

北漂行者 > 移动 > 正文

## Android实时获取音量（单位：分贝）

转载wzxwxz20112015-01-07 10:11评论(0)

467人阅读

### 基础知识

度量声音强度，大家最熟悉的单位就是分贝（decibel，缩写为dB）。这是一个无纲量的相对单位，计算公式如下：

$$L_p = 20\log_{10}\left(\frac{p_{rms}}{p_{ref}}\right) \text{ dB}$$

分子是测量值的声压，分母是参考值的声压（20微帕，人类所能听到的最小声压）。因此日常中说道声音强度是多少多少分贝时，都是默认了一个很小的参考值的。

而Android设备传感器可以提供的物理量是场的幅值（amplitude），常用下列公式计算分贝值：

$$L_{dB} = 10\log_{10}\left(\frac{A_1^2}{A_0^2}\right) = 20\log_{10}\left(\frac{A_1}{A_0}\right).$$

从SDK中读取了某段音频数据的振幅后，取最大振幅或平均振幅（可以用平方和平均，或绝对值的和平均），代入上述公式的A1。

现在问题是，作为参考值的振幅A0取多少呢？

博主查阅很多帖子、博文，这里是最一团浆糊的地方。有的博文取600，是基于它视噪音的振幅为600的假设，此时算出来的是相对背景噪音的分贝值，要是用户不对麦克风发出声音，算出的基本都是0分贝。而用户实际使用场景下的背景噪音大小千差万别，咱要是也照葫芦画瓢就不对了，尤其是对于那些制作绝对分贝计的需求，应找出20微帕声压值对应的振幅（或者也可以拿一个标准分贝计做校准参考）。

博主比较懒，把A0定为1，即Android设备麦克风所能“听”到的最小声音振幅。这样拿到测量值振幅直接代入第二个公式的A1中，即可算出分贝值了。

### Android API

使用麦克风需要在AndroidManifest.xml里申请相应权限：

[html] view plaincopyprint?</p>
</div>
<div data-bbox="107 541 591 565" data-label="Text">
<pre>1. <uses-permission android:name="android.permission.RECORD\_AUDIO" />
 <uses-permission android:name="android.permission.RECORD\_AUDIO" />
</pre>
</div>
<div data-bbox="107 579 490 591" data-label="Text">
<p>能够获得音源数据的类有两个：android.media.<b>MediaRecorder</b>和android.media.<b>AudioRecord</b>。</p>
</div>
<div data-bbox="107 608 194 619" data-label="Section-Header">
<h3>MediaRecorder：</h3>
</div>
<div data-bbox="107 623 674 678" data-label="Text">
<p>这个类的对象初始化比较麻烦，因为它<b>是被设计用来录制一段完整的音频并写入文件系统</b>中的。但是初始化之后获得振幅却比较方便，我们直接用它的无参方法<b>getMaxAmplitude</b>即可获得一小段时间内音源数据中的最大振幅。不过取最大值的可能弊端是会受到极端数据的影响，使得后来计算的分贝值波动比较大。不过这个方法是很多录音应用计算音量等级所采用的办法。</p>
</div>
<div data-bbox="107 681 674 752" data-label="Text">
<p>该方法返回的是0到32767范围的16位整型，原理可能是对一段值域为-32768到32767的音源数据取其中绝对值最大的值并返回。这个值与单位为帕斯卡的声压值是有线性函数关系的。另外需要注意的是第一次调用这个方法取得的值是0，代入公式中算出的分贝值是负无穷大，故需要在代码中对这种情况做判断。可以算出，由于getMaxAmplitude返回的数值最大是32767，因此算出的最大分贝值是90.3。也就是说，博主令参考振幅值为1，计算出的分贝值正常值域为<b>0 dB到90.3 dB</b>。</p>
</div>
<div data-bbox="107 755 316 767" data-label="Text">
<p>演示代码如下，基于hongfa.yy的代码改写：</p>
</div>
<div data-bbox="107 784 252 796" data-label="Text">
<p>[java] view plaincopyprint?</p>
</div>
<div data-bbox="107 799 304 899" data-label="Text">
<pre>1. package com.example.myapplication;
2.
3. import java.io.File;
4. import java.io.IOException;
5.
6. import android.media.MediaRecorder;
7. import android.os.Handler;
</pre>
</div>
<div data-bbox="714 122 764 134" data-label="Section-Header">
<h3>北漂行者</h3>
</div>
<div data-bbox="714 148 897 172" data-label="Table">
<table>
<tr>
<td>11</td>
<td>0</td>
<td>0</td>
<td>1W+</td>
</tr>
<tr>
<td>文章</td>
<td>评论</td>
<td>点赞</td>
<td>人气</td>
</tr>
</table>
</div>
<div data-bbox="774 205 845 217" data-label="Section-Header">
<h3>wzxwxz2011</h3>
</div>
<div data-bbox="774 238 871 250" data-label="Text">
<div>TA的主页</div>
<div>私信</div>
</div>
<div data-bbox="738 273 872 285" data-label="Text">
<p>喜欢我的文章，请分享到朋友圈</p>
</div>
<div data-bbox="774 296 834 307" data-label="Section-Header">
<h3>TA的其他文章</h3>
</div>
<div data-bbox="724 319 895 396" data-label="List-Group">
<div>Android项目快速开发框架探索（Mysq...</div>
<div>使用Robolectric对android程序实现自...</div>
<div>我的友情链接</div>
<div>带附件的 SOAP 消息</div>
<div>文本无关的声纹识别 验证</div>
</div>
<div data-bbox="714 421 756 433" data-label="Section-Header">
<h3>七日热门</h3>
</div>
<div data-bbox="724 450 895 526" data-label="List-Group">
<div>利用binlog2sql快速闪回误删除数据 - ...</div>
<div>python魔术方法之装饰器</div>
<div>使用ansible结合keepalived高可用，nq...</div>
<div>企业级硬件及性能调优（01）</div>
<div>后端服务性能压测实践</div>
</div>
</div>

```
8. import android.util.Log;
9.
10. /**
11.  * amr音频处理
12.  *
13.  * @author hongfa.yy
14.  * @version 创建时间2012-11-21 下午4:33:28
15.  */
16. public class MediaRecorderDemo {
17.     private final String TAG = "MediaRecord";
18.     private MediaRecorder mMediaRecorder;
19.     public static final int MAX_LENGTH = 1000 * 60 * 10; // 最大录音时长1000*60*10;
20.     private String filePath;
21.
22.     public MediaRecorderDemo(){
23.         this.filePath = "/dev/null";
24.     }
25.
26.     public MediaRecorderDemo(File file) {
27.         this.filePath = file.getAbsolutePath();
28.     }
29.
30.     private long startTime;
31.     private long endTime;
32.
33.     /**
34.     * 开始录音 使用amr格式
35.     *
36.     *      录音文件
37.     * @return
38.     */
39.     public void startRecord() {
40.         // 开始录音
41.         /* ①Initial：实例化MediaRecorder对象 */
42.         if (mMediaRecorder == null)
43.             mMediaRecorder = new MediaRecorder();
44.         try {
45.             /* ②setAudioSource/setVedioSource */
46.             mMediaRecorder.setAudioSource(MediaRecorder.AudioSource.MIC); // 设置麦克风
47.             /* ②设置音频文件的编码：AAC/AMR_NB/AMR_MB/Default 声音的（波形）的采样 */
48.             mMediaRecorder.setOutputFormat(MediaRecorder.OutputFormat.DEFAULT);
49.             /*
50.             * ②设置输出文件的格式：THREE_GPP/MPEG-4/RAW_AMR/Default THREE_GPP(3gp格式
51.             * , H263视频/ARM音频编码)、MPEG-4、RAW_AMR(只支持音频且音频编码要求为AMR_NB)
52.             */
53.             mMediaRecorder.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB);
54.
55.             /* ③准备 */
56.             mMediaRecorder.setOutputFile(filePath);
57.             mMediaRecorder.setMaxDuration(MAX_LENGTH);
58.             mMediaRecorder.prepare();
59.             /* ④开始 */
60.             mMediaRecorder.start();
61.             // AudioRecord audioRecord.
62.             /* 获取开始时间 */
63.             startTime = System.currentTimeMillis();
```



```
64.     updateMicStatus();
65.     Log.i("ACTION_START", "startTime" + startTime);
66. } catch (IllegalStateException e) {
67.     Log.i(TAG,
68.         "call startAmr(File mRecAudioFile) failed!"
69.         + e.getMessage());
70. } catch (IOException e) {
71.     Log.i(TAG,
72.         "call startAmr(File mRecAudioFile) failed!"
73.         + e.getMessage());
74. }
75. }
76.
77. /**
78.  * 停止录音
79.  *
80.  */
81. public long stopRecord() {
82.     if (mMediaRecorder == null)
83.         return 0L;
84.     endTime = System.currentTimeMillis();
85.     Log.i("ACTION_END", "endTime" + endTime);
86.     mMediaRecorder.stop();
87.     mMediaRecorder.reset();
88.     mMediaRecorder.release();
89.     mMediaRecorder = null;
90.     Log.i("ACTION_LENGTH", "Time" + (endTime - startTime));
91.     return endTime - startTime;
92. }
93.
94. private final Handler mHandler = new Handler();
95. private Runnable mUpdateMicStatusTimer = new Runnable() {
96.     public void run() {
97.         updateMicStatus();
98.     }
99. };
100.
101. /**
102.  * 更新话筒状态
103.  *
104.  */
105. private int BASE = 1;
106. private int SPACE = 100;// 间隔取样时间
107.
108. private void updateMicStatus() {
109.     if (mMediaRecorder != null) {
110.         double ratio = (double)mMediaRecorder.getMaxAmplitude() /BASE;
111.         double db = 0;// 分贝
112.         if (ratio > 1)
113.             db = 20 * Math.log10(ratio);
114.         Log.d(TAG,"分贝值 : "+db);
115.         mHandler.postDelayed(mUpdateMicStatusTimer, SPACE);
116.     }
117. }
118. }
package com.example.myapplication;
```



```
import java.io.File;
import java.io.IOException;

import android.media.MediaRecorder;
import android.os.Handler;
import android.util.Log;

/**
 * amr音频处理
 *
 * @author hongfa.yy
 * @version 创建时间2012-11-21 下午4:33:28
 */
public class MediaRecorderDemo {
    private final String TAG = "MediaRecord";
    private MediaRecorder mMediaRecorder;
    public static final int MAX_LENGTH = 1000 * 60 * 10; // 最大录音时长1000*60*10;
    private String filePath;

    public MediaRecorderDemo(){
        this.filePath = "/dev/null";
    }

    public MediaRecorderDemo(File file) {
        this.filePath = file.getAbsolutePath();
    }

    private long startTime;
    private long endTime;

    /**
     * 开始录音 使用amr格式
     *
     * 录音文件
     * @return
     */
    public void startRecord() {
        // 开始录音
        /* ①Initial: 实例化MediaRecorder对象 */
        if (mMediaRecorder == null)
            mMediaRecorder = new MediaRecorder();
        try {
            /* ②setAudioSource/setVedioSource */
            mMediaRecorder.setAudioSource(MediaRecorder.AudioSource.MIC); // 设置麦克风
            /* ②设置音频文件的编码：AAC/AMR_NB/AMR_MB/Default 声音的（波形）的采
            mMediaRecorder.setOutputFormat(MediaRecorder.OutputFormat.DEFAULT);
            /*
            * ②设置输出文件的格式：THREE_GPP/MPEG-4/RAW_AMR/Default THREE_Gf
            * , H263视频/ARM音频编码)、MPEG-4、RAW_AMR(只支持音频且音频编码要求为
            */
            mMediaRecorder.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB);

            /* ③准备 */
            mMediaRecorder.setOutputFile(filePath);
            mMediaRecorder.setMaxDuration(MAX_LENGTH);
```



```
mMediaRecorder.prepare();
    /* ④开始 */
    mMediaRecorder.start();
    // AudioRecord audioRecord.
    /* 获取开始时间 */
    startTime = System.currentTimeMillis();
    updateMicStatus();
    Log.i("ACTION_START", "startTime" + startTime);
} catch (IllegalStateException e) {
    Log.i(TAG,
        "call startAmr(File mRecAudioFile) failed!"
        + e.getMessage());
} catch (IOException e) {
    Log.i(TAG,
        "call startAmr(File mRecAudioFile) failed!"
        + e.getMessage());
}
}

/**
 * 停止录音
 *
 */
public long stopRecord() {
    if (mMediaRecorder == null)
        return 0L;
    endTime = System.currentTimeMillis();
    Log.i("ACTION_END", "endTime" + endTime);
    mMediaRecorder.stop();
    mMediaRecorder.reset();
    mMediaRecorder.release();
    mMediaRecorder = null;
    Log.i("ACTION_LENGTH", "Time" + (endTime - startTime));
    return endTime - startTime;
}

private final Handler mHandler = new Handler();
private Runnable mUpdateMicStatusTimer = new Runnable() {
    public void run() {
        updateMicStatus();
    }
};

/**
 * 更新话筒状态
 *
 */
private int BASE = 1;
private int SPACE = 100; // 间隔取样时间

private void updateMicStatus() {
    if (mMediaRecorder != null) {
        double ratio = (double)mMediaRecorder.getMaxAmplitude() / BASE;
        double db = 0; // 分贝
        if (ratio > 1)
            db = 20 * Math.log10(ratio);
    }
}
```



```
        Log.d(TAG, "分贝值 : "+db);
        mHandler.postDelayed(mUpdateMicStatusTimer, SPACE);
    }
}
```

#### AudioRecord :

这个类可以获得具体的音源数据值。将一段音源数据用`read(byte[] audioData, int offsetInBytes, int sizeInBytes)`方法从缓冲区读取到我们传入的字节数组`audioData`后，便可以对其进行操作，如求平方和或绝对值的平均值。这样可以避免个别极端值的影响，使计算的结果更加稳定。求得平均值之后，如果是平方和则代入常数系数为**10**的公式中，如果是绝对值的则代入常数系数为**20**的公式中，算出分贝值。

演示代码如下：

[\[java\] view plaincopyprint?](#)

```
1. package com.example.myapplication;
2.
3. import android.media.AudioFormat;
4. import android.media.AudioRecord;
5. import android.media.MediaRecorder;
6. import android.util.Log;
7.
8. /**
9.  * Created by greatpresident on 2014/8/5.
10. */
11. public class AudioRecordDemo {
12.
13.     private static final String TAG = "AudioRecord";
14.     static final int SAMPLE_RATE_IN_HZ = 8000;
15.     static final int BUFFER_SIZE = AudioRecord.getMinBufferSize(SAMPLE_RATE_IN_HZ,
16.         AudioFormat.CHANNEL_IN_DEFAULT, AudioFormat.ENCODING_PCM_16BIT);
17.     AudioRecord mAudioRecord;
18.     boolean isGetVoiceRun;
19.     Object mLock;
20.
21.     public AudioRecordDemo() {
22.         mLock = new Object();
23.     }
24.
25.     public void getNoiseLevel() {
26.         if (isGetVoiceRun) {
27.             Log.e(TAG, "还在录着呢");
28.             return;
29.         }
30.         mAudioRecord = new AudioRecord(MediaRecorder.AudioSource.MIC,
31.             SAMPLE_RATE_IN_HZ, AudioFormat.CHANNEL_IN_DEFAULT,
32.             AudioFormat.ENCODING_PCM_16BIT, BUFFER_SIZE);
33.         if (mAudioRecord == null) {
34.             Log.e("sound", "mAudioRecord初始化失败");
35.         }
36.         isGetVoiceRun = true;
37.
38.         new Thread(new Runnable() {
39.             @Override
40.             public void run() {
41.                 mAudioRecord.startRecording();
42.                 short[] buffer = new short[BUFFER_SIZE];
43.                 while (isGetVoiceRun) {
```



```
44.         //r是实际读取的数据长度，一般而言r会小于bufferSize
45.         int r = mAudioRecord.read(buffer, 0, BUFFER_SIZE);
46.         long v = 0;
47.         // 将 buffer 内容取出，进行平方和运算
48.         for (int i = 0; i < buffer.length; i++) {
49.             v += buffer[i] * buffer[i];
50.         }
51.         // 平方和除以数据总长度，得到音量大小。
52.         double mean = v / (double) r;
53.         double volume = 10 * Math.log10(mean);
54.         Log.d(TAG, "分贝值:" + volume);
55.         // 大概一秒十次
56.         synchronized (mLock) {
57.             try {
58.                 mLock.wait(100);
59.             } catch (InterruptedException e) {
60.                 e.printStackTrace();
61.             }
62.         }
63.     }
64.     mAudioRecord.stop();
65.     mAudioRecord.release();
66.     mAudioRecord = null;
67. }
68. }).start();
69. }
70. }

package com.example.myapplication;

import android.media.AudioFormat;
import android.media.AudioRecord;
import android.media.MediaRecorder;
import android.util.Log;

/**
 * Created by greatpresident on 2014/8/5.
 */

public class AudioRecordDemo {

    private static final String TAG = "AudioRecord";
    static final int SAMPLE_RATE_IN_HZ = 8000;
    static final int BUFFER_SIZE = AudioRecord.getMinBufferSize(SAMPLE_RATE_IN_HZ,
        AudioFormat.CHANNEL_IN_DEFAULT, AudioFormat.ENCODING_PCM_16BIT);
    AudioRecord mAudioRecord;
    boolean isGetVoiceRun;
    Object mLock;

    public AudioRecordDemo() {
        mLock = new Object();
    }

    public void getNoiseLevel() {
        if (isGetVoiceRun) {
            Log.e(TAG, "还在录着呢");
            return;
        }
    }
}
```



```
mAudioRecord = new AudioRecord(MediaRecorder.AudioSource.MIC,
    SAMPLE_RATE_IN_HZ, AudioFormat.CHANNEL_IN_DEFAULT,
    AudioFormat.ENCODING_PCM_16BIT, BUFFER_SIZE);
if (mAudioRecord == null) {
    Log.e("sound", "mAudioRecord初始化失败");
}
isGetVoiceRun = true;

new Thread(new Runnable() {
    @Override
    public void run() {
        mAudioRecord.startRecording();
        short[] buffer = new short[BUFFER_SIZE];
        while (isGetVoiceRun) {
            //r是实际读取的数据长度，一般而言r会小于buffersize
            int r = mAudioRecord.read(buffer, 0, BUFFER_SIZE);
            long v = 0;
            // 将 buffer 内容取出，进行平方和运算
            for (int i = 0; i < buffer.length; i++) {
                v += buffer[i] * buffer[i];
            }
            // 平方和除以数据总长度，得到音量大小。
            double mean = v / (double) r;
            double volume = 10 * Math.log10(mean);
            Log.d(TAG, "分贝值:" + volume);
            // 大概一秒十次
            synchronized (mLock) {
                try {
                    mLock.wait(100);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        }
        mAudioRecord.stop();
        mAudioRecord.release();
        mAudioRecord = null;
    }
}).start();
}
```

实测结果（设备小米2S），MediaRecorderDemo波动很大，只要对麦克风一吹气，分贝值就能上90：

```
D/MediaRecord: 分贝值: 48.755011256407755
D/MediaRecord: 分贝值: 46.4029257222108
D/MediaRecord: 分贝值: 50.04854239968865
D/MediaRecord: 分贝值: 48.62727528317975
D/MediaRecord: 分贝值: 48.912084065471944
D/MediaRecord: 分贝值: 50.65508757984995
D/MediaRecord: 分贝值: 43.52182518111363
D/MediaRecord: 分贝值: 49.24795995797912
D/MediaRecord: 分贝值: 88.25559281640594
D/MediaRecord: 分贝值: 89.26770538432122
D/MediaRecord: 分贝值: 89.15945361032414
D/MediaRecord: 分贝值: 89.2885985690169
D/MediaRecord: 分贝值: 89.5346042738216
D/MediaRecord: 分贝值: 90.03268574617067
D/MediaRecord: 分贝值: 89.48024370339701
```

而AudioRecordDemo就很稳定了，很用力吹气也很难到88以上：





```
D/AudioRecord: 分贝值:42.40506113085837
D/AudioRecord: 分贝值:41.5141017079473
D/AudioRecord: 分贝值:48.74564019284746
D/AudioRecord: 分贝值:45.9311212744289
D/AudioRecord: 分贝值:82.901236544703
D/AudioRecord: 分贝值:86.49338151219047
D/AudioRecord: 分贝值:87.12168737484804
D/AudioRecord: 分贝值:86.75565907140991
```


[基础知识](#)   [麦克风](#)   [Android](#)   [传感器](#)   [绝对值](#)

0

[收藏](#)   [分享](#)

[上一篇：带附件的 SOAP 消息](#)   [下一篇：我的友情链接](#)

发表评论



写下你的评论

Ctrl+Enter 发布

取消

发布

