FaceSync 概要

目次

1 概要

FaceSync はカメラ入力をゲームに応用する事例の一つとして、カメラ入力とグラフィックス処理の組み合わせを実現したサンプルプログラムです。PlayStation®Eye の入力画像から顔を検出して性別や年代などの属性と表情を抽出し、抽出したそれらのパラメータを 3D モデルデータに反映してリアルタイム描画します。

図 1　FaceSync の描画例

2 特徴

機能の特徴

FaceSync サンプルでは、libface による顔認識の結果を使って、アニメーションに反映しています。認識しているパラメータには以下のものがあります。

- 顔の位置
- 顔の向き
- 性別・年代判別
  - 成人男性→"恐竜"モデル
  - 成人女性→"牛"モデル
  - 子供→"子供"モデル
- 表情判別
  - 標準
  - 左目閉
  - 右目閉
  - 口開閉
  - 笑顔

FaceSync サンプルでは、メニューまたはコントローラから以下の振る舞いを変更できます。

- PlayStation®Eye の機能
- 3D モデルデータ種別 (Simple, Kyouryuu, Cow, Child, Duck)
- 安定化フィルタ

トラッキング手法の特徴

- SDK で提供されている顔認識ライブラリには、強力な顔検出器である libface と、顔の向きにロバストに追跡可能な libhead_tracker があります。
- 高い時間・位置精度の必要な用途（視点位置の検出や、検出した顔への高精度のペイント）には、テンプレートマッチングや KLT などのトラッキング手法が適している場合があります。顔認識ライブラリは、これら追跡アルゴリズムに対し、初期追跡位置情報を提供することができます。
- FaceSync サンプルでは、全体サーチとローカルサーチの 2 つの顔検出法による高速検出アルゴリズムを使っています。

内蔵 3Dモデルデータ

図 2　Simple モデル

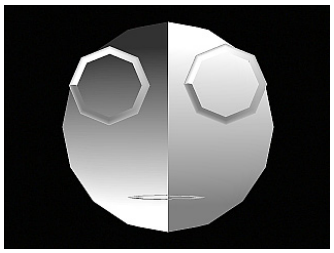


図 3　Kyouryuu モデル

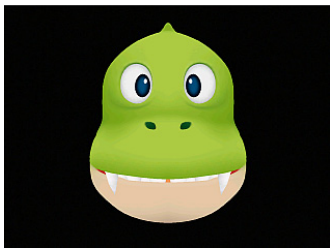


図 4　Cow モデル

図 5　Child モデル

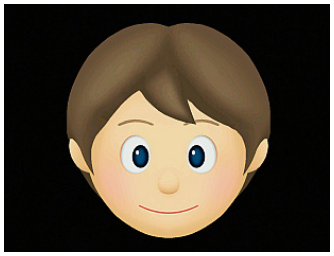
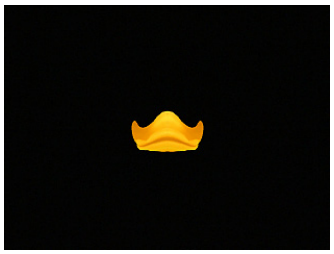


図 6　Duck モデル



スクリーンショット

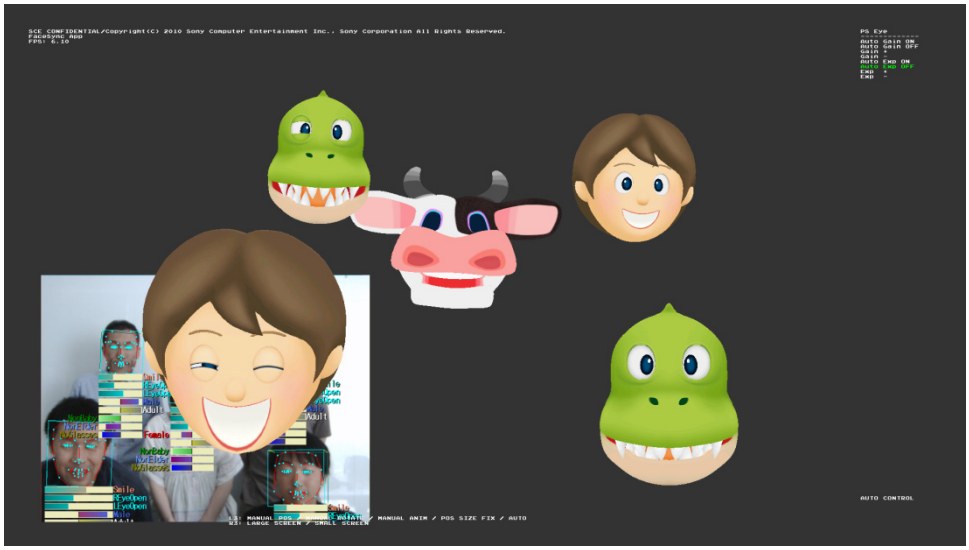
例 1

図 7　スクリーンショット 1

この例では、人間 4 人と写真に写った赤ちゃんの顔を認識し、それらの位置・向き・表情を反映した CG キャラクタを入力映像に重ね合わせて描画しています。

例 2

図 8　スクリーンショット 2



この例では、人間 5 人の顔を認識しています。
入力映像を左下に表示し、CG キャラクタを描画しています。

3 実行方法

Visual Studioによるコンパイルと実行

Visual Studio を起動し、サンプルに含まれる FaceSync.sln を開いてコンパイルしてください。コンパイルしてできた SELF ファイルを Reference Tool で実行してください。

HDD起動ゲームパッケージの作成手順

PlayStation®3 で実行できる HDD 起動ゲームパッケージを作成するには、以下の手順を実行してください。

(1) face_sync_macro.h の先頭にある#if 0 を 1 に変更してから、プログラムをコンパイルします。HDD 起動ゲームでは/app_home が使えなくなりますので、絶対パスで /dev_hdd0/game/FSYN00001/USRDIR を使っています。

(2) pkg/以下がパッケージングされるフォルダです。data/以下のモデルデータを pkg/USRDIR/以下にコピーします。

(3) 以下のコマンドにより、.elf ファイルから EBOOT.BIN をつくります。

```
FaceSync > copy src/PS3_PPU_Release/FaceSync.ppu.elf
FaceSync > make_fself_npdrm.exe FaceSync.ppu.elf EBOOT.BIN
FaceSync > move EBOOT.BIN pkg/USRDIR
```

(4) 以下のコマンドにより、HDD 起動ゲームパッケージをつくります。

```
FaceSync > make_package_npdrm.exe pkg.cfg pkg
```

以上で、HDD 起動ゲームパッケージが作成できます。インストールは、USB メモリ経由か、ProDG Target Manager for PlayStation®3 で接続したうえ/app_home 経由で行います。

4 操作

準備

PlayStation®Eye を接続し、顔をカメラに映してください。

メニュー操作

コントローラのボタン	説明
↑	メニューカーソルの移動
↓	メニューカーソルの移動
○	メニューの決定
×	メニューの決定

メニュー構成

PlayStation®Eye の設定

項目	説明
Auto Gain On	自動ゲイン機能を使う
Auto Gain Off	自動ゲイン機能を使わない
Gain+	カメラのゲイン値を増やす
Gain-	カメラのゲイン値を減じる
Auto Exp On	自動エクスポージャ機能を使う
Auto Exp Off	自動エクスポージャ機能を使わない
Exp+	カメラのエクスポージャ値を増やす
Exp-	カメラのエクスポージャ値を減じる

3D モデルデータの設定

項目	説明
Auto Model	顔認識結果を適用してモデルを自動で選択する
Simple	単純モデルを適用する
Kyouryuu	恐竜モデルを適用する
Child	子供モデルを適用する
Cow	牛モデルを適用する
Duck	アヒルのくちばしモデルを適用する

安定化フィルタの設定

項目	説明
Stabilize Filter On	安定化フィルタを使う
Stabilize Filter Off	安定化フィルタを使わない
Sync LR Eyes On	両目開閉の同期を行う
Sync LR Eyes Off	両目開閉の同期を行わない

その他の操作

拡大縮小

コントローラの R3 ボタンを押すと、カメラ入力画像を拡大表示するか縮小表示するかの切り替えができます。

図 9　カメラ入力画像の拡大表示

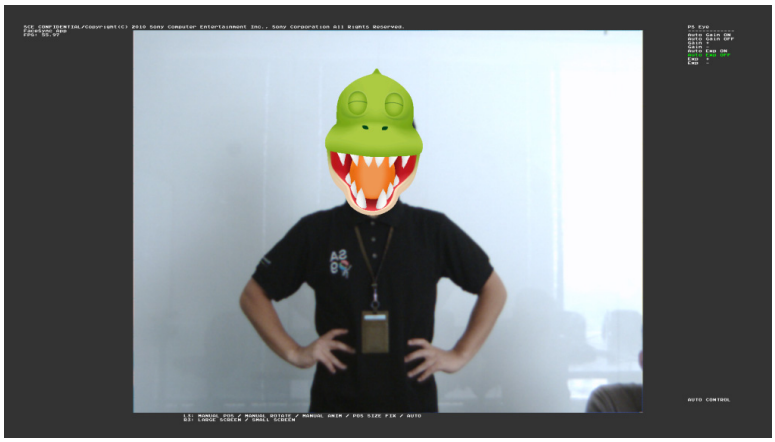
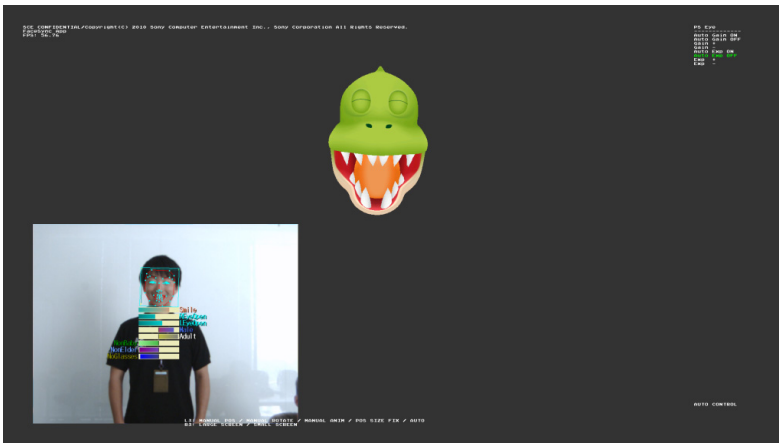


図 10　カメラ入力画像の縮小表示



モデル操作の切り替え

コントローラのL3 ボタンを押すと、以下の (a)→(b)→(c)→(d)→(e)→(a)→(b)…の順で循環して、モデルに関して行える操作が切り替わります。

(a)　スケール・位置・向き・アニメーション、すべて自動追従

(b)　以下の位置操作が可能、ほかのパラメータは固定

- 左アナログスティックでモデルを移動する

(c)　以下の回転操作が可能、ほかのパラメータは固定

- 左アナログスティックでロール
- 右アナログスティックでピッチ
- L2、R2 ボタンでヨー

(d)　以下のモデルアニメーション操作が可能、ほかのパラメータは固定

- 左アナログスティックでアニメーション 1
- 右アナログスティックでアニメーション 2
- L2 ボタンでアニメーション 3
- R2 ボタンでアニメーション 4

(e)　向き・アニメーション自動追従、位置とスケールは固定される

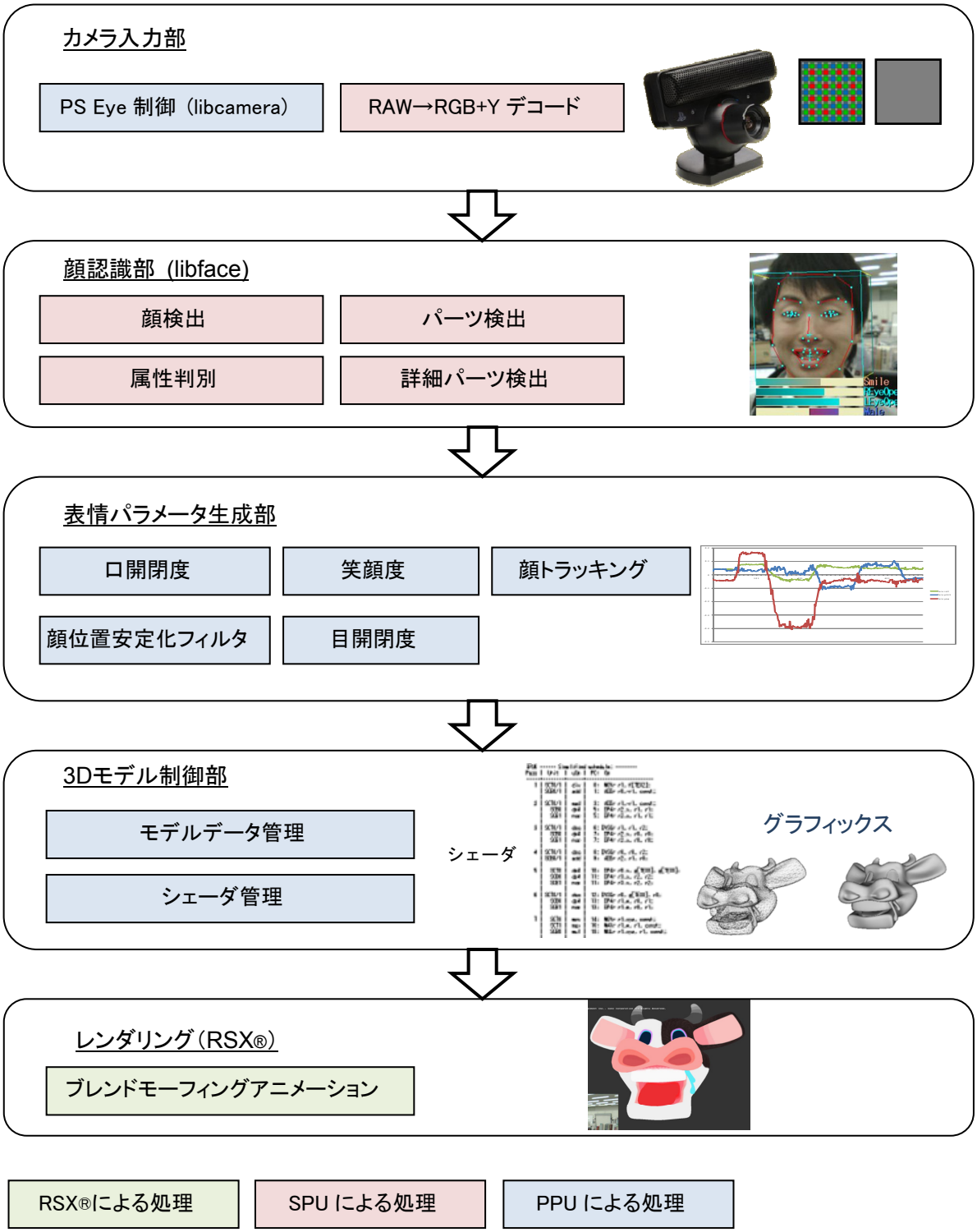
画面キャプチャ

コントローラの START ボタンを押すと、画面をキャプチャして PPM 画像ファイルとして保存できます。

5 解説

処理フロー

図 11　処理フロー図

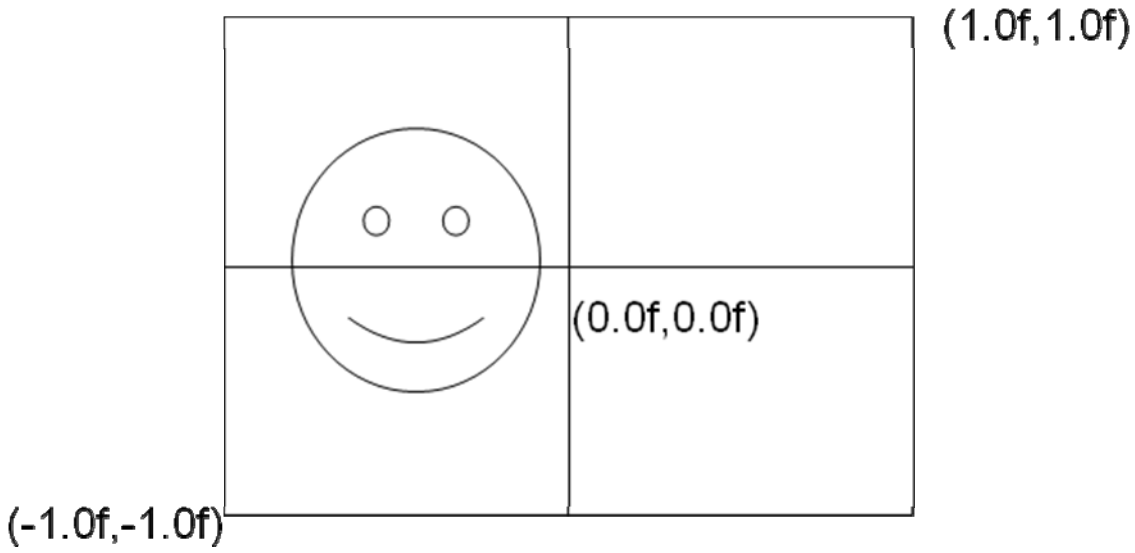


処理の要点

座標系

FaceSync サンプルでは、画面中心を(0.0f,0.0f)とし画面右上端を(1.0f,1.0f)、左端を(-1.0f,-1.0f)とする座標系を使用しています。

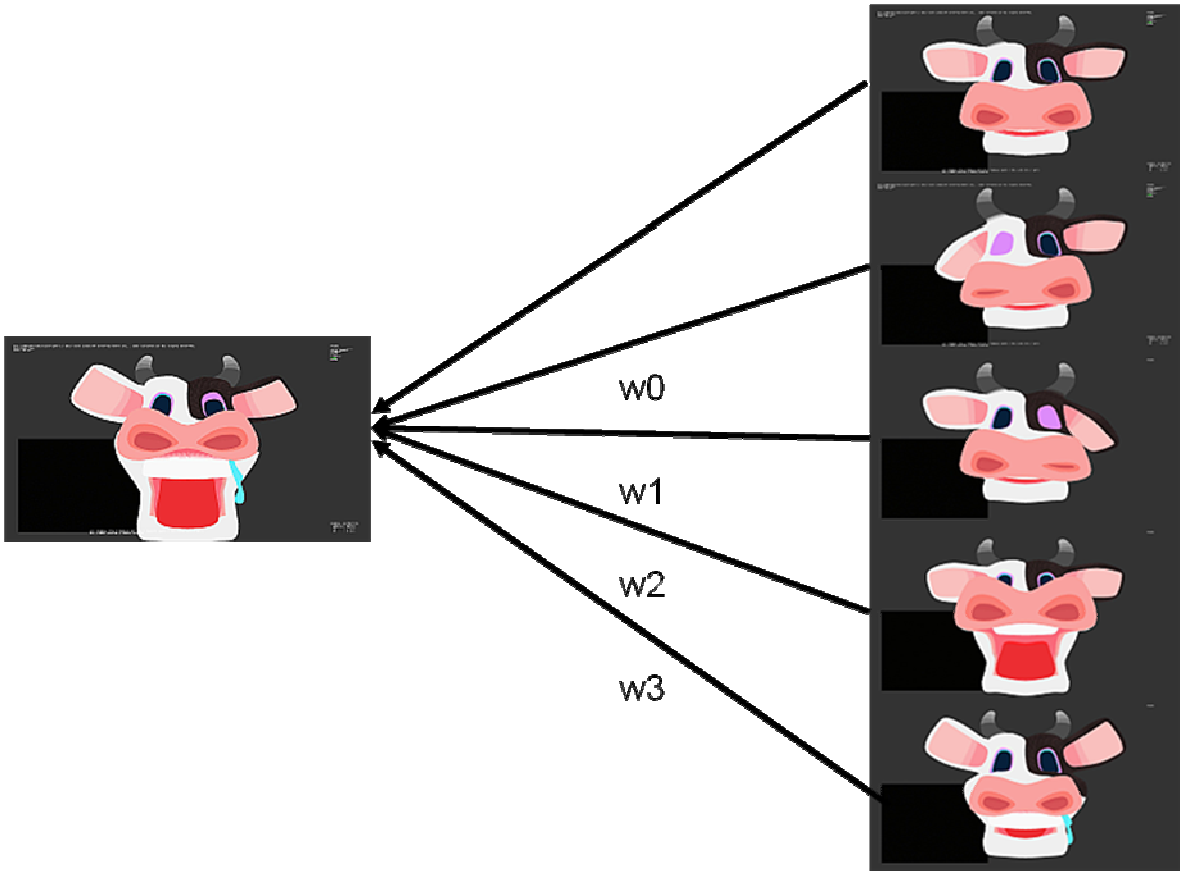
図 12　フレームバッファ上の座標系



頂点ブレンディングアニメーション

認識した表情に合わせた 3D モデルを生成するために頂点ブレンディングを行っています。つまり、基本のジオメトリに、右目閉じ・左目閉じ・口開き・笑顔のジオメトリを libface で取得したパラメータに応じた重み w0-w3 をかけて合成します。
この処理はバーテクスシェーダで行っています。

図 13　ジオメトリデータの合成



式 1　頂点ブレンディング

$$o' = o + (p0 - o) \times w0 + (p1 - o) \times w1 + (p2 - o) \times w2 + (p3 - o) \times w3$$

o ：	基本ジオメトリ（標準顔）の頂点位置
o’：	合成後のジオメトリデータ
p0, p1, p2, p3：	変形後ジオメトリの頂点位置 p0 は右目閉じの頂点位置、p1 は左目閉じの頂点位置、p2 は口開きの頂点位置、p3 は笑顔の頂点位置を示しています。
w0, w1, w2, w3：	libface からのブレンドパラメータ w0 は右目閉じの度合い、w1 は左目閉じの度合い、w2 は口開きの度合い、w3 は笑顔の度合いを示しています。

ジオメトリのデータフォーマット

ビッグエンディアンで格納されています。

基本ジオメトリデータファイル(.pnt)

```
uint32_t vertexCount;
{ // VertexData
  { // VertexPosition
    float x;
    float y;
    float z;
  };
  { // VertexTexcoord
    float u;
    float v;
  };
  { // VertexNormal
    float x;
    float y;
    float z;
  };
} vertexAos[vertexCount];
```

頂点ブレンディングに使用する位置のみのデータファイル(.p)

```
uint32_t vertexCount;
{ // VertexData
  { // VertexPosition
    float x;
    float y;
    float z;
  };
}
```

6 クラス構成

FaceSync サンプルは以下のクラス群で構成されます。

- FaceSyncApp
- FaceAnimationModel
- FaceFilter
- ImagePlaneModel
- PsEye
- Menu
- FaceTracker
- シェーダ

この章では、各クラスとその構成について説明します。

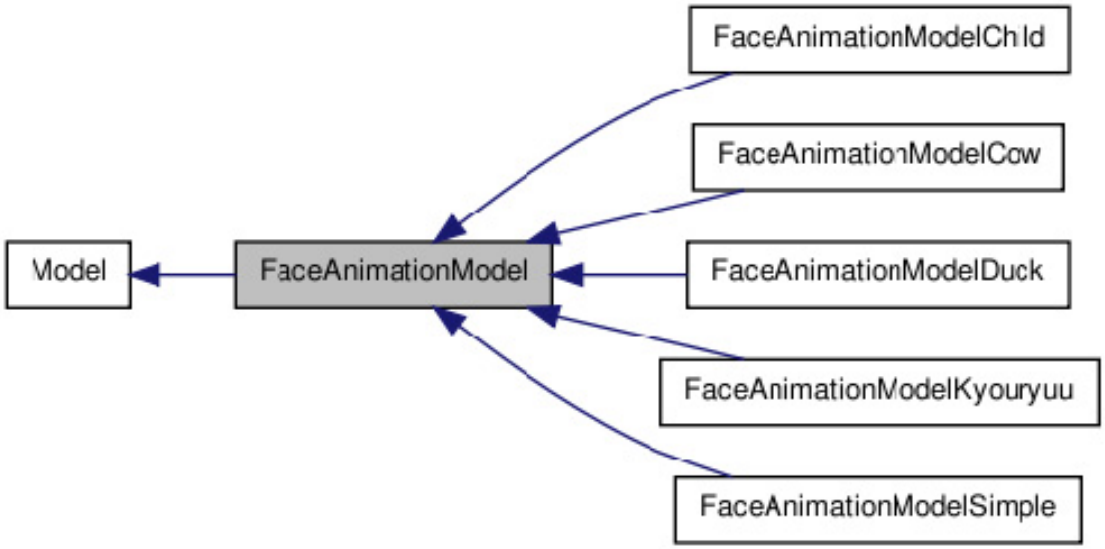
FaceSyncAppクラス

アプリケーションクラスです。

FaceAnimationModelクラス

CG キャラクタのアニメーション描画を扱います。

図 14　FaceAnimationModel クラス



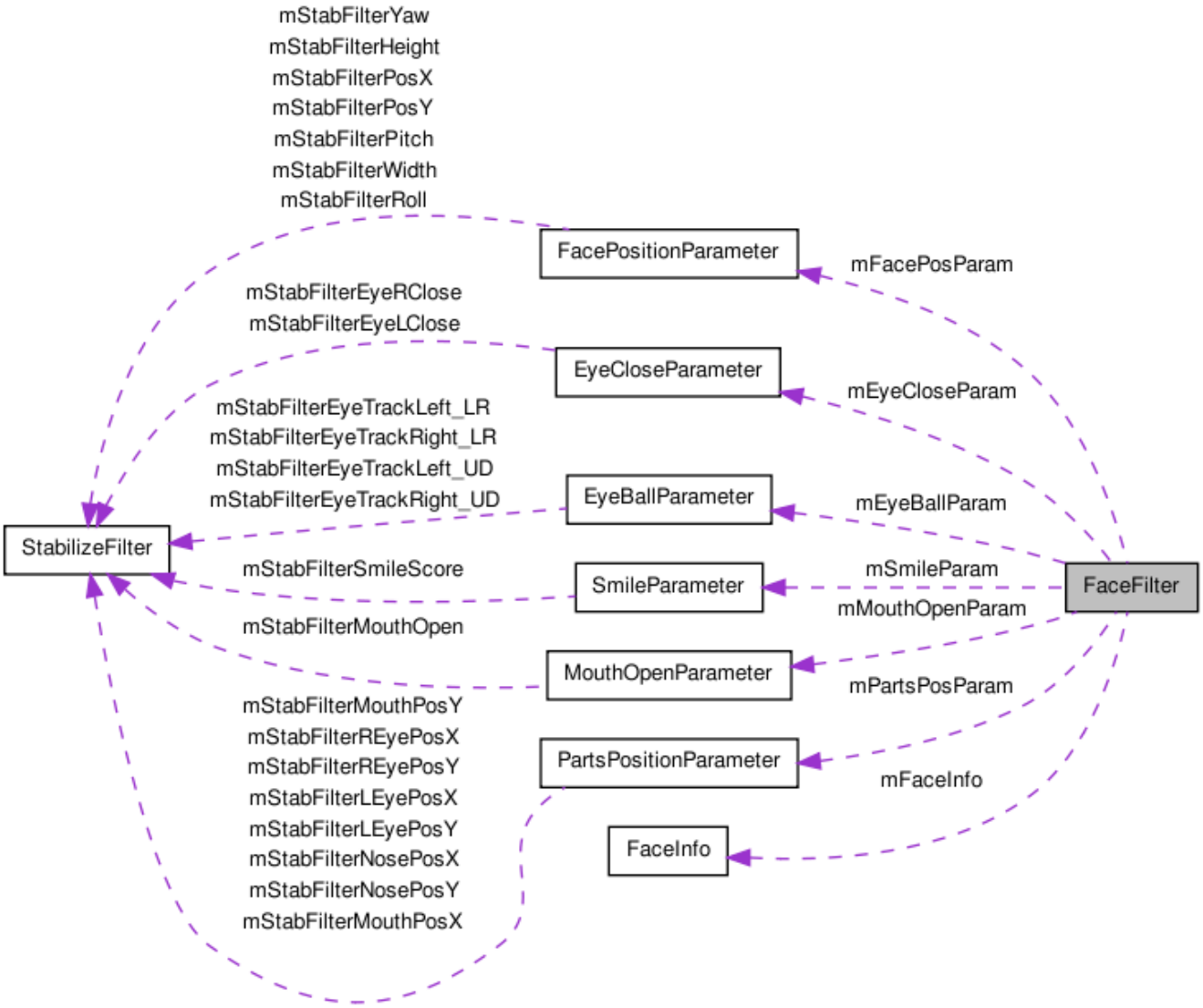
- Object3D
  描画オブジェクトの頂点データを保持する構造体。ImagePlaneModel および FaceAnimationModelXxx がこれを保持する。
- Texture
  GTF から読み込んだテクスチャを保持する構造体。ImagePlaneModel および FaceAnimationModelXxx がこれを保持する。

- PhonegParam
  FaceAnimationModel を継承したクラスが、フラグメントシェーダに渡すパラメータを保持する構造体
- Model
  描画モデルに共通する処理をまとめたクラス
- FaceAnimationModel
  顔（パーツ）モデルを抽象化したクラス。FaceSyncApp では、FaceAnimationModel のインタフェースを通して各種顔モデルを扱うことでモデルの切り替えを実現している。
- FaceAnimationModelSimple / FaceAnimationModelKyouryuu / FaceAnimationModelChild / FaceAnimationModelCow
  それぞれ、顔のモデルを表現するクラス。顔の動き、目の開閉、口の開閉、笑顔に追従する。
- FaceAnimationModelDuck
  アヒルのくちばしモデルを表現するクラス。口位置の動き、口の開閉に追従する。

FaceFilterクラス

libface で認識したパラメータを扱います。

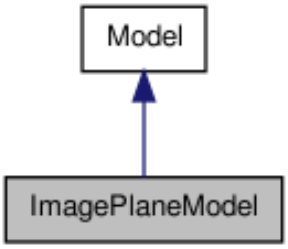
図 15　FaceFilter クラス



ImagePlaneModelクラス

カメラ入力画像を表示するポリゴンを扱います。

図 16　ImagePlaneModel クラス

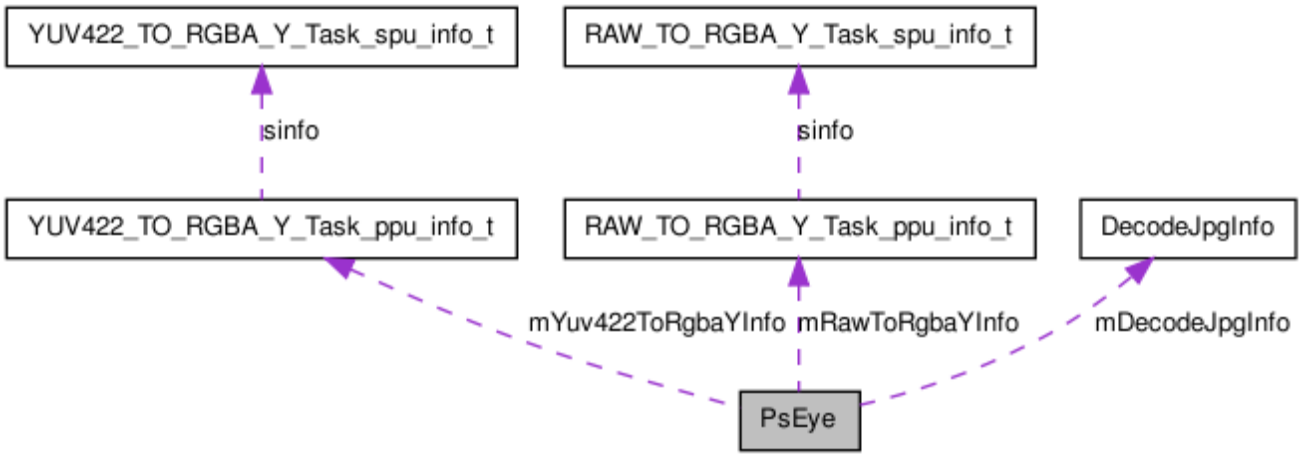


- ImagePlaneModel
  テクスチャを貼った矩形のモデルを表現するクラス。カメラプレビューを表示するのに使用される。

PsEyeクラス

PlayStation®Eye の処理を扱います。

図 17　PsEye クラス

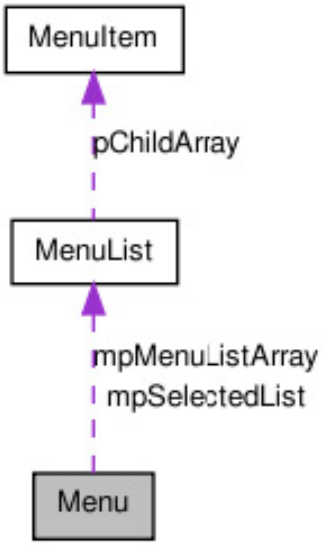


- PsEye
  PlayStation®Eye カメラクラス

Menuクラス

メニュー関連の処理を扱います。

図 18　Menu クラス

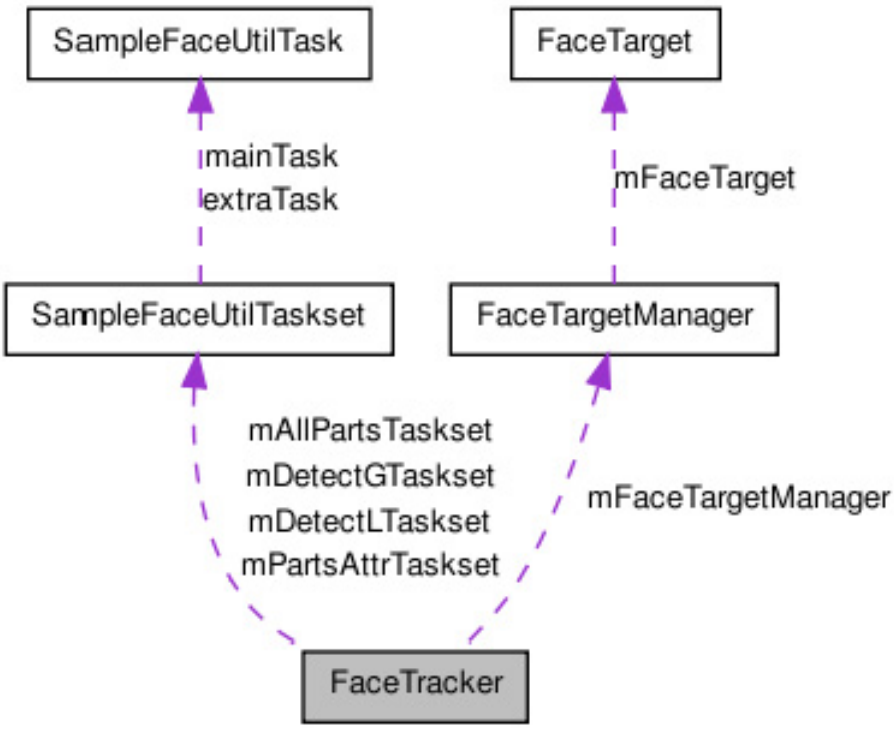


- MenuList / MenuItem / Menu
  画面上に表示するメニューを表現するクラスおよび構造体

FaceTrackerクラス

顔追従の処理を扱います。

図 19　FaceTracker クラス



- FaceInfo
  顔のアニメーションに用いる各種パラメータを保持する構造体。FaceFilter を通して取得できる。
- FaceTracker
  顔の認識およびトラッキング（追従）を実現するクラス
- FaceTargetManager
  複数の FaceTarget を制御するクラス
- FaceTarget
  顔検出結果を表現するクラス
- FaceFilter
  顔認識結果の安定化、および FaceTarget から FaceInfo への変換を実現するクラス
- FacePositionParameter / PartsPositionParameter / MouthOpenParameter / EyeCloseParameter / SmileParameter / EyeBallParameter
  それぞれ、顔位置、パーツ位置、口開閉、目開閉、笑顔、視線に関する結果の安定化、および FaceTarget 内の情報から FaceInfo 内のパラメータへの変換を移譲されたクラス

シェーダクラス

- ShaderController
  VertexShader, FragmentShader のインスタンスを管理するクラス
- VertexShader
  頂点シェーダに共通する処理をまとめたクラス
- VertexShaderAnimation5
  FaceAnimationModelXxx により利用される頂点シェーダクラス。基本モデル、および合成対象となるモデルとそれらの重みを与えることで、モデルの合成を実現する。
- VertexShaderTex
  ImageModel により利用される頂点シェーダクラス
- FragmentShader
  フラグメントシェーダに共通する処理をまとめたクラス
- FragmentShaderPhongAlpha
  FaceAnimationModelXxx により利用されるフラグメントシェーダクラス
- FragmentShaderTexUpsidedown
  ImageModel により利用される頂点シェーダクラス