubry, "An Energy-Conserving Hair Reflectance Model," *Computer Graphics Forum*, vol. 30, no. 4, pp. 1467–8659, 2011. Cited on p. 641, 643

[347] DeRose, T., M. Kass, and T. Truong, "Subdivision Surfaces in Character Animation," in *SIGGRAPH '98: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, ACM, pp. 85–94, July 1998. Cited on p. 761, 764, 767, 777

[348] Deshmukh, Priyamvad, Feng Xie, and Eric Tabellion, "DreamWorks Fabric Shading Model: From Artist Friendly to Physically Plausible," in *ACM SIGGRAPH 2017 Talks*, article no. 38, July 2017. Cited on p. 359

[349] Deshpande, Adit, "The 9 Deep Learning Papers You Need To Know About," *Adit Deshpande* blog, Aug. 24, 2016. Cited on p. 1043

[350] Didyk, P., T. Ritschel, E. Eisemann, K. Myszkowski, and H.-P. Seidel, "Adaptive Image-Space Stereo View Synthesis," in *Proceedings of the Vision, Modeling, and Visualization Workshop 2010*, Eurographics Association, pp. 299–306, 2010. Cited on p. 523

[351] Didyk, P., E. Eisemann, T. Ritschel, K. Myszkowski, and H.-P. Seidel, "Perceptually-Motivated Real-Time Temporal Upsampling of 3D Content for High-Refresh-Rate Displays," *Computer Graphics Forum*, vol. 29, no. 2, pp. 713–722, 2011. Cited on p. 523

[352] Dietrich, Andreas, Enrico Gobbetti, and Sung-Eui Yoon, "Massive-Model Rendering Techniques," *IEEE Computer Graphics and Applications*, vol. 27, no. 6, pp. 20–34, Nov./Dec. 2007. Cited on p. 587, 879

[353] Dietrich, Sim, "Attenuation Maps," in Mark DeLoura, ed., *Game Programming Gems*, Charles River Media, pp. 543–548, 2000. Cited on p. 221

[354] Dimitrijević, Aleksandar, "Performance State Tracking," in Patrick Cozzi & Christophe Riccio, eds., *OpenGL Insights*, CRC Press, pp. 527–534, 2012. Cited on p. 789

[355] Dimov, Rossen, "Deriving the Smith Shadowing Function for the GTR BRDF," Chaos Group White Paper, June 2015. Cited on p. 343

[356] Ding, Vivian, "In-Game and Cinematic Lighting of *The Last of Us*," *Game Developers Conference*, Mar. 2014. Cited on p. 229

[357] Dmitriev, Kirill, and Yury Uralsky, "Soft Shadows Using Hierarchical Min-Max Shadow Maps," *Game Developers Conference*, Mar. 2007. Cited on p. 252

[358] Dobashi, Yoshinori, Kazufumi Kaneda, Hideo Yamashita, Tsuyoshi Okita, and Tomoyuki Nishita, "A Simple, Efficient Method for Realistic Animation of Clouds," in *SIGGRAPH '00: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, ACM Press/Addison-Wesley Publishing Co., pp. 19–28, July 2000. Cited on p. 556

[359] Dobashi, Yoshinori, Tsuyoshi Yamamoto, and Tomoyuki Nishita, "Interactive Rendering of Atmospheric Scattering Effects Using Graphics Hardware," in *Graphics Hardware 2002*, Eurographics Association, pp. 99–107, Sept. 2002. Cited on p. 604

[360] Dobbie, Will, "GPU Text Rendering with Vector Textures," *Will Dobbie* blog, Jan. 21, 2016. Cited on p. 677

[361] Dobbyn, Simon, John Hamill, Keith O'Conor, and Carol O'Sullivan, "Geopostors: A Real-Time Geometry/Impostor Crowd Rendering System," in *Proceedings of the 2005 Symposium on Interactive 3D Graphics and Games*, ACM, pp. 95–102, Apr. 2005. Cited on p. 551

[362] Doggett, M., "Texture Caches," *IEEE Micro*, vol. 32, no. 3, pp. 136–141, 2005. Cited on p. 1017

[363] Doghramachi, Hawar, and Jean-Normand Bucci, "Deferred+: Next-Gen Culling and Rendering for the Dawn Engine," in Wolfgang Engel, ed., *GPU Zen*, Black Cat Publishing, pp. 77–103, 2017. Cited on p. 715, 849, 908

[364] Dolby Laboratories Inc., "ICtCp Dolby White Paper," Dolby website. Cited on p. 276, 287

[365] Dominé, Sébastien, "OpenGL Multisample," *Game Developers Conference*, Mar. 2002. Cited on p. 145

[366] Dong, Zhao, Bruce Walter, Steve Marschner, and Donald P. Greenberg, "Predicting Appearance from Measured Microgeometry of Metal Surfaces," *ACM Transactions on Graphics*, vol. 35, no. 1, article no. 9, 2015. Cited on p. 361

[367] Donnelly, William, "Per-Pixel Displacement Mapping with Distance Functions," in Matt Pharr, ed., *GPU Gems 2*, Addison-Wesley, pp. 123–136, 2005. Cited on p. 218

[368] Donnelly, William, and Andrew Lauritzen, "Variance Shadow Maps," in *Proceedings of the 2006 Symposium on Interactive 3D Graphics*, ACM, pp. 161–165, 2006. Cited on p. 252

[369] Donner, Craig, and Henrik Wann Jensen, "Light Diffusion in Multi-Layered Translucent Materials," *ACM Transactions on Graphics (SIGGRAPH 2005)*, vol. 24, no. 3, pp. 1032–1039, 2005. Cited on p. 635

[370] Doo, D., and M. Sabin, "Behaviour of Recursive Division Surfaces Near Extraordinary Points," *Computer-Aided Design*, vol. 10, no. 6, pp. 356–360, Sept. 1978. Cited on p. 761

[371] Dorn, Jonathan, Connelly Barnes, Jason Lawrence, and Westley Weimer, "Towards Automatic Band-Limited Procedural Shaders," *Computer Graphics Forum (Pacific Graphics 2015)*, vol. 34, no. 7, pp. 77–87, 2015. Cited on p. 200

[372] Doss, Joshua A., "Art-Based Rendering with Graftal Imposters," in Mark DeLoura, ed., *Game Programming Gems 7*, Charles River Media, pp. 447–454, 2008. Cited on p. 672

[373] Dou, Hang, Yajie Yan, Ethan Kerzner, Zeng Dai, and Chris Wyman, "Adaptive Depth Bias for Shadow Maps," *Journal of Computer Graphics Techniques*, vol. 3, no. 4, pp. 146–162, 2014. Cited on p. 250

[374] Dougan, Carl, "The Parallel Transport Frame," in Mark DeLoura, ed., *Game Programming Gems 2*, Charles River Media, pp. 215–219, 2001. Cited on p. 102

[375] Drago, F., K. Myszkowski, T. Annen, and N. Chiba, "Adaptive Logarithmic Mapping for Displaying High Contrast Scenes,' *Computer Graphics Forum*, vol. 22, no. 3, pp. 419–426, 2003. Cited on p. 286

[376] Driscoll, Rory, "Cubemap Texel Solid Angle," *CODEITNOW* blog, Jan. 15, 2012. Cited on p. 419

[377] Drobot, Michal, "Quadtree Displacement Mapping with Height Blending," in Wolfgang Engel, ed., *GPU Pro*, A K Peters, Ltd., pp. 117–148, 2010. Cited on p. 220

[378] Drobot, Michał, "A Spatial and Temporal Coherence Framework for Real-Time Graphics," in Eric Lengyel, ed., *Game Engine Gems 2*, A K Peters, Ltd., pp. 97–118, 2011. Cited on p. 518

[379] Drobot, Michal, "Lighting of *Killzone: Shadow Fall*," *Digital Dragons* conference, Apr. 2013. Cited on p. 116

[380] Drobot, Michal, "Physically Based Area Lights," in Wolfgang Engel, ed., *GPU Pro*^{5}, CRC Press, pp. 67–100, 2014. Cited on p. 116, 388

[381] Drobot, Michal, "GCN Execution Patterns in Full Screen Passes," *Michal Drobot* blog, Apr. 1, 2014. Cited on p. 514

[382] Drobot, Michał, "Hybrid Reconstruction Anti Aliasing," *SIGGRAPH Advances in Real-Time Rendering in Games course*, Aug. 2014. Cited on p. 141, 142, 146, 165

[383] Drobot, Michał, "Hybrid Reconstruction Antialiasing," in Wolfgang Engel, ed., *GPU Pro*^{6}, CRC Press, pp. 101–139, 2015. Cited on p. 141, 146, 165

[384] Drobot, Michal, "Rendering of *Call of Duty Infinite Warfare*," *Digital Dragons* conference, May 2017. Cited on p. 262, 325, 371, 420, 502, 503, 509, 569

[385] Drobot, Michal, "Improved Culling for Tiled and Clustered Rendering," *SIGGRAPH Advances in Real-Time Rendering in Games course*, Aug. 2017. Cited on p. 902, 905

[386] Drobot, Michał, "Practical Multilayered Materials in *Call of Duty Infinite Warfare*," *SIGGRAPH Physically Based Shading in Theory and Practice course*, Aug. 2017. Cited on p. 151, 363, 364, 623, 625, 629

[387] Duff, Tom, "Compositing 3-D Rendered Images," *Computer Graphics (SIGGRAPH '85 Proceedings)*, vol. 19, no. 3, pp. 41–44, July 1985. Cited on p. 149

[388] Duff, Tom, James Burgess, Per Christensen, Christophe Hery, Andrew Kensler, Max Liani, and Ryusuke Villemin, "Building an Orthonormal Basis, Revisited," *Journal of Computer Graphics Techniques*, vol. 6, no. 1, pp. 1–8, 2017. Cited on p. 75

[389] Duffy, Joe, "CLR Inside Out," *MSDN Magazine*, vol. 21, no. 10, Sept. 2006. Cited on p. 791

[390] Dufresne, Marc Fauconneau, "Forward Clustered Shading," *Intel Software Developer Zone*, Aug. 5, 2014. Cited on p. 900, 914

[391] Duiker, Haarm-Pieter, and George Borshukov, "Filmic Tone Mapping," Presentation at Electronic Arts, Oct. 27, 2006. Cited on p. 286

[392] Duiker, Haarm-Pieter, "Filmic Tonemapping for Real-Time Rendering," *SIGGRAPH Color Enhancement and Rendering in Film and Game Production course*, July 2010. Cited on p. 286, 288, 289, 290

[393] Dummer, Jonathan, "Cone Step Mapping: An Iterative Ray-Heightfield Intersection Algorithm," *lonesock* website, 2006. Cited on p. 219

[394] Dunn, Alex, "Transparency (or Translucency) Rendering," *NVIDIA GameWorks* blog, Oct. 20, 2014. Cited on p. 155, 157, 159, 204, 569

[395] Dupuy, Jonathan, Eric Heitz, Jean-Claude Iehl, Pierre Poulin, Fabrice Neyret, and Victor Ostromoukhov, "Linear Efficient Antialiased Displacement and Reflectance Mapping," *ACM Transactions on Graphics*, vol. 32, no. 6, pp. 211:1–211:11, Nov. 2013. Cited on p. 370

[396] Dupuy, Jonathan, "Antialiasing Physically Based Shading with LEADR Mapping," *SIGGRAPH Physically Based Shading in Theory and Practice course*, Aug. 2014. Cited on p. 370

[397] Dupuy, Jonathan, Eric Heitz, and Eugene d'Eon, "Additional Progress Towards the Unification of Microfacet and Microflake Theories," in *Proceedings of the Eurographics Symposium on Rendering: Experimental Ideas & Implementations*, Eurographics Association, pp. 55–63, 2016. Cited on p. 352, 648

[398] Durand, Frédo, *3D Visibility: Analytical Study and Applications*, PhD thesis, Université Joseph Fourier, Grenoble, July 1999. Cited on p. 879

[399] Dutré, Philip, *Global Illumination Compendium*, webpage, Sept. 29, 2003. Cited on p. 372, 443, 512

[400] Dutré, Philip, Kavita Bala, and Philippe Bekaert, *Advanced Global Illumination*, Second Edition, A K Peters, Ltd., 2006. Cited on p. 269, 442, 512, 684

[401] Dyken, C., M. Reimers, and J. Seland, "Real-Time GPU Silhouette Refinement Using Adaptively Blended Bézier Patches," *Computer Graphics Forum*, vol. 27, no. 1, pp. 1–12, 2008. Cited on p. 747

[402] Dyn, Nira, David Levin, and John A. Gregory, "A 4-Point Interpolatory Subdivision Scheme for Curve Design," *Computer Aided Geometric Design*, vol. 4, no. 4, pp. 257–268, 1987. Cited on p. 755

[403] Eberly, David, "Triangulation by Ear Clipping," *Geometric Tools* website, 2003. Cited on p. 686

[404] Eberly, David, *3D Game Engine Design: A Practical Approach to Real-Time Computer Graphics*, Second Edition, Morgan Kaufmann, 2006. Cited on p. 82, 772, 829, 950, 951, 959, 976, 990

[405] Eberly, David, "Reconstructing a Height Field from a Normal Map," *Geometric Tools* blog, May 3, 2006. Cited on p. 214

[406] Eberly, David, "A Fast and Accurate Algorithm for Computing SLERP," *Journal of Graphics, GPU, and Game Tools*, vol. 15, no. 3, pp. 161–176, 2011. Cited on p. 82

[407] Ebert, David S., John Hart, Bill Mark, F. Kenton Musgrave, Darwyn Peachey, Ken Perlin, and Steven Worley, *Texturing and Modeling: A Procedural Approach*, Third Edition, Morgan Kaufmann, 2002. Cited on p. 198, 200, 222, 672

[408] Eccles, Allen, "The Diamond Monster 3Dfx Voodoo 1," *GameSpy Hall of Fame*, 2000. Cited on p. 1

[409] Eisemann, Martin, and Xavier Décoret, "Fast Scene Voxelization and Applications," in *ACM SIGGRAPH 2006 Sketches*, ACM, article no. 8, 2006. Cited on p. 581, 586

[410] Eisemann, Martin, Marcus Magnor, Thorsten Grosch, and Stefan Müller, "Fast Ray/Axis-Aligned Bounding Box Overlap Tests Using Ray Slopes," *journal of graphics tools*, vol. 12, no. 4, pp. 35–46, 2007. Cited on p. 961

[411] Eisemann, Martin, and Xavier Décoret, "Occlusion Textures for Plausible Soft Shadows," *Computer Graphics Forum*, vol. 27, no. 1, pp. 13–23, 2008. Cited on p. 230

[412] Eisemann, Martin, Michael Schwarz, Ulf Assarsson, and Michael Wimmer, *Real-Time Shadows*, A K Peters/CRC Press, 2011. Cited on p. 223, 244, 249, 253, 265

[413] Eisemann, Martin, Michael Schwarz, Ulf Assarsson, and Michael Wimmer, *SIGGRAPH Efficient Real-Time Shadows course*, Aug. 2012. Cited on p. 265

[414] El Garawany, Ramy, "Deferred Lighting in *Uncharted 4*," *SIGGRAPH Advances in Real-Time Rendering in Games course*, July 2016. Cited on p. 472, 886, 887, 898, 904

[415] El Mansouri, Jalal, "Rendering Tom Clancy's Rainbow Six Siege," *Game Developers Conference*, Mar. 2016. Cited on p. 146, 246, 252, 805, 850, 887

[416] Elcott, Sharif, Kay Chang, Masayoshi Miyamoto, and Napaporn Metaaphanon, "Rendering Techniques of *Final Fantasy XV*," in *ACM SIGGRAPH 2016 Talks*, ACM, article no. 48, July 2016. Cited on p. 620

[417] Eldridge, Matthew, Homan Igehy, and Pat Hanrahan, "Pomegranate: A Fully Scalable Graphics Architecture," in *SIGGRAPH '00: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, ACM Press/Addison-Wesley Publishing Co., pp. 443–454, July 2000. Cited on p. 1020, 1021, 1022

[418] Eldridge, Matthew, *Designing Graphics Architectures around Scalability and Communication*, PhD thesis, Stanford University, June 2001. Cited on p. 1020, 1022, 1023

[419] Elek, Oskar, "Rendering Parametrizable Planetary Atmospheres with Multiple Scattering in Real Time," *Central European Seminar on Computer Graphics*, 2009. Cited on p. 615

[420] Elek, Oskar, "Layered Materials in Real-Time Rendering," in *Proceedings of the 14th Central European Seminar on Computer Graphics*, Vienna University of Technology, pp. 27–34, May 2010. Cited on p. 364

[421] Elinas, Pantelis, and Wolfgang Stuerzlinger, "Real-Time Rendering of 3D Clouds," *journal of graphics tools*, vol. 5, no. 4, pp. 33–45, 2000. Cited on p. 556

[422] van Emde Boas, P., R. Kaas, and E. Zijlstra, "Design and Implementation of an Efficient Priority Queue," *Mathematical Systems Theory*, vol. 10, no. 1, pp. 99–127, 1977. Cited on p. 827

[423] Enderton, Eric, Erik Sintorn, Peter Shirley, and David Luebke, "Stochastic Transparency," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 8, pp. 1036–1047, 2011. Cited on p. 149, 206

[424] Endres, Michael, and Frank Kitson, "Perfecting The Pixel: Refining the Art of Visual Styling," *Game Developers Conference*, Mar. 2010. Cited on p. 289

[425] Eng, Austin, "Tighter Frustum Culling and Why You May Want to Disregard It," *Cesium* blog, Feb. 2, 2017. Cited on p. 986

[426] Engel, Wolfgang, ed., *Direct3D ShaderX: Vertex & Pixel Shader Tips and Techniques*, Wordware, 2002. Cited on p. xvii

[427] Engel, Wolfgang, ed., *ShaderX^2: Introduction & Tutorials with DirectX 9*, Wordware, 2004. Cited on p. xvi

[428] Engel, Wolfgang, ed., *ShaderX^2: Shader Programming Tips & Tricks with DirectX 9*, Wordware, 2004. Cited on p. xvi

[429] Engel, Wolfgang, ed., *ShaderX^3*, Charles River Media, 2004. Cited on p. 1148

[430] Engel, Wolfgang, "Cascaded Shadow Maps," in Wolfgang Engel, ed., *ShaderX^5*, Charles River Media, pp. 197–206, 2006. Cited on p. 242, 243

[431] Engel, Wolfgang, "Designing a Renderer for Multiple Lights: The Light Pre-Pass Renderer," in Wolfgang Engel, ed., *ShaderX^7*, Charles River Media, pp. 655–666, 2009. Cited on p. 892

[432] Engel, Wolfgang, "Light Pre-Pass; Deferred Lighting: Latest Development," *SIGGRAPH Advances in Real-Time Rendering in Games course*, Aug. 2009. Cited on p. 892, 901

[433] Engel, Wolfgang, "The Filtered and Culled Visibility Buffer," *Game Developers Conference Europe*, Aug. 2016. Cited on p. 833, 851, 907

[434] Engelhardt, Thomas, and Carsten Dachsbacher, "Octahedron Environment Maps," in *Proceedings of the Vision, Modeling, and Visualization Conference 2008*, Aka GmbH, pp. 383–388 Oct. 2008. Cited on p. 413

[435] Ericson, Christer, *Real-Time Collision Detection*, Morgan Kaufmann, 2005. Cited on p. 827, 879, 946, 948, 950, 955, 977, 978, 979, 990

[436] Ericson, Christer, "Collisions Using Separating-Axis Tests," *Game Developers Conference*, Mar. 2007. Cited on p. 980

[437] Ericson, Christer, "More Capcom/CEDEC Bean-Spilling," *realtimecollisiondetection.net—the blog*, Oct. 1, 2007. Cited on p. 537

[438] Ericson, Christer, "Order Your Graphics Draw Calls Around!" *realtimecollisiondetection .net—the blog*, Oct. 3, 2008. Cited on p. 803

[439] Ericson, Christer, "Optimizing the Rendering of a Particle System," *realtimecollision detection.net—the blog*, Jan. 2, 2009. Cited on p. 556, 568

[440] Ericson, Christer, "Optimizing a Sphere-Triangle Intersection Test," *realtimecollision detection.net—the blog*, Dec. 30, 2010. Cited on p. 974

[441] Eriksson, Carl, Dinesh Manocha, and William V. Baxter III, "HLODs for Faster Display of Large Static and Dynamic Environments," in *Proceedings of the 2001 Symposium on Interactive 3D Graphics*, ACM, pp. 111–120, 2001. Cited on p. 866

[442] Estevez, Alejandro Conty, and Christopher Kulla, "Production Friendly Microfacet Sheen BRDF," Technical Report, Sony Imageworks, 2017. Cited on p. 358

[443] Etuaho, Olli, "Bug-Free and Fast Mobile WebGL," in Patrick Cozzi, ed., *WebGL Insights*, CRC Press, pp. 123–137, 2015. Cited on p. 702, 796, 802, 805, 814

[444] Evans, Alex, "Fast Approximations for Global Illumination on Dynamic Scenes," *SIGGRAPH Advanced Real-Time Rendering in 3D Graphics and Games course*, Aug. 2006. Cited on p. 454, 488

[445] Evans, Alex, and Anton Kirczenow, "Voxels in *LittleBigPlanet 2*," *SIGGRAPH Advances in Real-Time Rendering in Games course*, Aug. 2011. Cited on p. 571

[446] Evans, Alex, "Learning from Failure: A Survey of Promising, Unconventional and Mostly Abandoned Renderers for 'Dreams PS4', a Geometrically Dense, Painterly UGC Game," *SIGGRAPH Advances in Real-Time Rendering in Games course*, Aug. 2015. Cited on p. 577, 679

[447] Evans, Martin, "Drawing Stuff on Other Stuff with Deferred Screenspace Decals," *Blog 3.0*, Feb. 27, 2015. Cited on p. 889

[448] Everitt, Cass, "One-Pass Silhouette Rendering with GeForce and GeForce2," NVIDIA White Paper, June 2000. Cited on p. 656

[449] Everitt, Cass, "Interactive Order-Independent Transparency," NVIDIA White Paper, May 2001. Cited on p. 154

[450] Everitt, Cass, and Mark Kilgard, "Practical and Robust Stenciled Shadow Volumes for Hardware-Accelerated Rendering," NVIDIA White Paper, Mar. 2002. Cited on p. 232

[451] Everitt, Cass, and John McDonald, "Beyond Porting," *Steam Dev Days*, Feb. 2014. Cited on p. 795, 805

[452] Everitt, Cass, Graham Sellers, John McDonald, and Tim Foley, "Approaching Zero Driver Overhead," *Game Developers Conference*, Mar. 2014. Cited on p. 191, 192

[453] Everitt, Cass, "Multiview Rendering," *SIGGRAPH Moving Mobile Graphics course*, July 2016. Cited on p. 927, 928

[454] Ewins, Jon P., Marcus D. Waller, Martin White, and Paul F. Lister, "MIP-Map Level Selection for Texture Mapping," *IEEE Transactions on Visualization and Computer Graphics*, vol. 4, no. 4, pp. 317–329, Oct.–Dec. 1998. Cited on p. 185

[455] Eyles, J., S. Molnar, J. Poulton, T. Greer, A. Lastra, N. England, and L. Westover, "PixelFlow: The Realization," in *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Workshop on Graphics Hardware*, ACM, pp. 57–68, Aug. 1997. Cited on p. 1022

[456] Fairchild, Mark D., *Color Appearance Models*, Third Edition, John Wiley & Sons, Inc., 2013. Cited on p. 276, 278, 291

[457] Farin, Gerald, "Triangular Bernstein-Bézier Patches," *Computer Aided Geometric Design*, vol. 3, no. 2, pp. 83–127, 1986. Cited on p. 745, 781

[458] Farin, Gerald, *Curves and Surfaces for Computer Aided Geometric Design—A Practical Guide*, Fourth Edition, Academic Press Inc., 1996. Cited on p. 718, 720, 721, 724, 725, 728, 732, 734, 738, 742, 745, 749, 754, 756, 781

[459] Farin, Gerald E., *NURBS: From Projective Geometry to Practical Use*, Second Edition, A K Peters, Ltd., 1999. Cited on p. 781

[460] Farin, Gerald, and Dianne Hansford, *The Essentials of CAGD*, A K Peters, Ltd., 2000. Cited on p. 781

[461] Farin, Gerald E., and Dianne Hansford, *Practical Linear Algebra: A Geometry Toolbox*, A K Peters, Ltd., 2004. Cited on p. 102, 991

[462] Fatahalian, Kayvon, and Randy Bryant, *Parallel Computer Architecture and Programming course*, Carnegie Mellon University, Spring 2017. Cited on p. 30, 55

[463] Fauconneau, M., "High-Quality, Fast DX11 Texture Compression with ISPC," *Game Developers Conference*, Mar. 2015. Cited on p. 198

[464] Fedkiw, Ronald, Jos Stam, and Henrik Wann Jensen, "Visual Simulation of Smoke," in *SIGGRAPH '01: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, ACM, pp. 15–22, Aug. 2001. Cited on p. 649

[465] Fenney, Simon, "Texture Compression Using Low-Frequency Signal Modulation," in *Graphics Hardware 2003*, Eurographics Association, pp. 84–91, July 2003. Cited on p. 196

[466] Fernandes, António Ramires, and Bruno Oliveira, "GPU Tessellation: We Still Have a LOD of Terrain to Cover," in Patrick Cozzi & Christophe Riccio, eds., *OpenGL Insights*, CRC Press, pp. 145–161, 2012. Cited on p. 46, 879

[467] Fernando, Randima, "Percentage-Closer Soft Shadows," in *ACM SIGGRAPH 2005 Sketches*, ACM, article no. 35, Aug. 2005. Cited on p. 250

[468] Ferwerda, James, "Elements of Early Vision for Computer Graphics," *IEEE Computer Graphics and Applications*, vol. 21, no. 5, pp. 22–33, Sept./Oct. 2001. Cited on p. 278

[469] Feynman, Richard, Robert B. Leighton, and Matthew Sands, *The Feynman Lectures on Physics*, 1963. Available at *Feynman Lectures* website, 2006. Cited on p. 298, 373

[470] de Figueiredo, L. H., "Adaptive Sampling of Parametric Curves," in Alan Paeth, ed., *Graphics Gems V*, Academic Press, pp. 173–178, 1995. Cited on p. 771

[471] Filion, Dominic, and Rob McNaughton, "Starcraft II: Effects and Techniques," *SIGGRAPH Advances in Real-Time Rendering in 3D Graphics and Games course*, Aug. 2008. Cited on p. 257, 459, 885

[472] Fisher, F., and A. Woo, "R.E versus N.H Specular Highlights," in Paul S. Heckbert, ed., *Graphics Gems IV*, Academic Press, pp. 388–400, 1994. Cited on p. 421

[473] Flavell, Andrew, "Run Time Mip-Map Filtering," *Game Developer*, vol. 5, no. 11, pp. 34–43, Nov. 1998. Cited on p. 185, 186

[474] Floater, Michael, Kai Hormann, and Géza Kós, "A General Construction of Barycentric Coordinates over Convex Polygons," *Advances in Computational Mathematics*, vol. 24, no. 1–4, pp. 311–331, Jan. 2006. Cited on p. 970

[475] Floater, M., "Triangular Bézier Surfaces," Technical Report, University of Oslo, Aug. 2011. Cited on p. 741

[476] Fog, Agner, "Optimizing Software in C++," *Software Optimization Resources*, 2007. Cited on p. 815

[477] Fogal, Thomas, Alexander Schiewe, and Jens Krüger, "An Analysis of Scalable GPU-Based Ray-Guided Volume Rendering," in *Proceedings of the IEEE Symposium on Large Data Analysis and Visualization (LDAV 13)*, IEEE Computer Society, pp. 43–51, 2013. Cited on p. 586

[478] Foley, Tim, "Introduction to Parallel Programming Models," *SIGGRAPH Beyond Programmable Shading course*, Aug. 2009. Cited on p. 815

[479] Fong, Julian, Magnus Wrenninge, Christopher Kulla, and Ralf Habel, *SIGGRAPH Production Volume Rendering course*, Aug. 2017. Cited on p. 589, 590, 591, 592, 594, 649

[480] Forest, Vincent, Loic Barthe, and Mathias Paulin, "Real-Time Hierarchical Binary-Scene Voxelization," *journal of graphics, GPU, and game tools*, vol. 14, no. 3, pp. 21–34, 2011. Cited on p. 581

[481] Forsyth, Tom, "Comparison of VIPM Methods," in Mark DeLoura, ed., *Game Programming Gems 2*, Charles River Media, pp. 363–376, 2001. Cited on p. 707, 711, 859

[482] Forsyth, Tom, "Impostors: Adding Clutter," in Mark DeLoura, ed., *Game Programming Gems 2*, Charles River Media, pp. 488–496, 2001. Cited on p. 561, 562

[483] Forsyth, Tom, "Making Shadow Buffers Robust Using Multiple Dynamic Frustums," in Wolfgang Engel, ed., *ShaderX4*, Charles River Media, pp. 331–346, 2005. Cited on p. 242

[484] Forsyth, Tom, "Extremely Practical Shadows," *Game Developers Conference*, Mar. 2006. Cited on p. 234, 241, 242

[485] Forsyth, Tom, "Linear-Speed Vertex Cache Optimisation," *TomF's Tech Blog*, Sept. 28, 2006. Cited on p. 701, 705

[486] Forsyth, Tom, "Shadowbuffers," *Game Developers Conference*, Mar. 2007. Cited on p. 234, 242

[487] Forsyth, Tom, "The Trilight: A Simple General-Purpose Lighting Model for Games," *TomF's Tech Blog*, Mar. 22, 2007. Cited on p. 382, 432

[488] Forsyth, Tom, "Renderstate Change Costs," *TomF's Tech Blog*, Jan. 27, 2008. Cited on p. 795, 796, 802, 803

[489] Forsyth, Tom, "VR, AR and Other Realities," *TomF's Tech Blog*, Sept. 16, 2012. Cited on p. 917

[490] Forsyth, Tom, "Premultiplied Alpha Part 2," *TomF's Tech Blog*, Mar. 18, 2015. Cited on p. 208

[491] Forsyth, Tom, "The sRGB Learning Curve," *TomF's Tech Blog*, Nov. 30, 2015. Cited on p. 161, 162, 163

[492] Fowles, Grant R., *Introduction to Modern Optics*, Second Edition, Holt, Reinhart, and Winston, 1975. Cited on p. 373

[493] Franklin, Dustin, "Hardware-Based Ambient Occlusion," in Wolfgang Engel, ed., *ShaderX4*, Charles River Media, pp. 91–100, 2005. Cited on p. 452

[494] Frey, Ivo Zoltan, "Spherical Skinning with Dual-Quaternions and QTangents," in *ACM SIGGRAPH 2011 Talks*, article no. 11, Aug. 2011. Cited on p. 209, 210, 715

[495] Frisken, Sarah, Ronald N. Perry, Alyn P. Rockwood, and Thouis R. Jones, "Adaptively Sampled Distance Fields: A General Representation of Shape for Computer Graphics," in *SIGGRAPH '00: Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, ACM Press/Addison-Wesley Publishing Co., pp. 249–254, July 2000. Cited on p. 677, 751

[496] Frisvad, Jeppe Revall, "Building an Orthonormal Basis from a 3D Unit Vector Without Normalization," *journal of graphics tools*, vol. 16, no. 3, pp. 151–159, 2012. Cited on p. 75

[497] Fry, Alex, "High Dynamic Range Color Grading and Display in Frostbite," *Game Developers Conference*, Feb.–Mar. 2017. Cited on p. 283, 287, 288, 290

[498] Frykholm, Niklas, "The BitSquid Low Level Animation System," *Autodesk Stingray* blog, Nov. 20, 2009. Cited on p. 715, 905

[499] Frykholm, Niklas, "What Is Gimbal Lock and Why Do We Still Have to Worry about It?" *Autodesk Stingray* blog, Mar. 15, 2013. Cited on p. 73

[500] Fuchs, H., Z. M. Kedem, and B. F. Naylor, "On Visible Surface Generation by A Priori Tree Structures," *Computer Graphics (SIGGRAPH '80 Proceedings)*, vol. 14, no. 3, pp. 124–133, July 1980. Cited on p. 823

[501] Fuchs, H., G. D. Abram, and E. D. Grant, "Near Real-Time Shaded Display of Rigid Objects," *Computer Graphics (SIGGRAPH '83 Proceedings)*, vol. 17, no. 3, pp. 65–72, July 1983. Cited on p. 823

[502] Fuchs, H., J. Poulton, J. Eyles, T. Greer, J. Goldfeather, D. Ellsworth, S. Molnar, G. Turk, B. Tebbs, and L. Israel, "Pixel-Planes 5: A Heterogeneous Multiprocessor Graphics System Using Processor-Enhanced Memories," *Computer Graphics (SIGGRAPH '89 Proceedings)*, vol. 23, no. 3, pp. 79–88, July 1989. Cited on p. 8, 1026

[503] Fuhrmann, Anton L., Eike Umlauf, and Stephan Mantler, "Extreme Model Simplification for Forest Rendering," in *Proceedings of the First Eurographics Conference on Natural Phenomena*, Eurographics Association, pp. 57–66, 2005. Cited on p. 563

[504] Fujii, Yasuhiro, "A Tiny Improvement of Oren-Nayar Reflectance Model," http://mimosa-pudica.net, Oct. 9, 2013. Cited on p. 354

[505] Fünfzig, C., K. Müller, D. Hansford, and G. Farin, "PNG1 Triangles for Tangent Plane Continuous Surfaces on the GPU," in *Graphics Interface 2008*, Canadian Information Processing Society, pp. 219–226, 2008. Cited on p. 747

[506] Fung, James, "Computer Vision on the GPU," in Matt Pharr, ed., *GPU Gems 2*, Addison-Wesley, pp. 649–666, 2005. Cited on p. 521

[507] Funkhouser, Thomas A., *Database and Display Algorithms for Interactive Visualization of Architectural Models*, PhD thesis, University of California, Berkeley, 1993. Cited on p. 866

[508] Funkhouser, Thomas A., and Carlo H. Séquin, "Adaptive Display Algorithm for Interactive Frame Rates During Visualization of Complex Virtual Environments," in *SIGGRAPH '93: Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, ACM, pp. 247–254, Aug. 1993. Cited on p. 710, 864, 865, 866

[509] Fürst, René, Oliver Mattausch, and Daniel Scherzer, "Real-Time Deep Shadow Maps," in Wolfgang Engel, ed., *GPU Pro*^{4}, CRC Press, pp. 253–264, 2013. Cited on p. 258

[510] Gaitatzes, Athanasios, and Georgios Papaioannou, "Progressive Screen-Space Multichannel Surface Voxelization," in Wolfgang Engel, ed., *GPU Pro*^{4}, CRC Press, pp. 137–154, 2013. Cited on p. 582

[511] Galeano, David, "Rendering Optimizations in the Turbulenz Engine," in Patrick Cozzi, ed., *WebGL Insights*, CRC Press, pp. 157–171, 2015. Cited on p. 795, 796, 802, 803

[512] Gallagher, Benn, and Martin Mittring, "Building Paragon in UE4," *Game Developers Conference*, Mar. 2016. Cited on p. 527, 556, 637

[513] Garcia, Ismael, Mateu Sbert, and Lázló Szirmay-Kalos, "Tree Rendering with Billboard Clouds," *Third Hungarian Conference on Computer Graphics and Geometry*, Jan. 2005. Cited on p. 563

[514] Garland, Michael, and Paul S. Heckbert, "Fast Polygonal Approximation of Terrains and Height Fields," Technical Report CMU-CS-95-181, Carnegie Mellon University, 1995. Cited on p. 708, 877

[515] Garland, Michael, and Paul S. Heckbert, "Surface Simplification Using Quadric Error Metrics," in *SIGGRAPH '97: Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, ACM Press/Addison-Wesley Publishing Co., pp. 209–216, Aug. 1997. Cited on p. 708

[516] Garland, Michael, and Paul S. Heckbert, "Simplifying Surfaces with Color and Texture Using Quadric Error Metrics," in *Proceedings of IEEE Visualization 98*, IEEE Computer Society, pp. 263–269, July 1998. Cited on p. 706, 707, 708

[517] Garland, Michael, *Quadric-Based Polygonal Surface Simplification*, PhD thesis, Technical Report CMU-CS-99-105, Carnegie Mellon University, 1999. Cited on p. 709

[518] Gautron, Pascal, Jaroslav Křivánek, Sumanta Pattanaik, and Kadi Bouatouch, "A Novel Hemispherical Basis for Accurate and Efficient Rendering," on *Proceedings of the Fifteenth Eurographics Conference on Rendering Techniques*, Eurographics Association, pp. 321–330, June 2004. Cited on p. 404

[519] Geczy, George, "2D Programming in a 3D World: Developing a 2D Game Engine Using DirectX 8 Direct3D," *Gamasutra*, June 2001. Cited on p. 550

[520] Gehling, Michael, "Dynamic Skyscapes," *Game Developer*, vol. 13, no. 3, pp. 23–33, Mar. 2006. Cited on p. 549

[521] Geiss, Ryan, "Generating Complex Procedural Terrains Using the GPU," in Hubert Nguyen, ed., *GPU Gems 3*, Addison-Wesley, pp. 7–37, 2007. Cited on p. 171

[522] Geiss, Ryan, and Michael Thompson, "NVIDIA Demo Team Secrets—Cascades," *Game Developers Conference*, Mar. 2007. Cited on p. 171, 571

[523] Geldreich, Rich, "crunch/crnlib v1.04," *GitHub* repository, 2012. Cited on p. 870

[524] General Services Administration, "Colors Used in Government Procurement," Document ID FED-STD-595C, Jan. 16, 2008. Cited on p. 349

[525] Gerasimov, Philipp, "Omnidirectional Shadow Mapping," in Randima Fernando, ed., *GPU Gems*, Addison-Wesley, pp. 193–203, 2004. Cited on p. 234

[526] Gershun, Arun, "The Light Field," Moscow, 1936, translated by P. Moon and G. Timoshenko, *Journal of Mathematics and Physics*, vol. 18, no. 2, pp. 51–151, 1939. Cited on p. 379

[527] Gibson, Steve, "The Distant Origins of Sub-Pixel Font Rendering," *Sub-pixel Font Rendering Technology*, Aug, 4, 2006. Cited on p. 675

[528] Giegl, Markus, and Michael Wimmer, "Unpopping: Solving the Image-Space Blend Problem for Smooth Discrete LOD Transition," *Computer Graphics Forum*, vol. 26, no. 1, pp. 46–49, 2007. Cited on p. 856

[529] Giesen, Fabian, "View Frustum Culling," *The ryg blog*, Oct. 17, 2010. Cited on p. 983, 986

[530] Giesen, Fabian, "A Trip through the Graphics Pipeline 2011," *The ryg blog*, July 9, 2011. Cited on p. 32, 42, 46, 47, 48, 49, 52, 53, 54, 55, 141, 247, 684, 701, 784, 1040

[531] Giesen, Fabian, "Fast Blurs 1," *The ryg blog*, July 30, 2012. Cited on p. 518

[532] Gigus, Z., J. Canny, and R. Seidel, "Efficiently Computing and Representing Aspect Graphs of Polyhedral Objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 6, pp. 542–551, 1991. Cited on p. 831

[533] Gilabert, Mickael, and Nikolay Stefanov, "Deferred Radiance Transfer Volumes," *Game Developers Conference*, Mar. 2012. Cited on p. 478, 481

[534] van Ginneken, B., M. Stavridi, and J. J. Koenderink, "Diffuse and Specular Reflectance from Rough Surfaces," *Applied Optics*, vol. 37, no. 1, Jan. 1998. Cited on p. 335

[535] Ginsburg, Dan, and Dave Gosselin, "Dynamic Per-Pixel Lighting Techniques," in Mark DeLoura, ed., *Game Programming Gems 2*, Charles River Media, pp. 452–462, 2001. Cited on p. 211, 221

[536] Ginsburg, Dan, "Porting Source 2 to Vulkan," *SIGGRAPH An Overview of Next Generation APIs course*, Aug. 2015. Cited on p. 814

[537] Giorgianni, Edward J., and Thomas E. Madden, *Digital Color Management: Encoding Solutions*, Second Edition, John Wiley & Sons, Inc., 2008. Cited on p. 286, 291

[538] Girshick, Ahna, Victoria Interrante, Steve Haker, and Todd Lemoine, "Line Direction Matters: An Argument for the Use of Principal Directions in 3D Line Drawings," in *Proceedings of the 1st International Symposium on Non-photorealistic Animation and Rendering*, ACM, pp. 43–52, June 2000. Cited on p. 672

[539] Gjøl, Mikkel, and Mikkel Svendsen, "The Rendering of *Inside*," *Game Developers Conference*, Mar. 2016. Cited on p. 521, 524, 527, 572, 587, 604, 609, 892, 1010

[540] Glassner, Andrew S., ed., *Graphics Gems*, Academic Press, 1990. Cited on p. 102, 991

[541] Glassner, Andrew S., "Computing Surface Normals for 3D Models," in Andrew S. Glassner, ed., *Graphics Gems*, Academic Press, pp. 562–566, 1990. Cited on p. 695

[542] Glassner, Andrew, "Building Vertex Normals from an Unstructured Polygon List," in Paul S. Heckbert, ed., *Graphics Gems IV*, Academic Press, pp. 60–73, 1994. Cited on p. 691, 692, 695

[543] Glassner, Andrew S., *Principles of Digital Image Synthesis*, vol. 1, Morgan Kaufmann, 1995. Cited on p. 372, 512, 1010

[544] Glassner, Andrew S., *Principles of Digital Image Synthesis*, vol. 2, Morgan Kaufmann, 1995. Cited on p. 268, 271, 280, 372, 512

[545] Gneiting, A., "Real-Time Geometry Caches," in *ACM SIGGRAPH 2014 Talks*, ACM, article no. 49, Aug. 2014. Cited on p. 92

[546] Gobbetti, Enrico, and Fabio Marton, "Layered Point Clouds," *Symposium on Point-Based Graphics*, Jun. 2004. Cited on p. 573

[547] Gobbetti, E., D. Kasik, and S.-E. Yoon, "Technical Strategies for Massive Model Visualization," *ACM Symposium on Solid and Physical Modeling*, June 2008. Cited on p. 879

[548] Goldman, Ronald, "Intersection of Two Lines in Three-Space," in Andrew S. Glassner, ed., *Graphics Gems*, Academic Press, p. 304, 1990. Cited on p. 990

[549] Goldman, Ronald, "Intersection of Three Planes," in Andrew S. Glassner, ed., *Graphics Gems*, Academic Press, p. 305, 1990. Cited on p. 990

[550] Goldman, Ronald, "Matrices and Transformations," in Andrew S. Glassner, ed., *Graphics Gems*, Academic Press, pp. 472–475, 1990. Cited on p. 75

[551] Goldman, Ronald, "Some Properties of Bézier Curves," in Andrew S. Glassner, ed., *Graphics Gems*, Academic Press, pp. 587–593, 1990. Cited on p. 722

[552] Goldman, Ronald, "Recovering the Data from the Transformation Matrix," in James Arvo, ed., *Graphics Gems II*, Academic Press, pp. 324–331, 1991. Cited on p. 74

[553] Goldman, Ronald, "Decomposing Linear and Affine Transformations," in David Kirk, ed., *Graphics Gems III*, Academic Press, pp. 108–116, 1992. Cited on p. 74

[554] Goldman, Ronald, "Identities for the Univariate and Bivariate Bernstein Basis Functions," in Alan Paeth, ed., *Graphics Gems V*, Academic Press, pp. 149–162, 1995. Cited on p. 781

[555] Gollent, M., "Landscape Creation and Rendering in REDengine 3," *Game Developers Conference*, Mar. 2014. Cited on p. 262, 263, 873

[556] Golub, Gene, and Charles Van Loan, *Matrix Computations*, Fourth Edition, Johns Hopkins University Press, 2012. Cited on p. 102

[557] Golus, Ben, "Anti-aliased Alpha Test: The Esoteric Alpha to Coverage," *Medium.com* website, Aug. 12, 2017. Cited on p. 204, 205, 206, 207

[558] Gomes, Abel, Irina Voiculescu, Joaquim Jorge, Brian Wyvill, and Callum Galbraith, *Implicit Curves and Surfaces: Mathematics, Data Structures and Algorithms*, Springer, 2009. Cited on p. 583, 683, 751, 753, 781, 944

[559] Gonzalez, Rafael C., and Richard E. Woods, *Digital Image Processing*, Third Edition, Addison-Wesley, 2007. Cited on p. 130, 543, 661

[560] Gonzalez-Ochoa, C., and D. Holder, "Water Technology in *Uncharted*," *Game Developers Conference*, Mar. 2012. Cited on p. 879

[561] Gooch, Amy, Bruce Gooch, Peter Shirley, and Elaine Cohen, "A Non-Photorealistic Lighting Model for Automatic Technical Illustration," in *SIGGRAPH '98: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, ACM, pp. 447–452, July 1998. Cited on p. 103