ビデオゲームコンテンツにおけるシーンの時間変化に応じた色指定に基づくトゥーンシェーディング手法の提案

塩路彩夏 [1]　鈴木雅幸 [2]　菅野昌人 [2]　山口翔平 [2]　川島基展 [3]　三上浩司 [3]

概要：商業アニメ制作において、色指定の工程は線画に着色する色を管理する役割を持つ。色指定の工程では担当者がシチュエーションごとにカラーパレットを作成し、作成したカラーパレットをもとにカットごとの色を決定する。しかし、近年のビデオゲームでは、キャラクタの位置や時間帯、天候などが動的に変化することが多い。そういった作品においては、アニメ制作同様に作成したカラーパレットと対応させてシチュエーションに応じた着色を行うことは難しい。そこで本研究では、ゲーム内の太陽の位置から時刻を特定し、アニメ制作の色指定工程に即しつつも時刻に応じて適正に着色できるトゥーンシェーディング手法を提案する。

Time-Dependent Dynamic Colorization for Toon Shading Based on Color Design

AYAKA SHIOJI[†1]　MASAYUKI SUZUKI[†2]　MASATO KANNO[†2]
SHOHEI YAMAGUCHI[†2]　MOTONOBU KAWASHIMA[†3]　KOJI MIKAMI[†3]

1. はじめに

近年、漫画やアニメを原作とするビデオゲーム作品が多く制作されている．このようなビデオゲーム作品ではトゥーンシェーディングを行い，漫画やアニメのルックを表現している．

図 1 は商業アニメ制作における色彩設計のワークフローである[1][2]. キャラクタの色はシチュエーションごとに細かく指定を行う．この工程を色指定といい，カットのシチュエーションに応じて，キャラクタに対し指定した色で着色を行う．このように彩色以前の工程で色彩設計と色指定を行うことにより，シチュエーションに応じたデザイン性のある彩色表現を行っている．

一方，現在のビデオゲームで用いられている主要なトゥーンシェーディング手法は光源のベクトルとモデルの表面法線から計算された内積の値をもとにモデルの色を塗分けている．このような手法では光源の向き以外の情報を用いてマテリアルの色を連続的に変化させることは不可能である．そのため，オープンワールドなどシチュエーションが連続的に変化する従来のビデオゲーム作品では，シチュエーションに対応した色の変化を行うことができず，商業アニメ制作の色彩設計および色指定に基づくシェーディングが行われていない．

本研究では，アニメ制作のワークフローに則った，屋外のシチュエーションにおける時間帯の変化に応じたトゥーンシェーディング手法を提案する．

図 1 アニメ制作における色彩設定のワークフロー

1 東京工科大学大学院　バイオ・情報メディア研究科
Graduate School of Bionics, Computer and Media Science, Tokyo University of Technology
2 株式会社バンダイナムコスタジオ
Bandai Namco Studios Inc.
3 東京工科大学
Tokyo University of Technology

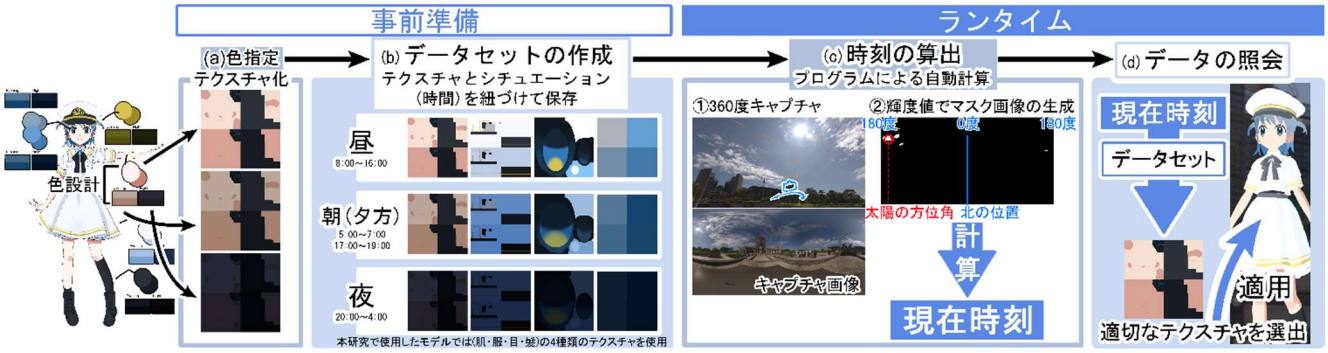

図 2　提案手法のワークフロー

2. 関連研究

これまでにも，光源ベクトル以外の情報を使用したトゥーンシェーディングの研究が存在した．Barla らの研究[3]では，従来のトーンに対応した 1D テクスチャに縦軸を追加することで，光源ベクトル以外の深度やスペキュラーなどの情報を反映したトゥーンシェーディングが可能となった．また，縦軸の情報はユーザーが自由に設定することが可能であるため，トゥーンシェーディングにおける表現の幅が広がった．

トゥーンシェーディングの研究は X-toon[3]のようによりルックを狙い通りに制御するための手法だけでなく，主光源以外の光源の影響を反映したレンダリングを行うための研究も盛んである．Moon ら[4]はレイトレーシングの結果をトゥーン調でレンダリングするための手法を提案した．また，West ら[5]は物理ベースレンダリングのリアルなライティングの特徴を有したノンフォトリアリスティックレンダリングを行うための手法を提案した．West ら[5]の手法ではトゥーンシェーディングだけでなく，クロスハッチングやカラーマッピングなどの表現も可能である．

これらの手法では，1 つの光源ベクトルのみに対応したトゥーンレンダリングではなく，主光源以外の照明や間接光の影響を正確に反映することが可能である．また，連続的な光源情報の変化にも対応することが可能である．しかし，前章で示したとおり，アニメ制作におけるワークフローでは，キャラクタの色の彩色とライティングの処理は別工程であり，彩色ではキャラクタの色を色指定どおり正確に着色する必要がある．このようなフローはこれまでに挙げた手法では実現することができない．

このことから，本研究では，アニメ制作の色彩設計および色指定のワークフローに基づくトゥーンシェーディングにより，設定したシチュエーションどおりに彩色することができる手法を提案する．

3. 提案手法

本手法はビデオゲーム内の時間帯を算出し，算出した結果とシチュエーションごとに色指定を行った情報から適切な色を出力する．

前述したとおり，アニメ制作において，キャラクタの色は設定した色指定どおりに着色を行う必要がある．また，一般的に，トゥーンレンダリングを用いたアニメ制作では，色指定に基づいてベースカラーや影色のテクスチャを複数枚作成し，それを用いてトゥーンシェーディングを行うことで，指定どおりの着色を実現している[6]．そのため，本研究では [3]のように 2D テクスチャを用いるのではなく，モデルのカラーテクスチャを用いた手法を提案する．

また，ビデオゲームではダイナミックに時間が変化するため，色も時間に応じて変化させる必要がある．しかし，すべての時刻に対し，色指定を行うことは現実的ではない．このことから，本研究では，いくつかのシチュエーションのみでの色指定を行い，そのシチュエーション間では色が遷移するように実装を行う．加えて，本研究では，晴天時の屋外というシチュエーションに限定する．晴天時の屋外において，色指定は時間によって決定する．例えば，図 2 提案手法のワークフローでは朝(夕方)，昼，夜の 3 種類のカラーパレットが設定されており，彩色を行うシーンが夕方であったことから，対応する夕方の色で彩色が行われている．ビデオゲームで実装するにあたり，現在時刻は太陽の位置から計算し，算出された時刻に応じて対応するカラーパレットを特定する．これにより，リアルタイムで精度の高い時刻の算出し，色の出力が可能となった．

先述のとおり，本研究では，ビデオゲーム内の時刻に応じて色指定どおりに彩色を行う．まず，色指定によるカラーパレットとそれらのデータセットを作成する．次に，ビデオゲーム内の太陽の位置から現在時刻を求める，そして，算出された時刻から適切なカラーパレットを求め，キャラクタに彩色する．本手法のワークフローを図 2 に示す．手法は大きく 4 つの工程に分かれている．

3.1 色指定とデータセットの作成

まず，色指定ごとにテクスチャの作成を行う（図 2(a)）．モデルや仕様に応じて制作を行う．本手法ではマテリアルごとにベースカラーと影色とを 1 枚にまとめており，肌・

図 3　色の出力の仕方

服・目・髪の 4 種類のマテリアルを使用している．

つぎに，作成したテクスチャと時刻、シチュエーションを組み合わせて整理し，1 つのデータの集合として管理を行う図 2(b))．本論文では，このデータの集合を「データセット」と呼ぶ．このとき，「データセット」に入力するデータは「シチュエーション名」・「シチュエーションの開始時刻」・「シチュエーションの終了時刻」・「作成した全てのテクスチャマップ」の 4 つであり，必要なシチュエーション分の入力を行う．

ここで，「シチュエーションの開始時刻」と「終了時刻」について述べる．先述のとおり，アニメ制作では色指定どおりの色を出力することが重要である．シーンに合わせて適切な色が出力されることで，洗練された色彩表現が可能である．しかし，時間の経過があるビデオゲームでは，つねに太陽の位置に変化が生じる．そのため，つねに同じ色を出力するだけではシチュエーションに対応した表現とは言えない．そこで，このビデオゲームにおけるリアルタイムの変化と色指定どおりの色の表示を両立するために，指定した色を表示する期間を定め，各シチュエーションの間はテクスチャマップをブレンドさせ，色の遷移を行う（図 3)．このように，「シチュエーションの開始時刻」と「終了時刻」を設定し，その間の色を遷移させることで，設定された時間内では色指定どおりの色が出力され，リアルタイムのシチュエーションの遷移に対応した表現を行うことが可能である．また，シチュエーションなどは自由に設定することが可能であるため，アーティストが意図した通りの画づくりが可能である．

3.2 現在の時刻の計算手法

続いて，設定した時間に対応させるため，現在の時刻を算出するための方法について述べる．

時刻は太陽の位置と大きな関係があり，任意地点における任意時刻の太陽の高度hと方位角Aには以下の関係式が成り立つ[7]．ただし，δは太陽の視赤緯，Hは時角，φは場所の緯度である．

$$\cos h \sin A = -\cos\delta \sin H \quad (1)$$

$$\cos A = \cos\varphi \cos\delta \cos H \quad (2)$$

$$\sin h = \sin\varphi \sin\delta + \cos\varphi \cos\delta \cos H \quad (3)$$

ゲーム開発におけるプロジェクトシーン内において，360 度キャプチャを行い，太陽の方位角を求めることで(1)~(3)式から時間を求めることが可能である（図 2(b)）．

方位角とは，北から東回りに測った角度である[8]．これを踏まえ，パノラマ画像から太陽の方位角を求める．まず，プロジェクトシーンをキャプチャしたパノラマ画像から太陽のマスク画像を作成する．続いて，そのマスク画像から得られた太陽の中心位置と画像内の北の位置をもとに，太陽の方位角を算出する（図 2(b)）．このとき，プロジェクトシーンをキャプチャする際に，プロジェクトシーンにおける北の位置をパノラマ画像の中心位置にすることで，画像内における太陽の位置のみで計算を行うことが可能である．また，太陽のマスク画像は画像の輝度値を利用して作成を行う．パーセンタイルを利用し，太陽光のマスク画像を作成する[9]．作成したマスク画像から輝度値の重心を求め，太陽の中心位置のx座標を特定する．輝度値の重心は，位置が(x,y)の画素の輝度値を$array(x,y)$とすると，(4)式で算出される．

$$重心のx座標 = \frac{\sum_{xy}(array(x,y)\cdot x)}{\sum_{xy}(array(x,y))} \quad (4)$$

そして，求めた太陽の方位角および年月日，場所の緯度から(1)~(3)式を用いて時刻の算出を行う．

ここで，太陽の視赤緯δ，時角Hの計算を(8)式，(9)式のように簡略化させる[10]．また，求めたい時刻をt_h，計算対象年，月，日をそれぞれ$year$，$month$，day，1 月 1 日から経過した日数を$nday$とする．

$$n = year - 1968 \quad (5)$$

$$M = \frac{360}{365.2596}\left\{nday - \left(3.71 + 0.2596 \times n - \left[\frac{n+3}{4}\right]\right)\right\} \quad (6)$$

$$\varepsilon = 12.3901 + 0.0172\left(n + \frac{M}{360}\right) \quad (7)$$

$$\sin\delta = \cos(\varepsilon + v)\sin\delta_0 \quad (8)$$

$$t = 15(t_h - 12) \times 15 \quad (9)$$

このとき，時刻は太陽の動きに合わせて連続的に変化し，カラーパレットの判定に用いることができればよいため，計算を簡略化することによる誤差は問題とならない．

3.3 トゥーンシェーディングによる彩色

3.2 章で求めた時刻をもとに，3.1 章で作成したデータセットを参照し，適切なテクスチャを選出する．このとき，

現在時刻がシチュエーションとして設定した時間帯の中であった場合はそのシチュエーションでのテクスチャを適用すればよい（図 4(b)）．しかし，現在時刻に設定されたシチュエーションがなかった場合，前後のシチュエーションで設定されたテクスチャを取得し，ブレンドさせる（図 4(a)）．テクスチャのブレンド率は「シチュエーションの開始時刻」と「終了時刻」，現在時刻に応じて変化させ，シチュエーション間が滑らかに遷移するように実装を行う．

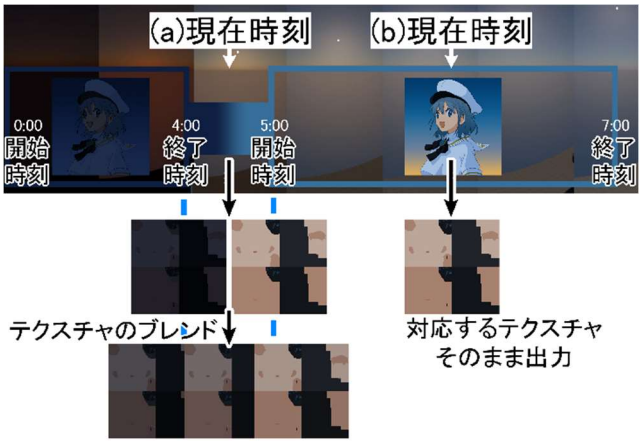

図 4　テクスチャの取得と出力について

4. 実用検証

本章では，提案手法の実証とその結果について述べる．

本研究では，現在時刻の計算が問題なく行われていること，テクスチャの色は正しく出力できているか，およびリアルタイムに対応できているかの 3 点について検証を行った．

4.1 Unreal Engine での実装

本節ではゲーム開発に利用されるゲームエンジンのひとつである Unreal Engine 5[11]にて実装を行った結果について述べる．

図 5 は Unreal Engine5 にて実装したデータセットである．本研究では，シチュエーションを Morning, Daytime, Evening, Night の 4 種類とし，それぞれ 5 時~6 時，7 時~16 時，17 時~19 時，20 時~翌 4 時が対応する時間帯となる．

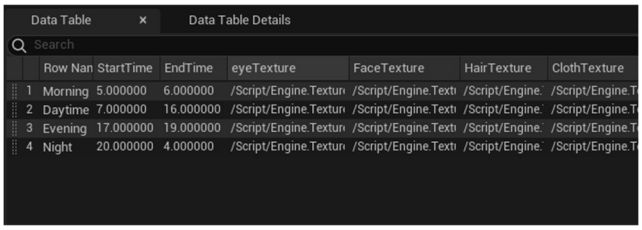

図 5　データセット

マテリアルは基本的なトゥーンマテリアルを実装し，色は Emissive Color で出力している（図 6）．時刻とデータセットをもとに，キャラクタのブループリント内でテクスチャの特定および，テクスチャのブレンド率の計算を行っており，取得したテクスチャとブレンド率をマテリアルへ送っている（図 6(a)）．光源のベクトルとモデルの法線の内積の計算（図 6(b)）によりベースカラーと影色の塗分けを行い，その結果を出力する（図 6(c)）．Base Color で出力を行った場合，光源の色や明るさの影響を受けてテクスチャの色が変化してしまうため，それら影響を受けない Emissive Color で出力を行うことで，色指定どおりの色の出力を行っている．

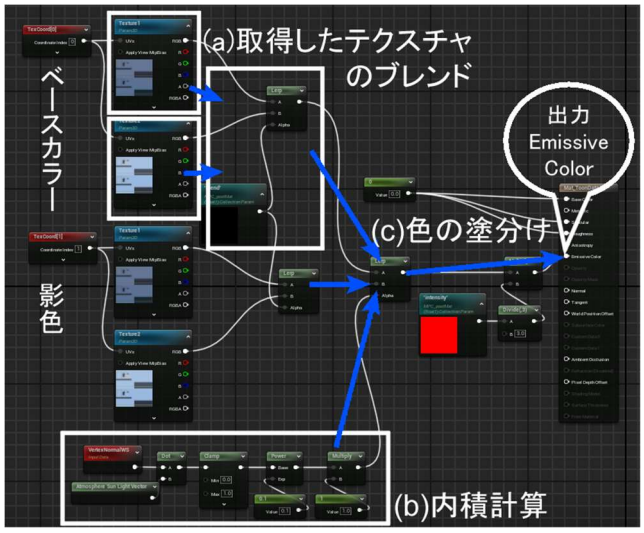

図 6 実装したマテリアルのノード

4.2 時刻を算出するプログラムおよび色の出力の検証

本節では 3.2 章, 3.3 章の手法が問題ないか検証を行った．検証には TrueHDRI プロジェクト[12]により作成された HDRI 画像である TrueHDRI を用いた．TrueHDRI は正しい輝度の情報および，正確な撮影日時，緯度経度の情報を持っているため，本研究のプログラムの精度を正確に検証することが可能である．検証で使用した TrueHDRI 画像は表 1，図 7 のとおりである．

番号	場所	撮影日時
1	恩納村①	2024 年 01 月 01 日 07:34
2	恩納村②	2024 年 01 月 01 日 14:05
3	お台場	2020 年 12 月 31 日 14:10
4	東京ヘリポート	2019 年 08 月 17 日 18:45

表 1 検証で利用した TrueHDRI データ概要

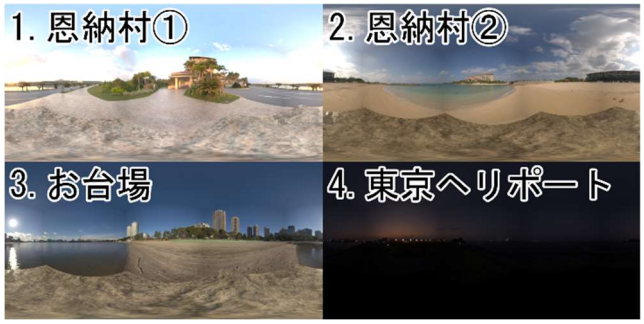

図 7　検証で利用した TrueHDRI 画像(パノラマ画像)

また，出力されている色の確認は RenderDoc[13]を用いて

行う．彩色はコンポジットより前の工程であることから，ポストプロセスが影響していない段階での色の確認を行う．

検証は以下の 3 点に着目して行った．

① 算出した時刻の誤差
② データセットから正しくテクスチャを取得できているか
③ 出力した色の誤差

表 2 は評価項目①・②の結果である．時刻は算出した時刻を示しており，誤差は撮影時間と計算された時刻との誤差である．また，テクスチャは取得したテクスチャのシチュエーションである．時刻が設定されたシチュエーションの間でない場合，その前後のテクスチャを取得しブレンドしている．表では取得した 2 枚のテクスチャとそのテクスチャをブレンドした割合を示している．一番右の列は，取得したテクスチャの内容が，計算した時刻を正しく反映しているものであるかを記述しており，ブレンド率含め適切なテクスチャを取得できている場合は”〇”，できていない場合は”×”を記した．

	撮影時刻	計算時刻	誤差(分)	テクスチャ	〇/×
1	7:34	7:27	-7	Morning Daytime ブレンド率(46%)	〇
2	14:5	14:4	-1	Daytime	〇
3	14:10	13:52	-18	Daytime	〇
4	18:45	17:10	-95	Evening	〇

表 2 ①・②の結果

時刻の誤差は最大で 1.6 時間程度になる場合もあるが，大体で正しく算出できていることが分かった．また，時刻をもと正しくテクスチャを取得することができた．番号 2 では現在時刻と前後のシチュエーションの時刻をもとに適切に 2 枚のテクスチャをブレンドすることができた（図 8）．

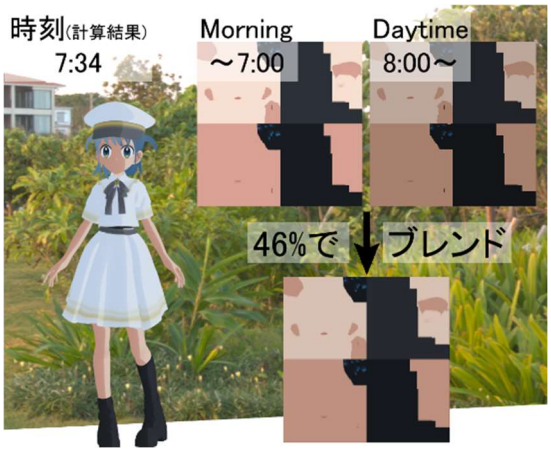

図 8 ブレンドしたテクスチャが出力された様子

つぎに，出力した色の検証を行う．比較検証には図 9 で示した 10 種類の色を使用した．図 10 はテクスチャの色と最終的に出力された色の結果の比較画像である．色の誤差の評価は，L*a*b*表色系でのユークリッド距離を用いた式さの計算結果に基づいて行う．色差ΔEは，入力したテクスチャの色の値を$L*_1, a*_1, b*_1$，出力色の値を$L*_2, a*_2, b*_2$としたとき，式(10)によって計算される．

$$\Delta E = \sqrt{(L*_2 - L*_1)^2 + (a*_2 - a*_1)^2 + (b*_2 - b*_1)^2} \quad (10)$$

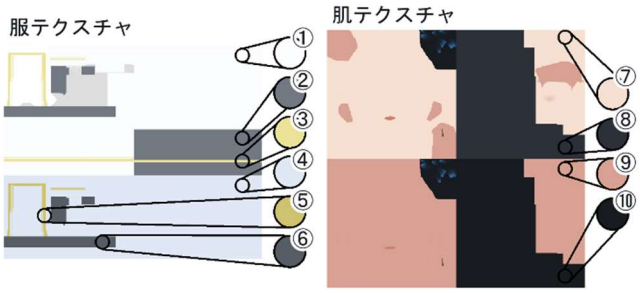

図 9 比較を行った色について

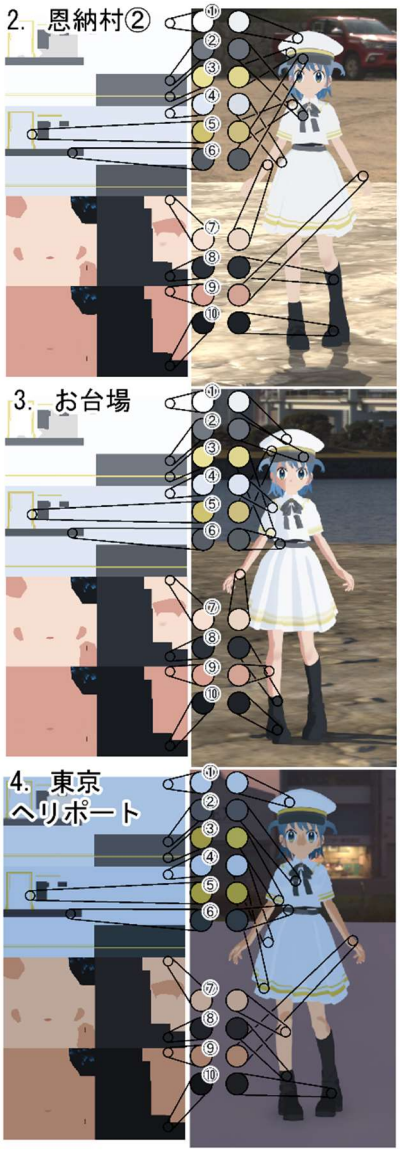

図 10 入力したテクスチャの色と出力した色の比較

図 11 コンスタントな時間変化に対応して色が遷移している様子

	①	②	③	④	⑤
2	0.063	0.013	0.099	0.149	0.098
3	0.148	0.024	0.13	0.081	0.117
4	0.072	0.011	0.102	0.076	0.102

	⑥	⑦	⑧	⑨	⑩
	0.06	0.055	0.002	0.045	0.006
	0.008	0.1	0.074	0.108	0.005
	0.017	0.051	0.002	0.062	0.006
平均色差	0.063	最大値	0.149	最小値	0.002

表 3 入出力した色の色差

表 3 は図 9 で指定した色に対して入出力を行った際に生じた色差をまとめたものである．色差は最大でも 0.149 であり，L*a*b*空間における許容範囲内であるといえる．

4.3 動的な太陽の位置の変化に対する検証

Unreal Engine 5 で動的に変化する太陽を実装し，太陽の動きに応じて色が変化しているか検証を行う．

図 11 は太陽の位置の変化と色指定に対応して，適切にトゥーンシェーディングが行われている様子である．シチュエーションとして設定を行った時間帯ではそのシチュエーションに対応したテクスチャが適用された．また，時間の設定が行われていないシチュエーション間では，2 枚のテクスチャが現在時刻に応じて適切にブレンドしている様子が確認できた．

本提案手法によって，設定したシチュエーション内では指定したテクスチャが適切に出力され，時間経過に応じてテクスチャがブレンドしながらつぎのシチュエーションに遷移している様子を検証することができた．

5. おわりに

本論文では，アニメ制作で行われている工程である，色指定と彩色に基づいたトゥーンシェーディングをビデオゲームで行うための手法を提案した．本手法により，リアルタイムで時間の変化が行われるビデオゲームにおいて，アーティストがより細かく色の制御を行うことが可能となった．

本研究では，色指定と彩色にのみ焦点を当てたが，アニメ制作ではこの後に，ライティングやコンポジットといった工程が行われる．このライティングやコンポジットでは太陽光だけでなく，関節光などの影響も付与する必要がある．また，本研究では，屋外でのトゥーンシェーディングを前提としたが，室内でもその場所に応じた色指定が行われる．これらのような，太陽光以外の照明の反映，室内での色指定が今後の課題である．

参考文献

[1] AUTODESK：アニメーションの制作工程（ワークフロー）（オンライン），入手先〈https://area.autodesk.jp/animation/workflow.html〉（参照 2024 10 22).

[2] CGWORLD.jP：『Fate/kaleid liner プリズマ☆イリヤ』シリーズが実践する、新世代のアニメ撮影ワークフロー（旭プロダクション）（オンライン），入手先〈https://cgworld.jp/feature/201512-prisma-illya-cgw208t2.html〉（参照 2024 10 22).

[3] Pascal Barla, Joëlle Thollot, Lee Markosian: X-toon: an extended toon shader, In Proceedings of the 4th International Symposium on Non-photorealistic Animation and Rendering, ACM, 127–132(2006).

[4] Andreas Bauer, Marc Salvati: The use of CG in Japanese animation. SIGGRAPH 2022 Courses, Article No.: 16, ACM, 1-207(2022).

[5] Nicholas Moon, Megan Reddy, Luther Tychonievich: Non-photorealistic ray tracing with paint and toon shading, SIGGRAPH '21: ACM SIGGRAPH 2021 Posters, Article No.: 36, 1-2(2021)

[6] Rex West, Sayan Mukherjee: Stylized Rendering as a Function of Expectation, ACM Transactions on Graphics , Volume 43, Issue 4, Article No.: 96, 1-19(2024)

[7] 天文国立台：太陽の高度，方位角および影の位置の概略値の求め方（オンライン），入手先〈https://eco.mtk.nao.ac.jp/koyomi/topics/html/topics2005.html〉（参照 2024 10 22).

[8] 天文国立台：月の高度・方位角について（オンライン），入手先〈https://eco.mtk.nao.ac.jp/koyomi/topics/html/topics1993.html〉

（参照 2024 10 22).
[9] 横山 雅来，鈴木 雅幸， 菅野 昌人， 山口 翔平， 川島 基展， 三上 浩司：ビデオゲームコンテンツに向けた現実の光を正確に再現した HDRI を用いる LookDev 環境の構築法，エンタテインメントコンピューティングシンポジウム，2024， pp.79 85 (2024).
[10] MetDS 株式会社 気象データシステム：年差を考慮した太陽位置の簡易計算（オンライン），入手先〈https://www.metds.co.jp/wp-content/uploads/2023/07/TE_Simplified_SP_230724.pdf〉（参照 2024 10 22).
[11] Epic Games, Inc.：Unreal Engine5（オンライン），入手先〈https://www.unrealengine.com/ja/unreal-engine-5〉（参照 2024 10 22).
[12] Bandai Namco Studios Inc.：TrueHDRI（オンライン），入手先〈https://www.bandainamcostudios.com/projects/truehdri〉（参照 2024 10 22).
[13]Baldur Karlsson：RenderDoc（オンライン），入手先〈https://renderdoc.org/〉（参照 2024 10 22