

My Project

Generated by Doxygen 1.8.13

Contents

1	Namespace Index	1
1.1	Namespace List	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Namespace Documentation	9
5.1	Ui Namespace Reference	9
6	Class Documentation	11
6.1	coordinate Struct Reference	11
6.1.1	Detailed Description	12
6.1.2	Member Data Documentation	12
6.1.2.1	x	12
6.1.2.2	y	12
6.1.2.3	z	12
6.2	dir_ratios Struct Reference	12
6.2.1	Detailed Description	13
6.2.2	Member Data Documentation	13
6.2.2.1	a	13

6.2.2.2	b	13
6.2.2.3	c	13
6.3	direction Struct Reference	14
6.3.1	Detailed Description	14
6.3.2	Member Data Documentation	14
6.3.2.1	theta_x	14
6.3.2.2	theta_y	15
6.3.2.3	theta_z	15
6.4	edge Struct Reference	15
6.4.1	Detailed Description	16
6.4.2	Member Data Documentation	17
6.4.2.1	node1	17
6.4.2.2	node2	17
6.5	graph Struct Reference	17
6.5.1	Detailed Description	18
6.5.2	Member Data Documentation	18
6.5.2.1	edge_corr	18
6.5.2.2	edges	18
6.5.2.3	nodes	18
6.6	MainWindow Class Reference	18
6.6.1	Constructor & Destructor Documentation	19
6.6.1.1	MainWindow()	19
6.6.1.2	~MainWindow()	19
6.7	node Struct Reference	20
6.7.1	Detailed Description	20
6.7.2	Member Data Documentation	20
6.7.2.1	adj_list	21
6.7.2.2	coord	21
6.8	pair_ Struct Reference	21
6.8.1	Detailed Description	21

6.8.2	Member Data Documentation	22
6.8.2.1	a	22
6.8.2.2	b	22
6.9	QMainWindow Class Reference	22
6.10	rot_matrix Struct Reference	23
6.10.1	Detailed Description	24
6.10.2	Member Data Documentation	24
6.10.2.1	Rx	24
6.10.2.2	Ry	24
6.10.2.3	Rz	24
6.11	threeView Struct Reference	24
6.11.1	Detailed Description	25
6.11.2	Member Data Documentation	25
6.11.2.1	e1	25
6.11.2.2	e2	25
6.11.2.3	e3	25
6.11.2.4	n1	25
6.11.2.5	n2	25
6.11.2.6	n3	25
7	File Documentation	27
7.1	draw.cpp File Reference	27
7.1.1	Function Documentation	28
7.1.1.1	draw_views()	28
7.1.1.2	draw_xy()	28
7.1.1.3	draw_xz()	29
7.1.1.4	draw_yz()	29
7.1.1.5	get_average()	29
7.1.1.6	get_max_x()	29
7.1.1.7	get_max_y()	29
7.1.1.8	get_max_z()	29

7.1.1.9	get_min_x()	29
7.1.1.10	get_min_y()	30
7.1.1.11	get_min_z()	30
7.1.1.12	get_three_views()	30
7.1.1.13	print_edges()	30
7.1.1.14	print_nodes()	30
7.2	input.cpp File Reference	31
7.2.1	Function Documentation	32
7.2.1.1	get_3D_graph()	32
7.2.1.2	get_mx4_matrix()	32
7.3	main.cpp File Reference	32
7.3.1	Function Documentation	33
7.3.1.1	main()	33
7.4	mainwindow.cpp File Reference	33
7.4.1	Function Documentation	34
7.4.1.1	add_coord()	34
7.4.1.2	translate_graph()	34
7.4.2	Variable Documentation	34
7.4.2.1	edge_codes_1	34
7.4.2.2	mode	34
7.4.2.3	node_0	34
7.4.2.4	node_1	34
7.5	mainwindow.h File Reference	35
7.6	struct.h File Reference	36
7.7	threeD_to_ortho.cpp File Reference	37
7.7.1	Macro Definition Documentation	38
7.7.1.1	PI	38
7.7.2	Function Documentation	38
7.7.2.1	find_ortho()	38
7.7.2.2	find_projection()	38
7.7.2.3	find_rot()	38
7.7.2.4	get_dir_ratios()	39
7.7.2.5	graph_to_mat()	39
7.7.2.6	mat_to_graph()	39
7.7.2.7	rot_about_coord_axis()	39
7.7.2.8	rotate_graph()	39
7.7.2.9	translate_graph()	40
7.8	twoD_to3D.cpp File Reference	40
7.8.1	Function Documentation	41
7.8.1.1	check_graph()	41
7.8.1.2	get_2D_graph()	41
7.8.1.3	get_pair_2D()	41

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Ui	9
----	---

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

coordinate	11
dir_ratios	12
direction	14
edge	15
graph	17
node	20
pair_	21
QMainWindow	22
MainWindow	18
rot_matrix	23
threeView	24

Chapter 3

Class Index

3.1 Class List

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

coordinate	11
dir_ratios	12
direction	14
edge	15
graph	17
MainWindow	18
node	20
pair_	21
QMainWindow	22
rot_matrix	23
threeView	24

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

draw.cpp	27
input.cpp	31
main.cpp	32
mainwindow.cpp	33
mainwindow.h	35
struct.h	36
threeD_to_ortho.cpp	37
twoD_to3D.cpp	40

Chapter 5

Namespace Documentation

5.1 Ui Namespace Reference

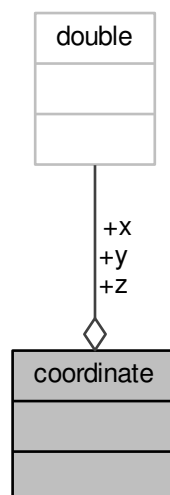
Chapter 6

Class Documentation

6.1 coordinate Struct Reference

```
#include <struct.h>
```

Collaboration diagram for coordinate:



Public Attributes

- double `x`
- double `y`
- double `z`

6.1.1 Detailed Description

A structure to represent 3d coordinate

6.1.2 Member Data Documentation

6.1.2.1 x

```
double coordinate::x
```

6.1.2.2 y

```
double coordinate::y
```

6.1.2.3 z

```
double coordinate::z
```

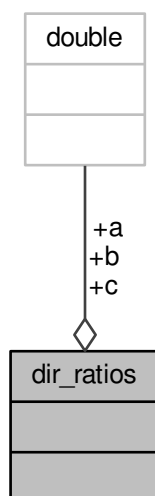
The documentation for this struct was generated from the following file:

- [struct.h](#)

6.2 dir_ratios Struct Reference

```
#include <struct.h>
```

Collaboration diagram for dir_ratios:



Public Attributes

- double [a](#)
- double [b](#)
- double [c](#)

6.2.1 Detailed Description

structure for storing direction ratios

6.2.2 Member Data Documentation

6.2.2.1 [a](#)

```
double dir_ratios::a
```

6.2.2.2 [b](#)

```
double dir_ratios::b
```

6.2.2.3 [c](#)

```
double dir_ratios::c
```

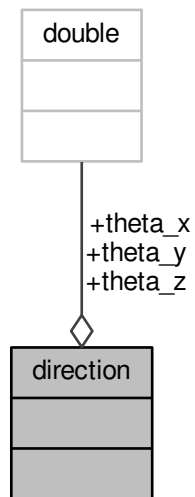
The documentation for this struct was generated from the following file:

- [struct.h](#)

6.3 direction Struct Reference

```
#include <struct.h>
```

Collaboration diagram for direction:



Public Attributes

- double [theta_x](#)
- double [theta_y](#)
- double [theta_z](#)

6.3.1 Detailed Description

structure for representing direction

6.3.2 Member Data Documentation

6.3.2.1 theta_x

```
double direction::theta_x
```

6.3.2.2 theta_y

```
double direction::theta_y
```

6.3.2.3 theta_z

```
double direction::theta_z
```

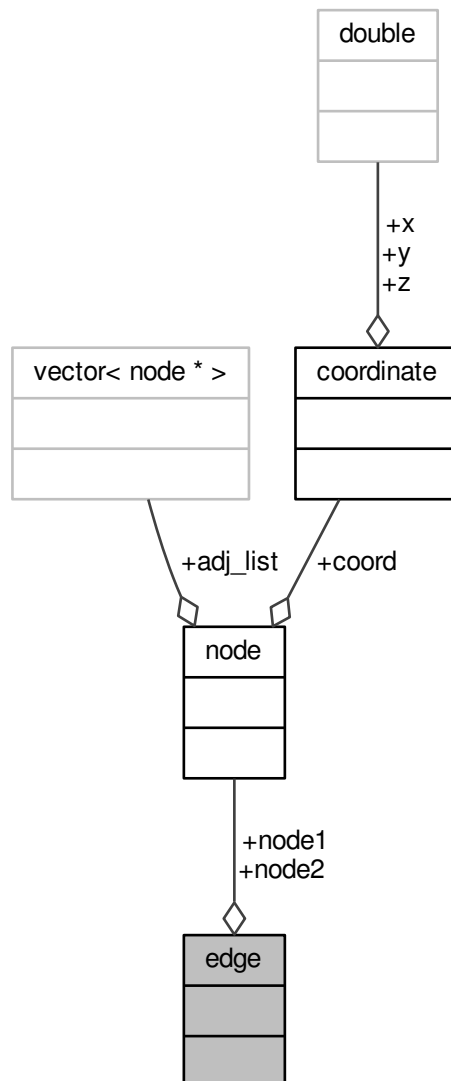
The documentation for this struct was generated from the following file:

- [struct.h](#)

6.4 edge Struct Reference

```
#include <struct.h>
```

Collaboration diagram for edge:



Public Attributes

- `node * node1`
- `node * node2`

6.4.1 Detailed Description

structure for edge in a graph with reference to nodes at its endpoints

6.4.2 Member Data Documentation

6.4.2.1 node1

`node*` `edge::node1`

6.4.2.2 node2

`node*` `edge::node2`

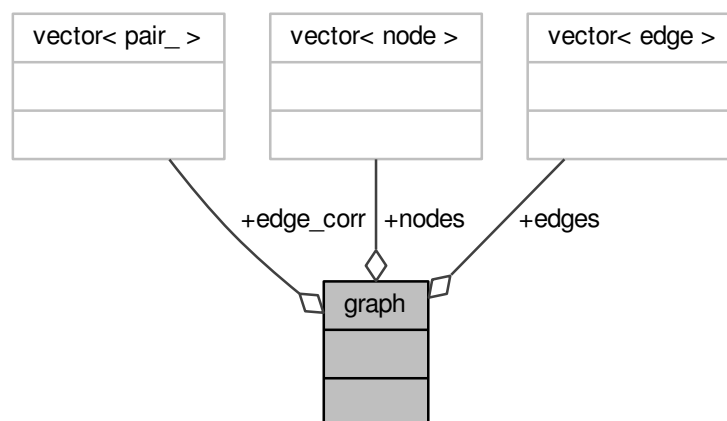
The documentation for this struct was generated from the following file:

- [struct.h](#)

6.5 graph Struct Reference

```
#include <struct.h>
```

Collaboration diagram for graph:



Public Attributes

- `vector< node >` `nodes`
- `vector< edge >` `edges`
- `vector< pair_ >` `edge_corr`

6.5.1 Detailed Description

structure for a graph with a vector of nodes, edges and edge correspondence which stores indexes of nodes which have a edge between them

6.5.2 Member Data Documentation

6.5.2.1 edge_corr

```
vector<pair_> graph::edge_corr
```

6.5.2.2 edges

```
vector<edge> graph::edges
```

6.5.2.3 nodes

```
vector<node> graph::nodes
```

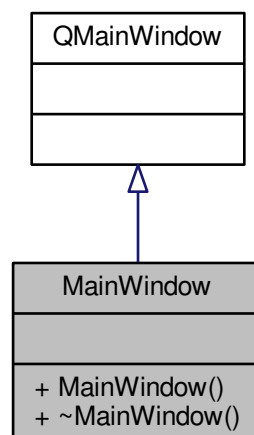
The documentation for this struct was generated from the following file:

- [struct.h](#)

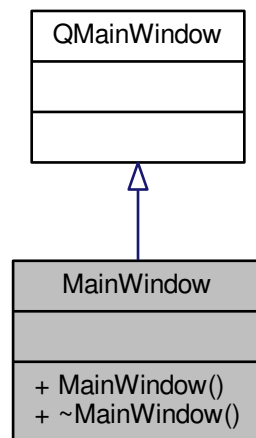
6.6 MainWindow Class Reference

```
#include <mainwindow.h>
```

Inheritance diagram for MainWindow:



Collaboration diagram for MainWindow:



Public Member Functions

- [MainWindow](#) (`QWidget *parent=0`)
- [~MainWindow](#) ()

6.6.1 Constructor & Destructor Documentation

6.6.1.1 MainWindow()

```
MainWindow::MainWindow (
    QWidget * parent = 0 ) [explicit]
```

6.6.1.2 ~MainWindow()

```
MainWindow::~MainWindow ( )
```

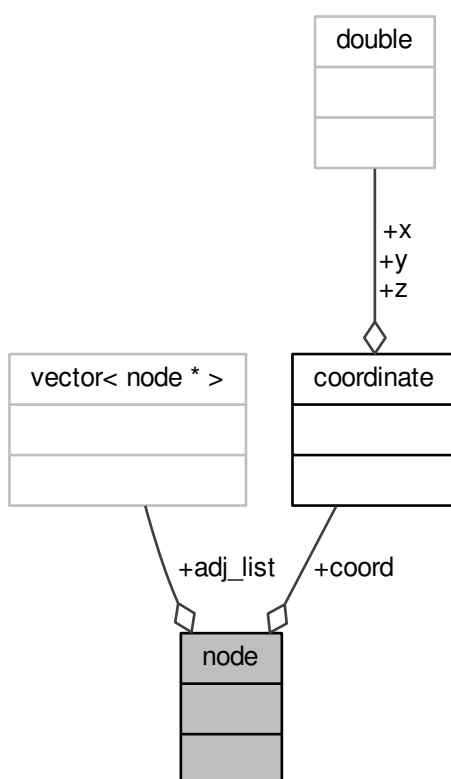
The documentation for this class was generated from the following files:

- [mainwindow.h](#)
- [mainwindow.cpp](#)

6.7 node Struct Reference

```
#include <struct.h>
```

Collaboration diagram for node:



Public Attributes

- `std::vector< node * > adj_list`
- `coordinate coord`

6.7.1 Detailed Description

structure for node of a graph which has coordinates as well as an adjacency list

6.7.2 Member Data Documentation

6.7.2.1 adj_list

```
std::vector<node *> node::adj_list
```

6.7.2.2 coord

```
coordinate node::coord
```

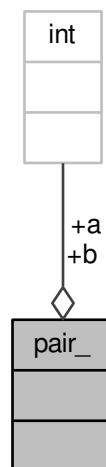
The documentation for this struct was generated from the following file:

- [struct.h](#)

6.8 pair_ Struct Reference

```
#include <struct.h>
```

Collaboration diagram for pair_:



Public Attributes

- int [a](#)
- int [b](#)

6.8.1 Detailed Description

structure used for a representing a pair of integers

6.8.2 Member Data Documentation

6.8.2.1 a

```
int pair_::a
```

6.8.2.2 b

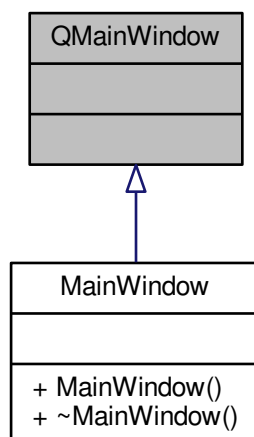
```
int pair_::b
```

The documentation for this struct was generated from the following file:

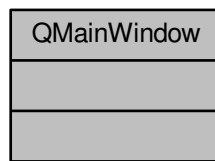
- [struct.h](#)

6.9 QMainWindow Class Reference

Inheritance diagram for QMainWindow:



Collaboration diagram for QMainWindow:



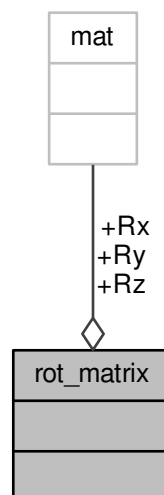
The documentation for this class was generated from the following file:

- [mainwindow.h](#)

6.10 rot_matrix Struct Reference

```
#include <struct.h>
```

Collaboration diagram for rot_matrix:



Public Attributes

- mat [Rx](#)
- mat [Ry](#)
- mat [Rz](#)

6.10.1 Detailed Description

structure for representing rotation matrix

6.10.2 Member Data Documentation

6.10.2.1 Rx

```
mat rot_matrix::Rx
```

6.10.2.2 Ry

```
mat rot_matrix::Ry
```

6.10.2.3 Rz

```
mat rot_matrix::Rz
```

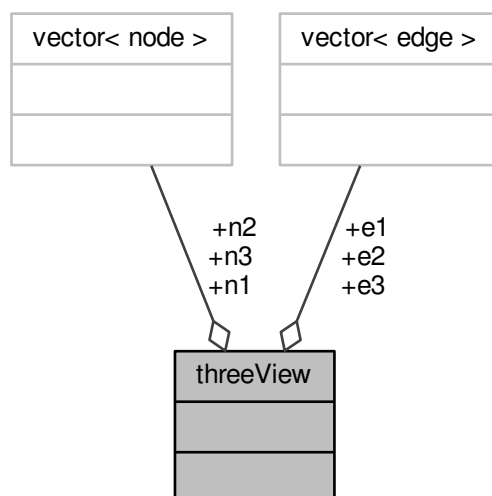
The documentation for this struct was generated from the following file:

- [struct.h](#)

6.11 threeView Struct Reference

```
#include <struct.h>
```

Collaboration diagram for threeView:



Public Attributes

- vector< [node](#) > n1
- vector< [edge](#) > e1
- vector< [node](#) > n2
- vector< [edge](#) > e2
- vector< [node](#) > n3
- vector< [edge](#) > e3

6.11.1 Detailed Description

structure for three orthographic views

6.11.2 Member Data Documentation

6.11.2.1 e1

```
vector<edge> threeView::e1
```

6.11.2.2 e2

```
vector<edge> threeView::e2
```

6.11.2.3 e3

```
vector<edge> threeView::e3
```

6.11.2.4 n1

```
vector<node> threeView::n1
```

6.11.2.5 n2

```
vector<node> threeView::n2
```

6.11.2.6 n3

```
vector<node> threeView::n3
```

The documentation for this struct was generated from the following file:

- [struct.h](#)

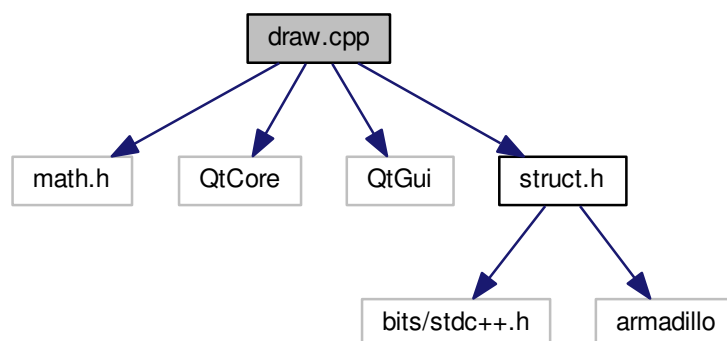
Chapter 7

File Documentation

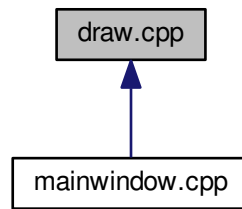
7.1 draw.cpp File Reference

```
#include <math.h>
#include <QtCore>
#include <QtGui>
#include "struct.h"
```

Include dependency graph for draw.cpp:



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



Functions

- `threeView get_three_views` (string filename)
- void `print_nodes` (vector< `node` > v)
- void `print_edges` (vector< `edge` > v)
- `coordinate get_average` (vector< `node` > v)
- int `get_max_y` (vector< `edge` > v)
- int `get_max_x` (vector< `edge` > v)
- int `get_min_x` (vector< `edge` > v)
- int `get_min_y` (vector< `edge` > v)
- int `get_min_z` (vector< `edge` > v)
- int `get_max_z` (vector< `edge` > v)
- `QPicture draw_views` (vector< `edge` > v)
- `QPicture draw_xy` (vector< `edge` > v)
- `QPicture draw_yz` (vector< `edge` > v)
- `QPicture draw_xz` (vector< `edge` > v)

7.1.1 Function Documentation

7.1.1.1 `draw_views()`

```
QPicture draw_views (
    vector< edge > v )
```

Takes a graph as an input. Renders view from the same graph.

7.1.1.2 `draw_xy()`

```
QPicture draw_xy (
    vector< edge > v )
```

Takes a graph description (vector of nodes) as and input. Draws the XY coordinates of this graph.

7.1.1.3 draw_xz()

```
QPicture draw_xz (
    vector< edge > v )
```

Takes a graph description (vector of nodes) as and input. Draws the XZ coordinates of this graph.

7.1.1.4 draw_yz()

```
QPicture draw_yz (
    vector< edge > v )
```

Takes a graph description (vector of nodes) as and input. Draws the YZ coordinates of this graph.

7.1.1.5 get_average()

```
coordinate get_average (
    vector< node > v )
```

Returns a coordinate average of the coordinates of all the nodes in a vector of nodes.

7.1.1.6 get_max_x()

```
int get_max_x (
    vector< edge > v )
```

Helper function that gets maximum x coordinate of a node in a vector of edges.

7.1.1.7 get_max_y()

```
int get_max_y (
    vector< edge > v )
```

Helper function that gets maximum y coordinate of a node in a vector of edges.

7.1.1.8 get_max_z()

```
int get_max_z (
    vector< edge > v )
```

Helper function that gets maximum z coordinate of a node in a vector of edges.

7.1.1.9 get_min_x()

```
int get_min_x (
    vector< edge > v )
```

Helper function that gets minimum x coordinate of a node in a vector of edges.

7.1.1.10 get_min_y()

```
int get_min_y (
    vector< edge > v )
```

Helper function that gets minimum y coordinate of a node in a vector of edges.

7.1.1.11 get_min_z()

```
int get_min_z (
    vector< edge > v )
```

Helper function that gets minimum z coordinate of a node in a vector of edges.

7.1.1.12 get_three_views()

```
threeView get_three_views (
    string filename )
```

Reads the 2D input from a file given as an argument. Converts the input into a graph and returns the graph of three orthographic views

7.1.1.13 print_edges()

```
void print_edges (
    vector< edge > v )
```

Helper function that prints coordinates of two nodes that make up an edge in a vector of edges.

7.1.1.14 print_nodes()

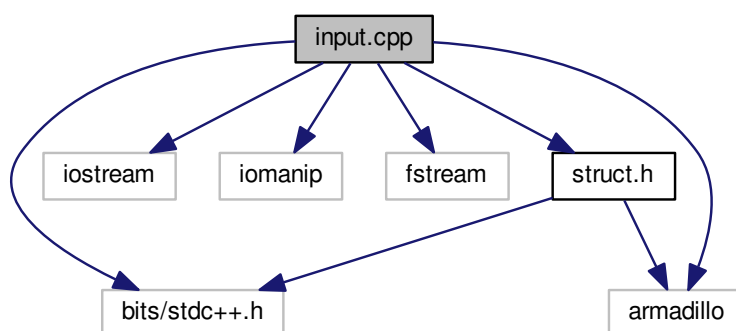
```
void print_nodes (
    vector< node > v )
```

Helper function that prints coordinates of a node in a vector of nodes.

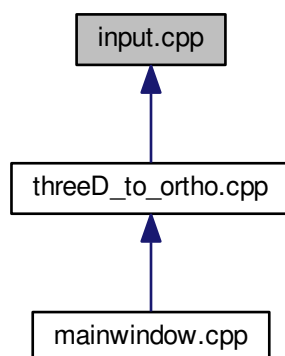
7.2 input.cpp File Reference

```
#include <bits/stdc++.h>
#include <iostream>
#include <iomanip>
#include <fstream>
#include <armadillo>
#include "struct.h"
```

Include dependency graph for input.cpp:



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



Functions

- [graph get_3D_graph](#) (string filename="input_3D.txt")
- [mat get_mx4_matrix](#) (vector< [node](#) > v, int cols=4)

7.2.1 Function Documentation

7.2.1.1 get_3D_graph()

```
graph get_3D_graph (
    string filename = "input_3D.txt" )
```

Reads the 3D input from a file given as an argument. Converts the input into a graph and returns the graph.

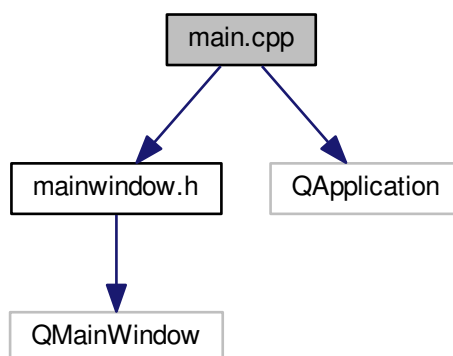
7.2.1.2 get_mx4_matrix()

```
mat get_mx4_matrix (
    vector< node > v,
    int cols = 4 )
```

Takes a graph as an input. Converts the graph into a 4X4 coordinate matrix as specified in the mathematical model.

7.3 main.cpp File Reference

```
#include "mainwindow.h"
#include <QApplication>
Include dependency graph for main.cpp:
```



Functions

- int `main` (int argc, char *argv[])

7.3.1 Function Documentation

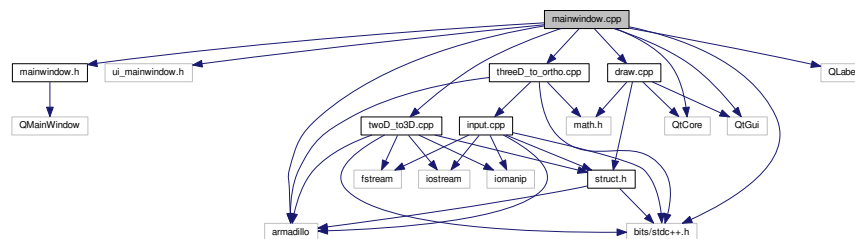
7.3.1.1 main()

```
int main (
    int argc,
    char * argv[] )
```

7.4 mainwindow.cpp File Reference

```
#include "mainwindow.h"
#include <ui_mainwindow.h>
#include <bits/stdc++.h>
#include <QtCore>
#include <QtGui>
#include <QLabel>
#include <armadillo>
#include "draw.cpp"
#include "threeD_to_ortho.cpp"
#include "twoD_to3D.cpp"
```

Include dependency graph for mainwindow.cpp:



Functions

- coordinate [add_coord](#) (coordinate c1, coordinate c2)
- void [translate_graph](#) (vector< [node](#) > &v, coordinate c)

Variables

- vector< [node](#) > [node_0](#)
- vector< [node](#) > [node_1](#)
- vector< [pair_](#) > [edge_codes_1](#)
- int [mode](#)

7.4.1 Function Documentation

7.4.1.1 add_coord()

```
coordinate add_coord (
    coordinate c1,
    coordinate c2 )
```

Helper func that adds two coordinates

7.4.1.2 translate_graph()

```
void translate_graph (
    vector< node > & v,
    coordinate c )
```

Translates graph taking a vector of nodes and an average coordinate

7.4.2 Variable Documentation

7.4.2.1 edge_codes_1

```
vector<pair_> edge_codes_1
```

7.4.2.2 mode

```
int mode
```

7.4.2.3 node_0

```
vector<node> node_0
```

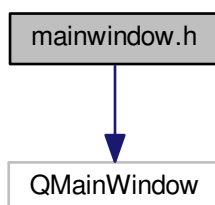
7.4.2.4 node_1

```
vector<node> node_1
```

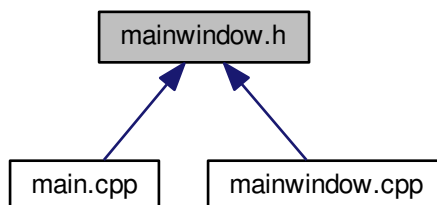
7.5 mainwindow.h File Reference

```
#include <QMainWindow>
```

Include dependency graph for mainwindow.h:



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



Classes

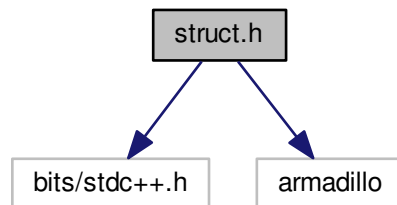
- class [MainWindow](#)

Namespaces

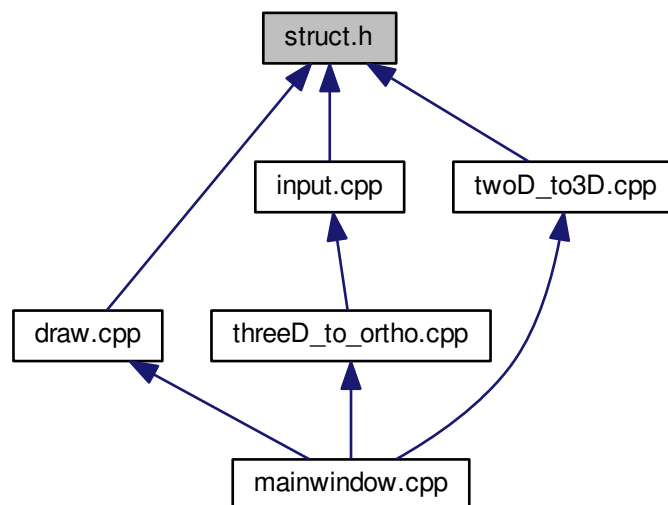
- [Ui](#)

7.6 struct.h File Reference

```
#include <bits/stdc++.h>
#include <armadillo>
Include dependency graph for struct.h:
```



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



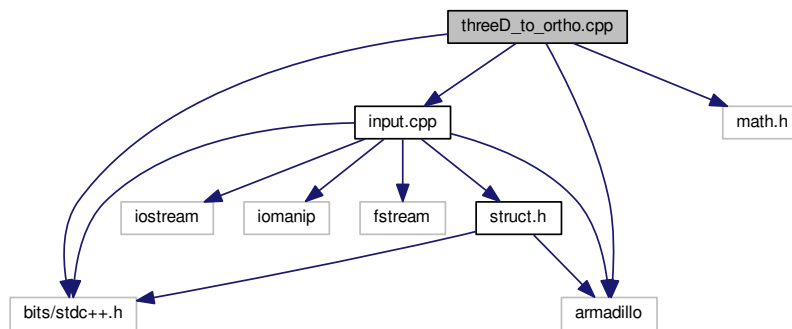
Classes

- struct [coordinate](#)
- struct [node](#)
- struct [edge](#)
- struct [pair_](#)
- struct [graph](#)
- struct [threeView](#)
- struct [direction](#)
- struct [rot_matrix](#)
- struct [dir_ratios](#)

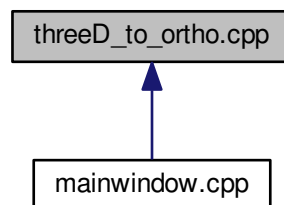
7.7 threeD_to_ortho.cpp File Reference

```
#include <bits/stdc++.h>
#include <armadillo>
#include "input.cpp"
#include <math.h>
```

Include dependency graph for threeD_to_ortho.cpp:



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



Macros

- `#define` [PI](#) 3.1415926536

Functions

- [dir_ratios](#) [get_dir_ratios](#) ()
- `mat` [graph_to_mat](#) (vector< [node](#) > nodes, int cols=4)
- vector< [node](#) > [mat_to_graph](#) (mat A, vector< [node](#) > vec)
- `mat` [translate_graph](#) (mat A, `coordinate` t_factor)
- `rot_matrix` [rot_about_coord_axis](#) (`direction` theta)
- `mat` [find_rot](#) (mat A, [dir_ratios](#) d)
- `mat` [find_projection](#) (mat A)
- vector< [node](#) > [find_ortho](#) (vector< [node](#) > graph)
- void [rotate_graph](#) (vector< [node](#) > &v, `direction` theta, int x, int y, int z)

7.7.1 Macro Definition Documentation

7.7.1.1 PI

```
#define PI 3.1415926536
```

7.7.2 Function Documentation

7.7.2.1 find_ortho()

```
vector<node> find_ortho (
    vector< node > graph )
```

Takes the graph whose projection is required to be taken as an argument. Then asks the user about the direction ratios of the desired line of sight. Then converts the graph to a matrix. Finds the transitions to be done. Applies the transitions to the coordinate matrix. Converts the matrix back to a graph and returns it.

7.7.2.2 find_projection()

```
mat find_projection (
    mat A )
```

Returns the projection of the graph after applying all the required transitions using z axis as the line-of-sight.

7.7.2.3 find_rot()

```
mat find_rot (
    mat A,
    dir_ratios d )
```

Takes two inputs

- The matrix A which has to be translated
- The direction ratios of the direction which has to become the line of sight for taking the projections. This function calls the function `rot_about_coord_axis` to get the rotation matrices and then multiplies the matrices with the coordinate matrix. It returns the coordinate matrix after multiplying it with the rotation matrices.

7.7.2.4 get_dir_ratios()

```
dir_ratios get_dir_ratios ( )
```

Asks the user for the direction along which it is desired to take the projection.

7.7.2.5 graph_to_mat()

```
mat graph_to_mat (
    vector< node > nodes,
    int cols = 4 )
```

Takes a graph as an input. Converts the graph into a 4X4 coordinate matrix as specified in the mathematical model.

7.7.2.6 mat_to_graph()

```
vector<node> mat_to_graph (
    mat A,
    vector< node > vec )
```

Takes a coordinate matrix as an input. Converts 4X4 coordinate matrix to the graph which would lead to construction of this matrix.

7.7.2.7 rot_about_coord_axis()

```
rot_matrix rot_about_coord_axis (
    direction theta )
```

Takes the direction ratios as the input. It returns the rotation matrices which are to be multiplied to the coordinate matrix so as to make the Z axis coincide with the given direction theta. It returns a struct of `rot_matrix` type which contains the three rotation matrices namely Rx, Ry and Rz.

7.7.2.8 rotate_graph()

```
void rotate_graph (
    vector< node > & v,
    direction theta,
    int x,
    int y,
    int z )
```

Rotate the given graph about axis (X, Y or Z axis) depending on the input x, y and z. Finds the rotation matrix for the given direction theta and applies only those whose corresponding input is 1. For Rx, x is to be 1. For Ry, y is to be 1. For Rz, z is to be 1.

7.7.2.9 translate_graph()

```
mat translate_graph (
    mat A,
    coordinate t_factor )
```

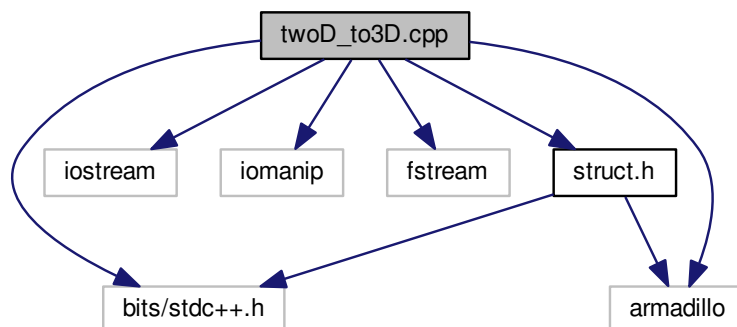
Takes two inputs

- The matrix A (for the corresponding graph) which has to be translated
- The coordinate t_factor of the point that would become the origin in the translated system. This function makes multiple calls to the function translate_coordinate for all the vertices of the input graph.

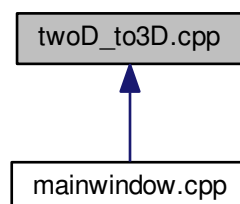
7.8 twoD_to3D.cpp File Reference

```
#include <bits/stdc++.h>
#include <iostream>
#include <iomanip>
#include <fstream>
#include <armadillo>
#include "struct.h"
```

Include dependency graph for twoD_to3D.cpp:



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



Functions

- vector< [node](#) > [get_2D_graph](#) (string filename)
- vector< [pair_](#) > [get_pair_2D](#) (string filename)
- void [check_graph](#) (vector< [node](#) > v)

7.8.1 Function Documentation

7.8.1.1 [check_graph\(\)](#)

```
void check_graph (
    vector< node > v )
```

7.8.1.2 [get_2D_graph\(\)](#)

```
vector<node> get_2D_graph (
    string filename )
```

Reads the 2D input from a file given as an argument. Converts the input into a graph and returns the graph.

7.8.1.3 [get_pair_2D\(\)](#)

```
vector<pair\_> get_pair_2D (
    string filename )
```

Reads the 2D input from a file given as an argument. Returns all pairs of vertices that have an edge between them according to the input in the file.

Index

~MainWindow
 MainWindow, [19](#)

a
 dir_ratios, [13](#)
 pair_, [22](#)
add_coord
 mainwindow.cpp, [34](#)
adj_list
 node, [20](#)

b
 dir_ratios, [13](#)
 pair_, [22](#)

c
 dir_ratios, [13](#)
check_graph
 twoD_to3D.cpp, [41](#)
coord
 node, [21](#)
coordinate, [11](#)
 x, [12](#)
 y, [12](#)
 z, [12](#)

dir_ratios, [12](#)
 a, [13](#)
 b, [13](#)
 c, [13](#)

direction, [14](#)
 theta_x, [14](#)
 theta_y, [14](#)
 theta_z, [15](#)

draw.cpp, [27](#)
 draw_views, [28](#)
 draw_xy, [28](#)
 draw_xz, [28](#)
 draw_yz, [29](#)
 get_average, [29](#)
 get_max_x, [29](#)
 get_max_y, [29](#)
 get_max_z, [29](#)
 get_min_x, [29](#)
 get_min_y, [29](#)
 get_min_z, [30](#)
 get_three_views, [30](#)
 print_edges, [30](#)
 print_nodes, [30](#)
draw_views

 draw.cpp, [28](#)
draw_xy
 draw.cpp, [28](#)
draw_xz
 draw.cpp, [28](#)
draw_yz
 draw.cpp, [29](#)

e1
 threeView, [25](#)

e2
 threeView, [25](#)

e3
 threeView, [25](#)

edge, [15](#)
 node1, [17](#)
 node2, [17](#)
edge_codes_1
 mainwindow.cpp, [34](#)

edge_corr
 graph, [18](#)

edges
 graph, [18](#)

find_ortho
 threeD_to_ortho.cpp, [38](#)

find_projection
 threeD_to_ortho.cpp, [38](#)

find_rot
 threeD_to_ortho.cpp, [38](#)

get_2D_graph
 twoD_to3D.cpp, [41](#)

get_3D_graph
 input.cpp, [32](#)

get_average
 draw.cpp, [29](#)

get_dir_ratios
 threeD_to_ortho.cpp, [38](#)

get_max_x
 draw.cpp, [29](#)

get_max_y
 draw.cpp, [29](#)

get_max_z
 draw.cpp, [29](#)

get_min_x
 draw.cpp, [29](#)

get_min_y
 draw.cpp, [29](#)

get_min_z

- draw.cpp, 30
- get_mx4_matrix
 - input.cpp, 32
- get_pair_2D
 - twoD_to3D.cpp, 41
- get_three_views
 - draw.cpp, 30
- graph, 17
 - edge_corr, 18
 - edges, 18
 - nodes, 18
- graph_to_mat
 - threeD_to_ortho.cpp, 39
- input.cpp, 31
 - get_3D_graph, 32
 - get_mx4_matrix, 32
- main
 - main.cpp, 33
- main.cpp, 32
 - main, 33
- MainWindow, 18
 - ~MainWindow, 19
 - MainWindow, 19
- mainwindow.cpp, 33
 - add_coord, 34
 - edge_codes_1, 34
 - mode, 34
 - node_0, 34
 - node_1, 34
 - translate_graph, 34
- mainwindow.h, 35
- mat_to_graph
 - threeD_to_ortho.cpp, 39
- mode
 - mainwindow.cpp, 34
- n1
 - threeView, 25
- n2
 - threeView, 25
- n3
 - threeView, 25
- node, 20
 - adj_list, 20
 - coord, 21
- node1
 - edge, 17
- node2
 - edge, 17
- node_0
 - mainwindow.cpp, 34
- node_1
 - mainwindow.cpp, 34
- nodes
 - graph, 18
- pair_, 21
 - a, 22
 - b, 22
- PI
 - threeD_to_ortho.cpp, 38
- print_edges
 - draw.cpp, 30
- print_nodes
 - draw.cpp, 30
- QMainWindow, 22
- rot_about_coord_axis
 - threeD_to_ortho.cpp, 39
- rot_matrix, 23
 - Rx, 24
 - Ry, 24
 - Rz, 24
- rotate_graph
 - threeD_to_ortho.cpp, 39
- Rx
 - rot_matrix, 24
- Ry
 - rot_matrix, 24
- Rz
 - rot_matrix, 24
- struct.h, 36
- theta_x
 - direction, 14
- theta_y
 - direction, 14
- theta_z
 - direction, 15
- threeD_to_ortho.cpp, 37
 - find_ortho, 38
 - find_projection, 38
 - find_rot, 38
 - get_dir_ratios, 38
 - graph_to_mat, 39
 - mat_to_graph, 39
 - PI, 38
 - rot_about_coord_axis, 39
 - rotate_graph, 39
 - translate_graph, 39
- threeView, 24
 - e1, 25
 - e2, 25
 - e3, 25
 - n1, 25
 - n2, 25
 - n3, 25
- translate_graph
 - mainwindow.cpp, 34
 - threeD_to_ortho.cpp, 39
- twoD_to3D.cpp, 40
 - check_graph, 41
 - get_2D_graph, 41
 - get_pair_2D, 41

U_i , [9](#)

x

coordinate, [12](#)

y

coordinate, [12](#)

z

coordinate, [12](#)