『Tales of ARISE』におけるレンダリング技術と高速化

株式会社バンダイナムコスタジオ　小林弘幸

CEDEC2019
Computer Entertainment Developers Conference

TALES of ARISE

今世代機を主軸とする最初のシリーズ作品

- 今作の絵作りで目指したもの
  - 大人っぽさ、実存感、緊迫感
  - シリーズ最新作として飛躍を感じられる品質
    - それを実現するための技術、ノウハウを獲得する
  - シリーズの特徴であるアニメ・漫画調の雰囲気
    - 手描きの水彩イラスト感、色の使い方、情報の誇張・省略
- アーティスト側からの要望として
  - 技術的な要素を活用した絵作り（職人芸への依存を下げる）
    - 物量が多い＆多くのスタッフが関わる開発

グラフィック開発の方針

- 開発環境としては Unreal Engine 4 を使用
- 描画機能は独自実装、カスタム実装で作りこんでいくことに
  - 絵作りのチューニング
  - **処理負荷を抑える、対処のしやすさの向上**
    - **より円滑な開発とグラフィックの品質向上**
    - **自分達でチューニング出来る領域を増やす**
  - 今後に生かせるノウハウの獲得

- 本講演ではこの独自実装の部分に焦点を当てて説明します
  （画像は全て開発中のものとなります）

ARISEの背景ルック

ライティング
（Deferredベース）
空気感の表現
手描きの水彩感

TALES of ARISE

ARISEの背景ルック

空間の広がり
ボリューム感の
ある雲

CEDEC 2019
Computer Entertainment Developers Conference

TALES of ARISE

ARISEのキャラルック

ライティング
イラスト調の表現

CEDEC2019
Computer Entertainment Developers Conference

TALES of ARISE

講演概要

1. レンダリングパイプライン概要
2. 背景のライティング・シェーディング
3. 空気感の表現とボリューメトリックライティング・フォグ
4. ボリューム感のある雲表現
5. キャラクタのライティング・シェーディング
6. 半透明描画の高速化

※本講演ではPS4における負荷数値をいくつか提示していきます（1080p）。
XboxOneについてもARISEでは同様の実装・高速化を活用しています。

1. レンダリングパイプライン概要

レンダリングフローの設計方針

- Deferred + 別パスでのシェーディング・フィルタ処理
  - DeferredLightingのような設計
- DepthPrePass（フル）+ BasePass描画（Gbuffer生成）
  - マテリアルのGPU負荷を下げることを第一
    - アーティスト側での実機確認・チューニングが難しい部分
  - 頂点負荷は増加しやすいが・・・
    - リダクション・LODツールを活用する（アーティスト側で調整しやすい部分）
- モデル描画にAsyncComputeを極力重ねる
  - ARISEの描画要素の負荷を隠蔽する実装

グラフィックス

DepthPrePass (Full)	BasePass (GBuffer)	ShadowMap	Occlusion Culling	Direct Lighting	Shading + Filter	Translucent	PostProcess

非同期コンピュート

AsyncPrePass	AsyncBasePass	AsyncAO	Async Deferred Shadow				Async Temporal AA

モデル描画のパス

グラフィックス

DepthPrePass (Full)	BasePass (GBuffer)	ShadowMap	Occlusion Culling	Direct Lighting	Shading + Filter	Translucent	PostProcess

非同期コンピュート

AsyncPrePass	AsyncBasePass	AsyncAO	Async Deferred Shadow				Async Temporal AA

Async PrePass

- DepthPrePassと同時に実行
- フレームの先頭からすぐに計算可能な要素が対象
  - ポストプロセスの一部
  - 一部バッファの初期化
  - スクリーンスペースGI（Diffuse GI）
  - ボリューメトリックなライティングとフォグ
  - ボリュームのある雲描画

グラフィックス

DepthPrePass (Full)	BasePass (GBuffer)	ShadowMap	Occlusion Culling	Direct Lighting	Shading + Filter	Translucent	PostProcess

非同期コンピュート

AsyncPrePass	AsyncBasePass	AsyncAO	Async Deferred Shadow				Async Temporal AA

Async BasePass

- BasePassと同時に非同期実行
- デプスのみが確定していれば計算可能な処理が対象
  - タイルベースのライトカリング
  - タイルベースのスクリーンスペースシャドウ
  - スクリーンスペースリフレクション
  - レイマーチでのハイトマップシャドウ（中景～遠景のキャストシャドウ）
    - 低コスト＆太陽方向の動的変化に対応（太陽方向にレイマーチ）

グラフィックス

DepthPrePass (Full)	BasePass (GBuffer)	ShadowMap	Occlusion Culling	Direct Lighting	Shading + Filter	Translucent	PostProcess

非同期コンピュート

AsyncPrePass	AsyncBasePass	AsyncAO	Async Deferred Shadow				Async Temporal AA

Async AO
- シャドウマップ描画と同時に非同期実行
- SSAO
  - Scalable Ambient Obscurance をベース
- ハイトマップAO
  - 高い位置からのスケールのやや大きいAO

Async Deferred Shadow
- **OcclusionCullingやDecal描画と非同期実行**
- **キャストシャドウをスクリーンに展開**
  - シャドウマップ
  - ハイトマップシャドウ
  - 雲からの落ち影

グラフィックス

DepthPrePass (Full)	BasePass (GBuffer)	ShadowMap	Occlusion Culling	Direct Lighting	Shading + Filter	Translucent	PostProcess

非同期コンピュート

AsyncPrePass	AsyncBasePass	AsyncAO	Async Deferred Shadow	Async Temporal AA

Direct Lighting
- 平行光源（太陽光）
- ポイントライト、スポットライト
  - タイルベース
- 乗算ライト
  - 影付け、色付け用途

Shading + Filter
- 間接光の適用
- フォグ、雲描画の適用
- イラスト感を足すための画面効果
  - 影色制御、輪郭線、フィルタ

グラフィックス

DepthPrePass (Full)	BasePass (GBuffer)	ShadowMap	Occlusion Culling	Direct Lighting	Shading + Filter	Translucent	PostProcess

非同期コンピュート

AsyncPrePass	AsyncBasePass	AsyncAO	Async Deferred Shadow	Async Temporal AA

- **GPU負荷に注意が必要な描画要素**
  - どちらもフルスクリーンのパスが多く走る
  - シェーディングモデル（キャラ・背景）ごとに必要な挙動も異なる
- **負荷の観点でキャラ、背景で専用シェーダを実行出来ると理想**
  - UberShaderは避けたい（レジスタプレッシャー、Wave Occupancyの低下）
  - 「シェーダIDのフェッチ ⇒ 条件分岐 ⇒ 専用シェーダ実行」も無駄が多い（フルスクリーン）

⇒デプス（Hizテスト）でのシェーダマスクを使ったシェーダ実行

デプスを使ったシェーダマスク作成

- シェーダIDに応じたデプスマスクの作成
  - 16bitデプス
  - シェーダID ⇒ PSからDepth Export（DefaultLitを初期値⇒z圧縮）
  - PS4で約0.1ms

シーンカラー

シェーダマスク

デプスを使ったシェーダマスク実行

- EarlyZ + HiZ（タイル単位のデプステスト）を活用

実行したいシェーダに
対応するデプスを出力

特定のマテリアルの
ピクセルのみ実行

  - DefaultLit + Foliage を一括実行（背景ライティング）
  - DefaultLit + Foliage + Unlit を一括実行(背景シェーディング、フィルタ)
  - キャラマテリアル一括実行 or 個別（ライティング、シェーディング、フィルタ）
- 特定のマテリアルのみ専用シェーダを実行（テストの粒度に注意）

HiZ & z圧縮の効果

PS4でのGPU負荷	DirectLighting + Shading + Filter
HiZ + EarlyZ	約5.2ms
EarlyZテストのみ（HiZなし）	約5.8ms

XboxOneではESRAMに配置せず運用

※ステンシルについて

– 別用途での利用、圧縮・HiSの扱いなどの観点でデプスを選択

グラフィックス

DepthPrePass (Full)	BasePass (GBuffer)	ShadowMap	Occlusion Culling	Direct Lighting	Shading + Filter	Translucent	PostProcess

非同期コンピュート

AsyncPrePass	AsyncBasePass	AsyncAO	Async Deferred Shadow				Async Temporal AA

- **Translucent**
  - Mixed Resolution Rendering（後のスライドで）
- **PostProcess**
  - Bloom + Glow（閾値なしのぼかし発光）
    - Glow専用のマテリアルを描画出来るように（モンスターで活用）
  - TemporalAAとSMAAのハイブリッド
    - TemporalAAは非同期で計算

グラフィックス

DepthPrePass (Full)	BasePass (GBuffer)	ShadowMap	Occlusion Culling	Direct Lighting	Shading + Filter	Translucent	PostProcess

非同期コンピュート

AsyncPrePass	AsyncBasePass	AsyncAO	Async Deferred Shadow		Async Temporal AA

GPU負荷（PS4）

AsyncCompute	コンピュートの負荷合計
On	3.2ms
Off	6.9ms

AsyncComputeがGPUを占有しては意味がない
（グラフィックとWaveがほどよく混ざると良い）
Waveの同時実行数の調整は重要（Wave Limit）

- シェーダごと
- 機種ごと

マテリアル描画の負荷を抑えつつ、
非同期コンピュートで全体を軽量化

2. 背景のライティング・シェーディング（間接光とフィルタ処理）

背景の間接光表現

- 基本的にリアルタイム計算（IBL除く）
  - ライティング環境の動的変化に対応（太陽光,街灯など）
  - アセット配置 ⇒ 見た目を即確認出来る環境を重視（作業コスト）
- 間接光の要素
  - スカイライト + リフレクションIBL
  - AO(SSAO + ハイトマップAO)
  - Screen Space Diffuse GI
  - Screen Space Reflection

Screen Space Diffuse GI

- 周囲への照り返しや色の移りこみを疑似的に表現
  - ライティング結果やマテリアルのEmissiveなど


DiffuseGI OFF
CEDEC2019
Computer Entertainment Developers Conference
TALES of ARISE
©BANDAI NAMCO Entertainment Inc.


DiffuseGI ON
CEDEC2019
Computer Entertainment Developers Conference
TALES of ARISE
©BANDAI NAMCO Entertainment Inc.


DiffuseGI OFF
CEDEC2019
Computer Entertainment Developers Conference
TALES of ARISE
©BANDAI NAMCO Entertainment Inc.


DiffuseGI ON
CEDEC 2019
Computer Entertainment Developers Conference
TALES of ARISE
©BANDAI NAMCO Entertainment Inc.

Screen Space Diffuse GI
セットアップ（AsyncPrePass）

前フレームのバッファ

直接光ライティング + Emissve

シーンデプス

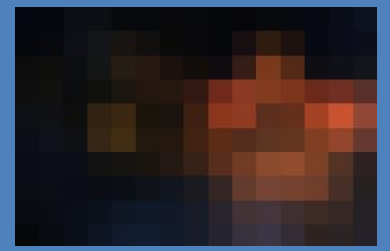
デプスレイヤー作成

近景用

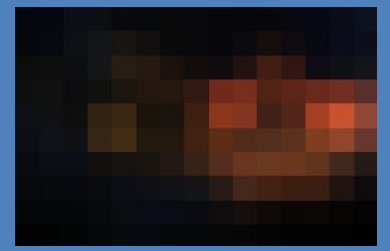
中景用

遠景用

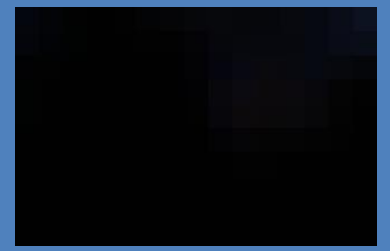
縮小ぼかし作成



近景用



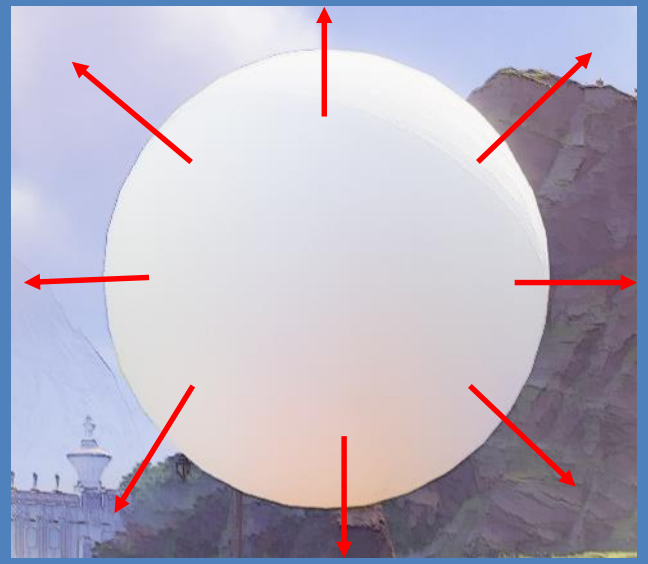
中景用



遠景用

Screen Space Diffuse GI
シーンへの適用（Shadingパス）

シーンに適用



ピクセル法線、デプスを考慮して
・縮小ぼかしバッファにアクセス
（ARISEでは3回）
・強弱補正

手前のEmissiveが遠景に作用してしまう

Screen Space Diffuse GI
PS4のGPU負荷

AsyncCompute	セットアップ	シーン適用
On	0.2ms	0.2ms
Off	0.4～0.5ms	0.2ms

Screen Space Reflection

- デプスのみで計算可能な平面反射
  - カットシーンやフィールド移動で一番目立つ反射は床面（負荷対効果）
  - 疑似的なGlossy Reflectionにも対応
    - ラフネスに応じてボケ幅変化
  - レイマーチベースの多階層の反射として実装


半透明へのSSR Off
CEDEC2019
Computer Entertainment Developers Conference
TALES of ARISE
©BANDAI NAMCO Entertainment Inc.

半透明へのSSR On

Screen Space Reflection
反射計算(AsyncBasePass)

前フレームの
カラーバッファ

現フレームの
デプスバッファ

反射計算

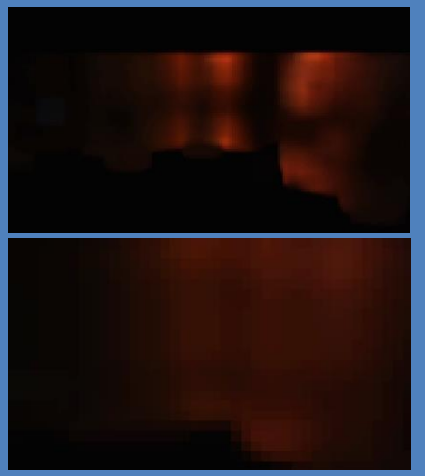
・法線が真上を前提として計算
・カメラより上の位置のピクセルは計算から除外
・レイマーチ（ヒット位置計算）+ バイナリサーチ（リファイン）
・Temporal Reprojection + Temporal Filter
・解像度は画面サイズの縦横 1/2

入力カラー

出力画像

縮小ぼかし処理

・反射カラーをぼかす
（Glossy Reflection）
・Asyncパスで一緒に処理



Screen Space Reflection
シーンへの適用（Shadingパス）

- ピクセル法線と真上方向を比較、フェード
  - 法線方向でUV歪み（若干）
- ラフネスに応じて反射バッファを補間

AsyncCompute	SSR計算(画面2/3)	SSR計算(画面全体)
On	0.4ms	0.6ms
Off	0.6ms	0.9ms

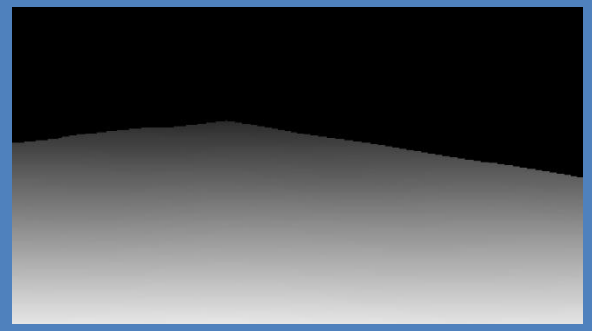
PS4でのGPU負荷

半透明へのSSR

- 不透明と半透明で計算を共通にする手法を採用
  - マテリアル内での計算だとSSRが2回走ってしまう

1. SSR半透明専用デプスを描画

解像度は画面の1/4*1/4



不透明のデプス

2. デプスをマージ

縮小デプス作成パスで同時に

SSR専用のデプス

背景のShadingとFilter

- 手描きの水彩イラスト感を増すための処理
  - 色の変化
    - 影色、影の際の色、陰部分の彩度
  - 空気感とフォグ効果
    - 後のスライドで
  - 手描き感向上のためのフィルタ処理

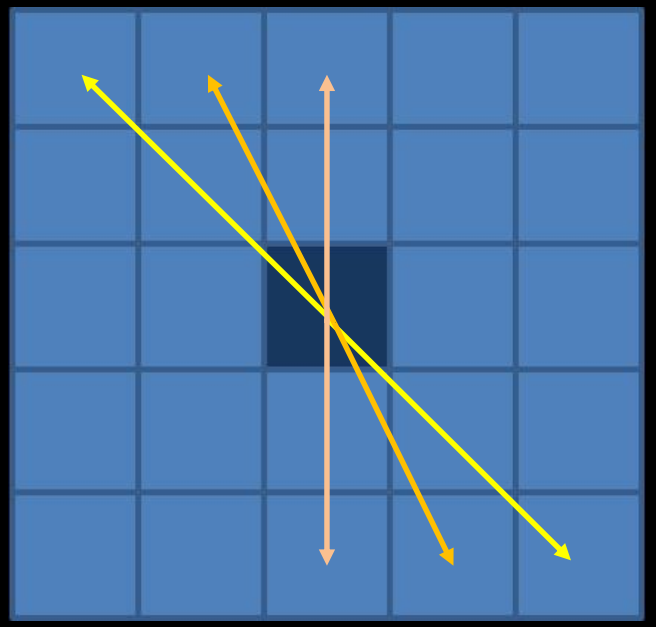

Shading + Filter Off
（直接光 + スカイライトのみ）
CEDEC2019
Computer Entertainment Developers Conference
TALES of ARISE
©BANDAI NAMCO Entertainment Inc.


Shading + Filter On
CEDEC 2019
Computer Entertainment Developers Conference
TALES of ARISE
©BANDAI NAMCO Entertainment Inc.

Shading + Filter On

フィルタ処理概要

- カスタムSNN + シャープ + 輪郭線
  - カスタムSNN
    - 手描きっぽいブレ感やにじみ、面のつぶし
  - シャープ + 輪郭線（法線）
    - 細かい情報量の書き足し
- SNNフィルタとは？
  - 対角のピクセル同士を比較、平均を求める
  - ピクセルの混ざり方やノイズ感は良い
    - ボケ具合やフィルタサイズ、負荷などチューニングが必要



中心ピクセルと色がより近い
ピクセルの平均を取る

SNNのカスタム要素

- ブレンドウェイトの導入
  - バイラテラル的なウェイト付け（ボケ具合の対策）
- フィルタ形状
  - 輪郭強度によるフィルタ幅調整
    - 水平・垂直どちら寄りか、その強度など
- フィルタ強度
  - カメラ距離に応じて変更
- 負荷対策
  - 水平・垂直方向にSeparateなフィルタ実装
    - 7+7のサンプリング数

フィルタ処理のGPU負荷

	カスタムSNN + シャープ + 輪郭線
PS4のGPU負荷	約1.3ms

- サンプル数、キャッシュ、計算量

3. 空気感の表現と
ボリューメトリックライティング・フォグ

Ariseにおける空気感表現

- 環境表現の向上
  - 塵や砂埃、場のみずみずしさ、天候表現（風、雪）など
- 絵的な誇張表現
  - 距離感の誇張や情報量の整理
  - 日差しの綺麗さと影表現の強調
  - 手描きの絵っぽい繊細さ・丁寧さ
- 物理的な正しさよりも上記を優先した実装・チューニング
  - 処理負荷も同様に重視


Fog効果 Off
CEDEC2019
Computer Entertainment Developers Conference
TALES of ARISE
©BANDAI NAMCO Entertainment Inc.


Fog効果 On
CEDEC2019
Computer Entertainment Developers Conference
TALES of ARISE
©BANDAI NAMCO Entertainment Inc.


Fog効果 Off
CEDEC2019
Computer Entertainment Developers Conference
TALES of ARISE
©BANDAI NAMCO Entertainment Inc.


Fog効果 On
CEDEC2019
Computer Entertainment Developers Conference
TALES of ARISE
©BANDAI NAMCO Entertainment Inc.

空気感表現の構成要素

フォグ = Merge(高さフォグ, Voluemtric Fog)

フォグ濃度のムラ・アニメーション
（レイマーチと3Dノイズで実装）

（フォグ + 角度散乱）* Sun Volumetric Shadow + フォグ * Local Voluemtric Lighting

太陽方向からの光の散乱効果
・視線と光源の角度のみ
・フォグやデプスに依存しない加算
・スモッグっぽさを抑えつつ綺麗に見せるため

ポイント、スポットライトのライティング
（レイキャスト + レイマーチ）

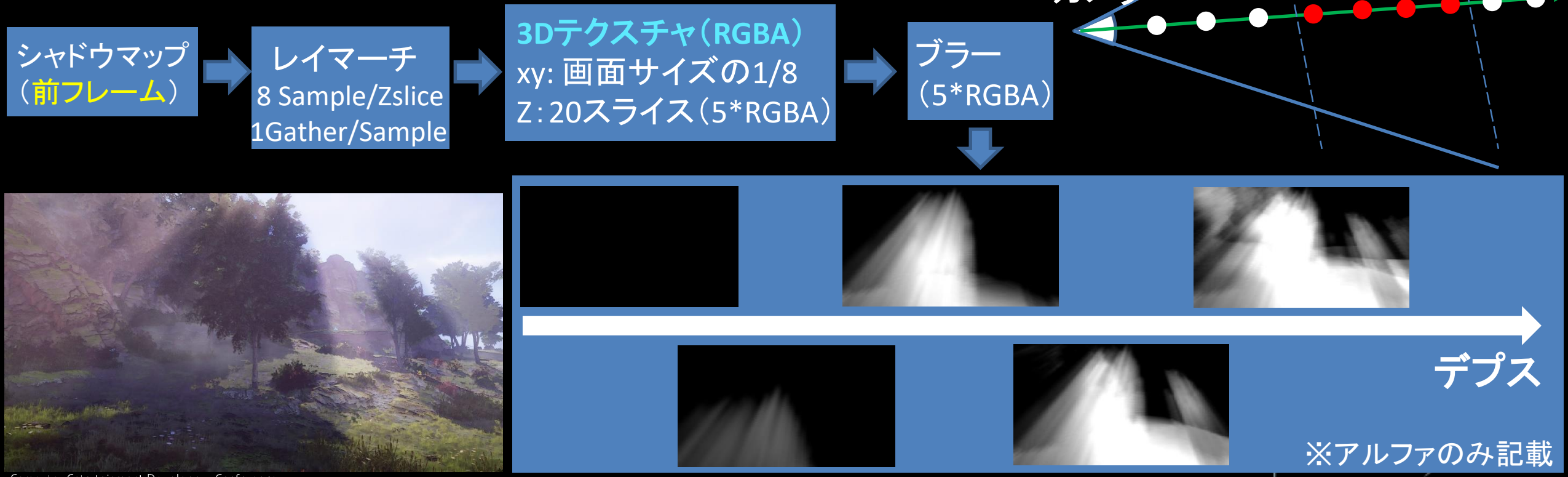
太陽方向のシャドウ
（光の方向を強調・印象付ける目的で使用）

- 各要素をそれぞれ独立して計算、シーン適用時にマージ
  - Volumetricな計算部分の簡略化、計算精度の調節


VolumetricFog On
CEDEC 2019
Computer Entertainment Developers Conference
TALES of ARISE
©BANDAI NAMCO Entertainment Inc.

Sun Volumetric Shadow

- レイマーチを使ったシンプルな実装
  - 影の累積のみ
  - DepthPrePassと非同期計算（AsyncPrePass）



Sun Volumetric Shadow

- シャドウのフェードをレイマーチ時に計算
  - 光の散乱感、空気の繊細な感じを追加
- シーンへの適用
  - デプスを使ってマニュアルの線形補間
  - マップごとの影色を適用

Volumetric Shadow Off

Volumetric Shadow On

CEDEC 2019
Computer Entertainment Developers Conference

TALES of ARISE

PS4でのGPU負荷

Async Compute	Volumetric Shadow + Volumetric Fog
On	0.3ms
Off	0.6ms

- 解像度、サンプル数/レイ、テクスチャ数と計算量/サンプル
- 3Dテクスチャは非常に便利
  - 解像度を抑える(xy)
  - コンピュートでの利用

4. ボリューム感のある雲の表現

ARISEにおける空の表現

- 空の広がり感（高低差、奥行き感）を大事に
  - 空に浮かぶ二つの星も実は超巨大なモデル
  - 空間的なレイヤー構造として表現

鳥 ⇒ ボリューム雲 ⇒ 山の頂上 ⇒ 遠景の薄雲 ⇒ 星

- ボリューム雲で目標としたこと
  - 形状の複雑さ（+ 異なる天候の雲）
  - 色を綺麗に
  - 動的に沸いて消えるアニメーション
- レイマーチを使って実装

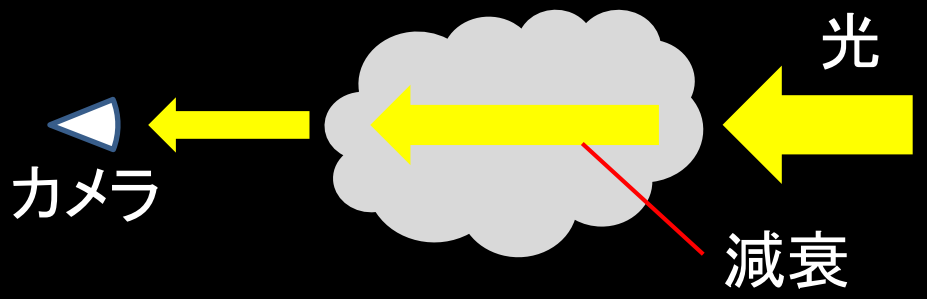
ボリューム感のある雲表現

ボリューム感のある雲表現

ボリューム感のある雲表現

雲描画の概要

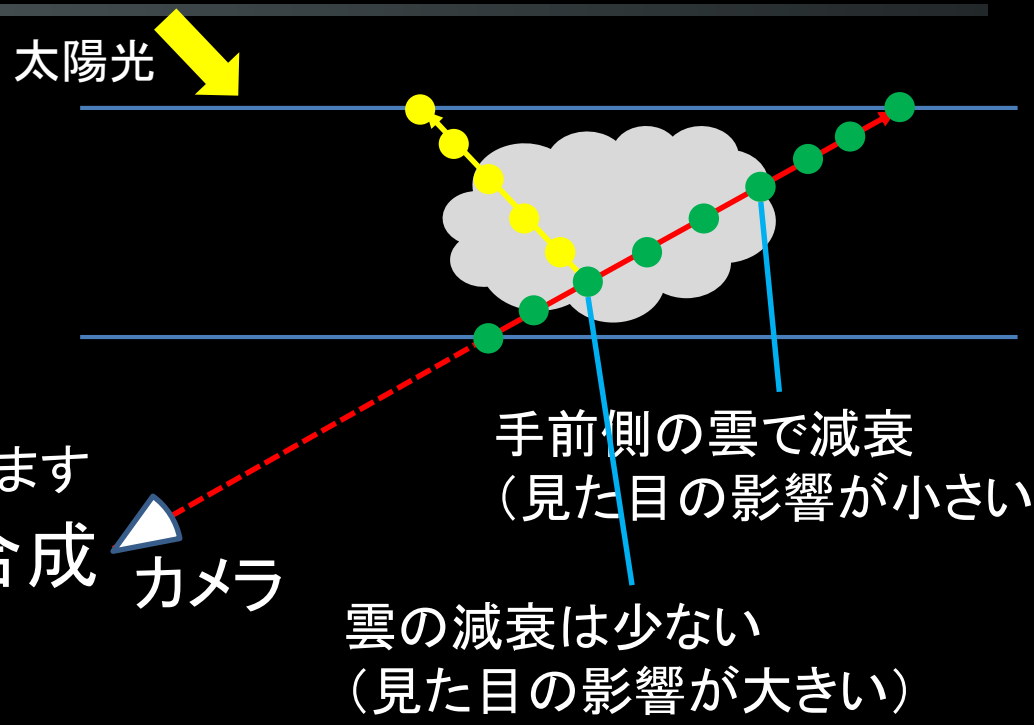
- 雲の見た目
  - 雲を通過・反射してカメラが受け取る光
  - 雲の描画 = 光の減衰・遮蔽（影）の計算
- 減衰の度合いは何で決まるか
  - 雲の厚み・濃さ
- 厚みの計算
  - レイマーチで計算する
- 濃さの計算
  - いい感じの濃度計算の関数を定義



雲描画においてPrimitiveな要素

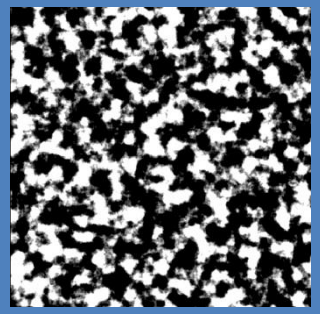
レイマーチでの雲描画

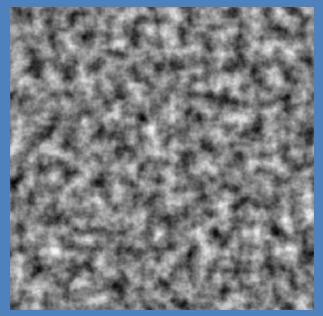
- 雲レイヤー間をレイマーチ
  1. 雲の濃度を計算（緑）
  2. 各サンプルごとにライティング計算
     1. セルフシャドウのレイマーチを実行（黄）
        1. このスライドではシャドウレイマーチと呼びます
  3. 濃度を考慮してライティング結果を合成
     1. レイの経路上の手前側の雲で減衰
- 処理負荷に注意が必要
  - 50サンプル + 10サンプル(シャドウ) = 550サンプル
  - 濃度のテクスチャ枚数も重要 ⇒ 2枚：1100回、3枚：1650回、4枚：2200

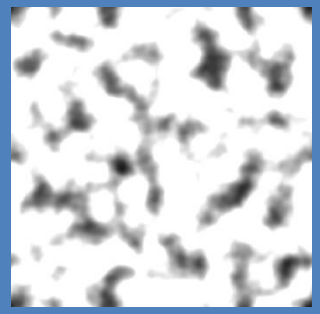


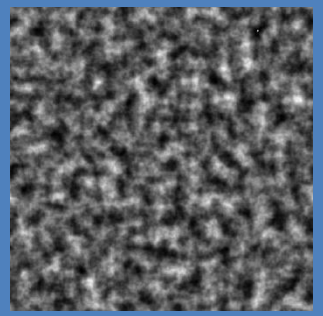
雲の濃度計算

雲の分布マップ：大まかな形を決定


水平方向の分布（R）


厚み分布（G）

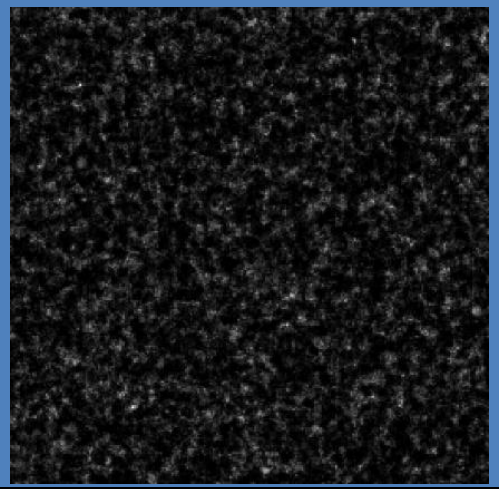

開始地点の高さ（B）


分布の追加補正（A）

3Dテクスチャ
xy: 512*512　空間の水平方向
z: 16　時間方向

Zは時間軸で変更（MirrorRepeat）
毎フレームzの2枚を線形補間
RはPerlin + Voronoiノイズ

詳細ノイズ：細かい形状を追加



3Dテクスチャ
xy: 256*256　空間の水平方向
z: 32　空間の高さ方向

Perlin + Voronoi ノイズ
z方向の枚数は少なめ(レイマーチのサンプル回数を考慮して)


テクスチャなし
（全部が雲）
CEDEC2019
Computer Entertainment Developers Conference
TALES of ARISE
©BANDAI NAMCO Entertainment Inc.

水平分布のみ（Rのみ）

水平分布 + 厚み・高さ分布（RGB）

分布マップ + 詳細ノイズ

雲の描画フロー

- シャドウと雲描画を別パス（別解像度）で描画

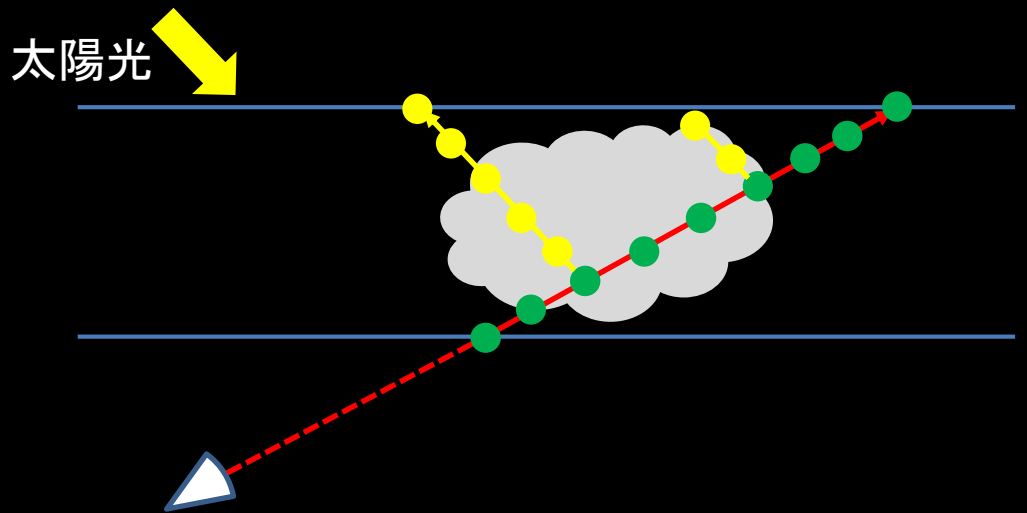
シャドウレイマーチ（下図黄色）
- シャドウ強度だけを計算
- 雲描画より低解像度にして負荷を抑える
- サンプル回数は5回（上端近くではさらに削減）
- 濃度テクスチャ：荒いMipを使用

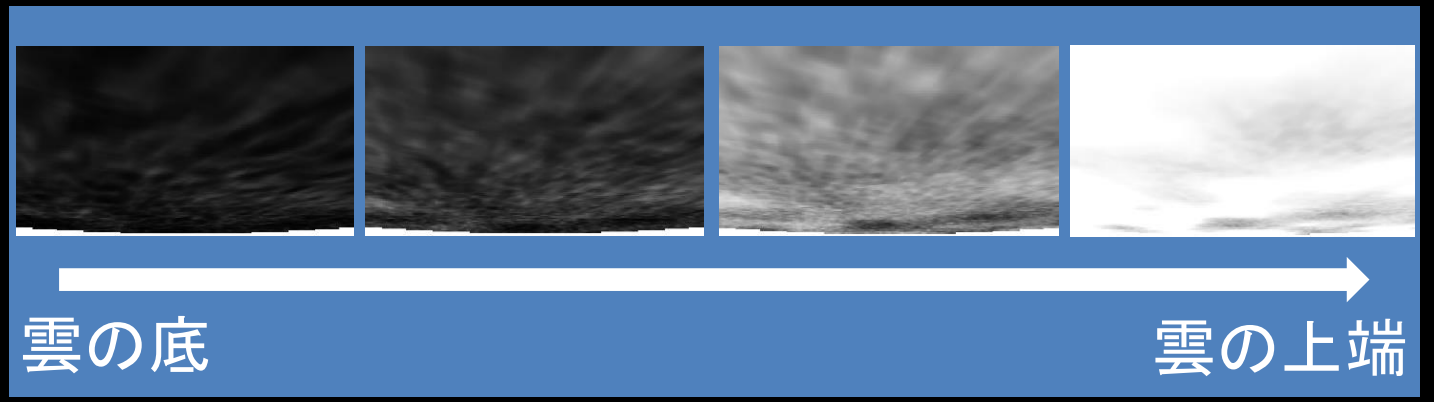
シャドウテクスチャ（3D）
xy: 画面サイズの1/8*1/8
z: 40スライス

Z方向はレイマーチ方向（緑）

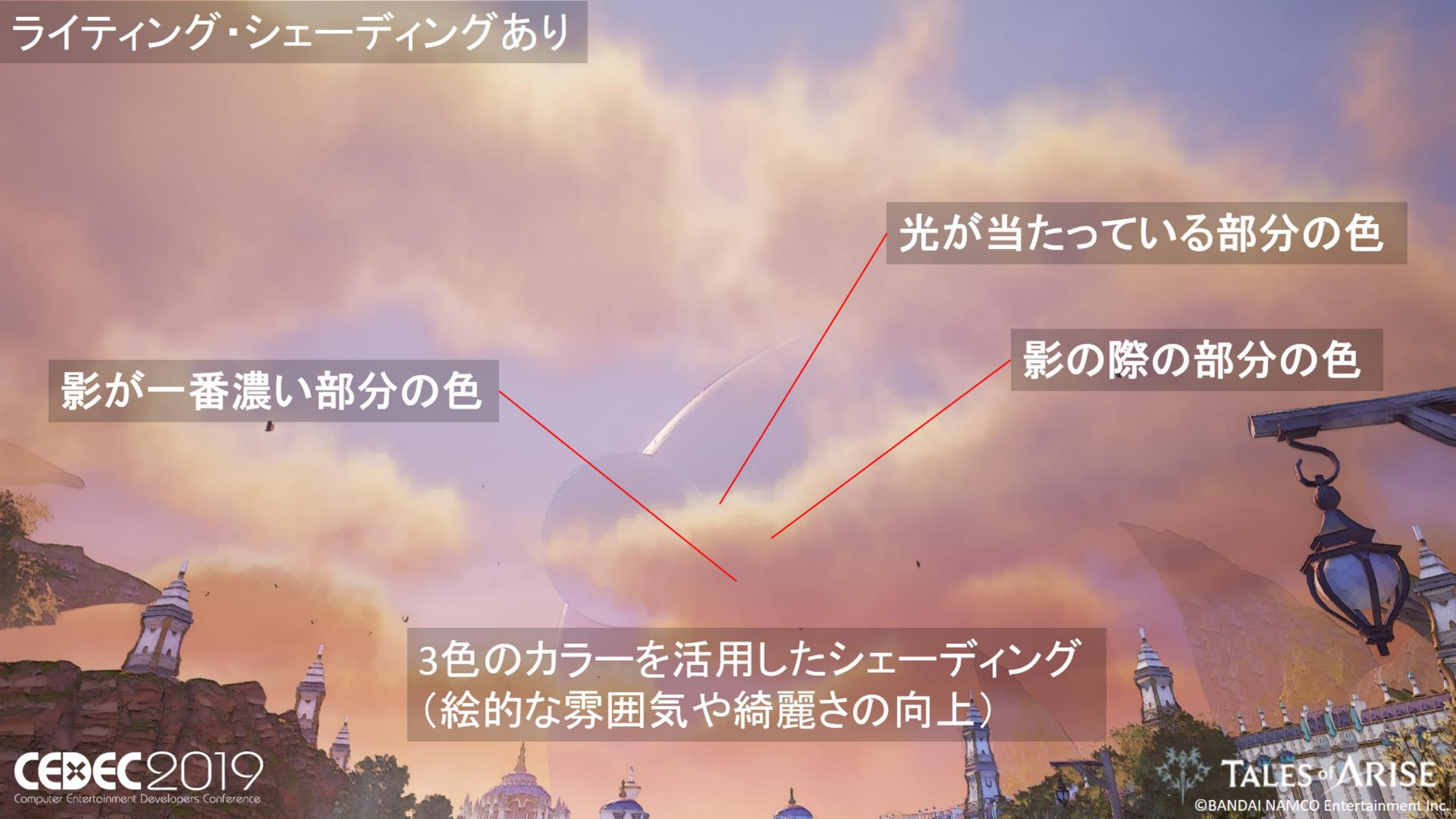
雲描画の
レイマーチ

xy: 1/3*1/3
3 Tex/Sample






ライティング・シェーディングなし
CEDEC2019
Computer Entertainment Developers Conference
TALES of ARISE
©BANDAI NAMCO Entertainment Inc.



雲描画のGPU負荷（PS4）

AsyncCompute	画面上半分（カメラ水平）	画面全体
On	0.5ms	1.0ms
Off	0.9ms (Shadow:0.3ms, Main: 0.6ms)	1.8ms

5. キャラクタのライティング・シェーディング

ARISEのキャラシェーダ

アルベドのみ

ARISEのキャラシェーダ

アルベドのみ

ARISEのキャラシェーダ

CEDEC2019
Computer Entertainment Developers Conference

TALES of ARISE

今作のキャラシェーダ表現

- ライティングをベース＋イラスト調の文法を取り入れる
  - 大人っぽさと実存感の表現
    - スカルプトをベースにしたノーマルの導入・調整（服・金属）
    - テクスチャの描きでの絵作りからライティングを考慮したアセット調整へ
  - イラスト調の文法：陰影の制御と色変化の豊かさ

材質ごとの陰影制御
（肌、髪、衣服、金属、瞳）

色変化・陰影の補強
材質感の補強
（金属の反射など）

イラスト感の向上
人形っぽい立体感抑制

材質ごとにそれぞれ別シェーダとして実行
（デプスシェーダマスクの活用）

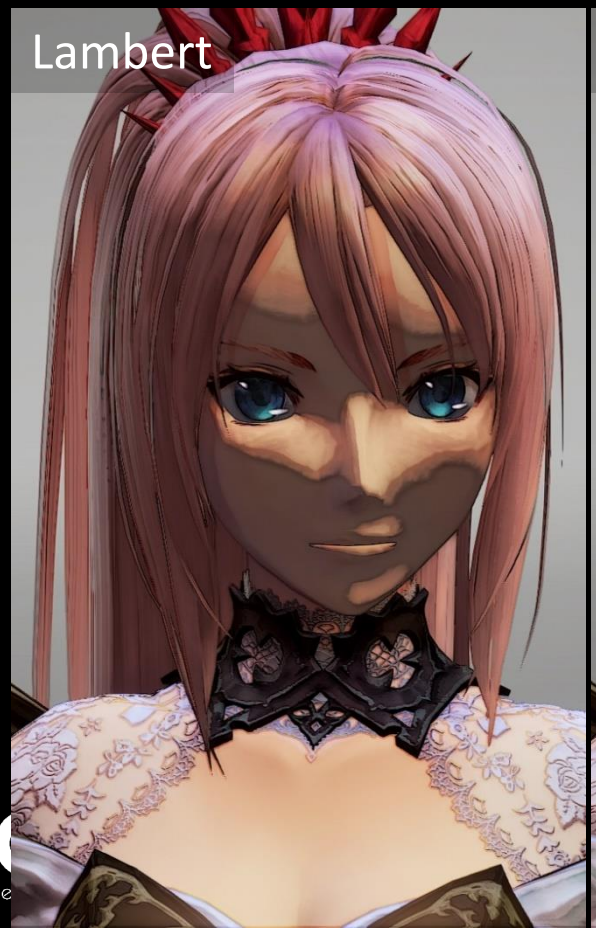
色変化の補強（シェーディングパス）

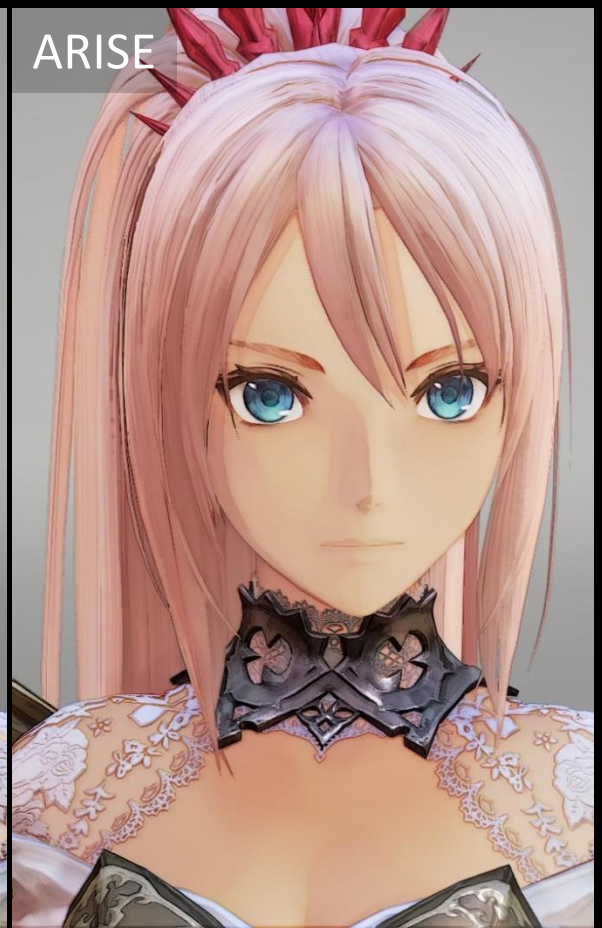
- ライティング結果に対してシェーダ的な色変化を適用
  - カラーLUTを適用（材質ごとに異なる）

以降は
今作で特に重要な
肌（顔）と金属に
関して説明します

肌シェーダのライティング

- 嫌な陰影を落ちにくくすることを重視した実装（特に顔）
  - 見た目 + アーティストの法線調整のコストを下げる

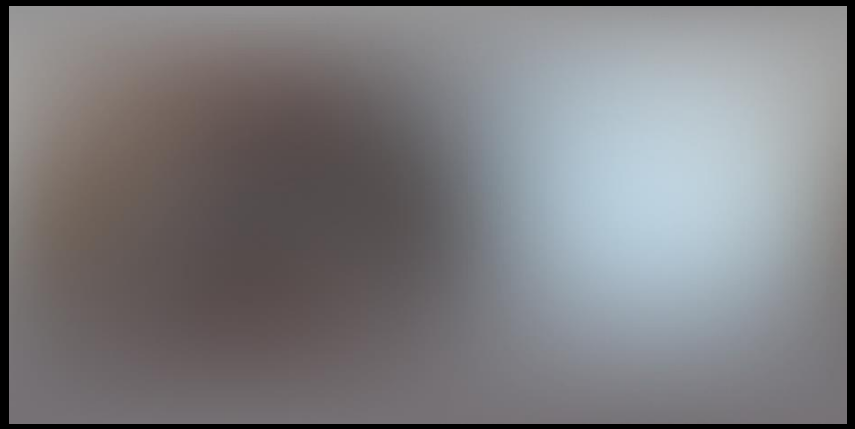




主光源が
上から落ちる場合

肌の陰影の制御

- キューブマップに補助ライト込みのライトをベイク
  - 光源の光軸に合わせて回転
    - 明るい面を光源の方向に合わせる
  - 方向を伴った陰影の補正
  - 明度とコントラストはアーティスト調整
    - 肌と瞳：コントラスト弱め
    - UE4だとエディタ上で簡単に変更可能



肌の陰影の制御

- 肌のみ光源のz方向を0にスケール
  - 主光源は上方向から来ることが多い
  - 顎周りに嫌な陰影を落ちにくくする
  - 他の材質との違和感は基本的に無い
- シャドウマップの強度
  - 嫌なセルフシャドウを抑える
  - アーティストが調整
    - 環境との馴染み
    - 肌、衣服など材質ごとの馴染み

顔周りの情報量の補強

- 顔周りは情報量・コントラストが減りがち
  - 前髪と首周りの動的な落ち影を追加
- 前髪の落ち影
  - 髪型の変更 + 髪の動きに対応
  - スクリーンデプスをレイマーチ
- 首周りの落ち影
  - 専用のシャドウボリューム
  - デプス比較のソフトシャドウ

顔周りのライティング
（ライトを回転させて比較）

顔周りのライティング
（ライトを回転させて比較）

顔周りのライティング
（ライトを回転させて比較）

顔周りのライティング
（ライトを回転させて比較）

フィルタ+輪郭線の適用

- 背景と似たような描画フロー
  - キャラは情報量をより維持しやすい調整
- 主な目的
  - 顔周りのヌルっとした立体感を抑える
  - 手描きのイラスト感の向上
- キャラ全身に適用


フィルタ + 輪郭線 なし
CEDEC2019
Computer Entertainment Developers Conference
TALES of ARISE
©BANDAI NAMCO Entertainment Inc.


フィルタ + 輪郭線 あり
CEDEC2019
Computer Entertainment Developers Conference
TALES of ARISE
©BANDAI NAMCO Entertainment Inc.

フィルタ + 輪郭線 なし

フィルタ + 輪郭線 あり

キャラの金属表現

- キャラ表現において重要な要素（鎧、武器など）
- リフレクションIBLを前提とした処理
  - キューブマップはマップごとに固定
- 開発途中で生じた課題
  - 昼夜における馴染み
  - キャラの他の材質との馴染み（明るさ）
  - 光源方向の変化によるコントラスト表現
- 実存感（金属っぽさ）を維持しつつフェイクの表現へ

キャラ金属のライティング

- ライティングは基本的に衣服などと同じ
  - ベースカラー(Diffuse)やスペキュラを考慮したライティング
  - 他の材質との明るさの馴染みやすくする
  - 主光源方向からの明暗のコントラスト
- IBLを金属っぽさの補正に利用

```
MetalLighting = Lighitng*GetNormalizedReflection(Normal, Roughness);
```

金属の全体的な明るさを維持しつつ金属っぽさを足す

IBL(Roughness)/IBL(MaxMip)

- SSGIのカラーでローカルな反射色を補正

リフレクション補正 あり

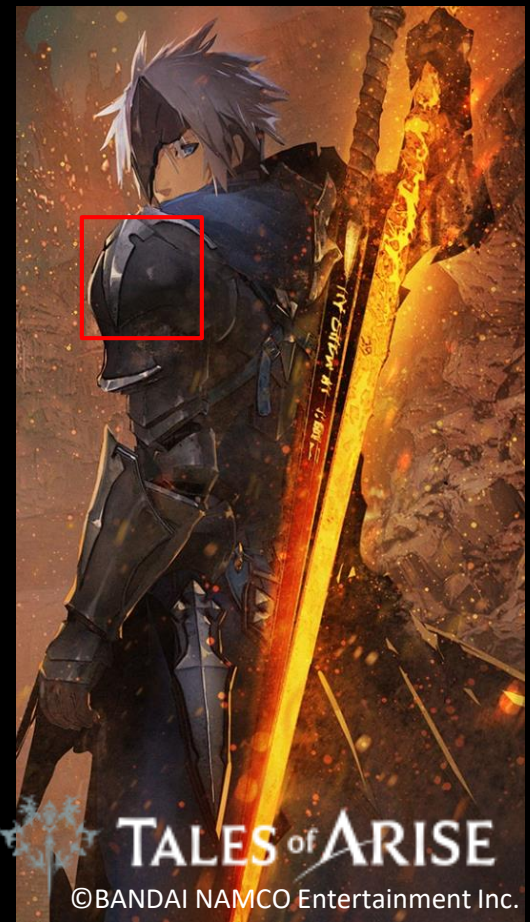
リフレクション補正 なし

SSGI補正 あり

SSGI補正 なし

金属のスペキュラ(ラフネス)オフセット

- 直接光のハイライトの鋭さ(と強さ)を調節
  - リフレクションラフネスは粗く、ハイライトは鋭く



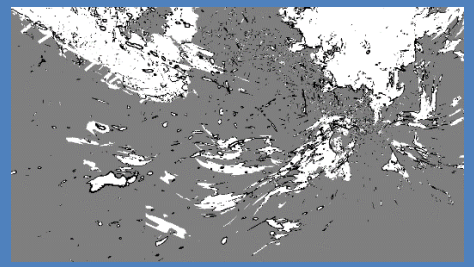
6. 半透明描画の高速化

Mixed Resolution Rendering

- 二つの解像度を組み合わせた半透明描画
  - 描画面積（半透明の重なり）による負荷を下げる

半透明描画
（縮小バッファ）

変化の大きい
箇所を検出
（縮小バッファ）



シーン合成＆
ステンシルマスク
（画面解像度）

半透明描画
（画面解像度）

  - 変化の大きい箇所のみ画面解像度で描画
    - シャープな情報を維持
  - 別バッファ＆シーンバッファの混合描画⇒ESRAM対策（XboxOne）

Mixed Resolution Rendering

半透明のGPU負荷（PS4）：6.1ms

Full Resolution Rendering

半透明のGPU負荷（PS4）：11.5ms

全体内容まとめ

まとめ

- Lighting + Shading + Filterというアプローチ
  - シリーズ最新作として新たな品質 + 手描きの水彩イラスト感の表現
- 非同期コンピュートの積極利用による負荷削減
- レイマーチをベースにした描画を活用
  - フォグ表現、雲描画、ライティングなど
- HiZやz圧縮などGPUの固定機能も非常に強力

ご清聴ありがとうございました!!

Special Thanks

- アートディレクター
  - 岩本 稔
- 背景アーティスト
  - 梶原 優子、中村 基典、土佐 香織
- キャラクタアーティスト、キャラクターTA
  - 小林 美由紀、新井 誠馨
- モンスターアーティスト
  - 井爪 広樹、川口 徹
- 株式会社バンダイナムコエンターテインメント

参考文献

- Ramy El Garawany, 2016, Deferred Lighting in Uncharted 4, ACM SIGGRAPH 2016
- Emil Persson, 2007, Depth In-depth
- Cyril Soler & Olivier Hoel & Frank Rochet, 2010, A Deferred Shading Algorithm for Real-Time Indirect Illumination, ACM SIGGRAPH 2010
- Tobias Ritschel & Thorsten Grosch & Hans-Peter Seidel, 2009, Approximating dynamic global illumination in image space, Symposium on Interactive 3D Graphics and Games 2009
- Peter Sikachev & Nicolas Longchamps, 2014, Reflection System in Thief, ACM SIGGRAPH 2014
- Michele Giacalone, 2016, Screen Space Reflections in The Surge
- Yasin Uludag, 2013, GPU Pro 4: Hi-Z Screen-Space Cone-Traced Reflections
- Nathan Vos, 2014, GPU Pro 5: Volumetric Light Effects in Killzone: Shadow Fall
- Bartłomiej Wroński , 2015, GPU Pro 6: Volumetric fog and lighting
- Benjamin Glatzel, 2014, Volumetric Lighting for Many Lights in Lords of the Fallen, Digital Dragons 2014
- Andrew Schneider, 2015, The real-time volumetric cloudscapes of horizon: Zero dawn, ACM SIGGRAPH 2015
- Andrew Schneider, 2016, GPU Pro 7: Real Time Volumetric Cloudscapes
- Andrew Schneider & Nathan Vos, 2017, NUBIS: Authoring Real-Time Volumetric Cloudscapes with the Decima Engine, ACM SIGGRAPH 2017
- Sebastien Hillaire, 2016, Physically based Sky, Atmosphere and Cloud Rendering, ACM SIGGRAPH 2016
- Stephen McAule, 2018, The Challenges of Rendering an Open World in Far Cry 5, ACM SIGGRAPH 2018
- Jeremy Shop, 2009, Mixed Resolution Rendering, GDC 2009
- Padraic Hennessy, 2016, Mixed Resolution Rendering in Skylanders: SuperChargers, GDC 2016