**게임브리오를 이용한 MAX 작업시 주의사항**

**(부제 : 게임브리오 테크니컬 디렉터가 되는 길)**

 by neojzs

2009-10-08

1. **매터리얼에 2 Side 기능은 쓰지 않는다.** 즉, 필요하면 뒷면에 폴리곤을 더 만든다.

why ? 그래픽카드 성능이 좋아지면서 처리할 수 있는 폴리곤개수는 더욱 증가한 반면 렌더스테이트 전환(기능 on/off)은 여전히 시간을 잡아먹는 요소이기 때문이다.

예외사항 : 나무나 풀등과 같이 2 Side를 사용하지 않고는 힘든 오브젝트인 경우는 논외로 한다.

1. **매터리얼은 Gamebryo Shader 매터리얼을 꼭 사용한다.**

why ? 게임브리오가 맥스의 기본 매터리얼도 지원하나 Gamebryo Shader 매터리얼이 성능적이나 최적화측면에서 보다 효율적인 부분이 많으므로 오브젝트에 지정하는 매터리얼은 꼭 Gamebryo Shader 매터리얼을 쓰도록 한다.



1. **텍스처는 모두 DDS 포맷으로 한다. 그리고 밉맵은 생성하도록 한다.**

why ? DDS포맷은 가속과 메모리 최적화부분에서 효율적이기 때문

다만, DDS포맷도 여러 용도가 있으므로 필요에 따라 DDS포맷을 정한다. 그리고 DDS을 제작 할 때는 포토샵에 기본으로 있는 dds export가 아니라

<http://developer.nvidia.com/object/photoshop_dds_plugins.html>

에서 제공하는 포토샵 플러그인을 다운받아 사용해 제작해야 한다.

|  |  |
| --- | --- |
| DDS포맷 | 설명 |
| DXT1 RGB | **알파없는 이미지**인 경우 사용 |
| DXT1 ARGB | 알파가 있지만 1bit이므로 알파값이 딱 떨어지는 이미지에 사용  |
| DXT3 ARGB | 1bit 알파로는 부족한 이미지의 경우 |
| DXT5 ARGB | 부드러운 알파 이미지인 경우 사용 |

당연하게도 용량은 아래방향으로 커지므로 윗단계 포맷부터 고려를 해보고 어쩔 수

없을 때만 아래단계의 포맷을 사용하도록 한다.

주의사항 : 텍스처는 2의 승수(2, 4, 8, 16, 32, 64, 128, 256, 512, 1024...) 로 제작한다.

 예) 가능 - 256x128, 64x512, 128x128 등

 불가 – 200x500, 300x300 등

예외사항 : UI용이나 스크린이펙트에 사용하는 텍스처는 논외로 한다.

1. **Gamebryo Shader 매트리얼 설명**
2. Transparency Modes (투명도 모드 : 알파블렌딩)



 혼합모드는 Automatic을 쓰지 않는다. 투과되지 않는 재질인 경우 None을

지정하고 투과되는 재질인 경우 적절한 것은 선택하도록 한다.

why ? 오토 기능보다 수동 기능이 보다 효율적이고 정확한 세팅을 할 수

있기 때문이다.

**주의사항** : 투명도 모드를 None으로 했어도 알파테스트를 On시키면 알파값이

 있는 텍스처를 사용한 경우 투과되어 보이게 된다.



더 정확히 말하자면 해당경우는 알파블렌딩은 off 하고 알파테스팅만 하는

경우로 빠른 알파처리를 할 오브젝트인 경우에는 이렇게 세팅하는 것이 좋겠다.

(알파테스트가 알파블렌딩보다 기능은 부족하지만 빠르기때문)

단, 염두에 두어야 할 것은 알파경계부분이 자연스럽게 나오지 않는다는 것이다.

 이렇게 나오지 않고  이렇게 나온다는 의미

이럴 때 조금 더 자연스럽게 만드는 방법은 알파값이 보다 더 자연스러운

DXT5 포맷을 쓰고 Test Ref(0:완전투명~255:완전불투명)와 Test Mode를

적절히 변경하는 것이다. (적절히 변경한다는 것은 다양한 테스트를 통한

경험으로 지정한다는 말이다. 즉, 최적화된 데이터는 경험에서 나오는 것이다.

날로 먹으려 들지 마라..ㅋㅋ)

**Test Mode 설명**

ALWAYS : 모든 픽셀이 프레임버퍼로 알파블렌딩 된다.

LESS , LESSEQUAL : 알파값이 임계값보다 작거나 같으면 블렌딩 된다

GREATER, GREATEREQUAL : 알파값이 임계값보다 크거나 같으면 블렌딩 된다.

EQUAL, NOTEQUAL : 같거나 같지않으면 블렌딩된다.

1. Vertex Colors 와 Apply Mode에 대한 이해

게임브리오의 해당 부분 도움말을 **반드시** 보도록 합시다. 올바른 이해를 하고 사용하는 게 좋겠다.

* 정적라이트/동적라이트 상관없을 때 지정 (예 : 이펙트, 아이템의 효과등)



* 버텍스칼라는 상관없이 동적라이팅만 사용할 때



* 정적라이트와 동적라이팅을 사용할 때



1. Alpha Testing

알파테스팅을 On 시키면 해당 재질 오브젝트는 알파검사를 반드시 하게 된다. 즉, Transparency Modes 모드에서 None을 지정했더라도 알파검사를 하게 되므로 주의하여야 한다.



알파테스팅을 하는 경우 알파값이 있는 텍스처를 사용해야 한다.

* 알파기능에 대해 다시 한번 더 설명
* 투과되지 않는 오브젝트는 알파값이 없는 텍스처를 지정한다.
* 알파블렌딩보다 알파테스팅이 더 빠르므로 알파테스팅만으로 처리되는 오브젝트는 알파테스팅만으로 처리한다.
* 알파블렌딩만 하는 것보다 알파테스팅도 같이 하는 게 더 빠르다. 왜냐하면..

설명하기 귀찮아 관련링크를 걸어둔다.

알파블렌딩과 알파테스팅의 차이

( <http://www.gpgstudy.com/forum/viewtopic.php?topic=15821> )

1. **데이터 Clone**

게임브리오에서는 맥스의 Clone방식을 지원한다. Copy는 동일한 오브젝트를 새로

만드는것이고 Instance는 데이터를 공유해서 오브젝트를 생성하는 것이다.

즉, Instance로 데이터를 클론해야 한다. 다만, 클론시킨 오브젝트라 할지라도 피봇을 변경하거나 스케일을 조절하면 게임브리오에서는 해당오브젝트를 새로운 오브젝트로 인식하므로 주의해야 한다.



1. **Gamebryo의 이해와 한계**
* Non-uniform scale을 지원하지 않는다

(**x, y, z축으로 동일한 값의 스케일만 지원**한다)

* 게임브리오 데이터추출옵션에 따라 최적화와 node tree가 달라진다

(Scene Graph라는 용어를 이해해야 한다)

* Physique보다는 **Skin의 사용을 권장**한다.

다만, Skin사용시 morphing을 주의해야 한다. (morphing을 쓰지 않으면 상관없다)

* 보다 많은 컬링을 위해 오브젝트를 너무 조각화시키는 것은 좋지 않다.

이런 분리로 인해 개별 폴리곤의 렌더링에 필요한 시간이 줄어들긴 하지만 Gamebryo가 개별 메쉬 마다 처리해야 하는 고정된 작업량을 변화시키진 않기 때문에 오히려 병목현상이 발생할수 있으며 그래픽 카드의 렌더링 성능을 완벽히 사용하지 못할 수도 있다. 그러므로 폴리곤이 대략 1000개이하(시스템마다 틀리다므로 딴지 걸지 말것)인 오브젝트는 분리하지 않는게 좋다.

알아두면 좋은 참조내용

1GHz CPU에서는 초당 30fps을 유지하려면 1frame에 오브젝트 개수를 1000개이하로 유지해야 한다. (하나의 오브젝트가 구성되는 폴리곤수가 적정수준일 경우)

* 한 버텍스당 최대 지정할 수 있는 본의 개수는 대부분의 하드웨어에서 **4개**이다.
1. **데이터 추출 옵션**

추출하고자 하는 대상에 따라 옵션이 달라지므로 이를 신경써야 한다. 또한, Process Script 의 설정값에 따라 추출되는 데이터에 대한 최적화 및 노드트리구조가 달라지므로 추출후 반드시 AssetViewer로 데이터를 확인하는 습관을 가지는 것이 좋다.

일단은 Common한 옵션은 다음과 같다.



 Textures는 NIF에 삽입하는 것보다는 원본 텍스처화일을 그대로 사용(Original images)

하는 게 추후 데이터변경시에도 용이하므로 위와 같이 지정하는 게 좋다. Platform은

멀티플랫폼용 게임을 만드는 것이 아니라면 DX 9으로 지정하는 게 좋다.

Lights 체크 버튼을 On 해 놓은 경우 맥스에서 라이트를 생성시키지 않았더라도

디폴트라이트 2개가 생성되므로 Light가 없는 맥스데이타의 경우 체크 버튼을 Off해주는

것이 최적화된 데이터추출에 좋다.

Process Script와 File Export Script는 추출하고자 하는 대상에 따라 변동의 폭이 크므로

프로그래머가 별도로 제공하도록 한다.

**마지막으로 제발 맥스에서 된다고 게임브리오에서 다 될거라 생각하지 마시고 Gamebryo의 도움말중 Artist’s Guides는 틈틈히 읽어보고 사용하고자 하는 맥스의 기능이 Gamebryo에서 지원하는지 확인하고 데이터를 AssetViewer로 보고 node tree 구성이 제대로 되어 있는 체크하는 습관을 가지도록 합시다.**