2025-2 미디어프로젝트
Final Presentation

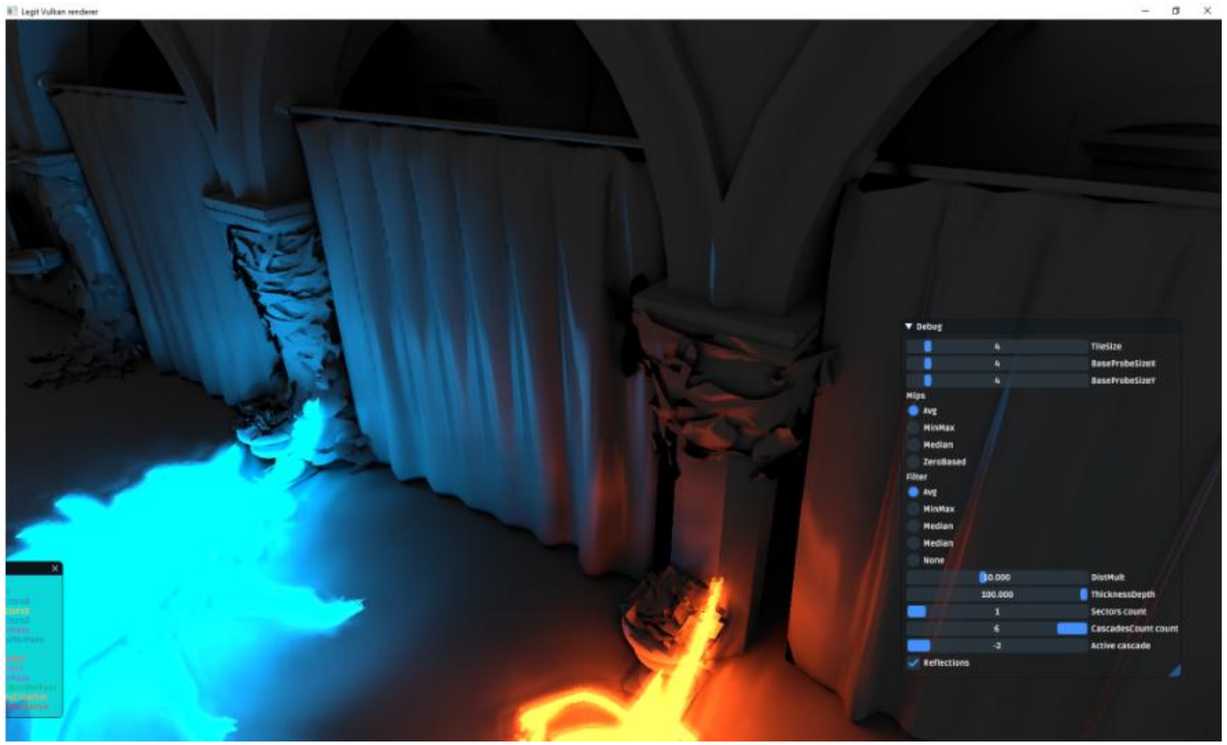
Radiance Cascades를 활용한 실시간 global illumination 구현

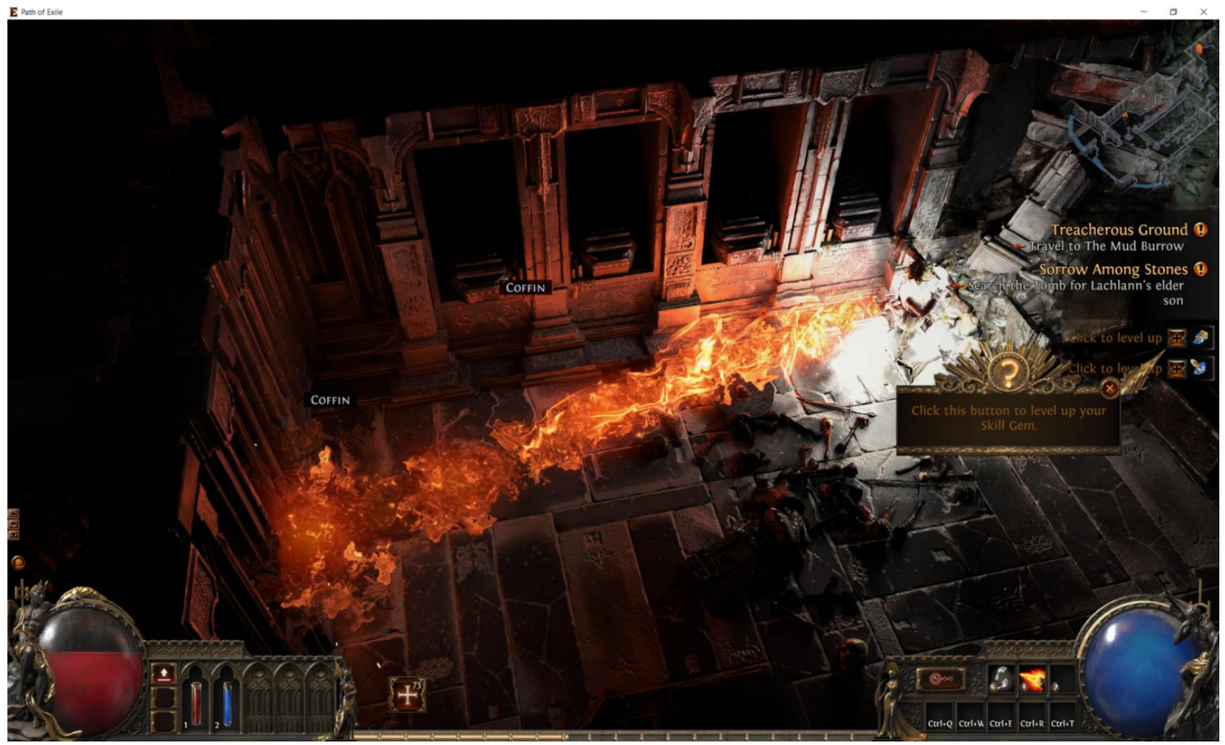
2025.11.24.

31조 / Team Radiant
소프트웨어학과 한동엽 강전찬

Radiance Cascades*

노이즈나 성능 저하 없이 실시간 전역 조명을 저사양 하드웨어에서도 구현할 수 있는 빠르고 효율적인 방식





Light probe가 적은 경우 더 많은 방향으로 ray marching을 하고,
Light probe를 많이 배치하는 대신, 더 적은 방향으로 ray marching을 진행하여

이를 여러 단계에 걸쳐 “cascade”시켜서 통합하는 것으로 간접광을 근사하는 알고리즘

**Path of Exile 2*, Grinding Gear Games. In-game screenshot by Alexander Sannikov

Radiance Cascades란?

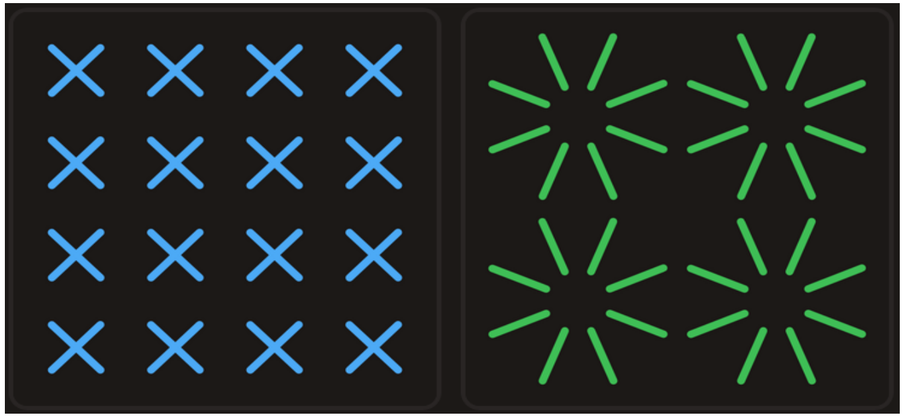
Q: 렌더링의 품질을 높이기 위해서는?

- 광선의 개수를 최대한 많이 늘린다 (X)

Radiance Cascades란?

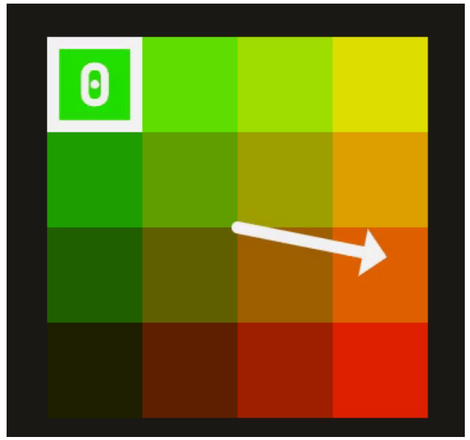
Q: 렌더링의 품질을 높이기 위해서는?

- 광선의 개수를 최대한 많이 늘린다 (X)
- 가까운 곳은 촘촘하게
  먼 곳은 듬성듬성하게 샘플링을 진행하는
  cascade의 구조를 도입
- *spatial* frequency와 *angular* frequency가
  서로 반비례 관계가 되는 점을 활용함.
- 이를 통해 모든 pixel에 광선을 쏘지 않고도
  효율적인 연산이 가능
- Direction-first layout을 적용하여
  GPU의 coalesced memory access 유도



(왼쪽)
4x4 spatial sampling
with 4 angular sampling

(오른쪽)
2x2 spatial sampling
With 16 angular sampling



16방향 샘플이
4x4 texture의 형태로 저장된 예시.

하나의 texel이 하나의 방향 샘플

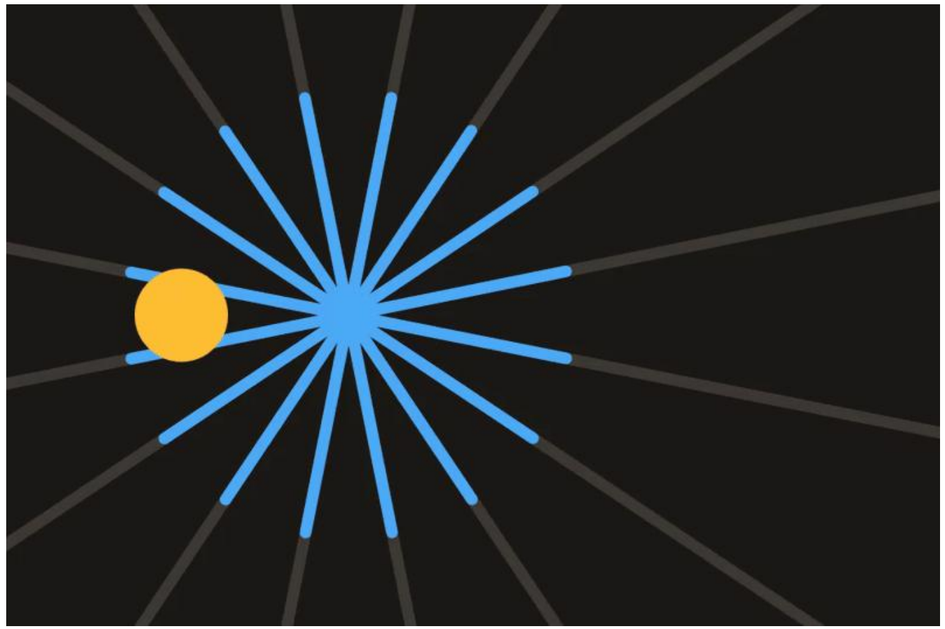
Figures from m4xc.dev

Adaptive Optimization의 필요성

Main Contribution

IDEA

- Scene geometry의 복잡도를 실시간으로 분석하여,
  - 변화가 많은 곳에는 sample을 더 많이 할당하고
  - 변화가 적은 곳에는 sample을 더 적게 할당하면 어떨까?



물체와의 거리가 멀어질수록,
angular frequency를 늘려야 샘플링이 성공한다.

반대로, scene geometry에 대한 prior가 있으면,
이 angular frequency를 동적으로 조절하여
성능과 품질 간의 trade-off를 더 효과적으로 맞출 것이다.

Figures from m4xc.dev

Main Contribution

Adaptive Sampling Density

```hlsl
int GetAdaptiveStepCount(int cascadeLevel, float2 probeUV)
{
    // 1. 현재 픽셀의 기하학적 복잡도(Variance) 샘플링
    float variance = SAMPLE_TEXTURE2D_LOD(_VarianceDepth, sampler, probeUV, cascadeLevel).x;

    // 2. 분산도에 따라 Ray Marching 횟수 동적 할당
    // 복잡도가 낮으면 _MinRaySteps, 높으면 _MaxRaySteps로 선형 보간
    float stepCount = lerp((float)_MinRaySteps, (float)_MaxRaySteps, saturate(variance * _Influence));

    return (int)stepCount;
```

- 현재 pixel 주변의 depth 변화를 확인하여:
  - Low variance일 경우 최소 횟수만큼
  - High variance일 경우 최대 횟수까지
  - ray marching 과정의 step count 한계를 설정합니다.

빨간 원 안에 있는 부분은 depth 변화가 있으며,
초록 원 안에 있는 부분은 (너무 멀어서) 변화가 없다.

Main Contribution

Adaptive Ray Scale / Cascade Scale Factor

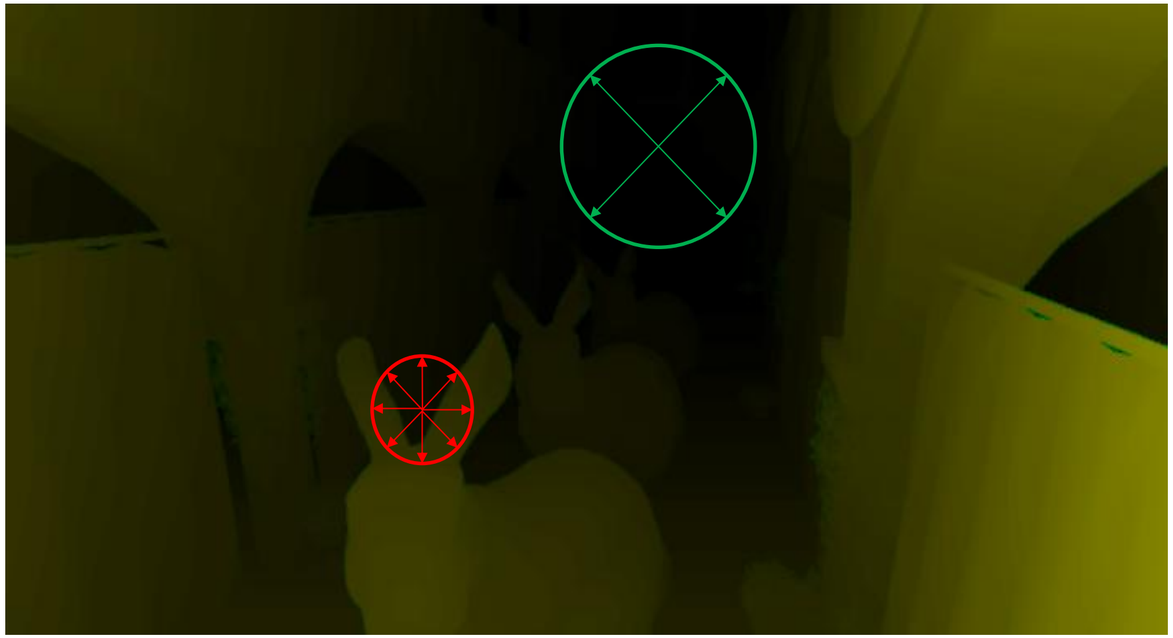
Cascade Scale Factor (CSF)

- Global한 scaling 효과
- Ray의 길이를 전부 scale함
- 적은 비용으로 넓은 계산 범위를 확보 가능
- 복잡한 곳에서 품질 저하

$$L_{final} = L_{base} \times CSF \times ARS(\mathbf{v})$$

Adaptive Ray Scaling (ARS)

- Local한 제어 가능
- Scene 복잡도에 따라 ray 길이 조절
  - 단순한 scene에서는 ray를 길게 할당
  - 복잡한 scene에서는 ray를 짧게 할당



Rendering Results*

Cornell Box Rendering

Options			Results
Adaptive Ray Scale	Cascade Scale Factor	Adaptive Sampling Density	1프레임 렌더링 시간
X	X	X	1.29ms
O	X	X	1.27ms
X	O	X	1.30ms
X	X	O	1.31ms
O	O	X	1.51ms
O	X	O	1.30ms
X	O	O	1.33ms
O	O	O	**1.54ms**

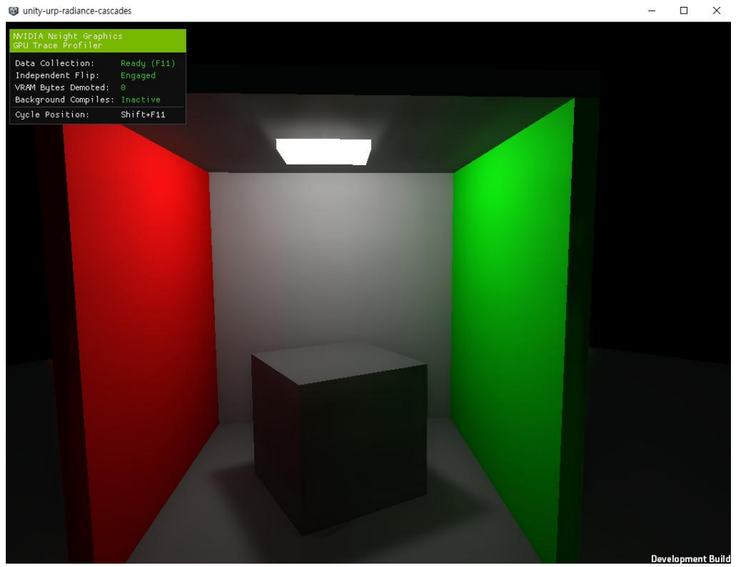
Top-down Scene Rendering

Options			Results
Adaptive Ray Scale	Cascade Scale Factor	Adaptive Sampling Density	1프레임 렌더링 시간
X	X	X	1.53ms
O	X	X	1.89ms
X	O	X	1.77ms
X	X	O	3.87ms
O	O	X	1.57ms
O	X	O	3.90ms
X	O	O	3.95ms
O	O	O	**6.14ms**

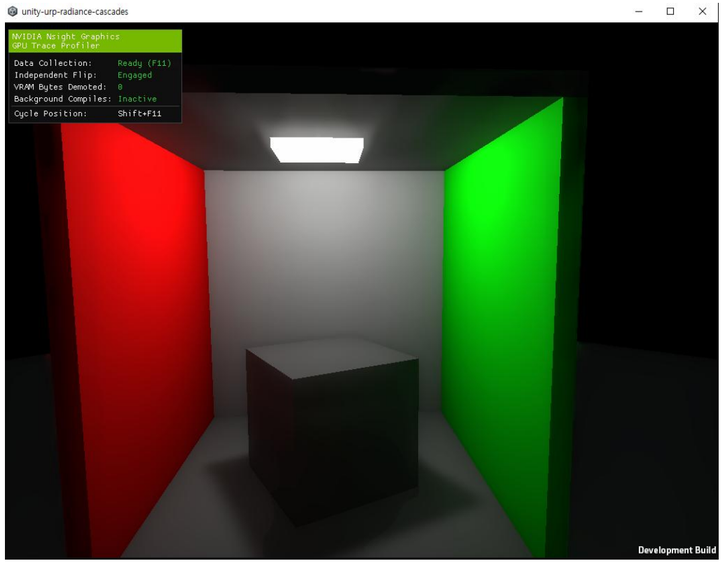
*Measured with NVIDIA Nsight on RTX 3080 w/ Ryzen 9 3950X

실험 결과

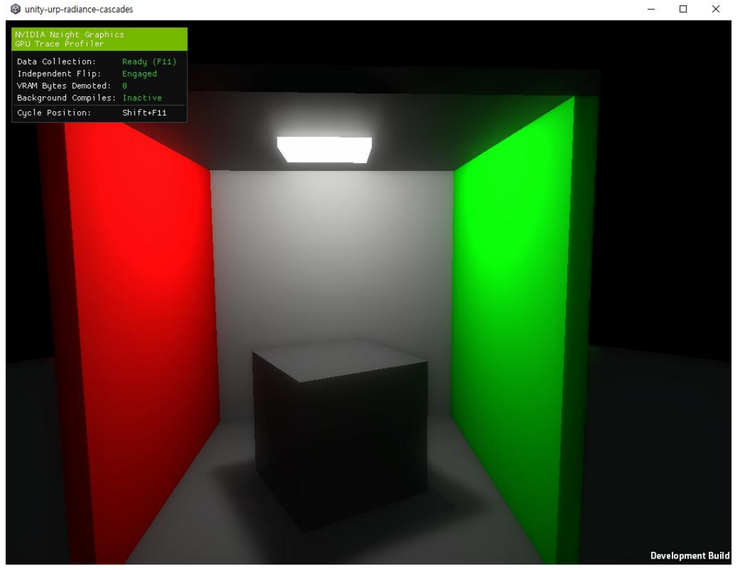
Rendering Results* on Cornell Box Scene



Base
(1.29ms)



Adaptive Sampling Density
(1.31ms)



ALL
(1.54ms)

간단한 scene의 경우에는 알고리즘이 자동으로 판단하여
품질을 망가뜨리지 않는 한에서 최소한의 연산만을 수행하여 효율성을 유지합니다.

*Measured with NVIDIA Nsight on RTX 3080 w/ Ryzen 9 3950X

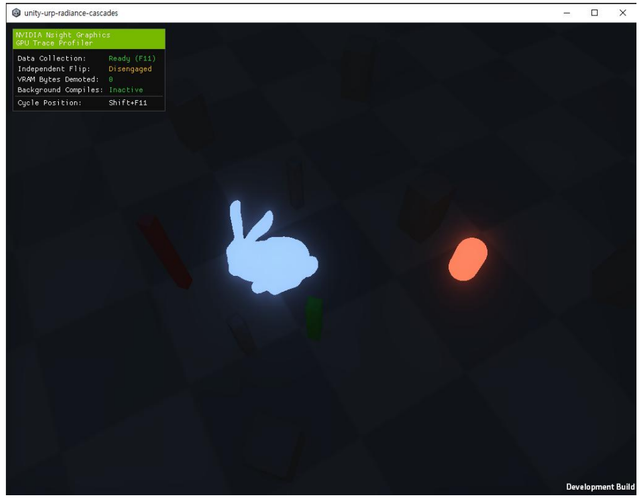
실험 결과

Rendering Results* on Top-Down Scene

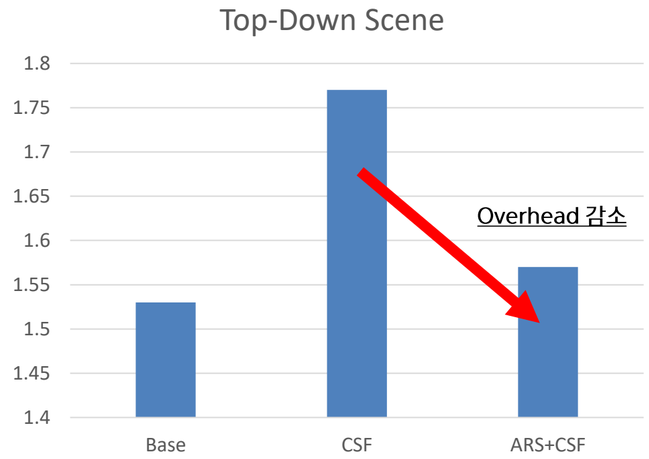
Base
(1.53ms)

CSF
(1.77ms)

ARS+CSF
(1.57ms)



ALL
(6.14ms)



품질을 높이면서도
성능 손실을 최소화하였음

*Measured with NVIDIA Nsight on RTX 3080 w/ Ryzen 9 3950X

결론 및 향후 과제

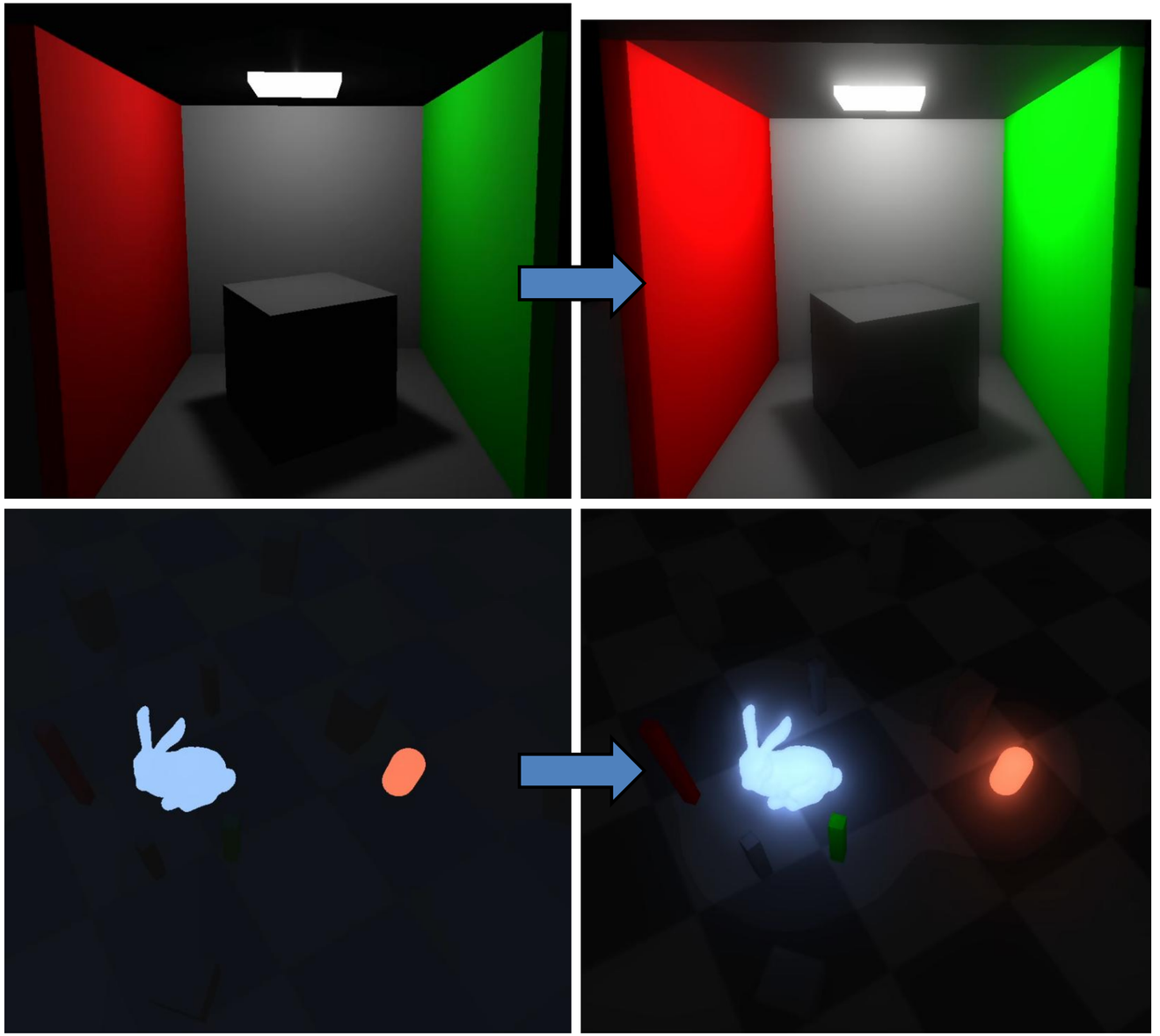
결론: 실시간 GI의 한계 극복 및 최적화 시도

- 실시간성 보장을 위한 안정적이고 강건한 알고리즘
  - **문제**: 기존 방법들의 light leaking 및 메모리 관리 측면의 어려움

    → direction-first layout 및 bilateral merge로 해결하고자 함

- 동적 환경에 대응하는 adaptive performance
  - **문제**: Scene의 복잡도에 관계없이 고정된 computing cost

    → Scene variance를 분석하여, 알고리즘이 스스로 연산량 조절
    → 다양한 조합으로 성능-품질 tradeoff의 균형을 찾을 수 있음



결론 및 향후 과제

향후 과제: 더 가벼운 실시간성 확보를 위해서는?

- 조금만 복잡해진 scene에서도 **2~4배 time이 증가**하는 결과
- 만약, 훨씬 더 복잡한 scene이라면 overhead가 발생할 수 있다는 추측

- 이전 frame 재활용 및 early exit 등, 다양한 방법으로 추가 최적화 필요

Top-down Scene Rendering

Options			Results	
Adaptive Ray Scale	Cascade Scale Factor	Adaptive Sampling Density	1프레임 렌더링 시간	증감률
X	X	X	1.53ms	-
O	X	X	1.89ms	123%
X	O	X	1.77ms	115%
X	X	O	3.87ms	253%
O	O	X	1.57ms	102%
O	X	O	3.90ms	255%
X	O	O	3.95ms	258%
O	O	O	**6.14ms**	**401%**

감사합니다.

2025-2 미디어프로젝트

Radiance Cascades를 활용한
실시간 Global Illumination 구현

31조 / Team Radiant
소프트웨어학과 한동엽 강전찬