基于光学映射虚物体的并行绘制

李亚峰 秦开怀
(清华大学计算机科学与技术系,北京 100084)

摘 要 尽管当前基于全局光照模型的图形绘制方法可以渲染出高质量的图象,但因为其计算量巨大,难以适用于诸如建筑物漫游、虚拟现实等对绘制速度有严格要求的场合.为此引入光学映射虚物体的概念,利用构建在联网 PC 机上的集群系统,并行创建反射和折射虚物体,然后利用集群中各节点的图形加速硬件,像处理实际三维物体一样绘制这些虚物体,可以快速地绘制出反射/折射效果的图象.实验结果证明,该方法利用 CUP 的计算能力和图形硬件的加速特性实现了真实感图形的高性能绘制,特别适用于诸如建筑物漫游、计算机动画和虚拟现实等要求交互式绘制的场合.

关键词 计算机图象处理(520·6040) 虚物体 集群 硬件加速的绘制 并行绘制
中图法分类号: TP391.4 **文献标识码**: A **文章编号**: 1006-8961(2003)12-1432-06

Parallel Rendering Based on Optical-mapping Virtual Objects

LI Ya-feng, QIN Kai-huai
(*Department of Computer Science & Technology, Tsinghua University, Beijing* 100084)

Abstract Current rendering algorithms of global illumination models can generate photo-realistic images, but the high computing cost may discourage their applications in walk-through and virtual reality etc. Especially, rendering specular reflections and refractions is always a problem in computer graphics community. In this paper, virtual optical-mapping objects are introduced, and a unified method for generating virtual objects of reflections and refractions is proposed. The virtual objects are treated in the same way as real objects in the scene, so the reflection and refraction images can be rendered by the graphical hardware. Cluster based connected PCs is utilizes as parallel platform, and task scheme and load balance are discussed. Virtual objects are generated by CPU and rendered by graphical hardware. So, this method utilizes simultaneously both CPUs and graphical hardware on nodes of clusters to achieve high performance rendering. In the last, examples demonstrate this method is powerful for interactive applications such as real-time walk-throughs in buildings, animation and virtual reality.

Keywords Virtual objects, Clusters, Hardware-accelerated rendering, Parallel rendering

0 引 言

生成具有照片质量的图象是计算机图形学研究的目标之一.基于全局光照模型的光线跟踪和辐射度等方法虽然可以精确地绘制出镜面反射、折射、阴影和色渗等效果,但计算量却很庞大.尽管各国学者从不同角度研究了图形绘制的加速问题,但快速地绘制出大规模复杂场景的高质量真实感图象仍然超出了单机系统的绘制能力.这样,并行计算机就自然成为了一个合适的选择.目前,已经实现了光线跟踪和辐射度等算法的并行化,并对数据划分和任务划分策略等进行了广泛的研究[1].随着 PC 机性能的提高和网络技术的发展,集群系统(clusters)[2]成为进行高性能计算的一种新选择.所谓集群是将多台独立的计算机通过高速网络连接起来,这些计算机可以是单机或多处理器系统(PC、工作站或 SMP),每个结点都有自己的存储器、I/O 设备和操作系统,整个集群系统对用户和应用来说可以看作是一个虚拟的并行机(virtual parallel machine),它可以提供

基金项目:国家自然科学基金资助项目(69873025)
收稿日期:2002-04-16;改回日期:2003-07-17

低价高效的高性能环境和快速可靠的服务. 这为快速绘制大规模复杂景物提供了一种新的有效手段.

近些年，图形硬件加速卡的绘制能力不断提高并成为个人机的基本配置，这些图形硬件可以直接支持隐藏面消隐、简单的局部光照模型、多边形裁剪和纹理映射等操作，所以，基于图形硬件加速的真实感图形绘制成为了一个新的研究热点. 如利用为图形硬件所支持的软件图形标准平台 OpenGL 交互式地绘制出了平面场景的接近光线跟踪的效果[3]. 这样，并行环境下利用图形加速卡成为高性能绘制的一种有效方法，有学者提出了利用现有的商业图形加速卡进行并行绘制[4,5]的方法.

本文引入光学映射虚物体[6]的概念，利用基于联网 PC 的集群系统作为平台并行创建虚物体，然后，利用集群系统各节点的图形加速卡绘制实际物体和虚物体，这样，整个系统充分利用了并行虚拟机中各节点 CPU 的计算能力和图形加速硬件来绘制真实感图形，从而获得了更快的绘制速度.

1　基于光学映射虚物体的绘制

1.1　虚物体的概念

日常生活中，在观察像镜子这样具有反射性质的物体时，除看到反射体外，还看到被反射物体在镜子中所成的像. 如果镜子是平面的，可以假想在镜子后面存在着与真实世界关于镜子对称的虚拟世界，最初，Wallace 通过引入虚世界来解决漫射光能的间接传递问题[7]. 有些学者则用这种思想完成了基于图形硬件加速的平面反射图象绘制[8]，然而，这些方法只能处理诸如镜子地板之类的简单场景. Ofek[9]则更详细地讨论了曲面反射虚物体的概念：对于一反射体 R，给定视点 E，那么视点 E 处所见到的真实物体 O 在反射体中所成的像 O' 即称为 O 的反射虚物体. 对于平面反射体时，被反射物体关于反射平面对称，虚物体可以很容易求出，当反射物体为哈哈镜这样的非平面反射体时，真实世界与虚拟世界的对称关系就不再成立，虚拟世界表现为几何变形后的真实世界. Ofek[9]求解曲面反射形成的虚物体的方法是：用多边形近似表示曲面反射体，将被反射物体网格化，根据几何光学定律求出网格每一个顶点反射后的像，可称之为虚顶点，将这些虚顶点按原来的顺序连接起来构成的虚多边形就是真实物体反射后的几何模型，称之为反射虚物体. 但这种方法并未讨论折射的情况，而且随着递归深度的增加，求解过程不仅会变得很复杂，而且累积误差也越来越严重，因此这里采用光学映射[6]的方法以类似光线跟踪的过程计算虚顶点，可以统一处理反射和折射虚物体. 图 1 显示的是从视点 E 出发沿视线 L 进行光线跟踪的过程.

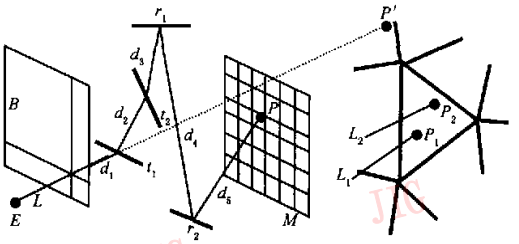

图 1

图 1 中，B 为投影平面；t_1、t_2 是折射平面；r_1，r_2 是反射平面；M 是物体平面. L 经若干次反射、折射后交漫反射物体平面 M 于点 P. 根据光学映射的计算方法，从视点 E 处所看到的点 P 成像于 P'，P' 在 L 的延长线上，与 E 的距离就是光线 L 经过若干次反射折射所走过的距离之和，P' 即为 P 的虚顶点. 由于 P 一般不会恰好为物体面片的顶点，所以把 P' 看作是离 P 最近的网格顶点 V 的虚顶点. 求得虚顶点的空间位置后，记入其所属反射/折射物体的虚物体链表中. 当有新的光线交于某网格时，重新进行这一最近顶点的比较和选定.（与光线跟踪不同，这一过程不必判定离视点最近的交点；不计算交点的光亮度）. 求得虚顶点后，连接所有虚顶点所构成的多边形网格即得到相应的虚物体. 每个虚物体都对应于真实场景中的实际物体，可以认为二者材质属性相同，差别仅在于几何形状的改变，所以我们把反射虚物体和折射虚物体统称为光学映射虚物体. 如果虚物体仍具有反射/折射属性（实际上是虚物体对应的真实物体具有反射/折射属性），则可以采用同样的方法定义二次虚物体. 可以看出，每个反射/折射体对应一组虚物体，而物体可能同时具有反射和折射属性.

1.2　绘制

创建虚物体后，把反射体和折射体看作是观察虚物体的一个“观察窗口”，利用图形硬件，像绘制实际物体一样绘制虚物体，根据“观察窗口”由 OpenGL 的 Stencil Buffer 对虚物体进行裁剪和层次关系的控制[9]，便可得到反射和折射的图象. 根据

物体的反射率/折射率将这个反射/折射的图象与真实物体生成的图象进行 ALPHA 混合就会得到逼真的反射和折射效果.

1.3 阴影

虽然通过创建虚物体,并利用图形硬件绘制出了镜面反射和折射效果,但由于目前的图形硬件只支持局部光照模型和 Z-buffer 消隐,而没考虑物体之间的遮挡关系,所以无法表现出阴影效果.对于静态场景,阴影是固定的,为加快实时绘制速度,采用了预处理的方法.由于计算虚物体之前已经对场景中的物体进行了网格剖分,所以可以采用 Aeherton 的阴影算法[10]将场景中的多边形划分为光照多边形和阴影多边形,即依次取景物多边形作为裁剪窗口对位于其后面的景物多边形进行裁剪,显然,后面多边形中位于窗口内的部分将因遮挡而不为光源所照射,这部分即成为阴影多边形.这一过程可以利用 BSP 树提高速度[11].绘制时,对于每个多边形首先判断其是否为阴影多边形,如果是,则关闭该光源,只应用环境光绘制该网格.事实上,由于在计算虚物体时已经对场景进行了比较细致的网格剖分,所以如果对阴影的质量要求不高,可以采用下面的简化算法.首先,以光源为视点,以各面片(即场景中物体剖分后得到的网格)的编号为颜色值进行无光照的绘制,那么,可见或部分可见的面片即为光源所照射到的,而不可见的面片则处于该光源的阴影区域,为阴影多边形,即根据绘制结果图象中各像素的颜色值就可以挑选出阴影多边形和光照多边形;然后把这个结果保存下来,再采用 1.2 节描述的方法进行绘制.

2 并行创建虚物体

基于光学映射虚物体的并行绘制算法主要包括 3 个过程:(1)剖分场景中物体并确认光照多边形和阴影多边形;(2)并行创建虚物体;(3)利用图形硬件绘制实际物体和虚物体得到最终图象.为实现交互绘制,采用联网 PC 机的机群系统作为并行平台,用并行算法创建虚物体.其中,算法第 2 步主要为光线与场景中各物体的求交运算,由各节点的 CPU 完成,而第 3 步则由各节点的图形加速卡完成.

上节中介绍的创建虚物体的算法与光线跟踪过程是一致的,不同之处在于前者是跟踪每根光线求出对应像素的光亮度,而本文的方法是求出每根光线对应的虚顶点.所以并行创建虚物体的过程可以借鉴并行光线跟踪算法[12].对于光线跟踪,由于各像素的光亮度计算是独立的,所以光线跟踪的并行化[1]是自然的,主要的问题之一是适当的任务划分以实现负载平衡.静态任务划分和动态划分是主要的两种任务划分方法,前者是在计算开始前将任务分配给各个节点,而后者则是在计算开始后,根据系统当前的运行情况动态地将任务分配给各节点,这种方法可以利用实时的系统信息,更容易实现负载平衡,但是如果任务动态调度算法过于复杂反而会导致系统效率下降.对于基于网络的分布式并行虚拟机环境,各节点的计算能力差异很大,诸如数据延迟等网络环境变化也难以预测,所以在计算前将任务分配给各节点机的静态任务分配方案难以实现负载平衡,而采用动态任务分配策略,可以根据运行时的网络状态和各节点机运行情况动态调整任务分配,尽可能地使每个进程承担与其计算能力相匹配的任务,从而实现负载平衡.另外,由于网络中各节点机之间的通讯时间开销要大于多处理器并行机中各处理器之间的通讯时间,而且当前的个人机都有较大的内存,数据划分不像以前那么重要,所以,在每个节点都保留完整的场景模型数据,这样,运行时各节点间可以避免交换光线数据,从而降低了系统的通讯代价.

采用主从运算方式,集群系统的并行虚拟机由一个主进程和多个从进程构成,其中主进程运行于主机,它除负责与用户的交互外,还负责向各从进程分配任务并收集各进程的运算结果.当完成全部任务后,主进程向各从进程发送结束标志,从进程接收到结束标志后退出执行.

为改善负载平衡并尽可能降低通讯开销,在运算过程中记录集群系统中每个节点的性能指数,并在后来的动态任务分配过程中根据这个性能指数动态调整为其分配的任务量大小.记节点 i 已经跟踪过的光线数为 N_i,本次计算的光线数为 n_i,本次的计算时间为 t_i,则定义节点 i 的加权平均响应时间为其性能指数

$$T_i^{(k+1)}=\frac{T_i^{(k)}\times N_i}{N_i+n_i}+\frac{t_i}{n_i} \tag{1}$$

初始情况下,$T_i^{(0)}=0$.

系统运行后,主进程首先从系统参数中获得从机的数目,然后根据这个数目进行首次任务分配,任务量为一个基本任务单位 $Task$(一般为 8×8 根光

线).主进程将待计算光线的参数分别送至各从进程,各从进程接收到主进程送来的数据后,分别计算各光线对应的虚顶点.首次任务分配完毕后,剩下的任务以动态竞争的方式进行,当有从进程完成任务后立即以广播的方式向包括主进程在内其他进程发送结果(虚顶点),主进程接到从进程 i 的计算结果后,根据式(1)更新 i 的性能指数,如果有还未完成的任务(未跟踪的光线),即根据它的性能指数为其分配新任务,任务量为

$$M_i=\begin{cases}\left\lfloor\dfrac{\left(\sum_{j=0}^{N}T_j\right)/N}{T_i}\right\rfloor * Task & N\neq 0\\ 1*Task & N=0\end{cases}\qquad(2)$$

其中,N 为 $T_i\neq 0$ 的从进程个数.

当全部光线被跟踪完毕后,主进程通知各从进程结束.这样,通过动态的任务分配实现了负载平衡.当完成对所有采样光线的追踪后,每个节点都得到了整个场景的虚物体模型.

3　并行绘制

图形绘制可以看作是给出某一环境中所有物体的数学模型描述,然后计算每一物体对应特定视平面的像素,这是一种排序问题[13].这一过程主要包括几何处理(坐标变换,裁剪等)和光栅化(光照,扫描转换等)两个过程.只有几何处理和光栅化都被并行执行,才认为这个绘制系统是完全并行的.根据在绘制流水线的不同阶段完成空间坐标到屏幕坐标的排序,可以把并行绘制分为 sort-first、sort-middle 和 sort-last 3 种方法[12].针对目前通用的商业图形加速卡,sort-last 方法更为合适.Humpherys 等人将图形绘制任务分为 Compute-limited 和 Render-limited 等[5],对应地,可以将 clusters 中的各节点在逻辑上分为计算单元和绘制单元,前者完成创建虚物体的工作,这部分任务主要由 CPU 完成,而绘制任务,则由各节点的图形加速硬件来完成.

如图 2 所示,本文算法所创建的虚物体为树形(tree)结构,整个场景模型的数据是以基本绘制图元(primitive)为结点的森林(forest).其中根节点为实际物体,森林中各层节点为不同反射/折射虚物体,如果采用 sort-last 方法对虚物体场景进行并行绘制,实际上只有处于虚物体链表(list)中同一层次的虚物体由不同节点分别绘制后进行图象合成时才

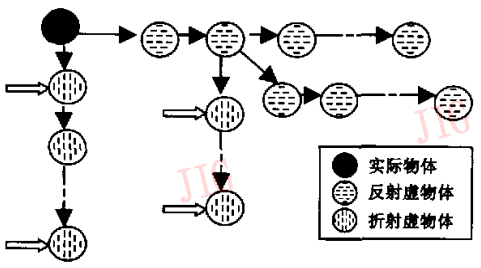

图 2　虚物体的链表结构

需要进行深度比较,而对于不同层次虚物体的绘制结果只要进行简单的叠加或者 alpha blending 即可,因为进行深度比较需要(R,G,B,$alpha$,Z)5 维数据,而 Alpha Bending 操作只需要(R,G,B,$alpha$)4 维数据,利用这个性质,把位于同一层次虚物体的绘制分配到同一个节点绘制,就可以避免图象合成时的 5 维数据传输,从而降低绘制时的通讯量,提高了这个系统的效率.为达到这一目的,在遍历场景模型森林进行场景图形数据库划分时,采用广度优先遍历的策略,依次根据各绘制节点硬件性能为其分配相应数目的图元,分配的结果是各绘制节点得到它所绘制图元序列在虚物体链表中的指针,这种分配策略可以尽量保证处于同一反射/折射层次的虚物体被分配到同一绘制节点,但是仍然会有一部分处于同一层次的虚物体被分配到不同绘制节点,所以将绘制结果数据分为两类,一类数据和传统的 sort-last 方法一样,除包括像素的颜色值外还包括 Z-buffer 值,而另一类数据是已经被完整绘制的部分,无需再保留 Z-buffer 值.

4　实现与结果

MPI(Message Passing Interface)[14]是目前最常用的并行软件平台之一,它定义了基于消息传递机制的分布式并行计算规范,具有较高的通讯效率和程序可移植性,特别适合异构环境实现.采用 Argonne National Laboratory 和 Mississippi State University 共同开发的 MPICH 作为并行软件支撑环境实现了创建虚物体的并行算法.硬件平台为通过网络相联的 PC 机,每台机器的配置为:CPU,P Ⅱ 450MHz;图形卡支持 OpenGL;操作系统为 Windows98;网络速度为 10M/s.对于图 3、图 4 的模型,创建虚物体运行时间统计数据见表 1,图 5 给出了采用不同规模虚拟机的加速比.

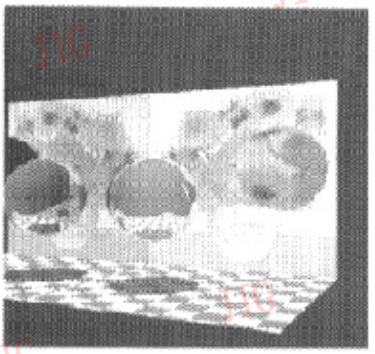

图 3 场景 1

图 4 场景 2

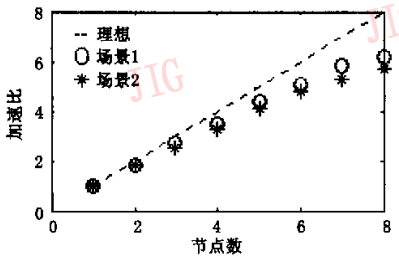

图 5 不同规模虚拟机的加速比

表 1 不同规模虚拟机创建虚拟物体的时间

单位:s

模型	节点数							
	1	2	3	4	5	6	7	8
场景 1	21.20	11.78	7.85	6.06	4.83	4.16	3.65	3.42
场景 2	78.44	43.58	29.05	23.77	19.13	16.34	14.80	13.76

对于本文的算法,创建虚物体为计算过程,由各节点 CPU 完成.而绘制任务由图形加速硬件执行,绘制的速度取决于场景的复杂度和图形加速卡的性能.现在的图形加速卡每秒可以绘制数百万个面片,所以对一般创建了虚物体的场景在显示阶段都可以达到实时绘制的速度.例如对于场景 2,如果使用传统的光线跟踪方法绘制,在单机上运行需要 118.56s,而采用光学映射虚物体的方法,预处理创建虚物体需要 78.44s,绘制时间仅为 0.052s.从实验结果可以看出,并行创建虚物体的算法具有较好的线性加速比,但随着节点的增加,通讯开销随之增大,会导致系统效率下降.

5 结论及展望

与传统的光线跟踪方法不同,本文提出的算法把 tracing 和 shading 过程分开,分别交给 CPU 和图形加速硬件来完成,这样,不但利用了各节点的 CPU 计算能力,而且还充分利用了各节点绘制时的硬件加速特性,这相当于针对 Humpherys 提出的 Compute-limited 和 Render-limited[5] 同时并行化,从而提高了整个系统的绘制速度.

对于基于集群这种松耦合的分布式系统,由于节点间的通讯速度小于 CPU 的计算速度和图形硬件的绘制速度,因此,通讯效率仍然是系统的瓶颈.可以采用粗粒度(coarse grain)的任务划分,同时在计算时避免过多的通讯和同步操作,以降低系统的通讯开销.

本文通过基于联网 PC 的集群系统并行建立光学映射虚物体,然后利用集群中各节点的图形加速硬件实现了照片质量真实感图象的交互绘制.经实验证明其是进行真实感图形高性能绘制的有效手段.随着图形硬件的发展以及网络性能的提高,这种方法必将在建筑物漫游、虚拟现实和动画设计等领域发挥更大的应用.

参 考 文 献

1 潘志庚.分布式并行图形处理技术[M].北京:人民邮电出版社,1997.

2 郑纬民,汤志忠.计算机系统结构[M].北京:清华大学出版社,2001.

3 Diefenbach P J, Badler N I. Pipeline rendering: interactive refraction, reflections and shadows[J]. Displays, 1994,15(3): 173~180.

4 Blanke W, Bajaj C, Fussell D, *et al*. Metabuffer: a scalable multiresolution multidisplays 3-D graphics system using commodity rendering engines[R]. TR2000-16, University of Texas at Austin,2000.

5 Greg Humpherys, Matlhew Eldridge, Ian Buck, *et al*. WireGL: a scalable graphics system for clusters[J]. Computer Graphics,

2001,20(3):129～140.
6 曾旭. 计算机实时真实感图形技术及其在建筑 CAD 中的应用[D]. 北京:清华大学, 2000.
7 Wallace J R, Chen M F, Greeberg D P. A two-pass solution to the rendering equation: a synthesis of ray tracing and radiosity methods[J]. Computer Graphics. 1987,21(4): 311～320.
8 Tom McReynolds. Advanced graphics programming techniques using OpenGL[R]. SIGGRAPH'98 Course notes. April, 1998.
9 Ofek E, Rappoport A. Interactive reflections on curved objects [A]. In:SIGGRAPH'98[C] Orlando, Florida, 1998:333～342.
10 Atherton P R, Weilier K, Greenberg D P. Polygon shadow generation[J]. Computer Graphics, 1978,12(3):275～281.
11 Norman Chin, Steven Feiner. Near real-time shadow generation using BSP trees[J]. Computer Graphics, 1989,23(3):99～106.
12 Mounir Hamdi et al. Parallel computing on an Ethernet cluster of workstations: opportunities and constraints[J]. The Journal of Supercomputing, 1999,13(2):111～132.
13 Samanta R. A sorting classification of parallel rendering[J]. IEEE Computer Graphics & Application, 1994,14(4):23～32.
14 都志辉. 高性能计算并行编程技术 MPI 并行程序设计[M]. 北京:清华大学出版社,2001.

李亚峰　1973 年生,1998 年于哈尔滨工业大学数学系获理学硕士学位,现为清华大学博士研究生. 主要研究方向为真实感图形绘制、并行计算等.

秦开怀　1958 年生,清华大学计算机科学与技术系高性能计算研究所,教授,博士生导师,1990 年获博士学位. 1999～2000 年,美国哈佛大学访问教授. 主要研究兴趣包括计算机图形学、计算机辅助几何造型、小波分析、医学图象处理、可视化、动画、虚拟现实(VR)、CAD/CAM 等.

ELSA 全新推出跨世代显卡幻雷者 980XT/960XT

近日，艾尔沙公司推出 ELSA 幻雷者 980XT 与 960XT 新产品，两款新品搭载最新的 ATi Radeon 9800XT 与 Radeon 9600XT 显示芯片，属于目前显卡中最顶级的产品。

高效能的 ELSA 幻雷者 980XT 搭载 ATi Radeon 9800XT 显示芯片，核心频率达到 412MHz，比 Radeon 9800Pro 高出 32MHz，256MB DDR 显存及绝佳的八条同步运算的绘图管线，每秒可处理 30 亿万个像素；高达 256 位的显存带宽，为显卡的高速运算提供了最佳的保证。新一代的 SMARTSHADER™ 2.1 及 SMOOTHVISION™ 2.1 能够完全支持 OpenGL® API 及 Microsoft DirectX® 9.0 以及先进的全屏反锯齿功能，每秒可处理 180 亿个全景取样运算，展现出空前的运算效率。

ELSA 幻雷者 960XT 搭载 ATi Radeon 9600XT 显示芯片，核心频率达到 500MHz，比 Radeon 9600Pro 的 275MHz 高出 125MHz，是 ATi 第一款使用 0.13 微米制程的产品，耗电量低，核心频率较高，而发热量却低了许多。幻雷者 960XT 拥有四条同步运算的绘图管线；可选择 128MB 或 256MB DDR 显存；拥有 SMARTSHADER™ 2.0、SMOOTHVISION™2.1 与 HyperZ™ III+功能，同样支持 DirectX® 9.0 及 OpenGL® API 应用程序和先进的全屏反锯齿功能。

全新的幻雷者 980XT 与 960XT 可通过软件执行的 Overdrive 改变传统的超频方法（新版驱动程序 Catalyst V3.8 加入使用风扇控制与硬件监控等功能进行动态超频，一般称为 Overdrive）；通过其 SMARTSHADER™ 、SMOOTHVISION™ 、FULLSTREAM™ 及 VIDEO IMMERSION™ 等先进的技术，以及高度精确的颜色校正架构，为复杂度高的纹理架构改善了视觉品质，更进一步达到图形计算的准确品质与正确性，让玩家拥有如电影般的真实体验。