Temporal Anti-Aliasing C++/OpenGL Demo

Generated by Doxygen 1.8.16

Chapter 1

Masterclass

TAA implementation for Criterion masterclass

Based off of SimpleNGL and SimpleFBO demo by Jon Macey.

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Class Documentation

5.1 Camera Struct Reference

A struct to handle the camera parameters which are driven by the mouse and keyboard input.

```
#include <Camera.h>
```

Public Attributes

- QCursor **cursor**

  Qcursor instance (need it to reset the cursor to the center of the screen)
- float **cursorX**

  the screen space X coordinate of the cursor
- float **cursorY**

  the screen space X coordinate of the cursor
- bool **zoom** = false

  flag to indicate if the left mouse button is pressed when dragging
- int **zoomSize** = 250

  size (in pixels) of the square that zooms into our image on LMB click
- float **zoomScale** = 5

  zoomed dimensions
- glm::vec4 **zoomSquare**

  the bounding box of the zoom square
- bool **rotate** = false

  flag to indicate if the right mouse button is pressed when dragging
- float **translateF** = 0

  flag indicating if the W key was pressed
- float **translateB** = 0

  flag indicating if the S key was pressed
- float **translateL** = 0

  flag indicating if the A key was pressed
- float **translateR** = 0

  flag indicating if the D key was pressed
- float **pitch** = -45

  total pitch amount

- float **yaw** = -90

  total yaw amount
- glm::vec3 **pos**

  the current position of our camera
- glm::vec3 **front**

  the directional vector of where our camera is looking at
- glm::vec3 **up**

  where is up

5.1.1 Detailed Description

A struct to handle the camera parameters which are driven by the mouse and keyboard input.

Definition at line 15 of file Camera.h.

5.1.2 Member Data Documentation

5.1.2.1 cursor

```
QCursor Camera::cursor
```

Qcursor instance (need it to reset the cursor to the center of the screen)

Definition at line 20 of file Camera.h.

5.1.2.2 cursorX

```
float Camera::cursorX
```

the screen space X coordinate of the cursor

Definition at line 24 of file Camera.h.

5.1.2.3 cursorY

```
float Camera::cursorY
```

the screen space X coordinate of the cursor

Definition at line 28 of file Camera.h.

5.1.2.4 front

```
glm::vec3 Camera::front
```

the directional vector of where our camera is looking at

Definition at line 80 of file Camera.h.

5.1.2.5 pitch

```
float Camera::pitch = -45
```

total pitch amount

Definition at line 68 of file Camera.h.

5.1.2.6 pos

```
glm::vec3 Camera::pos
```

the current position of our camera

Definition at line 76 of file Camera.h.

5.1.2.7 rotate

```
bool Camera::rotate = false
```

flag to indicate if the right mouse button is pressed when dragging

Definition at line 48 of file Camera.h.

5.1.2.8 translateB

```
float Camera::translateB = 0
```

flag indicating if the S key was pressed

Definition at line 56 of file Camera.h.

5.1.2.9 translateF

```
float Camera::translateF = 0
```

flag indicating if the W key was pressed

Definition at line 52 of file Camera.h.

5.1.2.10 translateL

```
float Camera::translateL = 0
```

flag indicating if the A key was pressed

Definition at line 60 of file Camera.h.

5.1.2.11 translateR

```
float Camera::translateR = 0
```

flag indicating if the D key was pressed

Definition at line 64 of file Camera.h.

5.1.2.12 up

```
glm::vec3 Camera::up
```

where is up

Definition at line 84 of file Camera.h.

5.1.2.13 yaw

```
float Camera::yaw = -90
```

total yaw amount

Definition at line 72 of file Camera.h.

5.1.2.14 zoom

```
bool Camera::zoom = false
```

flag to indicate if the left mouse button is pressed when dragging

Definition at line 32 of file Camera.h.

5.1.2.15 zoomScale

```
float Camera::zoomScale = 5
```

zoomed dimensions

Definition at line 40 of file Camera.h.

5.1.2.16 zoomSize

```
int Camera::zoomSize = 250
```

size (in pixels) of the square that zooms into our image on LMB click

Definition at line 36 of file Camera.h.

5.1.2.17 zoomSquare

```
glm::vec4 Camera::zoomSquare
```

the bounding box of the zoom square

Definition at line 44 of file Camera.h.

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

- /home/emma/Documents/Masterclass/Demo/include/ **Camera.h**

5.2 Jitter Class Reference

A class to handle the generation and evolution of the scene's jitter.

```
#include <Jitter.h>
```

Public Member Functions

- **Jitter** ()

  ctor for the jitter class
- ∼**Jitter** ()

  dtor
- void **make2x** ()

  generates a translation matrix that jitters the current frame (2 subsamples)
- void **makeQuincunx** ()

  generates a translation matrix that jitters the current frame (5 subsamples in quincunx pattern)
- ngl::Mat4 **getMatrix** () const

  returns the jitter translation matrix
- ngl::Vec2 **getOffset** () const

  returns the jitter offset vector

Private Member Functions

- void **updateCycle** (int samples)

  updates the jitterCycle id - incrementing it and resetting when the maximum index is reached (2 or 5)

Private Attributes

- int **m_cycle** = 0

  an id which keeps track of the number of subsamples we are iterating through in the jitter pattern
- ngl::Mat4 **m_matrix**

  a matrix containing the translations for our current jitter
- ngl::Vec2 **m_offset**

  the translations for our current jitter

5.2.1 Detailed Description

A class to handle the generation and evolution of the scene's jitter.

Definition at line 15 of file Jitter.h.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 Jitter()

```
Jitter::Jitter ( )
```

ctor for the jitter class

Definition at line 4 of file Jitter.cpp.

5.2.2.2 ∼Jitter()

```
Jitter::∼Jitter ( )
```

dtor

Definition at line 10 of file Jitter.cpp.

5.2.3 Member Function Documentation

5.2.3.1 getMatrix()

```
ngl::Mat4 Jitter::getMatrix ( ) const  [inline]
```

returns the jitter translation matrix

Definition at line 37 of file Jitter.h.

5.2.3.2 getOffset()

```
ngl::Vec2 Jitter::getOffset ( ) const  [inline]
```

returns the jitter offset vector

Definition at line 41 of file Jitter.h.

5.2.3.3 make2x()

```
void Jitter::make2x ( )
```

generates a translation matrix that jitters the current frame (2 subsamples)

Definition at line 15 of file Jitter.cpp.

5.2.3.4 makeQuincunx()

```
void Jitter::makeQuincunx ( )
```

generates a translation matrix that jitters the current frame (5 subsamples in quincunx pattern)

Definition at line 30 of file Jitter.cpp.

5.2.3.5 updateCycle()

```
void Jitter::updateCycle (
             int samples )  [private]
```

updates the jitterCycle id - incrementing it and resetting when the maximum index is reached (2 or 5)

Definition at line 61 of file Jitter.cpp.

5.2.4 Member Data Documentation

5.2.4.1 m_cycle

```
int Jitter::m_cycle = 0  [private]
```

an id which keeps track of the number of subsamples we are iterating through in the jitter pattern

Definition at line 51 of file Jitter.h.

5.2.4.2 m_matrix

```
ngl::Mat4 Jitter::m_matrix  [private]
```

a matrix containing the translations for our current jitter

Definition at line 55 of file Jitter.h.

5.2.4.3 m_offset

```
ngl::Vec2 Jitter::m_offset  [private]
```

the translations for our current jitter

Definition at line 59 of file Jitter.h.

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

- /home/emma/Documents/Masterclass/Demo/include/ **Jitter.h**
- /home/emma/Documents/Masterclass/Demo/src/ **Jitter.cpp**

5.3 Scene Class Reference

This class inherits from the Qt OpenGLWindow and draws the scene with the TAA shader before passing the result to the screen.

`#include <Scene.h>`

Inheritance diagram for Scene:

class_scene-eps-converted-to.pdf

Public Member Functions

- **Scene** ()
  ctor for our NGL drawing class
- ∼**Scene** () override
  dtor must close down ngl and release OpenGL resources
- void **initializeGL** () override
  the initialize class is called once when the window is created and we have a valid GL context use this to setup any default GL stuff
- void **paintGL** () override
  this is called everytime we want to draw the scene

Private Member Functions

- void **loadMatricesToShader** (const char ∗shaderName, ngl::Mat4 _M)
  method to load transform matrices to the shader, originally by Jon Macey, modified
- void **loadTexture** ()
  method to load a texture from file, code by Jon Macey
- void **createTextureObjects** ()
  creates the texture objects for the FBOs
- void **createFramebufferObject** ()
  creates the FBOs
- void **drawSceneGeometry** (ngl::VAOPrimitives ∗_prim)
  draw the objects in the scene
- void **drawScreenOrientedPlane** (GLuint _pid, ngl::VAOPrimitives ∗_prim)
  draw a screen-oriented plane
- void **renderText** ()
  draw text over the screen (glitches the screen every now and then, so not currently used)
- void **keyPressEvent** (QKeyEvent ∗_event) override
  Qt Event called when a key is pressed.
- void **keyReleaseEvent** (QKeyEvent ∗_event) override
  Qt Event called when a key is released.
- void **mouseMoveEvent** (QMouseEvent ∗_event) override
  this method is called every time a mouse is moved
- void **mousePressEvent** (QMouseEvent ∗_event) override
  this method is called everytime the mouse button is pressed inherited from QObject and overridden here.
- void **mouseReleaseEvent** (QMouseEvent ∗_event) override
  this method is called everytime the mouse button is released inherited from QObject and overridden here.

Private Attributes

- QTime **m_time**

    a QTime object used for tracking the framerate
- int **m_deltaTime**

    elapsed time since the last frame
- int **m_framerate**

    the current framerate
- bool **m_start** = true

    a flag indicating whether this is the first frame
- bool **m_TAA** = true

    TAA subroutine flag.
- bool **m_subNbrClamp** = true

    Neighborhood clamping subroutine flag.
- bool **m_subReproject** = true

    Reprojection subroutine flag.
- **Camera** **m_cam**

    the virtual camera of the scene
- **Jitter** **m_jitter**

    the jitter struct of the scene
- ngl::Mat4 **m_view**

    the view matrix for camera
- ngl::Mat4 **m_viewPrev**

    the view matrix for camera of the previous frame
- ngl::Mat4 **m_projection**

    the projection matrix for camera
- bool **m_transformLight** =false
- ngl::Vec4 **m_lightPos**
- std::unique_ptr< ngl::Obj > **m_meshStreet**

    street model
- std::unique_ptr< ngl::Obj > **m_meshGround**

    street ground model
- std::unique_ptr< ngl::Obj > **m_meshWalls**

    street large walls model
- std::unique_ptr< ngl::Obj > **m_meshFence**

    fence model (error on load, so not currently used)
- GLuint **m_brickTexture**

    id for the scene's brick texture
- GLuint **m_fboTexId** [3]

    ids for the color attachments used by the FBOs. (two for the history ping pongs, one for the current frame, and one for the current frame with MSAA)
- GLuint **m_fboDepthId**

    ids for the depth attachments used by the current frame fbo
- int **m_ping**

    an index that alternates between 0 and 1 to help ping pong texture locations and fbos
- GLuint **m_fboId** [3]

    our FBO IDs used by the FBOs

5.3.1 Detailed Description

This class inherits from the Qt OpenGLWindow and draws the scene with the TAA shader before passing the result to the screen.

Definition at line 19 of file Scene.h.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 Scene()

```
Scene::Scene ( )
```

ctor for our NGL drawing class

Parameters

in	*parent*	the parent window to the class

Definition at line 14 of file Scene.cpp.

5.3.2.2 ∼Scene()

```
Scene::∼Scene ( )  [override]
```

dtor must close down ngl and release OpenGL resources

Definition at line 20 of file Scene.cpp.

5.3.3 Member Function Documentation

5.3.3.1 createFramebufferObject()

```
void Scene::createFramebufferObject ( )  [private]
```

creates the FBOs

Definition at line 334 of file Scene.cpp.

5.3.3.2 createTextureObjects()

```
void Scene::createTextureObjects ( )  [private]
```

creates the texture objects for the FBOs

Definition at line 259 of file Scene.cpp.

5.3.3.3 drawSceneGeometry()

```
void Scene::drawSceneGeometry (
             ngl::VAOPrimitives * _prim )  [private]
```

draw the objects in the scene

Parameters

in	*passes*	an instance of the NGL primitives to draw primitive geometry from the NGL library

Definition at line 189 of file Scene.cpp.

5.3.3.4 drawScreenOrientedPlane()

```
void Scene::drawScreenOrientedPlane (
             GLuint _pid,
             ngl::VAOPrimitives * _prim )  [private]
```

draw a screen-oriented plane

Parameters

in	*passes*	an instance of the shader handler as well as NGL primitives

Definition at line 249 of file Scene.cpp.

5.3.3.5 initializeGL()

```
void Scene::initializeGL ( )  [override]
```

the initialize class is called once when the window is created and we have a valid GL context use this to setup any default GL stuff

Definition at line 34 of file Scene.cpp.

5.3.3.6 keyPressEvent()

```
void Scene::keyPressEvent (
            QKeyEvent * _event )  [override], [private]
```

Qt Event called when a key is pressed.

Parameters

in	*_event*	the Qt event to query for size etc

Definition at line 100 of file MouseKeyEvents.cpp.

5.3.3.7 keyReleaseEvent()

```
void Scene::keyReleaseEvent (
            QKeyEvent * _event )  [override], [private]
```

Qt Event called when a key is released.

Parameters

in	*_event*	the Qt event to query for size etc

Definition at line 179 of file MouseKeyEvents.cpp.

5.3.3.8 loadMatricesToShader()

```
void Scene::loadMatricesToShader (
            const char * shaderName,
            ngl::Mat4 _M )  [private]
```

method to load transform matrices to the shader, originally by Jon Macey, modified

Parameters

in	*takes*	in the name of the shader shaderName to draw the object with as well as its model matrix _M

Definition at line 139 of file Scene.cpp.

5.3.3.9 loadTexture()

```
void Scene::loadTexture ( )  [private]
```

method to load a texture from file, code by Jon Macey

Definition at line 167 of file Scene.cpp.

5.3.3.10 mouseMoveEvent()

```
void Scene::mouseMoveEvent (
            QMouseEvent * _event )  [override], [private]
```

this method is called every time a mouse is moved

Parameters

_event	the Qt Event structure

Definition at line 9 of file MouseKeyEvents.cpp.

5.3.3.11 mousePressEvent()

```
void Scene::mousePressEvent (
            QMouseEvent * _event )  [override], [private]
```

this method is called everytime the mouse button is pressed inherited from QObject and overridden here.

Parameters

_event	the Qt Event structure

Definition at line 63 of file MouseKeyEvents.cpp.

5.3.3.12 mouseReleaseEvent()

```
void Scene::mouseReleaseEvent (
            QMouseEvent * _event )  [override], [private]
```

this method is called everytime the mouse button is released inherited from QObject and overridden here.

Parameters

_event	the Qt Event structure

Definition at line 84 of file MouseKeyEvents.cpp.

5.3.3.13 paintGL()

```
void Scene::paintGL ( )  [override]
```

this is called everytime we want to draw the scene

Definition at line 381 of file Scene.cpp.

5.3.3.14 renderText()

```
void Scene::renderText ( )  [private]
```

draw text over the screen (glitches the screen every now and then, so not currently used)

Definition at line 372 of file Scene.cpp.

5.3.4 Member Data Documentation

5.3.4.1 m_brickTexture

```
GLuint Scene::m_brickTexture  [private]
```

id for the scene's brick texture

Definition at line 121 of file Scene.h.

5.3.4.2 m_cam

```
Camera Scene::m_cam  [private]
```

the virtual camera of the scene

Definition at line 74 of file Scene.h.

5.3.4.3 m_deltaTime

```
int Scene::m_deltaTime  [private]
```

elapsed time since the last frame

Definition at line 50 of file Scene.h.

5.3.4.4 m_fboDepthId

```
GLuint Scene::m_fboDepthId  [private]
```

ids for the depth attachments used by the current frame fbo

Definition at line 130 of file Scene.h.

5.3.4.5 m_fboId

```
GLuint Scene::m_fboId[3]  [private]
```

our FBO IDs used by the FBOs

Definition at line 138 of file Scene.h.

5.3.4.6 m_fboTexId

```
GLuint Scene::m_fboTexId[3]  [private]
```

ids for the color attachments used by the FBOs. (two for the history ping pongs, one for the current frame, and one for the current frame with MSAA)

Definition at line 126 of file Scene.h.

5.3.4.7 m_framerate

```
int Scene::m_framerate  [private]
```

the current framerate

Definition at line 54 of file Scene.h.

5.3.4.8 m_jitter

```
Jitter Scene::m_jitter  [private]
```

the jitter struct of the scene

Definition at line 78 of file Scene.h.

5.3.4.9 m_lightPos

```
ngl::Vec4 Scene::m_lightPos  [private]
```

Definition at line 92 of file Scene.h.

5.3.4.10 m_meshFence

```
std::unique_ptr<ngl::Obj> Scene::m_meshFence  [private]
```

fence model (error on load, so not currently used)

Definition at line 117 of file Scene.h.

5.3.4.11 m_meshGround

```
std::unique_ptr<ngl::Obj> Scene::m_meshGround  [private]
```

street ground model

Definition at line 109 of file Scene.h.

5.3.4.12 m_meshStreet

```
std::unique_ptr<ngl::Obj> Scene::m_meshStreet  [private]
```

street model

Definition at line 105 of file Scene.h.

5.3.4.13 m_meshWalls

```
std::unique_ptr<ngl::Obj> Scene::m_meshWalls  [private]
```

street large walls model

Definition at line 113 of file Scene.h.

5.3.4.14 m_ping

```
int Scene::m_ping  [private]
```

an index that alternates between 0 and 1 to help ping pong texture locations and fbos

Definition at line 134 of file Scene.h.

5.3.4.15 m_projection

```
ngl::Mat4 Scene::m_projection  [private]
```

the projection matrix for camera

Definition at line 90 of file Scene.h.

5.3.4.16 m_start

```
bool Scene::m_start = true  [private]
```

a flag indicating whether this is the first frame

Definition at line 58 of file Scene.h.

5.3.4.17 m_subNbrClamp

```
bool Scene::m_subNbrClamp = true  [private]
```

Neighborhood clamping subroutine flag.

Definition at line 66 of file Scene.h.

5.3.4.18 m_subReproject

```
bool Scene::m_subReproject = true  [private]
```

Reprojection subroutine flag.

Definition at line 70 of file Scene.h.

5.3.4.19 m_TAA

```
bool Scene::m_TAA = true  [private]
```

TAA subroutine flag.

Definition at line 62 of file Scene.h.

5.3.4.20 m_time

```
QTime Scene::m_time  [private]
```

a QTime object used for tracking the framerate

Definition at line 46 of file Scene.h.

5.3.4.21 m_transformLight

```
bool Scene::m_transformLight =false  [private]
```

Definition at line 91 of file Scene.h.

5.3.4.22 m_view

```
ngl::Mat4 Scene::m_view  [private]
```

the view matrix for camera

Definition at line 82 of file Scene.h.

5.3.4.23 m_viewPrev

```
ngl::Mat4 Scene::m_viewPrev  [private]
```

the view matrix for camera of the previous frame

Definition at line 86 of file Scene.h.

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

- /home/emma/Documents/Masterclass/Demo/include/ **Scene.h**
- /home/emma/Documents/Masterclass/Demo/src/ **MouseKeyEvents.cpp**
- /home/emma/Documents/Masterclass/Demo/src/ **Scene.cpp**

Chapter 6

File Documentation

6.1 /home/emma/Documents/Masterclass/Demo/include/Camera.h File Reference

```
#include <ngl/NGLInit.h>
#include <QTime>
#include <QCursor>
#include <ngl/Mat4.h>
```

Classes

- struct **Camera**

  A struct to handle the camera parameters which are driven by the mouse and keyboard input.

Variables

- constexpr float **SENSITIVITY** = 0.1f

  multiplier for the camera rotation with mouse movement
- constexpr float **SPEED** = 0.01f

  multiplier for the camera translation with key presses

6.1.1 Detailed Description

Author

Emma Koo

6.1.2 Variable Documentation

6.1.2.1 SENSITIVITY

```
constexpr float SENSITIVITY = 0.1f  [constexpr]
```

multiplier for the camera rotation with mouse movement

Definition at line 90 of file Camera.h.

6.1.2.2 SPEED

```
constexpr float SPEED = 0.01f  [constexpr]
```

multiplier for the camera translation with key presses

Definition at line 94 of file Camera.h.

6.2 /home/emma/Documents/Masterclass/Demo/include/Jitter.h File Reference

```
#include <ngl/NGLInit.h>
#include <ngl/Vec2.h>
#include <ngl/Vec4.h>
#include <ngl/Mat4.h>
```

Classes

- class **Jitter**
  A class to handle the generation and evolution of the scene's jitter.

6.2.1 Detailed Description

Author

Emma Koo

6.3 /home/emma/Documents/Masterclass/Demo/include/Scene.h File Reference

```
#include <ngl/Text.h>
#include <ngl/Mat4.h>
#include <ngl/ShaderLib.h>
#include <ngl/Obj.h>
#include "Camera.h"
#include "Jitter.h"
#include <QOpenGLWindow>
```

Classes

- class **Scene**

  This class inherits from the Qt OpenGLWindow and draws the scene with the TAA shader before passing the result to the screen.

Variables

- constexpr int **TEXTURE_WIDTH** =1024

  the width of the window
- constexpr int **TEXTURE_HEIGHT** =720

  the height of the window

6.3.1 Variable Documentation

6.3.1.1 TEXTURE_HEIGHT

```
constexpr int TEXTURE_HEIGHT =720  [constexpr]
```

the height of the window

Definition at line 197 of file Scene.h.

6.3.1.2 TEXTURE_WIDTH

```
constexpr int TEXTURE_WIDTH =1024  [constexpr]
```

the width of the window

Definition at line 193 of file Scene.h.

6.4 /home/emma/Documents/Masterclass/Demo/README.md File Reference

6.5 /home/emma/Documents/Masterclass/Demo/src/Jitter.cpp File Reference

```
#include "Jitter.h"
#include "Scene.h"
```

6.6 /home/emma/Documents/Masterclass/Demo/src/main.cpp File Reference

```
#include "Scene.h"
#include <QtGui/QGuiApplication>
#include <iostream>
```

Functions

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

6.6.1 Function Documentation

6.6.1.1 main()

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

Definition at line 9 of file main.cpp.

6.7 /home/emma/Documents/Masterclass/Demo/src/MouseKeyEvents.cpp File Reference

```
#include "Camera.h"
#include "Scene.h"
#include <QMouseEvent>
#include <QGuiApplication>
#include <ngl/Mat4.h>
#include <iostream>
```

6.8 /home/emma/Documents/Masterclass/Demo/src/Scene.cpp File Reference

```
#include "Scene.h"
#include "GLFW/glfw3.h"
#include "glm/ext.hpp"
#include <glm/gtc/type_ptr.hpp>
#include <ngl/NGLInit.h>
#include <ngl/NGLStream.h>
#include <ngl/ShaderLib.h>
#include <ngl/VAOPrimitives.h>
#include <ngl/Obj.h>
#include <QMouseEvent>
#include <QGLWidget>
#include <QPainter>
```