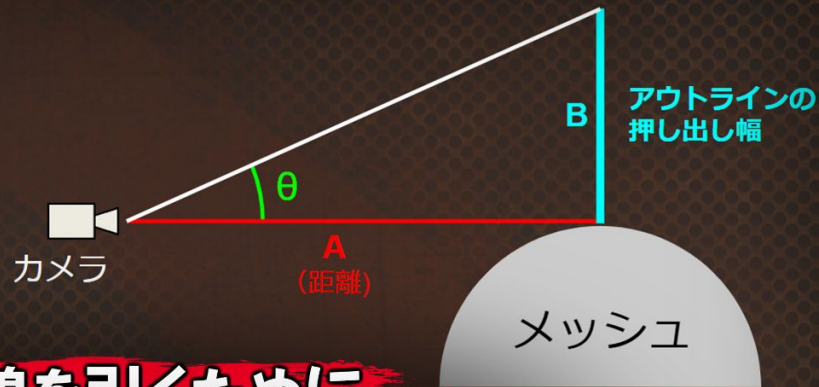
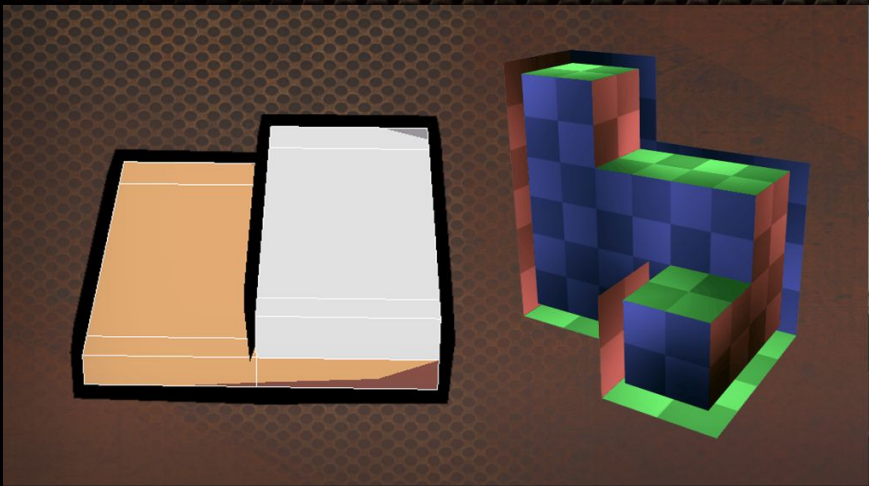




## 3Dでキレイな線を引くために ギルティギアシリーズのトゥーンライン制御テクニック



# 本日の講演について



撮影OK



SNS投稿OK



# 講演概要

## 背面法アウトラインの基礎と応用を身に着けよう

「背面法」によるアウトライン は20年以上の長い歴史のある手法ですが、現在も最前線のゲームで採用され続けている、実戦的、効果的な表現手段です。

弊社の格闘ゲームタイトル「ギルティギア」シリーズにおいても、3Dビジュアルを採用するようになってから輪郭線の描画はずっとこの手法を活用し続けています。

非常にメジャーな手法ではあるのですが、それだけにその **理屈や仕組み** をしっかりと理解しないまま使用している人も多いと思います。本セッションでは「背面法」のアウトライン描画の **基礎、その理屈や仕組み** をわかりやすく解説するとともに、そのポテンシャルを最大限活用するための **応用方法、直面しがちな問題とその解決方法** を解説したいと思います。

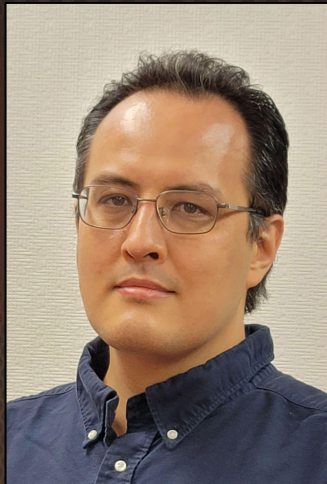
モデラーはより意図した **ビジュアルの表現** に近づくことができ、シェーダー作成者は **どのような要件を満たす必要があるのか** について理解を深められるのではないかと思います。

### 想定受講者:

- ・アニメ調のキャラクターモデリングに興味のあるモデラー
- ・アニメ調ビジュアル作成に興味のあるシェーダー作成者

# 講演者プロフィール

## 本村・C・純也



アークシステムワークス株式会社  
リードモデラー / テクニカルアーティスト / 他いろいろ

モデラー出身でシェーダーも書きたくてテクニカルアーティストに。  
モデリング・リギング・シェーダー作成、講演とかいろいろ。  
現在はテクニカルアートサポートおよびR&D、後進モデラーの育成を担当。

代表作：

GUILTY GEAR Xrdシリーズ

リードモデラー/テクニカルアーティスト

DRAGON BALL FighterZ (バンダイナムコエンターテインメント)

ディレクター/モデリング監修/テクニカルアーティスト

DNF Duel (ネクソン)

シェーダー、リグ基礎開発/モデリング監修



# 講演の流れ

PART1: そもそも輪郭線とは何か。その役割

PART2: 主要な輪郭線手法の利点と弱点

PART3: 背面法による輪郭線描画の理屈

PART4:「マルチパスシェーダー」による背面法

PART5: GGシリーズにおける背面法実装の工夫

PART6: GGシリーズにおける背面法の表現テクニック

# 最初に

## シェーダー初心者～中級者向けの内容です

本講演は背面法シェーダーの具体的なコード内容については触れません。かわりに「何が起きているのか」「どうすればそれを活用できるか」という「運用のためのノウハウ」にフォーカスします。

そのため、モデラーなど、シェーダーについては初心者の方でも問題なく理解できる内容になっていると思います。一方で、背面法を活用するための様々な課題とその解決法については、シェーダー中級者にとっても益のあるお話ができると思いますのでお付き合いいただければ幸いです。

## 特定のエンジン、3Dソフトに依存しない内容です

背面法アウトラインはこれまでも様々な3Dソフトやゲームエンジン で活用されてきました。そのため、本講演での背面法のノウハウはその大部分が特定のエンジンや3Dソフトに依存しない ものになっています。一部、ギルティギアシリーズで使用しているUnreal Engineでの実装例を紹介しますが、原理的には他エンジンでもほぼ同様の対応ができる手法になっています。

## ややハイペースで進めます

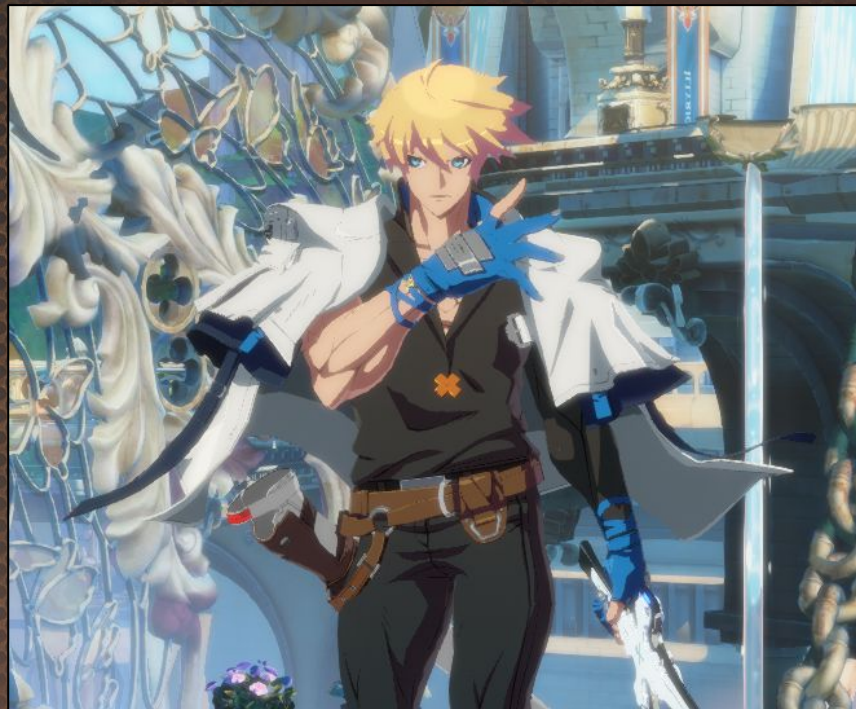
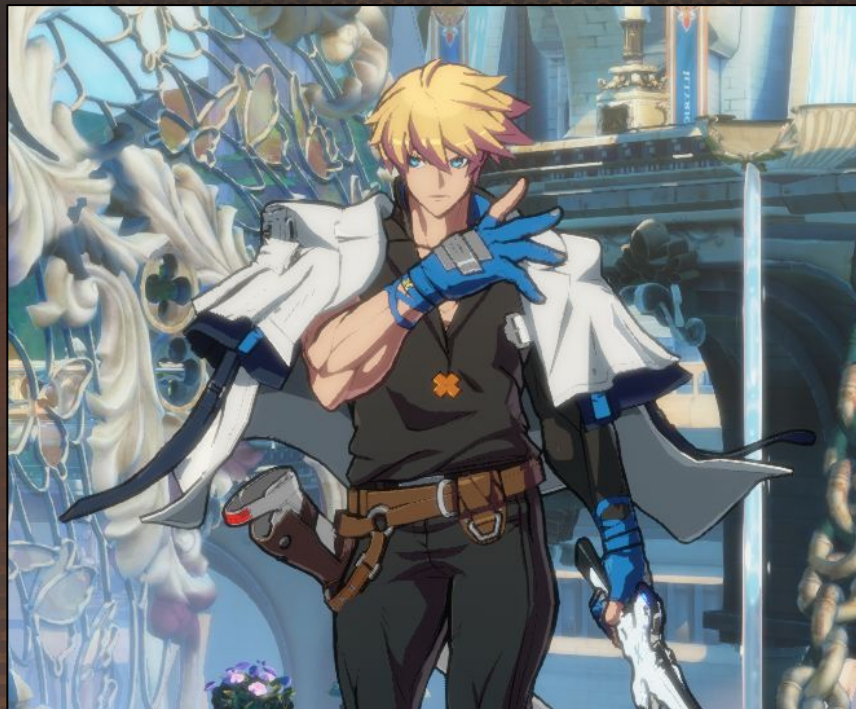
スライドの枚数が多いのでテンポよく進めます。  
資料は後日公開するのでメモなどは取らなくてもOKです。



Part 1

**そもそも輪郭線とは何か。その役割**

「輪郭線」があったら、なかったら





# 輪郭線は「AとBの識別を助けるもの」



この例では、キャラクターと背景の間に輪郭線が存在することで、「この範囲はキャラクター」「この範囲は背景」というのをよりわかりやすく見る人に伝える効果が発生しています。

キャラクターの右肩付近を見ると顕著ですが、衣服の白と背景の白とが混ざりあってしまうのを、輪郭線が防いでくれているのがわかると思います。

逆に言えば、人間の視覚には「線」によってAとBを区別するという機能が備わっているということができるでしょう。

どっちが目飛び込んでくる？

新発売!!

新発売!!



# 「輪郭線」ってそもそも何？

輪郭線は「強調」するための道具

**復活** **新発売!!**  
**優勝!!!**

日本語に「際立たせる」という言葉があるように、「キワ」を「目立たせる」ことで特定の事物を強調、誇張することができます。

雑誌や新聞の見出し、広告などでは、大きな文字に二重三重に強いコントラストの色で輪郭線をつける強調表現が多用されています。

これらはまさに「輪郭線」の持つ「強調表現」を極限まで活用したものといえるでしょう。

# まとめ



理由はさだかではありませんが人間の視覚は輪郭線に反応し、輪郭線はそれを利用し**物体の識別**をしやすくする機能を持ちます。そのため、輪郭線は**図やイラスト、文字の装飾**などあらゆる形態で活用される表現手法となりました。

当然、ゲームにおいても同様です。むしろ、「**AとBを識別**」して**瞬時になんらかの判断**をすることが多いゲームにおいては、輪郭線は特に有用な表現手段と言えます。



Part2

# 主要な輪郭線手法の利点と弱点

# 主要な輪郭線手法

## ポストプロセスアウトライン



ポストプロセスシェーダーで隣り合うピクセルと深度情報や法線情報を比較して輪郭部分を検出し、輪郭線を描写する手法

## 背面法アウトライン



頂点シェーダーなどでメッシュを一回り大きく裏向きに再描画することで輪郭線を作る手法

**本セッション**で解説するのはこっち



# ポストプロセスアウトラインの概要

ポストプロセス の一種として、**全画面**を対象に適用できる処理です。

描画済みの画面上のピクセルの一つ一つについて、**近隣のピクセル** との**深度の差** や、**法線の差** を計算し、**一定以上の差** があればそこを抽出して、輪郭線として任意の色でラインを引きます。



## メリット

- ・画面全体に対して常にほぼ **一定の負荷、品質** でかけられる
- ・カメラ距離が変わっても **線の太さを一定** に保てる
- ・法線が大きく変化する **ポリゴン同士が交差する** 部分にもラインを描画できたりする

## デメリット

- ・部位ごとの線の強弱や色などの **細かい調整が難しい**
- ・ポスト処理なので **3Dソフト上で確認、調整するのが難しい**
- ・画面全体にかかるエフェクトなので「どこにラインを描画するか、あるいはしないか」を **細かく指定するのが難しい**

# 背面法アウトラインの概要

同じオブジェクトを少しだけ膨らませて裏返したメッシュを重ねて描画することで、はみ出した部分を輪郭線として表示させる手法です。

主に**頂点シェーダー**などを使って同じメッシュを二回描画する手法を使うことが多いです。



## メリット

- ・輪郭線を出す場所、出さない場所を個別に指定しやすい
- ・3Dソフト上でプレビューするのが容易
- ・部位ごと、あるいは頂点ごとに輪郭線の太さや色を調整しやすい

## デメリット

- ・処理負荷は表示する輪郭線ありオブジェクトの数で変動
- ・線の太さを一定にたもつのに工夫が必要
- ・ポリゴン同士が交差するいわゆる「めり込み」部分を輪郭線として検出できない



# ギルティギアシリーズで背面法を選んだ理由

「GUILTY GEAR Xrd」以降のアーケシステムワークスの3D格闘ゲームでは、輪郭線表現に **背面法** を選択しています。  
その主な理由は .....

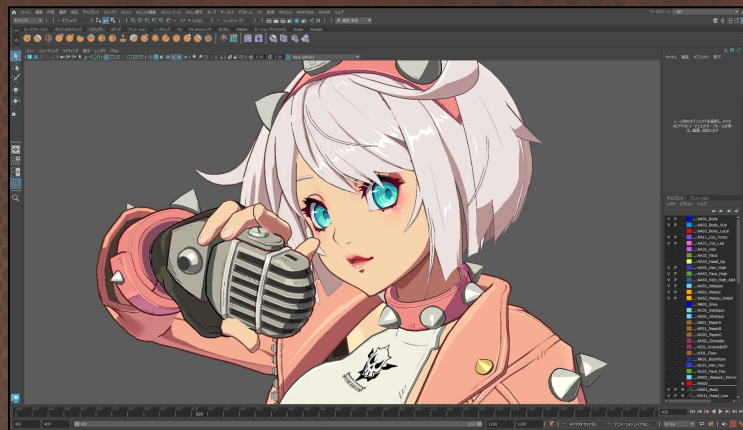
## ●輪郭線の調整による表現の幅広さ

シェーダー上の工夫により任意の部位の輪郭線を太くしたり細くしたり、色を変えたり、あるいは輪郭線を部分的にOFFにしたりできるため、目指しているアニメ調ビジュアルの再現に適していた。

## ●3Dソフト上での再現、調整が可能

モデリング、モーションなどの作業中もゲーム中と同等のビジュアルを確認しながら作業できるため、クオリティ向上をしやすいかった。

また、細かい輪郭線の調整そのものも3Dソフト内で完結する形で作りこむことができ効率が良い。



Part3

# 背面法による輪郭線描画の理屈



# 「背面法」の始まり

「背面法」が初めて登場したのは2000年に発売されたSEGA社の「ジェットセットラジオ」でした。細かな実装方法は時代によって変化していますが、その基本的な原理は20年以上変わらずに様々なゲームで活用され続けています。

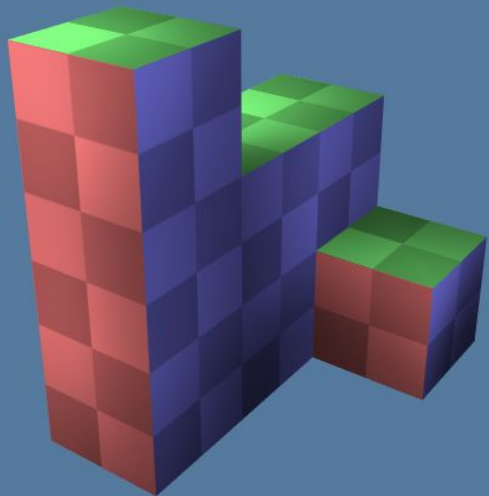


©SEGA (2000) 「JET SET RADIO」 (ドリームキャスト)

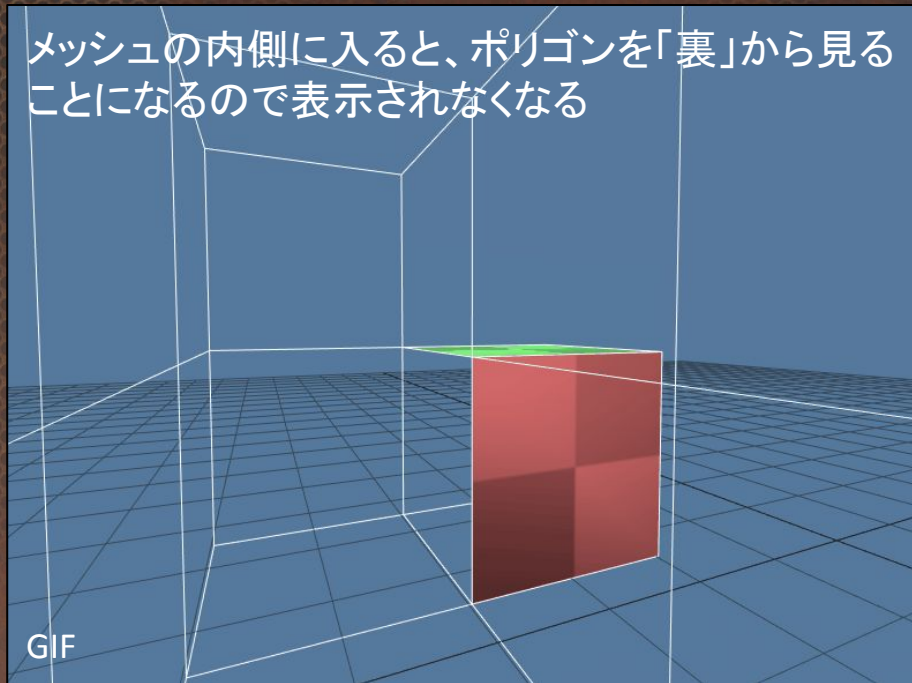
# 「背面法」を理解するための基礎知識：表裏①

ポリゴンには「表」と「裏」があります。多くの場合、ポリゴンメッシュは「表」だけを表示する設定になっています。この設定ではカメラがメッシュの内側に入ってしまった時など裏から見たポリゴンは表示されません。

メッシュの外側からはポリゴンの「表」が見える



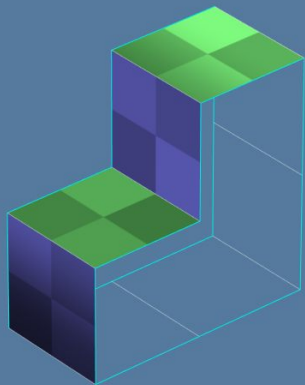
メッシュの内側に入ると、ポリゴンを「裏」から見ることになるので表示されなくなる



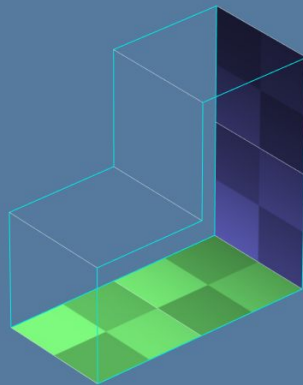


## 「背面法」を理解するための基礎知識：表裏②

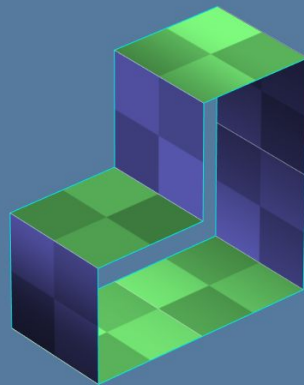
ポリゴンメッシュはシェーダー側の設定によって、「表面だけ表示」「背面だけ表示」「両面表示」の中から、どの設定で描画するかを選べます。



「表面だけ表示」では、裏を向いた面は表示されない



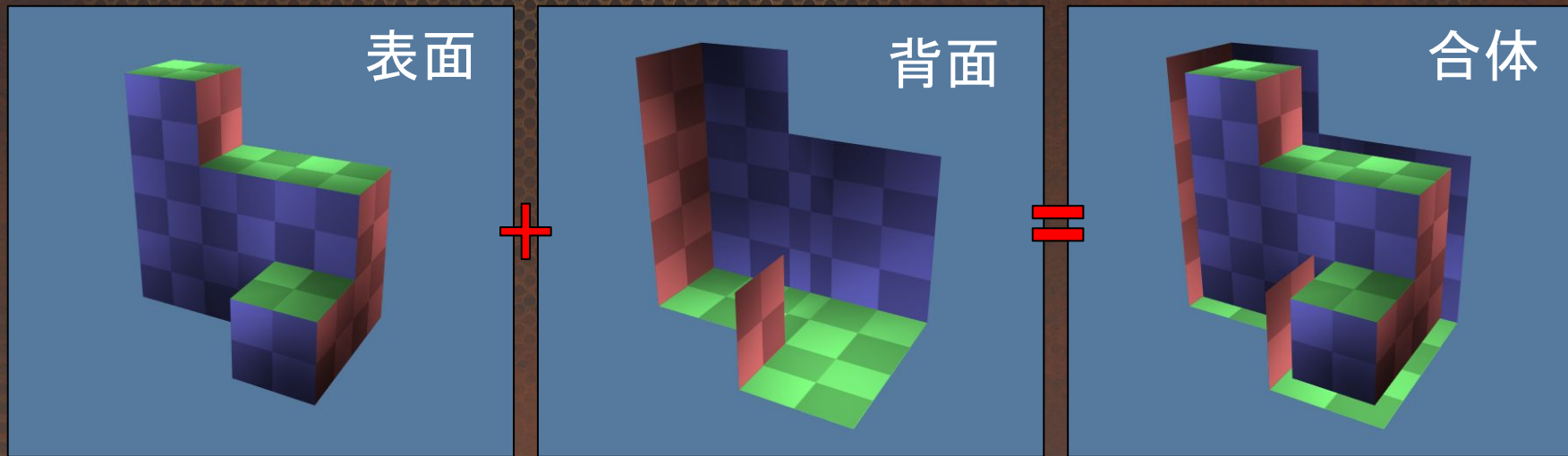
「背面だけ表示」では、裏を向いた面だけ表示される



「両面表示」では、表裏どちらの面も表示される

# 「背面法」を理解するための基礎知識：表裏③

「表面だけ表示」に設定されたメッシュと、それよりも一回り大きな「背面だけ表示」に設定されたメッシュの二つを重ね合わせると、下図のようになります。

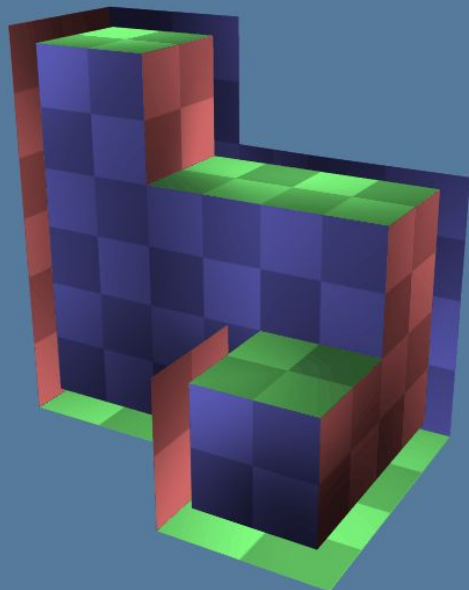


この時点で、ある程度「背面法」の理屈が垣間見えてきたのではないのでしょうか。

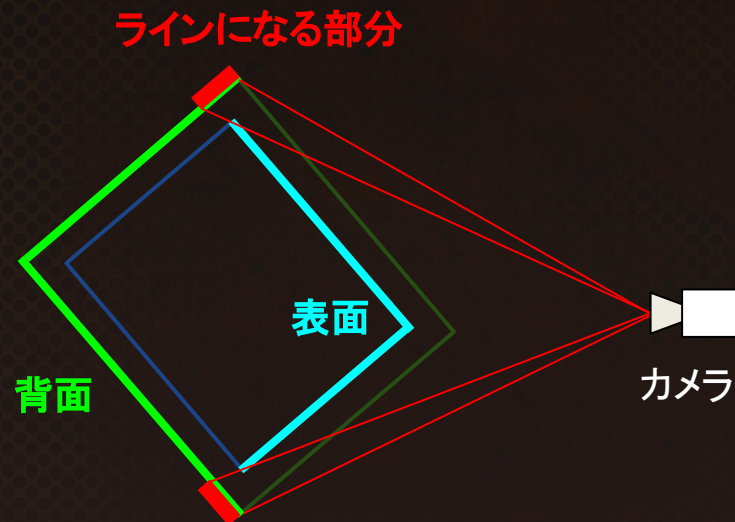


# 「背面法」を理解するための基礎知識：表裏④

「背面だけ表示」のメッシュのほうがほんの少し大きい場合、そのはみ出した分だけが余分に表示されます。このはみ出した部分を輪郭線として利用するのが「背面法」の基本原則となります。



GIF

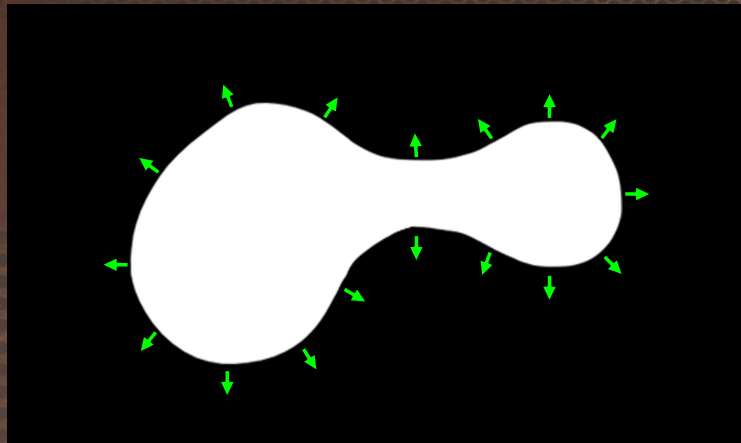


カメラから見てはみ出した部分がラインになる

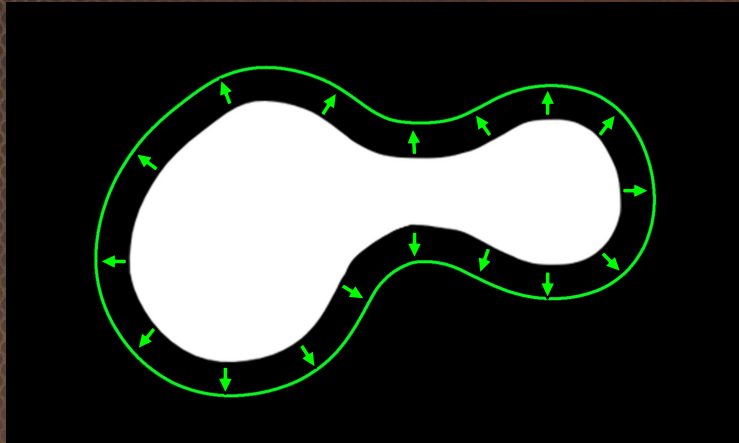
# 「背面法」を理解するための基礎知識：法線

どうやって「一回り大きなメッシュ」を得るかという、オリジナルのメッシュを「法線」の方向に一定量膨らませることで用意します。

「法線」というのは、メッシュを構成する一つ一つの頂点がそれぞれどっちを向いているのかという方向です。



メッシュを構成する頂点それぞれがもつ「向き」と考えればOK

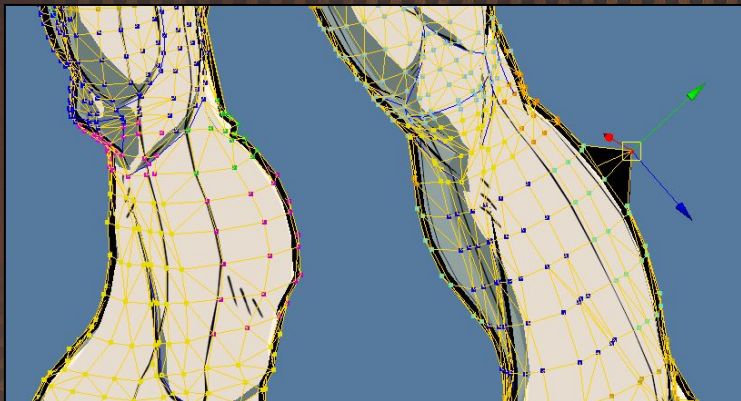


「法線方向」に少し膨らませることで、一回り大きい形を得られる

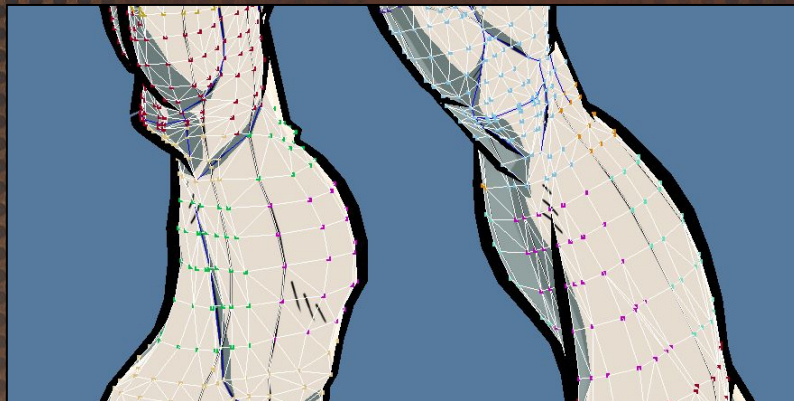


# 「背面法」のための基礎知識：メッシュ複製

一回り大きなメッシュの作り方について、3Dソフト上で手作業でメッシュを複製して加工してもいいのですが、それだとポリゴン数や管理コストが増えてしまいます。



手作業でやるとめんどくさいしヒューマンエラーも発生しがち



シェーダーの機能で膨らませて再描画したほうが描画も工数も軽い

これを解決できる「マルチパスシェーダー」という方法があるので紹介します。

Part4

# 「マルチパスシェーダー」による 背面法



# シェーダー基礎知識①

通常、シェーダーは主に「頂点シェーダー」と「ピクセルシェーダー」に別れます。

※ピクセルシェーダーはフラグメントシェーダーと呼ばれることもあります

## 頂点シェーダー

形を決める！

※概念図



そのメッシュがカメラ内でどの位置に表示されるかを計算

## ピクセルシェーダー

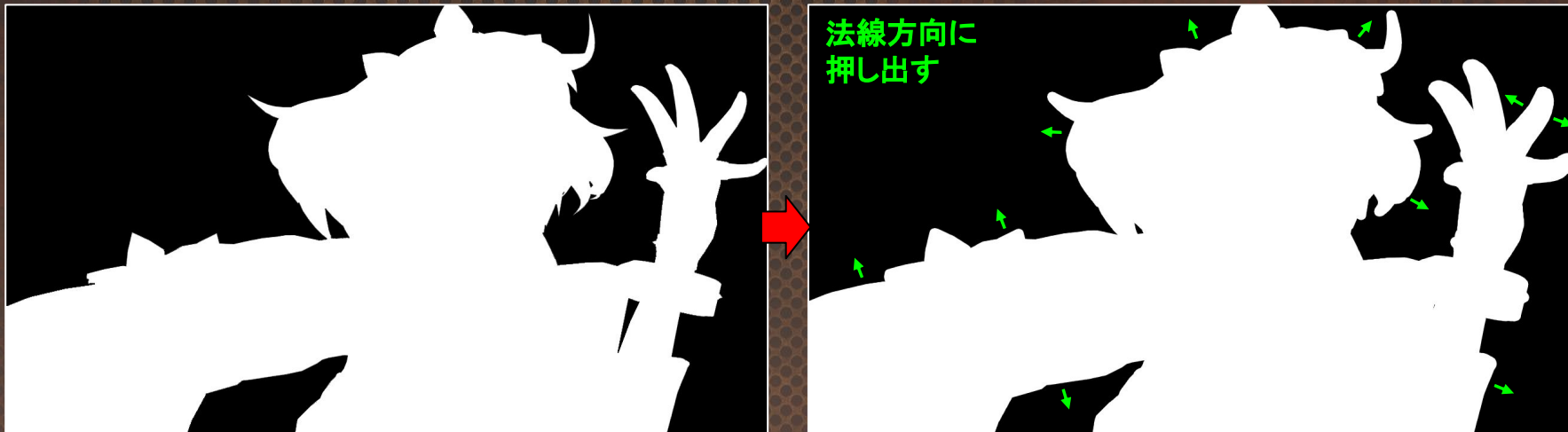
色を塗る！



ライト方向やテクスチャから最終的な色を計算

## シェーダー基礎知識②

「頂点シェーダー」内で各頂点を法線方向に押し出すようにコードを書くことで、描画するメッシュを一回りふくらませることができます。



さらにこのメッシュについては「背面のみ表示」に設定すると輪郭を構成する背面メッシュの用意ができます。



# 二つのメッシュで背面法

「中身」を描く本体メッシュと、「輪郭」を描く背面メッシュを用意して、二つを重ね合わせると、「背面法」による輪郭線付きのビジュアルが得られます。

本体メッシュ



色！



背面メッシュ



色！



合体！



でもメッシュを二つ用意するのって効率悪いよね.....

# マルチパスシェーダーの基本①

おおざっぱに図解すると普通のシェーダーはこんな感じに処理が行われます。

そのフレームに  
表示したい  
メッシュデータ  
(ボーン変形済みのメッシュ  
形状など)



## 頂点シェーダー

画面内の頂点の位置  
を決める処理

(カメラ位置、頂点位置オフセット  
など)



## ピクセルシェーダー

ピクセルの色を決める  
処理

(ライティング、テクスチャなど)

フレーム  
を表示

いわゆる「シェーダー」と呼ばれてる処理



# マルチパスシェーダーの基本②

「マルチパス」シェーダーを使うと、同じメッシュを設定を変えてもう一回描画することができます。GGシリーズの「背面法」のアウトラインはこの手法で作られています。

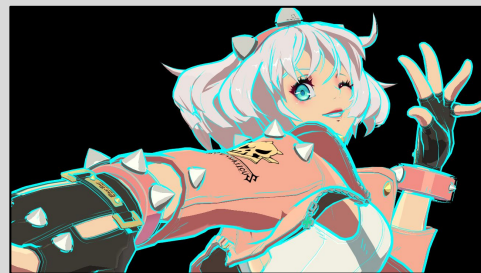
そのフレームに  
表示したい  
メッシュデータ  
(ボーン変形済みのメッシュ  
形状など)

※2回描画するけど、ボーン変形の計算は一回だけでいいのでメッシュを二つ持つよりも安上り！

パス1:まず本体メッシュを描画



二つの描画結果が重なって表示される



パス2:次に「背面メッシュ」を描画



メッシュ一個で  
背面法輪郭線が実現！

# マルチパスシェーダーのエンジンへの実装

残念ながらUnreal Engineにはデフォルトでマルチパスシェーダーを扱う仕組みがありません。そのため、**エンジン改造**によってマルチパスでレンダリングをできるように手を加える必要がありました。



このマテリアルを...

このマテリアルで  
もう一回描画して！

エンジンにインポートしたキャラクターメッシュのマテリアル設定内で特殊な設定を行うことで、  
**「任意のマテリアルを、別のマテリアルでもう一度描画」**  
する仕組みが実装されています。



# マルチパスシェーダーを扱える環境

ここまでマルチパスシェーダーの解説を行ってきましたが、実はこのマルチパスシェーダーを扱えるかはどうか環境によります。ゲームエンジンや3DCGソフトによってレンダリングのパイプラインが異なるためです。

マルチパスシェーダーを使える3DCGソフト

|           |   |
|-----------|---|
| MAYA      | ○ |
| 3DSMAX    | ○ |
| Softimage | ○ |
| Blender   | × |

マルチパスシェーダーを使えるゲームエンジン

|               |                |
|---------------|----------------|
| Unreal Engine | △(要改造)         |
| UNITY         | ○(HLSLかCGFXのみ) |
| GODOT         | ? (未検証)        |

※プレビュー画面でのリアルタイム描画

# マルチパスシェーダーまとめ

- ・特殊なシェーダーの描き方をすれば、一つのメッシュを違う設定で2回描画できる
- ・一回目の描画をトゥーン、二回目の描画を背面輪郭線とすれば「背面法」が可能
- ・ボーン計算は一回ですむからお得！
- ・メッシュを二つ持たなくていいから管理コストもメモリコストも効率よい

ただし.....

- ・使える環境は制限あるので注意
- ・UnrealEngineで使うには改造が必要だった
- ・「背面法」を使う上で必須ではない



Part5

# GGシリーズの背面法実装の工夫

# GGシリーズにおける背面法の実装の工夫

「背面法」の理屈がわかったところで、実際にギルティギアシリーズにおいて背面法がどのように実装されているのかを紹介していきます。

2000年に登場してから多くのゲームで活用され続けている「背面法」ですが、各タイトル毎の具体的な実装の仕方は当然**ひとつひとつ異なります**。

**原理は同じでも、実装の仕方**の部分ではタイトル毎にさまざまな**独自の工夫**がされています。当然、ギルティギアシリーズでも**多種多様な工夫**をしています。

背面法の基本の原理は比較的シンプルなものですが、それゆえに実際に製品に組み込んで表現に用いるためには、様々なエッジケースへの対応を**自前で実装**していく必要があります。

このパートでは、背面法を実装するうえで立ちはいだける**さまざまな問題**と、それぞれの**解決法**について、ギルティギアシリーズでの**工夫**を紹介していきます。



実装の工夫①

大前提：再限度の高いプレビューシェーダー

# 大前提:再限度の高いレビュー環境

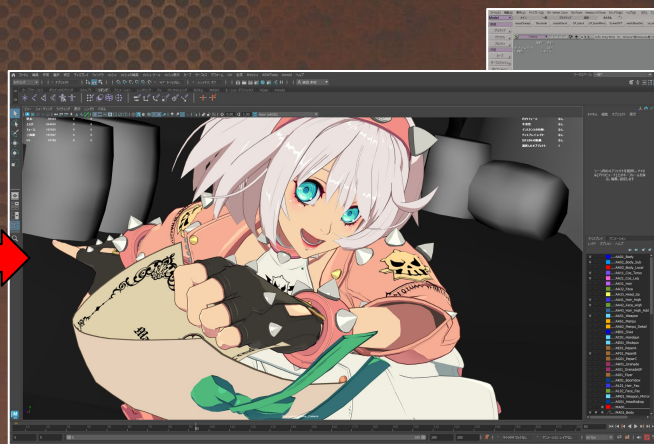
背面法の大きなメリットの一つが、適切なシェーダーを用意すれば3DCGソフト上でもゲーム中と同等の**アウトライン表現**が実現できることです。

モデラー、アニメーターが「**最終結果がどうなるか**」を常に確認しながら作業できること、そして3DCGツール上で行った**調整がゲームにもそのまま反映**されることはクオリティを追及するうえで計り知れないメリットがあります。

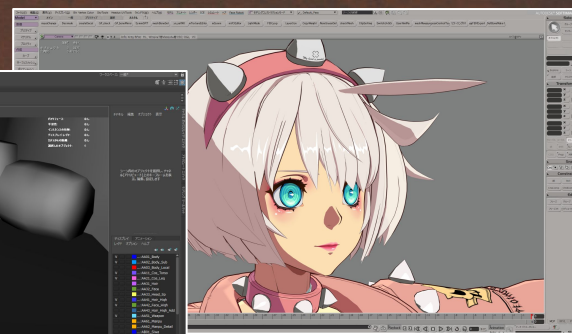
アークシステムワークスの3D格闘ゲームタイトルではこれを実現するために、各3DCGツール用にゲームエンジンと同等の**描画結果**が得られる**レビューシェーダー**を用意しています。



Unreal Engine



Maya



Softimage



実装の工夫②

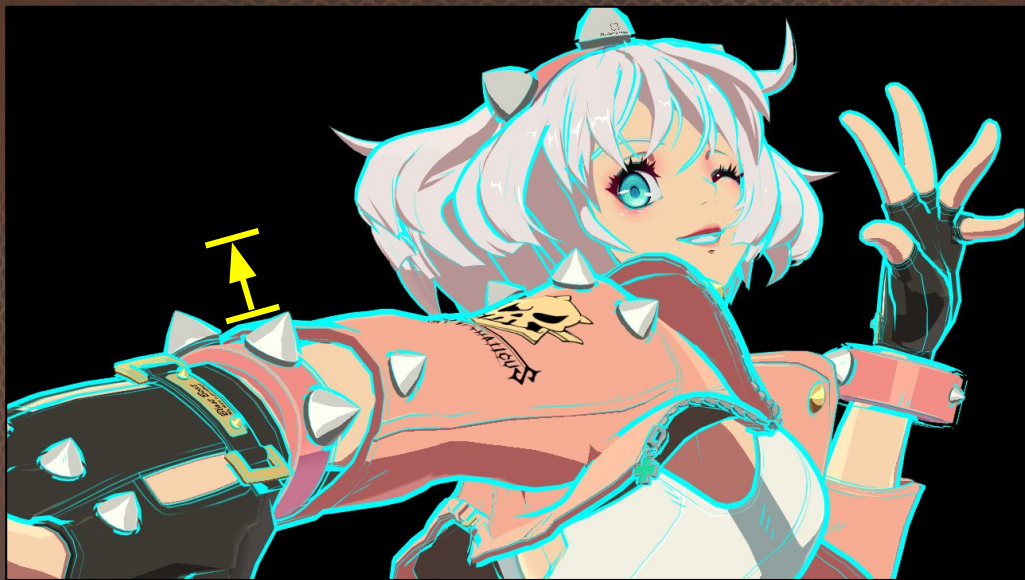
どんな構図でも安定した線の太さを実現する

# どんな構図でも安定した線の太さを実現する

背面法で描画するアウトラインですが、実際に画面上に表示するうえでどのくらいの「太さ」のアウトラインを出すべきか というのは避けて通れない課題です。

背面法では、シェーダー内の処理で背面用のメッシュを押し出して輪郭線を作りますが、では実際にどのくらい押し出せば適切なのでしょうか？

簡単そうに見えますが、様々な要素が絡み、結構気を使わないといけない問題です。



何mm押し出す？

1mm？

2mm？

それとも1cm？



# アップでもロングでも適切な太さ？

最初に直面するのがカメラとの距離の問題です。  
顔アップの構図の時 と、格闘ゲームのバトル時のような全身が映るロングの構図の時 とでは、  
適切な輪郭線の押し出し距離が変わって きます。



アップ時の顔の輪郭の太さは1 ~3mm程度？



ロング時の太さは7 ~10mm程度？

# もしカメラ距離によって太さを変えなかったら

もしも、カメラとの距離は考慮せず、常に一定の太さの背面法アウトラインだけで済ませようとする  
と、どうしても齟齬が出てきます。「ロング時に合わせた太さ」でアップにすると線が太すぎたり、逆  
に「アップの時の太さ」をロングで映すと線が細すぎたり。



ロング時の線の太さ10mmでアップにした場合



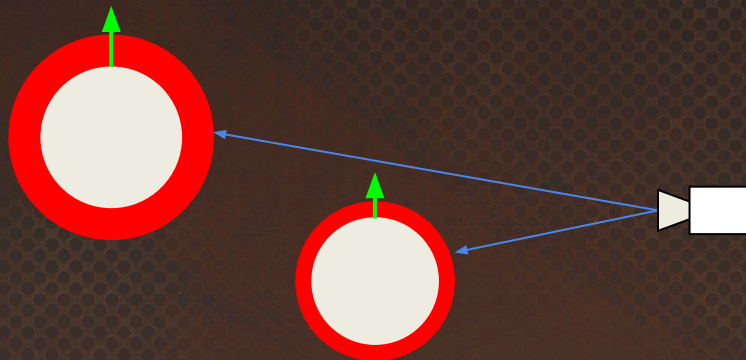
アップ時の線の太さ2mmでロングにした場合



# 解決法：距離に比例して輪郭線の太さを変える

この問題を解決するためには、シェーダー内の処理でカメラと頂点の間の距離に比例して背面メッシュの押し出し距離が大きくなるようにします。

こうすることで、アップの時は背面メッシュの押し出し距離が相対的に短く、距離が離れるにつれて背面メッシュが大きく押し出されるようになります。結果として、アップ、ロングどちらの構図でも画面上で適切なピクセル数の線の太さを得られるようになります。



視点からの距離に応じて押し出し距離を変える



これで万事解決！ .....ではない！

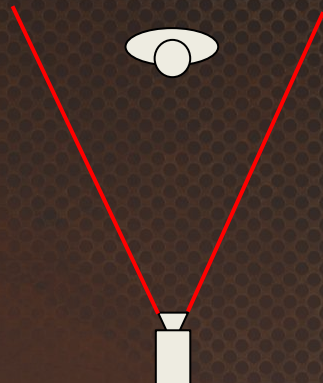
# ちょっと待て！距離だけじゃない！FOVも！

FOV(Field Of View)＝視野角は、おおざっぱにいうとカメラの視界の幅です。

対象までの距離が同じでも視野角が違えば、見える範囲が変化し、それにつられて被写体も大きく映ったり小さく映ったりします。



カメラ距離：9m  
FOV:55°



カメラ距離：9m  
FOV:15°



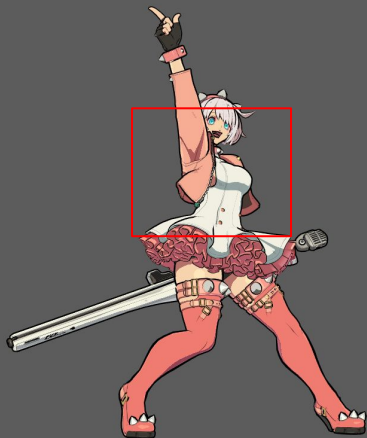
上図にて、カメラと被写体の距離が一定でも、FOVの違いによって被写体の大きさが違って見えるのがわかると思います。背面法アウトラインでは、この差にも対応しないといけません。



# FOVによるズームインで発生する問題

FOVを絞ることで遠いキャラにズームインした場合、特に工夫しなければ輪郭線は太く表示 されてしまいます。カメラからの距離を考慮して押し出し幅を変えているからです。

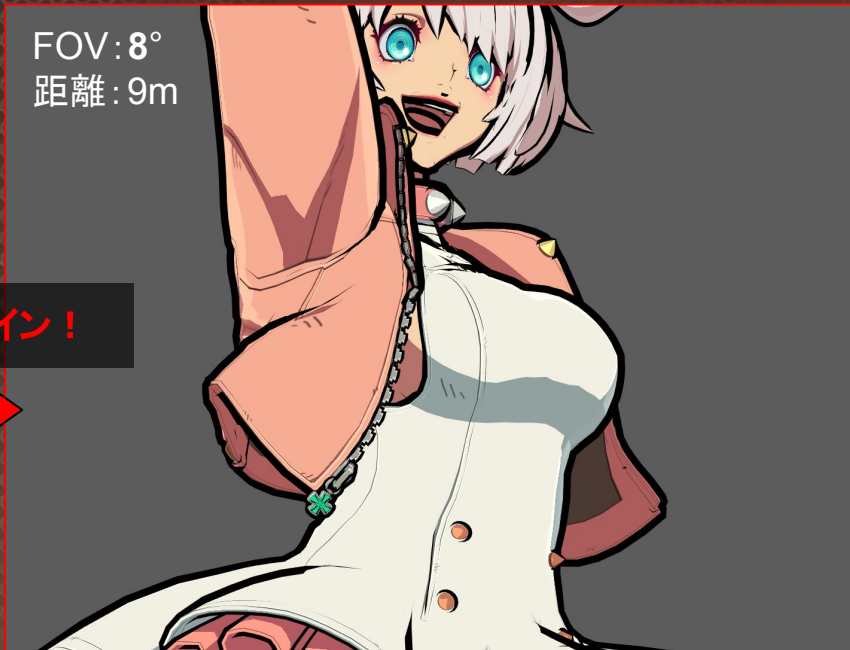
FOV: 35°  
距離: 9m



ズームイン！



FOV: 8°  
距離: 9m

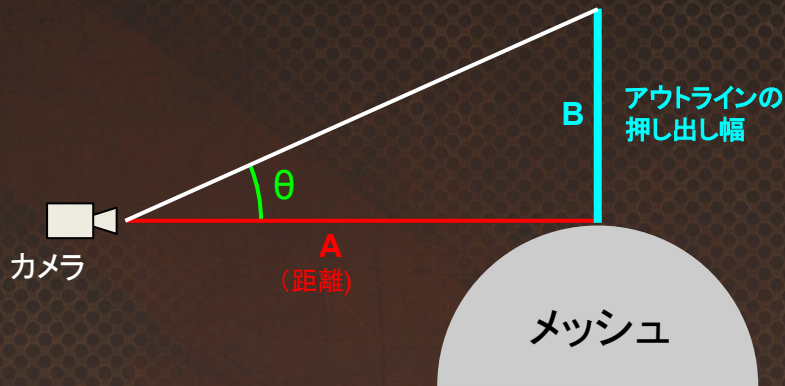


カメラ距離による調整とは別に、FOVによっても押し出し幅を調整 する必要があります。

# FOV問題への対策: 三角関数で解決

FOVに合わせた適切な線の幅を求める一つの方法は三角関数を用いることです。都合のいいことに距離と角度から幅を求める「そのものずばり」な式があります。

この式でカメラ距離とFOVの両方を加味した適切な線の太さを求めることができます。



$$B = A \tan(\theta)$$

要はカメラからの距離  $A$  に  $\tan(\text{FOV}/2)$  を乗算するようにシェーダーを組んでやれば良いわけです。

## ※FOVを扱う上での注意

3DCGソフトのシェーダーや、ゲームエンジンのマテリアルでカメラのFOVの値を取得するのはちょっとややこしいので注意が必要です。

一つの方法は、ゲームのプログラム側で計算量の高い  $\tan(\text{FOV})$  を事前に計算しておいてから、その結果だけをシェーダーに渡すというものです。

他にカメラのプロジェクトンマトリクス からFOVを算出する方法もあります。(  $1 / \text{Proj}[0][0]$  など )



# FOV問題解決

距離とFOVによる線の太さの調整が正しく実装されれば、カメラを近づけた顔アップの時も、FOVを絞ることで作った顔アップのどちらの場合でもアウトラインの太さを安定して描画することができるようになり、演出の幅が広がります。



カメラを近づけた顔アップ構図



FOVを絞った顔アップ構図

実装の工夫③

頂点カラーで輪郭線の太さを調整できるように

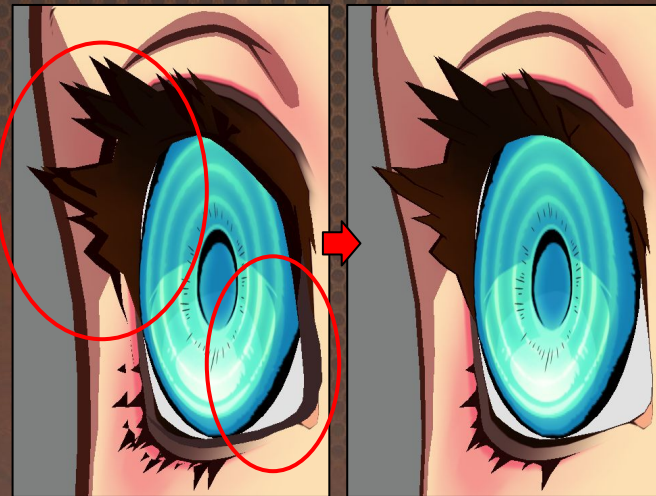


# 頂点カラーによる輪郭線の太さ調整



ギルティギアシリーズでは頂点カラーのアルファチャンネル の値で、背面メッシュの押し出し量を調整できるようにシェーダーが作られています。これにより、ペン画のような強弱のある線の表現 が可能になります。

背面法のメリットの一つが、このような頂点単位での線のコントロールです。



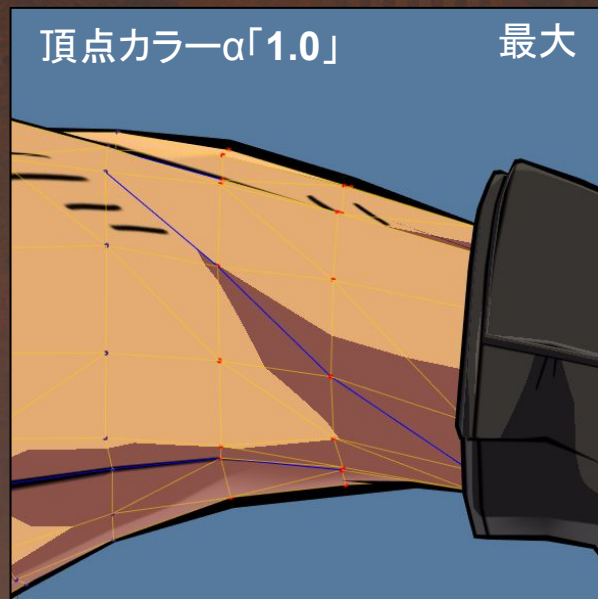
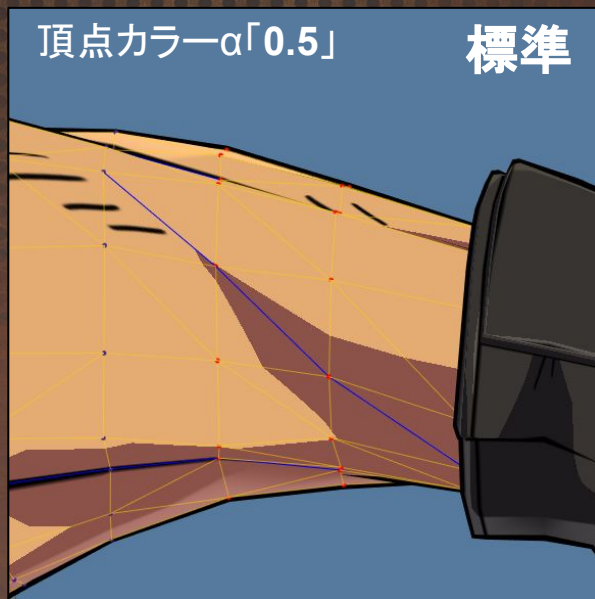
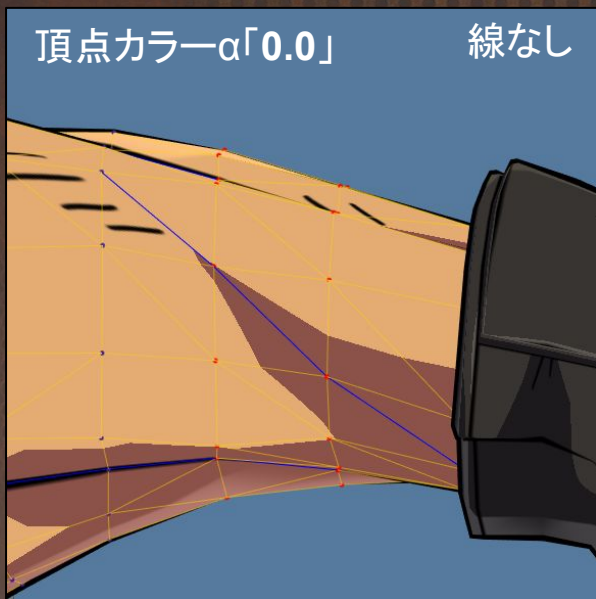
強弱だけでなく、「線を出したくない所」は線の太さを0にすることで意図しない描線を排除することもできます。

まぶたや睫毛の立体など、あえて輪郭線を出したくない部位 も多くあるからです。

# 頂点カラーで輪郭線の太さを制御する

頂点カラーは「1.0」以上の値を入れられないので、ラインを通常よりも「細くする」「太くする」両方の調整を行うためにGGシリーズの実装では「0.5」を基準としています。

「0.1」など「0」に近づくほど細くなり、逆に最大値の「1.0」に設定するとデフォルトの倍の太さに設定することができる、という感じです。





# 部位毎の頂点カラーによる輪郭線の太さ調整

部位ごとの特性に併せて輪郭線の太さを調整するのが重要です。

髪の毛や羽毛などの繊細なものや、入り組んだ形状の小さなパーツなどは意図して輪郭線を細くしたりします。逆に筋肉などボリューム感を強調したい部分は太くしたり。

この辺の強弱のつけ方は手書きイラストや漫画における線描のテクニックと重なる部分が多分にあります。どの媒体でも「見やすく表現する」という目的は同じだからです。



毛先などデリケートな部分は細め



細かいパーツも細くしておく



筋肉などはボリューム感を強調

実装の工夫④

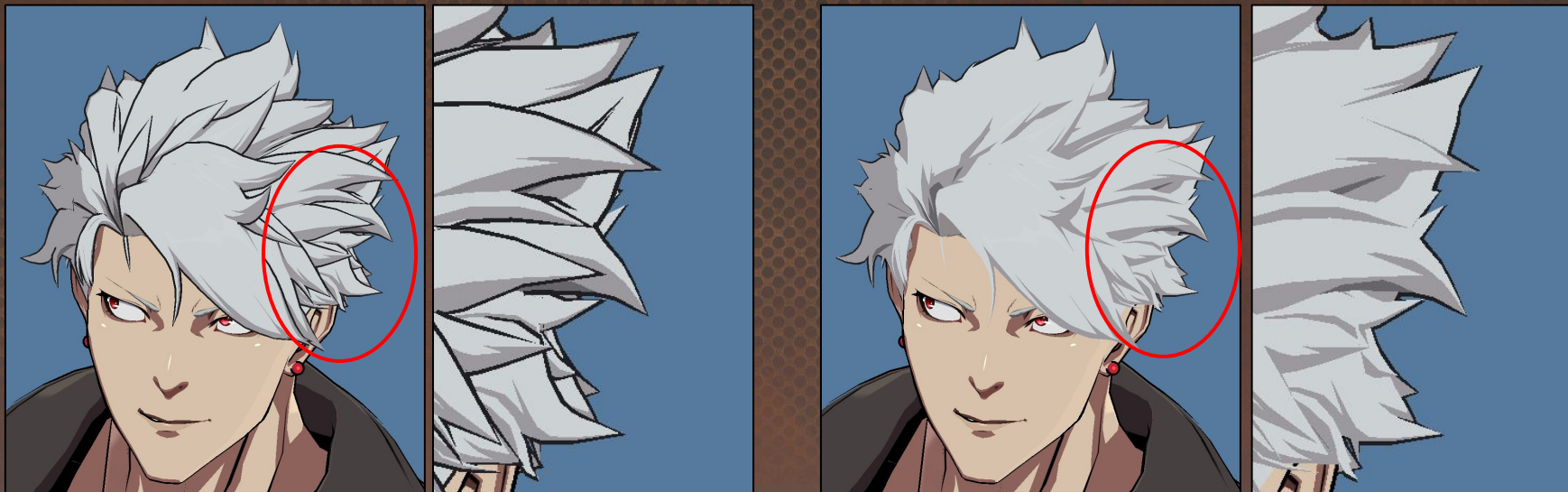
背面メッシュを「奥」にずらせるようにする



# アウトラインの描画深度を調整できるようにする

キャラクター表現を行う際にはアウトラインを「出す」ことと同じくらい意図しない部分に「出さない」ことも重要になります。「表現としてここには線を出したくない」というケースにも対応できるようにしておく必要があるわけです。

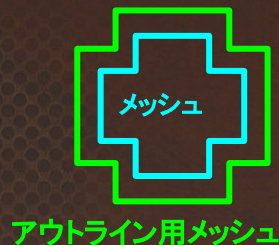
幸い、背面法のアウトラインはシェーダーでの記述次第で頂点単位でアウトラインの表示をいじることができます。キャラクターの任意の部位のアウトラインを出したり出さなかったり、あるいは、「一番外側の輪郭部分だけに線を出す」ように設定することができたりします。



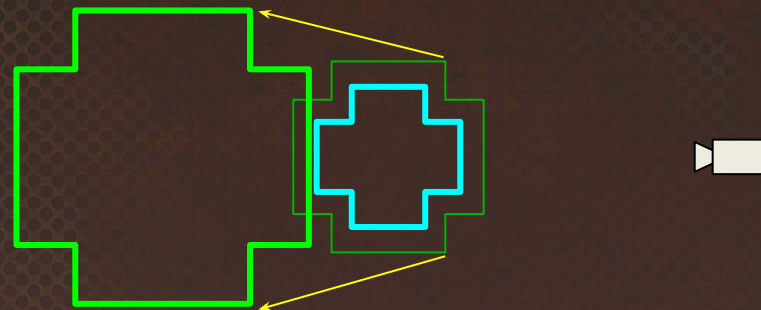
# アウトラインメッシュを「奥」にずらす

「一番外側のアウトラインだけ残す」処理は、実は結構シンプルです。アウトラインを構成する背面メッシュを、頂点シェーダー内で実際の位置よりも「奥に」あるものとしてレンダリングしてあげれば実現できます。これはシェーダーで各頂点のDepthの値をずらすことで実現できます。アーティストにわかりやすい表現に言い換えると、  
「カメラの視線方向に背面メッシュをずらして描画」するのとほぼ同じです。

通常の描画



アウトライン用の頂点を奥にずらして描画

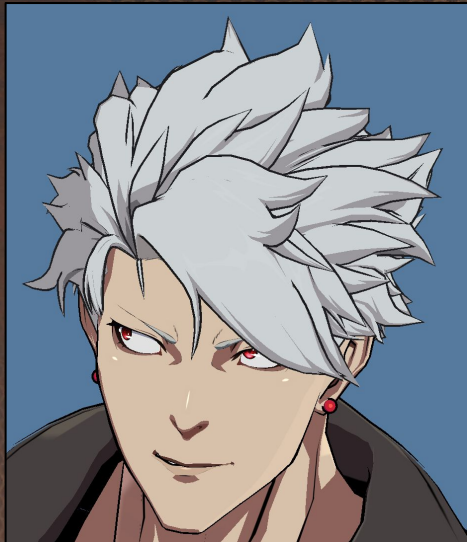


結果、一番外の輪郭だけが残って見える

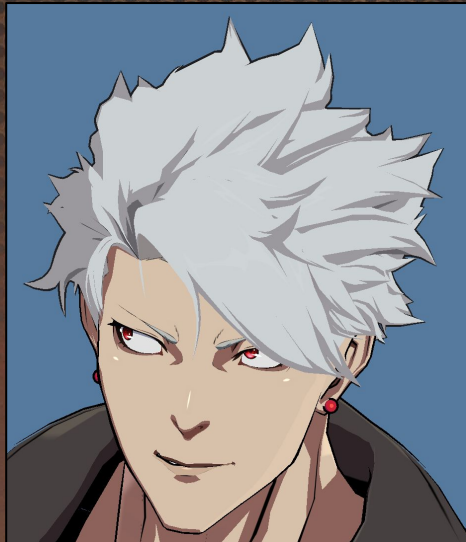


# 背面メッシュを「奥」にずらす処理

「背面メッシュを奥にずらす」処理をかけることで、任意の場所に「線を出したり」「出さなかったり」を設定できるようになります。



通常の表示



髪の毛の部位だけ頂点カラーを編集

ギルティギアシリーズでは、どの部位の線を出すのか、あるいは出さないのか、は頂点カラーのBチャンネルを使って指定できるようになっています。

左図の例では、髪の毛の部分だけ頂点カラーBをいじって、「一番外の輪郭」を残して髪の毛のアウトラインが描画されないように設定しています。

実装の工夫⑤

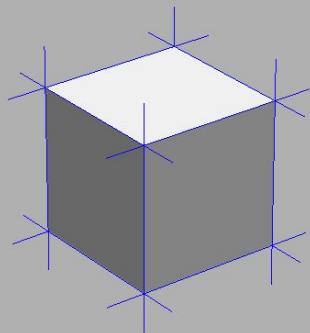
# ハードエッジ、カスタム法線の問題と解決法



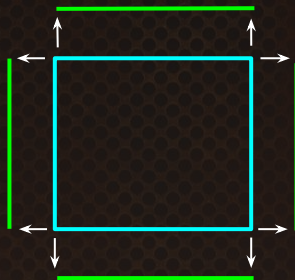
# ハードエッジの問題

背面法では「法線」の方向にメッシュを押し出すことで輪郭線を描画します。この時、法線がハードエッジに設定されていたり、ライティングを調整するために法線がカスタム化されていたりすると、その結果が輪郭線用のメッシュを押し出す方向に影響を与えてしまいます。

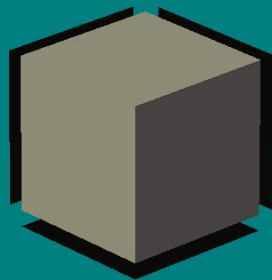
特にハードエッジの場合は、ポリゴン毎に法線が異なる方向を向くため、押し出されたメッシュがバラバラになってしまい、輪郭線もとぎれとぎれになってしまいます。



ハードエッジはシャープな陰影が得られるが、これは法線がポリゴンごとに別方向を向いているため



法線がバラバラなので、押し出された背面メッシュもバラバラになってしまう



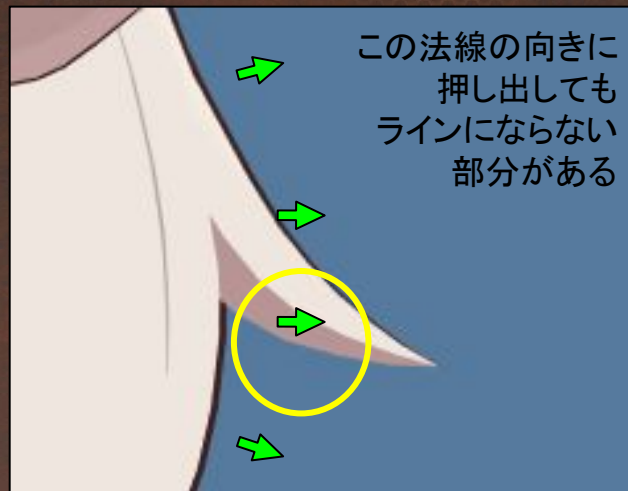
背面メッシュがバラバラだと、アウトラインに途切れる箇所が発生してしまう

# カスタム法線の問題

トゥーンシェーダーを用いた3D表現では、**法線を編集**することで陰影の**ラインティングをコントロール**する手法がよくとられます。この**法線の編集**は、実は背面法による表現と**相性が良くありません**。ラインティングのために編集されたカスタム法線が、必ずしも適切な背面メッシュの押し出し方向になるとは限らないためです。



ポピュラーな「球状法線の焼き付け」  
手法で表現された髪の陰影

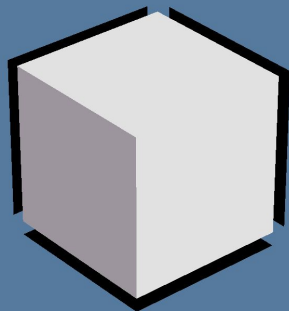


結果、背面の押し出し方向も影響を受け、内側に線が出ない！



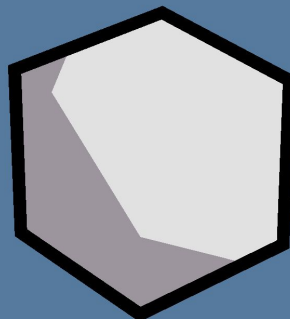
# ハードエッジやカスタム法線の解決法

ハードエッジはシャープな陰影  
しかし線は途切れる



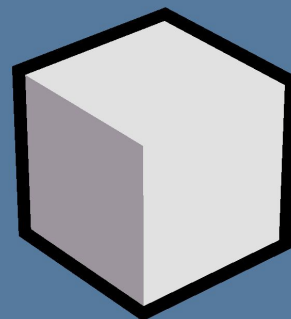
+

非ハードエッジは線はつながる が陰影を  
シャープにできない



=

なんとか両方のいいとこどりをしたい！

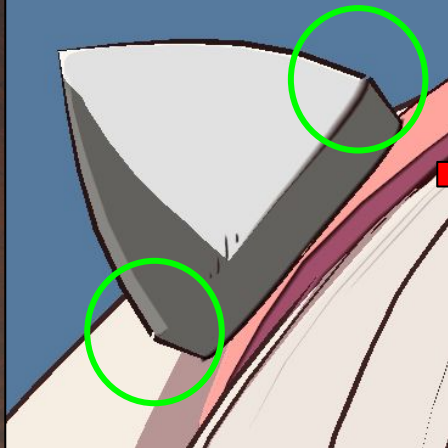


この問題の解決法の一つが「**法線を二つ持つ**」こと。つまり、**ライティング用の法線**とは別に、**背面法のための法線**を用意して、その二つを使い分けるという手法です。

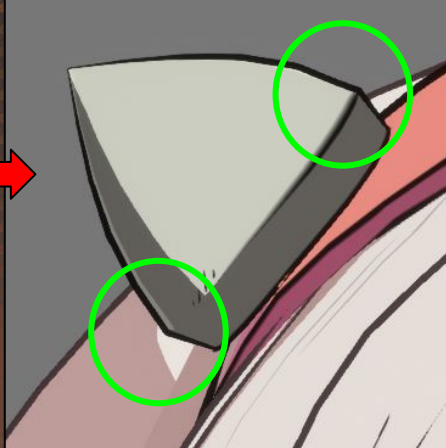
**法線を二つ持つ**ことで、**ライティング時**にはハードエッジやカスタムされた法線によるデザインされた陰影を表現でき、**背面法メッシュ生成時**には線が途切れないように**スムーズな法線**を使う、という「いいとこどり」が可能になります。

# 「二つ目の法線」で解決された結果

ライティング用  
法線で描線



第二の法線  
で描線

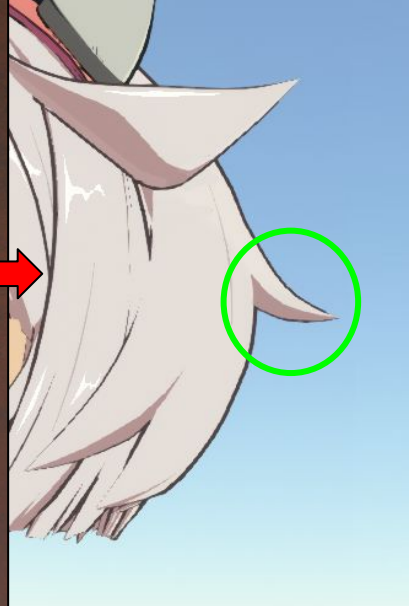


第二の法線によって、線の途切れや押し出し方向  
の不備が解消されている

ライティング用  
法線で描線



第二の法線  
で描線





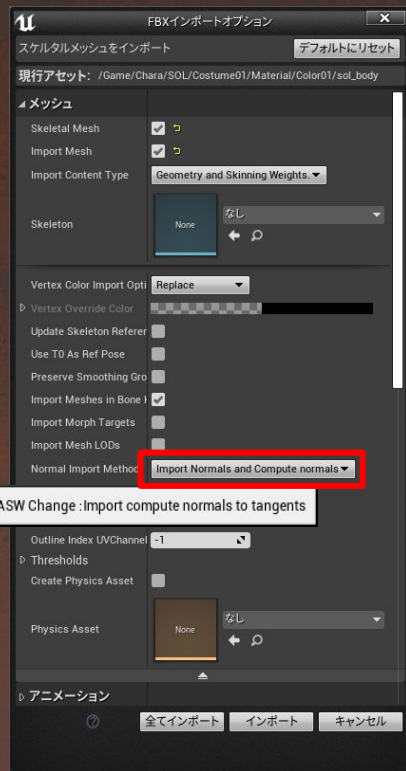
# 二つ目の法線？どうやって？

しかし一般的なゲームエンジンでは、このような「二つ目の法線」はデフォルトではサポートされていません。

ギルティギアシリーズでは「**タンジェント**」のパラメータを「**第二の法線**」として扱うことでこの問題を解決するアプローチをとっています。

「**タンジェント**」は、主に**ノーマルマップ**などを表現するために、各頂点が持っているパラメータです。アークシステムワークスの格闘ゲームのトゥーンシェーダーでは**ノーマルマップ**を使用しないため、この「タンジェント」のパラメータが好都合にも使われずに宙ぶらりんになっていました。

これを利用し、メッシュインポート時に、「タンジェント」のパラメータに、再計算された**第二の法線を上書き**することで、実質的にシェーダー内で**二つの法線を使い分けられる**ようになりました。(ただし、これはエンジンの改造を要します)



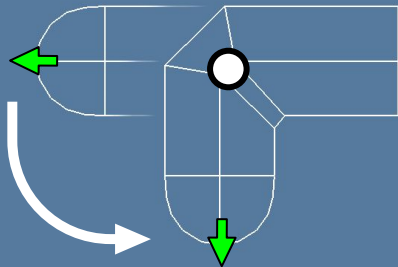
メッシュインポート時に再計算した法線をタンジェントに上書きする(要エンジン改造)

# なんでタンジェント？

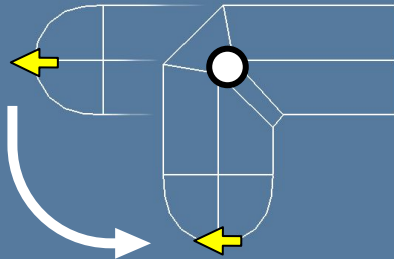
タンジェントをではなく、法線パラメータをあらかじめ**頂点カラー**に格納しておく、という方法も考えられますが、実は問題があります。

通常の法線はボーンの変形に追従して変化しますが、頂点カラーに格納された法線データはボーンの変形に追従しないため、アニメーションさせると法線がついてこないのです。そのため、ボーンによる変形が起こるとライティングやアウトラインの押し出し方向がマッチしなくなってしまいます。

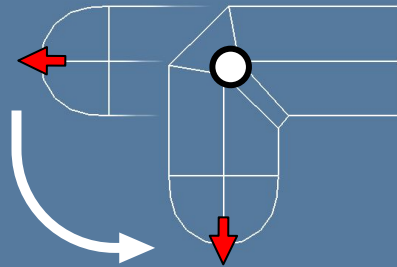
法線はライティングに使うのでボーンと一緒に回転する



頂点カラーに入れた値はボーンの回転についていけない



タンジェントは法線と同じくボーン変形の影響を受ける



その点、**タンジェント**はもともと法線と同じくライティングに使用するため、ボーンの回転の影響を受けて毎フレーム変化します。そのため、第二の法線として扱うのに適しているのです。



# GGシリーズの背面法実装の要点まとめ

- ・3DCGツールでもゲームエンジンでも同じ見た目を得られるレビュー環境
- ・どんな距離やFOVでも安定した線の太さを得るための工夫
- ・頂点カラーで線の太さを調整できる工夫
- ・頂点カラーで線を「奥」にずらせるようにする工夫
- ・ハードエッジ、カスタム法線でも線が崩れないよう「第二の法線」を持つ工夫

Part6

# GGシリーズにおける背面法の 制御テクニック



# 背面法のポテンシャルを活かす「運用」

ここまで、アークシステムワークスの格闘ゲームにおいて**背面法の機能**がどのように「**実装**」されているのかを紹介してきました。しかし、機能は「**実装**」されているだけでは不十分です。その機能のポテンシャルを引き出す「**運用**」のテクニックも同時に非常に重要となります。

このパートではここまで紹介してきた背面法の機能を活用して望む表現を行うための細かな**制御テクニック**を二つ 紹介していきます。

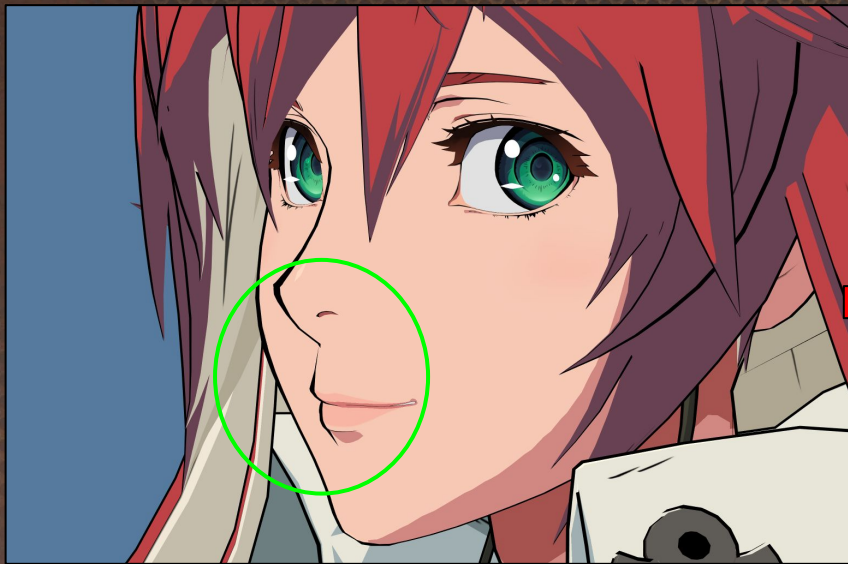
- ①余計な線が出ないようにする
- ②重要テクニック:「背面を閉じる」

意図しない線が出ないようにする



# 意図しない線が出ないようにする

特に顔などで、**意図していない**ところにラインが出てしまうと困ってしまいます。**特定の表情、特定の角度、特定の距離** など、思わぬタイミングで**余計なライン**が出てしまうことはできれば避けたいところです。頂点カラーの編集で局所的に背面メッシュを「奥」に**ずらす**機能は、このような問題を回避することにも使えます。

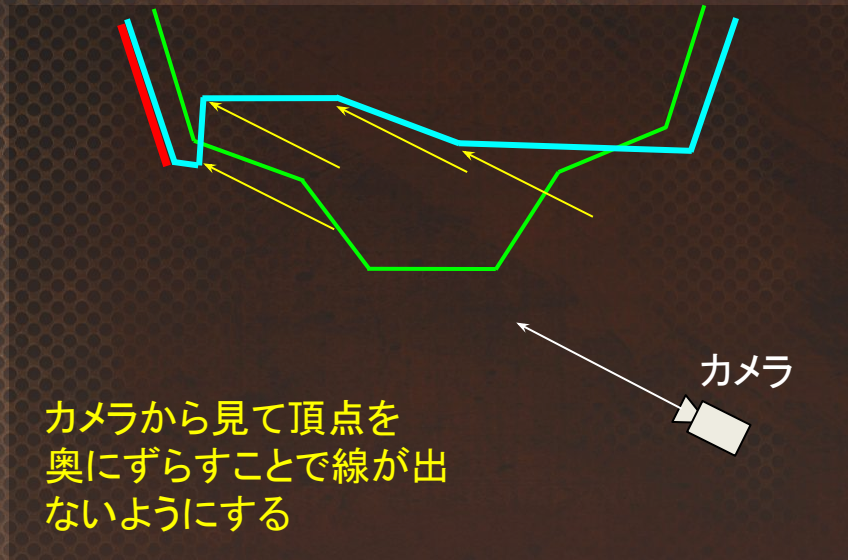
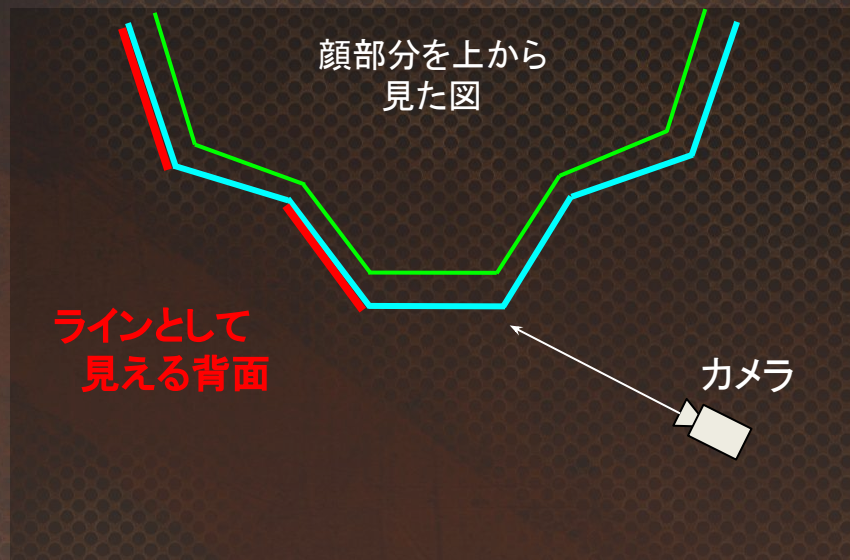


意図しない部分に線が出てしまった！



その部分の頂点カラーを調整！

# 線を消す仕組み



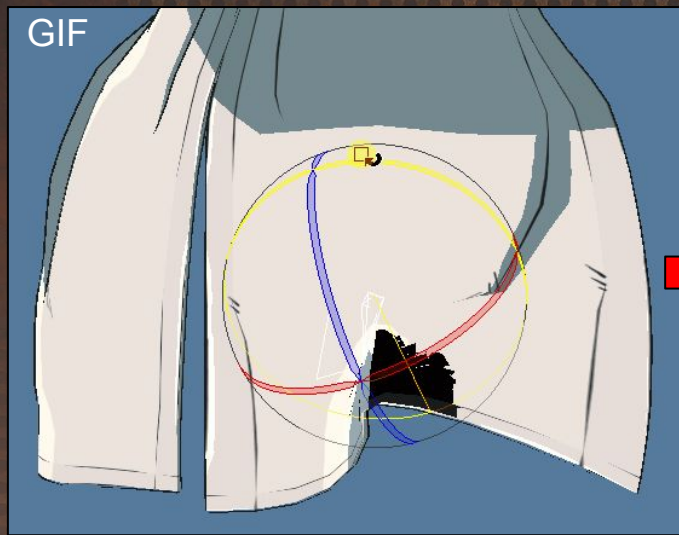
先に説明した頂点カラーを使って「奥にずらす」機能を活用すれば、**局所的にラインが出ない**ように調整することができます。

単純に線の**強弱で消した場合と違い**、ずらす方向はあくまでもカメラから見て「奥」なので、**一番外の輪郭は残る**ことに留意してください。

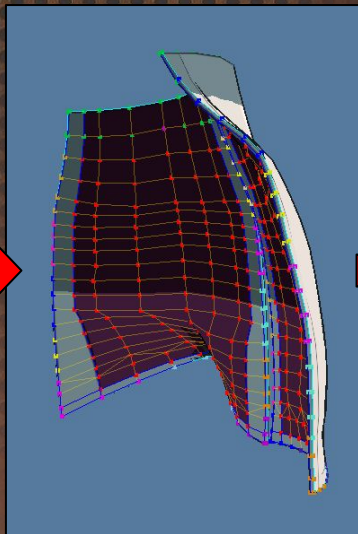


# 布の裏地がはみ出ないように「奥」にずらす

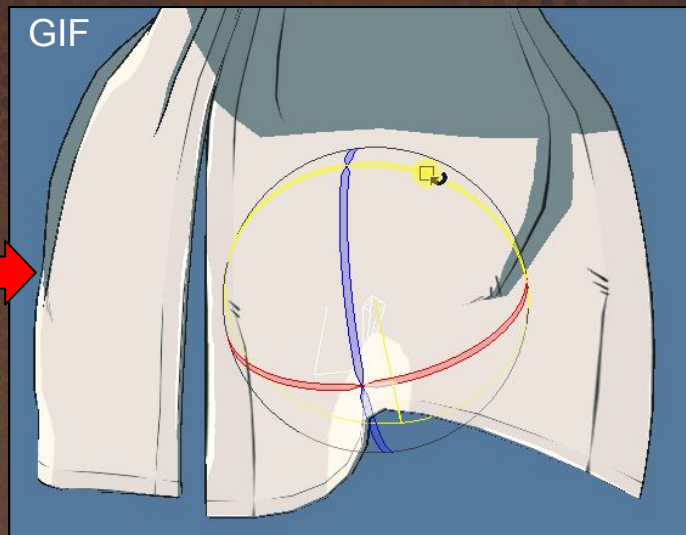
揺れ物などの布の一部が大きくひねられた時に、布の裏地のアウトライン用メッシュが表面を貫通して汚いビジュアルになってしまうことがまれにあります。これも頂点カラーの編集で裏地の部分だけ、アウトラインを「奥」に飛ばすことで予防することができます。



布を動かすボーンが強くねじられた結果、布の裏地のアウトラインが貫通してしまうことも



裏地部分の頂点カラーを調整



裏地部分はアウトラインが「奥」にしか出ないように設定しておけば貫通を未然に防ぐことが可能

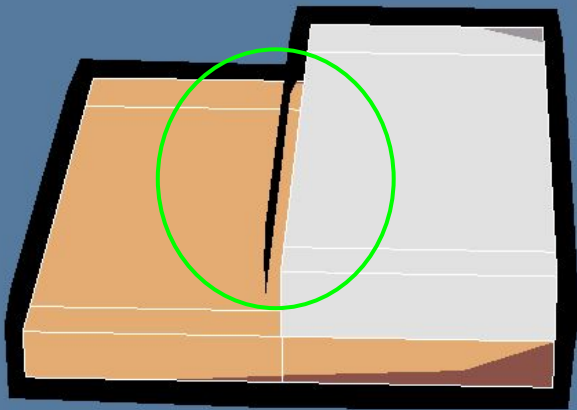
# 意図して線を出すテクニック



# 線が「浮いて」しまう問題

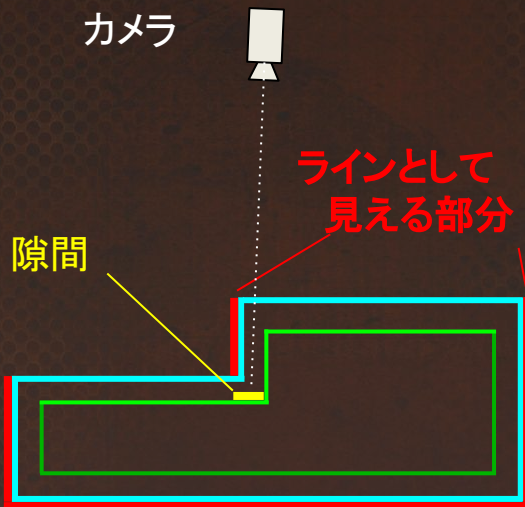
背面法のアウトラインの弱点として、入り組んだ形状では角度によって「谷」の部分の線が「浮いて」見えてしまうことがあります。背面用のメッシュを法線方向に押し出すという原理を理解していると、なぜこうなるのかということが説明できます。

GIF



角度によって入り組んだ部分のラインが「浮いて」見えてしまっている例

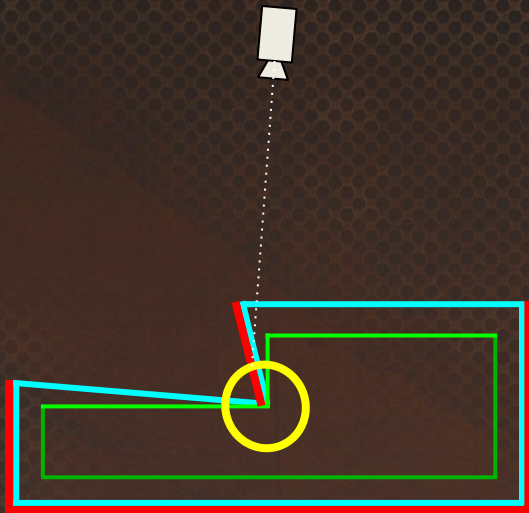
カメラ



押し出された背面メッシュと本体の間に実際に隙間があるためラインが「浮いて」見える

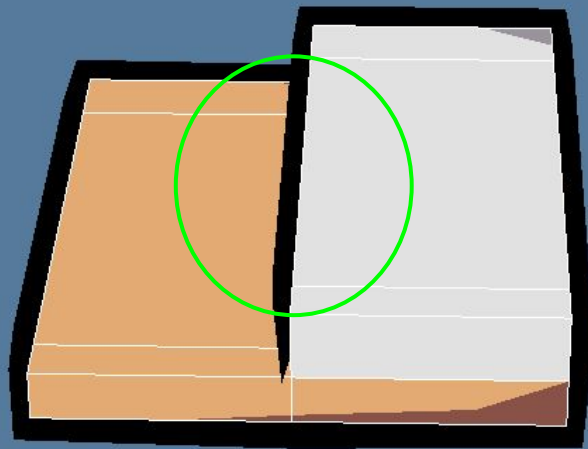
# 線が「浮いてしまう」問題の解決法

押し出された背面メッシュと本体の間に隙間があるのが「線が浮いてしまう」問題の原因です。なので、この部位の頂点カラーの値を0にしてこの隙間を「閉じて」やれば、この問題は解消できます。



頂点カラーで、**この部分**の背面メッシュの厚みを0に設定し、「背面を閉じる」

GIF

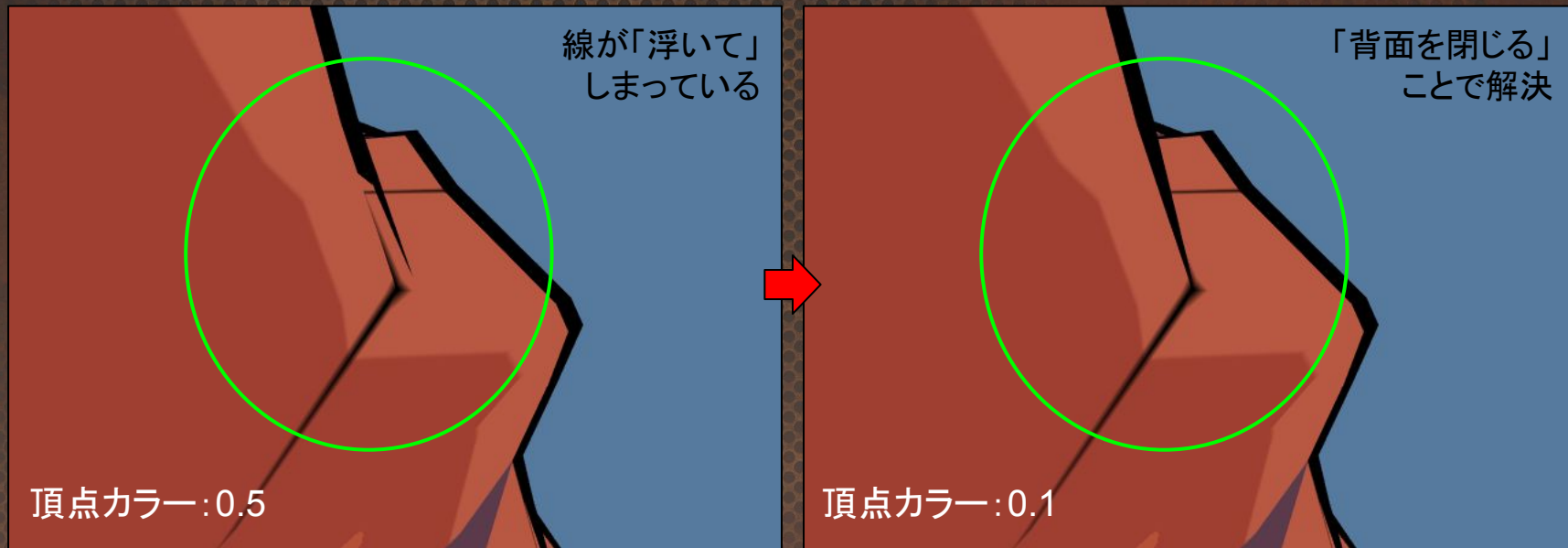


頂点カラーの調整によって線が「浮いて」しまう問題を解消



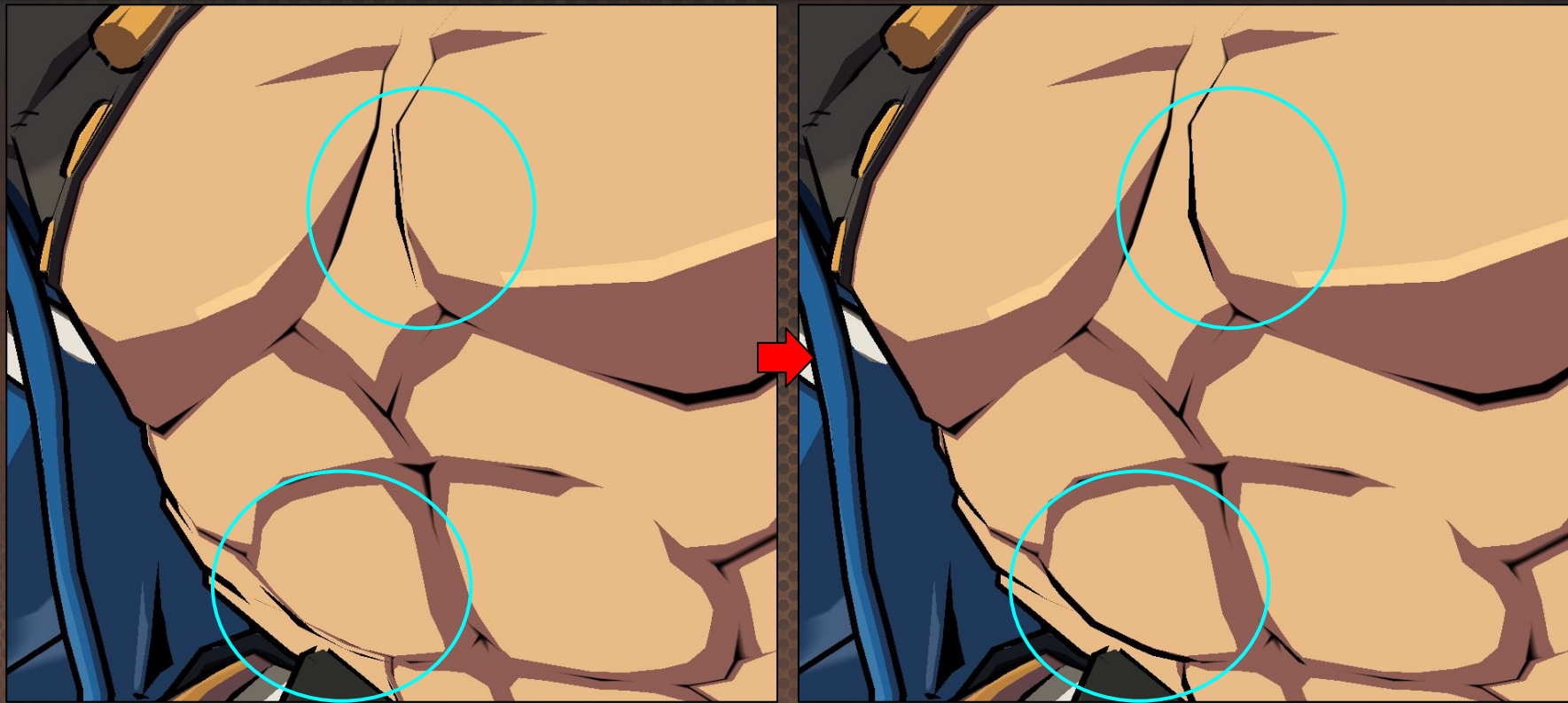
# 「背面を閉じる」技法の活用① シワを閉じる

この「背面を閉じる」ことで**ライン描画を保証**する技法はさまざまに応用が可能です。



衣服のシワの「谷」の部分の輪郭線の厚みを減らして「**背面を閉じる**」ことで、意図せず線が「浮いて」しまうのを防げる

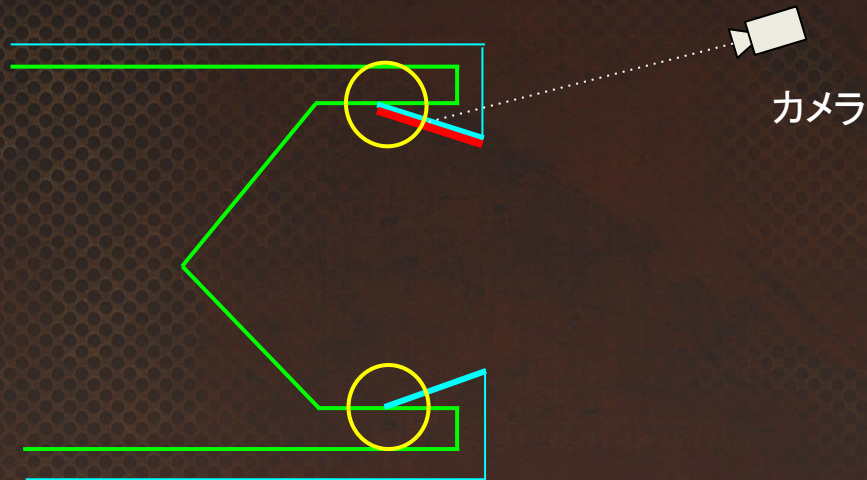
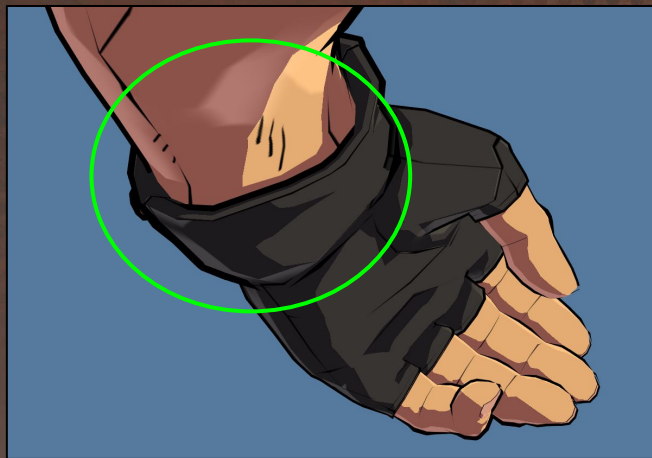
## 「背面を閉じる」技法の活用② 筋肉の谷を閉じる



同様に、筋肉の隆起の「谷」に相当する部分も、「背面を閉じる」ことでラインが整理されます。調整しないとでは描線のキレイさには大きな差が出ます。



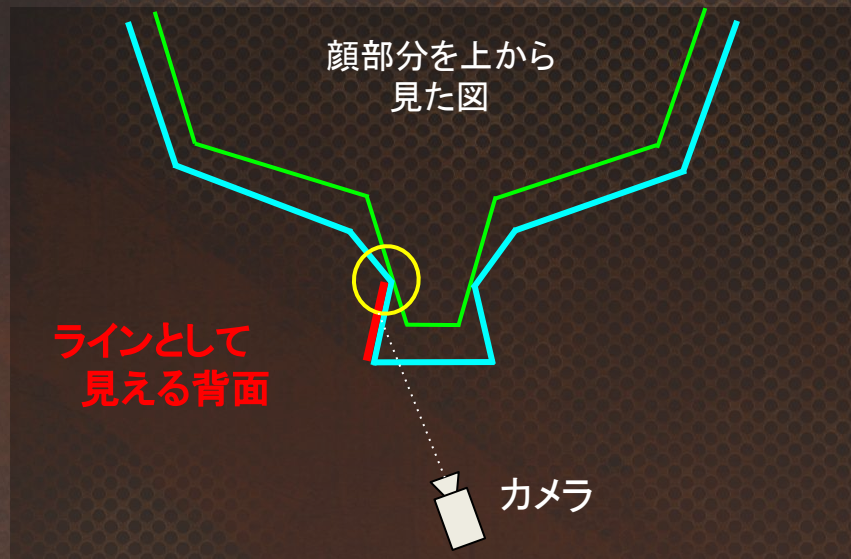
# 「背面を閉じる」技法の活用③ 開口部を閉じる



開口部の近くで**頂点カラー**を0にして「**背面を閉じる**」

手袋、衣服のソデ、スソなど筒状の構造では、開口部に近い部分で「背面を閉じる」ことでフチ部分に安定したラインを引くことができます。

## 「背面を閉じる」技法の活用④ 浅い角度で線を出す



同じように男性キャラクターの鼻筋の側面などに「背面を閉じる」部分を作ることによって、浅い角度でもラインが出やすくなるように設定することができます。**意図して「立体を強調」**するとき非常に効果的なテクニックです。



# 背面法の制御テクニックまとめ

「背面法」などのアウトラインを描画するテクニックは多くの場合「自動的に」ラインを引いてくれる便利なものととらえられています。しかし、「自動的に」引かれたラインは、常にアーティストの思った通りの線になってくれるわけではありません。

意図しない部分に線が出たり出なかったり。あるいは「この角度から見たときに線を出したい！」という時に線が出てくれなかったり。そのような線の見え方を「**制御**」して、**意図したとおりの見た目** を作れるようになって初めて、「**表現**」と呼べるものになるのではないかと考えています。

そして今回紹介した頂点カラーを用いた制御テクニックは、モデラーが「背面法」の**特性を理解**して初めて可能となります。

モデラーが**仕組みを理解した上で調整** をしていること、そしてシェーダーを作る人が**最終的な表現の要求を理解**していること。この両方がそろうことで、より良いクオリティのビジュアルの追及が可能になります。

# この講演のまとめ



# この講演のまとめ

技法が発明されてから20年以上がたちますが、「背面法」アウトラインは今もNPRビジュアルの最前線を支える手法として活用され続けています。

基本となる原理は**わかってしまえばシンプル**なものではありますが、それを**効果的に表現に落とし込む**には様々な**工夫と応用が必要**なことがわかっていただけたかと思います。

背面法を使って表現を行う**モデラー**は、その技法の**原理**や**実装の仕組み**を知ること、より**効果的な運用**ができるようになるかと思っています。

また背面法のシェーダーを実装する**テクニカルアーティスト**や**プログラマー**も、背面法を使ってモデラーがどのような表現を行いたいのかの**勘所**を知ること、より**適切な実装**が可能になると思います。

**技法を使う側、実装する側**、その**双方**が技法の根本原理と、そこから派生する様々な**制約や特性**を知っておくことは、最終的な表現のクオリティを高めるうえで大いに有益です。

このセッションがその一助になれば幸いです。