**人脸比对算法库使用说明**

当前文档描述版本：V3.02

* **算法使用文件介绍**

使用人脸检测算法需要使用到的文件：

FaceNative.jar // Java层调用接口Jar包

libfaceDetect.so // 人脸检测算法封装库

libfaceFeature.so // 人脸比对算法封装库

libTHFaceImage.so // 人脸检测算法库

libTHFeature.so // 人脸比对算法库

Libcmp.so // 人脸算法配置包

libface.so // 人脸算法配置包

libfea.so // 人脸算法配置包

liblm.so // 人脸算法配置包

文件在项目中配置目录结构：

在工程中添加libs文件夹，将算法相关文件按照如下目录放置。



* **接口文件介绍**

**1、人脸检测接口文件：**

**Com.face.sv.FaceDetect** {

/\*\*

\* 获取算法加密KEY

\* **@return** 加密字符串;

\*/

**public** String getDetectSN();

/\*\*

\* DM2016验证KEY

\* **@sn** 解密后的字节数组

\* **@return** 成功返回1，失败返回0, 编码失败返回-1。

\*/

**public** **int** CheckDetectSN(String sn);

/\*\*

\* 设置临时目录

\* @ libDir 算法库包路径

\* @ tempDir 临时目录地址，当前应用必须拥有操作权限

\*/

**public** **void** setDir(String libDir, String tempDir);

/\*\*

\* 初始化人脸检测算法库

\* **@chlNum 算法通道数**，默认设置1。

\* **@return** 初始化成功返回true, 否则返回false;

\*/

**public** **boolean** initFaceDetectLib(**int** chlNum);

/\*\*

\* 释放人脸检测算法库

\*/

**public** **void** releaseFaceDetectLib();

/\*\*

\* 获取RGB24图片数据中人脸信息

\* **@param** chlID 算法通道ID，默认设置0。

\* **@param** gray 需要检测的人脸照片的灰度图数据

\* **@param** width 图片宽度

\* **@param** height 图片高度

\* **@return**

\* length = 5; 数组长度

retArr[0] = ret; 检测结果，成功返回人脸数量，失败返回0或负数

\* 0 ： 没有检测到人脸

\* -1 ： 传入的gray数据为空

\* 1： 人脸数量为1，后续4个整形数据为

retArr[1] = pfps->rcFace.left; 人脸左边界

retArr[2] = pfps->rcFace.top; 人脸上边界

retArr[3] = pfps->rcFace.right; 人脸右边界

retArr[4] = pfps->rcFace.bottom; 人脸下边界

\*/

**public** **int**[] getFacePositionFromGray(**short** chlID, **byte**[] gray, **int** width, **int** height);

/\*\*

\* 获取图片中人脸信息

\* **@param** chlID 算法通道ID，默认设置0。

\* **@param** bmp 需要检测的人脸照片

\* **@return**

\* length = 5; 数组长度

retArr[0] = ret; 检测结果，成功返回人脸数量，失败返回0或负数

\* 0 : 没有检测到人脸

\* -1 ： 传入的gray数据为空

\* 1： 人脸数量为1，后续4个整形数据为

retArr[1] = pfps->rcFace.left; 人脸左边界

retArr[2] = pfps->rcFace.top; 人脸上边界

retArr[3] = pfps->rcFace.right; 人脸右边界

retArr[4] = pfps->rcFace.bottom; 人脸下边界

\*/

**public** **int**[] getFacePositionFromBitmap(**short** chlID, Bitmap bmp);

}

**2、人脸比对接口文件：**

**Com.face.sv.FaceFeature** {

/\*\*

\* 获取算法加密KEY

\* **@return** 加密字符串;

\*/

**public** String getFeatureSN();

/\*\*

\* DM2016验证KEY

\* **@sncode** 解密后的字节数组

\* **@return** 成功返回1，失败返回0, 编码失败返回-1。

\*/

**public** **int** CheckFeatureSN(String sn);

/\*\*

\* 设置临时目录

\* @ libDir 算法库包路径

\* @ tempDir 临时目录地址，当前应用必须拥有操作权限

\*/

**public** **void** setDir(String libDir, String tempDir)；

/\*\*

\* 初始化人脸比对算法库

\* **@chlNum 算法通道数**，默认设置1。

\* **@return** 成功返回true, 失败返回false;

\*/

**public** **boolean** initFaceFeatureLib(**int** chlNum)；

/\*\*

\* 释放人脸比对算法库

\*/

**public** **void** releaseFaceFeatureLib()；

/\*\*

\* 获取人脸图片中人脸模板

\* **@param** chlID 算法通道ID，默认设置0。

\* **@param** rgb24 人脸图片

\* **@param** rect 图片人脸坐标{left, top, right, bottom}

\* **@param** width 图片宽度

\* **@param** height 图片高度

\* **@return** 人脸模板数据, 失败数组长度为1，byte[0]返回0或负数，成功返回2008字节模板数据。

。

\* 0 : 提取模板失败

\* -1 ： 传入的RGB数据为空或rect为空

\* -2 ： 分配模板内存失败

\*/

**public** **byte**[] getFaceFeatureFromRGB(**short** chlID, **byte**[] rgb24, **int**[] rect, **int** width, **int** height)；

/\*\*

\* 获取人脸图片中人脸模板

\* **@param** chlID 算法通道ID，默认设置0。

\* **@param** img 人脸图片

\* **@return** 人脸模板数据, 失败数组长度为1，float[0]返回0或负数，成功返回2008字节模板数据。

\* 0 : 提取模板失败

\* -1 ： 传入的RGB数据为空或rect为空

\* -2 ： 分配模板内存失败

\*/

**public** **byte**[] getFaceFeatureFromBitmap(**short** chlID, Bitmap bmp, **int**[] rect)；

/\*\*

\* 比对两个人脸模板相似度

\* **@param** chlID1 算法通道ID，默认设置0。

\* **@param** chlID2 算法通道ID，默认设置0。

\* **@param** bmp1 人脸图片1

\* **@param** bmp2 人脸图片2

\* **@return** 相识度（分值范围0 ~ 100之间）

\*/

**public** **int** compareFaces(**short** chlID1, Bitmap bmp1, **int**[] rect1, **short** chlID2, Bitmap bmp2, **int**[] rect2)；

/\*\*

\* 比对两个人脸模板相似度

\* **@param** feature1 人脸模板

\* **@param** feature2 人脸模板

\* **@return** 相识度（分值范围0 ~ 100之间）， -1 表示参数feature1或feature2为null.

\*/

**public** **int** compareFeatures(**final** **byte**[] feature1, **final** **byte**[] feature2)；

* **算法调用步骤**

使用算法需要注意的步骤：

1. 算法使用需要配置算法库文件路径和临时文件路径。

/\*\*

\* 设置临时目录

\* @ libDir 算法库包路径

\* @ tempDir 临时目录地址，当前应用必须拥有操作权限

\*/

**public** **void** setDir(String libDir, String tempDir)；

默认设置如下目录：

File file = this.getCacheDir();

tempDir = file.getAbsolutePath();

libDir = tempDir.replace("cache", "lib");

1. 加密芯片DM2016鉴权

（使用DM2016鉴权版本算法必要步骤，限制时间版本算法不用该步骤。）

A.先调用算法接口获取KEY字符串；

/\*\*

\* 获取算法加密KEY

\* **@return** 加密字符串;

\*/

**public** String getFeatureSN()；

B.通过加密芯片DM2016接口加密KEY密字符串；

C. 调用算法接口输入加密后的字符串，提供给算法鉴权。

/\*\*

\* DM2016验证KEY

\* **@sncode** 解密后的字节数组

\* **@return** 成功返回1，失败返回0, 编码失败返回-1。

\*/

**public** **int** CheckFeatureSN(String sn)；

D. 鉴权成功，才能进行后续的初始化，否则会返回鉴权失败（-99）。

3、使用前需要先初始化算法库

/\*\*

\* 初始化人脸比对算法库

\* **@chlNum 算法通道数**，默认设置1。

\* **@return** 成功返回true, 失败返回false;

\*/

**public** **boolean** initFaceFeatureLib(**int** chlNum)；

4、使用完成后需要释放算法库

/\*\*

\* 释放人脸比对算法库

\*/

**public** **void** releaseFaceFeatureLib()；

四、**算法使用演示**

1、使用步骤如下图：



2、算法基本使用

参考代码：

.MainActivity.java、

3、人脸检测在摄像头中的使用

参考代码：

.DetectActivity.java

4、多线程人脸检测在摄像头中的使用

参考代码：

.DetectTaskActivity.java

5、多线程人脸比对

参考代码：

.FeatureActivity.java

五、图像数据使用说明

1. 该算法是针对彩色人图像人脸识别，不建议用于红外摄像头的图像识别和比对。
2. 在人脸检测算法中使用的是灰度图数据。
3. 在人脸模板获取中使用的是RGB888数据，每个像素占用3个字节，每个色彩使用一个字节，色彩顺序为BGR排列方式。

4、该算法用于ARM芯片Android设备