



## ArcSoft ArcFace SDK

---

开发说明文档

版本历史

SDK 版本	日期	作者	描述
2. 2. 3902010101. 2	06/05/2019	ArcSoft	优化算法库，新增 IR 活体检测、活体检测阈值设置接口

# 目录

目录 .....	3
1. 简介 .....	5
1.1 产品概述 .....	5
1.2 环境要求 .....	5
1.2.1 系统要求 .....	5
1.2.2 开发环境 .....	5
1.2.3 支持的颜色空间格式 .....	5
1.3 产品功能简介 .....	6
1.3.1 人脸检测 .....	6
1.3.2 人脸跟踪 .....	6
1.3.3 人脸属性检测 .....	6
1.3.4 人脸三维角度检测 .....	6
1.3.5 人脸特征提取 .....	6
1.3.6 人脸比对 .....	6
1.3.7 活体检测 .....	7
1.4 SDK 授权说明 .....	7
2. SDK 接入指南 .....	7
2.1 获取 SDK .....	7
2.1.1 注册开发者账号 .....	7
2.1.2 SDK 下载 .....	7
2.1.3 SDK 包结构 .....	8
2.1.4 工程配置 .....	9
2.1.5 调用流程图 .....	10
2.1.6 阈值推荐 .....	10
2.2 数据结构 .....	10
2.2.1 ASF_VERSION .....	10
2.2.2 ASF_ActiveFileInfo .....	11
2.2.3 ASF_SingleFaceInfo .....	11
2.2.4 ASF_MultiFaceInfo .....	11
2.2.5 ASF_FaceFeature .....	12
2.2.6 ASF_AgeInfo .....	12
2.2.7 ASF_GenderInfo .....	12
2.2.8 ASF_Face3DAngle .....	12
2.2.9 ASF_LivenessThreshold .....	13
2.2.10 ASF_LivenessInfo .....	13
2.3 枚举 .....	13
2.3.1 人脸检测方向 .....	13
2.3.2 检测到的人脸角度（按逆时针方向） .....	13
3. 接口 .....	14

3.1	接口说明.....	14
3.1.1	ASFGetActiveFileInfo .....	14
3.1.2	ASFOnlineActivation.....	14
3.1.3	ASFActivation.....	15
3.1.4	ASFInitEngine .....	15
3.1.5	ASFDetectFaces .....	16
3.1.6	ASFFaceFeatureExtract.....	17
3.1.7	ASFFaceFeatureCompare .....	17
3.1.8	ASFSetLivenessParam.....	18
3.1.9	ASFProcess .....	18
3.1.10	ASFProcess_IR .....	19
3.1.11	ASFGetAge.....	20
3.1.12	ASFGetGender.....	20
3.1.13	ASFGetFace3DAngle .....	20
3.1.14	ASFGetLivenessScore.....	21
3.1.15	ASFGetLivenessScore_IR .....	21
3.1.16	ASFGetVersion .....	22
3.1.17	ASFUninitEngine .....	22
3.2	错误码列表.....	22
3.3	示例代码.....	25
4.	常见问题.....	31
4.1	FAQ.....	31
4.2	其他帮助.....	33

# 1.简介

## 1.1 产品概述

ArcFace 离线 SDK，包含人脸检测、性别检测、年龄检测、人脸识别、RGB 活体检测、IR 活体检测等能力，初次使用时需联网激活，激活后即可在本地无网络环境下工作，可根据具体的业务需求结合人脸识别 SDK 灵活地进行应用层开发。

## 1.2 环境要求

### 1.2.1 系统要求

Win7 及以上

### 1.2.2 开发环境

VS2013 及以上

### 1.2.3 支持的颜色空间格式

常量名	常量值	颜色格式说明
ASVL_PAF_NV21	2050	8-bit Y 通道，8-bit 2x2 采样 V 与 U 分量交织通道
ASVL_PAF_NV12	2049	8-bit Y 通道，8-bit 2x2 采样 U 与 V 分量交织通道
ASVL_PAF_RGB24_B8G8R8	513	RGB 分量交织，按 B, G, R, B 字节序排布
ASVL_PAF_I420	1537	8-bit Y 通道，8-bit 2x2 采样 U 通道，8-bit 2x2 采样 V 通道
ASVL_PAF_YUYV	1289	YUV 分量交织，V 与 U 分量 2x1 采样，按 Y0, U0, Y1, V0 字节序排布
ASVL_PAF_GRAY	1793	8-bit IR 图像
ASVL_PAF_DEPTH_U16	3074	16-bit IR 图像

## 1.3 产品功能简介

### 1.3.1 人脸检测

对传入的图像数据进行人脸检测，返回人脸的边框以及朝向信息，可用于后续的人脸识别、活体检测等操作；

支持 image 模式和 video 模式；

支持单人脸、多人脸检测，最多支持检测人脸检测数为 50。

### 1.3.2 人脸跟踪

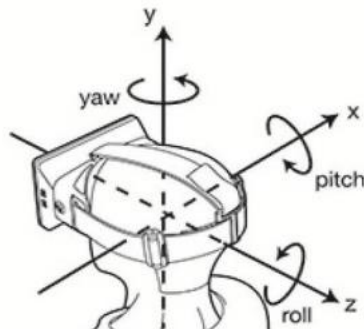
对来自于视频流中的图像数据，进行人脸检测，并对检测到的人脸进行持续跟踪。

### 1.3.3 人脸属性检测

人脸属性分析，支持性别、年龄等。

### 1.3.4 人脸三维角度检测

分析人脸的三维角度信息，具体为：俯仰角(pitch)，横滚角(roll)，偏航角(yaw)。



### 1.3.5 人脸特征提取

提取人脸视觉特征信息。

### 1.3.6 人脸比对

对两个人脸特征数据进行比对，来判断是否为同一个人，返回比对相似度值。

### 1.3.7 活体检测

离线活体检测，静默式识别，在人脸识别过程中判断操作用户是否为真人，有效防御照片、视频、纸张等不同类型的作弊攻击，提高业务安全性，让人脸识别更安全、更快捷，体验更佳。支持单目 RGB 活体检测、双目（IR/RGB）活体检测，可满足各类人脸识别终端产品活体检测应用。

## 1.4 SDK 授权说明

SDK 授权按设备进行授权，每台硬件设备需要一个独立的授权，此授权的校验是基于设备的唯一标识，被授权的设备在初次授权时需在线进行授权，授权成功后可以离线运行 SDK。

#### 在线授权：

- a) 首次激活需保证与公网连通；
- b) 调用在线激活接口激活 SDK；

#### 注意事项：

- a) 设备授权后，若设备授权信息被删除（重装系统/应用被卸载等），需联网重新激活；
- b) 硬件信息发生变更，需要重新激活；

## 2.SDK 接入指南

### 2.1 获取 SDK

#### 2.1.1 注册开发者账号

访问 ArcSoft AI 开放平台门户：<https://ai.arcsoft.com.cn>，注册开发者账号并登录。

#### 2.1.2 SDK 下载

创建应用，添加 SDK。



选择对应平台，确认后即可下载 SDK 和查看激活码。

点击【查看激活码】即可查看所需要 APPID、SDKKEY，点击【下载】获取 SDK 开发包。



### 2.1.3 SDK 包结构

---doc	
---ARCSOFT_ARC_FACE_DEVELOPER'S_GUIDE.PDF	开发说明文档
---inc	
---amcomdef.h	平台文件
---asvloffscreen.h	平台文件
---arcsoft_face_sdk.h	接口文件
---merror.h	错误码文件
---lib	
--- ---Win64	
--- ---libarcsoft_face.dll	算法库
--- ---libarcsoft_face_engine.dll	引擎库
--- ---libarcsoft_face_engine.lib	引擎库
---samplecode	
---samplecode.txt	示例代码
---releasenotes.txt	说明文件

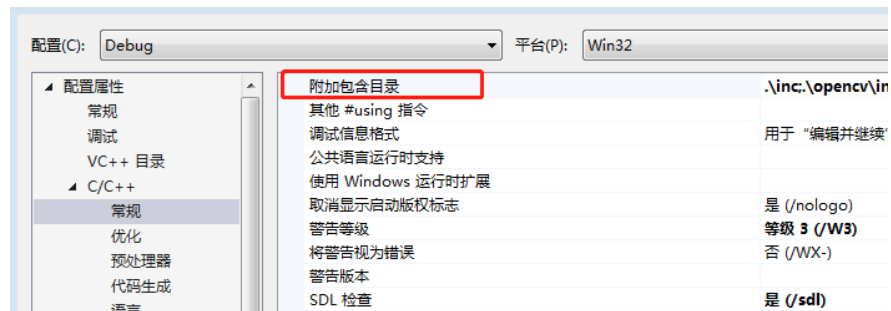


## 2.1.4 工程配置

### VS 配置 SDK 步骤：

#### 1、添加工程的头文件目录：

- a) 右键单击工程名，选择属性---配置属性---C/C++---常规---附加包含目录
- b) 添加头文件存放目录路径

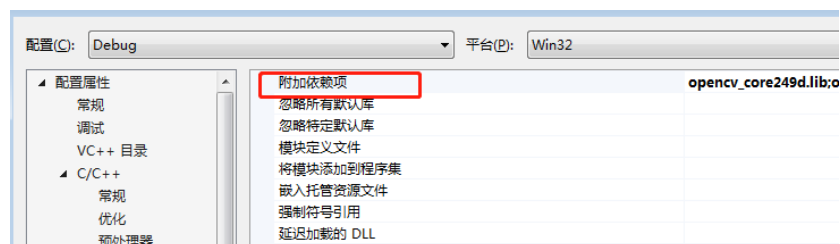


#### 2、添加文件引用的 lib 静态库路径：

- a) 右键单击工程名，选择属性---配置属性---链接器---常规---附加库目录
- b) 添加 lib 文件存放目录

#### 3、添加工程引用的 lib 库：

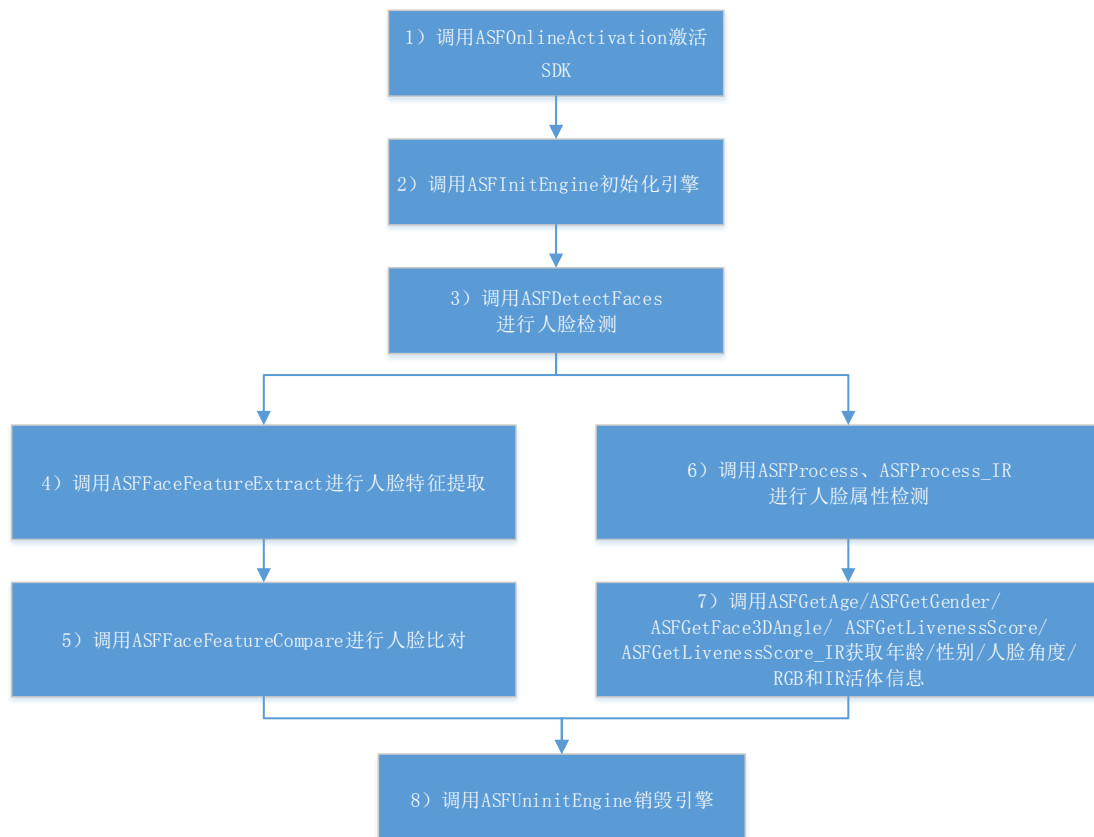
- a) 右键单击工程名，选择属性---配置属性---链接器---输入---附加依赖项
- b) 添加依赖的 lib 库名称



#### 4、添加工程引用的 dll 动态库：

- a) 把引用的 dll 放到工程的可执行文件所在的目录下（例如：debug、release 或工程目录下）

### 2.1.5 调用流程图



### 2.1.6 阈值推荐

- a) 人脸比对阈值,相似度区间为[0~1], 推荐阈值为 0.8;
- b) RGB 活体检测阈值, 设置区间为[0~1], 默认阈值 0.75;
- c) IR 活体检测阈值, 设置区间为[0~1], 默认阈值 0.7;

以上阈值可根据实际使用场景具体调整。

## 2.2 数据结构

### 2.2.1 ASF\_VERSION

结构体描述:

SDK 版本信息

定义:

```
typedef struct {
```

```

        MPChar Version;                // 版本号
        MPChar BuildDate;              // 构建日期
        MPChar Copyright;              // 版权说明
    } ASF_VERSION;

```

### 2.2.2 ASF\_ActiveFileInfo

**结构体描述：**

激活文件信息

**定义：**

```

typedef struct {
    MPChar startTime;                  //SDK 开始时间
    MPChar endTime;                   //SDK 截止时间
    MPChar platform;                  //平台版本
    MPChar sdkType;                   //SDK 类型
    MPChar appId;                     //APPID
    MPChar sdkKey;                     //SDKKEY
    MPChar sdkVersion;                //SDK 版本号
    MPChar fileVersion;               //激活文件版本号
} ASF_ActiveFileInfo, *LPASF_ActiveFileInfo;

```

### 2.2.3 ASF\_SingleFaceInfo

**结构体描述：**

单人脸信息

**定义：**

```

typedef struct {
    MRECT      faceRect;              // 人脸框
    MInt32      faceOrient;           //人脸角度
} ASF_SingleFaceInfo, *LPASF_SingleFaceInfo;

```

### 2.2.4 ASF\_MultiFaceInfo

**结构体描述：**

多人脸信息

**定义：**

```

typedef struct {
    MRECT*      faceRect;              // 人脸框数组
    MInt32*      faceOrient;           // 人脸角度数组
    MInt32      faceNum;               // 检测到的人脸个数
    MInt32*      faceID;               //在 VIDEO 模式下有效，IMAGE 模式下为空
} ASF_MultiFaceInfo, *LPASF_MultiFaceInfo;

```

## 2.2.5 ASF\_FaceFeature

结构体描述:

人脸特征

定义:

```
typedef struct {  
    MByte*      feature;        // 人脸特征  
    MInt32      featureSize;    // 人脸特征长度  
} ASF_FaceFeature, *LPASF_FaceFeature;
```

## 2.2.6 ASF\_AgeInfo

结构体描述:

年龄信息

定义:

```
typedef struct {  
    MInt32* ageArray;    // 0:未知; >0:年龄  
    MInt32  num;        // 检测的人脸个数  
} ASF_AgeInfo, *LPASF_AgeInfo;
```

## 2.2.7 ASF\_GenderInfo

结构体描述:

性别信息

定义:

```
typedef struct {  
    MInt32* genderArray;    // 0:男性; 1:女性; -1:未知  
    MInt32  num;          // 检测的人脸个数  
} ASF_GenderInfo, *LPASF_GenderInfo;
```

## 2.2.8 ASF\_Face3DAngle

结构体描述:

3D 角度信息

定义:

```
typedef struct {  
    MFloat* roll;          // 横滚角  
    MFloat* yaw;           // 偏航角  
    MFloat* pitch;        // 俯仰角  
    MInt32* status;       // 0:正常; 非 0:异常  
    MInt32  num;          // 检测的人脸个数
```

```
}ASF_Face3DAngle, *LPASF_Face3DAngle;
```

## 2.2.9 ASF\_LivenessThreshold

结构体描述：

活体置信度

定义：

```
typedef struct{
    MFloat      thresholdmodel_BGR;          //RGB 活体置信度
    MFloat      thresholdmodel_IR;          //IR 活体置信度
}ASF_LivenessThreshold, *LPASF_LivenessThreshold;
```

## 2.2.10 ASF\_LivenessInfo

结构体描述：

活体信息

定义：

```
typedef struct{
    MInt32* isLive;      //0:非真人； 1:真人； -1: 不确定； -2:传入人脸数>1
    MInt32 num;          //检测的人脸个数
}ASF_LivenessInfo, *LPASF_LivenessInfo;
```

## 2.3 枚举

### 2.3.1 人脸检测方向

根据应用场景，推荐选择单一角度，检测效果更优；

```
enum ArcSoftFace_OrientPriority {
    ASF_OP_0_ONLY = 0x1,          // 仅检测 0 度
    ASF_OP_90_ONLY = 0x2,         // 仅检测 90 度
    ASF_OP_270_ONLY = 0x3,        // 仅检测 270 度
    ASF_OP_180_ONLY = 0x4,        // 仅检测 180 度
    ASF_OP_0_HIGHER_EXT = 0x5,    // 全角度检测
};
```

注：IMAGE 模式下为了提高检测识别率，不支持 ASF\_OP\_0\_HIGHER\_EXT 检测

### 2.3.2 检测到的人脸角度（按逆时针方向）

```
enum ArcSoftFace_OrientCode {
    ASF_OC_0 = 0x1,              // 0 度
    ASF_OC_90 = 0x2,            // 90 度
};
```

```

        ASF_OC_270 = 0x3,        // 270 度
        ASF_OC_180 = 0x4,        // 180 度
        ASF_OC_30  = 0x5,        // 30 度
        ASF_OC_60  = 0x6,        // 60 度
        ASF_OC_120 = 0x7,        // 120 度
        ASF_OC_150 = 0x8,        // 150 度
        ASF_OC_210 = 0x9,        // 210 度
        ASF_OC_240 = 0xa,        // 240 度
        ASF_OC_300 = 0xb,        // 300 度
        ASF_OC_330 = 0xc        // 330 度
};

```

## 3.接口

### 3.1 接口说明

#### 3.1.1 ASFGetActiveFileInfo

接口

```

MRESULT ASFGetActiveFileInfo(
    LPASF_ActiveFileInfo  activeFileInfo
);

```

功能描述

获取激活文件信息

参数

activeFileInfo     [out]     激活文件信息

返回值

成功返回 MOK，失败详见 3.2 错误码列表

#### 3.1.2 ASFOnlineActivation

接口

```

MRESULT ASFOnlineActivation(
    MPChar     AppId,
    MPChar     SDKKey
);

```

功能描述

用于在线激活 SDK

注:

- 1) 初次使用 SDK 时需要对 SDK 先进行激活, 激活后无需重复调用;
- 2) 调用此接口时必须为联网状态, 激活成功后即可离线使用;

#### 参数

AppId	[in]	官网获取的 APPID
SDKKey	[in]	官网获取的 SDKKEY

#### 返回值

成功返回 MOK 或 MERR\_ASF\_ALREADY\_ACTIVATED, 失败详见 3.2 错误码列表

### 3.1.3 ASFActivation

#### 接口

```
MRESULT ASFActivation(  
    MPChar    AppId,  
    MPChar    SDKKey  
);
```

#### 功能描述

在线激活 SDK, 与 ASFOnlineActivation 接口功能相同, 首次需要联网激活, 生成激活文件之后即可进行本地校验。(ASFOnlineActivation 接口做了激活方案的优化, 推荐使用 ASFOnlineActivation 接口)

#### 参数

AppId	[in]	官网获取的 APPID
SDKKey	[in]	官网获取的 SDKKEY

#### 返回值

成功返回 MOK 或 MERR\_ASF\_ALREADY\_ACTIVATED, 失败详见 3.2 错误码列表

### 3.1.4 ASFInitEngine

#### 接口

```
MRESULT ASFInitEngine(  
    MUInt32          detectMode,  
    ASF_OrientPriority detectFaceOrientPriority,  
    MInt32           detectFaceScaleVal,  
    MInt32           detectFaceMaxNum,  
    MInt32           combinedMask,  
    MHandle*         hEngine  
);
```

#### 功能描述

## 初始化引擎

### 参数

detectMode	[in]	VIDEO 模式/IMAGE 模式 VIDEO 模式:处理连续帧的图像数据 IMAGE 模式:处理单张的图像数据
detectFaceOrientPriority	[in]	人脸检测角度，推荐单一角度检测；IMAGE 模式下不支持全角度（ASF_OP_0_HIGHER_EXT）检测
detectFaceScaleVal	[in]	识别的最小人脸比例（图片长边与人脸框长边的比值） VIDEO 模式取值范围[2, 32]，推荐值为 16 IMAGE 模式取值范围[2, 32]，推荐值为 30
detectFaceMaxNum	[in]	最大需要检测的人脸个数，取值范围[1, 50]
combinedMask	[in]	需要启用的功能组合，可多选
hEngine	[out]	引擎句柄

### 返回值

成功返回 MOK，失败详见 3.2 错误码列表

## 3.1.5 ASFDetectFaces

### 接口

```
MRESULT ASFDetectFaces(  
    MHandle          hEngine,  
    MInt32           width,  
    MInt32           height,  
    MInt32           format,  
    MUInt8*          imgData,  
    LPASF_MultiFaceInfo detectedFaces  
);
```

### 功能描述

人脸检测

### 参数

hEngine	[in]	引擎句柄
width	[in]	图片宽度，为 4 的倍数
height	[in]	图片高度，YUYV/I420/NV21/NV12 格式为 2 的倍数； BGR24/GRAY/DEPTH_U16 格式无限制
format	[in]	颜色空间格式
imgData	[in]	图片数据
detectedFaces	[out]	检测到的人脸信息

### 返回值

成功返回 MOK，失败详见 3.2 错误码列表



### 3.1.6 ASFFaceFeatureExtract

#### 接口

```
MRESULT ASFFaceFeatureExtract(  
    MHandle          hEngine,  
    MInt32           width,  
    MInt32           height,  
    MInt32           format,  
    MUInt8*          imgData,  
    LPASF_SingleFaceInfo faceInfo,  
    LPASF_FaceFeature feature  
);
```

#### 功能描述

单人脸特征提取

#### 参数

hEngine	[in]	引擎句柄
width	[in]	图片宽度，为 4 的倍数
height	[in]	图片高度，YUYV/I420/NV21/NV12 格式为 2 的倍数； BGR24/GRAY 格式无限制
format	[in]	颜色空间格式
imgData	[in]	图片数据
faceInfo	[in]	单张人脸信息
feature	[out]	人脸特征

#### 返回值

成功返回 MOK，失败详见 3.2 错误码列表

### 3.1.7 ASFFaceFeatureCompare

#### 接口

```
MRESULT ASFFaceFeatureCompare(  
    MHandle          hEngine,  
    LPASF_FaceFeature feature1,  
    LPASF_FaceFeature feature2,  
    MFloat*          confidenceLevel  
);
```

#### 功能描述

人脸特征比对

#### 参数

hEngine	[in]	引擎句柄
---------	------	------

feature1	[in]	人脸特征值
feature2	[in]	人脸特征值
confidenceLevel	[out]	比对结果, 相似度

#### 返回值

成功返回 MOK, 失败详见 3.2 错误码列表

### 3.1.8 ASFSetLivenessParam

#### 接口

```
MRESULT ASFSetLivenessParam(
    MHandle          hEngine,
    LPASF_LivenessThreshold threshold
);
```

#### 功能描述

修改 RGB/IR 活体阈值, SDK 默认 RGB: 0.75, IR: 0.7

#### 参数

hEngine	[in]	引擎句柄
threshold	[in]	活体置信度, 推荐阈值 RGB:0.75, IR:0.7

#### 返回值

成功返回 MOK, 失败详见 3.2 错误码列表

### 3.1.9 ASFProcess

#### 接口

```
MRESULT ASFProcess(
    MHandle          hEngine,
    MInt32           width,
    MInt32           height,
    MInt32           format,
    MUInt8*          imgData,
    LPASF_MultiFaceInfo detectedFaces,
    MInt32           combinedMask
);
```

#### 功能描述

人脸信息检测 (年龄/性别/人脸 3D 角度), 最多支持 4 张人脸信息检测, 超过部分返回未知 (活体仅支持单张人脸检测, 超出返回未知), 接口不支持 IR 图像检测。

#### 参数

hEngine	[in]	引擎句柄
width	[in]	图片宽度, 为 4 的倍数

height	[in]	图片高度，YUYV/I420/NV21/NV12 格式为 2 的倍数，BGR24 格式无限制
format	[in]	颜色空间格式
imgData	[in]	图片数据
detectedFaces	[in]	检测到的人脸信息
combinedMask	[in]	检测的属性（ASF_AGE、ASF_GENDER、ASF_FACE3DANGLE、ASF_LIVENESS），支持多选 注：检测的属性须在引擎初始化接口的 combinedMask 参数中启用

## 返回值

成功返回 MOK，失败详见 3.2 错误码列表

## 3.1.10 ASFProcess\_IR

### 接口

```
MRESULT ASFProcess_IR(
    MHandle          hEngine,
    MInt32           width,
    MInt32           height,
    MInt32           format,
    MUInt8*          imgData,
    LPASF_MultiFaceInfo detectedFaces,
    MInt32           combinedMask
);
```

### 功能描述

IR 活体单人脸检测

### 参数

hEngine	[in]	引擎句柄
width	[in]	图片宽度，为 4 的倍数
height	[in]	图片高度，I420/NV21/NV12 格式为 2 的倍数，DEPTH_U16/GRAY 格式无限制
format	[in]	颜色空间格式，支持（I420/NV21/NV12/DEPTH_U16/GRAY）的检测
imgData	[in]	图片数据
detectedFaces	[in]	人脸信息，用户根据待检测的功能选择需要使用的人脸。
combinedMask	[in]	检测的属性（ASF_IR_LIVENESS） 注：检测的属性须在引擎初始化接口的 combinedMask 参数中启用

## 返回值

成功返回 MOK，失败详见 3.2 错误码列表

### 3.1.11 ASFGetAge

#### 接口

```
MRESULT ASFGetAge(  
    MHandle          hEngine,  
    LPASF_AgeInfo    ageInfo  
);
```

#### 功能描述

获取年龄信息

#### 参数

hEngine	[in]	引擎句柄
ageInfo	[out]	检测到的年龄信息

#### 返回值

成功返回 MOK，失败详见 3.2 错误码列表

### 3.1.12 ASFGetGender

#### 接口

```
MRESULT ASFGetGender(  
    MHandle          hEngine,  
    LPASF_GenderInfo genderInfo  
);
```

#### 功能描述

获取性别信息

#### 参数

hEngine	[in]	引擎句柄
genderInfo	[out]	检测到的性别信息

#### 返回值

成功返回 MOK，失败详见 3.2 错误码列表

### 3.1.13 ASFGetFace3DAngle

#### 接口

```
MRESULT ASFGetFace3DAngle(  
    MHandle          hEngine,  
    LPASF_Face3DAngle p3DAngleInfo  
);
```

## 功能描述

获取 3D 角度信息

## 参数

hEngine	[in]	引擎句柄
p3DAngleInfo	[out]	检测到脸部 3D 角度信息

## 返回值

成功返回 MOK，失败详见 3.2 错误码列表

## 3.1.14 ASFGetLivenessScore

### 接口

```
MRESULT ASFGetLivenessScore(  
    MHandle          hEngine,  
    LPASF_LivenessInfo  livenessInfo  
);
```

## 功能描述

获取 RGB 活体信息

## 参数

hEngine	[in]	引擎句柄
livenessInfo	[out]	RGB 活体信息，详见 2.2.10 ASF_LivenessInfo

## 返回值

成功返回 MOK，失败详见 3.2 错误码列表

## 3.1.15 ASFGetLivenessScore\_IR

### 接口

```
MRESULT ASFGetLivenessScore_IR(  
    MHandle          hEngine,  
    LPASF_LivenessInfo  livenessInfo  
);
```

## 功能描述

获取 IR 活体信息

## 参数

hEngine	[in]	引擎句柄
livenessInfo	[out]	IR 活体信息，详见 2.2.10 ASF_LivenessInfo

## 返回值

成功返回 MOK，失败详见 3.2 错误码列表

3.1.16 ASFGetVersion

接口

```
const ASF_VERSION* ASFGetVersion(  
    MHandle    hEngine  
);
```

功能描述

获取版本信息

参数

hEngine [in] 引擎句柄

返回值

成功返回版本信息，失败返回 MNull。

3.1.17 ASFUninitEngine

接口

```
MRESULT ASFUninitEngine(  
    MHandle hEngine  
);
```

功能描述

销毁引擎

参数

hEngine [in] 引擎句柄

返回值

成功返回 MOK，失败详见 3.2 错误码列表

3.2 错误码列表

错误码名	十六进制	十进制	错误码说明
MOK	0x0	0	成功
MERR_UNKNOWN	0x1	1	错误原因不明
MERR_INVALID_PARAM	0x2	2	无效的参数
MERR_UNSUPPORTED	0x3	3	引擎不支持
MERR_NO_MEMORY	0x4	4	内存不足
MERR_BAD_STATE	0x5	5	状态错误
MERR_USER_CANCEL	0x6	6	用户取消相关操作
MERR_EXPIRED	0x7	7	操作时间过期

MERR_USER_PAUSE	0x8	8	用户暂停操作
MERR_BUFFER_OVERFLOW	0x9	9	缓冲上溢
MERR_BUFFER_UNDERFLOW	0xA	10	缓冲下溢
MERR_NO_DISKSPACE	0xB	11	存贮空间不足
MERR_COMPONENT_NOT_EXIST	0xC	12	组件不存在
MERR_GLOBAL_DATA_NOT_EXIST	0xD	13	全局数据不存在
MERR_FSDK_INVALID_APP_ID	0x7001	28673	无效的 AppId
MERR_FSDK_INVALID_SDK_ID	0x7002	28674	无效的 SDKkey
MERR_FSDK_INVALID_ID_PAIR	0x7003	28675	AppId 和 SDKKey 不匹配
MERR_FSDK_MISMATCH_ID_AND_SDK	0x7004	28676	SDKKey 和使用的 SDK 不匹配（注意：调用初始化引擎接口时，请确认激活接口传入的参数，并重新激活）
MERR_FSDK_SYSTEM_VERSION_UNSUPPORTED	0x7005	28677	系统版本不被当前 SDK 所支持
MERR_FSDK_LICENCE_EXPIRED	0x7006	28678	SDK 有效期过期，需要重新下载更新
MERR_FSDK_FR_INVALID_MEMORY_INFO	0x12001	73729	无效的输入内存
MERR_FSDK_FR_INVALID_IMAGE_INFO	0x12002	73730	无效的输入图像参数
MERR_FSDK_FR_INVALID_FACE_INFO	0x12003	73731	无效的脸部信息
MERR_FSDK_FR_MISMATCHED_FEATURE_LEVEL	0x12005	73733	待比较的两个人脸特征的版本不一致
MERR_FSDK_FACEFEATURE_UNKNOWN	0x14001	81921	人脸特征检测错误未知
MERR_FSDK_FACEFEATURE_MEMORY	0x14002	81922	人脸特征检测内存错误
MERR_FSDK_FACEFEATURE_INVALID_FORMAT	0x14003	81923	人脸特征检测格式错误
MERR_FSDK_FACEFEATURE_INVALID_PARAM	0x14004	81924	人脸特征检测参数错误
MERR_FSDK_FACEFEATURE_LOW_CONFIDENCE_LEVEL	0x14005	81925	人脸特征检测结果置信度低
MERR_ASF_EX_FEATURE_UNSUPPORTED_ON_INIT	0x15001	86017	Engine 不支持的检测属性
MERR_ASF_EX_FEATURE_UNINITED	0x15002	86018	需要检测的属性未初始化
MERR_ASF_EX_FEATURE_UNPROCESSED	0x15003	86019	待获取的属性未在 process 中处理过
MERR_ASF_EX_FEATURE_UNSUPPORTED_ON_PROCESS	0x15004	86020	PROCESS 不支持的检测属性，例如 FR，有自己独立的处理函数
MERR_ASF_EX_INVALID_IMAGE_INFO	0x15005	86021	无效的输入图像

MERR_ASF_EX_INVALID_FACE_INFO	0x15006	86022	无效的脸部信息
MERR_ASF_ACTIVATION_FAIL	0x16001	90113	SDK 激活失败，请打开读写权限
MERR_ASF_ALREADY_ACTIVATED	0x16002	90114	SDK 已激活
MERR_ASF_NOT_ACTIVATED	0x16003	90115	SDK 未激活
MERR_ASF_SCALE_NOT_SUPPORT	0x16004	90116	detectFaceScaleVal 不支持
MERR_ASF_ACTIVEFILE_SDKTYPE_MISMATCH	0x16005	90117	激活文件与 SDK 类型不匹配，请确认使用的 sdk
MERR_ASF_DEVICE_MISMATCH	0x16006	90118	设备不匹配
MERR_ASF_UNIQUE_IDENTIFIER_ILLEGAL	0x16007	90119	唯一标识不合法
MERR_ASF_PARAM_NULL	0x16008	90120	参数为空
MERR_ASF_VERSION_NOT_SUPPORT	0x1600A	90122	版本不支持
MERR_ASF_SIGN_ERROR	0x1600B	90123	签名错误
MERR_ASF_DATABASE_ERROR	0x1600C	90124	激活信息保存异常
MERR_ASF_UNIQUE_CHECKOUT_FAIL	0x1600D	90125	唯一标识符校验失败
MERR_ASF_COLOR_SPACE_NOT_SUPPORT	0x1600E	90126	颜色空间不支持
MERR_ASF_IMAGE_WIDTH_HEIGHT_NOT_SUPPORT	0x1600F	90127	图片宽高不支持，宽度需四字节对齐
MERR_ASF_ACTIVATION_DATA_DESTROYED	0x16011	90129	激活数据被破坏，请删除激活文件，重新进行激活
MERR_ASF_SERVER_UNKNOWN_ERROR	0x16012	90130	服务端未知错误
MERR_ASF_ACTIVEFILE_SDK_MISMATCH	0x16014	90132	激活文件与 SDK 版本不匹配，请重新激活
MERR_ASF_DEVICEINFO_LESS	0x16015	90133	设备信息太少，不足以生成设备指纹
MERR_ASF_REQUEST_TIMEOUT	0x16016	90134	客户端时间与服务器时间(即北京时间)前后相差在 30 分钟以上
MERR_ASF_APPID_DATA_DECRYPT	0x16017	90135	数据校验异常
MERR_ASF_APPID_APPKEY_SDK_MISMATCH	0x16018	90136	传入的 AppId 和 AppKey 与使用的 SDK 版本不一致
MERR_ASF_NO_REQUEST	0x16019	90137	短时间大量请求会被禁止请求, 30 分钟之后解封
MERR_ASF_ACTIVE_FILE_NO_EXIST	0x1601A	90138	激活文件不存在
MERR_ASF_IMAGEMODE_0_HIGHER_EXT_UNsupport	0x1601B	90139	IMAGE 模式下不支持全角度 (ASF_OP_0_HIGHER_EXT) 检测
MERR_ASF_NETWORK_COULDNT_RESOLVE_HOST	0x17001	94209	无法解析主机地址
MERR_ASF_NETWORK_COULDNT_CONNECT	0x17002	94210	无法连接服务器



NECT_SERVER			
MERR_ASF_NETWORK_CONNECT_TIMEOUT	0x17003	94211	网络连接超时
MERR_ASF_NETWORK_UNKNOWN_ERROR	0x17004	94212	网络未知错误

### 3.3 示例代码

```

#include "stdafx.h"
#include <stdio.h>
#include <stdlib.h>
#include "arcsoft_face_sdk.h"
#include "amcomdef.h"
#include "asvloffscreen.h"
#include "merror.h"
#include <direct.h>
#include <iostream>
#include <stdarg.h>
#include <string>
#include <opencv2\opencv.hpp>

using namespace std;
#pragma comment(lib, "libarcsoft_face_engine.lib")

#define APPID "官网获取APPID"
#define SDKKey "官网获取ADKKey"

#define SafeFree(p) { if ((p)) free(p); (p) = NULL; }
#define SafeArrayDelete(p) { if ((p)) delete [] (p); (p) = NULL; }
#define SafeDelete(p) { if ((p)) delete (p); (p) = NULL; }

//裁剪图片,做四字节对齐
void CutIplImage(IplImage* src, IplImage* dst, int x, int y)
{
    CvSize size = cvSize(dst->width, dst->height); //区域大小
    cvSetImageROI(src, cvRect(x, y, size.width, size.height)); //设置源图像ROI
    cvCopy(src, dst); //复制图像
    cvResetImageROI(src); //源图像用完后,清空ROI
}

```

```

int main()
{
    //激活接口, 首次需联网激活
    MRESULT res = ASFOnlineActivation(APPID, SDKKey);
    //MRESULT res = ASFActivation(APPID, SDKKey);
    if (MOK != res && MERR_ASF_ALREADY_ACTIVATED != res)
        printf("ASFActivation fail: %d\n", res);
    else
        printf("ASFActivation sucess: %d\n", res);

    //获取激活文件信息
    ASF_ActiveFileInfo activeFileInfo;
    res = ASFGetActiveFileInfo(&activeFileInfo);
    if (res != MOK)
        printf("ASFGetActiveFileInfo fail: %d\n", res);

    //初始化接口
    MHandle handle = NULL;
    MInt32 mask = ASF_FACE_DETECT | ASF_FACERECOGNITION | ASF_AGE | ASF_GENDER |
ASF_FACE3DANGLE | ASF_LIVENESS | ASF_IR_LIVENESS;
    res = ASFInitEngine(ASF_DETECT_MODE_IMAGE, ASF_OP_0_ONLY, 30, 10, mask, &handle);
    if (res != MOK)
        printf("ASFInitEngine fail: %d\n", res);
    else
        printf("ASFInitEngine sucess: %d\n", res);

    // 人脸检测
    IplImage* img1 = cvLoadImage("RGB图像路径");
    IplImage* img2 = cvLoadImage("RGB图像路径");
    IplImage* img3 = cvLoadImage("IR图像路径");

    if (img1 && img2 && img3)
    {
        ASF_MultiFaceInfo detectedFaces1 = { 0 };
        ASF_SingleFaceInfo SingleDetectedFaces1 = { 0 };
        ASF_FaceFeature feature1 = { 0 };
        ASF_FaceFeature copyfeature1 = { 0 };
        IplImage* cutImg1 = cvCreateImage(cvSize(img1->width - img1->width % 4,
img1->height), IPL_DEPTH_8U, img1->nChannels);
        CutIplImage(img1, cutImg1, 0, 0);

        res = ASFDetectFaces(handle, cutImg1->width, cutImg1->height,
ASVL_PAF_RGB24_B8G8R8, (MUInt8*)cutImg1->imageData, &detectedFaces1);
    }
}

```

```

if (MOK == res)
{
    SingleDetectedFaces1.faceRect.left = detectedFaces1.faceRect[0].left;
    SingleDetectedFaces1.faceRect.top = detectedFaces1.faceRect[0].top;
    SingleDetectedFaces1.faceRect.right = detectedFaces1.faceRect[0].right;
    SingleDetectedFaces1.faceRect.bottom = detectedFaces1.faceRect[0].bottom;
    SingleDetectedFaces1.faceOrient = detectedFaces1.faceOrient[0];

    res = ASFFaceFeatureExtract(handle, cutImg1->width, cutImg1->height,
ASVL_PAF_RGB24_B8G8R8, (MUInt8*)cutImg1->imageData, &SingleDetectedFaces1, &feature1);
    if (res == MOK)
    {
        //拷贝feature
        copyfeature1.featureSize = feature1.featureSize;
        copyfeature1.feature = (MByte *)malloc(feature1.featureSize);
        memset(copyfeature1.feature, 0, feature1.featureSize);
        memcpy(copyfeature1.feature, feature1.feature, feature1.featureSize);
    }
    else
        printf("ASFFaceFeatureExtract 1 fail: %d\n", res);
}
else
    printf("ASFDetectFaces 1 fail: %d\n", res);

//第二张人脸处理
ASF_MultiFaceInfo detectedFaces2 = { 0 };
ASF_SingleFaceInfo SingleDetectedFaces2 = { 0 };
ASF_FaceFeature feature2 = { 0 };
IplImage* cutImg2 = cvCreateImage(cvSize(img2->width - img2->width % 4,
img2->height), IPL_DEPTH_8U, img2->nChannels);
CutIplImage(img2, cutImg2, 0, 0);
res = ASFDetectFaces(handle, cutImg2->width, cutImg2->height,
ASVL_PAF_RGB24_B8G8R8, (MUInt8*)cutImg2->imageData, &detectedFaces2);
if (MOK == res)
{
    SingleDetectedFaces2.faceRect.left = detectedFaces2.faceRect[0].left;
    SingleDetectedFaces2.faceRect.top = detectedFaces2.faceRect[0].top;
    SingleDetectedFaces2.faceRect.right = detectedFaces2.faceRect[0].right;
    SingleDetectedFaces2.faceRect.bottom = detectedFaces2.faceRect[0].bottom;
    SingleDetectedFaces2.faceOrient = detectedFaces2.faceOrient[0];

    res = ASFFaceFeatureExtract(handle, cutImg2->width, cutImg2->height,

```

```

ASVL_PAF_RGB24_B8G8R8, (MUInt8*)cutImg2->imageData, &SingleDetectedFaces2, &feature2);
    if (MOK != res)
        printf("ASFFaceFeatureExtract 2 fail: %d\n", res);
}
else
    printf("ASFDetectFaces 2 fail: %d\n", res);

// 单人脸特征比对
MFloat confidenceLevel;
res = ASFFaceFeatureCompare(handle, &copyfeature1, &feature2, &confidenceLevel);
if (res != MOK)
    printf("ASFFaceFeatureCompare fail: %d\n", res);
else
    printf("ASFFaceFeatureCompare sucess: %lf\n", confidenceLevel);

//设置活体置信度 SDK内部默认值为 IR: 0.7 RGB: 0.75 (无特殊需要, 可以不设置)
ASF_LivenessThreshold threshold = { 0 };
threshold.thresholdmodel_BGR = 0.75;
threshold.thresholdmodel_IR = 0.7;
res = ASFSetLivenessParam(handle, &threshold);
if (res != MOK)
    printf("ASFSetLivenessParam fail: %d\n", res);
else
    printf("RGB Threshold: %f IR Threshold: %f\n", threshold.thresholdmodel_BGR,
threshold.thresholdmodel_IR);

// RGB图像属性检测
MInt32 processMask = ASF_AGE | ASF_GENDER | ASF_FACE3DANGLE | ASF_LIVENESS;
res = ASFProcess(handle, cutImg2->width, cutImg2->height, ASVL_PAF_RGB24_B8G8R8,
(MUInt8*)cutImg2->imageData, &detectedFaces2, processMask);
if (res != MOK)
    printf("ASFProcess fail: %d\n", res);
else
    printf("ASFProcess sucess: %d\n", res);

// 获取年龄
ASF_AgeInfo ageInfo = { 0 };
res = ASFGetAge(handle, &ageInfo);
if (res != MOK)
    printf("ASFGetAge fail: %d\n", res);
else
    printf("Age: %d\n", ageInfo.ageArray[0]);

```

```

// 获取性别
ASF_GenderInfo genderInfo = { 0 };
res = ASFGetGender(handle, &genderInfo);
if (res != MOK)
    printf("ASFGetGender fail: %d\n", res);
else
    printf("Gender: %d\n", genderInfo.genderArray[0]);

// 获取3D角度
ASF_Face3DAngle angleInfo = { 0 };
res = ASFGetFace3DAngle(handle, &angleInfo);
if (res != MOK)
    printf("ASFGetFace3DAngle fail: %d\n", res);
else
    printf("3DAngle roll: %f yaw: %f pitch: %f\n", angleInfo.roll[0],
angleInfo.yaw[0], angleInfo.pitch[0]);

//获取RGB活体信息
ASF_LivenessInfo rgbLivenessInfo = { 0 };
res = ASFGetLivenessScore(handle, &rgbLivenessInfo);
if (res != MOK)
    printf("ASFGetLivenessScore fail: %d\n", res);
else
    printf("RGB Liveness: %d\n", rgbLivenessInfo.isLive[0]);

//*****进行IR活体检测*****
printf("\n*****IR LIVENESS*****\n");

ASF_MultiFaceInfo detectedFaces3 = { 0 };
IplImage* cutImg3 = cvCreateImage(cvSize(img3->width - img3->width % 4,
img3->height), IPL_DEPTH_8U, img3->nChannels);
CutIplImage(img3, cutImg3, 0, 0);

//opencv读图时会把灰度图读成RGB图，需要转换成GRAY图进行IR活体检测
cv::Mat grayMat;
cv::Mat copyCutImg3(cutImg3, false); //IplImage转Mat 设为ture为深拷贝
cv::cvtColor(copyCutImg3, grayMat, CV_BGR2GRAY);

res = ASFDetectFaces(handle, cutImg3->width, cutImg3->height,
ASVL_PAF_RGB24_B8G8R8, (MUInt8*)cutImg3->imageData, &detectedFaces3);
if (res != MOK)
    printf("ASFGetLivenessScore fail: %d\n", res);
else
    printf("Face num: %d\n", detectedFaces3.faceNum);

```

```

//IR图像活体检测
MInt32 processIRMask = ASF_IR_LIVENESS;
res = ASFProcess_IR(handle, grayMat.cols, grayMat.rows, ASVL_PAF_GRAY,
(MUInt8*)grayMat.data, &detectedFaces3, processIRMask);
if (res != MOK)
    printf("ASFProcess_IR fail: %d\n", res);
else
    printf("ASFProcess_IR sucess: %d\n", res);

//获取RGB活体信息
ASF_LivenessInfo irLivenessInfo = { 0 };
res = ASFGetLivenessScore_IR(handle, &irLivenessInfo);
if (res != MOK)
    printf("ASFGetLivenessScore_IR fail: %d\n", res);
else
    printf("IR Liveness: %d\n", irLivenessInfo.isLive[0]);

SafeFree(copyfeature1.feature);          //释放内存
cvReleaseImage(&cutImg1);
cvReleaseImage(&cutImg2);
cvReleaseImage(&cutImg3);
}
cvReleaseImage(&img1);
cvReleaseImage(&img2);
cvReleaseImage(&img3);

//获取版本信息
const ASF_VERSION* pVersionInfo = ASFGetVersion(handle);

//反初始化
res = ASFUinitEngine(handle);
if (res != MOK)
    printf("ALUinitEngine fail: %d\n", res);
else
    printf("ALUinitEngine sucess: %d\n", res);

getchar();
return 0;

```

## 4.常见问题

### 4.1 FAQ

**Q: 如何将人脸识别 1:1 进行开发改为 1:n?**

A: 先将人脸特征数据用本地文件、数据库或者其他的方式存储下来，若检测出结果需要显示图像可以保存对应的图像。之后循环对特征值进行对比，相似度最高者若超过您设置的阈值则输出相关信息。

**Q: 初始化引擎时检测方向应该怎么选择?**

A: SDK 初始化引擎中可选择仅对 0 度、90 度、180 度、270 度单角度进行人脸检测，对于 video 模式也可选择全角度进行检测；根据应用场景，推荐使用单角度进行人脸检测，因为选择全角度的情况下，算法中会对每个角度检测一遍，导致性能相对于单角度较慢。image 模式下为提高识别率不支持全角度检测。

**Q: 初始化引擎时（detectFaceScaleVal）参数多大比较合适?**

A: 用于数值化表示的最小人脸尺寸，该尺寸代表人脸尺寸相对于图片长边的占比。VIDEO 模式有效值范围[2,32]，推荐值为 16；IMAGE 模式有效值范围[2,32]，推荐值为 30，特殊情况下可根据具体场景进行设置。

**Q: 初始化引擎之后调用其他接口返回错误码 86018，该怎么解决?**

A: 86018 即需要检测的属性未初始化，需要查看调用接口的属性值有没有在初始化引擎时在 combinedMask 参数中加入。

**Q: 调用 detectFaces、extractFaceFeature 和 process 接口返回 90127 错误码，该怎么解决?**

A: ArcFace SDK 对图像尺寸做了限制，宽度为 4 的倍数，YUYV/I420/NV21/NV12 格式的图片高度为 2 的倍数，BGR24/GRAY/U16 格式的图片高度不限制；如果遇到 90127 请检查传入的图片尺寸是否符合要求，若不符合可对图片进行适当的裁剪。

**Q: 人脸检测结果的人脸框 Rect 为何有时会溢出传入图像的边界?**

A: Rect 溢出边界可能是人脸只有一部分在图像中, 算法会对人脸的位置进行估计。

**Q: 为何调用引擎有时会出现 crash?**

A: 若在引擎调用过程中进行销毁引擎则可能会导致 crash。在使用过程中应避免在销毁引擎时还在使用引擎, 尤其是做特征提取或活体检测等耗时操作时销毁引擎, 如加锁解决。

**Q: MERR\_FSDK\_FACEFEATURE\_LOW\_CONFIDENCE\_LEVEL, 人脸检测结果置信度低是什么情况导致的?**

A: 图片模糊或者传入的人脸框不正确。若是使用双目摄像头, 则很有可能是两者成像差距很大或两者画面成镜像或旋转的关系。

**Q: 哪些因素会影响人脸检测、人脸跟踪、人脸特征提取等 SDK 调用所用时间?**

A: 硬件性能、图片质量等。

**Q: 如何进行 IR 活体检测?**

A: 推荐的方案采用双目 (RGB/ IR) 摄像头, RGB 摄像头数据用于人脸检测, 将人脸检测的结果用于 IR 活体检测。需要注意的是, IR 活体检测不支持 BGR24 和 YUYV 颜色空间的图像数据。

**Q: 初始化引擎时, 传入检测模式为 Image 模式, 检测方向为全角度, 为何会创建失败?**

A: 为了提高人脸检测识别率, 该版本的 IMAGE 模式不支持全角度检测, 开发者需要根据图像中的人脸方向确定人脸检测的角度。

**Q: Windows 版本在进行 Image 模式人脸检测时, 回传的 faceId 数组为何为空?**

A: faceId 属性是 video 模式下的特有属性, 在一个人脸进入画面到离开, 该值不会改变, image 模式下不支持 faceId。



**Q:** 工程移植为何会报各种依赖库的错误？

**A:** 在 VS 工程中常常要设置头文件的包含路径和库路径，如果使用绝对路径，拷贝工程到其他路径下会导致无法找到依赖库，建议尽量配置相对路径。VS 相对路径以工程文件（XXXX.vcproj）为起点。

## 4.2 其他帮助

SDK 交流论坛：<https://ai.arcsoft.com.cn/bbs/>