**問題:**

音樂機器人中，聲音資訊的接收是非常重要的。但是要如何擷取聲音資訊呢?

**解決方法:**

[音訊擷取小幫手 OpenAL](http://musirobobo.blogspot.com/2011/10/openal.html)

當初一直摸索了openAL，搞了很久，終於成功利用麥克風擷取聲音資料啦~感動(泣)  
openCV我一下子就試出來，可是openAL我卻搞了好久才不容易有個東西!!!原因在於，目前網路上的程式太多了，而且資訊很零碎，所以我這次就"擷取聲音資訊並儲存資料在txt檔"來做程式碼簡單介紹。  
  
為了可以正常使用openAL，必須進行以下前置作業:  
1. 下載 OpenAL SDK並安裝:

* 官網下載頁面:<http://connect.creativelabs.com/openal/Downloads/Forms/AllItems.aspx>
* win系列，請選擇[OpenAL11CoreSDK.zipOpenAL11CoreSDK](http://connect.creativelabs.com/openal/Downloads/OpenAL11CoreSDK.zip)

2. VC++連結

* Visual Studio --> Tool --> Options --> Projects and Solutions --> VC++ Directories --> Include Files 加入 C:\Program Files\OpenAL 1.1 SDK\include (端看安裝位置)
* Visual Studio --> Tool --> Options --> Projects and Solutions --> VC++ Directories --> Library Files 加入 C:\Program Files\OpenAL 1.1 SDK\libs\Win32 (端看安裝位置和系統)

3. 建立專案之後，做連結

* 對專案按右鍵 --> Configuration and Properties --> Linker --> Input --> Additional dependencies --> 加入"openal32.lib"

進行完這些步驟之後，就可以盡情使用openAL囉~

|  |
| --- |
| /\*\*\*\*程式碼開始\*\*\*/ #include <cv.h>  #include <highgui.h>  #include <al.h>  #include <alc.h>  #include <iostream>  #include <stdio.h>  #include <windows.h>  #include <conio.h>  #define SRATE 44100 #define BUFFERSIZE 4410  using namespace std;  ALshort buffer[BUFFERSIZE]; ALint sample;   int main(int argc, char \*argv[]) { alGetError(); FILE \*file; file = fopen("test.txt","w");  ALCdevice \*device = alcCaptureOpenDevice(NULL, SRATE, AL\_FORMAT\_MONO16, BUFFERSIZE);  if (alGetError() != AL\_NO\_ERROR) { cout << "Error!" << endl; return 0; }  alcCaptureStart(device); const ALchar \*actualDeviceName; actualDeviceName = alcGetString(device, ALC\_DEVICE\_SPECIFIER);  cout << "actualDeviceName: " << actualDeviceName << endl; cout << "device: " << device << endl; system("PAUSE");  int line = 1;  while (!\_kbhit()) {  Sleep(10); alcGetIntegerv(device, ALC\_CAPTURE\_SAMPLES, 1, &sample);  if(sample >= 4410) { alcCaptureSamples(device, (ALCvoid \*)buffer, sample); cout << "sample = " << sample << endl; cout << "buffer[0] = " << (int)buffer[0] << endl; cout << "buffer[1] = " << (int)buffer[1] << endl; cout << "buffer[2] = " << (int)buffer[2] << endl; cout << "buffer[3] = " << (int)buffer[3] << endl;  fprintf(file,"==============LINE %d==================\n",line); for(int i = 0; i < BUFFERSIZE; i++) fprintf(file,"[%d] %d\n",i,(int)buffer[i]);  line++;  }  }  alcCaptureStop(device); alcCaptureCloseDevice(device);  return 0; } |

接下來我來重點講解一下...  
(剛接觸不久，可能有些地方理解的不夠正確，有錯請指正。)  
  
**#define SRATE 44100**  
**#define BUFFERSIZE 4410**  
sample rate 設定 44100Hz是標準CD的音質  
BUFFERSIZE 是由於我希望麥克風每儲存4410個samples(約0.1秒)就放進緩衝區，因此我緩衝區大小設定為4410 (samples)  
此外，每個sample大小是2bytes，因此宣告buffer型態為short。  
  
**ALCdevice \*device = alcCaptureOpenDevice(NULL, SRATE, AL\_FORMAT\_MONO16, BUFFERSIZE);**  
格式 ALCdevice \* alcCaptureOpenDevice(const ALCchar \*devicename, ALCuint frequency, ALCenum format, ALCsizei buffersize);  
devicename:若不知道裝置名稱，裝置名稱輸入NULL，程式會幫你找他讀到的音訊裝置。  
frequency: 即sample rate  
format: AL\_FORMAT\_MONO8/16，代表用單顆麥克風，一個sample以8/16bits來表示。  
另外還有AL\_FORMAT\_STEREO8/16，代表用麥克風陣列(可為聲源定位)，一個sample以8/16bits來表示。  
buffersize:緩衝區大小  
  
**alGetError()**  
This function returns the current error state and then clears the error state.  
  
  
**alcCaptureStart(device);**  
This function begins a capture operation  
  
**actualDeviceName = alcGetString(device, ALC\_DEVICE\_SPECIFIER);**  
格式 const ALCchar \* alcGetString(ALCdevice \*device, ALenum param);  
\*device:裝置的位址  
param:端看你希望獲得什麼樣的資訊。 輸入ALC\_DEVICE\_SPECIFIER就是獲得 \*device該位址的裝置名稱。  
  
**alcGetIntegerv(device, ALC\_CAPTURE\_SAMPLES, 1, &sample);**  
格式 void alcGetIntegerv(ALCdevice \*device, ALCenum param, ALCsizei size, ALCint \*data );  
\*device: 裝置的位址  
param: 端看你希望獲得什麼樣的資訊。 輸入ALC\_CAPTURE\_SAMPLES就是調查 \*device已經得到sample的個數。  
size: 獲取緩衝區有多少byte可供使用。輸入1以上即可。輸入0的話，程式以為緩衝區滿了，不進行存檔動作。  
\*data: 把得到的數據存在這裡。(麥克風收集了多少個samples)  
  
  
**alcCaptureSamples(device, (ALCvoid \*)buffer, sample);**  
格式 void alcCaptureSamples(ALCdevice \*device, ALCvoid \*buffer, ALCsizei samples);  
\*device:裝置的位址  
\*buffer: 緩衝區  
samples:取樣個數，由alcGetIntegerv(...)得知。  
  
**alcCaptureStop(device);**  
**alcCaptureCloseDevice(device);**這個功能我就不用多說了，寫上去就對了。  
  
程式的部分就先說明到這裡。希望對剛接觸openAL的人會有所幫助。  
  
讓大家看一下我們目前使用的麥克風設備，共有兩個。  
  
第一個是實驗室筆電上面附的裝置，中間黑色小洞就是收音孔。  
[](http://4.bp.blogspot.com/-fgYMcGAH8DQ/TqrlHgBN99I/AAAAAAAAABs/lZUHbZEOg7A/s1600/CIMG3075.JPG)  
  
第二個是書卷哥張峰鳴所提供的耳麥。  
[](http://3.bp.blogspot.com/-HFUfjQQxAmQ/TqrlwEWaeZI/AAAAAAAAAB4/TUSm1Xx9kz4/s1600/CIMG3077.JPG)  
  
當峰鳴的耳麥接上也有麥克風的ASUS，系統會以峰鳴的耳麥設為優先存取裝置。  
[](http://3.bp.blogspot.com/-9b3g4ri7q0k/TqrmPwtfMnI/AAAAAAAAACE/6h33zUkH_Jw/s1600/CIMG3078.JPG)  
  
  
接下來，是對麥克風的測試影片。

影片連結:<http://musirobobo.blogspot.com/2011/10/openal.html>

程式功能:若麥克風收到的訊號很小，則顯示一顆很SAD的洋蔥頭；若麥克風訊號很大，則顯示唱歌的洋蔥頭  
  
我沒有做任何動作時，周圍環境是安靜的，因此螢幕顯示SAD Onion。  
當我擊掌時，麥克風感應到聲音的瞬間，螢幕顯示Singing Onion。  
最後對麥克風大聲吼叫，Singing Onion則持續了好一段時間。  
  
麥克風資料已經可以讀取，為了完成音樂機器人，接下來工作是要把聲音資料做分析。

**問題:**

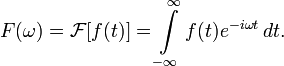
有了聲音資訊，該如何去辨認音樂?

**解決方法:**

音樂的主要特性有三: 音高、音色、音量。

音高就是指聲音的頻率；音色就是指聲音的特色(長相)；音量就是聲音大小。

我目前採用的聲音特徵的頻率和大小來分辨音樂的不同。

我來介紹一下我們音樂分析的方式，  
其實就是大家耳熟能詳的傅立葉轉換。  
  
介紹傅立葉轉換之前，先說明一下傅立葉級數。  
傅立葉級數其實就是把一串訊號用許多不同頻率的sin,cos訊號來表示  
[http://1.bp.blogspot.com/-QO2HTZnNZCI/TuF5WYXkdHI/AAAAAAAAACY/HowlhlNyL2I/s320/63300210376f639deb5ca350fc7fc804.png](http://1.bp.blogspot.com/-QO2HTZnNZCI/TuF5WYXkdHI/AAAAAAAAACY/HowlhlNyL2I/s1600/63300210376f639deb5ca350fc7fc804.png)  
以我們的音樂處理來說，f(x)就是音樂輸入，而a0,a1,a2,...;b1,b2,...就是各種頻率波形的權重，  
音樂是由許多種頻率的聲音所組成，  
若知道各種頻率波形的權重，就能找出音樂的主要頻率。  
  
以連續資料來看  
[http://2.bp.blogspot.com/-USnMvnTwlr0/TuF-cUItpgI/AAAAAAAAACw/O9crlGuNhM4/s320/CIFT.png](http://2.bp.blogspot.com/-USnMvnTwlr0/TuF-cUItpgI/AAAAAAAAACw/O9crlGuNhM4/s1600/CIFT.png)  
f(t)是音樂輸入,F(w)是各種頻率的訊號  
  
目前已經知道，訊號可以分解成許多頻率的訊號的組合體，  
那要如何找出各種頻率訊號的權重，這就靠Fourier Transform了。  
連續訊號 FT:  
[](http://3.bp.blogspot.com/-Y-o_fBnmTBM/TuGGDqthZlI/AAAAAAAAAC8/FBzerETUUDg/s1600/CFT.png)  
離散訊號 FT:  
[http://1.bp.blogspot.com/-CAO6Hm9pL68/TuGGenTz8qI/AAAAAAAAADI/LT1UECBIytI/s320/DIFT.png](http://1.bp.blogspot.com/-CAO6Hm9pL68/TuGGenTz8qI/AAAAAAAAADI/LT1UECBIytI/s1600/DIFT.png)  
  
電腦中的音樂資料是離散訊號，因此使用離散傅立葉轉換，  
不過DFT(Discrete Fourier transform)運算過程太繁瑣，  
計算時間太長，因此發展出(FFT)Fast Fourier Transform，限離散數據(數位資料)使用。  
  
FFT有多快?  
N組數據來說，處理時間上比例為N:logN  
以一筆1024組數據的資料來說，可以節省約100倍的時間。  
而我所處理的音樂資料，一筆有4410組數據，節省更多了。  
  
我所使用的FFT函式庫是FFTW  
<<一個計算DFT (Discrete Fourier Transform) 的C語言函式庫，是目前世界公認執行速度最快的傅立葉轉換軟體。>>  
軟體介紹及使用教學請參考下列網址:  
<http://blog.chinson.idv.tw/2006/05/fftw-fft-c.html>  
  
demo影片連結 - 音樂fft

<http://musirobobo.blogspot.com/2011/11/fast-fourier-transform.html>

<分析圖說明>  
橫軸:頻率(愈往右邊，代表頻率愈大)  
縱軸:振幅(柱子愈高，代表該頻率的成分愈多，以對數方式呈現)  
  
<影片說明>  
一開始周圍環境沒有聲音，因此各種頻率都差不多(雜訊)  
後來播放了兩首歌，可以看到音樂頻率分析的狀況有所不同。  
  
  
下面附上DEMO影片內小軟體的程式碼(有用到一些openCV)

|  |
| --- |
| /\*\*\*\*程式碼開始\*\*\*/ #include < cv.h > #include < highgui.h > #include < al.h > #include < alc.h > #include < iostream > #include < stdio.h > #include < windows.h > #include < conio.h > #include < fftw3.h > #include < math.h >  #define BUFFERSIZE 8820 #define Nr 4410 // # real number #define Nc floor( (double)Nr/2.0 )+1 // # fourier number  using namespace std;  const int SRATE = 44100; // sampling rate. means 44100 samples per second (CD quality) const int SSIZE = 4410; // the size of the ring buffer, Sampling-Rate \* Seconds \* Resolution \* Trackse  ALshort buffer[BUFFERSIZE]; // ALbyte: signed 8-bit 2's-complement integer ALint sample; // ALint: signed 32-bit 2's-complement integer  int main(int argc, char \*argv[]) { int i,j,k,l;  double \*FT\_in; fftw\_complex \*FT\_out; //直角坐標 double \*FT\_Amp; //極座標大小 fftw\_plan FT\_plan;  FT\_in = (double\*) fftw\_malloc(sizeof(double) \* Nr); FT\_out = (fftw\_complex\*) fftw\_malloc(sizeof(fftw\_complex) \* Nc); FT\_Amp = (double\*) fftw\_malloc(sizeof(double) \* Nc); FT\_plan = fftw\_plan\_dft\_r2c\_1d(Nr, FT\_in, FT\_out, FFTW\_ESTIMATE);   IplImage \*image1; image1 = cvCreateImage(cvSize(1103,500),IPL\_DEPTH\_8U,3); // FFT Transform  alGetError(); // This function returns the current error state and then clears the error state  ALCdevice \*device = alcCaptureOpenDevice(NULL, SRATE, AL\_FORMAT\_MONO16, SSIZE);  if (alGetError() != AL\_NO\_ERROR) { cout << "Error!" << endl; return 0; } alcCaptureStart(device);  const ALchar \*actualDeviceName; actualDeviceName = alcGetString(device, ALC\_DEVICE\_SPECIFIER);  cout << "actualDeviceName: " << actualDeviceName << endl; cout << "device adress: " << device << endl; system("PAUSE");  while(1) { Sleep(1); alcGetIntegerv(device, ALC\_CAPTURE\_SAMPLES, 1, &sample);  if(sample >= 4410) { alcCaptureSamples(device, (ALCvoid \*)buffer, sample);  for(i = 0; i < 4410; i++) FT\_in[i] = buffer[i]; fftw\_execute(FT\_plan); for(i = 0; i < Nc; i++) { FT\_Amp[i] = sqrt( pow(FT\_out[i][0],2) + pow(FT\_out[i][1],2) ); // amplitude of polar coordinate }  cvSetZero(image1); for(int j = 0; j < image1->widthStep; j+=3) { k = j/3; l = (float) log10(FT\_Amp[k]/10.0+1)\*80.0;  for(int i = image1->height -1 - l; i < image1->height; i++) {  image1->imageData[i\*image1->widthStep + j ] = 255; image1->imageData[i\*image1->widthStep + j +1] = 255; image1->imageData[i\*image1->widthStep + j +2] = 255; } }  cvNamedWindow("Freq",1); cvShowImage("Freq",image1);  if(cvWaitKey (1) == 'q') break; }  }  alcCaptureStop(device); alcCaptureCloseDevice(device);  cvDestroyWindow("Freq"); cvReleaseImage(&image1);  return 0; } |

**問題:**

由麥克風收到聲音資料，再利用FFT轉成頻譜資料。由影片中看出，轉換出來的頻譜有許多雜訊，是什麼原因導致這種結果? 有什麼解決方式嗎?

**解決方法:**

麥克風接收聲音資料，除了收到喇叭的聲音之外，環境中的雜訊和雜音也是一個很主要的產生干擾的因素。

因此，我打算先把雜訊和雜音的影響消除掉。於是，我希望直接從wav音樂檔去讀取聲音資料。具體方法是說，一邊播放wav的音樂，一邊用頻譜分析把分析資料顯示出來。不過，由於我無法使用openAL播放wav並同時進行擷取該wav內部資訊，所以依然繼續使用有雜訊干擾的聲音做分析。

畢竟，音樂機器人就是要接收外在的聲音來和外界互動，因此重點放在如何在雜訊干擾下做好音樂分析的動作。

**問題:**

如何讓機器人隨著音樂互動?

**解決方法:**

原先的想法是，找出音樂的節奏，然後讓機器人隨著音樂節奏快慢來做出激動到溫和的各種動作。

音樂節奏部分，希望利用頻譜比對法和前幾秒的頻譜做比對，去尋找相同音調的時間間隔，進而找出音樂節奏。但是後來發現，麥克風的雜訊很大，一樣的音樂段落他們的頻譜也會有不小的差異，因此後來放棄找出音樂節奏，改找音調高低和音量大小。

音量大小的找法比較簡單，但是雜訊很多，一般人說話的聲音很容易就會造成音量大小誤判，也很容易因為麥克風移動就讓聲音暴動。因此，有個安靜的環境是很重要的

音調高低的找法，概念上其實就是頻譜分析中挑選出音樂最主要的頻率。但是雜訊也很多，對環境的聲音非常敏感。此外，一般音樂都包含許多樂器和許多高低音，因此頻譜分析上面必須做許多複雜處理，才能夠準確判斷該段落的聲音頻率。所以我們所選用的音樂來源是---口哨。