作品ポートフォリオ

大阪情報コンピュータ専門学校

総合情報メディア学科 ３年

井上　颯騎

|  |  |
| --- | --- |
| 作品タイトル　M | |
| ジャンル | リズムゲーム |
| 製作期間 | 2020年6月~9月 |
| 製作人数 | 1人 |
| 開発環境 | Visual Studio 2017, 学校提供ライブラリ, picojson, C++ |
| ゲーム概要 | Midiファイルから譜面を作成し、Midiキーボードで本格的に遊べるリズムゲーム。  Midiファイルのトラックごとに音色を変更でき、遊ぶ以外のトラックはオートで演奏されます。 |

* ゲーム画面

|  |  |
| --- | --- |
|  | グラフィカル ユーザー インターフェイス, アプリケーション  自動的に生成された説明 |
|  |  |

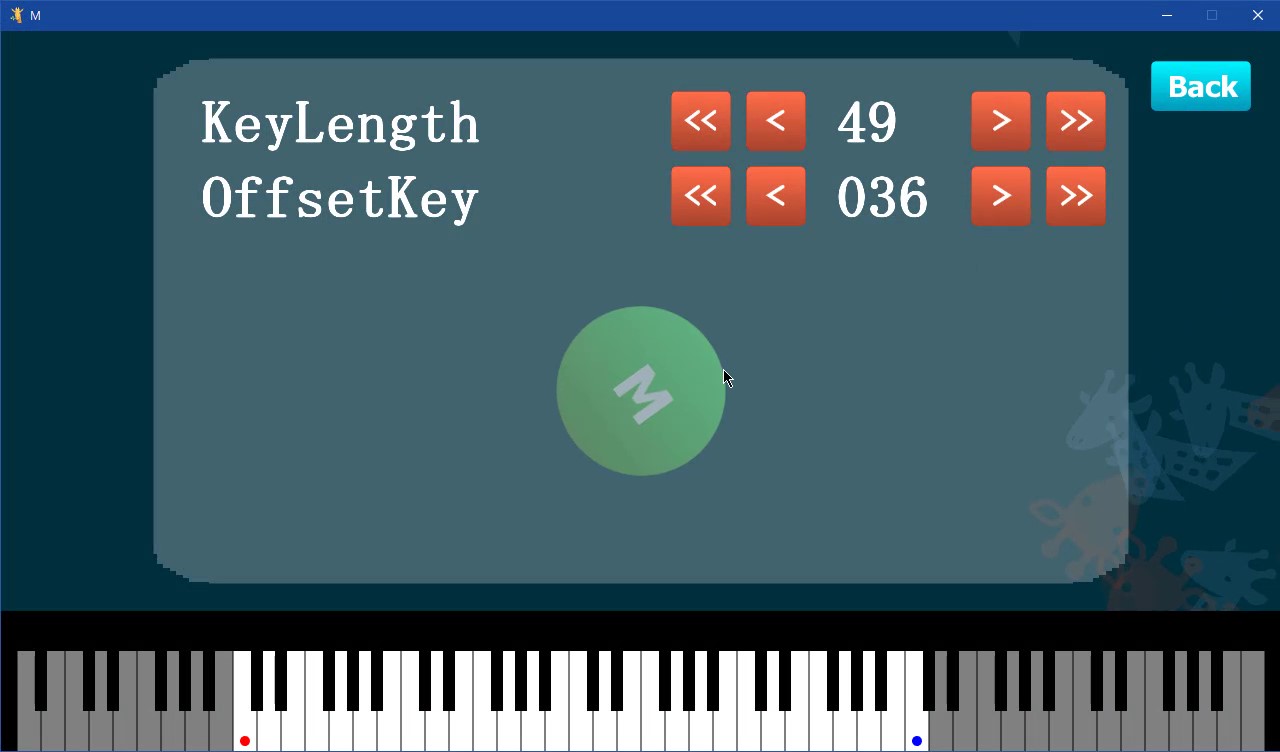
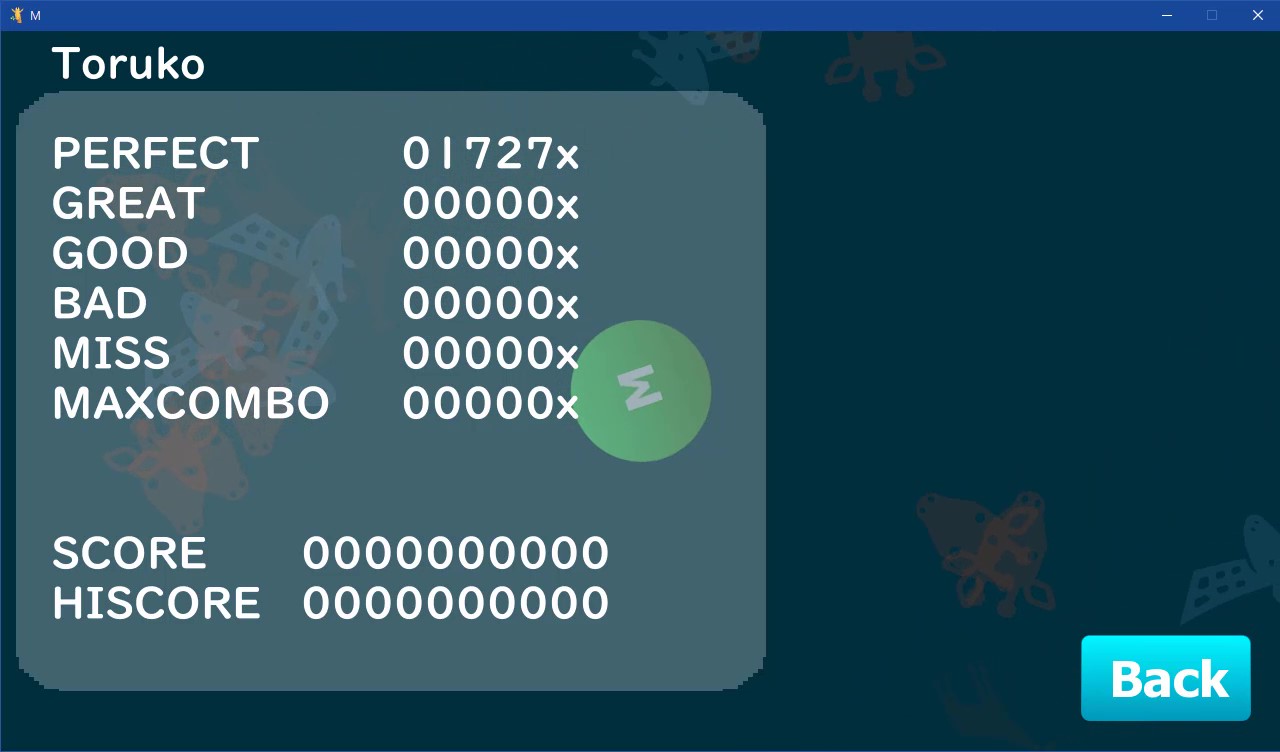
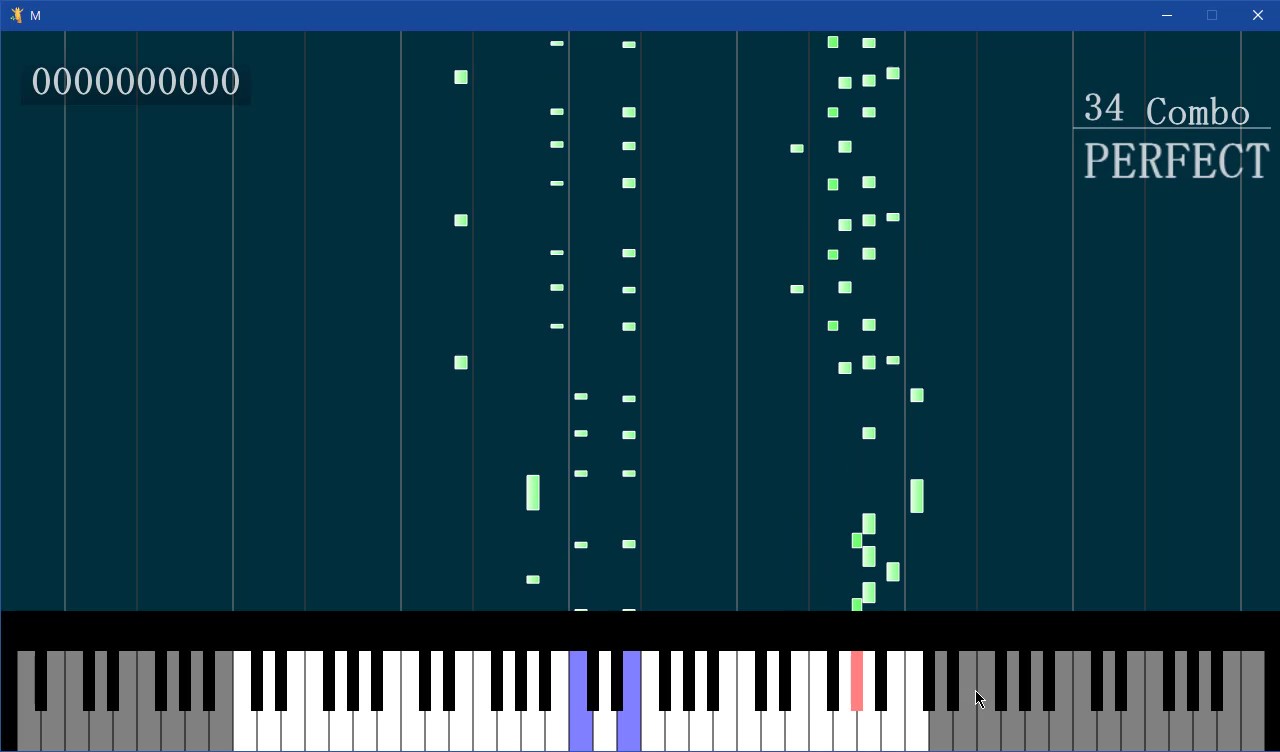
* 操作方法



・マウス

・Midiキーボード

* 画面遷移



タイトル

曲選択

設定

ゲーム画面

リザルト

* 画面UI

遊べる曲一覧

スクロール用ボタン

曲詳細



設定へ遷移するボタン

選択した曲をプレイするボタン



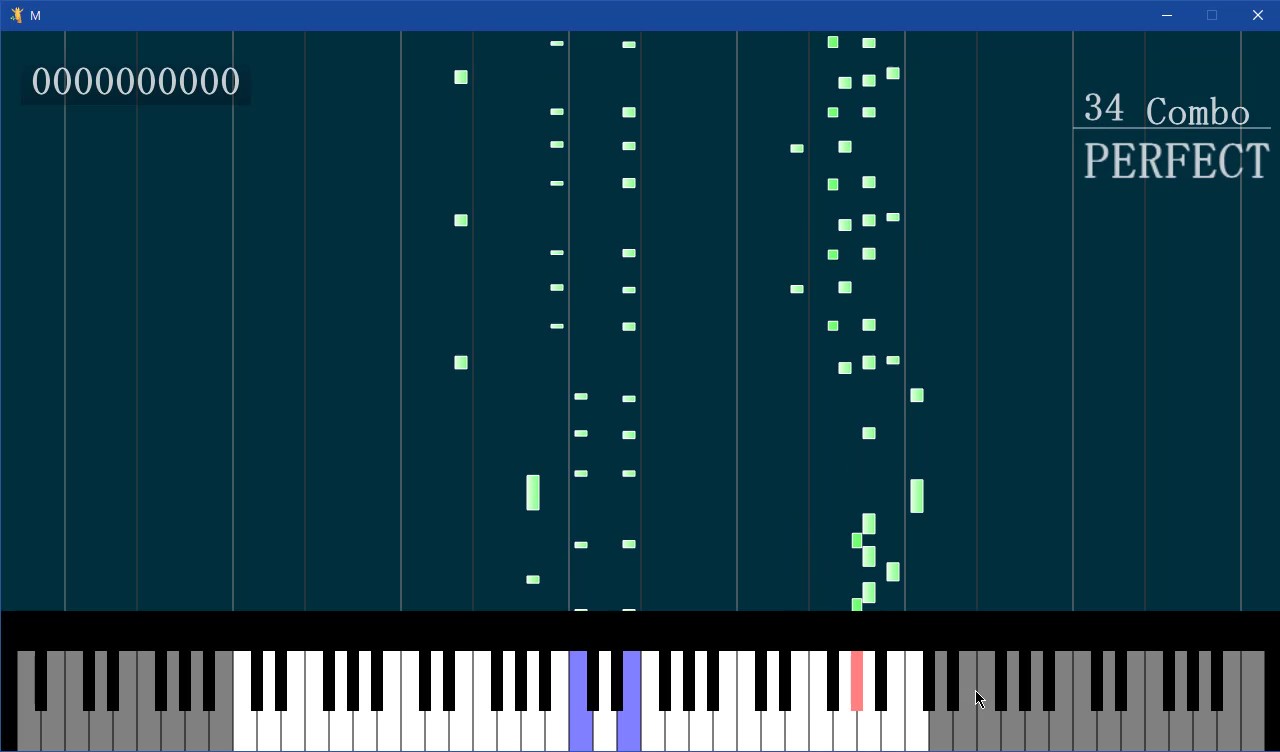
プレイヤーの記録

演奏するトラック選択

選択しているトラックの情報

曲選択画面

* 画面UI



現在のスコア

演奏する譜面

現在のコンボ数

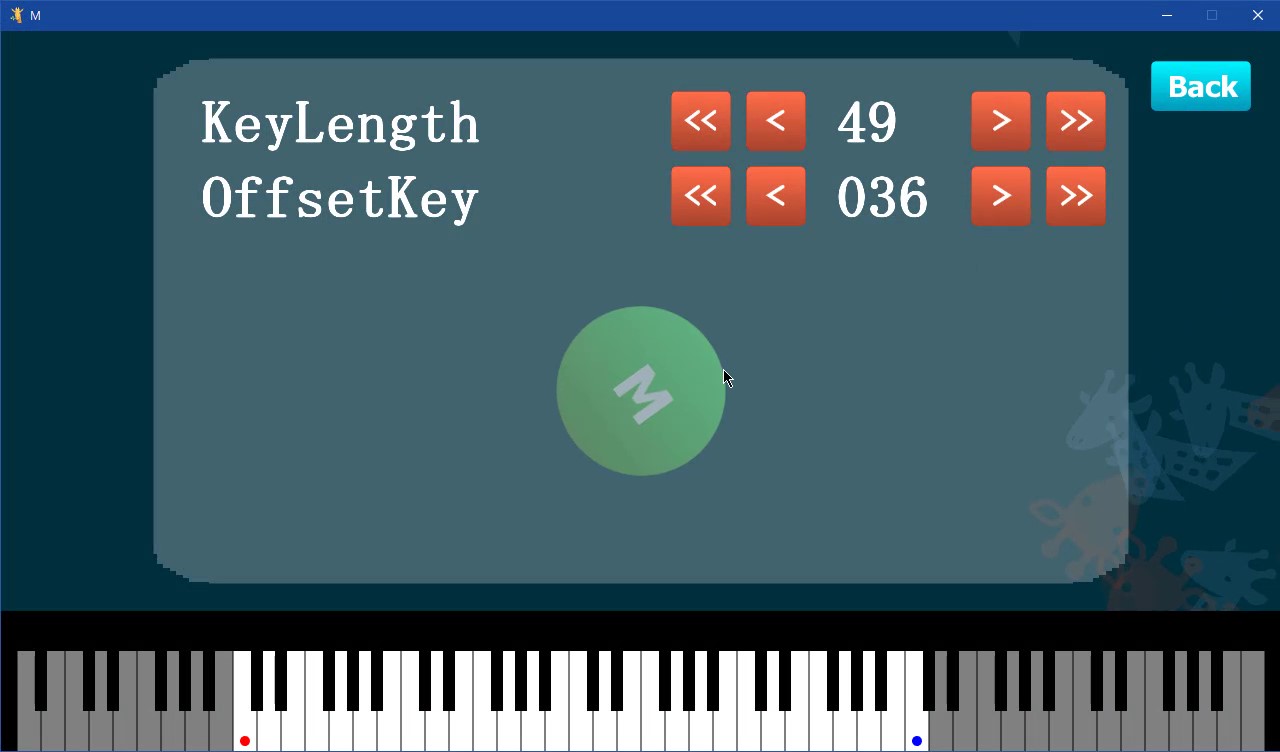
ゲーム画面

判定結果

判定ライン

プレイヤーが操作する範囲

設定



操作する範囲の基準位置

自分が操作するキーボードの長さ

* 譜面データは、一般的なMIDIファイルを利用

音楽ファイルを扱う中で簡単に編集ができ、広く普及されている規格がSMF(Standard Midi File)だったため採用しました。

* MIDIファイルの読み込み

「MIDI 1.0 規格書」を参考にSMFのFormat0, Format1のみ対応して読み込むようプログラムを組みました。

Format2を採用しなかった理由は現在ほとんど使われていないためです。

* MIDIデータを譜面に置き換え

SMFデータを譜面にするうえで必要のないデータが多くあるので処理速度を考慮して、すべては読み込まず、必要な部分のみ読み込むようにしました。

* 曲の管理は拡張性の高いjsonファイルを利用

拡張性を考慮して曲を追加していけるようにjsonファイルから読み込むように作りました。

jsonファイルを採用した理由は、広く普及しているフォーマットなので情報が多く扱いやすかったからです。

* 演奏するトラックの設定

演奏するトラックを選択できるようにして、左手用譜面であったり、メインメロディのみの譜面であったりを演奏できるようにjsonファイルの項目に演奏するトラックの項目を用意しました。

* トラックごとに音色の設定

Windowsで扱える128種類の音色をトラックごとに設定できるようにjsonファイルの項目に用意しました。

* 演奏範囲の設定

演奏するためのMidiキーボードですが、鍵盤数が25,37,49,61,88の5種類あるため、88のフル鍵盤以外の場合に演奏ができない範囲が出てきます。

そのため、設定された範囲以外では自動で演奏するようにプログラムしています。

さらに、設定を変更できるように設定画面でキーボードの長さと操作する範囲の基準位置を設定すればプレイヤーが演奏する範囲を設定することができます。

|  |  |
| --- | --- |
| AniMake | |
| ジャンル | ツール |
| 製作期間 | 2020年4月～6月 |
| 開発環境 | Windows, Visual Studio 2019, OpenSiv3D(ver.0.4.3), C++ |
| 概要 | このツールは学内ライブラリ用の２Dアニメーション制作補助ツールです。  学内ライブラリでアニメーションを作る際、可視化できなかったため作りづらいと感じ製作しました。 |

* ツール画面

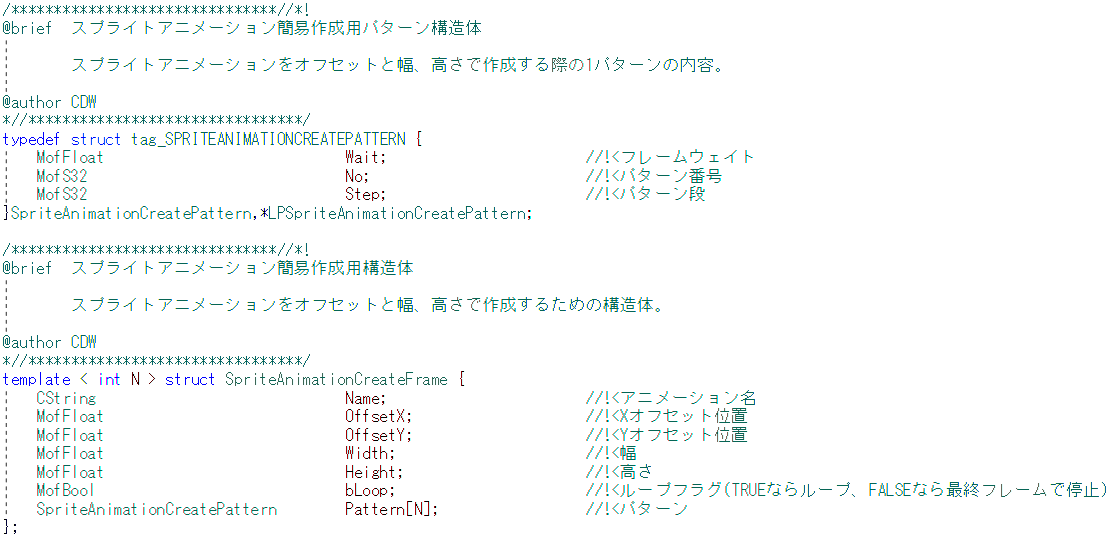
|  |  |
| --- | --- |
| グラフィカル ユーザー インターフェイス  自動的に生成された説明 | グラフィカル ユーザー インターフェイス  自動的に生成された説明 |
| グラフィカル ユーザー インターフェイス, アプリケーション  自動的に生成された説明 | グラフィカル ユーザー インターフェイス, アプリケーション  自動的に生成された説明 |

* 操作方法

・マウス

・キーボード

* アニメションデータ構造



このツールでは1枚の画像から矩形を作成しトリミング描画することでアニメーションをします。

* 画面UI

グラフィカル ユーザー インターフェイス

自動的に生成された説明

メインウィンドウです。ここではアニメーションや画像の情報を確認することができます。

グラフィカル ユーザー インターフェイス

自動的に生成された説明

アニメーションデータを設定する場所です。

ここで矩形のサイズや位置、パターンの編集をすることができます。

追加のボタンを押すとアニメーションリストにアニメーションデータが追加されます。

グラフィカル ユーザー インターフェイス

自動的に生成された説明

この場所でできること

・カレントディレクトリの変更

・アニメーションファイルの保存

・アニメーションファイルを開く

・画像ファイルを変更

・アニメーションパターン数の変更

・全パターン待機フレーム一括変更

・背景の色を変更

・グリッドの表示

・version情報の確認

です。

|  |  |
| --- | --- |
| MofSipLib | |
| ジャンル | 拡張ライブラリ |
| 製作期間 | 2020年4月～ |
| 開発環境 | Windows, Visual Studio 2017, C++, 学内提供ライブラリ |
| 概要 | この拡張ライブラリは学内提供ライブラリの拡張ライブラリです。  ゲームを作っていくうえで使いまわせるようなパーツを使いやすくまとめています。 |

* 実装クラス

|  |  |
| --- | --- |
| クラス | 内容 |
| CAssetBase< Key , T > | unorder\_mapを使用したアセット管理用汎用クラス |
| CEaseMotion< T > | 開始値から終了値まで指定した時間でEaseを使って補間するクラス |
| CEaseMotionController< T > | CEaseMotion<T>クラスを複数設定できるようにしたクラス |
| CHSVUtilities | 学内提供ライブラリではARGBの指定でしか色指定ができないためHSVで指定できるようにHSVからARGBへの変換する汎用スタティッククラス |
| CMyTime | プログラム開始時間を記録し、起動してからの経過時間を計測するスタティッククラス |
| CPeriodic | 周期的な動きをする計算をまとめた汎用スタティッククラス |
| CResourceFont | 学内提供ライブラリではWindowsにインストールされているフォントしか使えないため、ttfなどのフォントファイルからフォントデータを生成するクラス |
| CRoundRect | 角丸矩形を表現するためのクラス |
| CStopWatch | 時間の計測を行うクラス |
| CAnimationData | AniMakeで出力した.animファイルを読み込み、アニメーションデータに変換するクラス |
| CEffectBase | パーティクルエフェクト用ベースクラス |
| CEmitter<PT<T>> | パーティクルの各情報の最大、最小値をテンプレート化した放出機クラス |
| CParticle | パーティクルクラス |
| CSceneManager<State, Data> | シーンの追加や遷移を楽にでき、各シーンでの共通データの管理をするクラス |
| CToonShader | 学内提供ライブラリで用意されているトゥーンシェーダーを簡単に適応できるようにしたクラス |
| CFreeCamera | 3Dデバッグ用のシーン内を簡単に操作できるカメラクラス |
| CFileDialog | Tool用File操作関連を簡単に操作できるようまとめた汎用スタティッククラス(一部c++17の機能を使用) |
| CMyCamera | 3D用カメラの設定を楽に設定できるようにしたカメラクラス |
| CMyCamera2D | 2D用に疑似カメラを扱えるようにしたクラス |
| CCursor | マウスカーソルの見た目を簡単に変更できるようまとめたスタティッククラス |
| CSipFrameworkThread | 拡張ライブラリ用の初期設定やプロシージャの処理、解放処理を意識しないでいいように隠すためのスレッドクラス |

* 工夫した点

拡張ライブラリにまとめるときに使いまわしをしやすいように汎用的になるようにしました。

CSceneManagerでは各シーン共通データをテンプレート化し、ゲームごとに変更できるように工夫しました。ゲームの遷移に多用するフェード処理を内部で追加し、シーンの遷移時にフェードイン、アウトにかかる時間を指定できるようにしました。

Ease計算式を10パターン用意し、それぞれにEaseIn, EaseOut, EaseInOutの３パターン用意し、合計30パターンの表現が簡単にできるようにしました。

* 広がる表現の幅

HSVで色指定をできるようにすることで変数を使って簡単に表現できる色の幅が広がります。

さらに、CPeriodicクラスやEaseと組み合わせることで感覚的に変化を作ることができます。

　CEaseMotionクラスの扱う値をテンプレート化したことにより、四則演算が可能なクラスであれば指定したEaseの種類に対応して補間がされ、簡単に動きを作ることができるようになりました。

例えば、敵の移動先を作成するとき、CEaseMotionControllerで開始値と終了値までをn秒で移動するように指定し、複数の中間点と中間点毎にEaseの種類を設定すれば感覚的に作成できることができます。

これを利用し、シネマ用のカメラワークなども作成できます。