OpenAL - Das akustische Gegenstück zu OpenGL

Christoph Bichlmeier

22 May 2007

chair for computer aided medical procedures

department of computer science | technische universität münchen

Allgemeines

- OpenAL ist Plattformunabhängig (DirectSound & DirectSound3D gibt es dagegen nur für Windows)
- Wiedergeben & Manipulation von Sounds, 3D Sound
- Grundlegende Objekte für ein OpenAL Device:
  - Viele Buffer -- enthält Audiodaten, z.B. aus einer Datei
  - Viele Sources – Position, Orientierung der Geräuschquelle und Geschwindigkeit für Simulation der Schallgeschwindigkeit (Medium und Distanz sind OpenAL unbekannt).
  - Einen Listener – Position, Orientierung des Geräuschempfängers normalerweise äquivalent zu den Kameraparametern.
- Dokumentation: http://www.openal.org/documentation.html
- Homepage: http://www.openal.org/

Wie bekommt man OpenAL zum Laufen

- **SDK Installer** (für Windows) und **OpenAL Utility Toolkit (ALUT)** (entspricht GLUT und macht das Leben leichter) von http://www.openal.org/downloads.html
- Wie bereits bekannt einfach .h, .dll und .lib einbinden.

```
#include <stdlib.h>
#include <alut.h>

int main (int argc, char **argv){
  ALuint helloBuffer, helloSource;
  alutInit (&argc, argv);
  // erzeugt einen audio buffer mit Inhalt „hello world"
  helloBuffer = alutCreateBufferHelloWorld ();
  // eine neue Soundquelle erzeugen
  alGenSources (1, &helloSource);
  //audiodaten im buffer einer Soundquelle zuweisen
  alSourcei (helloSource, AL_BUFFER, helloBuffer);
  //das ganze abspielen
  alSourcePlay (helloSource);
  //Warte 4 Sekunden bis das Programm terminiert
  alutSleep (4);
  //die alut methode für exit();
  alutExit ();
  return 0;
}
```

OpenAL: Lade eine Audiodatei .wav

```
int main (int argc, char **argv)
{
  ALuint helloSource, bufferqueue[1];
  alutInit (&argc, argv);

  alGenBuffers(2, bufferqueue);

  ALenum format;
  ALsizei size;
  ALfloat freq;
  ALvoid* data;
  ALboolean loop;
  data = alutLoadMemoryFromFile((ALbyte*)"../data/test.wav", &format, &size, &freq);
  alBufferData(bufferqueue[0], format, data, size, freq);
  alutUnloadWAV(format, data, size, freq);
```

OpenAL: Definiere die Soundquelle

```
alGenSources (1, &helloSource);
alSourceQueueBuffers(helloSource, 1,bufferqueue);
float soundposition[] = {0,0,-2};
float sounddirection[] = {0,1,0};
alSourcefv(helloSource, AL_POSITION, soundposition);
alSourcefv(helloSource, AL_DIRECTION, sounddirection);
alSourcef (helloSource, AL_PITCH, 1.0f );
alSourcef (helloSource, AL_GAIN, 3.0f );
alSourcei (helloSource, AL_LOOPING, AL_TRUE);
alDistanceModel(AL_INVERSE_DISTANCE);
```

OpenAL: Definiere die Soundempfänger (Listener) & Aufräumen

```
  float listenerposition[] = {0,0,0};
  // At & Up Vektor
  float listenerorientation[] = {0.0,0.0,-1.0, 0.0,1.0,0.0};
  alListenerfv(AL_POSITION, listenerposition);
  alListenerfv(AL_ORIENTATION, listenerorientation);
  alSourcePlay (helloSource);
  alutSleep (4); // warte 4 Sekunden um Sound abzuspielen

  alDeleteSources(1,&helloSource);//delete our source
  alDeleteBuffers(1,&bufferqueue);//delete our buffer

  alutExit ();
  return 0;
}
```

Neben alSourcePlay () gibt es auch alSourceStop(), alSourcePause(), alSourceRewind();