Лекция №11

Средства воспроизведения звука
Основы OpenAL

Рябинин Константин Валентинович

e-mail: icosaeder@ya.ru
jabber: icosaeder@jabber.ru

Пермь, 2011

Средства воспроизведения звука

Средства воспроизведения звука – важная составляющая полноценной мультимедийной системы

- Звук, по природе своей, создаётся аналоговым сигналом, поэтому в контексте ЭВМ необходимо иметь средства для модуляции такого сигнала на основе цифровых описаний
- Как правило, мультимедийные системы позволяют лишь воспроизводить звук, ранее записанный при помощи аналоговой аппаратуры (микрофонов)
- Однако существуют средства генерации звука:
  - На основе «сэмплов» – предварительно записанных звуковых «примитивов»
  - На основе физически-точного моделирования музыкальных инструментов

Средства воспроизведения звука

Для воспроизведения звука существует целый ряд низкоуровневых средств, наиболее популярными из которых являются

- **DirectSound / DirectMusic – средства воспроизведения звука в составе DirectX**
- **OpenAL – открытая спецификация звуковой библиотеки, «собрат» OpenGL**

→ **Существуют кроссплатформенные реализации OpenAL; библиотеки этой спецификации принято использовать в мультимедийных системах, графика в которых основана на OpenGL**

OpenAL – это открытый интерфейс к звуковому оборудованию для работы с аудиоданными

Так же, как и OpenGL, OpenAL

- Предполагает процедурный характер работы
- Включает в себя только базовые функции управления звуком, то есть сочетает низкий уровень и универсальность
- Взаимодействует с аудиооборудованием напрямую
- Поддерживает систему нотации функций и типов (используя префиксы al / alc и суффикс типа параметров для функций и префикс AL для типов)
- Использует схожую терминологию. Так, например, процесс воспроизведения звука также называется рендерингом
- Использует машину состояний
- Обладает вспомогательной библиотекой ALUT, упрощающей процесс написания программ, использующих OpenAL в качестве подсистемы воспроизведения звука

OpenAL

В отличии от OpenGL, OpenAL

- **Предоставляет два API:**
  - **Ядро, включающее в себя вызовы функций OpenAL**
  - **ALC (Audio Library Context) – API, используемый для управления контекстом рендеринга, контролем использования ресурсов и задействования блокировок в мультипоточных вычислениях**
- **Не имеет в явном виде концепции конвейера**

Основные понятия OpenAL

Основными логическими компонентами в программе, использующей библиотеку стандарта OpenAL, являются

- **Единственный слушатель (*Listener*) – точка, из которой пользователь слышит звук; тесно связан со звуковоспроизводящим устройством**
- **Множество буферов (*Buffers*) – области памяти, содержащие несжатые аудиоданные**
- **Множество источников звука (*Sources*) – логические объекты, располагающиеся в трёхмерном пространстве и воспроизводящие звуки из буферов**

Основные понятия OpenAL

Слушатель содержит

- Скорость перемещения (может быть использована для моделирования эффекта Доплера)
- Позицию в пространстве
- Направление
- Показатель усиления звука

Буферы содержат

- Аудиоданные в формате PCM (*Pulse Code Modulation*, импульсно-кодовая модуляция – последовательность мгновенных значений аналогового сигнала, измеренных в равные промежутки времени и закодированных двоичными числами). Буферы поддерживают разрядность 8 и 16 бит, а так же могут иметь одну или две звуквых дорожки (моно и стерео)

Источники включают в себя

- Указатель на буфер
- Скорость перемещения
- Позицию в пространстве
- Направление
- Интенсивность звука

Основные понятия OpenAL

Функция рендеринга

- производит необходимые вычисления, основанные на параметрах источников и слушателя:
  - Моделирование затухания звука по мере удаления от источника
  - Эффект Доплера
  - . . .

Структура программы с OpenAL

Программа, использующая OpenAl во многом схожа с программой, использующей OpenGL. Основные этапы:

- **Инициализация**
  - **Открытие устройства**
    ```
    m_pDevice = alcOpenDevice(NULL);
    ```
  - **Создание контекста воспроизведения**
    ```
    m_pContext = alcCreateContext(m_pDevice, NULL);
    ```
  - **Активация контекста воспроизведения**
    ```
    alcMakeContextCurrent(m_pContext);
    ```
- **Установка параметров слушателя**
  - **Позиция**
    ```
    float listenerPos[] = { 0.0, 0.0, 0.0 };
    alListenerfv(AL_POSITION, listenerPos);
    ```
  - **Скорость**
    ```
    float listenerVel[] = { 0.0, 0.0, 0.0 };
    alListenerfv(AL_VELOCITY, listenerVel);
    ```
  - **Ориентация**
    ```
    float listenerOri[] = { 0.0, 0.0, -1.0, 0.0, 1.0, 0.0 };
    alListenerfv(AL_ORIENTATION, listenerOri);
    ```

Структура программы с OpenAL

- Загрузка данных
  - Генерация буфера
    ```
    alGenBuffers(1, &m_bufferID);
    ```
  - Загрузка данных в буфер
    ```
    alutLoadWAVFile((ALbyte*)fileName, &format, &data,
                    &size, &freq, &looping);
    alBufferData(m_bufferID, format, data, size, freq);
    alutUnloadWAV(format, data, size, freq);
    ```
- Подготовка источника
  - Создание источника
    ```
    alGenSources(1, &m_sourceID);
    ```
  - Связывание источника с буфером
    ```
    alSourcei(m_sourceID, AL_BUFFER, m_bufferID);
    ```
  - Установка параметров источника
    - Тон звука
      ```
      alSourcef(m_sourceID, AL_PITCH, 1.0);
      ```
    - Усиление звука (изменение громкости с изменением расстояния)
      ```
      alSourcef(m_sourceID, AL_GAIN, 1.0);
      ```

Структура программы с OpenAL

- **Положение источника в пространстве**
  `alSourcefv(m_sourceID, AL_POSITION, scrPos);`
- **Скорость перемещения источника**
  `alSourcefv(m_sourceID, AL_VELOCITY, scrVel);`
- **Флаг цикличности воспроизведения**
  `alSourcei(m_sourceID, AL_LOOPING, scrShouldLoop);`

- **Воспроизведение**
  `alSourcePlay(m_sourceID);`
- **Остановка**
  `alSourceStop(m_sourceID);`
- **Удаление структур**
  - **Удаление источника**
    `alDeleteSources(1, &m_sourceID);`
  - **Удаление буфера**
    `alDeleteBuffers(1, &m_bufferID);`
- **Деинициализация**
  - **Сброс контекста**
    `alcMakeContextCurrent(NULL);`
  - **Удаление контекста**
    `alcDestroyContext(m_pContext);`
  - **Закрытие устройства**
    `alcCloseDevice(m_pDevice);`

Звуки и музыка

При работе с аудиоданными, принято разграничивать два понятия: звуки и музыку

Звук (звуковой эффект) – короткая аудиозапись, занимающая в несжатом виде сравнительно немного места. Как правило, представлен одной дорожкой (моно-звучание)

Музыка (музыкальная композиция) – длинная аудиозапись, занимающая в несжатом виде очень много места. Как правило, представлена двумя и более дорожками (стерео-звучание)

- **Звуки приемлемо хранить в несжатом виде и использовать для их воспроизведения единственный буфер**
- **Для хранения музыки необходимо использовать сжатие, а воспроизведение осуществлять при помощи механизма подкачки данных в буфер (оставляя размер буфера небольшим)**

Форматы хранения аудиоданных

Несжатые аудиоданные (формат хранения – PCM):

- **WAV**

Сжатые аудиоданные:

- **Сжатие с потерями**
  - **Сжатые данные хатактеризуются битрейтом – плотностью информационного потока, т. е. объёмом информации в единицу времени**
  - **В основе – психоакустическая модель, определяющая слышимые диапазоны и области сигнала, точность которых может быть снижена**
    - **MP3 (множество реализаций кодеков, свободный: mp3lame)**
    - **Vorbis (для высококачественных записей – музыки; контейнер хранения – OGG)**
    - **Speex (для низкокачественных записей – речи; контейнер хранения – OGG)**
- **Сжатие без потерь**
  - **FLAC (контейнер хранения – OGG)**

Потоковое воспроизведение

- При воспроизведении музыки (аудиопотоков значительной продолжительности и высокого качества) использовать единственный буфер не представляется возможным
- Для решения данной задачи используется **потоковое воспроизведение**:
  - Создаётся циклическая очередь из нескольких буферов
  - Заполняется первый буфер (необходимо учитывать, что процесс загрузка данных из сжатого файла требует больше времени, в связи с затратами на раскодировку)
  - Первый буфер становится активным, его данные воспроизводятся, и во время этого заполняются последующие буферы
  - Как только все данные из первого буфера воспроизведены, активность передаётся второму, первый уходит в конец очереди и процесс повторяется

Потоковое воспроизведение

- OpenAL предоставляет возможность организации потокового воспроизведения на уровне своего API
  - Относительно алгоритма проигрывания единственного буфера изменяется этап связывания буфера с источником:

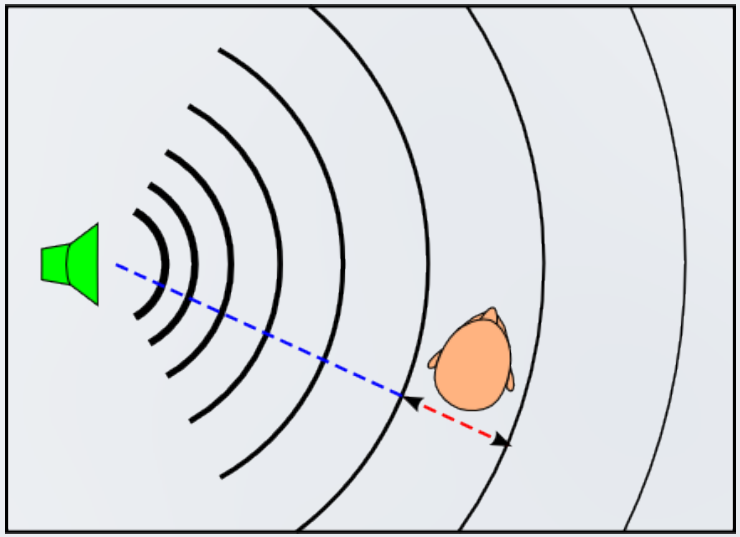
```
alGenBuffers(NUM_OF_DYNBUF, m_buffers);
alSourceQueueBuffers(m_sourceID, NUM_OF_DYNBUF, m_buffers);
```

  - Во время воспроизведения необходимо производить обновление состояния:

```
// Получить количество воспроизведённых буферов
alGetSourcei(m_sourceID, AL_BUFFERS_PROCESSED, &processed);
while (processed--)
{
    // Исключить буфер из очереди
    alSourceUnqueueBuffers(m_sourceID, 1, &bufID);
    // Считать данные в буфер
    readMyBuffer(bufID);
    // Вернуть буфер в очередь
    alSourceQueueBuffers(m_sourceID, 1, &bufID);
}
```

Трёхмерный звук

- OpenAL поддерживает имитацию объёмного звука на основе следующей модели



→ То есть при наличии двух динамиков – левого и правого – эффект восприятия объёма достигается несимметричным изменением громкости на них

- Для активации объёмного звука достаточно установить атрибуты положения для источника и слушателя
- Штатными средствами поддерживается моделирование объёмного звука только для аудиоданных, изначально обладающих одной звуковой дорожкой (моно-звука), так как стереозвук уже содержит в себе несимметричные данные для левого и правого воспроизводящих устройств