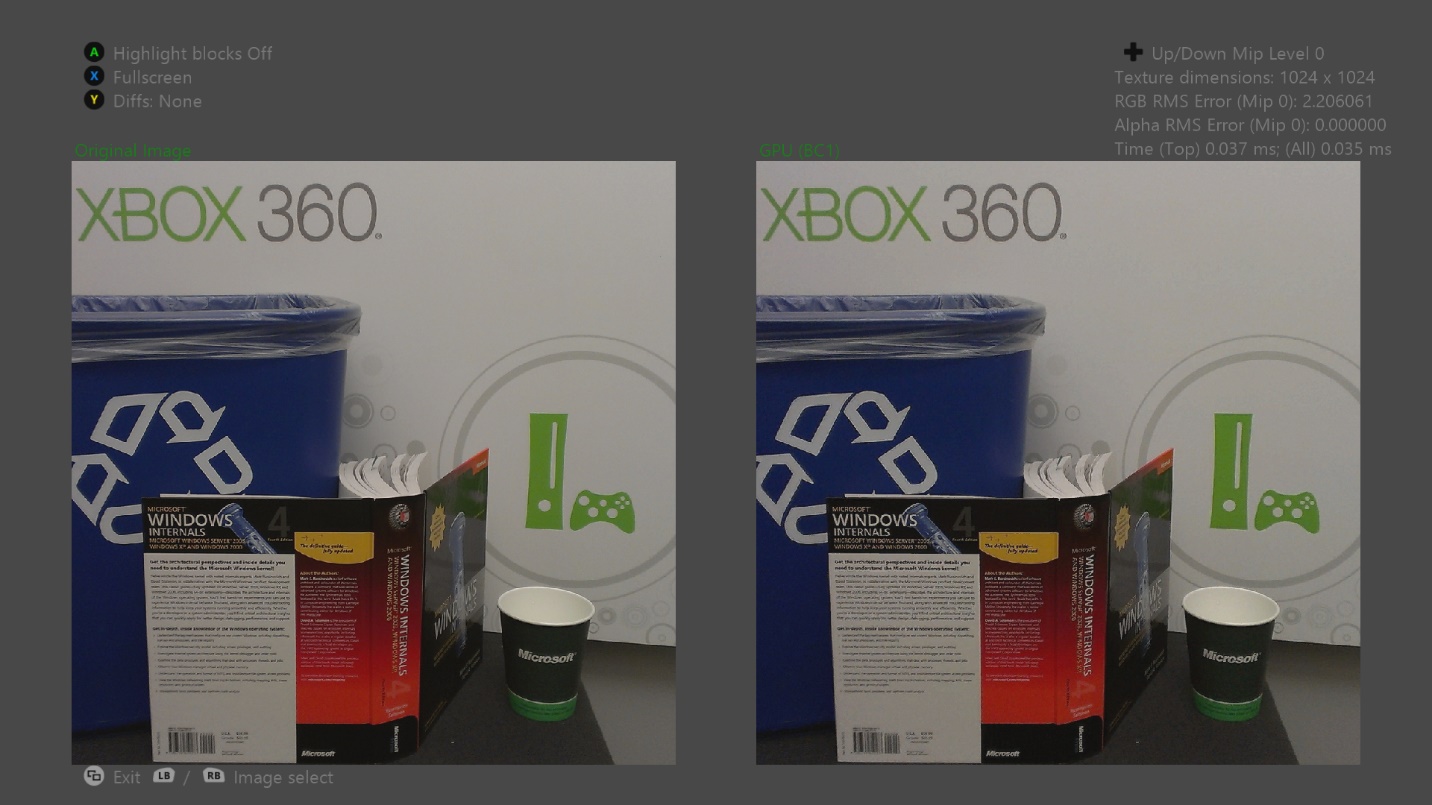


FastBlockCompress 样本

*此样本与 2017 年 3 月的 XDK 兼容。*

# 描述

# 此样本演示了如何使用 DirectCompute 在运行时执行基于经典*快速区块压缩*算法的 BC1、BC3 和 BC5 格式的快速纹理压缩。该样本还让您可以在运行和离线压缩模式之间切换，以比较视觉质量。



# 使用样本

|  |  |
| --- | --- |
| 操作 | 游戏手柄 |
| 上一张或下一张图片 | 左或右缓冲键 |
| 上一个或下一个压缩方法 | D-pad 左或右 |
| 上一个或下一个 mip 级别 | D-pad 下或上 |
| 移动相机 | 右杆 |
| 放大或缩小 | 左或右扳机键 |
| 全屏与并列显示 | X |
| 突显区块 | A |
| 循环差异模式 | Y |
| 退出 | 查看按键 |

# 背景

Xbox One 拥有 5 GB 统一内存，可使用专属应用程序，这比 Xbox 360 上的 512 MB 明显增长了 10 倍。很遗憾，IO 带宽和存储介质容量并未完全保持同步。蓝光媒体将容纳 49 Gb，比 Xbox 360 游戏光盘版本 3 的 7.8 GB 仅增加 6.3 倍。

这一事实结合流媒体安装的引入，意味着有效的压缩方法对于最小化加载时间以及将游戏资产打包至可用存储空间仍然很重要。

通过使用离线图像压缩格式编码游戏纹理，标题通常可以节省大量存储空间。Xbox One 内置硬件 JPEG 解码器，使得 JPEG 成为有吸引力的选择。但是，JPEG 硬件会将纹理解码为内存中未压缩的 YUV 格式，这对于渲染而言并非最佳格式。如果使用此方法，那么标题需要在运行时将纹理重新压缩为 GPU 支持的区块压缩格式之一。

此样本使用 GPU 有效地将纹理压缩为 BC1、BC3 和 BC5 格式。历史上用于离线区块压缩的标准算法太慢以致于不能实时运行，而样本使用的算法为了速度形成显着的质量妥协。

由于内存带宽会阻碍当前算法，使用其他技术，可以实现显着的质量改进，而性能损失很小。

# 实施说明

# 样本中的每个 DirectCompute 压缩着色器都有三种变体：一个单 mip 版本、一个双 mip 版本以及一个三 mip 版本：

# 单 mip 着色器压缩源纹理的单一 mip。

# 双 mip 着色器读取源纹理的单一 mip，然后将 mip 向下采样至本地数据存储 (LDS) 内存。接下来，着色器压缩原始和向下采样版本，然后着色器在输出纹理中写入相应的 mip 级别。

# 此过程通过避免从第二个 mip 级别的源纹理进行读取来节省内存带宽。但实际上，性能增益很大程度上被新增着色器的复杂性和由于更高 GPR 和 LDS 使用而导致的占用率降低所抵消。

# 尾部 mip 着色器通过选择不同的线程来处理不同的 mip 级别，在单个调度呼叫中将源纹理的 mip 级别从 16×16 压缩至 1×1。

# 由于最小波前尺寸为 64 线程，因此在单独的调度呼叫中压缩每个尾部 mip 的技术将浪费大多数可用线程。通过使用仅一个波前和调度呼叫，尾部 mip 着色器避免了大量的浪费工作。

# Direct3D 不允许将 BC 格式的纹理绑定为 UAV，因此无法直接从计算着色器直接写入区块压缩纹理。该样本通过在区块压缩纹理相同的内存位置为可写格式的中间纹理编写别名，从而解决此限制。中间纹理为尺寸的四分之一，并且每个纹素对应于压缩纹理中的一个区块。

# 以这种方式编写纹理内存的别名需要两种纹理平铺模式和内存布局之间的精确匹配。此外，Direct3D 无视内存别名，因此 GPU 可同时调度多个绘制或调度呼叫，在不同资源上进行操作，在同一内存位置编写别名。

# 换句话说，写入中间纹理的着色器可与从别名区块压缩纹理读取的绘制呼叫同时进行调度。为防止这些危险，应手动插入适当栏。

离线压缩算法实现于 [DirectXTex](https://github.com/Microsoft/DirectXTex/)。

# 参考资料

Microsoft Advanced Technology Group.快速区块压缩样本。Xbox 360 SDK。2010 年 2 月。

Tranchida，Jason。[使用 GPU 进行实时纹理压缩](http://www.gdcvault.com/play/1012554/Texture-compression-in-real-time)。2010 年 GDC。2010 年 3 月。

van Waveren，J.M.P[。实时 DXT 压缩](https://software.intel.com/sites/default/files/23/1d/324337_324337.pdf)。Intel 软件网络。2006 年 5 月。

van Waveren，J.M.P. 和 Castaño，Ignacio。[实时 YCoCg-DXT 压缩](https://www.nvidia.com/object/real-time-ycocg-dxt-compression.html)。NVIDIA 开发者网站。2007 年 9 月。

van Waveren，J.M.P. 和 Castaño，Ignacio。[实时法线贴图 DXT 压缩](http://developer.download.nvidia.com/whitepapers/2008/real-time-normal-map-dxt-compression.pdf)。NVIDIA 开发者网站。2008 年 2 月。

# 更新历史记录

最初发布于 2015 年。于 2018 年 12 月使用 DirectX 12 进行重写。

# 隐私声明

编译和运行示例时，示例可执行文件的文件名将发送给Microsoft以帮助跟踪示例使用情况。要选择退出此数据收集，您可以删除Main.cpp中标记为“Sample Usage Telemetry”的代码块。

有关 Microsoft 隐私政策的更多信息，请参阅 [Microsoft 隐私声明](https://privacy.microsoft.com/zh-cn/privacystatement/)。