

持続ジャンプ

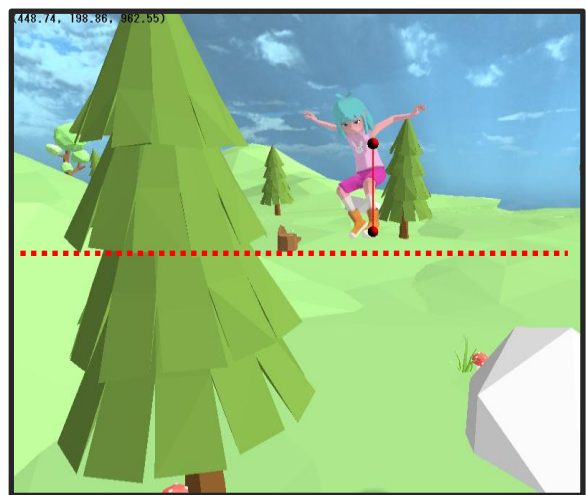
ASO Rockman同様に、ジャンプキーを入力し続けると、ジャンプ力が上がる持続ジャンプを実装していきます。

ジャンプの入力受付時間を数～数十フレームとって、その間、ジャンプ力を与え続けます。

初期のジャンプ力と、持続のジャンプ力は定数として分けておくことで、バランス調整しやすくします。



最小ジャンプ



最大ジャンプ

```
CharactorBase.h
```

```
protected:
```

```
    ～ 省略 ～
```

```
    // ジャンプの入力受付時間
```

```
    float stepJump_;
```

```
    ～ 省略 ～
```

```
CharactorBase.cpp
```

```
void CharactorBase::CollisionGravity(void)
```

```
{
```

```
    ～ 省略 ～
```

```
        if (hit.HitFlag > 0)
```

```
        {
```

```
            // 衝突地点から、少し上に移動
```

```
            transform_.pos =
```

```
                VAdd(hit.HitPosition, VScale(AsoUtility::DIR_U, 2.0f));
```

```
            // ジャンプリセット
```

```
            jumpPow_ = AsoUtility::VECTOR_ZERO;
```

```
            // ジャンプの入力受付時間をリセット
```

```
            stepJump_ = 0.0f;
```

```
            // ジャンプ判定
```

```
            isJump_ = false;
```

```
        }
```

```
    ～ 省略 ～
```

```
}
```

Player.h

private:

～ 省略 ～

// ジャンプ力

static constexpr float POW_JUMP = 5000.0f;

static constexpr float POW_JUMP_INIT = 3500.0f;

⇒ 削除
POW_JUMP_INIT
に名前変更

// 持続ジャンプ力

static constexpr float POW_JUMP_KEEP = 400.0f;

// ジャンプ受付時間

static constexpr float TIME_JUMP_INPUT = 0.5f;

Player.cpp

void Player::ProcessJump(void)

{

auto& ins = InputManager::GetInstance();

// 持続ジャンプ処理

bool isHitKeyNew = ins.IsNew(KEY_INPUT_BACKSLASH)

|| ins.IsPadBtnNew(

InputManager::JOYPAD_N0::PAD1, InputManager::JOYPAD_BTN::DOWN);

if (isHitKeyNew)

{

// ジャンプの入力受付時間を減少

stepJump_ += scnMng_.GetDeltaTime();

if (○○○)

{

// ジャンプ量の計算

float jumpSpeed = POW_JUMP_KEEP * scnMng_.GetDeltaTime();

jumpPow_ = VAdd(jumpPow_, VScale(AsoUtility::DIR_U, jumpSpeed));

}

}

else

```

{
    // ボタンを離したらジャンプ力に加算しない
    ○○○
}

// 初期ジャンプ処理
bool isHitKey = ins.IsTrgDown(KEY_INPUT_BACKSLASH)
    || ins.IsPadBtnTrgDown(
        InputManager::JOYPAD_N0::PAD1, InputManager::JOYPAD_BTN::DOWN);

// ジャンプ
if (isHitKey && !isJump_)
{

    // ジャンプ量の計算
    float jumpSpeed = POW_JUMP_INIT * scnMng_.GetDeltaTime();
    jumpPow_ = VScale(AsoUtility::DIR_U, jumpSpeed);
    isJump_ = true;

    // アニメーション再生
    animationController_->Play(
        static_cast<int>(ANIM_TYPE::JUMP), false);

}

}

```

【要件】

○○○を埋めて、持続ジャンプ機能を完成させること。

【目標】

持続ジャンプ機能がでること。