

Shape Demo

Generated by Doxygen 1.8.11

Contents

| | | |
|----------|---|----------|
| 1 | File Index | 1 |
| 1.1 | File List | 1 |
| 2 | File Documentation | 3 |
| 2.1 | multiloc.h File Reference | 3 |
| 2.1.1 | Typedef Documentation | 5 |
| 2.1.1.1 | one_Tuple | 5 |
| 2.1.1.2 | three_Tuple | 5 |
| 2.1.1.3 | two_Tuple | 5 |
| 2.1.2 | Function Documentation | 5 |
| 2.1.2.1 | classification(const one_Tuple &list_one, three_Tuple &list) | 5 |
| 2.1.2.2 | clearreplace(three_Tuple &means_result, three_Tuple &f_result, int &numberrun- count) | 5 |
| 2.1.2.3 | Demo1Single_Target(one_Tuple &angle, one_Tuple &data, MatrixXd &result) . . | 6 |
| 2.1.2.4 | Demo2(one_Tuple &m_data, two_Tuple &angle, three_Tuple &list_three, three_↵ _Tuple &list_three2) | 6 |
| 2.1.2.5 | Demo3(three_Tuple &f_result, three_Tuple &list_three, string &str, int &number- runcount) | 6 |
| 2.1.2.6 | Demo3count(two_Tuple &A, two_Tuple &B) | 7 |
| 2.1.2.7 | Demo3intersect(two_Tuple &A, two_Tuple &B, ofstream &out_stream, int number- runcount) | 7 |
| 2.1.2.8 | Demo3remove(two_Tuple &A, three_Tuple &B, ofstream &out_stream, int &num- berruncount) | 7 |
| 2.1.2.9 | Demo4Mult_Targetlocalization(three_Tuple &f_result, two_Tuple &list_two, string &str) | 8 |
| 2.1.2.10 | Demo5inputdata(two_Tuple &angles, one_Tuple &m_coordinate) | 8 |
| 2.1.2.11 | finditems(double value, two_Tuple &list) | 8 |

| | | |
|--------------|---|-----------|
| 2.1.2.12 | finditems(double value, one_Tuple &list) | 9 |
| 2.1.2.13 | isequal(const two_Tuple &list, const two_Tuple &list1) | 9 |
| 2.1.2.14 | productiondot(one_Tuple &m_data, two_Tuple &angle, three_Tuple &list_three, int &numberruncount) | 9 |
| 2.1.2.15 | sortarry(two_Tuple &list, int temp) | 10 |
| 2.1.3 | Variable Documentation | 10 |
| 2.1.3.1 | numberruncount | 10 |
| 2.2 | utils.h File Reference | 10 |
| 2.2.1 | Function Documentation | 11 |
| 2.2.1.1 | print(one_Tuple &list) | 11 |
| 2.2.1.2 | print(two_Tuple &list) | 11 |
| 2.2.1.3 | print(three_Tuple &list) | 11 |
| 2.2.1.4 | print(MatrixXd &list, int &total) | 12 |
| 2.2.1.5 | read_data(const char *filename, one_Tuple &tuples) | 12 |
| 2.2.1.6 | read_file(string &Docement, one_Tuple &m_data, two_Tuple &angle) | 12 |
| 2.2.1.7 | read_file(string &Docement, one_Tuple &m_data, one_Tuple &angle) | 12 |
| Index | | 15 |

Chapter 1

File Index

1.1 File List

Here is a list of all files with brief descriptions:

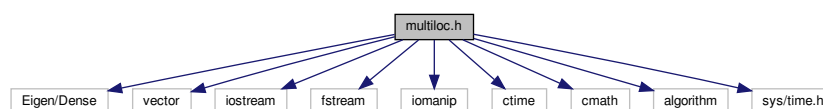
| | | |
|----------------------------|-------|----|
| multiloc.h | | 3 |
| utils.h | | 10 |

Chapter 2

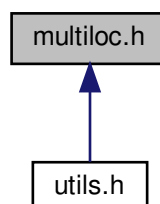
File Documentation

2.1 multiloc.h File Reference

```
#include "Eigen/Dense"  
#include <vector>  
#include <iostream>  
#include <fstream>  
#include <iomanip>  
#include <ctime>  
#include <cmath>  
#include <algorithm>  
#include <sys/time.h>  
Include dependency graph for multiloc.h:
```



This graph shows which files directly or indirectly include this file:



Typedefs

- typedef vector< double > [one_Tuple](#)
- typedef vector< [one_Tuple](#) > [two_Tuple](#)
- typedef vector< [two_Tuple](#) > [three_Tuple](#)

Functions

- void [sortarry](#) ([two_Tuple](#) &list, int temp)
对一个二维的vector<vector<double>>>的list进行按照每一个vector<double>的第2个元素进行升序
- bool [finditems](#) (double value, [two_Tuple](#) &list)
判断value元素是否属于2维vector中的第一个元素,如果value属于list中返回true,否则返回false.
- bool [finditems](#) (double value, [one_Tuple](#) &list)
判断value元素是否属于1维vector的list,如果value属于list中返回true,否则返回false.
- int [classification](#) (const [one_Tuple](#) &list_one, [three_Tuple](#) &list)
坐标点聚类算法,即判断一维的vector<double>类型是否属于交点坐标聚类集合list中的一类,如果属于其中的一类,则将该1维的vector归入到该类中,否则单独归为list中新的一类.
- bool [isequal](#) (const [two_Tuple](#) &list, const [two_Tuple](#) &list1)
判断两个二维vector即list和list1是否相等,如果两个相等,返回true.否则返回false.
- void [productiondot](#) ([one_Tuple](#) &m_data, [two_Tuple](#) &angle, [three_Tuple](#) &list_three, int &numberruncount)
测向线交点坐标聚类算法,即第一步读取靶点和测向线的数据信息,将两两测向线进行相交的到交点坐标.第2步即将交点坐标送入坐标聚类算法中,进行坐标点的聚类.其中m_data为靶点的坐标. angle为每个靶点对应的测向线的角度信息. list_three为输出的坐标点聚类集合. numberruncount为测向线数量用于剔除独立的测向线聚类集合.
- void [clearreplace](#) ([three_Tuple](#) &means_result, [three_Tuple](#) &f_result, int &numberruncount)
将坐标点聚类集合转化为测向线聚类集合.其中means_result为输入的坐标点聚类集合. f_result为输出的测向线聚类集合. numberruncount为测向线数量用于剔除独立的测向线聚类集合.
- void [Demo4Mult_Targetlocalization](#) ([three_Tuple](#) &f_result, [two_Tuple](#) &list_two, string &str)
将测向线聚类集合送入目标定位算法中,进行目标点定位.其中f_result为输入的测向线聚类集合. list_two为所有目标点的坐标. str为目标点的坐标信息以文件形式输出的文件路径.
- void [Demo1Single_Target](#) ([one_Tuple](#) &angle, [one_Tuple](#) &data, MatrixXd &result)
单目标定位算法根据输入的测向线角度和靶点坐标求解目标点的具体位置信息. data为所有靶点的坐标. angle为每个靶点对应测向线的角度信息. 定位出来的目标点坐标将保存在result.
- void [Demo2](#) ([one_Tuple](#) &m_data, [two_Tuple](#) &angle, [three_Tuple](#) &list_three, [three_Tuple](#) &list_three2)
对靶点数据和测向线数据通过聚类算法得到测向线聚类坐标集合.其中m_data为所有靶点的坐标. angle为每个靶点对应测向线的角度信息. 交点坐标聚类集合为list_three, 测向线组聚类集合为list_three2.
- void [Demo3count](#) ([two_Tuple](#) &A, [two_Tuple](#) &B)
判断测向线组中任意两类是否具有重复的测向线信息.将第一类中非重复的部分归入第2类中. A为一类的测向线聚类组. B为另一类的测向线聚类组.
- void [Demo3intersect](#) ([two_Tuple](#) &A, [two_Tuple](#) &B, ofstream &out_stream, int numberruncount)
判断测向线组中任意两类是否具有重复的测向线信息.将重复的多余的部分进行删除. A为一类的测向线聚类组. B为另一类的测向线聚类组. numberruncount为测向线数量用于剔除独立的测向线聚类集合.
- void [Demo3](#) ([three_Tuple](#) &f_result, [three_Tuple](#) &list_three, string &str, int &numberruncount)
判断一类测向线聚类组与整个测向线聚类集合中每一类是否具有重复的测向线信息.将重复多余的测向线信息进行删除,同时更新测向线聚类集合. A为一类的测向线聚类组. B为所有测向线聚类集合. numberruncount为测向线数量用于剔除独立的测向线聚类集合.
- void [Demo3remove](#) ([two_Tuple](#) &A, [three_Tuple](#) &B, ofstream &out_stream, int &numberruncount)
将测向线聚类集合进行测向线野值检测.将重复多余的测向线信息进行删除,同时更新测向线聚类集合.将删除的测向线信息保存到文件中. f_result为一类的测向线聚类组. list_three为所有测向线聚类集合. str为删除的测向线信息以文件形式输出的路径. numberruncount为测向线数量用于剔除独立的测向线聚类集合.
- bool [Demo5inputdata](#) ([two_Tuple](#) &angles, [one_Tuple](#) &m_coordinate)
测试数据数据通过命令行进行在线输入,对输入的靶点数据坐标和测向线角度信息进行保存,剔除重复输入的信息.输入的测向线角度信息保存在angles,输入的靶点坐标信息保存在m_coordinate.当输入出现数据格式出现错误时,返回false.正确返回true.

Variables

- int [numberruncount](#)

2.1.1 Typedef Documentation

2.1.1.1 typedef vector<double> one_Tuple

使用one_Tuple代替vector<double>

2.1.1.2 typedef vector<two_Tuple> three_Tuple

使用three_Tuple代替vector<vector<vector<double>>>

2.1.1.3 typedef vector<one_Tuple> two_Tuple

使用two_Tuple代替vector<vector<double>>

2.1.2 Function Documentation

2.1.2.1 int classification (const one_Tuple & list_one, three_Tuple & list)

坐标点聚类算法,即判断一维的vector<double>类型是否属于交点坐标聚类集合list中的一类,如果属于其中的一类,则将该1维的vector归入到该类中,否则单独归为list中新的一类.

Parameters

| | |
|---|--|
| <i>list_one</i> 是vector<double>类型 | |
| <i>list</i> 为vector<vector<vector<double>>>类型 | |

Returns

int类型

2.1.2.2 void clearreplace (three_Tuple & means_result, three_Tuple & f_result, int & numberruncount)

将坐标点聚类集合转化为测向线聚类集合. 其中means_result为输入的坐标点聚类集合. f_result为输出的测向线聚类集合. numberruncount为测向线数量用于剔除独立的测向线聚类集合.

Parameters

| | |
|---|--|
| <i>means_result</i> 为vector<vector<double>>类型 | |
| <i>f_result</i> 为vector<vector<double>>类型 | |
| <i>numberruncount</i> 为int类型 | |

Returns

无.

2.1.2.3 void Demo1Single_Target (one_Tuple & angle, one_Tuple & data, MatrixXd & result)

单目标定位算法根据输入的测向线角度和靶点坐标求解目标点的具体位置信息。data为所有靶点的坐标。angle为每个靶点对应测向线的角度信息。定位出来的目标点坐标将保存在result。

Parameters

| | |
|------------------------|--|
| angle为vector<double>类型 | |
| data为vector<double>类型 | |
| result为MatrixXd类型 | |

Returns

无.

2.1.2.4 void Demo2 (one_Tuple & m_data, two_Tuple & angle, three_Tuple & list_three, three_Tuple & list_three2)

对靶点数据和测向线数据通过聚类算法得到测向线聚类坐标集合。其中m_data为所有靶点的坐标。angle为每个靶点对应测向线的角度信息。交点坐标聚类集合为list_three，测向线组聚类集合为list_three2。

Parameters

| | |
|------------------------|--|
| angle为vector<double>类型 | |
| data为vector<double>类型 | |
| result为MatrixXd类型 | |

Returns

无

2.1.2.5 void Demo3 (three_Tuple & f_result, three_Tuple & list_three, string & str, int & numberruncount)

判断一类测向线聚类组与整个测向线聚类集合中每一类是否具有重复的测向线信息。将重复多余的测向线信息进行删除，同时更新测向线聚类集合。A为一类的测向线聚类组。B为所有测向线聚类集合。numberruncount为测向线数量用于剔除独立的测向线聚类集合。

Parameters

| | |
|------------------------------------|--|
| A为vector<vector<double>>类型 | |
| B为vector<vector<vector<double>>>类型 | |
| numberruncount为int类型 | |

Returns

无

2.1.2.6 void Demo3count (two_Tuple & A, two_Tuple & B)

判断测向线组中任意两类是否具有重复的测向线信息. 将第一类中非重复的部分归入第2类中. A为一类的测向线聚类组. B为另一类的测向线聚类组.

Parameters

| | |
|----------------------------|--|
| A为vector<vector<double>>类型 | |
| B为vector<vector<double>>类型 | |

Returns

无

2.1.2.7 void Demo3intersect (two_Tuple & A, two_Tuple & B, ofstream & out_stream, int numberruncount)

判断测向线组中任意两类是否具有重复的测向线信息. 将重复的多余的部分进行删除. A为一类的测向线聚类组. B为另一类的测向线聚类组. numberruncount为测向线数量用于剔除独立的测向线聚类集合.

Parameters

| | |
|----------------------------|--|
| A为vector<vector<double>>类型 | |
| B为vector<vector<double>>类型 | |
| numberruncount为int类型 | |

Returns

无

2.1.2.8 void Demo3remove (two_Tuple & A, three_Tuple & B, ofstream & out_stream, int & numberruncount)

将测向线聚类集合进行测向线野值检测. 将重复多余的测向线信息进行删除,同时更新测向线聚类集合.将删除的测向线信息保存到文件中. f_result为一类的测向线聚类组.list_three为所有测向线聚类集合.str为删除的测向线信息以文件形式输出的路径.numberruncount为测向线 数量用于剔除独立的测向线聚类集合.

Parameters

| | |
|-------------------------------------|--|
| A 为vector<vector<double>>类型 | |
| B 为vector<vector<vector<double>>>类型 | |
| str为string类型 | |
| numberruncount为int类型 | |

Returns

无

2.1.2.9 void Demo4Mult_Targetlocalization (three_Tuple & f_result, two_Tuple & list_two, string & str)

将测向线聚类集合送入目标定位算法中，进行目标点定位。其中f_result为输入的测向线聚类集合。list_two为所有目标点的坐标。str为目标点的坐标信息以文件形式输出的文件路径。

Parameters

| | |
|-----------------------------------|--|
| f_result为vector<vector<double>>类型 | |
| list_two为vector<vector<double>>类型 | |
| str为string类型 | |

Returns

无.

2.1.2.10 bool Demo5inputdata (two_Tuple & angles, one_Tuple & m_coordinate)

测试数据数据通过命令行进行在线输入,对输入的靶点数据坐标和测向线角度信息进行保存,剔除重复输入的信息. 输入的测向线角度信息保存在angles,输入的靶点坐标信息保存在m_coordinate.当输入出现数据格式出现错误时,返回false.正确返回true.

Parameters

| | |
|---------------------------------|--|
| angles为vector<vector<double>>类型 | |
| m_coordinate为<vector<double>类型 | |

Returns

bool类型

2.1.2.11 bool finditems (double value, two_Tuple & list)

判断value元素是否属于2维vector中的第一个元素,如果value属于list中返回true,否则返回false.

Parameters

| | |
|-------------------------------|--|
| value为double类型 | |
| list是vector<vector<double>>类型 | |

Returns

bool类型

2.1.2.12 bool finditems (double value, one_Tuple & list)

判断value元素是否属于1维vector的list,如果value属于list中返回true,否则返回false.

Parameters

| | |
|-------------------------------|--|
| <i>list</i> 是vector<double>类型 | |
| <i>value</i> 为double类型 | |

Returns

bool类型

2.1.2.13 bool isequal (const two_Tuple & list, const two_Tuple & list1)

判断两个二维vector即list和list1是否相等,如果两个相等,返回true.否则返回false.

Parameters

| | |
|--|--|
| <i>list</i> 为vector<vector<double>>类型 | |
| <i>list1</i> 为vector<vector<double>>类型 | |

Returns

bool类型

2.1.2.14 void productiondot (one_Tuple & m_data, two_Tuple & angle, three_Tuple & list_three, int & numberruncount)

测向线交点坐标聚类算法，即第一步读取靶点和测向线的数据信息，将两两测向线进行相交的到交点坐标。第2步即将交点坐标送入坐标聚类算法中，进行坐标点的聚类。其中m_data为靶点的坐标。angle为每个靶点对应的测向线的角度信息。list_three为输出的坐标点聚类集合。numberruncount为测向线数量用于剔除独立的测向线聚类集合。

Parameters

| | |
|---|--|
| <i>m_data</i> 为vector<double>类型 | |
| <i>angle</i> 为vector<vector<double>>类型 | |
| <i>list_three</i> 为vector<vector<double>>类型 | |
| <i>numberruncount</i> 为int类型 | |

Returns

无

2.1.2.15 void sortarry (two_Tuple &list, int temp)

对一个二维的`vector<vector<double>>`的list进行按照每一个`vector<double>`的第 2 个元素进行升序

Parameters

| | |
|--|--|
| <i>list</i> 是 <code>vector<vector<double>></code> 类型 | |
| <i>temp</i> 为 <code>int</code> 类型 | |

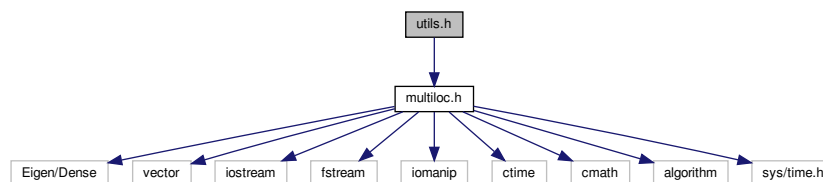
Returns

无返回

2.1.3 Variable Documentation**2.1.3.1 int numberruncount****2.2 utils.h File Reference**

```
#include <multiloc.h>
```

Include dependency graph for utils.h:

**Functions**

- void `print (one_Tuple &list)`
输出 1 维 `vector<double>` 的数据到命令行
- void `print (two_Tuple &list)`
输出 2 维 `vector<vector<double>>` 的数据到命令行
- void `print (three_Tuple &list)`
输出 3 维 `vector<vector<vector<double>>>` 的数据到命令行
- void `print (MatrixXd &list, int &total)`
输出 2 维 `MatrixXd` 前 `total` 个的数据到命令行
- bool `read_data (const char *filename, one_Tuple &tuples)`

: 从文件中读取数据内容, 将输出的结果保存在一维的 `vector<double>` 中. `filename` 为需要读取数据的文件名路径, `tuples` 保存读取的数据. 读取数据成功, 返回 `true`. 否则, 返回 `false`.

- `bool read_file (string &Docement, one_Tuple &m_data, two_Tuple &angle)`

从文件中读取用于多目标定位的靶点和测向线信息, 其中 `filename` 为需要读取数据的文件路径, 将读取的靶点坐标保存在 `m_data` 中, 将读取的测向线信息保存在 `angle` 中.

- `bool read_file (string &Docement, one_Tuple &m_data, one_Tuple &angle)`

从文件中读取用于单目标定位的靶点和测向线信息, 其中 `filename` 为需要读取数据的文件路径, 将读取的靶点坐标保存在 `m_data` 中, 将读取的测向线信息保存在 `angle` 中.

2.2.1 Function Documentation

2.2.1.1 void print (one_Tuple & list)

输出1维 `vector<double>` 的数据到命令行

Parameters

| | |
|--|--|
| <code>list</code> 为 <code>vector<double></code> 类型 | |
|--|--|

Returns

无

2.2.1.2 void print (two_Tuple & list)

输出2维 `vector<vector<double>>` 的数据到命令行

Parameters

| | |
|--|--|
| <code>list</code> 为 <code>vector<vector<double>></code> 类型 | |
|--|--|

Returns

无

2.2.1.3 void print (three_Tuple & list)

输出3维 `vector<vector<vector<double>>>` 的数据到命令行

Parameters

| | |
|--|--|
| <code>list</code> 为 <code>vector<vector<vector<double>>></code> 类型 | |
|--|--|

Returns

无

2.2.1.4 void print (MatrixXd & list, int & total)

输出2维MatrixXd前total个的数据到命令行

Parameters

| | |
|-------------------------|--|
| <i>list</i> 为MatrixXd类型 | |
| <i>total</i> 为int类型 | |

Returns

无

2.2.1.5 bool read_data (const char * filename, one_Tuple & tuples)

: 从文件中读取数据内容，将输出的结果保存在一维的vector<double>中。filename为需要读取数据的文件名路径，tuples保存读取的数据。读取数据成功，返回true。否则，返回false。

Parameters

| | |
|--|--|
| | |
|--|--|

2.2.1.6 bool read_file (string & Docement, one_Tuple & m_data, two_Tuple & angle)

从文件中读取用于多目标定位的靶点和测向线信息，其中filename为需要读取数据的文件路径，将读取的靶点坐标保存在m_data中，将读取的测向线信息保存在angle中。

Parameters

| | |
|--|--|
| <i>Docement</i> 为string类型 | |
| <i>m_data</i> 为vector<double>类型 | |
| <i>angle</i> 为vector<vector<double>>类型 | |

Returns

bool类型，读取数据成功，返回true。否则，返回false。

2.2.1.7 bool read_file (string & Docement, one_Tuple & m_data, one_Tuple & angle)

从文件中读取用于单目标定位的靶点和测向线信息，其中filename为需要读取数据的文件路径，将读取的靶点坐标保存在m_data中，将读取的测向线信息保存在angle中。

Parameters

| | |
|------------------------------------|-------------------|
| <i>Docement</i> 为 <i>string</i> 类型 | |
| <i>m_data</i> | 为vector<double>类型 |
| <i>angle</i> | 为vector<double>类型 |

Returns

bool类型，读取数据成功，返回true；否则，返回 false。

Index

classification
 multiloc.h, [5](#)

clearreplace
 multiloc.h, [5](#)

Demo1Single_Target
 multiloc.h, [6](#)

Demo2
 multiloc.h, [6](#)

Demo3
 multiloc.h, [6](#)

Demo3count
 multiloc.h, [7](#)

Demo3intersect
 multiloc.h, [7](#)

Demo3remove
 multiloc.h, [7](#)

Demo4Mult_Targetlocalization
 multiloc.h, [8](#)

Demo5inputdata
 multiloc.h, [8](#)

finditems
 multiloc.h, [8](#), [9](#)

isequal
 multiloc.h, [9](#)

multiloc.h, [3](#)
 classification, [5](#)
 clearreplace, [5](#)
 Demo1Single_Target, [6](#)
 Demo2, [6](#)
 Demo3, [6](#)
 Demo3count, [7](#)
 Demo3intersect, [7](#)
 Demo3remove, [7](#)
 Demo4Mult_Targetlocalization, [8](#)
 Demo5inputdata, [8](#)
 finditems, [8](#), [9](#)
 isequal, [9](#)
 numberruncount, [10](#)
 one_Tuple, [5](#)
 productiondot, [9](#)
 sortarry, [10](#)
 three_Tuple, [5](#)
 two_Tuple, [5](#)

numberruncount
 multiloc.h, [10](#)

one_Tuple
 multiloc.h, [5](#)

print
 utils.h, [11](#), [12](#)

productiondot
 multiloc.h, [9](#)

read_data
 utils.h, [12](#)

read_file
 utils.h, [12](#)

sortarry
 multiloc.h, [10](#)

three_Tuple
 multiloc.h, [5](#)

two_Tuple
 multiloc.h, [5](#)

utils.h, [10](#)
 print, [11](#), [12](#)
 read_data, [12](#)
 read_file, [12](#)