内容

[**１．作品概要** 2](#_Toc43728463)

[1.1 タイトル 2](#_Toc43728464)

[1.2 対応ハード 2](#_Toc43728465)

[1.3 製作人数 2](#_Toc43728466)

[1.4開発期間 2](#_Toc43728467)

[1.5開発環境 2](#_Toc43728468)

[1.6 使用言語 2](#_Toc43728469)

[**２．ナビゲーションメッシュの生成** 3](#_Toc43728470)

[**３．A\*経路探査とスムージング** 4](#_Toc43728471)

[**3.1 スムージングのアルゴリズム** 5](#_Toc43728472)

[**４．アニメーションブレンディング** 6](#_Toc43728473)

[**５．ポストエフェクト** 8](#_Toc43728474)

[**５.1 ブルーム** 8](#_Toc43728475)

[**６．背景の映り込み** 11](#_Toc43728476)

# **１．作品概要**

## 1.1 タイトル

Break Down Tank(ブレイクダウンタンク)

## 1.2 対応ハード

PC Windows10

## 1.3 製作人数

　１人

## 1.4開発期間

2019年8月～

## 1.5開発環境

エンジン

　 学校内製の簡易エンジン(DirectX11)

ツール

Microsoft VisualStudio2017

3DSMAX2018

FireAlpaca

Git

## 1.6 使用言語

C++

# **２．ナビゲーションメッシュの生成**

経路探査を行うためのナビゲーションメッシュをプログラムで自動生成しています。

３Dモデルの頂点バッファとインデックスバッファの情報をもとに地形データを生成します。

地形データと障害物で当たり判定をとって、衝突したポリゴンを除いて、残ったポリゴンをナビゲーションメッシュのセルとしてリストに登録しています。

当たり判定はBullet Physicsを使用しています。

図１(障害物判定を行わない場合のナビゲーションメッシュ)

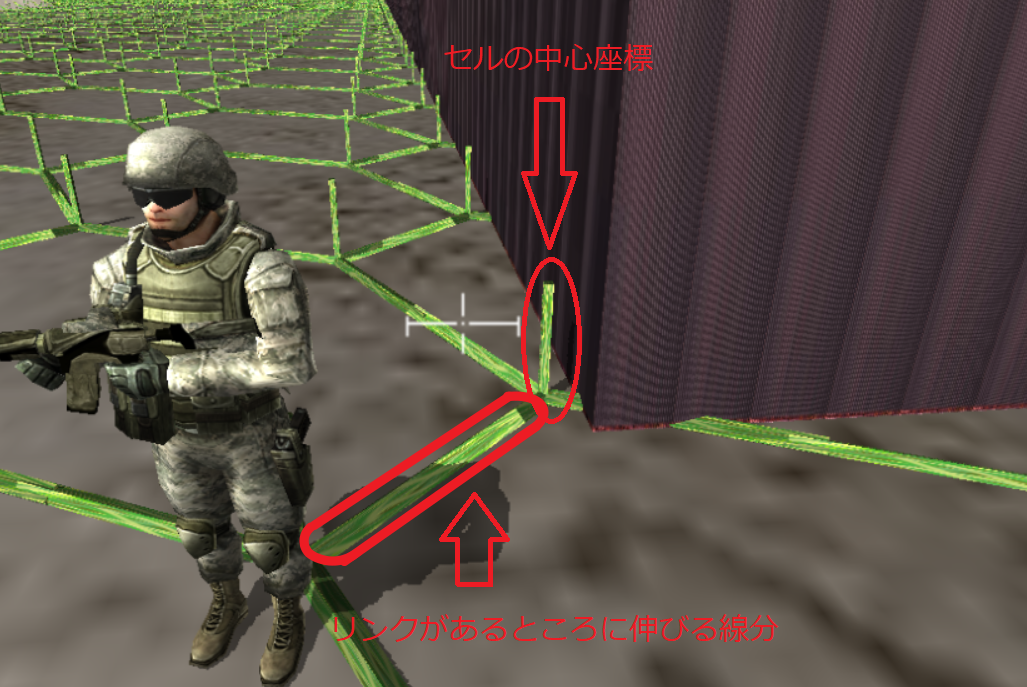


図２(障害物判定を行った場合のナビゲーションメッシュ)

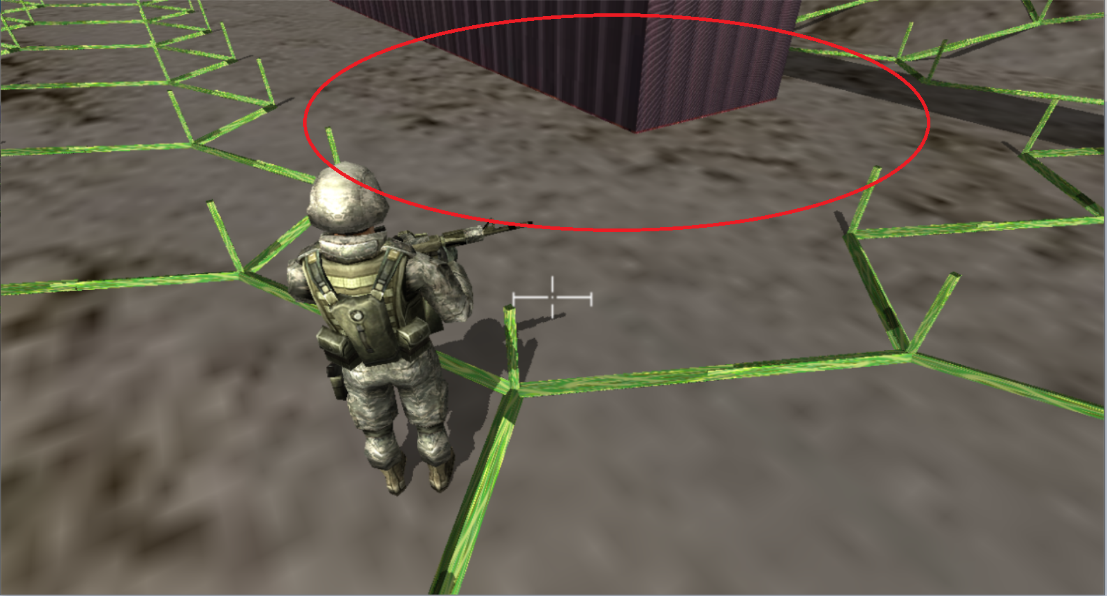
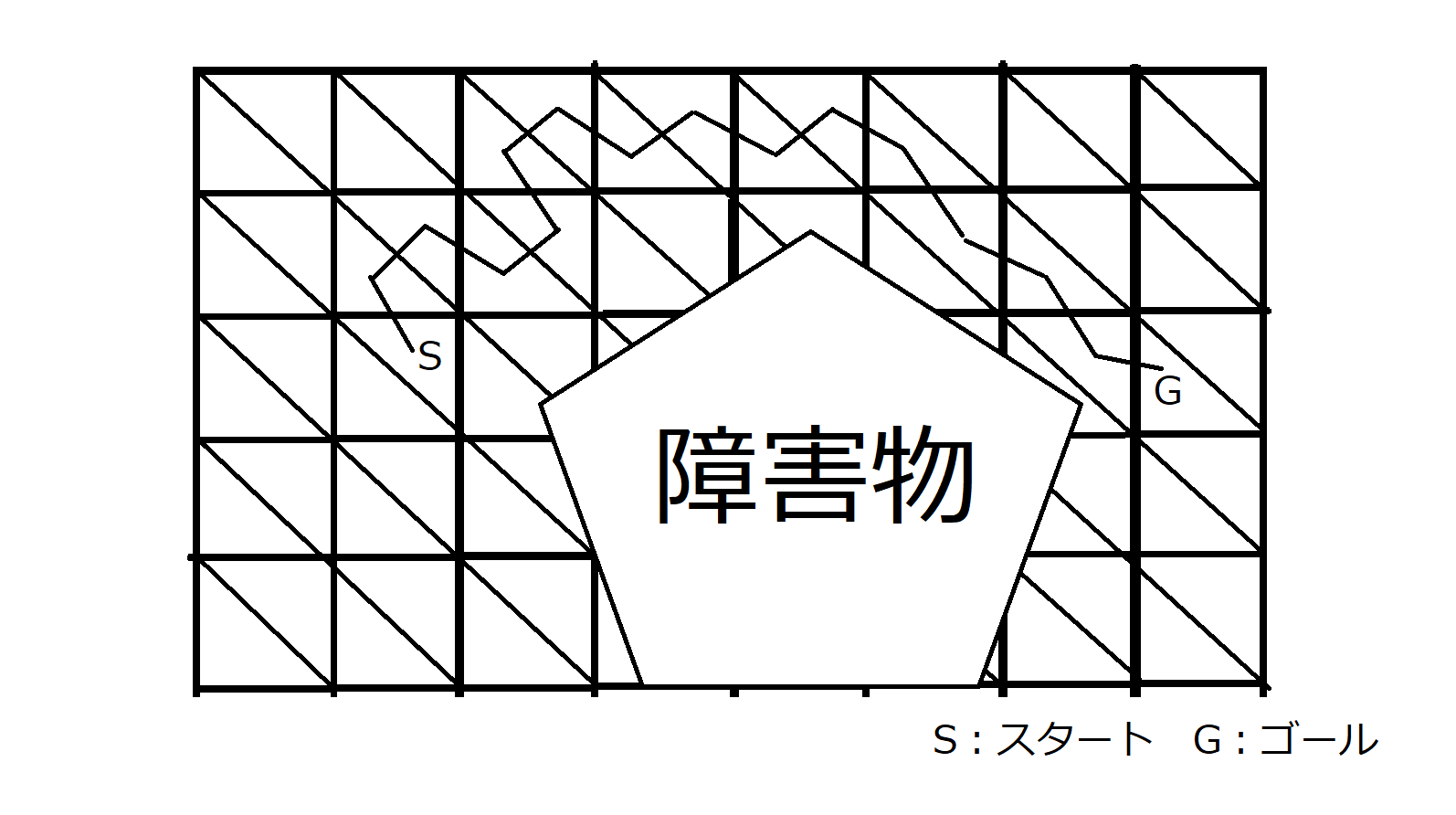


図2の壁付近のセルが除去されたことがわかります。

# **３．A\*経路探査とスムージング**

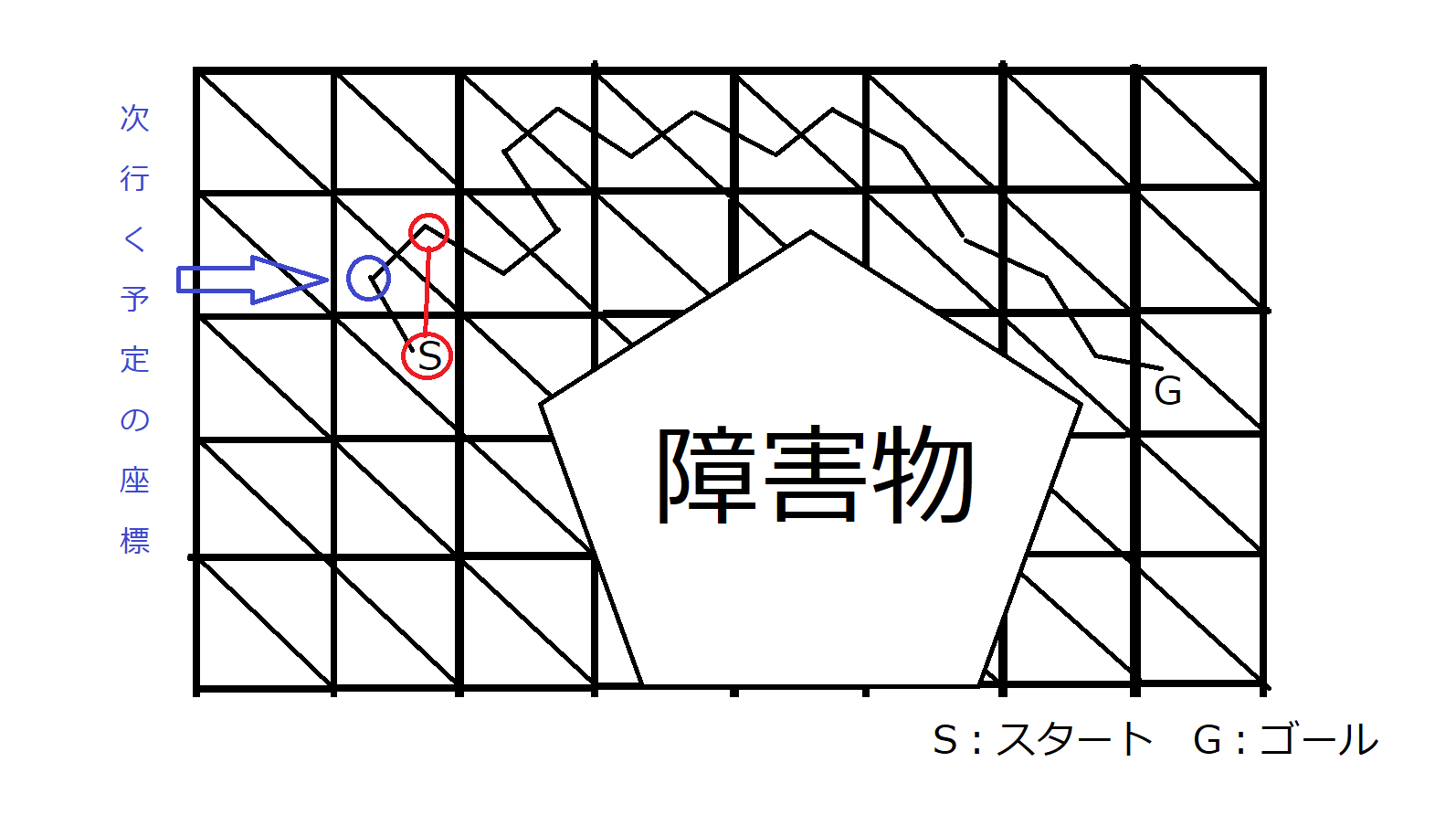
経路探査はA\*アルゴリズムを使用しています。ナビゲーションメッシュとA\*を利用して経路を作成すると、その経路はジグザグになってしまいます(図３参照)。

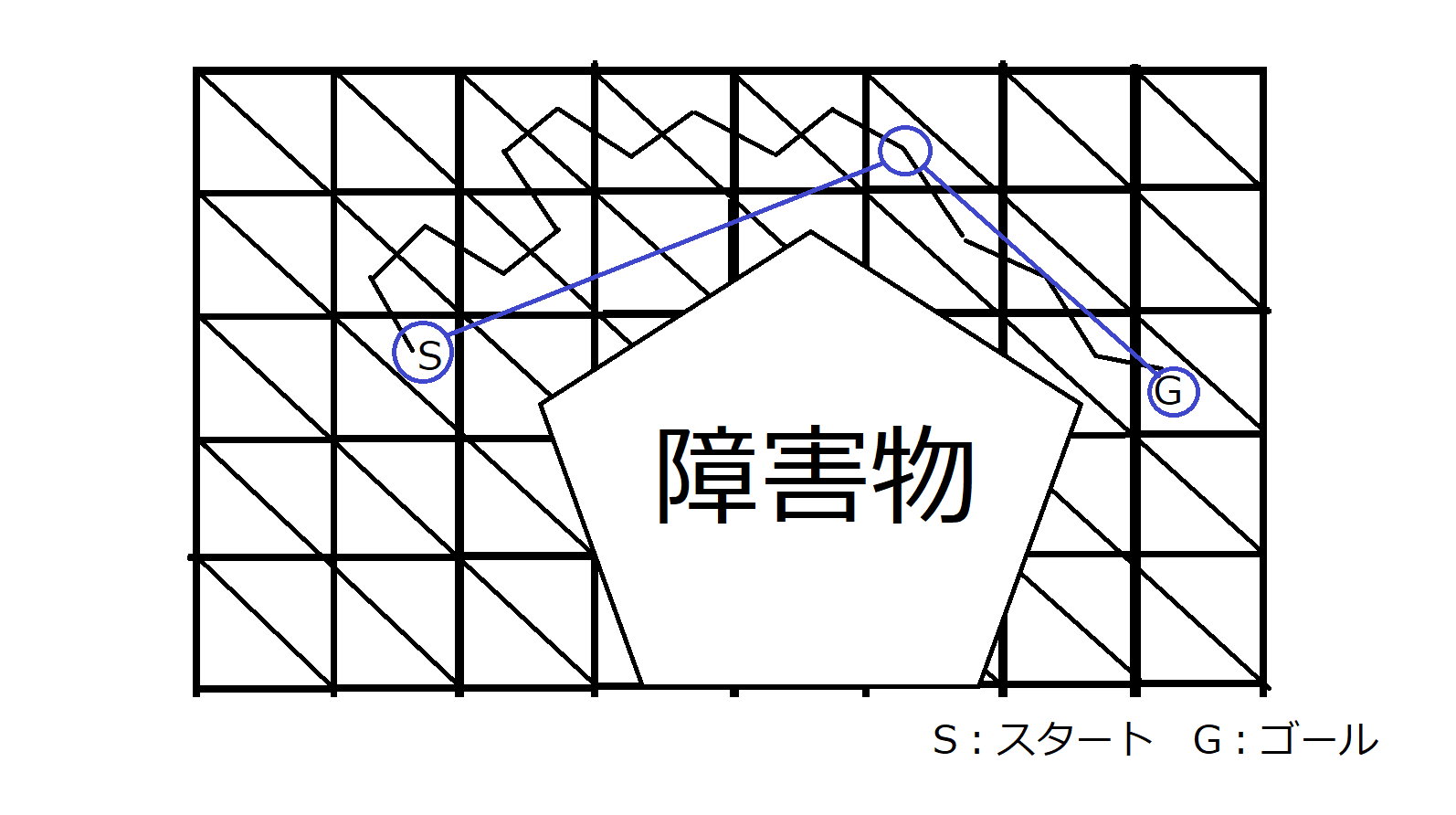
図３(経路のイメージ)

これを解消するために、経路のスムージングを行っています。

## **3.1 スムージングのアルゴリズム**

スムージングは今いる座標と次に進む予定の座標の次の座標の間(図４の赤い線分)で当たり判定をとって障害物がなければ次に進む予定の座標を進め、判定をとって当たるところまで繰り返し、当たったらループを抜け次行く予定の座標まで移動させたいものを進めることでスムージングをかけています。

図4

図５

この経路の場合だと最終的に通るルートは図５の青いラインになります。

# **４．アニメーションブレンディング**

学校内製の簡易エンジンにアニメーションを流す機能はもともとあったのですが、一つのアニメーションを流す機能しかなかったので、上半身と下半身で別のアニメーションを合成して流せるように改造しました。

最初は上半身と下半身で別々のアニメーションを流せるようにして実装していたのですが、学習成果発表会で、「アニメーションが上半身と下半身の境目で切り替わっているのがはっきりと分かって不自然」との指摘を受けたので、ブレンディングに重みをもたせることで、境目が自然になるように改良しました。

上半身と下半身のアニメーションを合成する際にボーンごとに上半身と下半身のアニメーションどちらがどれくらいの割合で影響するかの重みテーブルを作って合成することで上半身と下半身のアニメーションを自然に合成することができます。

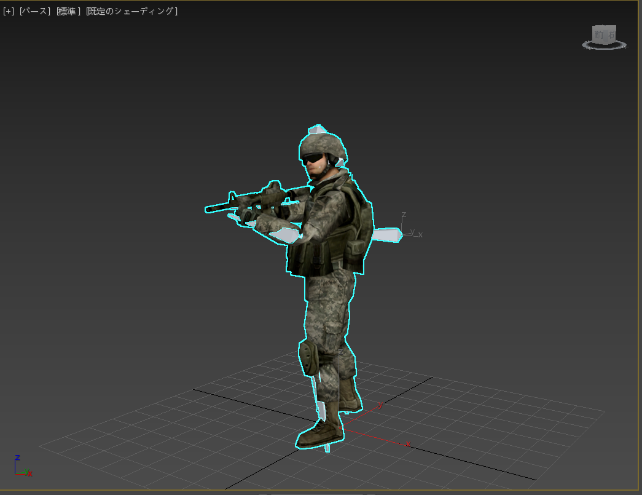
下記のような計算でブレンディングを行っています。

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

腰のボーン行列 = 歩きアニメーションの腰のボーン行列 × 0.6 + 銃を構えるアニメーションの腰のボーン行列 × 0.4

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

これをすることで銃を構えた状態で下半身だけ歩きアニメーションだったり銃を撃ちながら歩いたり止まったりするときに違和感がなくなるようになりました。



上半身と下半身で別のアニメーションを適用することで構えながら歩くことができる。

# **５．ポストエフェクト**

直接フレームバッファにレンダリングするのではなく、一度オフスクリーンレンダリングした画面をテクスチャ化して加工をおこなっています。

## **５.1 ブルーム**

ブルームは川瀬式ブルームを使用しています。

　まず一度シーン(図6)をレンダリングして、そこからカラーが強いところのみを輝度テクスチャ(図7)として抽出します。

図６



図7



図7は解像度が1280×720です。

そこからブラーをかけるためにダウンサンプリングしていきます。

バイリニアフィルタのサンプラステートを作成して、それを使って、テクスチャをサンプリングするようにしています。

ダウンサンプリング1回目(解像度640×360)



ダウンサンプリング2回目(解像度320×180)



ダウンサンプリング3回目(解像度160×90)



ダウンサンプリング4回目(解像度80×45)



そして、ダウンサンプリングしたテクスチャを合成。



それをシーンのテクスチャに加算合成することで完成します。



# **６．背景の映り込み**

リアルなグラフィックに近づけるために背景の映り込みを実装しました。

　空の光を反射している表現をしたかったのでモデルのスペキュラの強さに応じて空の光を反射しているように見えるようにしました。

図8(左映り込みなし、右映り込みあり)

図8の左右の画像を見比べるとわかるのですが、映り込みがある右側の画像のほうが空の色の影響を受けて、少し青白い印象を受けるようになりました。

これにより、屋外にいるように感じられるようになりました。