언리얼 페스트 2024 서울

분산 렌더링 시스템

고화질 VR 시네마틱 렌더링을 압도적으로 빠르게!

김범진
Senior Graphics Programmer
AmazeVR

목차

AMAZEVR

패널

김범진 (Jin Kim)
Senior Graphics Programmer
AmazeVR

Work Experience

Senior Graphics Programmer at AmazeVR	2022년 4월 - 현재
Senior Graphics Programmer at The Forge Interactive Inc.	2018년 6월 - 2022년 3월
Technical Artist at Bluehole Inc.	2013년 1월 - 2015년 2월

Education

University of Pennsylvania	2016년 8월 - 2018년 5월
Master of Computer Graphics and Game Technology	
Yonsei University	2006년 5월 - 2013년 7월
Bachelor of Science in Computer Science and Engineering	

Email

jin@amazevr.com

kimchistorm.3603@gmail.com

VR 시네마틱 렌더링 개요

101 of VR Cinematic Rendering

AMAZEVR

Virtual Reality

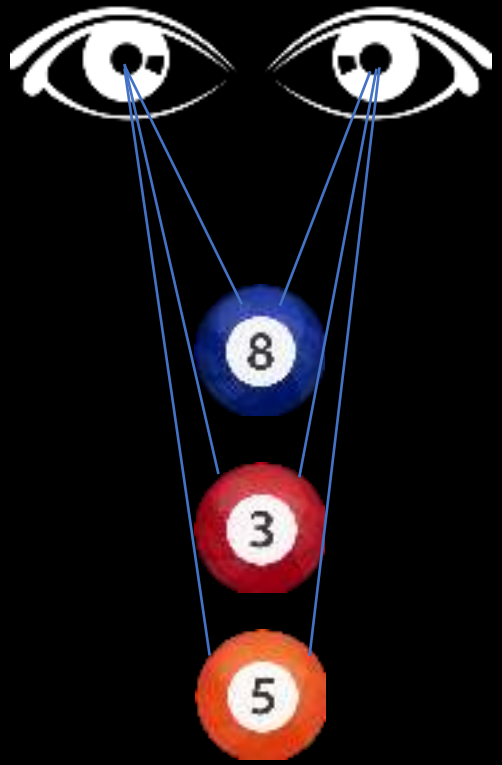
가상현실은 단순히 가상의 공간을 구현하는 것을 넘어서서 사용자의 오감에
직접적으로 작용하여 실제에 근접한 공간적, 시간적인 체험을 가능케 하는 기술을 의미한다.

[출처 - 나무위키]

실제에 근접한 공간적 체험

입체감 두 눈!

AMAZEVR



[출처 - Advanced VR Rendering Performance GDC 2016]

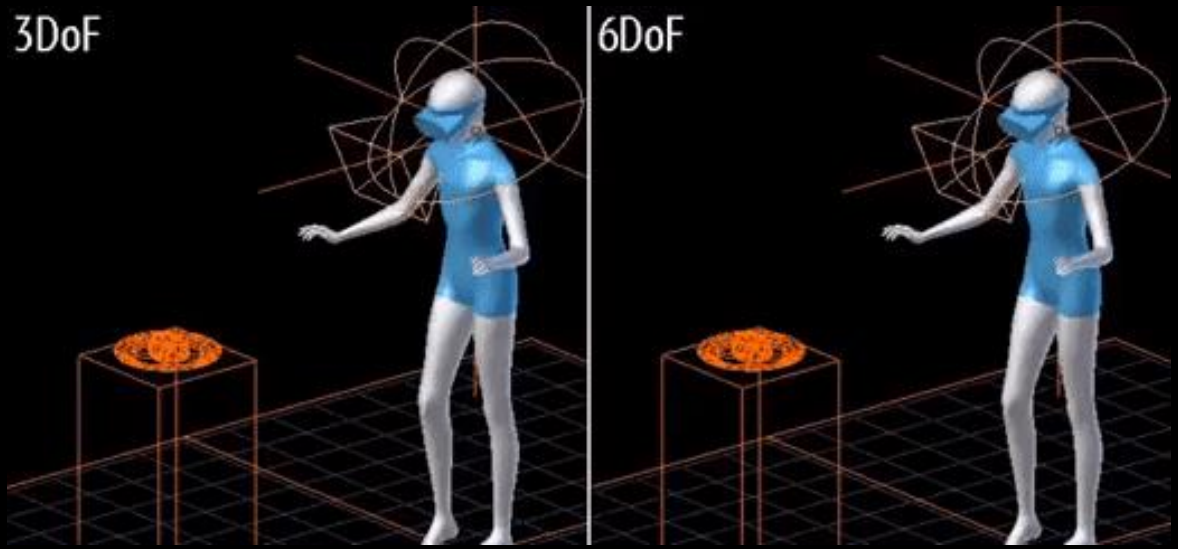
Virtual Reality Devices

AMAZEVR

VR Video (180° or 360°)

Real-time

AMAZEVR



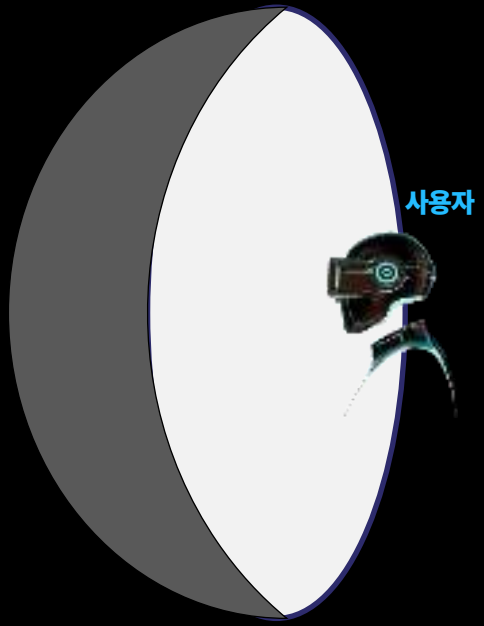
3DoF

사용자는 회전만 가능
사용자는 위치를 변경할 수 없음
사용자가 움직여도 월드는 그대로

6DoF

사용자는 회전과 더불어 위치를 변경할 수 있음
우리가 실제 존재하는 세상과 비슷

AMAZEVR



측면

정면

등장방형도법 - Equirectangular Projection

[**출처** - Drawing Equirectangular VR Panoramas with Ruler, Compass, and Protractor]

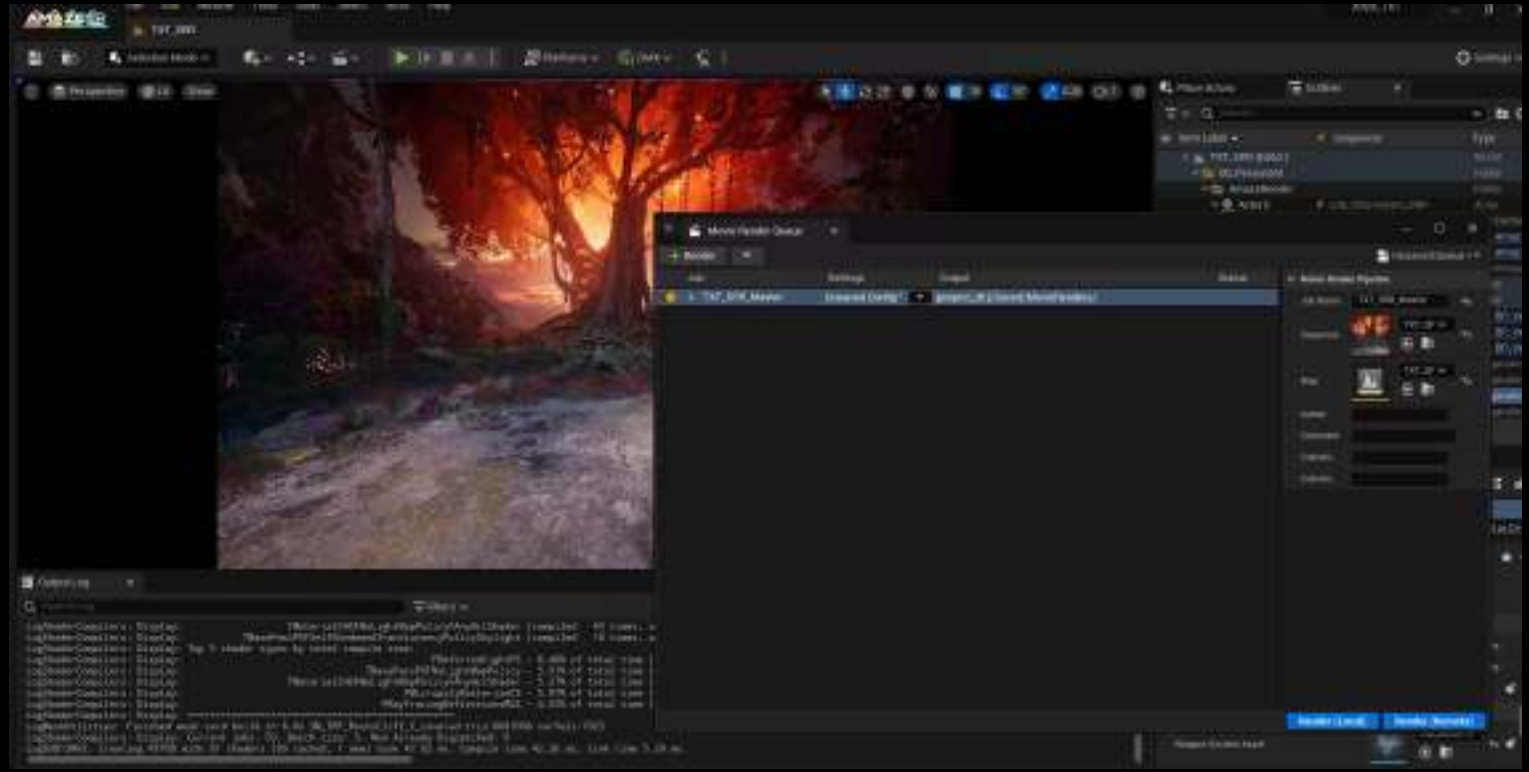
AMAZEVR

언리얼 엔진에서의 VR 시네마틱 렌더링

VR Cinematic Rendering in UE

AMAZEVR

MovieRenderQueue (MovieRenderPipeline)



AMAZEVR

MovieRenderQueue (MovieRenderPipeline)

레벨(Scene)

스태틱/다이나믹 오브젝트 배치, 라이팅, GI, 레이트레이싱

시퀀스

카메라, 라이팅, 이펙트, 애니메이션

Movie Render Queue

렌더링 옵션 세팅 후, 렌더링

AMAZEVR

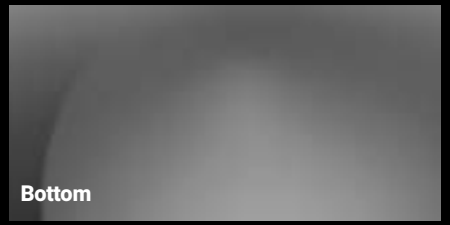
Top

Warping

Left

Center

Right

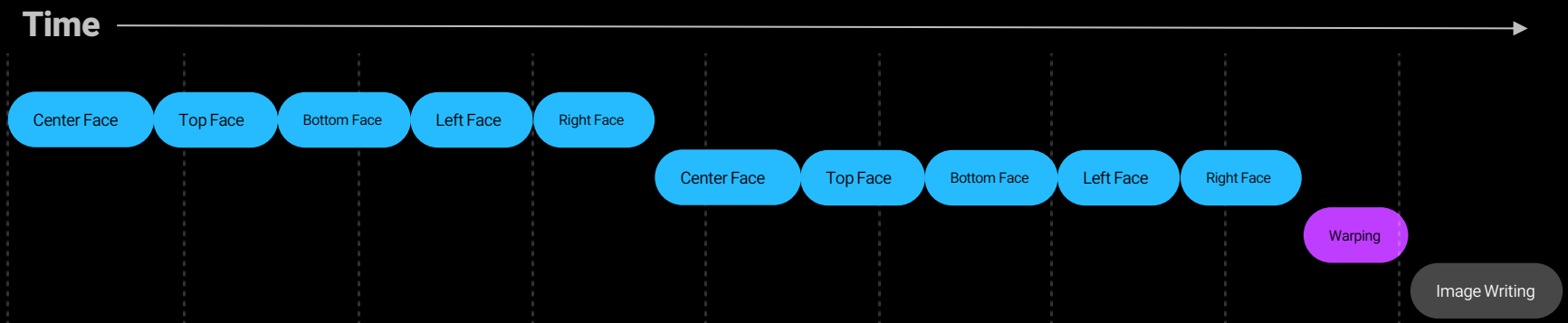


Bottom

Warped Image - 3840 x 3840

AMAZEVR

AmazeVR's Cinematic Rendering Pipeline



- Face 렌더링 정보를, 각각의 렌더타겟(RGBA16F)에 저장
- 모든 Face 렌더링 후, CustomRenderPasses를 이용하여 Warping
- Warping이 끝난 결과를 이미지로 쓰기

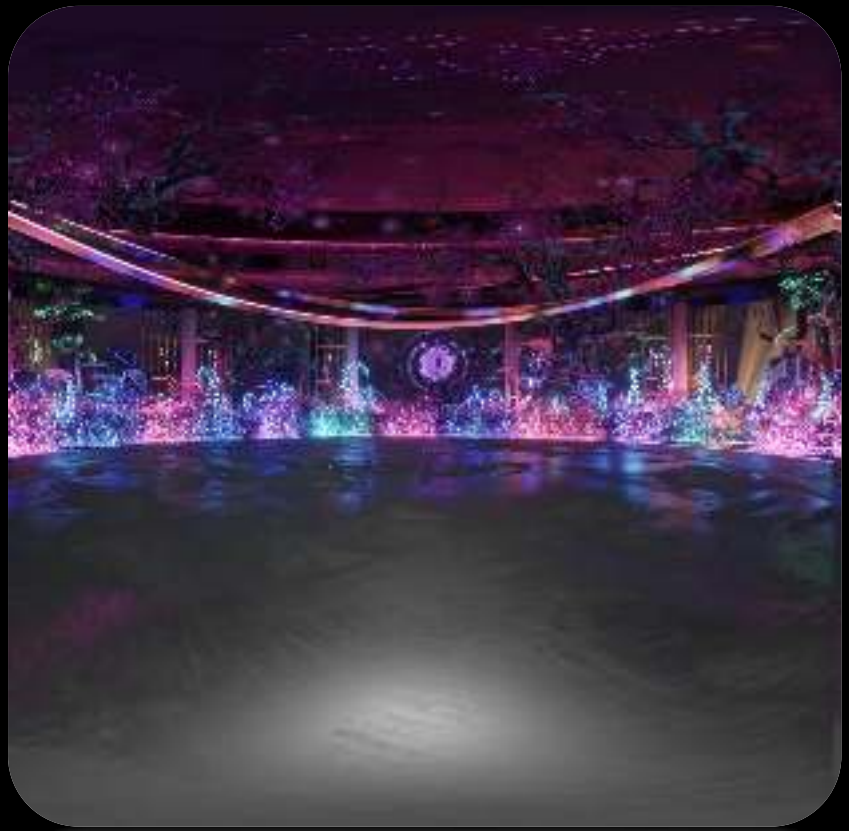
AMAZEVR

렌더링 해상도

- 최종 영상 해상도는 8K(7680x3840)
- Center 면 해상도는 3840x3840
- 나머지 면은 1920x3840
- 상당한 고해상도

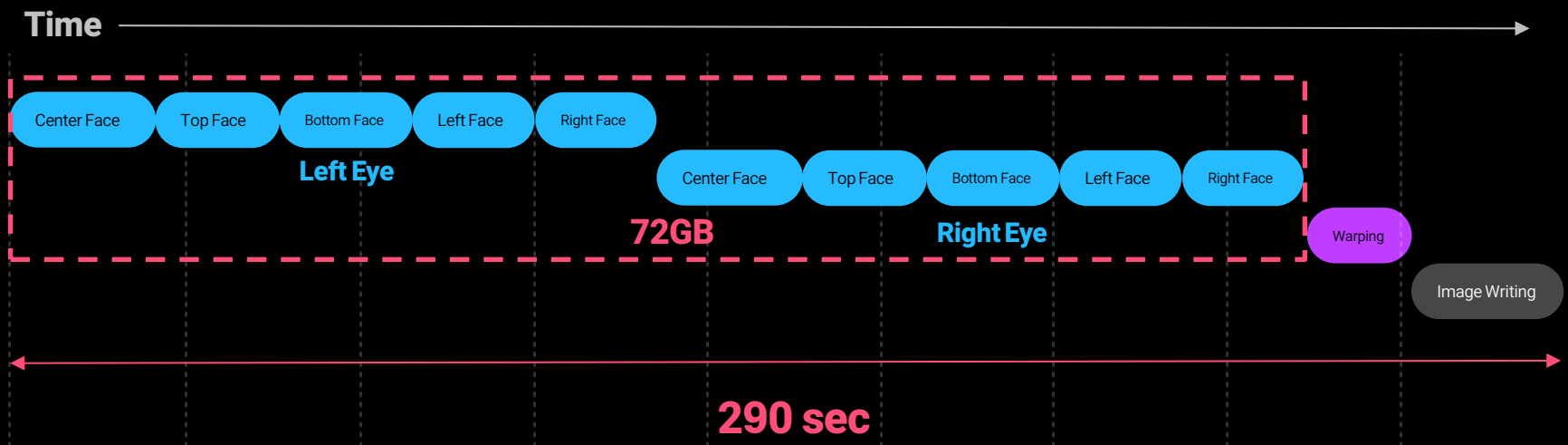
Case - Aespa : Black Mamba

AMAZEVR



Case - Aespa : Black Mamba

Rendered by RTX A6000(48GB)



- 1프레임 렌더링에 약 290초
- GPU 메모리 부족으로 인해 발생하는 Memory Page Up & Down 이 원인

AMAZEVR

렌더링 해상도를 줄인다

레벨의 오브젝트를 줄이고, 텍스쳐 해상도를 낮춘다

→ 퀄리티 저하

최적화를 그 만큼 할 수 없는 경우면?

한 번에 모두 그리지 말고,
각 면을 따로 분산시켜 그리면 어떨까?

어메이즈 분산 렌더링 시스템

Amaze Distribution Rendering System

AMAZEVR

어메이즈 분산 렌더링 시스템 (Amaze Distribution Rendering System)

각 면(Face)을 독립된 PC에서 렌더링

렌더링된 각 면 이미지들을 Warping

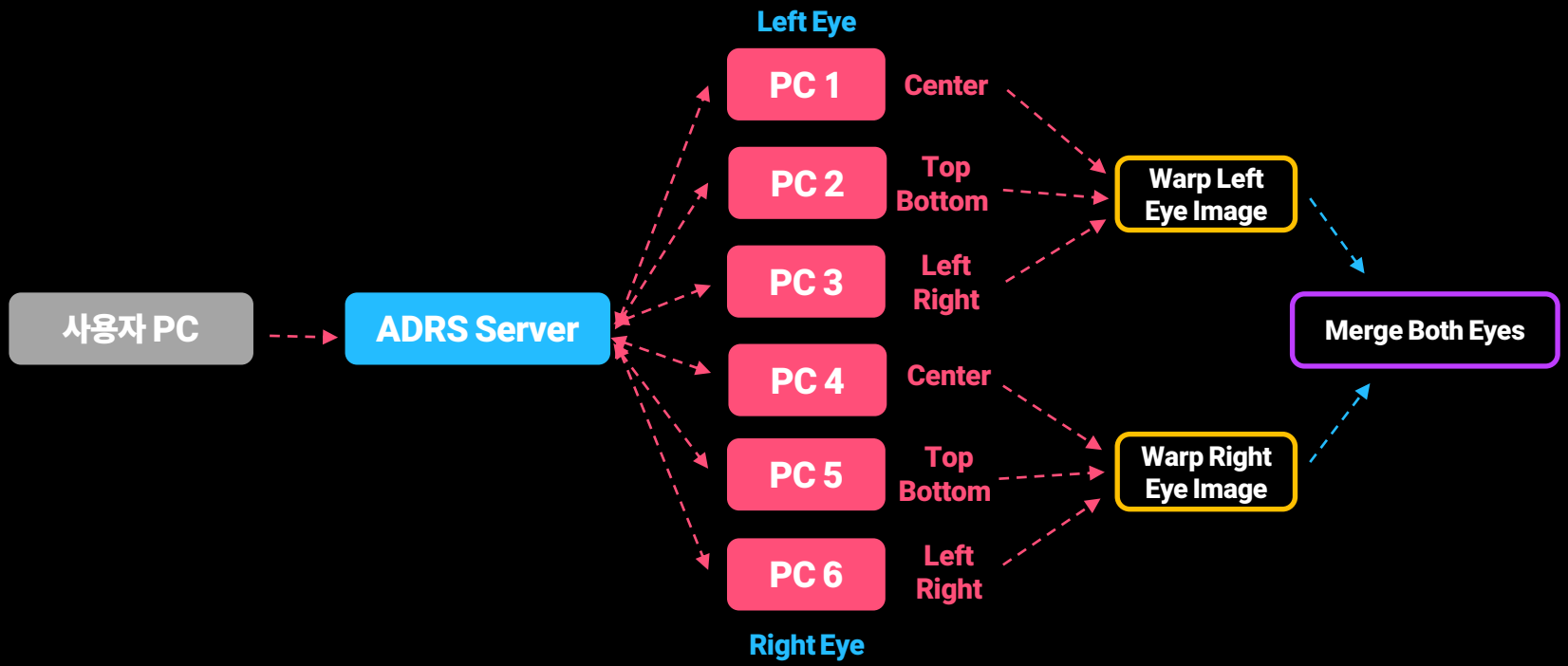
언리얼 엔진이 아닌 외부에서

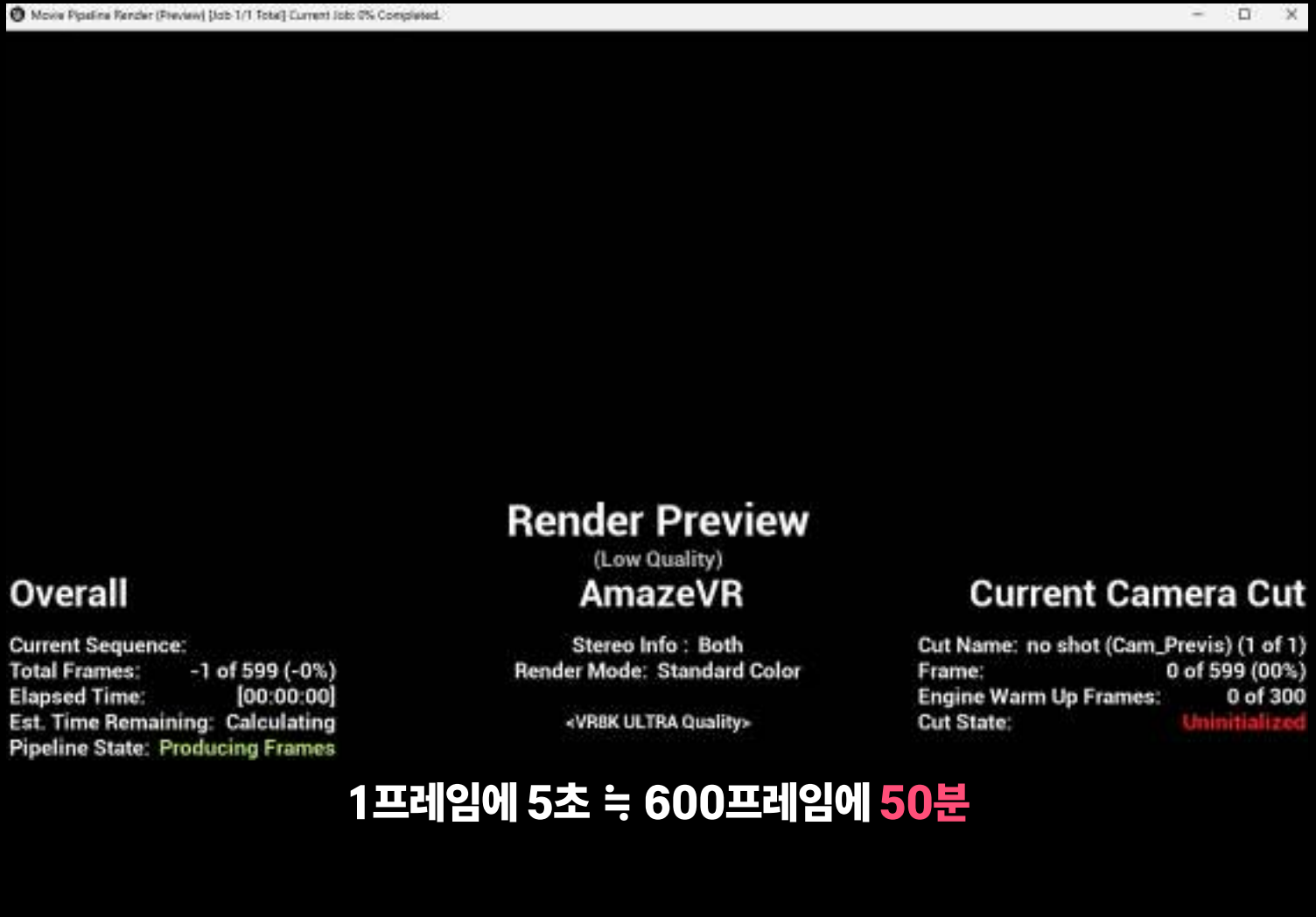
최종 이미지 Writing

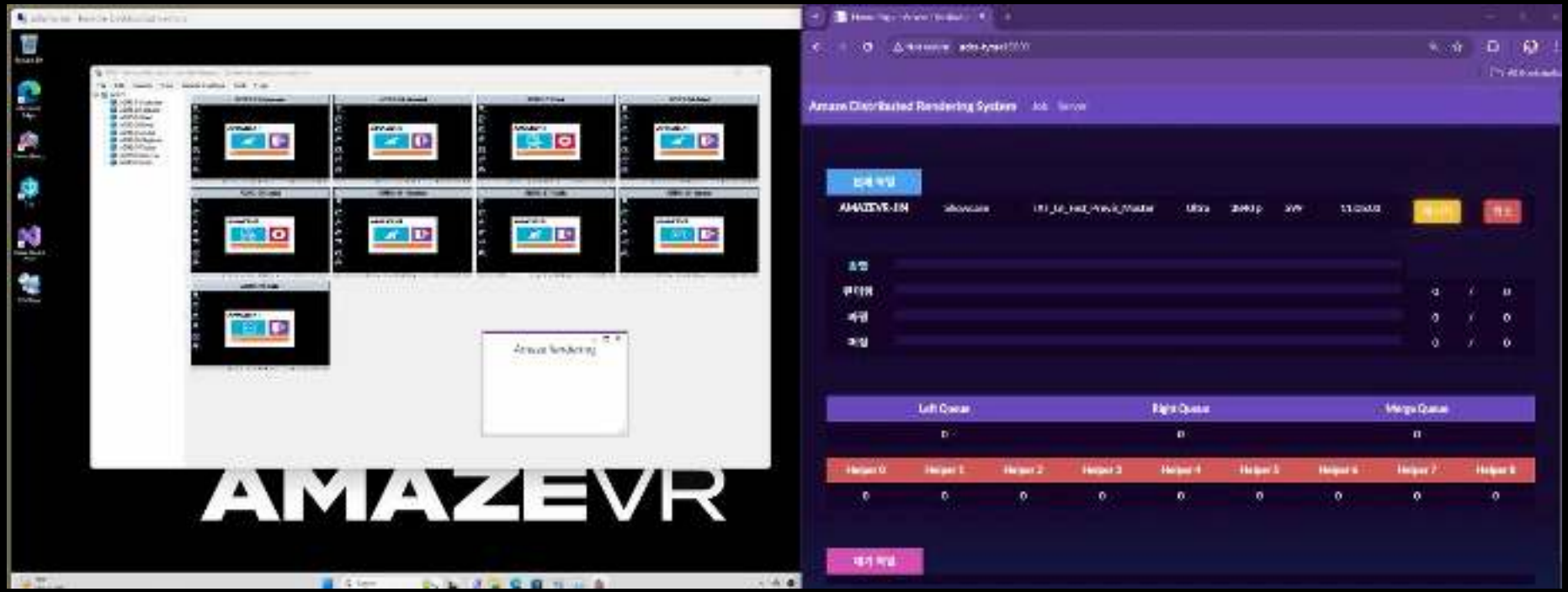
Warping 된 각 눈의 이미지를 합친 후, 최종 이미지 쓰기

AMAZEVR

어메이즈 분산 렌더링 시스템 (Amaze Distribution Rendering System)




Movie Pipeline Render (Preview) [Job 1/1 Total] Current Job: 0% Completed.
Render Preview
(Low Quality)
Overall
Current Sequence:
Total Frames: -1 of 599 (-0%)
Elapsed Time: [00:00:00]
Est. Time Remaining: Calculating
Pipeline State: Producing Frames
AmazeVR
Stereo Info : Both
Render Mode: Standard Color
<VR8K ULTRA Quality>
Current Camera Cut
Cut Name: no shot (Cam_Previs) (1 of 1)
Frame: 0 of 599 (00%)
Engine Warm Up Frames: 0 of 300
Cut State: Uninitialized
1프레임에 5초 ≒ 600프레임에 50분



600프레임에 단 2분 3초!

AMAZEVR

어메이즈 분산 렌더링 시스템 (Amaze Distribution Rendering System)

QueueManifest.utxt

각 면 렌더링을 위해 MovieRenderQueue를 렌더링을 담당할 PC에 전송

QueueManifest_Left_Center.utxt
QueueManifest_Left_Top.utxt
QueueManifest_Left_Bottom.utxt
.
.
.
QueueManifest_Right_Right.utxt

각 QueueManifest에는 고유의 카메라 회전값과 ViewSize를 갖고 있음

AMAZEVR

MoviePipelineQueueEngineSubsystem.cpp

```
void UMoviePipelineQueueEngineSubsystem::RenderQueueWithExecutorInstance(UMoviePipelineExecutorBase* InExecutor)
{
#if WITH_AMAZEVR
  if (IAmazeModule* AmazeModule = static_cast<IAmazeModule*>(FModuleManager::Get().LoadModule("Amaze")))
  {
    AmazeModule->SetRenderExecutorInformation(InExecutor);

#if SPLIT_RENDERING
    UMoviePipelineQueue* pCurrentQueue = GetQueue();
    UMoviePipelineExecutorJob* FirstJob = (pCurrentQueue->GetJobs())[0];

    uint8 RenderMode = AmazeModule->IsSplitRendering(FirstJob->GetConfiguration());

```

AMAZEVR

MoviePipelineQueueEngineSubsystem.cpp

```
170
171         AmazeModule->SetStereoCameraInfo(AdditionalJobForLeftCenter->GetConfiguration(), (uint8)1 /*eMovieRenderAmazeVRStereoCa
172         AmazeModule->SetFaceToRenderInfo(AdditionalJobForLeftCenter->GetConfiguration(), (uint8)5 /*AmazeVRCubeFace_MAX*/);
173         AmazeModule->SetMainFaceForEyeAdaptation(AdditionalJobForLeftCenter->GetConfiguration(), (uint8)0 /*AmazeVRCubeFace_
174         AmazeModule->SetOutputResolutionDivision(AdditionalJobForLeftCenter->GetConfiguration(), FIntPoint(1, 1));
175         AmazeModule->SetSplitRendering(AdditionalJobForLeftCenter->GetConfiguration(), true);
176         AmazeModule->ExpandFileName(AdditionalJobForLeftCenter->GetConfiguration(), FString(TEXT(".Left_Center")));
177
178         AmazeModule->SetStereoCameraInfo(AdditionalJobForLeftTop->GetConfiguration(), (uint8)1 /*eMovieRenderAmazeVRStereoCa
179         AmazeModule->SetFaceToRenderInfo(AdditionalJobForLeftTop->GetConfiguration(), (uint8)5 /*AmazeVRCubeFace_MAX*/);
180         AmazeModule->SetMainFaceForEyeAdaptation(AdditionalJobForLeftTop->GetConfiguration(), (uint8)1 /*AmazeVRCubeFace_PX*
181         AmazeModule->SetOutputResolutionDivision(AdditionalJobForLeftTop->GetConfiguration(), FIntPoint(2, 1));
182         AmazeModule->SetSplitRendering(AdditionalJobForLeftTop->GetConfiguration(), true);
183         AmazeModule->ExpandFileName(AdditionalJobForLeftTop->GetConfiguration(), FString(TEXT(".Left_Top")));
184
185         AmazeModule->SetStereoCameraInfo(AdditionalJobForLeftBottom->GetConfiguration(), (uint8)1 /*eMovieRenderAmazeVRStereoCa
186         AmazeModule->SetFaceToRenderInfo(AdditionalJobForLeftBottom->GetConfiguration(), (uint8)5 /*AmazeVRCubeFace_MAX*/);
187         AmazeModule->SetMainFaceForEyeAdaptation(AdditionalJobForLeftBottom->GetConfiguration(), (uint8)2 /*AmazeVRCubeFace_
188         AmazeModule->SetOutputResolutionDivision(AdditionalJobForLeftBottom->GetConfiguration(), FIntPoint(2, 1));
189         AmazeModule->SetSplitRendering(AdditionalJobForLeftBottom->GetConfiguration(), true);
190         AmazeModule->ExpandFileName(AdditionalJobForLeftBottom->GetConfiguration(), FString(TEXT(".Left_Bottom")));
191
192         AmazeModule->SetStereoCameraInfo(AdditionalJobForLeftLeft->GetConfiguration(), (uint8)1 /*EMovieRenderAmazeVRStereoC
193         AmazeModule->SetFaceToRenderInfo(AdditionalJobForLeftLeft->GetConfiguration(), (uint8)5 /*AmazeVRCubeFace_MAX*/);
194         AmazeModule->SetMainFaceForEyeAdaptation(AdditionalJobForLeftLeft->GetConfiguration(), (uint8)4 /*AmazeVRCubeFace_NX*
195         AmazeModule->SetOutputResolutionDivision(AdditionalJobForLeftLeft->GetConfiguration(), FIntPoint(2, 1));
196         AmazeModule->SetSplitRendering(AdditionalJobForLeftLeft->GetConfiguration(), true);
197         AmazeModule->ExpandFileName(AdditionalJobForLeftLeft->GetConfiguration(), FString(TEXT(".Left_Left")));
198
```

AMAZEVR

MoviePipeline 수정

MoviePipelineLinearExecutor

MoviePipelineNewProcessExecutor

MoviePipelineQueueEngineSubSystem

MoviePipelineRendering

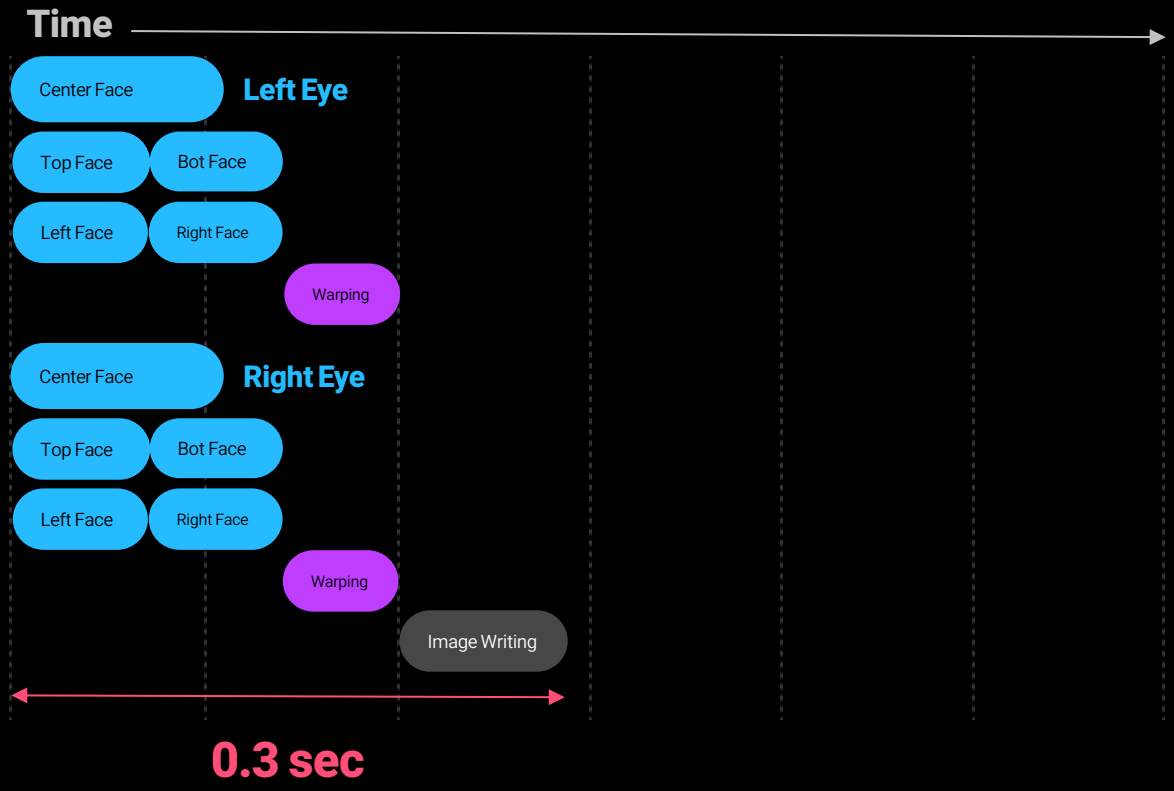
MoviePipelineSurfaceReader

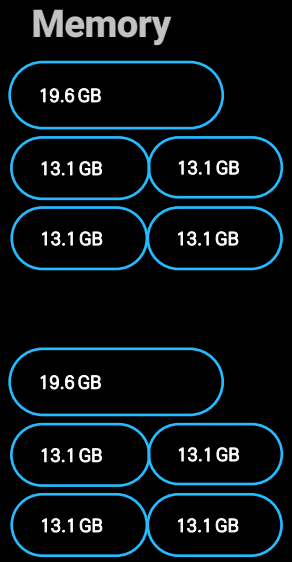
MoviePipelineCommandLine

AMAZEVR

Case - Aespa : Black Mamba
3090(24GB) x6

Rendered by RTX

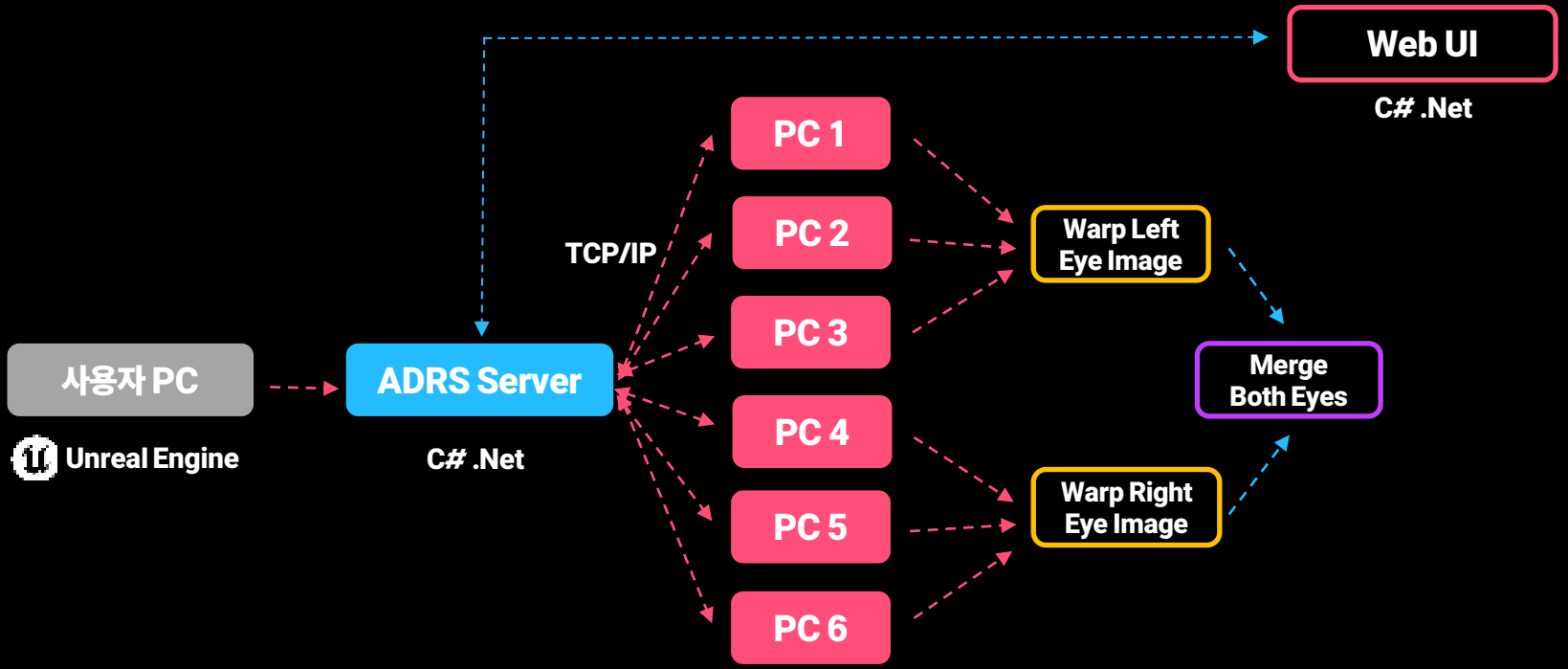




GPU 메모리 부족이 해결되어 정상적인 속도로 렌더링!

AMAZEVR

어메이즈 분산 렌더링 시스템 (Amaze Distribution Rendering System)



C# .net apps which call Unreal Engine for Rendering

AMAZEVR

어메이즈 분산 렌더링 시스템 (Amaze Distribution Rendering System)

GPU 메모리 부족으로 인한 성능 저하 해결

하나의 PC보다 훨씬 빠른 렌더링 속도

6개의 PC로 구성한 경우 약 3배~6배 속도 향상

결정론

Determinism

인게임 - CPU Side

AMAZEVR

Random Generator

GameTime

Global Frame Count

```
1 FMath::RandInit(0);
2 FMath::SRandInit(0);
3
4 // Reset
5 FApp::SetGameTime(0.0);
6 FApp::InvalidateCurrentFrameTime();
7
8 GFrameCounter = 0;
```

비쥬얼 이펙트 - VFX

AMAZEVR

Determinism or Deterministic이 포함된 옵션 모두 활성화

Random/Uniform Range 자료형 모듈의 Fixed Random Seed 활성화

Float, Int, Vector...

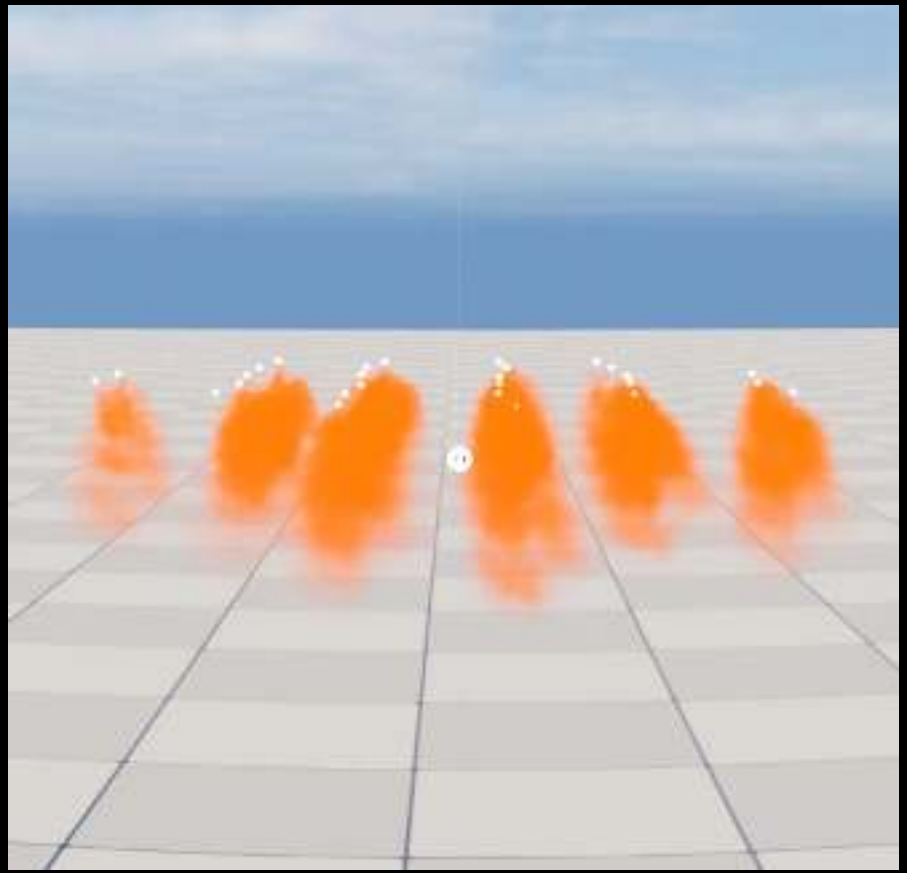
Noise Quality가 Bake(Low, Mid, High)일 경우, Evaluated(Ultra) 사용

Collision 모듈이 GPU를 사용할 경우

Deterministic한 Type을 사용, 모두 실패하면 CPU 연산으로 변경

비쥬얼 이펙트 - Sample Particles from Other Emitter 모듈 문제

AMAZEVR



비쥬얼 이펙트 - Sample Particles from Other Emitter

AMAZEVR

언리얼 엔진 내 옵션 변경만으로는 **불가능**

부모 Emitter에서 살아남는 Particle 갯수를 파악하기 위해 **atomic** 연산을 사용

Thread 실행 순서를 알 수 없음 > 완전히 **무작위** > 살아남는 파티클의 ID가 무작위 순으로 정렬되어 다음 Emitter에게 전달 됨

부모 Emitter에서 살아 남는 Particle를 정렬하자!

비쥬얼 이펙트 - Sample Particles from Other Emitter

AMAZEVR

SortByIDs 옵션 추가 - 활성화시 UniqueID를 이용해 파티클 정렬

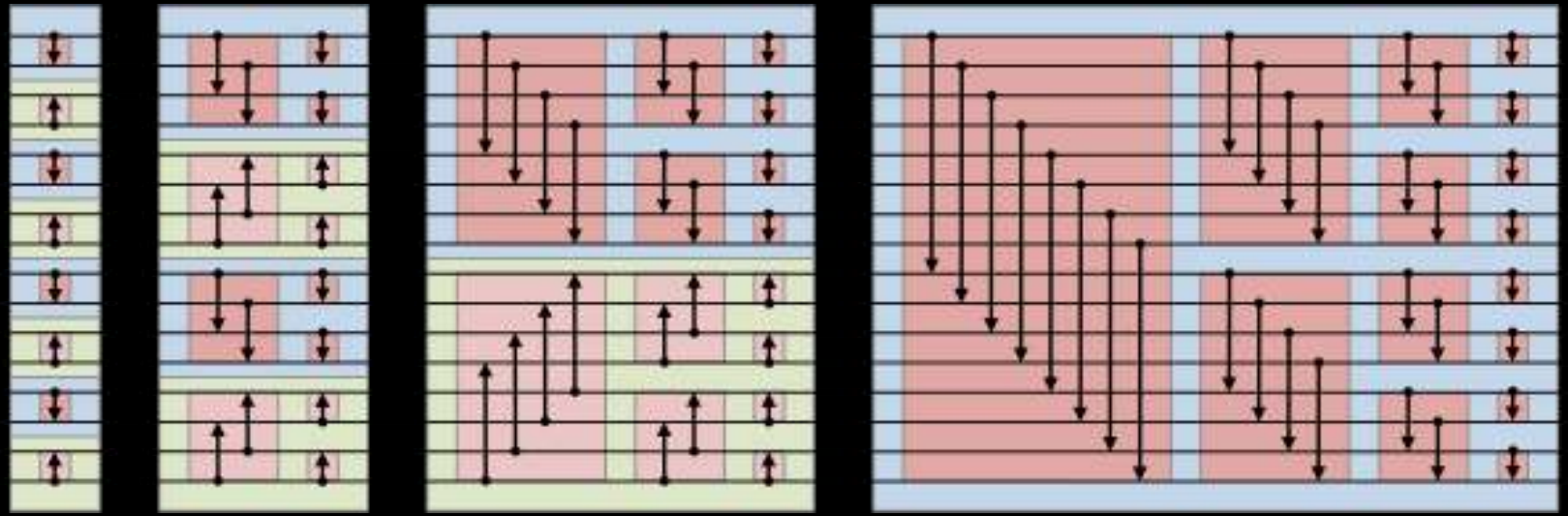
파티클 정렬을 위해 BitonicMergeSort를 수행하는 Compute Shader 추가

Niagara에서 정렬한 Index 배열을 이용할 RWBuffer 추가

NiagaraHlslTranslator.cpp

비쥬얼 이펙트 - Bitonic Merge Sort

AMAZEVR



GPU를 활용하기 가장 쉽고 퍼포먼스가 좋다고 판단

병렬적으로 모든 원소들에게 정렬을 수행

[출처 - https://en.wikipedia.org/wiki/Bitonic_sorter]

비쥬얼 이펙트 - Bitonic Merge Sort for Niagara

AMAZEVR

```
1  /*==============================================================================
2  NiagaraBitonicMergeSort.usf:
3  ==============================================================================*/
4  #pragma warning(disable:4008)
5  #include "/Engine/Public/Platform.ush"
6
7  // #include "NiagaraShaderVersion.ush"
8
9  #include "/Engine/Private/Common.ush"
10 #include "/Engine/Private/Definitions.usf"
11
12 #if FORCE_TO_BE_DETERMINISTIC
13 uint ReadInstanceCountOffset;
14 uint WriteInstanceCountOffset;
15
16 RWBuffer<uint> RWInstanceCounts;
17 RWBuffer<int> RWOriginalIndexTable;
18
19 RWBuffer<int> RWIndexTableForSorting;
20
21 RWBuffer<int> RWIndexToSortedIndexTable;
22 RWBuffer<int> RWCounterBuffer;
23 RWBuffer<int> RWOriginalIDToIndexTable;
```

비쥬얼 이펙트 - Bitonic Merge Sort for Niagara

AMAZEVR

```
25 bool IsPowerOfTwo(int number)
26 {
27     // Check if the number is greater than 0 and has only one bit set to 1
28     return number > 0 && (number & (number - 1)) == 0;
29 }
30
31 int GetLeastGreaterPowerOfTwo(int number)
32 {
33     number--;  // Decrement the number by 1
34
35     // Perform bitwise OR operation to set all bits to the right of the highest bit
36     number |= number >> 1;
37     number |= number >> 2;
38     number |= number >> 4;
39     number |= number >> 8;
40     number |= number >> 16;
41
42     number++;  // Increment the number by 1 to get the next power of 2
43
44     return number;
45 }
```

비쥬얼 이펙트 - Bitonic Merge Sort for Niagara

AMAZEVR

```
[numthreads(1, 1, 1)]
void InitCounterBuffersCS(uint GId : SV_GroupID, uint GTId : SV_GroupThreadID, uint DTId : SV_DispatchThreadID)
{
  RWCounterBuffer[0] = 2147483647;
  RWCounterBuffer[1] = 0;
  RWCounterBuffer[2] = 0;
  RWCounterBuffer[3] = 0;
}

#define SORTING_THREAD_COUNT 1024

[numthreads(SORTING_THREAD_COUNT, 1, 1)]
void InitSortingBuffersCS(uint GId : SV_GroupID, uint GTId : SV_GroupThreadID, uint DTId : SV_DispatchThreadID)
{
  uint ThreadID = DTid;

  //TODO: Assertion
  if (ThreadID >= 262144)
  {
    return;
  }

  int RetPrevIdx;
  InterlockedAdd(RWCounterBuffer[0], -1, RetPrevIdx);

  RWOriginalIndexTable[ThreadID] = RetPrevIdx;
  RWIndexTableForSorting[ThreadID] = RetPrevIdx;
  RWIndexToSortedIndexTable[ThreadID] = -1;
  RWOriginalIDToIndexTable[ThreadID] = -1;
}
```

비쥬얼 이펙트 - Bitonic Merge Sort for Niagara

AMAZEVR

```
78 groupshared uint ParticleCount;
79
80 [numthreads(SORTING_THREAD_COUNT, 1, 1)]
81 void BitonicMergeSortCS(uint Gid : SV_GroupID, uint GTid : SV_GroupThreadID, uint DTid : SV_DispatchThreadID)
82 {
83   if (GTid == 0)
84   {
85     ParticleCount = RWInstanceCounts[WriteInstanceCountOffset];
86   }
87
88   GroupMemoryBarrierWithGroupSync();
89
90   int SequenceLength = 0;
91
92   if (!IsPowerOfTwo((int)ParticleCount))
93   {
94     SequenceLength = GetLeastGreaterPowerOfTwo((int)ParticleCount);
95   }
96   else
97   {
98     SequenceLength = (int)ParticleCount;
99   }
100
101   DeviceMemoryBarrierWithGroupSync();
102
103   int CBS = 2;
104
105   uint ThreadID = DTid;
```

비쥬얼 이펙트 - Bitonic Merge Sort for Niagara

AMAZEVR

```
while (Cd > 0)
{
  for (int LoopIndex = 0; LoopIndex < ((SequenceLength + SORTING_THREAD_COUNT - 1) / SORTING_THREAD_COUNT); ++LoopIndex)
  {
    int CurrentThreadID = (int)ThreadID + LoopIndex * SORTING_THREAD_COUNT;

    int BitonicIndex = (CurrentThreadID * 2) / CKS;
    bool IsAscending = (BitonicIndex % 2 == 0);

    int BaseIndex = (CurrentThreadID / Cd) * (2 * Cd);

    int LocalIndex = CurrentThreadID % Cd;
    int IndexA = BaseIndex + LocalIndex;
    int IndexB = IndexA + Cd;


    if (IsAscending)
    {
      if (RWIndexTableForSorting[IndexA] > RWIndexTableForSorting[IndexB])
      {
        int temp = RWIndexTableForSorting[IndexA];
        RWIndexTableForSorting[IndexA] = RWIndexTableForSorting[IndexB];
        RWIndexTableForSorting[IndexB] = temp;
      }
    }
    else
    {
      if (RWIndexTableForSorting[IndexA] < RWIndexTableForSorting[IndexB])
      {
        int temp = RWIndexTableForSorting[IndexA];
        RWIndexTableForSorting[IndexA] = RWIndexTableForSorting[IndexB];
        RWIndexTableForSorting[IndexB] = temp;
      }
    }
  }

  Cd = Cd / 2;
```

비쥬얼 이펙트 - Bitonic Merge Sort for Niagara

AMAZEVR

```
149         // Barrier
150         //if (CS < SORTING_THREAD_COUNT)
151         {
152           DeviceMemoryBarrierWithGroupSync();
153         }
154       }
155
156       CBS *= 2;
157     }
158
159     return;
160   }
```

비쥬얼 이펙트 - Sample Particles from Other Emitter

AMAZEVR

```
65      if (bRequiresPersistentIDs && !TranslationStages[i].bPartialParticleUpdate)
66      {
67  #if FORCE_TO_BE_DETERMINISTIC
68        if (bSortByIDs)
69        {
70          HlslOutput += FString::Printf(TEXT("\t\tUpdateIDForSort(0, bValid ? RsParticles.ID.Index :  1, WriteIndex);\n"), *
71        }
72        else
73  #endif // #if FORCE_TO_BE_DETERMINISTIC
74        HlslOutput += FString::Printf(TEXT("\t\tUpdateID(0, bValid ? %sParticles.ID.Index : -1, WriteIndex);\n"), *ContextNa
75      }
76
77      HlslOutput += TEXT("\t\tif (bValid)\n\t\t{\n");
78
79  #if FORCE_TO_BE_DETERMINISTIC
80      if (bRequiresPersistentIDs && !TranslationStages[i].bPartialParticleUpdate && bSortByIDs)
81      {
82        HlslOutput += TEXT("\t\t\tRWOriginalIndexTable[WriteIndex] = (int)") + ContextName + TEXT("Particles.UniqueID;\n");
83        HlslOutput += TEXT("\t\t\tRWIndexTableForSorting[WriteIndex] = (int)") + ContextName + TEXT("Particles.UniqueID;\n")
84      }
85  #endif // #if FORCE_TO_BE_DETERMINISTIC
86
87      for (int32 DataSetIndex = 0, IntCounter = 0, FloatCounter = 0, HalfCounter = 0; DataSetIndex < DataSetWrites.Num(); ++
88      {
89        const FNiagaraDataSetID DataSetID = ReadDataSetIDs[DataSetIndex];
90        const TArray<FNiagaraVariable>& NiagaraVariables = DataSetVariables[DataSetWrites[DataSetID]];
91        for (const FNiagaraVariable& Var : NiagaraVariables)
92        {
93          const bool bWriteToHLSL = !TranslationStages[i].bPartialParticleUpdate || TranslationStages[i].SetParticleAttribut
```

나이아가라 수정

AMAZEVR

NiagaraDataSet

NiagaraDataSetCompiledData

NiagaraEmitter

NiagaraEmitterInstance

NiagaraScript

NiagaraSystem

NiagaraGpuComputeDispatch

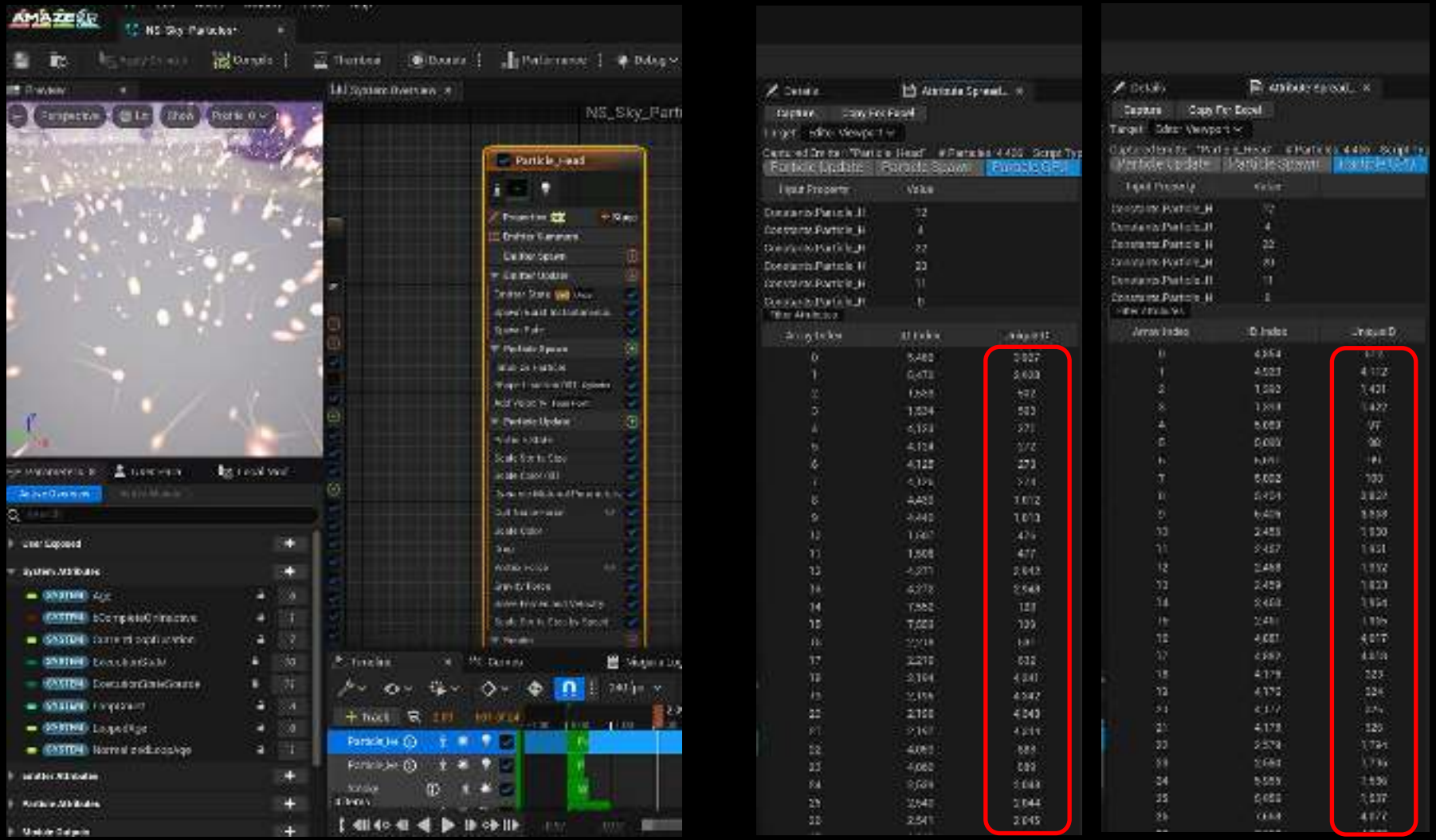
NiagaraRendererProperties

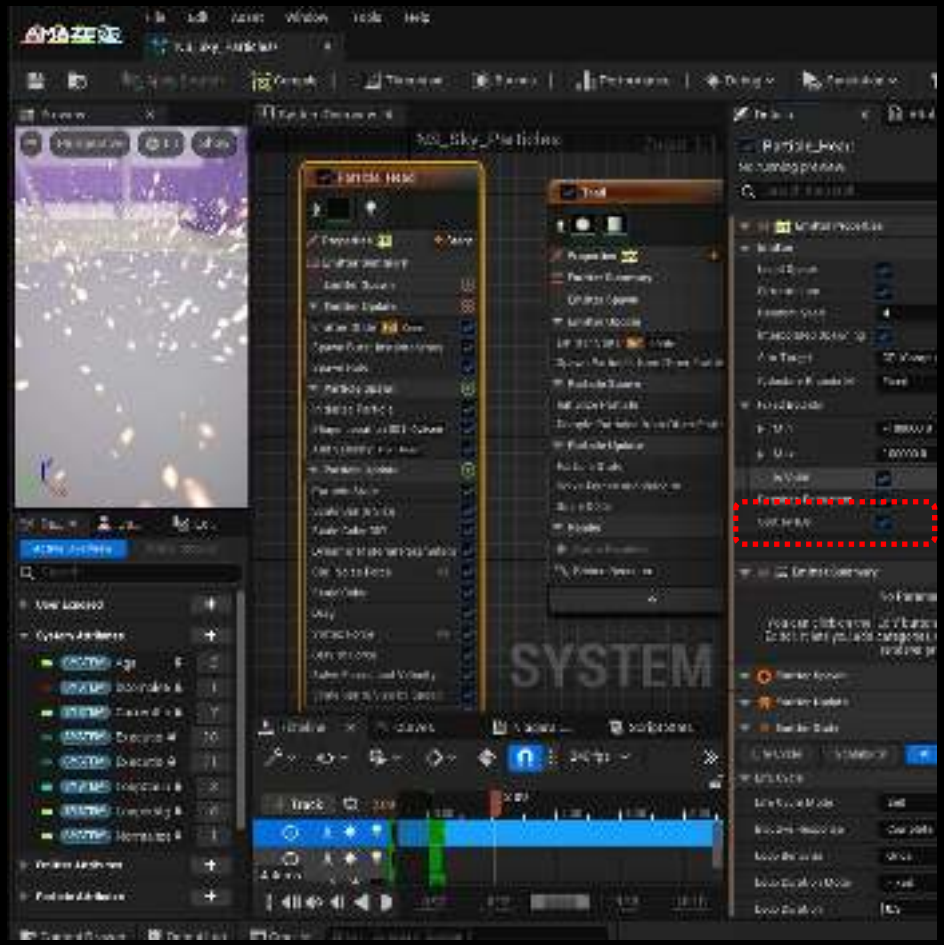
NiagaraHlslTranslator

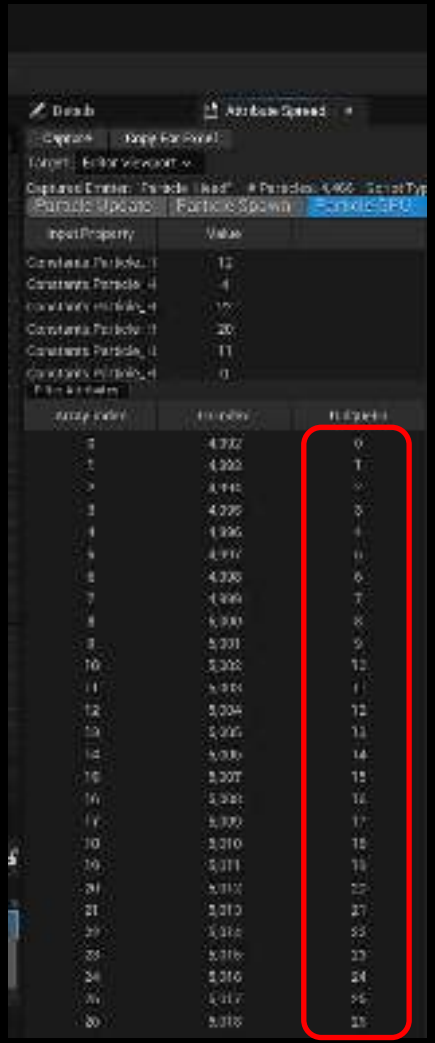
NiagaraShader

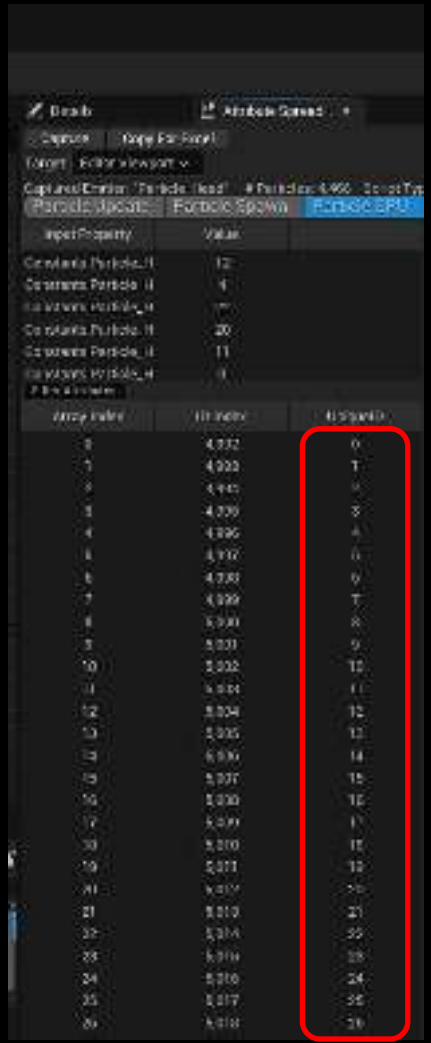
NiagaraEmitterInstanceShader

+ NiagaraBitonicMergeSortShader



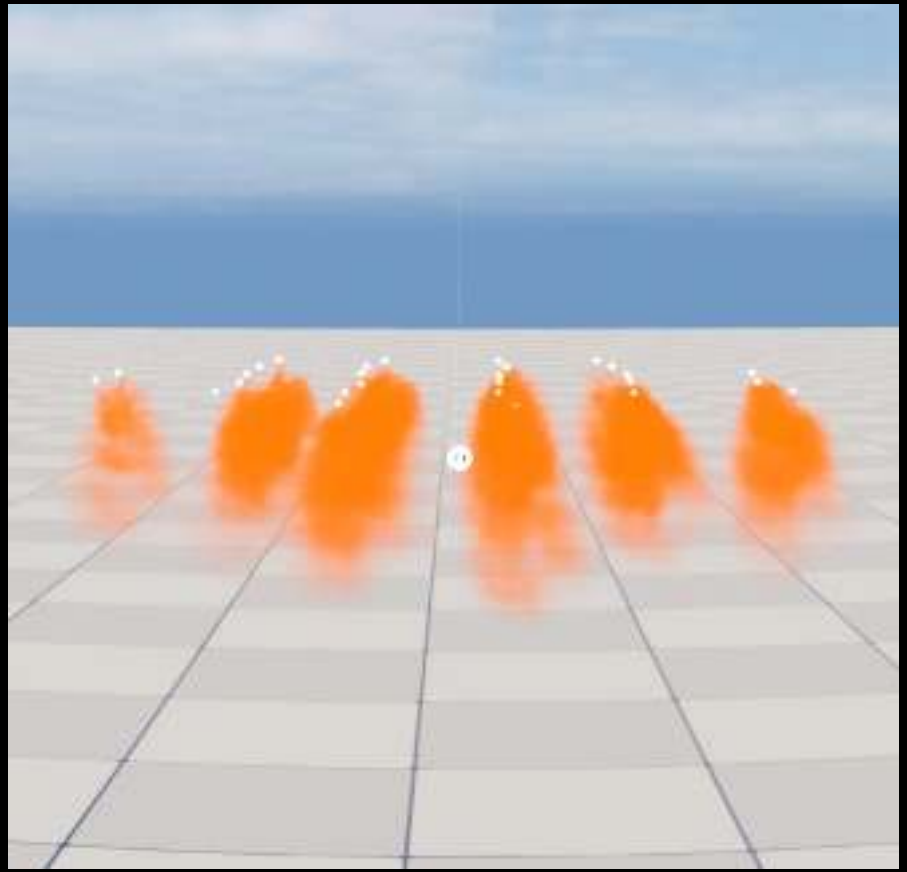

SYSTEM





비쥬얼 이펙트 - Sample Particles from Other Emitter 모듈 문제 해결 후

AMAZEVR



눈적응 - Eye Adaptaion (Exposure) 문제

AMAZEVR

눈적응 - Eye Adaptaion (Exposure)

AMAZEVR

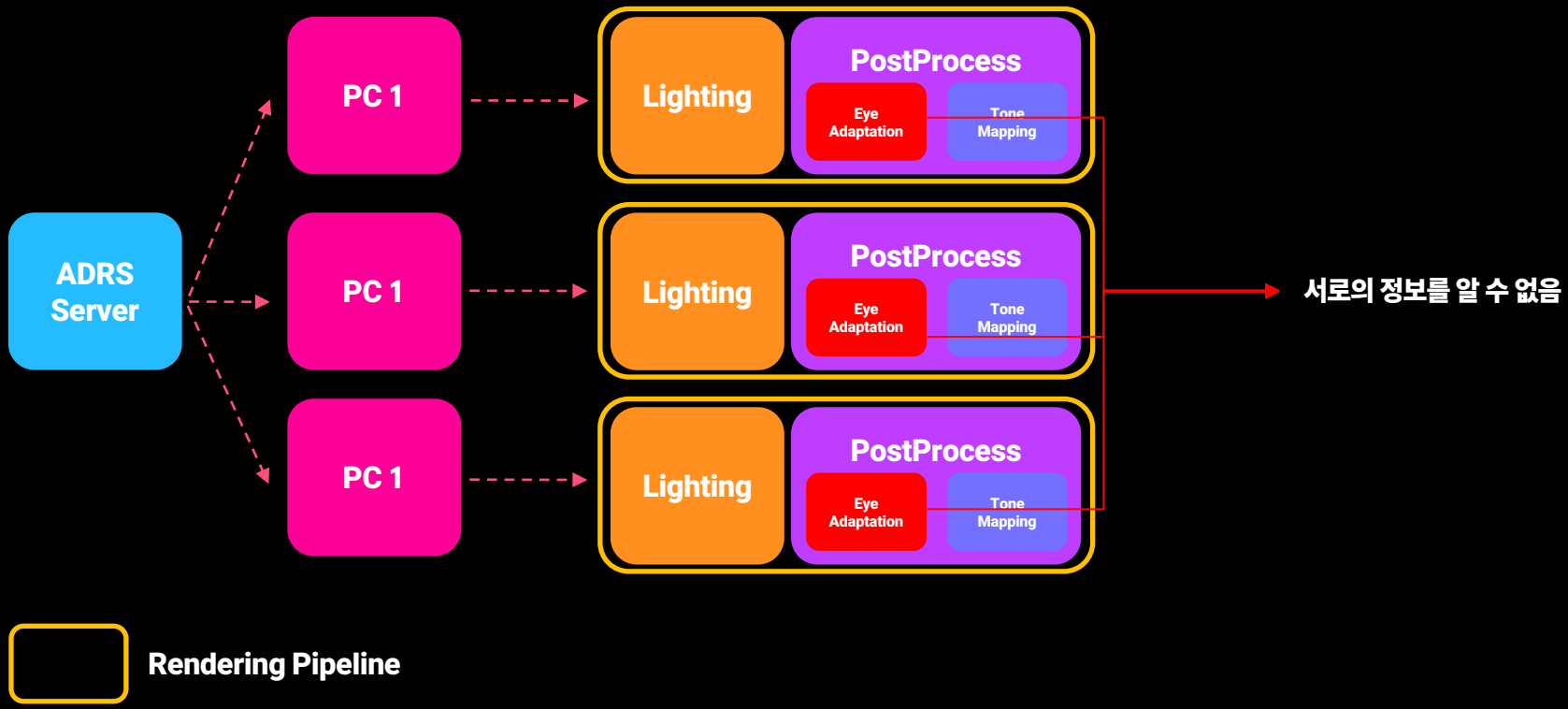
언리얼 엔진에서는 한 렌더링 파이프라인(한 Face)만을
위한 Eye Adaptation을 지원

분산 렌더링의 경우는, 다른 면을 그리지 않기 때문에,
다른 면의 노출 값을 알 수가 없음

한 PC에서 다른 면을 모두 그리게 되면
분산 렌더링의 이점이 사라짐

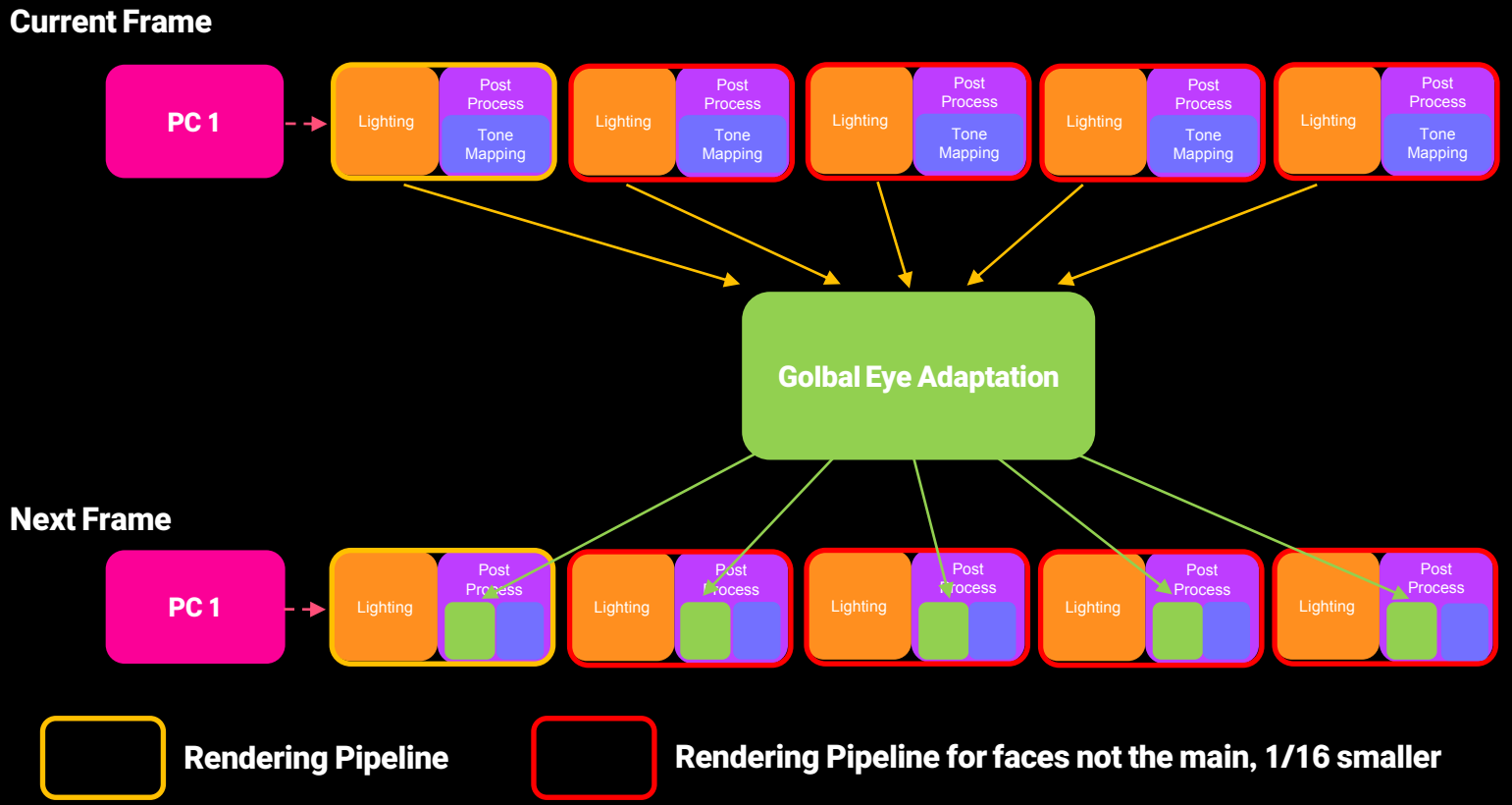
눈적응 - Eye Adaptaion (Exposure)

AMAZEVR



눈적응 - Eye Adaptaion (Exposure)

AMAZEVR



눈적응 - Eye Adaptaion (Exposure) 수정

AMAZEVR

PostProcessEyeAdaptation

PostProcessHistogram

PostProcessing

눈적응 - Eye Adaptaion (Exposure) 문제 해결 후

AMAZEVR

Before & After

AMAZEVR

한계와 앞으로 해야 할 일

Limitation and Future Works

한계 (Limitation) - Lumen

AMAZEVR

Global Illumination이 정확히 일치하지 않음

디버깅을 위한 LightProbe의 색상이, 렌더링마다 일정하지 않음
레이트레이싱 연산 중 Random한 부분이 있거나,
스크린 스페이스 정보를 사용때문이라 추정

한계 (Limitation) - VFX

AMAZEVR

GPU Collision

2D Fluid Simulation

3D Fluid Simulation의 경우 CINE 버전이 존재

앞으로 해야 할 일 - Future Works

AMAZEVR

RGBA16Float Format으로 이미지 쓰기

현재는 8Bit PNG 이미지로 쓴 후, Warping / Merging

같은 면 렌더링을 분산

GI가 가장 난관일 것 같지만, Warm up 프레임과 Time Reset만 잘 맞추면 이미 가능할 수도 있음

VR이 아닌 일반 시네마틱에도 적용 가능!

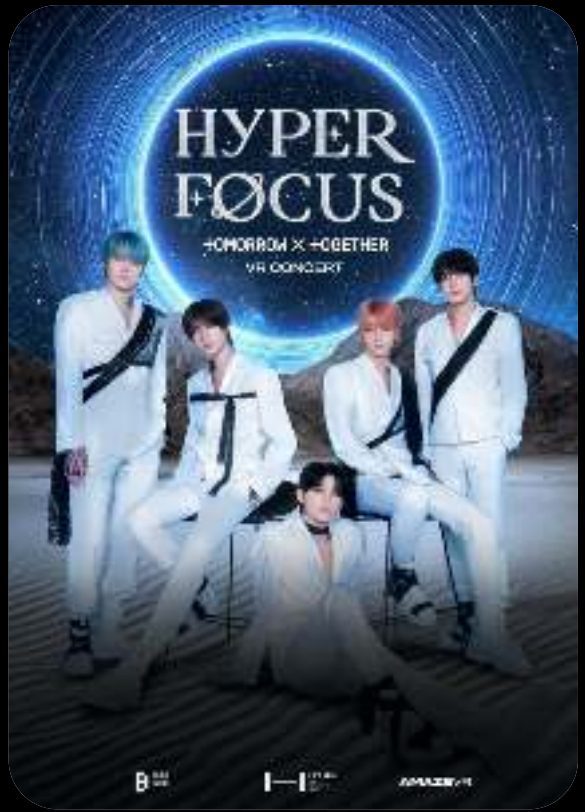
분산 렌더링 PC를 10대로 증설

VR 시네마틱의 경우 약 1.4배 ~ 2배 빨라질 것으로 기대

AMAZEVR


LYNKPOP
THE 1st VR CONCERT
aespa
2024.4


LYNKPOP
THE VR CONCERT
KAI


HYPER FOCUS
TOMORROW X TOGETHER
VR CONCERT


We are Hiring!
감사합니다
AMAZEVR