yuta yanagisawa

Portfolio

CONTENTS

PROFILE

名前

柳澤優太

生年月日

2001年6月14日

希望職種

プログラマー

メールアドレス

yuta20010614@gmail.com

自己PR

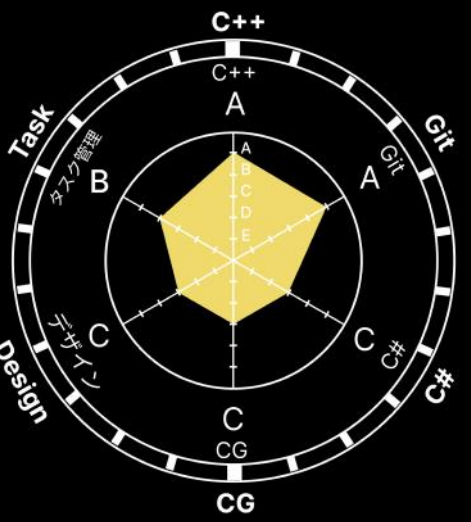
私は専門学校に入学してから、好奇心が強くなりました。独学で3DCGやエフェクトの制作方法を学び、専科ではWPFやAWSなどの知らない技術に触れる経験をしました。その経験の中で、こだわりや遊び心に力を入れることで楽しく学ぶことができました。また、学ぶときに気づいたことがあります。早く成長するには、何度も検証と失敗、反省を繰り返しこれを高速で回す。プログラミング、3DCGでは今の実力では難しいことをする必要が出てきます。そんなときに、何度も諦めずに挑戦し反省していれば必ず成長できると信じています。

趣味

- ギター
- プログラミング
- ポケモンカード
- 新しい技術やツールに触れてみる

Figma Notion NovelAI GitHub Jira make miro blender

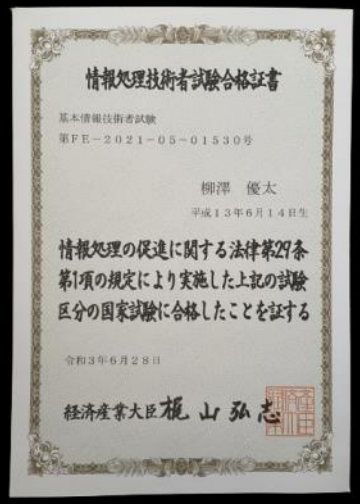
スキルグラフ



PROFILE

取得資格

2020年10月 J検 情報システム試験 基本スキル
2020年12月 CG エンジニア検定 ベーシック
2021年　1月 J検 情報活用試験 2級
2021年　3月 J検 プログラミングスキル
2021年　3月 J検 システムエンジニアスキル
2021年　6月 基本情報技術者試験


情報処理技術者試験合格証書

基本情報技術者試験
第ＦＥ－２０２１－０５－０１５３０号

柳澤　優太
平成１３年６月１４日生

情報処理の促進に関する法律第29条第1項の規定により実施した上記の試験区分の国家試験に合格したことを証する

令和３年６月２８日

経済産業大臣 梶山弘志


公益財団法人 画像情報教育振興協会認定
CGエンジニア検定 ベーシック

合格証

認定No.2013300749

柳澤　優太　殿

上記の検定に合格したことを証します。

合格認定日 2020 年 12 月 24 日

清口 稔

受賞歴

夏の3Dゲーム学内コンテスト
技術力賞

2022年の夏休み


夏の３Ｄゲーム
学内コンテスト
技術力賞

柳澤 優太 殿

受賞おめでとうございます。
優れた結果を得るためには、自らの行いの中に、創造性に溢れた活動をすること、確固たる思想を持って物事に前向きに取り組むこと、努力と研鑽を重ねることが大切です。
このたび、あなたは見事に、これらの条件を満たしました。
今日のこの感動を糧として、なお一層、今後の勉強に励み、目標を高く持って努力されることを希望し、表彰いたします。

2022 年 11 月 7 日

HAL東京

2021年


HAL EVENT WEEK
技創力賞
柳澤 優太

進級制作展(HAL EVENT WEEK)
独創力賞

2022年


わくわく！TPS ゲーム
学内コンテスト
独創力賞

柳澤 優太 殿

HAL東京

わくわく！TPSゲーム学内コンテスト
独創力賞

2022年


DX オリジナルゲーム
学内コンペ
構成力賞

柳澤 優太 殿

HAL東京

DXオリジナルゲーム学内コンペ
構成力賞

Air Shooting

弓矢を撃って敵を殲滅

縦横無尽に動く「お墓」に矢を命中させて、現世との縁を打ち破ろる！

神社風のステージ。
私の思うかっこよさに寄せたUIや見た目部分(シェーダーやモデル)をこだわりました

使用した技術・ツール

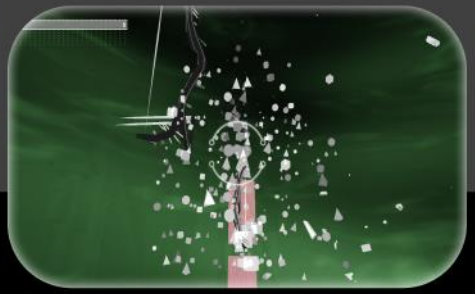
ツール

- DirectX11
- Blender
- ImGui


夏の3Dゲーム
学内コンテスト
技術力賞
HAL東京

技術

- Particle
- Component指向
- オフスクリーンレンダリング
- 環境マッピング



敵を倒したり、ジャンプをしたりするとパーティクル演出が発生。
また、ジャンプしながら弓を引くと「時」がゆっくりになって狙いを定めやすくしています。

ゲームへのリンク
https://drive.google.com/drive/folders/1gW3rdFAINvx4ex9QF544m1rS-k1thr5G?usp=share_

Air Shooting

CONCEPT

弓矢を撃って敵を殲滅

TARGET

TPSゲームが好き！

HOW

個人制作
期間：2022年4月～現在も継続

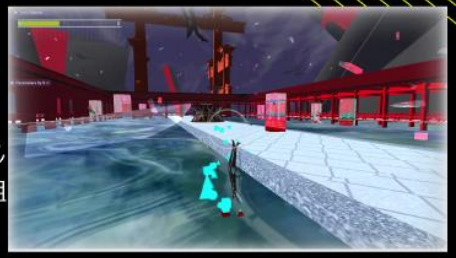
check!

きれいな見た目に注目！

PARTICLE

パーティクル

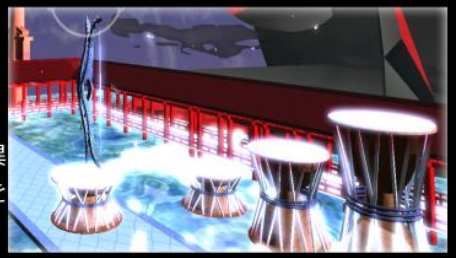
Unityのパラメータを模倣して作成しました。
プログラム面では、寿命に応じて、サイズや色を調整する仕組みをこだわりました



Bloom

ブルーム

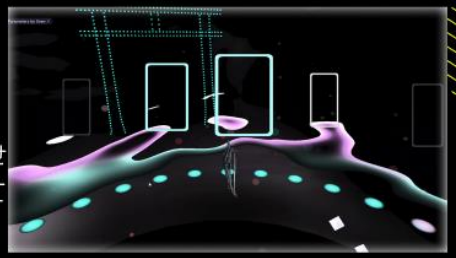
3Dゲームっぽい空間を表現したくて、MVを参考に作成しました。カメラの向きによって異なるシーン遷移をする仕組みを内積を使用して作成しました。



Shader

シェーダー

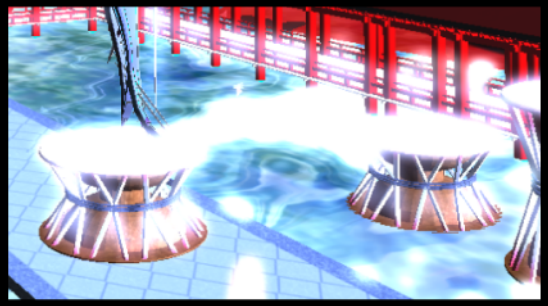
Noiseを使用したシェーダーを授業で教わる前に実装しました。距離とノイズをかけ合わせて、正規化して色を割り当てて制作しています。



Air Shooting

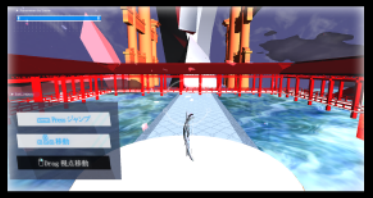
川瀬式ブルーム

この光が溢れ出ているような演出がブルームです

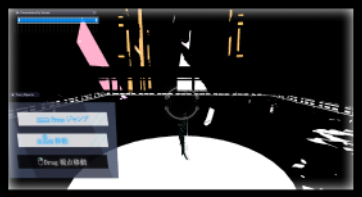


実装の流れ

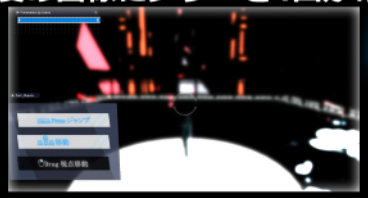
レンダリング結果をテクスチャに



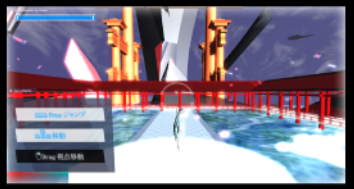
輝度抽出



輝度の画像にブラーを4回かける



元の画像に加算合成



苦労した点

ブルームの処理を作成するにあたり、【HLSLシェーダーの魔導書】という書籍を参考にしました。
しかし、この書籍ではDirectX12を使用し、筆者独自のエンジンを使用していたため、自分が使用しているDirectX11に置き換え、理屈をしっかりと理解しないと実装できない点が大変でした。↓のように言語化することで、実装ができました。

どう実装するか

- 今使ってるRTV,SRVを配列にする
- 輝度抽出をするのは一回だけ
  - ブラーをかけるのが4回
  - その時の解像度を変える？？？
  - ブラーをかけたテクスチャにさらにブラーをかける・・・

懸念点

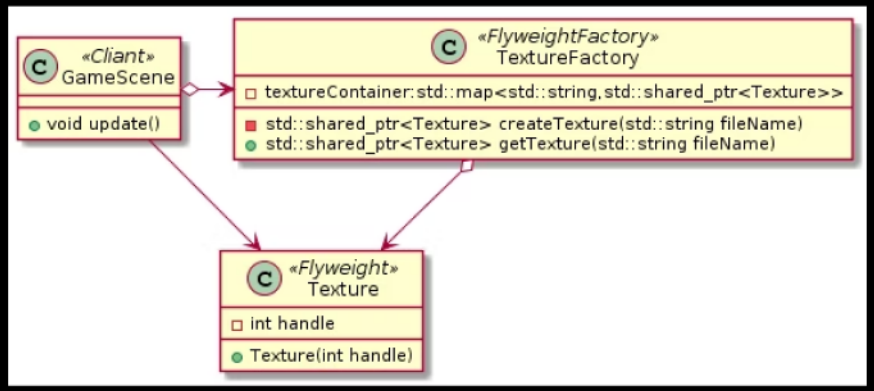
- ブラーを複数回かけることに対応させなきゃいけない
  - 関数化してしまう。
- レイヤーで描画順を操作していたのが問題。
  - ブラーをかける処理を一つのレイヤーにまとめる。
    - いけそう
  - 現状、ブラーX,ブラーYで分かれているのでそれを一つのレイヤーにしてしまう。
    - 順番はAddGameObjectでかえればOK
  - ちょい雑だけど、ブラーを統一するのに加えて、Layerを4つ増やす。
    - LAYER_BLUR_XとLAYER_BLUR_YをLAYER_BLURに統一
    - LAYER_BLUR_1,LAYER_BLUR_2,LAYER_BLUR_3,LAYER_BLUR4を追加
- そもそも、ブラーをかける処理を一つにまとめるべき
- ブラーレイヤーの中で全部やっちゃえばいいのでは

Air Shooting

コード面の工夫

同じリソースを使い回す！リソース管理クラス

メモリの使用量を抑える



一度ロードしたリソースは、
二度目以降同じリソースを取得させる

Template化して
- **Texture**
- **Model**
- **Shader**

などのリソースをこのクラスで管理しています

シーンを跨ぐときや、オブジェクトを複数使用するときに、ロードする数を最小限に抑えます

Flyweightパターンでの実装（Template）

```
/ 参考:https://qiita.com/FlatMountain/items/0b446900ccd058ec9d43
emplate <typename T>
lass ResourceManger {
rivate:
    // Resourceを管理するコンテナ
    static inline std::map<std::string, std::shared_ptr<T>> _container;

    // Resourceの生成
    static std::shared_ptr<T> CreateResource(std::string fileName) {

        std::shared_ptr<T> newResource = std::make_shared<T>();
        newResource->Load(fileName.c_str());

        return newResource;
    }

ublic:
    // Resourceの取得
    static std::shared_ptr<T> GetResource(std::string fileName)
    {
        // ファイル名で検索
        auto it = _container.find(fileName);

        // コンテナ内にあれば、そのリソースを返す
        if (it != _container.end()) {
            return it->second;
        }

        // コンテナ内になければ、新しくリソースを作成してコンテナに登録
        std::shared_ptr<T> newResource;
        newResource = CreateResource(fileName);
        _container.insert(std::make_pair(fileName, newResource));

        return newResource;
    }

    // ゲーム終了時にReleaseする
    static void AllRelease() {
        for (auto resource : _container) {
            resource.second->Unload();
        }
        _container.clear();
    }
;
```

Air Shooting

開発環境面の工夫 JiraとGitの連携

タスクをJira上で作成する

Jiraはブラウザ上で動くツールなので、タスク一つひとつをURLとして管理することができます。

コメント・Gitのコミットで URLを貼る

```
/**
 * ----------------------------------------------------------------------------
 *  [chain_of_responsibility.h]
 *                                      @author : yuta yanagisawa
 *                                      @date   : 2023/1/3
 * ----------------------------summary-----------------------------------------
 * @brief コマンド機能の実装
 *
 * @link https://yuta6686.atlassian.net/browse/AS-8
 ** ----------------------------------------------------------------------------*/
```

```
https://yuta6686.atlassian.net/browse/AS-48 ブラーのかけ方は間違ってなかった。けど、その後の描画部分... yuta6686 2023/01/2... 30413443
https://yuta6686.atlassian.net/browse/AS-48 ブラー4回重ねがけできない。このままだと、先に進まないので... yuta6686 2023/01/2... 9724d13b
https://yuta6686.atlassian.net/browse/AS-45 yuta6686 2023/01/2... ac5ad887
https://yuta6686.atlassian.net/browse/AS-45 CopySRV,RTV 配列対応 yuta6686 2023/01/2... 8fe7f76d
https://yuta6686.atlassian.net/browse/AS-45 コピーもBLURにまとめた yuta6686 2023/01/2... 6a2ae339
https://yuta6686.atlassian.net/browse/AS-45 BLUR_XとBLUR_Yのまとめまでと Debug_sence でStealth作... yuta6686 2023/01/2... 0e6c1302
https://yuta6686.atlassian.net/browse/AS-45 LAYER_BEGINだけSwitchingRendererから出せない謎 yuta6686 2023/01/2... 053f53e3
解像度変える変数場所移動 develop feature/AS-15-Bloom master yuta6686 2023/01/2... 17e1f32c
```

コード上にドキュメントとして残すより、URLで飛んで詳細を確認しに行くほうが、見やすい

タスクをURLとして管理することのメリット

- 後で見返しやすい
- コード上の無駄なコメントを減らせる
- コード上のコメントでは使えないような
  マークダウンや画像の記述が可能

Hop Slime

「反発」で跳ね回る！

触れたブロックに応じてジャンプ力が変化する！これを利用してゴールを目指そう！

スライムの「感触」をちょうどいい程度に伝えられるようにエフェクトを作成しました

使用した技術・ツール

ツール

- Unity

技術

- Particle
- Shader Graph

←どのブロックが反発するのか？をわかりやすくするために、Shader Graphを用いてブロックの枠を光らせました。

ゲームへのリンク
https://drive.google.com/drive/folders/1gW3rdFAINvx4ex9QF544m1rS-k1thr5G?usp=share_

Hop Slime

CONCEPT

スライムが反発移動を駆使して大冒険！！

TARGET

男女問わず、2Dアクションゲームやきれいなグラフィックが好きな人や新鮮なゲームをやりたい人

HOW

使用ツール↓

- Unity

制作期間↓

- 2022/3/1 ~ 5/31

制作人数↓

- 10人(プログラマ×3,プランナ×1,デザイナ×3)

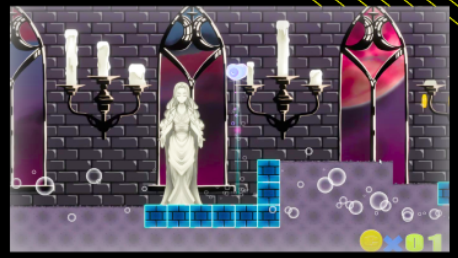
check!

きれいな見た目に注目！

EFFECT

エフェクト

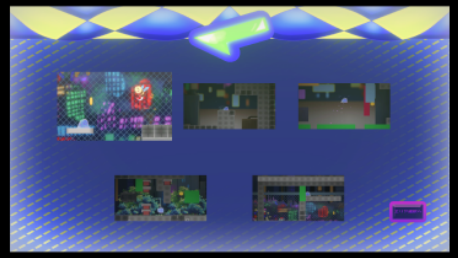
プレイヤーの軌跡・着地のエフェクトを特にこだわりました。
着地エフェクトのバブルはUI上にも表示し、着地するときの速度に応じてバブルの量も変化します。



STAGE SELECT

ステージ選択

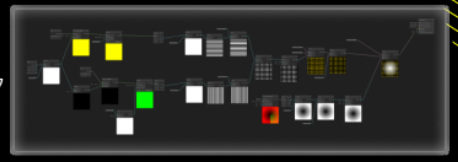
シェーダーグラフを使用して背景しを作成し、ステージのボタンの制御やアニメーション、シーン遷移を作成しました。



SHADER

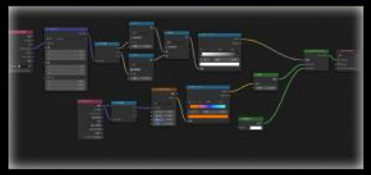
シェーダー

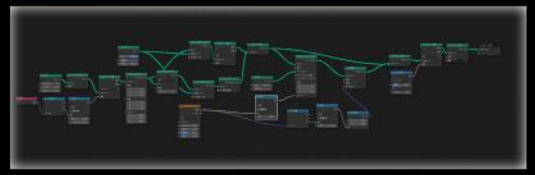
上記のステージ選択の背景をシェーダーグラフで作成しました。
波線のUVスクロールと中心からのマスク処理のみですが、いい感じにできました。



Other

Blender





ノードベースでの「モデリング」と「シェーディング」

左はシェーダーをノードで作成するもので、右はジオメトリをノードで作成するものです。
この作品では「ノードで作る」ことにこだわりました。ノードで作ることで、全体的なモデルをすぐに変更することができ、調整を繰り返しながら作成しました。また、プログラム的な考えにも応用できるので一石二鳥で学ぶことができました。

ダークリパルサー&エリュシデータ

#キリト #エリュシデータ #ダークリパルサー #blender #CG #SAO100users入り +

286 392 2,281

2020年11月1日 20:15

自分の【好きなもの】を作る楽しさ

Blenderを始めて、約4ヶ月が経過して作成しました。ソードアート・オンラインのキリトが使用している剣です。背景は海外の方のチュートリアルを見ながら作成し、自身でもライティングなどを工夫して撮影しました。この作品をPixivに投稿したら、そこそこいいねを頂いていて、自分の作ったものが評価されることに嬉しさを感じました！

その他作品

~Fin~