My Project

Generated by Doxygen 1.8.14

Contents

Chapter 1

CADapp

This short application was developed in C++ as an assignment for the course COP290 under Prof. Subhashis Banerjee during Spring 2017-18. This application provides a method to generate Orthographic projections of a polyhedral object from its 3D specification and to reconstruct a 3D object from its orthographic projections. The application checks for object validity and has a simple GUI to interact and input projection/object specification and generate and render the projection/3D object.

Libraries Used

- GLUT: OpenGL Utility Toolkit - To render the drawings
- GTK+ - To create the GUI
- stl: Standard Template Library - For *map* and *tuple* data structures

Authors

- **Ishan Tewari** - *Collaborator* - `IshanTewari`
- **Saksham Soni** - *Collaborator* - `sakshamsoni97`

See also the list of `contributors` who participated in this project.

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Class Documentation

4.1 edge Struct Reference

structure for an edge

```
#include <drawing.h>
```

Public Attributes

- vertex **v1**
- vertex **v2**
- bool visi

4.1.1 Detailed Description

structure for an edge

Edge is represented by tuple of vertices. It also stores the information of visibility of an edge when required

4.1.2 Member Data Documentation

4.1.2.1 visi

```
bool edge::visi
```

bool variable denoting the visibility

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

- drawing.h

4.2 edge2D Struct Reference

2D edge

```
#include <drawing.h>
```

Public Attributes

- vert2D **v1**
- vert2D **v2**

4.2.1 Detailed Description

2D edge

See also

edge

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

- drawing.h

4.3 face Struct Reference

structure for polygon face of a 3D object

```
#include <drawing.h>
```

Public Member Functions

- void compParam ()

Public Attributes

- float **A**
- float **B**
- float **C**
- float **D**
- map< string, edge > edges

4.3.1 Detailed Description

structure for polygon face of a 3D object

A polygon face is represented by a list of edges. The equation of plane of the face is represented by the A,B,C & D parameters

4.3.2 Member Function Documentation

4.3.2.1 compParam()

```
void face::compParam ( )
```

function to compute the equation of the plane from the list of edges.

4.3.3 Member Data Documentation

4.3.3.1 edges

```
map<string, edge> face::edges
```

list of edges

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

- drawing.h

4.4 HomeScreen Class Reference

Class for the home screen UI.

```
#include <GUI.h>
```

Public Member Functions

- ScreenPainter ()
- EventHandler ()
- ErrorHandler ()

4.4.1 Detailed Description

Class for the home screen UI.

Helps the user to select the kind of conversion to be performed (2D-3D or 3D-2D)

4.4.2 Member Function Documentation

4.4.2.1 ErrorHandler()

```
HomeScreen::ErrorHandler ( )
```

method to improve robustness and handle errors/incorrect calls with/or console output

4.4.2.2 EventHandler()

```
HomeScreen::EventHandler ( )
```

method to invoke correct behavior depending on event/action invoked by user

4.4.2.3 ScreenPainter()

```
HomeScreen::ScreenPainter ( )
```

method to display the UI elements composing the home screen

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

- GUI.h

4.5 inputScreen3d2d Class Reference

Class for the for 3D to 2D conversion input screen UI.

```
#include <GUI.h>
```

Public Member Functions

- ScreenPainter ()
- EventHandler ()
- ErrorHandler ()

4.5.1 Detailed Description

Class for the for 3D to 2D conversion input screen UI.

Helps the user to interact and specify the input specification of a 3D object and prepare its orthographic projection

4.5.2 Member Function Documentation

4.5.2.1 ErrorHandler()

```
inputScreen3d2d::ErrorHandler ( )
```

method to improve robustness and handle errors/incorrect calls with/or console output

4.5.2.2 EventHandler()

```
inputScreen3d2d::EventHandler ( )
```

method to invoke correct behavior depending on event/action invoked by user

4.5.2.3 ScreenPainter()

```
inputScreen3d2d::ScreenPainter ( )
```

method to display the UI elements composing the input screen

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

- GUI.h

4.6 Object3D Class Reference

Class for representing 3D objects.

```
#include <drawing.h>
```

Public Member Functions

- Projection projectTo2D (char ∗view)

  To compute Orthographic projections.
- void create (Projection FV, Projection TV, Projection SV, bool rightside=true, bool righthand=true)

  Initialize the 3D object using 3 Orthographic projections.
- void rotate (float alpha, float beta, float gamma)

  Rotation of the 3D object with respect to the given coordinate axes.
- void shift (float x0, float y0, float z0)

  shifting of origin of the 3D coordinate axes
- void display ()

  Method to render image of the object.

Public Attributes

- map< string, face > **flist**
- map< string, edge > **elist**
- map< string, vertex > **vlist**

Protected Member Functions

- tuple< map< string, edges >, map< string, vertex > > _wireframe (Projection FV, Projection TV, Projection SV, bool rightside=true, bool righthand=true)
- list< map< string, edge > > _planarGraph (map< string, edges > p_edges, map< string, vertex > p_↩ vertex)
- list< map< string, edge > > _hiddenEdge (list< map< string, edge >> planarGraphs)
- list< map< string, face > > _faceLoops (list< map< string, edge >> planarGraphs)
- list< map< string, face > > _bodyLoops (map< string, face > faceLoops)
- void _constructObject (list< map< string, face >>)
- static< map< string, edges >, map< string, vertex > > _PEVR (map< string, edges > edges, map< string, vertex > vertex)

  Protected Static Methods used to generate a 3D Object from its orthographic projections.
- static< map< string, edges >, map< string, vertex > > _RER (map< string, edges > edges, map< string, vertex > vertex)
- void _overlappingEdges (char ∗view)
- void _intersectingEdges (char ∗view)
- void _dashedLines (char ∗view)
- Projection _flatten (char ∗view)

4.6.1 Detailed Description

Class for representing 3D objects.

A 3D object is represented by a list of faces, edges and vertices.

4.6.2 Member Function Documentation

4.6.2.1 _bodyLoops()

```
list<map <string,face> > Object3D::_bodyLoops (
            map< string, face > faceLoops )  [protected]
```

Function that checks and creates possible body loops, from the set of face loops

Parameters

faceLoops	is the set of all possible face loops

Returns

List of all possible body loops

4.6.2.2 _constructObject()

```
void Object3D::_constructObject ( )  [protected]
```

Function that constructs a 3D object by combining body loop objects and also checks object validity

4.6.2.3 _dashedLines()

```
void Object3D::_dashedLines (
             char * view )  [protected]
```

Function to mark the hidden lines as dashed in the Orthographic projection

Parameters

view	a char∗ denoting the view of the projecion. It can take values - "front", "top", "rside", "lside"

4.6.2.4 _faceLoops()

```
list<map <string,face> > Object3D::_faceLoops (
             list< map< string, edge >> planarGraphs )  [protected]
```

Function that checks and creates possible face loops with valid normal direction, from the set of planar grpahs

Parameters

planarGraphs	is the pruned list of planar graphs after checking for hidden edge visibility

Returns

List of all possible face loops

4.6.2.5 _flatten()

```
Projection Object3D::_flatten (
             char * view )  [protected]
```

Function that generates the projection after processing all the edges for overlap and intersection.

Parameters

view	a char∗ denoting the view of the projecion. It can take values - "front", "top", "rside", "lside"

Returns

The corresponding orthographic view as Projection object

4.6.2.6 _hiddenEdge()

```
list<map <string,edge> > Object3D::_hiddenEdge (
            list< map< string, edge >> planarGraphs )  [protected]
```

Function that checks for visibilty of edges, as specified in views and eliminates, edges and planar graphs based on this

Parameters

planarGraphs	is the list of all possible planar graphs as obtained from the planar graph construction function

Returns

Pruned list of possible planar graphs

4.6.2.7 _intersectingEdges()

```
void Object3D::_intersectingEdges (
            char * view )  [protected]
```

Function to handle intersecting edges while generating Orthographic projection.

Parameters

view	a char∗ denoting the view of the projecion. It can take values - "front", "top", "rside", "lside"

4.6.2.8 _overlappingEdges()

```
void Object3D::_overlappingEdges (
            char * view )  [protected]
```

Function to handle overlapping edges while generating Orthographic projection.

Parameters

view	a char∗ denoting the view of the projecion. It can take values - "front", "top", "rside", "lside"

4.6.2.9 _PEVR()

```
static<map <string,edges>,map <string,vertex> > Object3D::_PEVR (
            map< string, edges > edges,
            map< string, vertex > vertex )  [protected]
```

Protected Static Methods used to generate a 3D Object from its orthographic projections.

Pathological Edge and Vertex Removal (PEVR) Method

Parameters

edges	is the set of all edges that have to be evaluated
vertex	is the set of all vertices that have to be evaluated

Returns

Tuple containing the set of vertices and edges after performing PEVR

4.6.2.10 _planarGraph()

```
list<map <string,edge> > Object3D::_planarGraph (
            map< string, edges > p_edges,
            map< string, vertex > p_vertex )  [protected]
```

Function that constructs Planar Graphs i.e. sets of valid coplanar edges

Parameters

p_edges	indicates the set of all possible edges constructed in the wireframe Function
p_vertex	indicates the set of vertices possible, as obtained from the wireframe reconstruction algorithm

Returns

List of all possible planar graphs

4.6.2.11 _RER()

```
static<map <string,edges>,map <string,vertex> > Object3D::_RER (
            map< string, edges > edges,
            map< string, vertex > vertex )  [protected]
```

Redundant Edge Removal (RER) Method

Parameters

edges	is the set of all edges that have to be evaluated
vertex	is the set of all vertices that have to be evaluated

Returns

Tuple containing the set of vertices and edges after performing RER

4.6.2.12 _wireframe()

```
tuple<map <string,edges>,map <string,vertex> > Object3D::_wireframe (
            Projection FV,
            Projection TV,
            Projection SV,
            bool rightside = true,
            bool righthand = true )  [protected]
```

Function that constructs the wireframe of the object i.e the edges outlning the 3D object

Parameters

FV	denotes the input front orthographic projection
TV	denotes the input top orthographic projection
SV	denoted the input side orthographic projection
rightside	boolean value indicating wether right side view is taken, default value is true
righthand	boolean value for right/left hand coordinate system to be followed, default value is true

Returns

Tuple containing list of possible edges and vertices

4.6.2.13 create()

```
void Object3D::create (
            Projection FV,
            Projection TV,
            Projection SV,
            bool rightside = true,
            bool righthand = true )
```

Initialize the 3D object using 3 Orthographic projections.

This function uses 3 Orthographic projection to reconstruct the 3D object from them and initialize itself

Parameters

FV	the front Orthographic projection
TV	the top Orthographic projection
SV	the side view Orthographic projection
rightside	boolean value telling which side view is taken, default true
righthand	boolean value for right/left hand coordinate system to be followed, default true

4.6.2.14 projectTo2D()

```
Projection Object3D::projectTo2D (
            char * view )
```

To compute Orthographic projections.

Function which computes and returns an Orthographic projection of the object

Parameters

view	a char∗ denoting the view of the projecion. It can take values - "front", "top", "rside", "lside"

Returns

Object of Class Projection

4.6.2.15 rotate()

```
void Object3D::rotate (
            float alpha,
            float beta,
            float gamma )
```

Rotation of the 3D object with respect to the given coordinate axes.

Parameters

alpha	Angular displacement about the x axis
beta	Angular displacement about the y axis
gamma	Angular displacement about the z axis

4.6.2.16 shift()

```
void Object3D::shift (
             float x0,
             float y0,
             float z0 )
```

shifting of origin of the 3D coordinate axes

Parameters

x0	Offset in x direction
y0	Offset in y direction
z0	Offset in z direction

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

- drawing.h

4.7 outputScreen3d2d Class Reference

Class for the for 3D to 2D conversion output screen UI.

```
#include <GUI.h>
```

Public Member Functions

- ScreenPainter ()
- EventHandler ()
- ErrorHandler ()

4.7.1 Detailed Description

Class for the for 3D to 2D conversion output screen UI.

Displays the corresponding orthographic projections for the 3D object specified in the input screen

4.7.2 Member Function Documentation

4.7.2.1 ErrorHandler()

```
outputScreen3d2d::ErrorHandler ( )
```

method to improve robustness and handle errors/incorrect calls with/or console output

4.7.2.2 EventHandler()

```
outputScreen3d2d::EventHandler ( )
```

method to invoke correct behavior depending on event/action invoked by user

4.7.2.3 ScreenPainter()

```
outputScreen3d2d::ScreenPainter ( )
```

method to display the UI elements composing the output screen

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

- GUI.h

4.8 Projection Class Reference

Class for representing projection.

```
#include <drawing.h>
```

Public Member Functions

- void display ()
- void getProjection ()

Public Attributes

- map< string, edge2D > **elist**
- map< string, vert2D > **vlist**

4.8.1 Detailed Description

Class for representing projection.

A projection is given by a list of 2D edges and a list of 2D vertices

4.8.2 Member Function Documentation

4.8.2.1 display()

```
void Projection::display ( )
```

function to display the projection in a window.

4.8.2.2 getProjection()

```
void Projection::getProjection ( )
```

Function which helps take projection specification as input

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

- drawing.h

4.9 vert2D Struct Reference

A 2D vertex.

```
#include <drawing.h>
```

Public Attributes

- float **x**
- float **y**

4.9.1 Detailed Description

A 2D vertex.

See also

vertex

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

- drawing.h

4.10 vertex Struct Reference

A 3D vertex.

```
#include <drawing.h>
```

Public Attributes

- float **x**
- float **y**
- float **z**

4.10.1 Detailed Description

A 3D vertex.

Structure for a point in 3 dimension.

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

- drawing.h

Chapter 5

File Documentation

5.1 cadapp.cpp File Reference

Main execution starting point.

```
#include <GTK/gtk.h>
#include "GUI.h"
```

Functions

- int **main** (int argc, char $**$argv)

5.1.1 Detailed Description

Main execution starting point.

The program execution starts here. It contains the main() function.

5.2 drawing.cpp File Reference

Implementation of the algorithms.

5.2.1 Detailed Description

Implementation of the algorithms.

This file contains the implementation of all the required algorithms for the CAD application. Libraries used:

- stl::list
- GLEW: OpenGL Extension Wrangler Library
- GLUT: OpenGL Utility Toolkit

5.3 drawing.h File Reference

Contains Projection and Object3D class definitions.

```
#include <map>
#include <tuple>
#include <string>
#include <GL/glew.h>
#include <GL/glut.h>
#include <gtk/gtk.h>
```

Classes

- struct vertex

  A 3D vertex.
- struct vert2D

  A 2D vertex.
- struct edge

  structure for an edge
- struct edge2D

  2D edge
- struct face

  structure for polygon face of a 3D object
- class Projection

  Class for representing projection.
- class Object3D

  Class for representing 3D objects.

5.3.1 Detailed Description

Contains Projection and Object3D class definitions.

This is the interface to the library developed for the CAD application which implements all the required algorithms. Libraries used:

- stl::list
- GLEW: OpenGL Extension Wrangler Library
- GLUT: OpenGL Utility Toolkit

OpenGL libraries are used to render orthographic views and the 3D object

5.4 GUI.cpp File Reference

Frontend of the application.

```
#include "drawing.h"
#include "GUI.h"
```

5.4.1 Detailed Description

Frontend of the application.

Front end and the UI of the application is handled here. UI is designed using GTK+ library. Provides an interface between the methods in drawing.h and the user. Libraries used:

- GTK+
- fstream
- stdlib

5.5 GUI.h File Reference

Header file for GUI functions definitions.

```
#include <fstream>
#include <stdlib.h>
#include <gtk/gtk.h>
```

Classes

- class HomeScreen

    Class for the home screen UI.
- class inputScreen3d2d

    Class for the for 3D to 2D conversion input screen UI.
- class outputScreen3d2d

    Class for the for 3D to 2D conversion output screen UI.

Functions

- class HomeScreen ScreenPainter ()

    Class for the for 2D to 3D conversion input screen UI.
- EventHandler ()

    Class for the for 2D to 3D conversion output screen UI.
- ErrorHandler ()

5.5.1 Detailed Description

Header file for GUI functions definitions.

This is the header file for the GUI part of the application. Libraries used:

- GTK+ GTK+ is chosen over other libraries such as Qt to develop the UI as it employs a widget based paradigm for UI development. This helps in conveying maximum information minimistically. It also ensures customizability for the user and increases the ease of interaction for the user.

5.5.2 Function Documentation

5.5.2.1 ErrorHandler()

```
ErrorHandler ( )
```

method to improve robustness and handle errors/incorrect calls with/or console output

5.5.2.2 EventHandler()

```
EventHandler ( )
```

Class for the for 2D to 3D conversion output screen UI.

method to invoke correct behavior depending on event/action invoked by user

Displays the corresponding 3D object for the orthographic views specified in the input screen

method to display the UI elements composing the output screen
method to invoke correct behavior depending on event/action invoked by user

5.5.2.3 ScreenPainter()

```
ScreenPainter ( )
```

Class for the for 2D to 3D conversion input screen UI.

Helps the user to interact and specify the input specification of orthographic views to convert to a 3D object

method to display the UI elements composing the input screen

method to display the UI elements composing the home screen

method to display the UI elements composing the input screen

Index