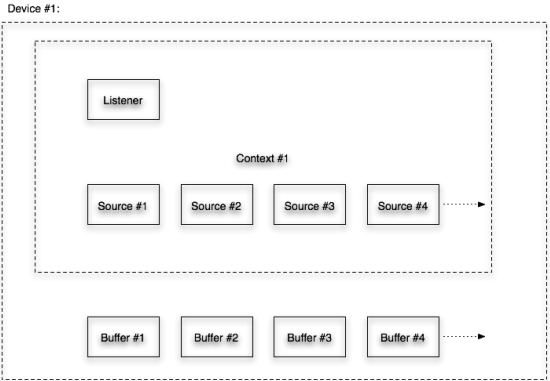
OpenAl学习笔记



Openal的基本组成就是上图所描述的，包括一个设备device，设备里面包含至少一个context，和若干个缓冲，而context里面只包含一个listener，和若干个源。

源（source）是指向放声音的空间。明白源是非常  
的重要。源只播放内存中的背景声音数据。源也给出了特殊的属性如位置和速度。

使用openal的第一步就是包含头文件了：  
   #include <al/al.h>  
   #include <al/alc.h>  
   #include <al/alu.h>  
   #include <al/alut.h>

使用过opengl的肯定对al的头文件感觉熟悉，其实al就是模仿gl写的。

关于设备的创建:

ALCdevice \*alcOpenDevice(

const ALCchar \*devicename

);

1，上面是创建的函数，返回一个指向设备的指针，如果返回NULL，那肯定是出了问题。

2，传入值是一个ALCchar\*，其实就是一个字符串，在alc.h头文件中有这样的代码：typedef char ALCchar;说明ALCchar就是char。那传入什么值才可以？你可以传入一个现有的设备的名字，或者是一个NULL，NULL代表新建一个默认设备。

插曲：如何知道当前可用的设备，那就使用glcGetString函数

const ALCchar \* alcGetString(

ALCdevice \*device,

ALenum param

);

Parameters

device a pointer to the device to be queried

param an attribute to be retrieved:

ALC\_DEFAULT\_DEVICE\_SPECIFIER

ALC\_CAPTURE\_DEFAULT\_DEVICE\_SPECIFIER

ALC\_DEVICE\_SPECIFIER

ALC\_CAPTURE\_DEVICE\_SPECIFIER

ALC\_EXTENSIONS

传给该函数ALC\_DEVICE\_SPECIFIER就可以返回一个指针，指向的是一个字符串(设备名).

3，判断是否出错，使用alGetError

该函数返回当前错误状态，并清除该项纪录。

if (alGetError() != AL\_NO\_ERROR)

{

exit(0);

}

4, 如果设备成功创建了，则下一步创建设备context

ALCcontext\* context = 0;

context = alcCreateContext(device,NULL);

alcMakeContextCurrent(context);

5,创建缓冲区buffer的索引

ALCuint bufferIndex = 0;

获得一个随机索引值: alGenBuffers(1,&bufferIndex);

可以将1改为任意数值，但是bufferIndex就必须是数组了，用于存放多个缓冲索引值。这个值用于表示每个缓冲，就和人的名字一样，不同的是他们不能重复，否则代表的就是同一个缓冲。

6,下面要做的就是给缓冲索引处填充内容了，就是音频数据，可以利用现有的alut.h里的函数加载wav格式音频，等熟练了可以自己练习mp3等格式了。

Alut.h有一个函数：alutLoadWAVFile.就是导入wav格式文件的函数，它需要很多参数，内部处理音频文件后，传入的参数都被赋值成相应意义的值。

ALenum format;

void\* data;

ALsizei size;

ALsizei frequency;

ALboolean loop;

alutLoadWAVFile("BEEP3.WAV",&format,&data,&size,&frequency,&loop);

可以看出参数就是音频的格式、数据内容的指针、尺寸、频率和循环次数。

7，将bufferIndex索引所代表的缓冲区填充数据：alBufferData(bufferIndex,format,data,size,frequency);

第一个参数就是缓冲索引值，可能你有好几个缓冲索引，那么根据需要填充相应的索引值就可以了。

第二个到底最后一个参数就是上面得到的一些数据，直接传进去。

8，填充完毕，记得调用alutUnloadWAV(format,data,size,frequency);释放一下内存。

9, 第九步了，就是处理源，源就是声音的位置，源可以改变位置

ALCuint sourceIndex = 0;

// 得到源的索引值

alGenSources(1,& sourceIndex);

// 将源和缓冲关联起来

alSourcei(sourceIndex,AL\_BUFFER,bufferIndex);

同样，alGenSources第一个参数比一定非得是1。

10，设置源的一些信息

// 设置源的信息

ALCfloat sourcePos[] = {2,0,2};

ALCfloat sourceVel[] = {1,1,1};

alSourcefv(sourceIndex,AL\_POSITION,sourcePos);

alSourcefv(sourceIndex,AL\_VELOCITY,sourceVel);

alSourcei(sourceIndex,AL\_LOOPING,AL\_TRUE);

alSourcef(sourceIndex,AL\_GAIN,1);

alSourcef(sourceIndex,AL\_PITCH,1);

坐标和opengl坐标只有z轴不同，opengl的z值正的向屏幕外，而openal是朝里。

11,设置听者的信息，如果你把听者位置设置的不同，听到声音就不同。

// 设置听者的信息

ALCfloat listenerPos[] = {0,0,-1};

ALCfloat listenerVel[] = {0,0,0};

ALCfloat listenerOri[] = {0,0,-1,0,1,0};

alListenerfv(AL\_POSITION,listenerPos);

alListenerfv(AL\_VELOCITY,listenerVel);

alListenerfv(AL\_ORIENTATION,listenerOri);

12,接下来就是播放声音了，很简单：alSourcePlay(sourceIndex);

如果你有多个源，可以调用多次，这样就可以在同一时刻听到不同声音。

比如我现在还有一个sourceTwoIndex源，关联了一个bufferTwoIndex，那么可以接着调用alSourcePlay(sourceTwoIndex);

这样就有两个声音在播放了。

13，注意在你播放声音的时候，不要马上退出程序，如果你刚让程序播放，就退出，就基本听不到声音了，openal不会耐心等声音播放完毕再友好的退出！所以加上下面的代码：

while (choose != 'q')

{

choose = getchar();

switch(choose)

{

case 'w':

sourcePos[0] -= sourceVel[0];

sourcePos[1] -= sourceVel[1];

sourcePos[2] -= sourceVel[2];

alSourcefv(sourceIndex,AL\_POSITION,sourcePos);

if (alGetError() != AL\_NO\_ERROR)

{

// 出错

exit(0);

}

alSourcei(sourceIndex, AL\_SOURCE\_RELATIVE, AL\_FALSE);

break;

case 's':

sourcePos[0] += sourceVel[0];

sourcePos[1] += sourceVel[1];

sourcePos[2] += sourceVel[2];

alSourcefv(sourceIndex,AL\_POSITION,sourcePos);

if (alGetError() != AL\_NO\_ERROR)

{

// 出错

exit(0);

}

alSourcei(sourceIndex, AL\_SOURCE\_RELATIVE, AL\_FALSE);

break;

}

}

就可以通过按键控制程序的继续运行，按下’w’，声音源位置就会离我们越来越近，因为他的位置z变小了。

按’s’相反效果。

另外，如果测试修改源位置没有效果，可能是你的音频不是单声道的，你可以转换成一个单声道音频，这样就可以感觉到声音位置的改变了。

附上两个声源一起播放的代码(可以结合上面的步骤，那样就不会被这么多的步骤弄晕了)：

#include <al/al.h>

#include <al/alc.h>

#include <al/alut.h>

#include <windows.h>

#include <iostream>

using namespace std;

void main()

{

// 设备

ALCdevice\* device = 0;

// 上下文

ALCcontext\* context = 0;

// 缓冲

ALCuint bufferIndex = 0;

ALCuint bufferTwoIndex = 0;

// 源

ALCuint sourceIndex = 0;

ALCuint sourceTwoIndex = 0;

// 创建设备

device = alcOpenDevice(NULL);

if (alGetError() != AL\_NO\_ERROR)

{

// 出错

exit(0);

}

else

{

// 未出错，创建并选择当前上下文

context = alcCreateContext(device,NULL);

alcMakeContextCurrent(context);

}

// 创建缓冲

alGenBuffers(1,&bufferIndex);

alGenBuffers(1,&bufferTwoIndex);

if (alGetError() != AL\_NO\_ERROR)

{

exit(0);

}

ALenum format;

void\* data;

ALsizei size;

ALsizei frequency;

ALboolean loop;

alutLoadWAVFile("BEEP3.WAV",&format,&data,&size,&frequency,&loop);

alBufferData(bufferIndex,format,data,size,frequency);

if (alGetError() != AL\_NO\_ERROR)

{

exit(0);

}

alutUnloadWAV(format,data,size,frequency);

// 源

alGenSources(1,&sourceIndex);

// 第二个源

alGenSources(1,&sourceTwoIndex);

alSourcei(sourceIndex,AL\_BUFFER,bufferIndex);

if (alGetError() != AL\_NO\_ERROR)

{

exit(0);

}

alutLoadWAVFile("EXP0.WAV",&format,&data,&size,&frequency,&loop);

alBufferData(bufferTwoIndex,format,data,size,frequency);

if (alGetError() != AL\_NO\_ERROR)

{

exit(0);

}

alutUnloadWAV(format,data,size,frequency);

alSourcei(sourceTwoIndex,AL\_BUFFER,bufferTwoIndex);

// 设置源的信息

ALCfloat sourcePos[] = {2,0,2};

ALCfloat sourceVel[] = {1,1,1};

alSourcefv(sourceIndex,AL\_POSITION,sourcePos);

alSourcefv(sourceIndex,AL\_VELOCITY,sourceVel);

alSourcei(sourceIndex,AL\_LOOPING,AL\_TRUE);

alSourcef(sourceIndex,AL\_GAIN,1);

alSourcef(sourceIndex,AL\_PITCH,1);

// 第二个源的位置

ALCfloat sourceTwoPos[] = {-2,0,-2};

ALCfloat sourceTwoVel[] = {1,1,1};

alSourcefv(sourceTwoIndex,AL\_POSITION,sourceTwoPos);

alSourcefv(sourceTwoIndex,AL\_VELOCITY,sourceTwoVel);

alSourcei(sourceTwoIndex,AL\_LOOPING,AL\_TRUE);

alSourcef(sourceTwoIndex,AL\_GAIN,1);

alSourcef(sourceTwoIndex,AL\_PITCH,1);

// 设置听者的信息

ALCfloat listenerPos[] = {0,0,-1};

ALCfloat listenerVel[] = {0,0,0};

ALCfloat listenerOri[] = {0,0,-1,0,1,0};

alListenerfv(AL\_POSITION,listenerPos);

alListenerfv(AL\_VELOCITY,listenerVel);

alListenerfv(AL\_ORIENTATION,listenerOri);

char choose = '0';

alSourcePlay(sourceIndex);

alSourcePlay(sourceTwoIndex);

while (choose != 'q')

{

choose = getchar();

switch(choose)

{

case 'w':

sourcePos[0] -= sourceVel[0];

sourcePos[1] -= sourceVel[1];

sourcePos[2] -= sourceVel[2];

alSourcefv(sourceIndex,AL\_POSITION,sourcePos);

if (alGetError() != AL\_NO\_ERROR)

{

// 出错

exit(0);

}

alSourcei(sourceIndex, AL\_SOURCE\_RELATIVE, AL\_FALSE);

break;

case 's':

sourcePos[0] += sourceVel[0];

sourcePos[1] += sourceVel[1];

sourcePos[2] += sourceVel[2];

alSourcefv(sourceIndex,AL\_POSITION,sourcePos);

if (alGetError() != AL\_NO\_ERROR)

{

// 出错

exit(0);

}

alSourcei(sourceIndex, AL\_SOURCE\_RELATIVE, AL\_FALSE);

break;

}

}

// 安全退出

context = alcGetCurrentContext();

device = alcGetContextsDevice(context);

alcMakeContextCurrent(NULL);

alcDestroyContext(context);

alcCloseDevice(device);

}

我也是粗略了解一下，还有很多效果没有试验，如果有问题可以一起讨论。

邮箱:chengcong0812@126.com