AVSSDK_For_Linux
用户集成指南V2.4.4

深圳声联网科技有限公司

AVSSDK_For_Linux_用户集成指南： 修正记录		
版本	发布日期	内容描述
V1.0	2016/06/20	正式版
V2.0	2017/03/20	优化了基础模型。更新了接口参数。增加了获取版本接口。
V2.2	2017/12/04	优化了基础模型、算法，更新了接口及参数。
V2.4.1	2017/12/20	优化了基础模型、算法，更新了接口及参数。
V2.4.2	2018/01/15	优化了基础模型、算法，修改了部分接口参数，更新了验证方式。
V2.4.3	2018/03/12	增加了参数设置接口。
V2.4.4	2018/06/05	增加了标准检测接口，丰富返回结果，增加了静音阈值的参数设置。

地址：广东省深圳市南山区松坪山路 1 号源兴科技大厦北座 406。

0755-33349168 33349598 **0755-33349798** **www.avsnest.com**

重要声明

版权声明

商标声明

深圳声联网科技有限公司的产品是深圳声联网科技有限公司专有。在提及其他公司及其产品时将使用各自公司所拥有的商标，这种使用的目的仅限于引用。本文档可能涉及深圳声联网科技有限公司的专利（或正在申请的专利）、商标、版权或其他知识产权，除非得到深圳声联网科技有限公司的明确书面许可协议，本文档不授予使用这些专利（或正在申请的专利）、商标、版权或其他知识产权的任何许可协议。

不作保证声明

深圳声联网科技有限公司不对此文档中的任何内容作任何明示或暗示的陈述或保证，而且不对特定目的的适销性及适用性或者任何间接、特殊或连带的损失承担任何责任。本手册内容若有变动，恕不另行通知。本手册例子中所用的公司、人名和数据若非特别声明，均属虚构。未得到深圳声联网科技有限公司明确的书面许可，不得为任何目的、以任何形式或手段（电子的或机械的）复制或传播手册的任何部分。

保密声明

本文档（包括任何附件）包含的信息是保密信息。接收人了解其获得的本文档是保密的，除用于规定的目的外不得用于任何目的，也不得将本文档泄露给任何第三方。

本软件产品受最终用户许可协议（EULA）中所述条款和条件的约束，该协议位于产品文档和/或软件产品的联机文档中，使用本产品，表明您已阅读并接受了EULA的条款。

目录

1. 概述 1-4

2. 平台支持 2-4

3. SDK使用向导 3-4

3.1 开发准备 3-4
3.2 枚举定义 3-4
3.3 接口 3-5

4. 注意事项 4-8

5. 引擎重新激活 5-8

6. 错误码表 6-9

1. 概述

AVSSDK_For_Linux 是深圳声联网科技有限公司在 Linux 平台上对声联网婴幼儿情感计算算法的封装。用户只需要简单的调用封装后的接口就可以在 Linux 平台的产品上集成声联网婴幼儿情感计算算法的功能。

2. 平台支持

AVSSDK_For_Linux 支持几乎所有 Linux 系统的嵌入式平台。

3. SDK 使用向导

3.1 开发准备

在使用 AVSSDK_For_Linux 进行开发之前，先准备开发需要的文件。

头文件：

文件名	说明
include/avs_sdk.h	AVSSDK 包的接口头文件

参考源码文件：

文件名	说明
sample/demo.c **sample/Makefile**	AVSSDK 包的参考代码及编译 Makefile 文件

库文件：

文件名	说明
lib/libavssdk.so	AVSSDK 动态库文件

3.2 枚举定义

声音事件返回类型

```
typedef enum
{
    AVS_EVENT_ERROR =-1,          //错误状态
    AVS_EVENT_NOR = 0,            //中间状态
    AVS_EVENT_MUTE,               //静音状态
    AVS_EVENT_PUBLIC,             //公共状态
    AVS_EVENT_CRY,                //哭声状态
    AVS_EVENT_LAUGH,              //笑声状态（暂无）
```

```
    AVS_EVENT_SOUND_ABNORMAL,      //异常声音
    AVS_EVENT_ACTIVE
}AVS_EVENT_E;
```

参数类型

```
typedef enum
{
    AVSP_ABNORMAL_DB, //异常声音阈值
    AVSP_MUTE_DB,//静音阈值
}AVS_PARAM_TYPE_E;
```

3.3 接口

■ int avs_init(unsigned char *eth, unsigned char *deviceid);

声音事件检测引擎初始化。

参数：

eth **[in]** 当前通信网卡名称字符串如“wlan0”。

deviceid **[in]** 用户输入的当前设备 id 号。

注：deviceid,代表当前设备的唯一 id 号（每设备唯一），由客户提供，通过接口传递给识别引擎。

返回值：成功返回 0；失败返回 -1。

当返回失败时，调用 **int avs_get_error()** 获取错误码。

设备初始化时会在内部进行网络通信，网络通信超时为 5 秒。如果 5 秒没有连接上服务器，会直接返回失败（-1）。这时使用 **avs_get_error()**可以获取到错误码。

■ int avs_active(unsigned char *eth, unsigned char *deviceid);

声音事件引擎重新激活

参数：

eth **[in]** 当前通信网卡名称字符串如“wlan0”。

deviceid **[in]** 用户输入的当前设备 id 号。

注：deviceid,代表当前设备的唯一 id 号（每设备唯一），由客户提供，通过接口传递给识别引擎。

返回值：成功返回 0；失败返回 -1

当返回失败时，调用 **int avs_get_error()** 获取错误码。

设备初始化时会在内部进行网络通信，网络通信超时为 5 秒。如果 5 秒没有连接上服务器，会直接返回失败（-1）。这时使用 **avs_get_error()**可以获取到错误码。

■ AVS_EVENT_E avs_detect(short* data, unsigned short size);

[简化接口]传入音频数据进行声音事件检测。

参数：

data [in] 需要传入的音频数据地址。

size [in] 需要传入的音频数据采样个数（范围：**[**160～8000**]**个采样）。

返回值：

返回检测结果 AVS_EVENT_E 中的一种。

注明：当返回值为 AVS_EVENT_ACTIVE 时，表示检测引擎失活，需要重新联网激活。

激活方式详见引擎重新激活

■ short avs_detects (short* data, unsigned short size, char *RetStata);

[标准接口]传入音频数据进行声音事件检测。

参数：

data [in] 需要传入的音频数据地址。

size [in] 需要传入的音频数据采样个数（范围：**[**160～8000**]**个采样）。

RetStata[out] bit 方式返回类型、分贝值、等状态信息。

下表所示：RetStata[x].bit[y] = 1 代表有效

RetStata[0]							
bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
RESERVE(保留)	*RESERVE(保留)*	*ABNORMAL(异常)*	*LAUGH(笑声)*	*CRY(哭声)*	*PUBLIC(公共)*	*MUTE(静音)*	*NOR(无结果)*
RetStata[1]							
bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
RESERVE(保留)	*RESERVE(保留)*	*RESERVE(保留)*	*RESERVE(保留)*	*RESERVE(保留)*	*RESERVE(保留)*	*RESERVE(保留)*	*RESERVE(保留)*
RetStata[2]							
bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
DB_VALUE(分贝值)							
RetStata[3]							
bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
RESERVE(保留)	*RESERVE(保留)*	*RESERVE(保留)*	*RESERVE(保留)*	*RESERVE(保留)*	*RESERVE(保留)*	*RESERVE(保留)*	*RESERVE(保留)*

返回值：成功返回 0；失败返回-1；引擎失活返回-2；

注明：当返回值为 -2 (引擎失活) 时，等同于 [简化接口] **avs_detect** 的返回值 AVS_EVENT_ACTIVE， 需

要重新联网激活。

激活方式详见引擎重新激活

■ int avs_setparam(AVS_PARAM_TYPE_E type, void *valueptr, int len);

设置运行参数。

参数：

type[in]：参数索引，可见枚举类型 AVS_PARAM_TYPE_E。

***valueptr[in]：此地址用来设置参数值。**

len[in]：参数长度。

返回值：成功返回 0；失败返回-1。

参数类型描述				
参数索引	**参数类型**	**参数长度**	**默认值**	**描述**
AVSP_ABNORMAL_DB	float	sizeof(float)	5.0	取值 0.0 ≤ x ≤ 9.0
AVSP_MUTE_DB	float	Sizeof(float)	0.0	取值 0.0 ≤ x ≤ 9.0

■ int avs_getparam(AVS_PARAM_TYPE_E type, void *valueptr, int *len);

获取运行参数。

参数：

type[in]：参数索引，可见枚举类型 AVS_PARAM_TYPE_E。

***valueptr[out]：此地址用来返回相应参数值。**

***len[out]：此地址用来返回参数长度。**

返回值：成功返回 0；失败返回-1。

■ void avs_fini(void);

销毁声音事件检测引擎。

参数：

无

返回值：

无

■ void avs_get_version(char *ver);

获取 **AVSSDK** 的版本号。传入接收版本号的指针 ver 时，长度不应小于 50Byte。

参数：

ver [out] 存放版本字符串的指针。

返回值：

无。

■ int avs_get_error(void);

当其 SDK 出错或者异常时，通过此接口获取错误码。

参数：

无

返回值：

错误码代表上次出错的，原因代号。

4. 注意事项

- AVSSDK 依赖数学库、所以在编调用程序时，请在链接时携带参数 –lm –lavssdk。具体可参考 sample/Makefile 文件。
- 接口使用方法请参照 demo 中的例程。
- 运行 demo 例程时，若您拿到的是 16K 版本，请务必放置一个采样率为 16000，采样精度 16 位，单声道且名称为“test.wav”的哭声音频文件在 demo 同级目录，若您拿到的是 8K 版本，请务必放置一个采样率为 8000，采样精度 16 位，单声道且名称为“test.wav”的哭声音频文件在 demo 同级目录，
- 交叉编译完毕后，请把 libavssdk.so 部署到目标机上的动态库目录，并更新目标机上的动态库搜索路径，以便 demo 例程或调用程序在执行时找到并加载 libavssdk.so。
- 由于 avs_init 与 avs_active 接口整合了后台验证机制，因此涉及到网络通信相关内容。当调用 avs_init 或 avs_active 时由于某种无法预见原因（常见于：网络状况不好、服务器繁忙等），有可能出现调用失败的问题。这时应该延时一段时间再次调用，并且当重复 3-5 次重复调用后，最终仍旧返回调用失败后，才能判定为真正调用失败,并通过 avs_get_error 并查表分析错误原因。

5. 引擎重新激活

本“声音事件检测引擎”以下简称称“引擎”，内置了联网验证和监控模块，引擎调用初始化接口 avs_init 时会联网进行验证。验证成功才能进行事件检测。

引擎在初始化成功后，并非永远保持存活状态。判断引擎失活的重要条件即是检查 avs_detect/avs_detects的返回值，当 avs_detect 返回值为 AVS_EVENT_ACTIVE / avs_detects 的返回值为 -2 时表示引擎即将失活，需要在短期内重新激活一下，否则会引擎失效。

激活方式可以采用如下两种方式：

- 重新初始化方式

当 avs_detect 接口返回值出现 AVS_EVENT_ACTIVE / avs_detects 的返回值为 -2 时，立即调用 avs_fini 接口销毁当前失活的引擎，然后重新调用 avs_init 建立新的检测引擎，继续调用 avs_detect / avs_detects 检测即可。

- 使用激活接口激活方式

当 avs_detect 接口返回值出现 AVS_EVENT_ACTIVE / avs_detects 接口返回 -2 时，立即调用 avs_active 接口联网激活当前引擎，激活成功后继续调用 avs_detect/avs_detects 进行检测。

avs_init 及 avs_active 需要联网授权获取激活码。联网授权需要短暂耗时，耗时情况视网络状况而定。因此在产品使用中需要适当注意，当需要进行激活操作时，需要进行录音缓存。

6. 错误码表

avs_get_error() 返回值请查表分析原因

错误码(int)	描述
100	基本错误码
101	未知错误
102	参数错误
103	内存操作错误
104	引擎内部错误
10000	获取MAC错误
10001	域名解析错误
10002	创建会话错误
10003	连接服务器错误
10004	发送错误
10005	接收错误
10006	关闭会话错误
10007	连接服务器超时
50005	非法授权
50006	用户被禁用
50007	未通过授权
50008	数量超限
50009	授权过期
50010	授权成功
60001	未知错误
其他值	如返回其他非上述错误码，请联系FAQ