

INF552

Generated by Doxygen 1.8.13

Contents

1	Projet-Inf552	1
2	Class Index	3
2.1	Class List	3
3	File Index	5
3.1	File List	5
4	Class Documentation	7
4.1	Plan Class Reference	7
4.1.1	Constructor & Destructor Documentation	7
4.1.1.1	Plan() [1/3]	7
4.1.1.2	Plan() [2/3]	7
4.1.1.3	Plan() [3/3]	8
4.1.2	Member Function Documentation	8
4.1.2.1	distance()	8
4.1.2.2	regression()	8
4.1.3	Friends And Related Function Documentation	9
4.1.3.1	operator<<	9
4.2	point3d Class Reference	9
4.2.1	Constructor & Destructor Documentation	9
4.2.1.1	point3d()	10
4.2.2	Member Function Documentation	10
4.2.2.1	getColor()	10
4.2.2.2	getPosition()	10

4.3	point3dCloud Class Reference	10
4.3.1	Constructor & Destructor Documentation	11
4.3.1.1	point3dCloud()	11
4.3.2	Member Function Documentation	11
4.3.2.1	operator[]()	11
4.3.2.2	push_back()	11
4.3.2.3	size()	11
4.4	Ransac Class Reference	11
4.4.1	Constructor & Destructor Documentation	12
4.4.1.1	Ransac()	12
4.4.2	Member Function Documentation	12
4.4.2.1	fit()	12
5	File Documentation	13
5.1	plan.cpp File Reference	13
5.1.1	Function Documentation	13
5.1.1.1	operator<<()	13
5.2	plan.hpp File Reference	13
5.3	point3d.cpp File Reference	14
5.4	point3d.hpp File Reference	14
5.5	point3dCloud.cpp File Reference	14
5.6	point3dCloud.hpp File Reference	14
5.7	projet.cpp File Reference	15
5.7.1	Typedef Documentation	15
5.7.1.1	json	15
5.7.2	Function Documentation	15
5.7.2.1	hasToBeTreated()	16
5.7.2.2	main()	17
5.7.2.3	pointCloud2ply()	17
5.7.2.4	pointCloudFromImages()	17
5.7.2.5	setcursor()	18
5.8	ransac.cpp File Reference	18
5.9	ransac.hpp File Reference	18
5.10	README.md File Reference	18
	Index	19

Chapter 1

Projet-Inf552

But du projet INPUT => image de gauche + disparité => à partir de deux images, créer le nuage de point 3D => pour chaque pixel, extraire les coordonnées x, y, z correspondantes => en déduire l'altitude des pixels (détecter le sol, objets verticaux, ...)

K deux images matrices rotation, translation Considérer origine Cam1

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Plan	7
point3d	9
point3dCloud	10
Ransac	11

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

plan.cpp	13
plan.hpp	13
point3d.cpp	14
point3d.hpp	14
point3dCloud.cpp	14
point3dCloud.hpp	14
projet.cpp	15
ransac.cpp	18
ransac.hpp	18

Chapter 4

Class Documentation

4.1 Plan Class Reference

```
#include <plan.hpp>
```

Public Member Functions

- **Plan** ()
- **Plan** (double a, double b, double c, double d)
- **Plan** (Vec3d p1, Vec3d p2, Vec3d p3)
- double **distance** (Vec3d p)
- void **regression** (**point3dCloud** pointcloud)

Friends

- ostream & **operator**<< (ostream &os, const **Plan** &p)

4.1.1 Constructor & Destructor Documentation

4.1.1.1 Plan() [1/3]

```
Plan::Plan ( )
```

Constructor for the class.

4.1.1.2 Plan() [2/3]

```
Plan::Plan (
    double a,
    double b,
    double c,
    double d )
```

Constructor for the class.

Parameters

<i>a</i>	1st parameter of $a*x + b*y + c*z + d = 0$
<i>b</i>	2nd parameter of $a*x + b*y + c*z + d = 0$
<i>c</i>	3rd parameter of $a*x + b*y + c*z + d = 0$
<i>d</i>	4th parameter of $a*x + b*y + c*z + d = 0$

4.1.1.3 Plan() [3/3]

```
Plan::Plan (
    Vec3d p1,
    Vec3d p2,
    Vec3d p3 )
```

Constructor for the class.

Parameters

<i>p1</i>	1st point of the plan
<i>p2</i>	2nd point of the plan
<i>p3</i>	3rd point of the plan

4.1.2 Member Function Documentation**4.1.2.1 distance()**

```
double Plan::distance (
    Vec3d p )
```

Find the distance between the plan and a point.

Parameters

<i>p</i>	The considered point.
----------	-----------------------

4.1.2.2 regression()

```
void Plan::regression (
    point3dCloud pointcloud )
```

Find the closest plan to the given point cloud.

Parameters

<i>pointcloud</i>	The considered point cloud.
-------------------	-----------------------------

Returns

The **Plan** (p. 7) of linear regression.

4.1.3 Friends And Related Function Documentation

4.1.3.1 operator<<

```
ostream& operator<< (
    ostream & os,
    const Plan & p ) [friend]
```

Overloads ofstream.

Parameters

<i>os</i>	Considered stream.
<i>p</i>	Considered plan.

The documentation for this class was generated from the following files:

- **plan.hpp**
- **plan.cpp**

4.2 point3d Class Reference

```
#include <point3d.hpp>
```

Public Member Functions

- **point3d** (Vec3d position, Vec3b color)
- Vec3d **getPosition** ()
- Vec3b **getColor** ()

4.2.1 Constructor & Destructor Documentation

4.2.1.1 point3d()

```
point3d::point3d (
    Vec3d position,
    Vec3b color )
```

Constructor for the class.

Parameters

<i>position</i>	The position of the 3dPoint.
<i>color</i>	The color of the 3dPoint.

4.2.2 Member Function Documentation

4.2.2.1 getColor()

```
Vec3b point3d::getColor ( )
```

4.2.2.2 getPosition()

```
Vec3d point3d::getPosition ( )
```

The documentation for this class was generated from the following files:

- **point3d.hpp**
- **point3d.cpp**

4.3 point3dCloud Class Reference

```
#include <point3dCloud.hpp>
```

Public Member Functions

- **point3dCloud** ()
- void **push_back** (**point3d** point)
- **point3d** **operator[]** (int i)
- int **size** ()

4.3.1 Constructor & Destructor Documentation

4.3.1.1 point3dCloud()

```
point3dCloud::point3dCloud ( )
```

Constructor for the class.

4.3.2 Member Function Documentation

4.3.2.1 operator[]()

```
point3d point3dCloud::operator[] (
    int i )
```

4.3.2.2 push_back()

```
void point3dCloud::push_back (
    point3d point )
```

4.3.2.3 size()

```
int point3dCloud::size ( )
```

The documentation for this class was generated from the following files:

- **point3dCloud.hpp**
- **point3dCloud.cpp**

4.4 Ransac Class Reference

```
#include <ransac.hpp>
```

Public Member Functions

- **Ransac** (int n_iterations, double epsilon)
- **point3dCloud fit** (**point3dCloud** pointCloud)

4.4.1 Constructor & Destructor Documentation

4.4.1.1 Ransac()

```
Ransac::Ransac (
    int n_iterations,
    double epsilon )
```

Constructor for the class.

Parameters

<i>n_iterations</i>	The number of iterations for the algorithm.
<i>epsilon</i>	Threshold.

4.4.2 Member Function Documentation

4.4.2.1 fit()

```
point3dCloud Ransac::fit (
    point3dCloud pointCloud )
```

Extract the most correlated points (plane model).

Parameters

<i>pointCloud</i>	The considered point cloud.
-------------------	-----------------------------

Returns

List of points that correlate the most (plane model) as vector<pair<Vec3d, Vec3b>>.

The documentation for this class was generated from the following files:

- **ransac.hpp**
- **ransac.cpp**

Chapter 5

File Documentation

5.1 plan.cpp File Reference

```
#include "plan.hpp"
```

Functions

- ostream & **operator**<< (ostream &os, const **Plan** &p)

5.1.1 Function Documentation

5.1.1.1 operator<<()

```
ostream& operator<< (  
    ostream & os,  
    const Plan & p )
```

Overloads ofstream.

Parameters

<i>os</i>	Considered stream.
<i>p</i>	Considered plan.

5.2 plan.hpp File Reference

```
#include <iostream>  
#include <opencv2/highgui/highgui.hpp>
```

```
#include <opencv2/features2d/features2d.hpp>
#include <opencv2/calib3d/calib3d.hpp>
#include <opencv2/imgproc.hpp>
#include "point3dCloud.hpp"
```

Classes

- class **Plan**

5.3 point3d.cpp File Reference

```
#include "point3d.hpp"
```

5.4 point3d.hpp File Reference

```
#include <iostream>
#include <windows.h>
#include <fstream>
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/features2d/features2d.hpp>
#include <opencv2/calib3d/calib3d.hpp>
#include <opencv2/imgproc.hpp>
```

Classes

- class **point3d**

5.5 point3dCloud.cpp File Reference

```
#include "point3dCloud.hpp"
```

5.6 point3dCloud.hpp File Reference

```
#include <iostream>
#include <windows.h>
#include <fstream>
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/features2d/features2d.hpp>
#include <opencv2/calib3d/calib3d.hpp>
#include <opencv2/imgproc.hpp>
#include "point3d.hpp"
```

Classes

- class **point3dCloud**

5.7 projet.cpp File Reference

```
#include <iostream>
#include <windows.h>
#include <fstream>
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/features2d/features2d.hpp>
#include <opencv2/calib3d/calib3d.hpp>
#include <opencv2/imgproc.hpp>
#include "include\json.hpp"
#include "ransac.hpp"
```

Typedefs

- using **json** = nlohmann::json

Functions

- void **setcursor** (bool visible, DWORD size)
- bool **hasToBeTreated** (int i, int j, double d, const Mat &left_image)
- void **pointCloud2ply** (**point3dCloud** pointcloud)
- **point3dCloud** **pointCloudFromImages** (Mat &left_image, const Mat &disparity, Matx3d N)
- int **main** ()

5.7.1 Typedef Documentation

5.7.1.1 json

```
using json = nlohmann::json
```

This program intents to reconstruct a 3D scene from two images taken from a car in a street and to detect elements such as the road or the vertical objects

Author

Lucas Broux & Romain Loiseau

5.7.2 Function Documentation

5.7.2.1 hasToBeTreated()

```
bool hasToBeTreated (
    int i,
    int j,
    double d,
    const Mat & left_image ) [inline]
```

Determines if the pixel should be treated.

Parameters

<i>d</i>	The disparity between left/right images.
----------	--

Returns

Whether the pixel should be considered.

5.7.2.2 main()

```
int main ( )
```

5.7.2.3 pointCloud2ply()

```
void pointCloud2ply (
    point3dCloud pointcloud )
```

Generates .ply file from point cloud values.

Parameters

<i>pointcloud</i>	The corresponding point cloud.
-------------------	--------------------------------

5.7.2.4 pointCloudFromImages()

```
point3dCloud pointCloudFromImages (
    Mat & left_image,
    const Mat & disparity,
    Matx33d N )
```

Generates a 3d point cloud from left image + disparity + transformation matrix. Exports the result as .ply file.

Parameters

<i>left_image</i>	The left image.
<i>disparity</i>	The disparity.
<i>N</i>	The matrix of correspondence : it can transform the disparity into 3d point.

Returns

The point cloud as vector<pair<Vec3d, Vec3b>>.

5.7.2.5 setcursor()

```
void setcursor (
    bool visible,
    DWORD size )
```

Function for hiding/showing cursor : hiding with setcursor(0, 0); reinitialisation with setcursor(1, 10).

Parameters

<i>visible</i>	Whether the cursor should be visible.
<i>size</i>	The size of the cursor.

5.8 ransac.cpp File Reference

```
#include "ransac.hpp"
```

5.9 ransac.hpp File Reference

```
#include <iostream>
#include <windows.h>
#include <fstream>
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/features2d/features2d.hpp>
#include <opencv2/calib3d/calib3d.hpp>
#include <opencv2/imgproc.hpp>
#include "plan.hpp"
```

Classes

- class **Ransac**

5.10 README.md File Reference

Index

- distance
 - Plan, 8
- fit
 - Ransac, 12
- getColor
 - point3d, 10
- getPosition
 - point3d, 10
- hasToBeTreated
 - projet.cpp, 15
- json
 - projet.cpp, 15
- main
 - projet.cpp, 17
- operator<<
 - Plan, 9
 - plan.cpp, 13
- operator[]
 - point3dCloud, 11
- Plan, 7
 - distance, 8
 - operator<<, 9
 - Plan, 7, 8
 - regression, 8
- plan.cpp, 13
 - operator<<, 13
- plan.hpp, 13
- point3d, 9
 - getColor, 10
 - getPosition, 10
 - point3d, 9
- point3d.cpp, 14
- point3d.hpp, 14
- point3dCloud, 10
 - operator[], 11
 - point3dCloud, 11
 - push_back, 11
 - size, 11
- point3dCloud.cpp, 14
- point3dCloud.hpp, 14
- pointCloud2ply
 - projet.cpp, 17
- pointCloudFromImages
 - projet.cpp, 17
- projet.cpp, 15
 - hasToBeTreated, 15
 - json, 15
 - main, 17
 - pointCloud2ply, 17
 - pointCloudFromImages, 17
 - setcursor, 18
- push_back
 - point3dCloud, 11
- README.md, 18
- Ransac, 11
 - fit, 12
 - Ransac, 12
- ransac.cpp, 18
- ransac.hpp, 18
- regression
 - Plan, 8
- setcursor
 - projet.cpp, 18
- size
 - point3dCloud, 11