PathTrackingAndInductionOfTheRobot

構築: Doxygen 1.8.6

2016 年 01 月 25 日 (月) 12 時 00 分 33 秒

Contents

Chapter 1

クラス索引

1.1 クラス一覧

クラス・構造体・共用体・インターフェースの一覧です。

Chapter 2

ファイル索引

2.1 ファイル一覧

ファイル一覧です。

Chapter 3

クラス詳解

3.1 AttitudeAngle 構造体

姿勢角

`#include <PathTrackingAndInductionOfTheRobot.hpp>`

公開変数類

- double **yaw**
  ヨー角
- double **roll**
  ロール角
- double **pitch**
  ピッチ角

3.1.1 詳解

姿勢角

PathTrackingAndInductionOfTheRobot.hpp の 65 行目に定義があります。

3.1.2 メンバ詳解

3.1.2.1 double AttitudeAngle::pitch

ピッチ角

PathTrackingAndInductionOfTheRobot.hpp の 68 行目に定義があります。

参照元 LeastSquareMethod::calcYawRollPitch(), EV3Control::set6DoFEV3().

3.1.2.2 double AttitudeAngle::roll

ロール角

PathTrackingAndInductionOfTheRobot.hpp の 67 行目に定義があります。

参照元 LeastSquareMethod::calcYawRollPitch(), EV3Control::set6DoFEV3().

3.1.2.3 double AttitudeAngle::yaw

ヨー角

PathTrackingAndInductionOfTheRobot.hpp の 66 行目に定義があります。

参照元 LeastSquareMethod::calcYawRollPitch(), EV3Control::set6DoFEV3().

この構造体詳解は次のファイルから抽出されました:

- **PathTrackingAndInductionOfTheRobot.hpp**

3.2 ControlParamd 構造体

走行制御用構造体．double 型

`#include <PathTrackingAndInductionOfTheRobot.hpp>`

公開変数類

- double **velocity**
  速度
- double **yaw**
  ヨー角

3.2.1 詳解

走行制御用構造体．double 型

PathTrackingAndInductionOfTheRobot.hpp の 75 行目に定義があります。

3.2.2 メンバ詳解

3.2.2.1 double ControlParamd::velocity

速度

PathTrackingAndInductionOfTheRobot.hpp の 76 行目に定義があります。

参照元 EV3Control::getAverageVelocityAndYaw(), EV3Control::outputControlInformation().

3.2.2.2 double ControlParamd::yaw

ヨー角

PathTrackingAndInductionOfTheRobot.hpp の 77 行目に定義があります。

参照元 EV3Control::getAverageVelocityAndYaw(), EV3Control::outputControlInformation().

この構造体詳解は次のファイルから抽出されました:

- **PathTrackingAndInductionOfTheRobot.hpp**

3.3 DoF6d 構造体

姿勢構造体の定義 (c78)．doule 型

```
#include <PathTrackingAndInductionOfTheRobot.hpp>
```

公開変数類

- double **x**
  x 座標
- double **y**
  y 座標
- double **z**
  z 座標
- double **yaw**
  ヨー角
- double **roll**
  ロール角
- double **pitch**
  ピッチ角

3.3.1 詳解

姿勢構造体の定義 (c78). doule 型

PathTrackingAndInductionOfTheRobot.hpp の 39 行目に定義があります。

3.3.2 メンバ詳解

3.3.2.1 double DoF6d::pitch

ピッチ角

PathTrackingAndInductionOfTheRobot.hpp の 45 行目に定義があります。

3.3.2.2 double DoF6d::roll

ロール角

PathTrackingAndInductionOfTheRobot.hpp の 44 行目に定義があります。

3.3.2.3 double DoF6d::x

x 座標

PathTrackingAndInductionOfTheRobot.hpp の 40 行目に定義があります。

3.3.2.4 double DoF6d::y

y 座標

PathTrackingAndInductionOfTheRobot.hpp の 41 行目に定義があります。

3.3.2.5 double DoF6d::yaw

ヨー角

PathTrackingAndInductionOfTheRobot.hpp の 43 行目に定義があります。

3.3.2.6 double DoF6d::z

z 座標

PathTrackingAndInductionOfTheRobot.hpp の 42 行目に定義があります。

この構造体詳解は次のファイルから抽出されました:

- **PathTrackingAndInductionOfTheRobot.hpp**

3.4 DoF6i 構造体

姿勢構造体の定義．int 型

`#include <PathTrackingAndInductionOfTheRobot.hpp>`

公開変数類

- int **x**
  x **座標**
- int **y**
  y **座標**
- int **z**
  z **座標**
- int **yaw**
  ヨー角
- int **roll**
  ロール角
- int **pitch**
  ピッチ角

3.4.1 詳解

姿勢構造体の定義．int 型

PathTrackingAndInductionOfTheRobot.hpp の 52 行目に定義があります。

3.4.2 メンバ詳解

3.4.2.1 int DoF6i::pitch

ピッチ角

PathTrackingAndInductionOfTheRobot.hpp の 58 行目に定義があります。

参照元 EV3Control::output6DoF(), EV3Control::output6DoFContinuous(), EV3Control::set6DoFEV3().

3.4.2.2 int DoF6i::roll

ロール角

PathTrackingAndInductionOfTheRobot.hpp の 57 行目に定義があります。

参照元 EV3Control::output6DoF(), EV3Control::output6DoFContinuous(), EV3Control::set6DoFEV3().

3.4.2.3 int DoF6i::x

x 座標

PathTrackingAndInductionOfTheRobot.hpp の 53 行目に定義があります。

参照元 EV3Control::getVelocityinSec(), EV3Control::output6DoF(), EV3Control::output6DoFContinuous(), EV3Control::outputEV3RouteContinuous(), EV3Control::set6DoFEV3().

3.4.2.4 int DoF6i::y

y 座標

PathTrackingAndInductionOfTheRobot.hpp の 54 行目に定義があります。

参照元 EV3Control::getVelocityinSec(), EV3Control::output6DoF(), EV3Control::output6DoFContinuous(), EV3Control::outputEV3RouteContinuous(), EV3Control::set6DoFEV3().

3.4.2.5 int DoF6i::yaw

ヨー角

PathTrackingAndInductionOfTheRobot.hpp の 56 行目に定義があります。

参照元 EV3Control::getAverageVelocityAndYaw(), EV3Control::output6DoF(), EV3Control::output6DoFContinuous(), EV3Control::set6DoFEV3().

3.4.2.6 int DoF6i::z

z 座標

PathTrackingAndInductionOfTheRobot.hpp の 55 行目に定義があります。

参照元 EV3Control::getVelocityinSec(), EV3Control::output6DoF(), EV3Control::output6DoFContinuous(), EV3Control::outputEV3RouteContinuous(), EV3Control::set6DoFEV3().

この構造体詳解は次のファイルから抽出されました:

- **PathTrackingAndInductionOfTheRobot.hpp**

3.5 Drawing クラス

経路描画用のクラス

`#include <Drawing.hpp>`

公開メンバ関数

- **Drawing** ()
  コンストラクタ
- ~**Drawing** ()
  デストラクタ
- void **gnuplotScriptEV3** (int save_count, char *original_dirpath, char *output_filename, Eigen::Vector3f coefficient_plane)
  EV3 の点群をプロットするためのスクリプト *(c78)*
- void **gnuplotScriptEV3Route** (char *original_dirpath, char *output_filename)
  EV3 の軌道をプロットするためのスクリプト

- void **gnuplotScriptTime2V** ()

  時間と速度のプロット
- void **gnuplotScriptTime2Yaw** ()

  時間とヨー角のプロット

3.5.1 詳解

経路描画用のクラス

Drawing.hpp の 19 行目に定義があります。

3.5.2 構築子と解体子

3.5.2.1 Drawing::Drawing ( )

コンストラクタ

メソッドDrawing::Drawing(). コンストラクタ

Drawing.cpp の 15 行目に定義があります。

3.5.2.2 Drawing::~Drawing ( )

デストラクタ

メソッドDrawing::Drawing(). デストラクタ

Drawing.cpp の 32 行目に定義があります。

3.5.3 関数詳解

3.5.3.1 void Drawing::gnuplotScriptEV3 (int *save_count*, char * *original_dirpath*, char * *output_filename*, Eigen::Vector3f *coefficient_plane*)

EV3 の点群をプロットするためのスクリプト (c78)

メソッドDrawing::gnuplotScriptEV3(). EV3 の点群をプロットするスクリプトを生成するメソッド (c78)

引数

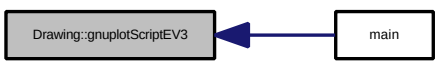
save_count	int 型. データの保存数のカウント
original_dirpath	char* 型. 基となるディレクトリ名
output_filename	char* 型. 出力するファイル名
coefficient_plane	Eigen::Vector3f 型. 平面の係数

Drawing.cpp の 49 行目に定義があります。

参照先 NOC.

参照元 main().

被呼び出し関係図:



3.5.3.2 void Drawing::gnuplotScriptEV3Route (char * *original_dirpath*, char * *output_filename*)

EV3 の軌道をプロットするためのスクリプト

メソッドDrawing::gnuplotScriptEV3Route(). EV3 の軌道をプロットするためのスクリプト

引数

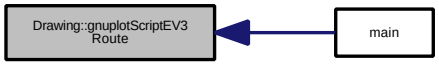
original_dirpath	char* 型. 基となるディレクトリ名
output_filename	char* 型. 出力するファイル名

Drawing.cpp の 72 行目に定義があります。

参照先 NOC.

参照元 main().

被呼び出し関係図:



3.5.3.3 void Drawing::gnuplotScriptTime2V ( )

時間と速度のプロット

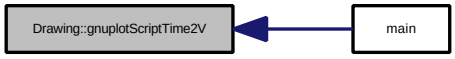
メソッドDrawing::gnuplotScriptTime2V(). 時間と速度の関係をプロットするためのスクリプト

Drawing.cpp の 93 行目に定義があります。

参照先 directoryName, NOC.

参照元 main().

被呼び出し関係図:



3.5.3.4 void Drawing::gnuplotScriptTime2Yaw ( )

時間とヨー角のプロット

メソッドDrawing::gnuplotScriptTime2Yaw(). 時間とヨー角の関係をプロットするためのスクリプト

Drawing.cpp の 110 行目に定義があります。

参照先 directoryName, NOC.

参照元 main().

被呼び出し関係図:

このクラス詳解は次のファイルから抽出されました:

- **Drawing.hpp**

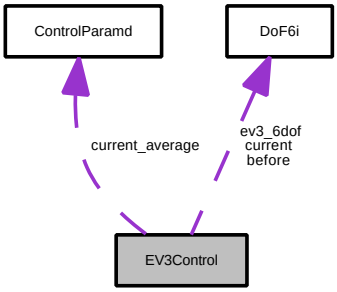
- **Drawing.cpp**

3.6 EV3Control クラス

EV3 を制御するためのクラス

`#include <EV3Control.hpp>`

EV3Control 連携図



公開メンバ関数

- **EV3Control** ()

  コンストラクタ
- ~**EV3Control** ()

  デストラクタ
- void **set6DoFEV3** (pcl::PointCloud< pcl::PointXYZRGB >::Ptr &inputPointCloud, Point3d centroid, **Attitude-Angle** attitude_angle)

  最小二乗法によって求めた平均座標と位置を*EV3* の制御のために構造体に格納する *(c80)*
- void **getVelocityinSec** (double time_ms)

  EV3 の速度を計算する *(c85)*
- void **getAverageVelocityAndYaw** ()

  平均の速度とヨー角を計算する
- void **output6DoF** (int save_count, char *original_dirpath, char *output_filename, pcl::PointCloud< pcl::Point-XYZRGB >::Ptr &outputPointCloud)

  現フレームの *6DoF* 情報をファイルに出力する
- void **output6DoFContinuous** (char *original_dirpath, char *output_filename, pcl::PointCloud< pcl::PointX-YZRGB >::Ptr &outputPointCloud)

  キーを入力したときの *6DoF* 情報を連続して *csv* 形式で保存する
- void **outputEV3RouteContinuous** (char *original_dirpath, char *output_filename)

  EV3 の走行軌道を保存する
- void **outputControlInformation** (double sumtime_ms, char *original_dirpath, char *output_filename)

  EV3 の制御情報を出力する
- void **outputControlInformation** ()

  EV3 に必要な速度とヨー角をファイルに出力

公開変数類

- **DoF6i ev3_6dof**

  EV3 の *6* 自由度 *(c80)*
- **DoF6i before**

前フレームの *6DoF* 情報

- **DoF6i current**

  現フレームの *6DoF* 情報

- double **velocity**

  速度 *v(c85)*

- bool **flag_velocity**

  最初の *1* フレームのためのフラグ

- **ControlParamd current_average**

  現在の平均の速度とヨー角

- int **count_average**

  速度とヨー角の過去 *5* フレーム分の平均を取るために最初の *5* フレームをカウントするための変数

- bool **flag_average**

  始めの *5* フレーム分用

- bool **save_flag**

  6DoF 情報を出力するかチェックするためのフラグ

3.6.1 詳解

EV3 を制御するためのクラス

EV3Control.hpp の 18 行目に定義があります。

3.6.2 構築子と解体子

3.6.2.1 EV3Control::EV3Control ( )

コンストラクタ

メソッドEV3Control::EV3Control(). コンストラクタ

EV3Control.cpp の 15 行目に定義があります。

参照先 before, count_average, current, current_average, flag_average, flag_velocity, save_flag.

3.6.2.2 EV3Control::～EV3Control ( )

デストラクタ

メソッドEV3Control::EV3Control(). コンストラクタ

EV3Control.cpp の 38 行目に定義があります。

3.6.3 関数詳解

3.6.3.1 void EV3Control::getAverageVelocityAndYaw ( )

平均の速度とヨー角を計算する

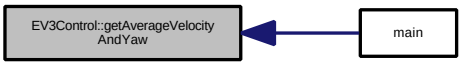
メソッドEV3Control::getAverageVelocityAndYaw(). 速度とヨー角を取得するメソッド

EV3Control.cpp の 89 行目に定義があります。

参照先 count_average, current_average, ev3_6dof, flag_average, velocity, ControlParamd::velocity, DoF6i::yaw, ControlParamd::yaw.

参照元 main().

被呼び出し関係図:



3.6.3.2 void EV3Control::getVelocityinSec (double *time_ms*)

EV3 の速度を計算する (c85)

メソッドEV3Control::getVelocity(). EV3 の速度を計算するメソッド

引数

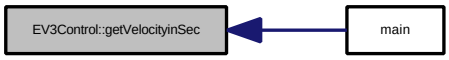
time_ms	double 型. 1 フレームの処理時間

EV3Control.cpp の 67 行目に定義があります。

参照先 before, current, ev3_6dof, flag_velocity, velocity, DoF6i::x, DoF6i::y, DoF6i::z.

参照元 main().

被呼び出し関係図:



3.6.3.3 void EV3Control::output6DoF (int *save_count*, char * *original_dirpath*, char * *output_filename*, pcl::PointCloud< pcl::PointXYZRGB >::Ptr & *outputPointCloud*)

現フレームの 6DoF 情報をファイルに出力する

メソッドEV3Control::output6Dof(). 現フレームの 6DoF 情報をファイルに出力する

引数

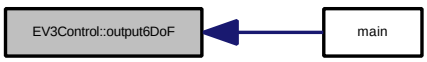
save_count	int 型. 保存したカウント
original_dirpath	char* 型. 基となるディレクトリ名
output_filename	char* 型. 出力するファイル名

EV3Control.cpp の 150 行目に定義があります。

参照先 ev3_6dof, NOC, DoF6i::pitch, DoF6i::roll, DoF6i::x, DoF6i::y, DoF6i::yaw, DoF6i::z.

参照元 main().

被呼び出し関係図:



3.6.3.4 void EV3Control::output6DoFContinuous (char * *original_dirpath*, char * *output_filename*, pcl::PointCloud< pcl::PointXYZRGB >::Ptr & *outputPointCloud*)

キーを入力したときの 6DoF 情報を連続して csv 形式で保存する

メソッドEV3Control::output6DoFContinuous(). キーを入力したときの 6DoF 情報を連続して csv 形式で保存する

引数

original_dirpath	char* 型．基となるディレクトリ名
output_filename	char* 型．出力するファイル名
&inputPoint-Cloud	pcl::PointCloud<pcl::PointXYZRGB>::Ptr

EV3Control.cpp の 169 行目に定義があります。

参照先 ev3_6dof, NOC, DoF6i::pitch, DoF6i::roll, save_flag, DoF6i::x, DoF6i::y, DoF6i::yaw, DoF6i::z.

参照元 main().

被呼び出し関係図:

3.6.3.5 void EV3Control::outputControlInformation (double *sumtime_ms*, char * *original_dirpath*, char * *output_filename*)

EV3 の制御情報を出力する

メソッドEV3Control::outputControlInformation(). EV3 の制御情報を出力する

引数

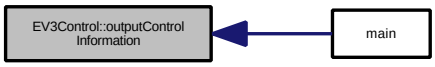
sumtime_ms	double 型．合計時間
original_dirpath	char* 型．基となるディレクトリ名
output_filename	char* 型．出力するファイル名

EV3Control.cpp の 208 行目に定義があります。

参照先 current_average, NOC, ControlParamd::velocity, ControlParamd::yaw.

参照元 main().

被呼び出し関係図:



3.6.3.6 void EV3Control::outputControlInformation ( )

EV3 に必要な速度とヨー角をファイルに出力

メソッドEV3Control::outputControlInformation(). EV3 に必要な速度とヨー角をファイルに出力

EV3Control.cpp の 223 行目に定義があります。

参照先 current_average, ControlParamd::velocity, ControlParamd::yaw.

3.6.3.7 void EV3Control::outputEV3RouteContinuous (char * *original_dirpath*, char * *output_filename*)

EV3 の走行軌道を保存する

メソッドEV3Control::outputEV3RouteContinuous(). EV3 の走行軌道を保存する

引数

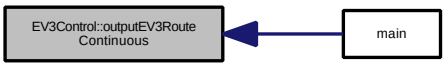
original_dirpath	char* 型．基となるディレクトリ名
output_filename	char* 型．出力するファイル名

EV3Control.cpp の 189 行目に定義があります。

参照先 ev3_6dof, NOC, DoF6i::x, DoF6i::y, DoF6i::z.

参照元 main().

被呼び出し関係図:



3.6.3.8 void EV3Control::set6DoFEV3 (pcl::PointCloud< pcl::PointXYZRGB >::Ptr & *inputPointCloud*, Point3d *centroid*, AttitudeAngle *attitude_angle*)

最小二乗法によって求めた平均座標と位置をEV3 の制御のために構造体に格納する (c80)

メソッドEV3Control::set6DoFEV3(). 最小二乗法によって求めた平均座標と位置をEV3 の制御のために構造体に格納する

引数

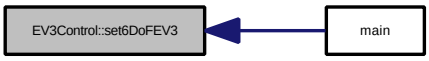
&inputPoint-Cloud	pcl::PointCloud<pcl::PointXYZRGB>::Ptr 型．入力するポイントクラウド
centroid	Point3d 型．重心座標
attitude_angle	AttitudeAngle 型．姿勢角

EV3Control.cpp の 49 行目に定義があります。

参照先 ev3_6dof, DoF6i::pitch, AttitudeAngle::pitch, DoF6i::roll, AttitudeAngle::roll, DoF6i::x, DoF6i::y, DoF6i::yaw, AttitudeAngle::yaw, DoF6i::z.

参照元 main().

被呼び出し関係図:



3.6.4 メンバ詳解

3.6.4.1 DoF6i EV3Control::before

前フレームの 6DoF 情報

EV3Control.hpp の 35 行目に定義があります。

参照元 EV3Control(), getVelocityinSec().

3.6.4.2 int EV3Control::count_average

速度とヨー角の過去 5 フレーム分の平均を取るために最初の 5 フレームをカウントするための変数

EV3Control.hpp の 42 行目に定義があります。

参照元 EV3Control(), getAverageVelocityAndYaw().

3.6.4.3 DoF6i EV3Control::current

現フレームの 6DoF 情報

EV3Control.hpp の 36 行目に定義があります。

参照元 EV3Control(), getVelocityinSec().

3.6.4.4 ControlParamd EV3Control::current_average

現在の平均の速度とヨー角

EV3Control.hpp の 41 行目に定義があります。

参照元 EV3Control(), getAverageVelocityAndYaw(), outputControlInformation().

3.6.4.5 DoF6i EV3Control::ev3_6dof

EV3 の 6 自由度 (c80)

EV3Control.hpp の 32 行目に定義があります。

参照元 getAverageVelocityAndYaw(), getVelocityinSec(), output6DoF(), output6DoFContinuous(), outputEV3Route-Continuous(), set6DoFEV3().

3.6.4.6 bool EV3Control::flag_average

始めの 5 フレーム分用

EV3Control.hpp の 43 行目に定義があります。

参照元 EV3Control(), getAverageVelocityAndYaw().

3.6.4.7 bool EV3Control::flag_velocity

最初の 1 フレームのためのフラグ

EV3Control.hpp の 38 行目に定義があります。

参照元 EV3Control(), getVelocityinSec().

3.6.4.8 bool EV3Control::save_flag

6DoF 情報を出力するかチェックするためのフラグ

EV3Control.hpp の 48 行目に定義があります。

参照元 EV3Control(), main(), output6DoFContinuous().

3.6.4.9 double EV3Control::velocity

速度 v(c85)

EV3Control.hpp の 37 行目に定義があります。

参照元 getAverageVelocityAndYaw(), getVelocityinSec().

このクラス詳解は次のファイルから抽出されました:

- **EV3Control.hpp**
- **EV3Control.cpp**

3.7 ImageProcessing クラス

画像処理用のクラス

`#include <ImageProcessing.hpp>`

公開メンバ関数

- **ImageProcessing** ()

  コンストラクタ
- ~**ImageProcessing** ()

  デストラクタ
- void **showImage** (string windowName, Mat &input_image)

  ウインドウの名前を引数に追加 *(c31)*. *Mat* の表示 *(c17)*
- void **showImageTogether** (Mat &image1, Mat &image2)

  2 つの画像を一緒に表示 *(c36)*
- void **showImageTogether** (Mat &image1, Mat &image2, Mat &image3)

  3 つの画像を一緒に表示 *(c36)*
- void **loadInternalCameraParameter** (const string cameraParamFile)

  カメラキャリブレーションによって得られたパラメータを適用する *(c54)*
- Mat **getUndistortionImage** (Mat &inputOriginalImage)

  キャリブレーションデータを用いて *Kinect* から取得した画像を補正する *(c71)*
- Mat **getBackgroundSubstractionBinImage** (Mat &current_image, Mat &backgound_gray_image)

  背景差分によって得られた二値画像 *(c75)*
- void **openCVSettingTrackbar** (const string trackbar_name)

  画像処理関連のトラックバーを表示するメソッド
- Mat **getUnitMask** (Mat &input_binimage)

  EV3 のユニット部のみのマスク画像を取得するメソッド
- void **outputImageSelectDirectory** (int save_count, char ∗original_dirpath, char ∗save_filename, Mat &output_image)

  メソッド *ImageProcessing::outputImageSelectDirectory()*. 出力したディレクトリにファイルを出力するメソッド

公開変数類

- int **th**

  二値化するときの閾値 *(c82)*
- int **neighborhood**

  平滑化を行うときの近傍
- int **closing_times**

  クロージングを行う回数

3.7.1 詳解

画像処理用のクラス

ImageProcessing.hpp の 19 行目に定義があります。

3.7.2 構築子と解体子

3.7.2.1 ImageProcessing::ImageProcessing ( )

コンストラクタ

メソッドImageProcessing::ImageProcessing(). コンストラクタ

ImageProcessing.cpp の 15 行目に定義があります。

参照先 closing_times, neighborhood, th.

3.7.2.2 ImageProcessing::~ImageProcessing ( )

デストラクタ

メソッドImageProcessing::~ImageProcessing(). デストラクタ

ImageProcessing.cpp の 26 行目に定義があります。

3.7.3 関数詳解

3.7.3.1 Mat ImageProcessing::getBackgroundSubstractionBinImage (Mat & *current_image*, Mat & *background_gray_image*)

背景差分によって得られた二値画像 (c75)

メソッドImageProcessing::getBackgroundSubstractionBinImage(). 背景差分によって得られた二値画像 (c75)

引数

&current_image	cv::Mat 型. 現フレームの入力画像
&background_image	cv::Mat 型. 背景画像の入力

戻り値

closing_image cv::Mat 型. 出力画像

< 現在のグレースケール画像 (c75)

< 背景差分画像 (c74)

< 背景差分画像の二値画像 (c75)

< 背景差分画像の二値画像を平滑化したもの (c75)

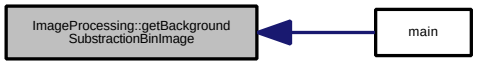
< クロージング処理御用変数

ImageProcessing.cpp の 130 行目に定義があります。

参照先 closing_times, neighborhood, th.

参照元 main().

被呼び出し関係図:



3.7.3.2 Mat ImageProcessing::getUndistortionImage (Mat & *inputOriginalImage*)

キャリブレーションデータを用いてKinect から取得した画像を補正する (c71)

メソッドImageProcessing::getUndistortionImage(). キャリブレーションデータを用いてKinect から取得した画像を補正する

引数

&inputOriginal-Image	cv::Mat 型. キャリブレーションを行いたい入力画像

戻り値

undistortionImage cv::Mat 型. キャリブレーション後の画像

ImageProcessing.cpp の 115 行目に定義があります。

参照元 main().

被呼び出し関係図:

3.7.3.3 Mat ImageProcessing::getUnitMask (Mat & *input_binimage*)

EV3 のユニット部のみのマスク画像を取得するメソッド

メソッドImageProcessing::getUnitMask(). EV3 のユニット部のみのマスク画像を取得するメソッド

引数

&input_binimage	cv::Mat 型. 入力画像 (二値画像)

戻り値

input_binimage cv::Mat 型. 出力画像 (マスク処理された二値画像)

ImageProcessing.cpp の 171 行目に定義があります。

3.7.3.4 void ImageProcessing::loadInternalCameraParameter (const string *cameraParamFile*)

カメラキャリブレーションによって得られたパラメータを適用する (c54)

ImageProcessing::loadInternalCameraParam(). カメラキャリブレーションによって得られたカメラパラメータを適用するメソッド (c54)

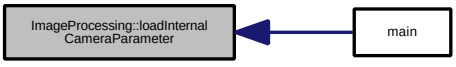
引数

cameraParam-File	const string 型. 入力したいカメラパラメータのファイル名

ImageProcessing.cpp の 98 行目に定義があります。

参照元 main().

被呼び出し関係図:



3.7.3.5 void ImageProcessing::openCVSettingTrackbar (const string *trackbar_name*)

画像処理関連のトラックバーを表示するメソッド

メソッドImageProcessing::openCVSettingTrackbar(). OpenCV のトラックバーを表示するメソッド

引数

trackbar_name	const string 型. OpenCV でマスク画像を設定するためのトラックバー名

ImageProcessing.cpp の 156 行目に定義があります。

参照先 closing_times, neighborhood, th.

参照元 main().

被呼び出し関係図:

3.7.3.6 void ImageProcessing::outputImageSelectDirectory (int *save_count*, char * *original_dirpath*, char * *save_filename*, Mat & *output_image*)

メソッドImageProcessing::outputImageSelectDirectory(). 出力したディレクトリにファイルを出力するメソッド

引数

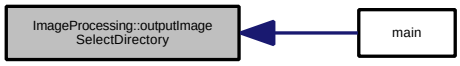
save_count	int 型. 保存数のカウント
original_dirpath	char* 型. 基となるディレクトリ名
save_filename	char*. 出力するファイル名
&output_image	cv::Mat 型. 出力する画像

ImageProcessing.cpp の 266 行目に定義があります。

参照先 NOC.

参照元 main().

被呼び出し関係図:



3.7.3.7 void ImageProcessing::showImage (string *windowName*, Mat & *input_image*)

ウインドウの名前を引数に追加 (c31). Mat の表示 (c17)

メソッドImageProcessing::showImage().cv::Mat を表示

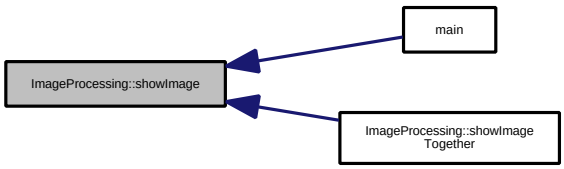
引数

windowName	std::string 型. 表示するウインドウ名
&input_image	cv::Mat 型. 入力画像

ImageProcessing.cpp の 36 行目に定義があります。

参照元 main(), showImageTogether().

被呼び出し関係図:



3.7.3.8 void ImageProcessing::showImageTogether (Mat & *image1*, Mat & *image2*)

2 つの画像を一緒に表示 (c36)

メソッドImageProcessing::showTogetherImage().2 つの cv::Mat を 1 つのウインドウに表示

引数

&image1	cv::Mat 型．入力画像 1
&image2	cv::Mat 型．入力画像 2

ImageProcessing.cpp の 48 行目に定義があります。

参照先 showImage().

呼び出し関係図:

3.7.3.9 void ImageProcessing::showImageTogether (Mat & *image1*, Mat & *image2*, Mat & *image3*)

3 つの画像を一緒に表示 (c36)

メソッドImageProcessing::showTogetherImage().3 つの cv::Mat を 1 つのウインドウに表示

引数

&image1	cv::Mat 型．入力画像 1
&image2	cv::Mat 型．入力画像 2
&image3	cv::Mat 型．入力画像 3

ImageProcessing.cpp の 73 行目に定義があります。

参照先 showImage().

呼び出し関係図:

3.7.4 メンバ詳解

3.7.4.1 int ImageProcessing::closing_times

クロージングを行う回数

ImageProcessing.hpp の 42 行目に定義があります。

参照元 getBackgroundSubstractionBinImage(), ImageProcessing(), openCVSettingTrackbar().

3.7.4.2 int ImageProcessing::neighborhood

平滑化を行うときの近傍

ImageProcessing.hpp の 41 行目に定義があります。

参照元 getBackgroundSubstractionBinImage(), ImageProcessing(), openCVSettingTrackbar().

3.7.4.3 int ImageProcessing::th

二値化するときの閾値 (c82)

ImageProcessing.hpp の 40 行目に定義があります。

参照元 getBackgroundSubstractionBinImage(), ImageProcessing(), openCVSettingTrackbar().

このクラス詳解は次のファイルから抽出されました:

- **ImageProcessing.hpp**
- **ImageProcessing.cpp**

3.8 Kinect クラス

Kinect 操作用のクラス

`#include <Kinect.hpp>`

公開メンバ関数

- **Kinect** ()
  コンストラクタ
- ~**Kinect** ()
  デストラクタ
- void **initialize** ()
  Kinect の初期化
- void **createInstance** ()
  インスタンスの生成
- Mat **drawRGBImage** (Mat &**image**)
  RGB カメラの処理
- pcl::PointCloud
  < pcl::PointXYZRGB >::Ptr **getPointCloud** (Mat &Mat_image)
  Depth カメラの処理 *(c57)*
- int **getDistance** (Mat &**image**)
  距離を取得 *(c49)*

公開変数類

- HANDLE **streamEvent**
  RGB,Depth カメラのフレーム更新イベントを待つためのイベントハンドル
- int **key**

ウィンドウ表示のウェイトタイム格納変数

- int **actualExtractedNum**

  実際に距離が抽出された数 *(0* 以外だった数*)(c31)*

3.8.1 詳解

Kinect 操作用のクラス

Kinect.hpp の 30 行目に定義があります。

3.8.2 構築子と解体子

3.8.2.1 Kinect::Kinect ( )

コンストラクタ

メソッドKinect::Kinect(). コンストラクタ

Kinect.cpp の 15 行目に定義があります。

3.8.2.2 Kinect::∼Kinect ( )

デストラクタ

メソッドKinect::∼Kinect(). デストラクタ

Kinect.cpp の 23 行目に定義があります。

3.8.3 関数詳解

3.8.3.1 void Kinect::createInstance ( )

インスタンスの生成

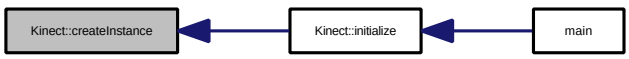
メソッドKinect::createInstance(). インスタンスの生成

Kinect.cpp の 35 行目に定義があります。

参照先 ERROR_CHECK.

参照元 initialize().

被呼び出し関係図:



3.8.3.2 Mat Kinect::drawRGBImage (Mat & *image*)

RGB カメラの処理

メソッドKinect::drawRGBImage().RGB カメラの処理

引数

image	cv::Mat&. 入力画像

戻り値

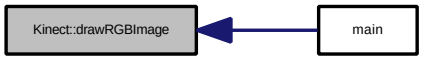
image cv::Mat. 出力画像

Kinect.cpp の 81 行目に定義があります。

参照先 ERROR_CHECK.

参照元 main().

被呼び出し関係図:



3.8.3.3 int Kinect::getDistance (Mat & *image*)

距離を取得 (c49)

3.8.3.4 pcl::PointCloud< pcl::PointXYZRGB >::Ptr Kinect::getPointCloud (Mat & *Mat_image*)

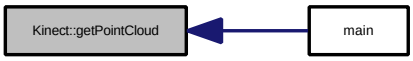
Depth カメラの処理 (c57)

Kinect.cpp の 109 行目に定義があります。

参照先 CAMERA_RESOLUTION, ERROR_CHECK, image.

参照元 main().

被呼び出し関係図:



3.8.3.5 void Kinect::initialize ( )

Kinect の初期化

メソッドKinect::initialize().Kinect の初期化

Kinect.cpp の 58 行目に定義があります。

参照先 CAMERA_RESOLUTION, createInstance(), ERROR_CHECK, streamEvent.

参照元 main().

呼び出し関係図:

被呼び出し関係図:

3.8.4 メンバ詳解

3.8.4.1 int Kinect::actualExtractedNum

実際に距離が抽出された数 (0 以外だった数)(c31)

Kinect.hpp の 54 行目に定義があります。

3.8.4.2 int Kinect::key

ウィンドウ表示のウェイトタイム格納変数

Kinect.hpp の 53 行目に定義があります。

参照元 main().

3.8.4.3 HANDLE Kinect::streamEvent

RGB,Depth カメラのフレーム更新イベントを待つためのイベントハンドル

Kinect.hpp の 52 行目に定義があります。

参照元 initialize(), main().

このクラス詳解は次のファイルから抽出されました:

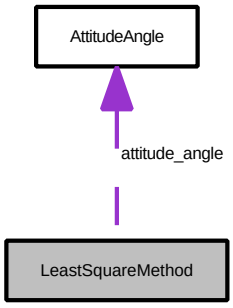
- **Kinect.hpp**
- **Kinect.cpp**

3.9 LeastSquareMethod クラス

最小二乗法を行うクラス

`#include <LeastSquareMethod.hpp>`

LeastSquareMethod 連携図



公開メンバ関数

- **LeastSquareMethod** ()
  コンストラクタ
- ～**LeastSquareMethod** ()
  デストラクタ
- Eigen::Vector3f **getCoefficient** (pcl::PointCloud< pcl::PointXYZRGB >::Ptr &inputPointCloud)
  最小二乗法によって平面 *ax+by+c=0* の係数 *[a b c]'* を求めるメソッド
- **AttitudeAngle3d calcYawRollPitch** (Eigen::Vector3f **coefficient_plane**)
  最小二乗法によって求めた *[a b c]'* を用いて平面の姿勢を計算する

公開変数類

- Eigen::Vector3f **coefficient_plane**
  平面の係数
- **AttitudeAngle3d attitude_angle**
  姿勢角 *(c78)*

3.9.1 詳解

最小二乗法を行うクラス

LeastSquareMethod.hpp の 19 行目に定義があります。

3.9.2 構築子と解体子

3.9.2.1 LeastSquareMethod::LeastSquareMethod ( )

コンストラクタ

メソッドLeastSquareMethod::LeastSquareMethod(). コンストラクタ

LeastSquareMethod.cpp の 15 行目に定義があります。

3.9.2.2 LeastSquareMethod::～LeastSquareMethod ( )

デストラクタ

メソッドLeastSquareMethod::～LeastSquareMethod(). デストラクタ

LeastSquareMethod.cpp の 23 行目に定義があります。

3.9.3 関数詳解

3.9.3.1 AttitudeAngle3d LeastSquareMethod::calcYawRollPitch (Eigen::Vector3f *coefficient_plane*)

最小二乗法によって求めた [a b c]' を用いて平面の姿勢を計算する

メソッドLeastSquareMethod::calcYawRollPitch(). 最小二乗法によって求めた [a b c]' を用いて平面の姿勢を計算する

引数

coefficient_plane	Eigen::Vector3f. 平面の係数

戻り値

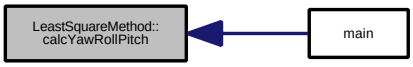
attitude_angle_deg AttitudeAngle3d 型. 姿勢角 [deg]

LeastSquareMethod.cpp の 77 行目に定義があります。

参照先 AttitudeAngle::pitch, AttitudeAngle::roll, AttitudeAngle::yaw.

参照元 main().

被呼び出し関係図:



3.9.3.2 Eigen::Vector3f LeastSquareMethod::getCoefficient (pcl::PointCloud< pcl::PointXYZRGB >::Ptr & *inputPointCloud*)

最小二乗法によって平面 ax+by+c=0 の係数 [a b c]' を求めるメソッド

メソッドLeastSquareMethod::getCoefficient(). 最小二乗法によって平面 ax+by+c=0 の係数 [a b c]' を求めるメソッド

引数

&inputPoint-Cloud	pcl::PointCloud<pcl::PointXYARGB>::Ptr 型. 入力するポイントクラウド

戻り値

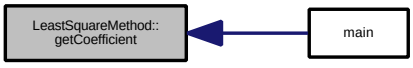
coefficient_plane Eigen::Vector3f 型. 平面の係数

LeastSquareMethod.cpp の 33 行目に定義があります。

参照先 coefficient_plane.

参照元 main().

被呼び出し関係図:



3.9.4 メンバ詳解

3.9.4.1 AttitudeAngle3d LeastSquareMethod::attitude_angle

姿勢角 (c78)

LeastSquareMethod.hpp の 31 行目に定義があります。

参照元 main().

3.9.4.2 Eigen::Vector3f LeastSquareMethod::coefficient_plane

平面の係数

LeastSquareMethod.hpp の 29 行目に定義があります。

参照元 getCoefficient(), main().

このクラス詳解は次のファイルから抽出されました:

- **LeastSquareMethod.hpp**
- **LeastSquareMethod.cpp**

3.10 outputData 構造体

ファイルに出力するデータ群 (c41)

`#include <PathTrackingAndInductionOfTheRobot.hpp>`

公開変数類

- double **totalTime**
  合計時間
- float **x**
  x **座標**
- float **y**
  y **座標**
- float **z**
  z **座標**

3.10.1 詳解

ファイルに出力するデータ群 (c41)

PathTrackingAndInductionOfTheRobot.hpp の 28 行目に定義があります。

3.10.2 メンバ詳解

3.10.2.1 double outputData::totalTime

合計時間

PathTrackingAndInductionOfTheRobot.hpp の 29 行目に定義があります。

3.10.2.2 float outputData::x

x 座標

PathTrackingAndInductionOfTheRobot.hpp の 30 行目に定義があります。

3.10.2.3 float outputData::y

y 座標

PathTrackingAndInductionOfTheRobot.hpp の 31 行目に定義があります。

3.10.2.4 float outputData::z

z 座標

PathTrackingAndInductionOfTheRobot.hpp の 32 行目に定義があります。

この構造体詳解は次のファイルから抽出されました:

- **PathTrackingAndInductionOfTheRobot.hpp**

3.11 Point3 構造体

抽出された座標を保存する構造体 (c37)

`#include <PathTrackingAndInductionOfTheRobot.hpp>`

公開変数類

- int **x**
  x **座標**
- int **y**
  y **座標**
- USHORT **z**
  z **座標**

3.11.1 詳解

抽出された座標を保存する構造体 (c37)

PathTrackingAndInductionOfTheRobot.hpp の 18 行目に定義があります。

3.11.2 メンバ詳解

3.11.2.1 int Point3::x

x 座標

PathTrackingAndInductionOfTheRobot.hpp の 19 行目に定義があります。

3.11.2.2 int Point3::y

y 座標

PathTrackingAndInductionOfTheRobot.hpp の 20 行目に定義があります。

3.11.2.3 USHORT Point3::z

z 座標

PathTrackingAndInductionOfTheRobot.hpp の 21 行目に定義があります。

この構造体詳解は次のファイルから抽出されました:

- **PathTrackingAndInductionOfTheRobot.hpp**

3.12 PointCloudLibrary クラス

点群処理を行うクラス

`#include <PointCloudLibrary.hpp>`

公開メンバ関数

- **PointCloudLibrary** ()
  コンストラクタ
- **PointCloudLibrary** (bool passthroughflag, bool downsamplingflag, bool statisticaloutlierremovalflag, bool mlsflag, bool extractplaneflag)
  コンストラクタ *(c64)*
- ∼**PointCloudLibrary** ()
  デストラクタ
- void **initializePCLVisualizer** (string pclvisualizer_name)
  PCL Visualizer の初期化
- void **loadPLY** (char ∗ply_name)
  .ply ファイルの読み込み
- void **flagChecker** ()
  フラグを判定するメソッド *(c64)*
- pcl::PointCloud
  < pcl::PointXYZRGB >::Ptr **passThroughFilter** (pcl::PointCloud< pcl::PointXYZRGB >::Ptr &inputPointCloud, char ∗axis, float min, float max)
  パススルーフィルタ．*z* の値の距離に応じてカット可能
- pcl::PointCloud
  < pcl::PointXYZRGB >::Ptr **removeOutlier** (pcl::PointCloud< pcl::PointXYZRGB >::Ptr &inputPointCloud)
  外れ値を除去するメソッド
- pcl::PointCloud
  < pcl::PointXYZRGB >::Ptr **radiusOutlierRemoval** (pcl::PointCloud< pcl::PointXYZRGB >::Ptr &inputPointCloud)
  外れ値を除去するメソッド
- pcl::PointCloud
  < pcl::PointXYZRGB >::Ptr **downSamplingUsingVoxelGridFilter** (pcl::PointCloud< pcl::PointXYZRGB >::Ptr &inputPointCloud, float leafSizeX, float leafSizeY, float leafSizeZ)
  ダウンサンプリングを行うメソッド
- pcl::PointCloud
  < pcl::PointXYZRGB >::Ptr **smoothingUsingMovingLeastSquare** (pcl::PointCloud< pcl::PointXYZRGB >::Ptr &inputPointCloud, bool compute_normals, bool polynomial_fit, double radius)
  スムージングを行うメソッド
- pcl::PointCloud
  < pcl::PointXYZRGB >::Ptr **getExtractPlaneAndClustering** (pcl::PointCloud< pcl::PointXYZRGB >::Ptr &inputPointCloud, bool optimize, int maxIterations, bool negative1, bool negative2, int minClusterSize, int maxClusterSize)
  平面検出とクラスタリング
- Point3d **getCentroidCoordinate3d** (pcl::PointCloud< pcl::PointXYZRGB >::Ptr &inputPointCloud)
  取得した点群の平均座標を取得するメソッド
- pcl::PointCloud< pcl::Normal >::Ptr **getSurfaceNormals** (pcl::PointCloud< pcl::PointXYZRGB >::Ptr &inputPointCloud)
  法線を計算する
- void **outputPointCloud** (int save_count, char ∗original_dirpath, char ∗output_filename, pcl::PointCloud< pcl::PointXYZRGB >::Ptr &outputPointCloud)
  点群を出力するメソッド

- void **outputPointCloudPLY** (int save_count, char *original_dirpath, char *output_filename, pcl::PointCloud< pcl::PointXYZRGB >::Ptr &**outputPointCloud**)

  点群を *ply* 形式で保存するメソッド

公開変数類

- pcl::PointCloud < pcl::PointXYZRGB >::Ptr **model**

  .ply ファイルの点群 *(*モデル*)*
- Point3d **centroid**

  平均座標
- pcl::visualization::PCLVisualizer* **visualizer**

  PCL Visualizer.
- int **th**

  平面検出とクラスタリング時用の閾値スライダー変数
- int **tor**

  平面検出とクラスタリング時用のスライダー変数
- bool **passthrough_flag**

  パススルーフィルターを用いるかどうかのフラグ
- bool **downsampling_flag**

  ダウンサンプリングを行うかどうかのフラグ
- bool **statisticaloutlierremoval_flag**

  外れ値を除去するかどうかのフラグ
- bool **mls_flag**

  スムージングを行うかどうかのフラグ
- bool **extractplane_flag**

  平面検出とクラスタリングを行うかどうかのフラグ

3.12.1 詳解

点群処理を行うクラス

PointCloudLibrary.hpp の 19 行目に定義があります。

3.12.2 構築子と解体子

3.12.2.1 PointCloudLibrary::PointCloudLibrary ( )

コンストラクタ

メソッドPointCloudLibrary::PointCloudLibrary(). コンストラクタ

PointCloudLibrary.cpp の 15 行目に定義があります。

3.12.2.2 PointCloudLibrary::PointCloudLibrary (bool *passthroughflag,* bool *downsamplingflag,* bool *statisticaloutlierremovalflag,* bool *mlsflag,* bool *extractplaneflag*)

コンストラクタ (c64)

メソッドPointCloudLibrary::PointCloudLibrary(). コンストラクタ (c64)

引数

flag_remove-Outlier	bool 型．外れ値除去のためのフラグ変数
flag_-downsampling	bool 型．ダウンサンプリングのためのフラグ変数
flag_MLS	bool 型．MLS のためのフラグ変数
flag_extract-Plane	bool 型．平面検出のためのフラグ変数

PointCloudLibrary.cpp の 27 行目に定義があります。

参照先 downsampling_flag, extractplane_flag, mls_flag, passthrough_flag, statisticaloutlierremoval_flag, th, tor.

3.12.2.3 PointCloudLibrary::∼PointCloudLibrary ( )

デストラクタ

メソッドPointCloudLibrary::∼PointCloudLibrary(). デストラクタ

PointCloudLibrary.cpp の 43 行目に定義があります。

3.12.3 関数詳解

3.12.3.1 pcl::PointCloud< pcl::PointXYZRGB >::Ptr PointCloudLibrary::downSamplingUsingVoxelGridFilter (pcl::PointCloud< pcl::PointXYZRGB >::Ptr & *inputPointCloud,* float *leafSizeX,* float *leafSizeY,* float *leafSizeZ*)

ダウンサンプリングを行うメソッド

メソッドPointCloudLibrary::downSamplingUsingVoxelGridFilter(). ダウンサンプリング処理を行うメソッド (c59)

引数

&inputPoint-Cloud	pcl::PointCloud<pcl::PointXYZ>::Ptr型．入力するポイントクラウド
leafSizeX	float 型．x のリーフサイズ
leafSizeY	float 型．y のリーフサイズ
leafSizeZ	float 型．z のリーフサイズ

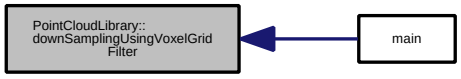
戻り値

filtered pcl::PointCloud<pcl::PointXYZ>::Ptr型．出力するポイントクラウド

PointCloudLibrary.cpp の 173 行目に定義があります。

参照元 main().

被呼び出し関係図:



3.12.3.2 void PointCloudLibrary::flagChecker ( )

フラグを判定するメソッド (c64)

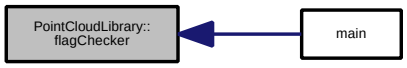
メソッドPointCloudLibrary::flagChecker().PCL 処理に関する処理の有無を判定するフラグ変数を反転させるメソッド (c64)

PointCloudLibrary.cpp の 77 行目に定義があります。

参照先 downsampling_flag, extractplane_flag, mls_flag, passthrough_flag, statisticaloutlierremoval_flag.

参照元 main().

被呼び出し関係図:



3.12.3.3 Point3d PointCloudLibrary::getCentroidCoordinate3d (pcl::PointCloud< pcl::PointXYZRGB >::Ptr & *inputPointCloud*)

取得した点群の平均座標を取得するメソッド

メソッド getCentroidCoordinate

引数

&inputPoint-Cloud	pcl::PointCloud<pcl::PointXYZRGB>::Ptr型．入力するポイントクラウド

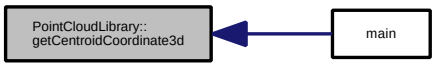
戻り値

centroid Point3f 型．抽出した範囲の重心

PointCloudLibrary.cpp の 331 行目に定義があります。

参照元 main().

被呼び出し関係図:



3.12.3.4 pcl::PointCloud< pcl::PointXYZRGB >::Ptr PointCloudLibrary::getExtractPlaneAndClustering (pcl::PointCloud< pcl::PointXYZRGB >::Ptr & *inputPointCloud*, bool *optimize*, int *maxIterations*, bool *negative1*, bool *negative2*, int *minClusterSize*, int *maxClusterSize*)

平面検出とクラスタリング

メソッドPointCloudLibrary::extractPlane(). 平面を検出するメソッド

引数

&inputPoint-Cloud	pcl::PointCloud<pcl::PointXYZ>::Ptr型．入力するポイントクラウド
optimize	bool 型．最適化するかどうかのフラグ
maxIterations	int 型．最大繰り返し回数
negative1	bool 型．抽出範囲を除くか保持するかのフラグ
negative2	bool 型．抽出範囲を除くか保持するかのフラグ

minClusterSize	int 型．クラスタの最小サイズ
maxClusterSizer	int 型．クラスタの最大サイズ

戻り値

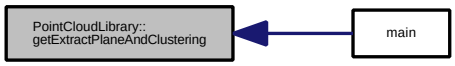
filtered pcl::PointCloud<pcl::PointXYZ>::Ptr 型．出力するポイントクラウド

PointCloudLibrary.cpp の 229 行目に定義があります。

参照先 th, tor.

参照元 main().

被呼び出し関係図:



3.12.3.5 pcl::PointCloud< pcl::Normal >::Ptr PointCloudLibrary::getSurfaceNormals (pcl::PointCloud< pcl::PointXYZRGB >::Ptr & *inputPointCloud*)

法線を計算する

メソッドPointCloudLibrary::getSurfaceNormals()．法線を計算する

引数

&inputPoint-Cloud	pcl::PointCloud<pcl::PointXYZRGB>::Ptr 型．入力するポイントクラウド

戻り値

cloud_normals pcl::PointCloud<pcl::Normal>::Ptr 型．出力するポイントクラウド

PointCloudLibrary.cpp の 379 行目に定義があります。

3.12.3.6 void PointCloudLibrary::initializePCLVisualizer (string *pclvisualizer_name*)

PCL Visualizer の初期化

メソッドPointCloudLibrary::initializePCLVisualizer()．PCL Visualizer の初期化を行うメソッド

引数

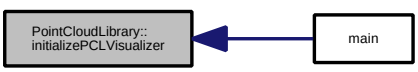
pclvisualizer_-name	string 型．PCL Visualizer の名前

PointCloudLibrary.cpp の 52 行目に定義があります。

参照先 visualizer.

参照元 main().

被呼び出し関係図:



3.12.3.7 void PointCloudLibrary::loadPLY (char * *ply_name*)

.ply ファイルの読み込み

メソッドPointCloudLibrary::loadPLY(). ply ファイルを読み込む

引数

ply_name	char* 型. 読み込みたい.ply ファイルの名前

PointCloudLibrary.cpp の 67 行目に定義があります。

参照先 model.

3.12.3.8 void PointCloudLibrary::outputPointCloud (int *save_count*, char * *original_dirpath*, char * *output_filename*, pcl::PointCloud< pcl::PointXYZRGB >::Ptr & *outputPointCloud*)

点群を出力するメソッド

メソッドPointCloudLibrary::outputPointCloud(). 時間と速度のプロット

引数

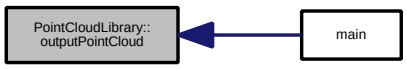
save_count	int 型. 保存数のカウント
original_dirpath	char* 型. 基となるディレクトリ名
output_filename	char* 型. 出力するファイル名
&outputPoint-Cloud	pcl::PointCloud<pcl::PointXYZRGB>::Ptr型. 入力するポイントクラウド

PointCloudLibrary.cpp の 404 行目に定義があります。

参照先 NOC.

参照元 main().

被呼び出し関係図:



3.12.3.9 void PointCloudLibrary::outputPointCloudPLY (int *save_count*, char * *original_dirpath*, char * *output_filename*, pcl::PointCloud< pcl::PointXYZRGB >::Ptr & *outputPointCloud*)

点群を ply 形式で保存するメソッド

メソッドPointCloudLibrary::outputPointCloudPLY(). 時間とヨー角のプロット

引数

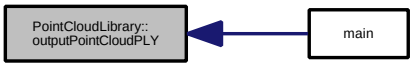
save_count	int 型. 保存数のカウント
original_dirpath	char* 型. 基となるディレクトリ名
output_filename	char* 型. 出力するファイル名
&outputPoint-Cloud	pcl::PointCloud<pcl::PointXYZRGB>::Ptr型. 出力するポイントクラウド

PointCloudLibrary.cpp の 425 行目に定義があります。

参照先 NOC.

参照元 main().

被呼び出し関係図:



3.12.3.10 pcl::PointCloud< pcl::PointXYZRGB >::Ptr PointCloudLibrary::passThroughFilter (pcl::PointCloud< pcl::PointXYZRGB >::Ptr & *inputPointCloud*, char * *axis*, float *min*, float *max*)

パススルーフィルタ．z の値の距離に応じてカット可能

メソッドPointCloudLibrary::passThroughFilter(). パススルーフィルタ

引数

&inputPoint-Cloud	pcl::PointCloud<pcl::PointXYZ>::Ptr型．入力するポイントクラウド
axis	char* 型．フィルタをかけたい軸の名前
min	float 型．フィルターの調整用の最小変数
max	float 型．フィルターの調整用の最大変数

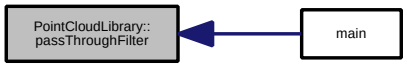
戻り値

filtered pcl::PointCloud<pcl::PointXYZ>::Ptr型．フィルタ処理後のポイントクラウド

PointCloudLibrary.cpp の 106 行目に定義があります。

参照元 main().

被呼び出し関係図:



3.12.3.11 pcl::PointCloud< pcl::PointXYZRGB >::Ptr PointCloudLibrary::radiusOutlierRemoval (pcl::PointCloud< pcl::PointXYZRGB >::Ptr & *inputPointCloud*)

外れ値を除去するメソッド

メソッドPointCloudLibrary::radiusOutlierRemoval(). 外れ値を除去するメソッド (c60)

引数

&inputPoint-Cloud	pcl::PointCloud<pcl::PointXYZ>::Ptr型．入力するポイントクラウド

戻り値

filtered pcl::PointCloud<pcl::PointXYZ>::Ptr型．出力するポイントクラウド

PointCloudLibrary.cpp の 149 行目に定義があります。

3.12.3.12 pcl::PointCloud< pcl::PointXYZRGB >::Ptr PointCloudLibrary::removeOutlier (pcl::PointCloud< pcl::PointXYZRGB >::Ptr & *inputPointCloud*)

外れ値を除去するメソッド

メソッドPointCloudLibrary::removeOutlier(). 外れ値を除去するメソッド (c59)

引数

&inputPoint-Cloud	pcl::PointCloud<pcl::PointXYZ>::Ptr型．入力するポイントクラウド

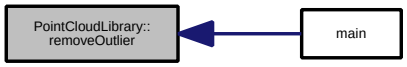
戻り値

filtered pcl::PointCloud<pcl::PointXYZ>::Ptr型．出力するポイントクラウド

PointCloudLibrary.cpp の 127 行目に定義があります。

参照元 main().

被呼び出し関係図:



3.12.3.13 pcl::PointCloud< pcl::PointXYZRGB >::Ptr PointCloudLibrary::smoothingUsingMovingLeastSquare (pcl::PointCloud< pcl::PointXYZRGB >::Ptr & *inputPointCloud*, bool *compute_normals*, bool *polynomial_fit*, double *radius*)

スムージングを行うメソッド

メソッドPointCloudLibrary::smoothingUsingMovingLeastSquare(). スムージングを行うメソッド (c60)

引数

&inputPoint-Cloud	pcl::PointCloud<pcl::PointXYZ>::Ptr型．入力するポイントクラウド
compute_-normals	bool 型．法線を計算するかどうか
polynominal_fit	bool 型．チェックするためのフラグ
radius	double 型．半径

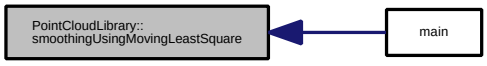
戻り値

filtered pcl::PointCloud<pcl::PointXYZ>::Ptr型．出力するポイントクラウド

PointCloudLibrary.cpp の 198 行目に定義があります。

参照元 main().

被呼び出し関係図:



3.12.4 メンバ詳解

3.12.4.1 Point3d PointCloudLibrary::centroid

平均座標

PointCloudLibrary.hpp の 42 行目に定義があります。

参照元 main().

3.12.4.2 bool PointCloudLibrary::downsampling_flag

ダウンサンプリングを行うかどうかのフラグ

PointCloudLibrary.hpp の 55 行目に定義があります。

参照元 flagChecker(), main(), PointCloudLibrary().

3.12.4.3 bool PointCloudLibrary::extractplane_flag

平面検出とクラスタリングを行うかどうかのフラグ

PointCloudLibrary.hpp の 58 行目に定義があります。

参照元 flagChecker(), main(), PointCloudLibrary().

3.12.4.4 bool PointCloudLibrary::mls_flag

スムージングを行うかどうかのフラグ

PointCloudLibrary.hpp の 57 行目に定義があります。

参照元 flagChecker(), main(), PointCloudLibrary().

3.12.4.5 pcl::PointCloud<pcl::PointXYZRGB>::Ptr PointCloudLibrary::model

.ply ファイルの点群 (モデル)

PointCloudLibrary.hpp の 30 行目に定義があります。

参照元 loadPLY().

3.12.4.6 bool PointCloudLibrary::passthrough_flag

パススルーフィルターを用いるかどうかのフラグ

PointCloudLibrary.hpp の 54 行目に定義があります。

参照元 flagChecker(), main(), PointCloudLibrary().

3.12.4.7 bool PointCloudLibrary::statisticaloutlierremoval_flag

外れ値を除去するかどうかのフラグ

PointCloudLibrary.hpp の 56 行目に定義があります。

参照元 flagChecker(), main(), PointCloudLibrary().

3.12.4.8 int PointCloudLibrary::th

平面検出とクラスタリング時用の閾値スライダー変数

PointCloudLibrary.hpp の 50 行目に定義があります。

参照元 getExtractPlaneAndClustering(), PointCloudLibrary().

3.12.4.9 int PointCloudLibrary::tor

平面検出とクラスタリング時用のスライダー変数

PointCloudLibrary.hpp の 51 行目に定義があります。

参照元 getExtractPlaneAndClustering(), PointCloudLibrary().

3.12.4.10 pcl::visualization::PCLVisualizer* PointCloudLibrary::visualizer

PCL Visualizer.

PointCloudLibrary.hpp の 48 行目に定義があります。

参照元 initializePCLVisualizer(), main().

このクラス詳解は次のファイルから抽出されました:

- **PointCloudLibrary.hpp**
- **PointCloudLibrary.cpp**

3.13 System クラス

システム関連の処理を行うクラス

`#include <System.hpp>`

公開メンバ関数

- **System** ()
  コンストラクタ
- ~**System** ()
  デストラクタ
- void **countdownTimer** (int time_sec)
  引数の時間 *[sec]* に応じてカウントダウンを開始する *(c75)*
- void **startMessage** ()
  プログラム開始時のメッセージを表示 *(c26)*
- void **endMessage** (int cNum)
  プログラム終了時のメッセージを表示 *(c38)*
- void **endMessage** ()
  プログラム終了時のメッセージを表示 *(c63)*
- void **showHelpMessage** ()
  キー入力に関するヘルプを表示 *(c86)*
- void **startTimer** ()
  タイマーを開始 *(c65)*
- void **endTimer** ()
  タイマーを終了 *(c65)*
- double **getProcessTimeinMiliseconds** ()
  計測した時間をミリ秒単位で取得.*startTimer()* と *endTimer()* が実行されていることが前提 *(c65)*
- double **getFrameRate** ()
  フレームレートを取得.*startTimer()* と *endTimer()* が実行されていることが前提 *(c65)*
- void **checkDirectory** (const char *check_dirname)
  引数に与えたファイルやディレクトリが存在するかチェックし，無ければ作成する *(c81)*
- void **makeDirectoryBasedDate** ()
  日付に基づいたディレクトリの作成
- void **makeDirectory** (char *original_path, int create_dirnum)
  指定したディレクトリ以下に新しいディレクトリを作成する *(*
- void **removeDirectory** ()

取得したデータが不要だった場合ディレクトリを削除する

- int **alternatives** ()

  数字の入力をチェックする

- void **openDirectory** ()

  ディレクトリを開く *(c38)*

- VideoWriter **outputVideo** (const string *outputVideoName)

  動画を出力する

公開変数類

- double **time**

  処理時間の結果

- double **fps**

  フレームレート

- double **sum_time**

  処理の合計時間

3.13.1 詳解

システム関連の処理を行うクラス

System.hpp の 19 行目に定義があります。

3.13.2 構築子と解体子

3.13.2.1 System::System ( )

コンストラクタ

System::System() (p. 43). コンストラクタ

System.cpp の 15 行目に定義があります。

参照先 fps, sum_time, time.

3.13.2.2 System::~System ( )

デストラクタ

System::~System() (p. 43). デストラクタ

System.cpp の 28 行目に定義があります。

3.13.3 関数詳解

3.13.3.1 int System::alternatives ( )

数字の入力をチェックする

System::alternatives() (p. 43).Yes/No の 2 択のチェック (c21)

戻り値

checkNum int 型．Yes かNo の値

<0 か 1 をチェックするための変数 (c27). このメソッドのみで有効な変数 (c30)

System.cpp の 295 行目に定義があります。

参照元 main().

被呼び出し関係図:

3.13.3.2 void System::checkDirectory (const char * *check_dirname*)

引数に与えたファイルやディレクトリが存在するかチェックし，無ければ作成する (c81)

System::checkDirectory() (p. 44)．ファイルやディレクトリがあるかチェックし，無ければ作成する (c81)

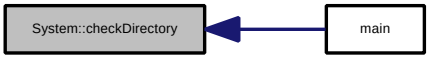
引数

checkdir_name	const char* 型．ディレクトリ名

System.cpp の 233 行目に定義があります。

参照元 main().

被呼び出し関係図:



3.13.3.3 void System::countdownTimer (int *time_sec*)

引数の時間 [sec] に応じてカウントダウンを開始する (c75)

メソッド countdownTimer()．タイマーによるカウントダウン

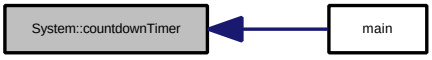
引数

time_sec	int 型．カウントダウンしたい時間 [sec]

System.cpp の 108 行目に定義があります。

参照元 main().

被呼び出し関係図:



3.13.3.4 void System::endMessage (int *cNum*)

プログラム終了時のメッセージを表示 (c38)

System::endMessage() (p. 45). プログラム終了時のメッセージ

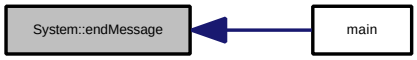
System.cpp の 60 行目に定義があります。

参照先 directoryName, openDirectory().

参照元 main().

呼び出し関係図:

被呼び出し関係図:



3.13.3.5 void System::endMessage ( )

プログラム終了時のメッセージを表示 (c63)

System::endMessage() (p. 45). プログラム終了時のメッセージ

System.cpp の 76 行目に定義があります。

3.13.3.6 void System::endTimer ( )

タイマーを終了 (c65)

メソッド endTimer(). タイマーを終了する

System.cpp の 168 行目に定義があります。

参照先 time.

参照元 main().

被呼び出し関係図:

3.13.3.7 double System::getFrameRate ( )

フレームレートを取得.startTimer() と endTimer() が実行されていることが前提 (c65)

メソッド getFrameRate(). フレームレートを取得する (c65)

戻り値

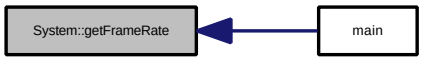
time double 型. フレームレート取得時間

System.cpp の 209 行目に定義があります。

参照先 fps, time.

参照元 main().

被呼び出し関係図:



3.13.3.8 double System::getProcessTimeinMiliseconds ( )

計測した時間をミリ秒単位で取得.startTimer() と endTimer() が実行されていることが前提 (c65)

メソッド getTime(). 計測した時間を取得する (c65)

戻り値

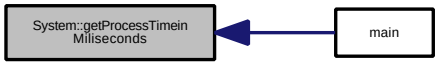
time double 型. 処理時間.

System.cpp の 186 行目に定義があります。

参照先 time.

参照元 main().

被呼び出し関係図:



3.13.3.9 void System::makeDirectory (char * *original_path*, int *create_dirnum*)

指定したディレクトリ以下に新しいディレクトリを作成する (

System::makeDirectory() (p. 46). 指定したディレクトリ以下に新しいディレクトリを作成する

引数

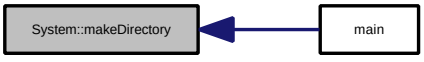
original_path	char* 型. 指定したベースとなるディレクトリ
create_dirnum	int 型. ディレクトリ番号

System.cpp の 266 行目に定義があります。

参照先 NOC.

参照元 main().

被呼び出し関係図:



3.13.3.10 void System::makeDirectoryBasedDate ( )

日付に基づいたディレクトリの作成

System::makeDirectory() (p. 46). ディレクトリを作成

System.cpp の 247 行目に定義があります。

参照先 directoryName, NOC.

参照元 main().

被呼び出し関係図:

3.13.3.11 void System::openDirectory ( )

ディレクトリを開く (c38)

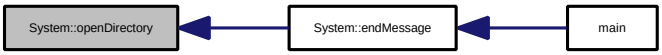
System::openDireectory(). 出力したディレクトリを開く (c39)

System.cpp の 324 行目に定義があります。

参照先 directoryName, NOC.

参照元 endMessage().

被呼び出し関係図:



3.13.3.12 VideoWriter System::outputVideo (const string * *outputVideoName*)

動画を出力する

System::outputVideo() (p. 47). 動作確認用に動画を出力するメソッド

引数

outputVideo-Name	const string* 型. 出力したい動画のファイル名

戻り値

writer VideoWriter 型. 出力する動画の情報

< 動画出力時のパス (c38)

System.cpp の 339 行目に定義があります。

参照先 directoryName, NOC.

3.13.3.13 void System::removeDirectory ( )

取得したデータが不要だった場合ディレクトリを削除する

System::removeDirectory() (p. 47). ディレクトリを削除 (c21)

System.cpp の 278 行目に定義があります。

参照先 directoryName, NOC.

参照元 main().

被呼び出し関係図:

3.13.3.14 void System::showHelpMessage ( )

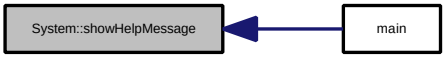
キー入力に関するヘルプを表示 (c86)

メソッドSystem::showHelpMessage(). ヘルプメッセージを表示する

System.cpp の 87 行目に定義があります。

参照元 main().

被呼び出し関係図:



3.13.3.15 void System::startMessage ( )

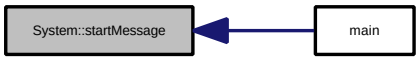
プログラム開始時のメッセージを表示 (c26)

System::startMessage() (p. 48). プログラム起動時のメッセージ

System.cpp の 36 行目に定義があります。

参照元 main().

被呼び出し関係図:



3.13.3.16 void System::startTimer ( )

タイマーを開始 (c65)

メソッド startTimer(). タイマーを開始する

System.cpp の 157 行目に定義があります。

参照元 main().

被呼び出し関係図:

3.13.4 メンバ詳解

3.13.4.1 double System::fps

フレームレート

System.hpp の 45 行目に定義があります。

参照元 getFrameRate(), System().

3.13.4.2 double System::sum_time

処理の合計時間

System.hpp の 46 行目に定義があります。

参照元 main(), System().

3.13.4.3 double System::time

処理時間の結果

System.hpp の 44 行目に定義があります。

参照元 endTimer(), getFrameRate(), getProcessTimeinMiliseconds(), System().

このクラス詳解は次のファイルから抽出されました:

- **System.hpp**
- **System.cpp**

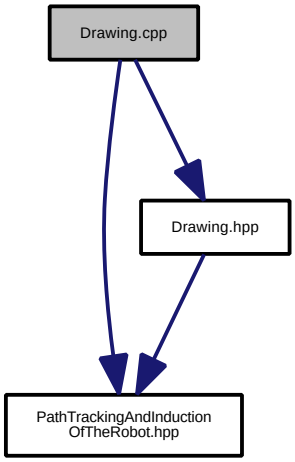
Chapter 4

ファイル詳解

4.1 Drawing.cpp ファイル

```
#include "PathTrackingAndInductionOfTheRobot.hpp"
#include "Drawing.hpp"
```

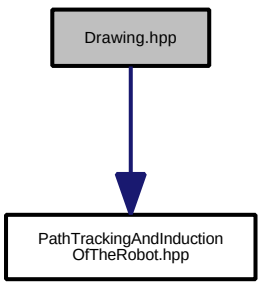
Drawing.cpp の依存先関係図:



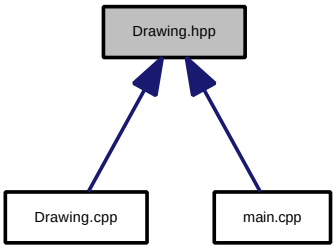
4.2 Drawing.hpp ファイル

```
#include "PathTrackingAndInductionOfTheRobot.hpp"
```

Drawing.hpp の依存先関係図:



被依存関係図:



クラス

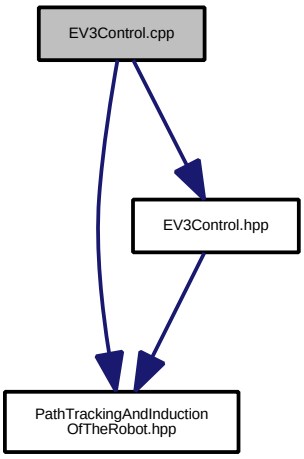
- class **Drawing**

  経路描画用のクラス

4.3 EV3Control.cpp ファイル

```
#include "PathTrackingAndInductionOfTheRobot.hpp"
#include "EV3Control.hpp"
```

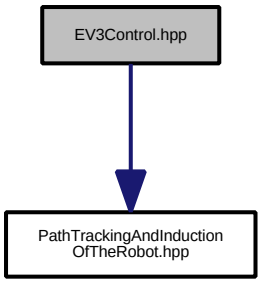
EV3Control.cpp の依存先関係図:



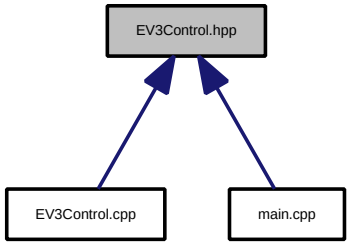
4.4 EV3Control.hpp ファイル

```
#include "PathTrackingAndInductionOfTheRobot.hpp"
```

EV3Control.hpp の依存先関係図:



被依存関係図:



クラス

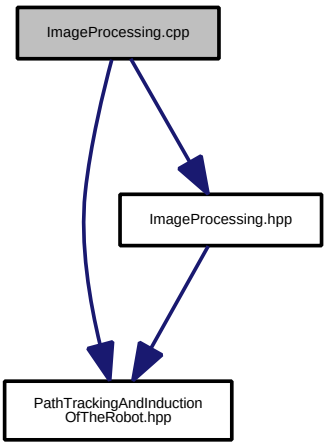
- class **EV3Control**

  EV3 を制御するためのクラス

4.5 ImageProcessing.cpp ファイル

```
#include "PathTrackingAndInductionOfTheRobot.hpp"
#include "ImageProcessing.hpp"
```

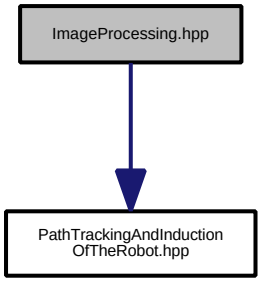
ImageProcessing.cpp の依存先関係図:



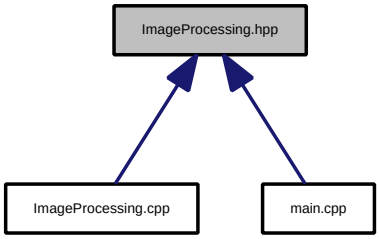
4.6 ImageProcessing.hpp ファイル

```
#include "PathTrackingAndInductionOfTheRobot.hpp"
```

ImageProcessing.hpp の依存先関係図:



被依存関係図:



クラス

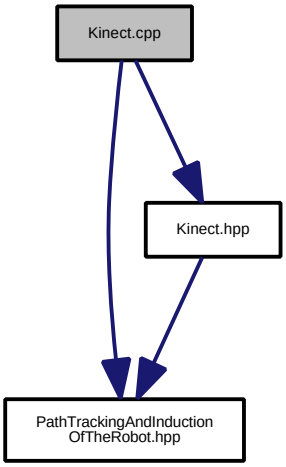
- class **ImageProcessing**

  画像処理用のクラス

4.7 Kinect.cpp ファイル

```
#include "PathTrackingAndInductionOfTheRobot.hpp"
#include "Kinect.hpp"
```

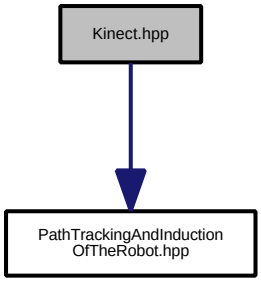
Kinect.cpp の依存先関係図:



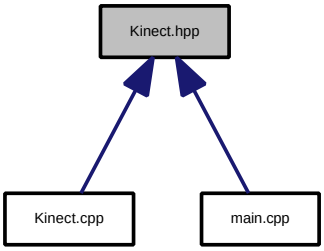
4.8 Kinect.hpp ファイル

```
#include "PathTrackingAndInductionOfTheRobot.hpp"
```

Kinect.hpp の依存先関係図:



被依存関係図:



クラス

- class **Kinect**

  Kinect 操作用のクラス

マクロ定義

- #define **ERROR_CHECK**(ret)
  エラーチェック

変数

- const NUI_IMAGE_RESOLUTION **CAMERA_RESOLUTION** = NUI_IMAGE_RESOLUTION_640x480

4.8.1 マクロ定義詳解

4.8.1.1 #define ERROR_CHECK(*ret*)

値:

```
if (ret != S_OK){ \
stringstream ss; \
ss << "faild" #ret " " << hex << ret << endl; \
throw runtime_error(ss.str().c_str()); \
}
```

エラーチェック

Kinect.hpp の 16 行目に定義があります。

参照元 Kinect::createInstance(), Kinect::drawRGBImage(), Kinect::getPointCloud(), Kinect::initialize().

4.8.2 変数詳解

4.8.2.1 const NUI_IMAGE_RESOLUTION CAMERA_RESOLUTION = NUI_IMAGE_RESOLUTION_640x480

Kinect の解像度の設定

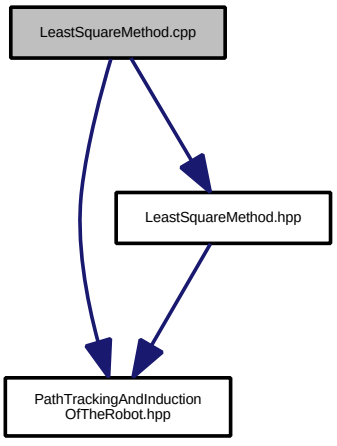
Kinect.hpp の 24 行目に定義があります。

参照元 Kinect::getPointCloud(), Kinect::initialize().

4.9 LeastSquareMethod.cpp ファイル

```
#include "PathTrackingAndInductionOfTheRobot.hpp"
#include "LeastSquareMethod.hpp"
```

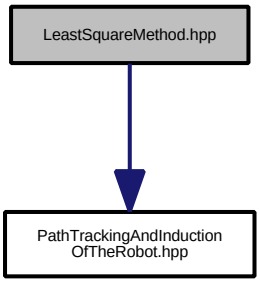
LeastSquareMethod.cpp の依存先関係図:



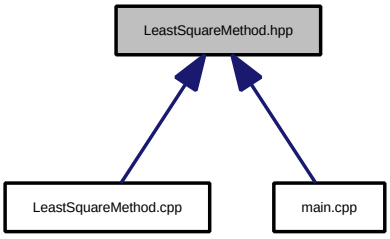
4.10 LeastSquareMethod.hpp ファイル

```
#include "PathTrackingAndInductionOfTheRobot.hpp"
```

LeastSquareMethod.hpp の依存先関係図:



被依存関係図:



クラス

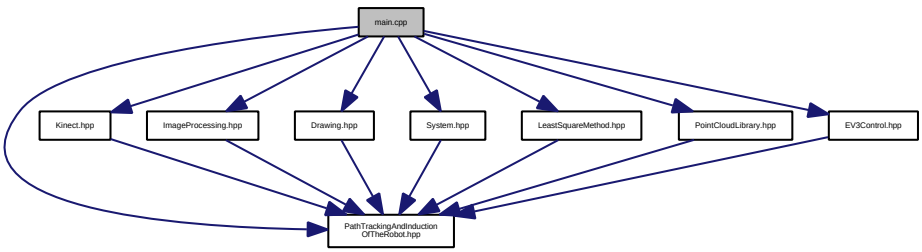
- class **LeastSquareMethod**

  最小二乗法を行うクラス

4.11 main.cpp ファイル

```
#include "PathTrackingAndInductionOfTheRobot.hpp"
#include "Kinect.hpp"
#include "ImageProcessing.hpp"
#include "Drawing.hpp"
#include "System.hpp"
#include "LeastSquareMethod.hpp"
#include "PointCloudLibrary.hpp"
#include "EV3Control.hpp"
```

main.cpp の依存先関係図:



関数

- void **onMouse** (int event, int x, int y, int flags, void *param)

  マウス操作
- int **main** ()

  関数 *main()*

変数

- Mat **image**

  RGB 画像格納用の変数
- char **directoryName** [**NOC**]

  フォルダ名
- bool **selectObject** = false

  オブジェクト選択
- int **trackObject** = 0

  追跡するオブジェクト
- Point **origin**

  オリジナルの座標
- Rect **selection**

  選択

4.11.1 関数詳解

4.11.1.1 int main ( )

関数 main()

戻り値

0 正常終了
-1 異常終了

pointcloudlibrary.viewer->wasStopped() &&

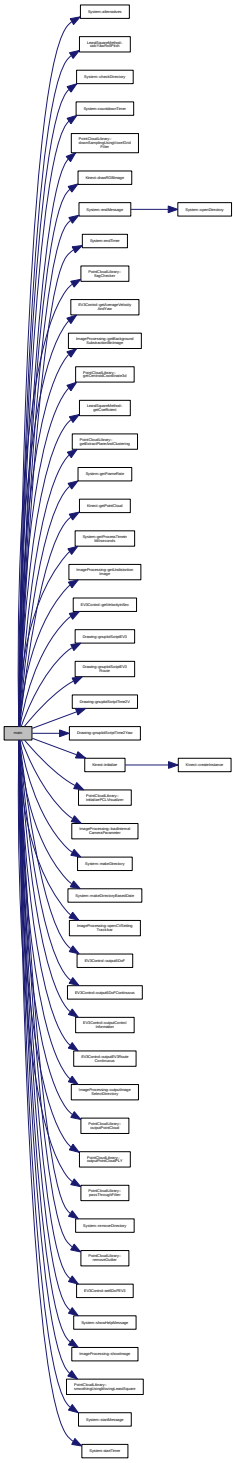
< 平均座標を確認するための球

main.cpp の 36 行目に定義があります。

参照先 System::alternatives(), LeastSquareMethod::attitude_angle, LeastSquareMethod::calcYawRollPitch(), PointCloudLibrary::centroid, System::checkDirectory(), LeastSquareMethod::coefficient_plane, System::countdownTimer(), directoryName, PointCloudLibrary::downsampling_flag, PointCloudLibrary::downSamplingUsingVoxelGridFilter(), Kinect::drawRGBImage(), System::endMessage(), System::endTimer(), PointCloudLibrary::extractplane_flag, PointCloudLibrary::flagChecker(), EV3Control::getAverageVelocityAndYaw(), ImageProcessing::getBackgroundSubstractionBinImage(), PointCloudLibrary::getCentroidCoordinate3d(), LeastSquareMethod::getCoefficient(), PointCloudLibrary::getExtractPlaneAndClustering(), System::getFrameRate(), Kinect::getPointCloud(), System::getProcessTimeinMiliseconds(), ImageProcessing::getUndistortionImage(), EV3Control::getVelocityinSec(), Drawing::gnuplotScriptEV3(), Drawing::gnuplotScriptEV3Route(), Drawing::gnuplotScriptTime2V(), Drawing::gnuplotScriptTime2Yaw(), image, Kinect::initialize(), PointCloudLibrary::initializePCLVisualizer(), Kinect::key, ImageProcessing::loadInternalCameraParameter(), System::makeDirectory(), System::makeDirectoryBasedDate(), PointCloudLibrary::mls_flag, NOC, ImageProcessing::openCVSettingTrackbar(), EV3Control::output6DoF(), EV3Control::output6DoFContinuous(), EV3Control::outputControlInformation(), EV3Control::outputEV3RouteContinuous(), ImageProcessing::outputImageSelectDirectory(), PointCloudLibrary::outputPointCloud(), PointCloudLibrary::outputPointCloudPLY(), PointCloudLibrary::passthrough_flag, PointCloudLibrary::passThroughFilter(),

System::removeDirectory(), PointCloudLibrary::removeOutlier(), EV3Control::save_flag, EV3Control::set6DoFEV3(), System::showHelpMessage(), ImageProcessing::showImage(), PointCloudLibrary::smoothingUsingMovingLeastSquare(), System::startMessage(), System::startTimer(), PointCloudLibrary::statisticaloutlierremoval_flag, Kinect::streamEvent, System::sum_time, PointCloudLibrary::visualizer (計 61 項目).

呼び出し関係図:



4.11.1.2 void onMouse (int *event,* int *x,* int *y,* int *flags,* void ∗ *param*)

マウス操作

マウス操作

引数

event	int 型．イベント
x	int 型．x 座標
y	int 型．y 座標
flags	int 型．フラグ
param	void∗ 型．その他パラメータ

Mouse.cpp の 19 行目に定義があります。

4.11.2 変数詳解

4.11.2.1 char directoryName[NOC]

フォルダ名

main.cpp の 22 行目に定義があります。

参照元 System::endMessage(), Drawing::gnuplotScriptTime2V(), Drawing::gnuplotScriptTime2Yaw(), main(), System::makeDirectoryBasedDate(), System::openDirectory(), System::outputVideo(), System::removeDirectory().

4.11.2.2 Mat image

RGB 画像格納用の変数

main.cpp の 20 行目に定義があります。

参照元 Kinect::getPointCloud(), main(), onMouse().

4.11.2.3 Point origin

オリジナルの座標

main.cpp の 27 行目に定義があります。

参照元 onMouse().

4.11.2.4 Rect selection

選択

main.cpp の 28 行目に定義があります。

参照元 onMouse().

4.11.2.5 bool selectObject = false

オブジェクト選択

main.cpp の 25 行目に定義があります。

参照元 onMouse().

4.11.2.6 int trackObject = 0

追跡するオブジェクト

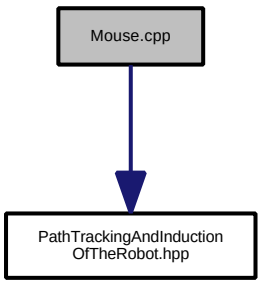
main.cpp の 26 行目に定義があります。

参照元 onMouse().

4.12 Mouse.cpp ファイル

```
#include "PathTrackingAndInductionOfTheRobot.hpp"
```

Mouse.cpp の依存先関係図:



関数

- void **onMouse** (int event, int x, int y, int flags, void *param)

  メソッド *onMouse()*. マウス左クリックで座標を取得するメソッド

4.12.1 関数詳解

4.12.1.1 void onMouse (int *event*, int *x*, int *y*, int *flags*, void * *param*)

メソッド onMouse(). マウス左クリックで座標を取得するメソッド

マウス操作

引数

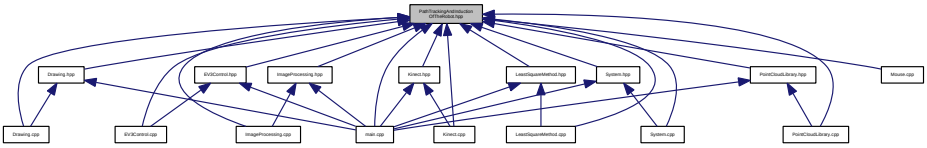
event	int 型. イベント
x	int 型. x 座標
y	int 型. y 座標
flags	int 型. フラグ
param	void* 型. その他パラメータ

Mouse.cpp の 19 行目に定義があります。

参照先 image, origin, selection, selectObject, trackObject.

4.13 PathTrackingAndInductionOfTheRobot.hpp ファイル

被依存関係図:



クラス

- struct **Point3**
  抽出された座標を保存する構造体 *(c37)*
- struct **outputData**
  ファイルに出力するデータ群 *(c41)*
- struct **DoF6d**
  姿勢構造体の定義 *(c78)*. *doule* 型
- struct **DoF6i**
  姿勢構造体の定義. *int* 型
- struct **AttitudeAngle**
  姿勢角
- struct **ControlParamd**
  走行制御用構造体. *double* 型

型定義

- typedef struct **Point3 Point3ius**
- typedef struct **outputData outputData**
- typedef struct **DoF6d DoF6d**
- typedef struct **DoF6i DoF6i**
- typedef struct **AttitudeAngle AttitudeAngle3d**
- typedef struct **ControlParamd ControlParamd**

関数

- void **onMouse** (int event, int x, int y, int, void ∗)
  マウス操作

変数

- Mat **image**
  RGB 画像格納用の変数
- char **directoryName** [**NOC**]
  フォルダ名
- bool **selectObject**
  オブジェクト選択
- int **trackObject**
  追跡するオブジェクト
- Point **origin**

オリジナルの座標

- Rect **selection**

  選択

4.13.1 型定義詳解

4.13.1.1 typedef struct AttitudeAngle AttitudeAngle3d

4.13.1.2 typedef struct ControlParamd ControlParamd

4.13.1.3 typedef struct DoF6d DoF6d

4.13.1.4 typedef struct DoF6i DoF6i

4.13.1.5 typedef struct outputData outputData

4.13.1.6 typedef struct Point3 Point3ius

4.13.2 関数詳解

4.13.2.1 void onMouse (int *event,* int *x,* int *y,* int *flags,* void ∗ *param*)

マウス操作

マウス操作

引数

event	int 型．イベント
x	int 型．x 座標
y	int 型．y 座標
flags	int 型．フラグ
param	void∗ 型．その他パラメータ

Mouse.cpp の 19 行目に定義があります。

参照先 image, origin, selection, selectObject, trackObject.

4.13.3 変数詳解

4.13.3.1 char directoryName[NOC]

フォルダ名

main.cpp の 22 行目に定義があります。

参照元 System::endMessage(), Drawing::gnuplotScriptTime2V(), Drawing::gnuplotScriptTime2Yaw(), main(), System::makeDirectoryBasedDate(), System::openDirectory(), System::outputVideo(), System::removeDirectory().

4.13.3.2 Mat image

RGB 画像格納用の変数

main.cpp の 20 行目に定義があります。

参照元 Kinect::getPointCloud(), main(), onMouse().

4.13.3.3 Point origin

オリジナルの座標

main.cpp の 27 行目に定義があります。

参照元 onMouse().

4.13.3.4 Rect selection

選択

main.cpp の 28 行目に定義があります。

参照元 onMouse().

4.13.3.5 bool selectObject

オブジェクト選択

main.cpp の 25 行目に定義があります。

参照元 onMouse().

4.13.3.6 int trackObject

追跡するオブジェクト

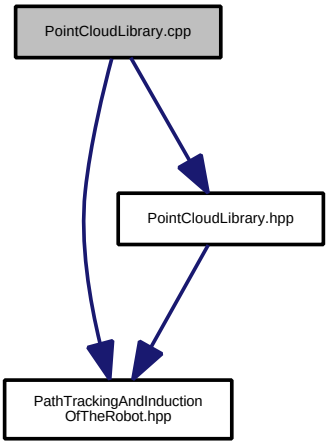
main.cpp の 26 行目に定義があります。

参照元 onMouse().

4.14 PointCloudLibrary.cpp ファイル

```
#include "PathTrackingAndInductionOfTheRobot.hpp"
#include "PointCloudLibrary.hpp"
```

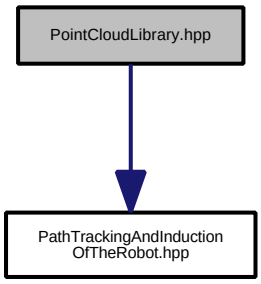
PointCloudLibrary.cpp の依存先関係図:



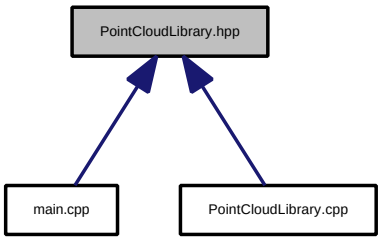
4.15 PointCloudLibrary.hpp ファイル

```
#include "PathTrackingAndInductionOfTheRobot.hpp"
```

PointCloudLibrary.hpp の依存先関係図:



被依存関係図:



クラス

- class **PointCloudLibrary**

  点群処理を行うクラス

4.16 stdafx.cpp ファイル

4.17 stdafx.h ファイル

標準のシステム，インクルードファイル，または参照回数が多く，かつあまり変更されない，プロジェクト専用のインクルードファイルを記述する．

```
#include <iostream>
#include <sstream>
#include <fstream>
#include <string>
#include <stdio.h>
#include <stdlib.h>
#include <direct.h>
#include <math.h>
#include <ctype.h>
#include <sys/stat.h>
#include <Windows.h>
#include <mmsystem.h>
#include <NuiApi.h>
#include <opencv2\opencv.hpp>
#include <opencv2\core\core.hpp>
#include <opencv2\highgui\highgui.hpp>
#include <opencv2\imgproc\imgproc.hpp>
#include <opencv2\video\tracking.hpp>
#include <opencv2\flann\flann.hpp>
#include <pcl\point_types.h>
#include <pcl\point_cloud.h>
#include <pcl\io\io.h>
#include <pcl\io\pcd_io.h>
#include <pcl\io\ply_io.h>
#include <pcl\common\common_headers.h>
#include <pcl\visualization\cloud_viewer.h>
#include <pcl\visualization\pcl_visualizer.h>
#include <pcl\filters\passthrough.h>
#include <pcl\filters\statistical_outlier_removal.h>
#include <pcl\filters\radius_outlier_removal.h>
#include <pcl\kdtree\kdtree_flann.h>
#include <pcl\surface\mls.h>
#include <pcl\filters\voxel_grid.h>
#include <pcl\ModelCoefficients.h>
#include <pcl\sample_consensus\method_types.h>
#include <pcl\sample_consensus\model_types.h>
#include <pcl\segmentation\sac_segmentation.h>
#include <pcl\filters\extract_indices.h>
#include <pcl\features\normal_3d.h>
#include <pcl\segmentation\extract_clusters.h>
#include <pcl\features\integral_image_normal.h>
#include <boost\thread\thread.hpp>
#include <pcl\registration\icp.h>
#include <pcl\PCLPointField.h>
```

stdafx.h の依存先関係図:

マクロ定義

- #define **_CRT_SECURE_NO_WARNINGS**
- #define **WIDTH** 640

  ＜ 名前空間
- #define **HEIGHT** 480

  画像の高さ
- #define **ALLPIXEL WIDTH∗HEIGHT**

1 フレームの全ピクセル数

- #define **NOC** 128

  Number of Characters. (ファイルの名前を付けるときの文字数制限)

4.17.1 詳解

標準のシステム，インクルードファイル，または参照回数が多く，かつあまり変更されない，プロジェクト専用のインクルードファイルを記述する．

日付

2015.11.22

著者

H.Shigehara

stdafx.h に定義があります。

4.17.2 マクロ定義詳解

4.17.2.1 #define _CRT_SECURE_NO_WARNINGS

stdafx.h の 11 行目に定義があります。

4.17.2.2 #define ALLPIXEL WIDTH*HEIGHT

1 フレームの全ピクセル数

stdafx.h の 80 行目に定義があります。

4.17.2.3 #define HEIGHT 480

画像の高さ

stdafx.h の 79 行目に定義があります。

4.17.2.4 #define NOC 128

Number of Characters. (ファイルの名前を付けるときの文字数制限)

stdafx.h の 81 行目に定義があります。

参照元 Drawing::gnuplotScriptEV3(), Drawing::gnuplotScriptEV3Route(), Drawing::gnuplotScriptTime2V(), Drawing::gnuplotScriptTime2Yaw(), main(), System::makeDirectory(), System::makeDirectoryBasedDate(), System::openDirectory(), EV3Control::output6DoF(), EV3Control::output6DoFContinuous(), EV3Control::output-ControlInformation(), EV3Control::outputEV3RouteContinuous(), ImageProcessing::outputImageSelectDirectory(), PointCloudLibrary::outputPointCloud(), PointCloudLibrary::outputPointCloudPLY(), System::outputVideo(), System-::removeDirectory() (計 17 項目).

4.17.2.5 #define WIDTH 640

＜名前空間

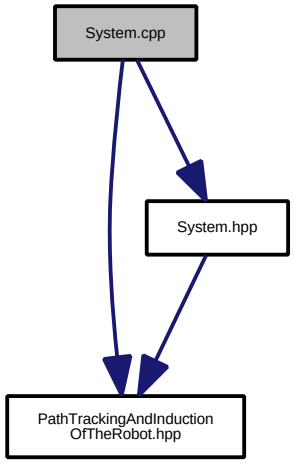
画像の幅

stdafx.h の 78 行目に定義があります。

4.18 System.cpp ファイル

```
#include "PathTrackingAndInductionOfTheRobot.hpp"
#include "System.hpp"
```

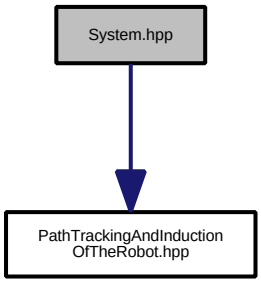
System.cpp の依存先関係図:



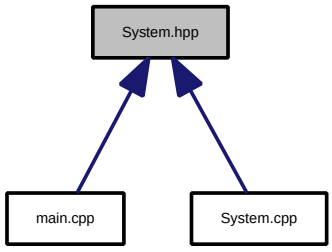
4.19 System.hpp ファイル

```
#include "PathTrackingAndInductionOfTheRobot.hpp"
```

System.hpp の依存先関係図:



被依存関係図:



クラス

- class **System**

システム関連の処理を行うクラス

Index