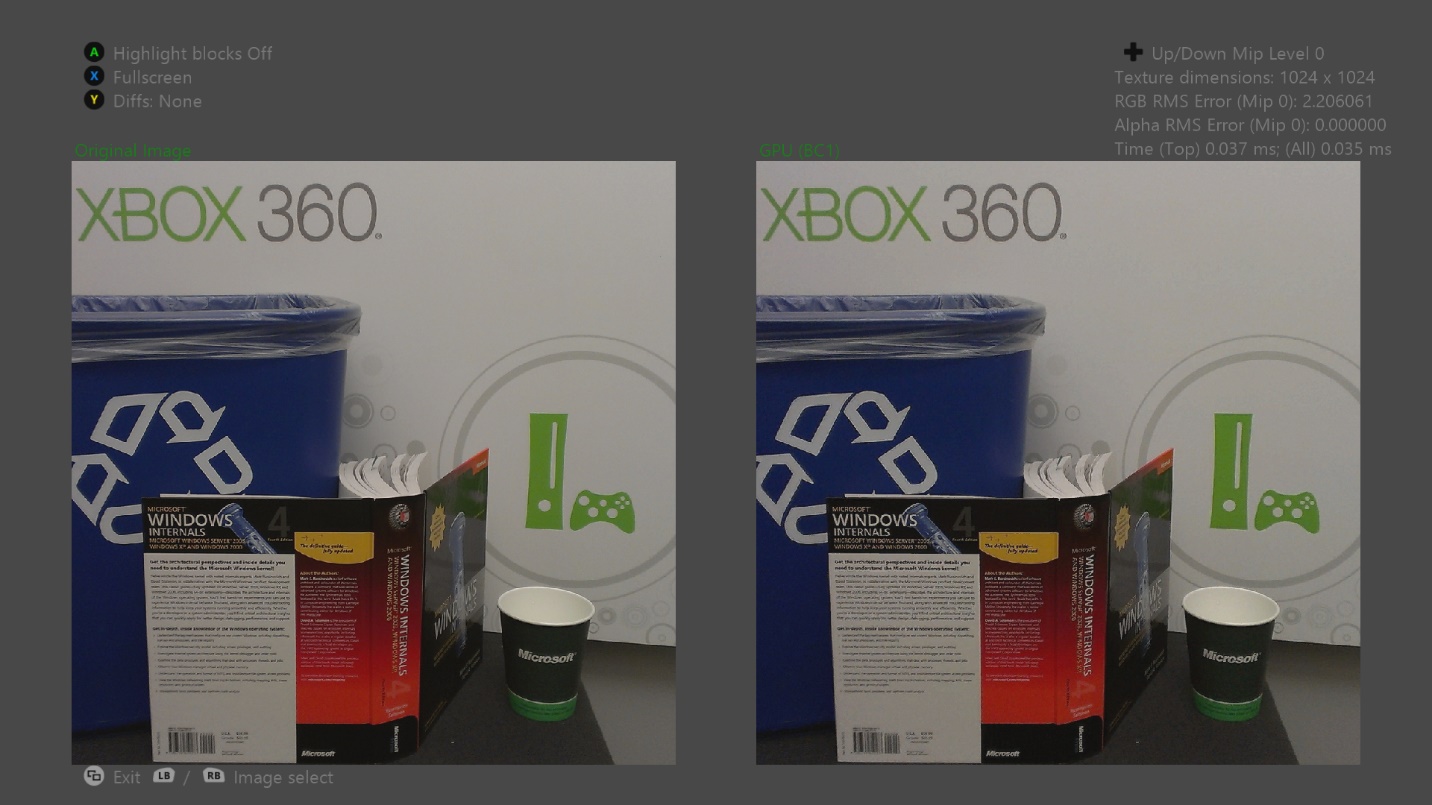


FastBlockCompress 샘플

*이 샘플은 2017년 3월 XDK와 호환됩니다.*

# 설명

# 이 샘플은 DirectCompute를 사용하여 런타임시 클래식 빠른 블록 압축 알고리즘을 기반으로 BC1, BC3 및 BC5 형식으로 빠른 텍스처 압축을 수행하는 방법을 보여줍니다. 이 샘플에서는 런타임 및 오프라인 압축 모드를 전환하여 시각적 품질을 비교할 수도 있습니다.



# 샘플 사용하기

|  |  |
| --- | --- |
| 액션 | 게임 패드 |
| 이전 또는 다음 이미지 | 왼쪽 또는 오른쪽 범퍼 |
| 이전 또는 다음 압축 방법 | 왼쪽 또는 오른쪽 D 패드 |
| 이전 또는 다음 밉 레벨 | D- 패드 위 아래 또는 위로 |
| 카메라 이동 | 오른쪽 스틱 |
| 확대 또는 축소 | 왼쪽 또는 오른쪽 트리거 |
| 전체 화면 대 나란히보기 | X |
| 블록 강조 표시 | A |
| Cycle diffs 모드 | Y |
| 종료 | 보기 버튼 |

# 배경

Xbox One은 Xbox 360에서 사용할 수있는 512MB의 10배에 달하는 독점적인 응용 프로그램 용으로 5GB의 통합 메모리를 제공합니다. 불행히도 입출력 대역폭과 저장 매체 용량은 아직 그 속도를 유지하지 못했습니다. 블루 레이 미디어는 Xbox 360 게임 디스크 버전 3에서 사용할 수 있는 7.8GB의 6.3 배에 불과한 49GB를 저장합니다.

스트리밍 설치의 도입과 결합된 이 사실은 효율적인 압축 방법이 로드 시간을 최소화하고 게임 자산을 사용 가능한 저장 공간에 압축하는 데 여전히 중요하다는 것을 의미합니다.

제목은 종종 오프라인 이미지 압축 형식을 사용하여 게임 텍스처를 인코딩하여 많은 양의 저장 공간을 절약할 수 있습니다. Xbox One에는 하드웨어 JPEG 디코더가 내장되어 있어 JPEG를 매력적인 선택으로 만듭니다. 그러나 JPEG 하드웨어는 텍스처를 비 압축 YUV 형식으로 메모리에서 디코딩하므로 렌더링에는 적합하지 않습니다. 이 방법을 사용하면 런타임에 텍스처가 GPU가 지원하는 블록 압축 형식 중 하나로 재압축해야 합니다.

이 샘플은 GPU를 사용하여 텍스처를 BC1, BC3 및 BC5 형식으로 효율적으로 압축합니다. 오프라인 블록 압축에 사용되는 표준 알고리즘은 역사적으로 실시간으로 실행하기에는 너무 느렸으며 샘플이 사용하는 알고리즘은 속도면에서 상당한 품질 저하를 낳습니다.

메모리 대역폭이 다른 알고리즘을 사용하여 현재 알고리즘을 병목 현상으로 만들기 때문에 성능 저하가 거의 없이 눈에 띄는 품질 향상을 달성할 수 있습니다.

# 구현 정보

# 샘플의 각 DirectCompute 압축 셰이더는 하나의 밉 버전, 두 밉스 버전 및 꼬리 밉스 버전의 세 가지 변형이 있습니다.

# 하나의 밉 셰이더는 소스 텍스처의 한 밉을 압축합니다.

# 두 개의 밉 셰이더는 원본 텍스처의 단일 밉을 읽고 립을 로컬 데이터 저장소 (LDS) 메모리로 다운 샘플링합니다. 다음으로 셰이더는 원본 샘플과 다운 샘플링된 버전을 모두 압축한 다음 셰이더가 해당 밉 레벨을 출력 텍스처에 씁니다.

# 이 프로세스는 두 번째 밉 레벨에 대한 소스 텍스처의 읽기를 피함으로써 메모리 대역폭을 절약합니다. 그러나 실제적으로 성능 향상은 셰이더 복잡성 증가와 GPR 및 LDS 사용 증가로 인한 점유율 감소로 대체로 상쇄됩니다.

# 테일 밉 셰이더는 서로 다른 밉 레벨에서 작동하도록 다른 스레드를 선택하여 단일 디스패치 호출에서 소스 텍스쳐의 16x16에서 1x1 밉 레벨을 압축합니다.

# 최소 파면 크기가 64 스레드이기 때문에 별도의 디스패치 호출에서 각 꼬리 밉을 압축하는 기술은 사용 가능한 스레드의 대부분을 낭비합니다. 하나의 웨이브 프론트 및 디스패치 콜만 사용하여 테일 밉 셰이더는 이러한 낭비되는 작업의 대부분을 피합니다.

# Direct3D에서는 UAV 형식으로 BC 형식 텍스처를 바인딩할 수 없으므로 계산 셰이더에서 블록 압축 텍스처에 직접 쓸 수 없습니다. 이 샘플에서는 쓰기 가능한 형식의 중간 텍스처를 블록 압축 텍스처와 동일한 메모리 위치에 앨리어싱을 적용하여 이 제한을 해결합니다. 중간 텍스처는 크기의 1/4이며, 각 텍셀은 압축된 텍스처의 블록에 해당합니다.

# 이러한 방식으로 텍스처 메모리를 앨리어싱하려면 두 텍스처의 타일링 모드와 메모리 레이아웃이 정확히 일치해야 합니다. 또한 Direct3D는 메모리 앨리어싱을 무시하므로 GPU는 동일한 메모리 위치에 별칭이 지정된 여러 리소스에서 작동하는 여러 그리기 또는 디스패치 호출을 동시에 예약할 수 있습니다.

# 즉, 중간 텍스처에 쓰는 셰이더는 앨리어싱된 블록 압축 텍스처에서 읽는 그리기 호출과 동시에 스케줄될 수 있습니다. 이러한 위험을 방지하려면 적절한 펜스를 수동으로 삽입해야 합니다.

오프라인 라인 압축 알고리즘은 [DirectXTex](https://github.com/Microsoft/DirectXTex/).

# 참조

Microsoft 고급 기술 그룹. 빠른 블록 샘플을 압축합니다. Xbox 360 SDK. 2010년 2월

트란 치다, 제이슨. [GPU를 사용하여 실시간으로 텍스처 압축](http://www.gdcvault.com/play/1012554/Texture-compression-in-real-time). GDC 2010. 2010년 3월

van Waveren, J.M.P[. 실시간 DXT 압축](https://software.intel.com/sites/default/files/23/1d/324337_324337.pdf). 인텔 소프트웨어 네트워크. 2006년 5월

van Waveren, J.M.P., and Castaño, Ignacio. [실시간 YCoCg-DXT 압축](https://www.nvidia.com/object/real-time-ycocg-dxt-compression.html). NVIDIA 개발자 사이트. 2007년 9월

van Waveren, J.M.P., and Castaño, Ignacio. [실시간 노멀 맵 DXT 압축](http://developer.download.nvidia.com/whitepapers/2008/real-time-normal-map-dxt-compression.pdf). NVIDIA 개발자 사이트. 2008년 2월

# 업데이트 기록

원래 2015년에 발표되었습니다. 이것은 2018년 12월 DirectX 12를 사용하여 재작성되었습니다.

# 개인정보처리방침

샘플을 컴파일하고 실행할 때 샘플 실행 파일의 이름이 Microsoft로 보내져 샘플 사용을 추적 할 수 있습니다. 이 데이터 수집을 거부하려면 Main.cpp에서 "샘플 사용 텔레메트리"라는 코드 블록을 제거하면 됩니다.

Microsoft의 개인 정보 취급 방침에 대한 일반적인 내용은 [Microsoft 개인 정보 취급 방침](https://privacy.microsoft.com/ko-kr/privacystatement/)을 참조하십시오.