4.1.1 SDK 初始化

函数声明

int PFD_Init(int imgSize)

参数

int	imgSize	内部处理图像的标准大小

返回值

PFD_STATUS_OK	正常结束
PFD_STATUS_NG	异常结束
PFD_STATUS_INVALID	验证失败

注： 1、所有函数的位图数据必须是 24 位 RGB 格式的。

2、imgSize 表示库内部处理图像的标准大小，可选值为 IMG_SIZE_VGA 和 IMG_SIZE_FULLHD。

(1)设置为 IMG_SIZE_VGA 的相关说明：

在内部会将图像变为 640*480 大小，影响范围是识别最小人脸宽度相对较大，但识别速度最快。

宽度计算公式是

最小可识别人脸宽度=MAX(MAX(图像宽度/640，图像高度/480)*20, 20)

(2)设置为 IMG_SIZE_FULLHD 的相关说明：

在内部会将图像变为 640*480 大小，影响范围是识别最小人脸宽度会变小，但相比 IMG_SIZE_VGA 识别速度变慢。

宽度计算公式是

最小可识别人脸宽度=MAX(MAX(图像宽度/1920，图像高度/1080)*20, 20)

4.1.2 获取图片内所有人脸的基本信息

函数声明

int PFD_FaceRecog(unsigned char* bmpData,
PFD_FACE_DETECT* faceInfo,
int faceInfoFlag,
short faceRote)

参数

unsigned char*	bmpData	位图数据
PFD_FACE_DETECT	faceInfo	人数和人脸位置信息
int	faceInfoFlag	是否识别年龄、性别、表情、角度等信息
short	faceRote	输入图像的人脸角度信息

返回值

PFD_STATUS_OK	正常结束
PFD_STATUS_NG	异常结束
PFD_STATUS_INVALID	验证失败

注意

1、人脸坐标结构体中的 rotate 是输入参数，需要明确输入图片是向右 0、90、180 还是 270 度。

2、左眼右眼定义：rb 表示右眼，lb 表示左眼，顺时针旋转相对位置不变。

人数和人脸信息结构体：

```
/*  人脸坐标结构体  */
typedef struct _pfd_face_position {
    short       conf;     /*  识别信赖度(0~1000  低~高) */
    short       rect_l;   /*  人脸最左侧坐标*/
    short       rect_r;   /*人脸最右侧坐标*/
    short       rect_t;   /*人脸最上侧坐标*/
    short       rect_b;   /*人脸最下侧坐标*/
    short       eye_lx;   /*人脸左眼x坐标*/
```

```
    short       eye_ly;   /*人脸左眼y坐标*/
    short       eye_rx;   /*人脸右眼x坐标*/
    short       eye_ry;   /*人脸右眼y坐标*/
} PFD_FACE_POSITION;

/*  人脸情报保存结构体  */
typedef struct _pfd_detect_info{
        PFD_FACE_POSITION faceInfo; /*  人脸坐标结构体  */
        short ageConf;              /*  年龄识别可信度  (0~1000  低~高) */
        short genConf;              /*  性别识别可信度  (0~1000  低~高) */
        short age;                  /*  年龄  */
        short gen;                  /*  性别  */
        short smile;                /*  笑脸程度(0~100  低~高)*/
        short pitch;                /*  抬头低头的角度(-180~180度，抬头是正值)*/
        short yaw;                  /*  摇头角度(-180~180度，向左摇头是正值)*/
        short roll;                 /*  倾斜角度(-180~180度，顺时针是正值)*/
        short lb;                   /*  左眼闭眼概率，数值越大代表眼睛越大*/
        short rb;                   /*  右眼闭眼概率，数值越大代表眼睛越大*/
        short flen;                 /*  特征值长度  */
}PFD_DETECT_INFO;

/*  图像内的人脸信息结构体  */
typedef struct _pfd_face_detect {
    short       num;    /*  图像中人脸数(最大:PFD_MAX_FACE_NUM) */
    PFD_DETECT_INFO  info[PFD_MAX_FACE_NUM];/*  人脸的位置和情报信息  */
} PFD_FACE_DETECT;

#define   PFD_MAX_FACE_NUM      （20）

faceRote:
#define   PFD_OP_FACE_ROLL_0     (0x10)   /*  人脸向上的场合(无旋转) */
#define   PFD_OP_FACE_ROLL_90    (0x20)   /*  人脸向右的场合(逆时针旋转90度) */
#define   PFD_OP_FACE_ROLL_180   (0x40)   /*  人脸向下的场合(倒置) */
#define   PFD_OP_FACE_ROLL_270   (0x80)   /*  人脸向左的场合(顺时针旋转 90 度) */

gen:
#define PFD_MALE          (0)         /* 男性   */
#define PFD_FEMALE        (1)         /* 女性   */
```

Age：

年龄的返回值	可能的年龄范围
3	0 到 5 岁
8	6 到 10 岁
13	11 到 15 岁
18	16 到 20 岁
23	21 到 25 岁
28	26 到 30 岁
33	31 到 35 岁
38	36 到 40 岁
43	41 到 45 岁
48	46 到 50 岁
53	51 到 55 岁
58	56 到 60 岁
63	61 到 65 岁
68	66 到 70 岁

faceInfoFlag:

```
#define PFD_ENABLEINFO    (1)        /* 识别性别、年龄、表情、角度、闭眼等信息 */
#define PFD_DISABLEINFO   (0)        /* 不识别性别、年龄、表情、角度、闭眼等信息  */
```

4.1.3 年龄性别识别功能

使用图片作为参数使用年龄性别识别函数。

函数声明：

```
int PFD_AgrRecogImg(unsigned char* bmpData,
                    PFD_AGR_DETECT agrInfo,
                    Short faceRote)
```

参数：

unsigned char*	bmpData	位图数据
PFD_AGR_DETECT*	agrInfo	年龄和性别信息
short	faceRote	输入图像的人脸角度信息

返回值：

PFD_STATUS_OK	正常结束
PFD_STATUS_NG	异常结束
PFD_STATUS_INVALID	验证失败

年龄和性别信息结构体

```
/*  年龄性别结构体*/
typedef struct _pfd_agr_info {
    PFD_FACE_POSITION faceInfo;   /*  人脸坐标结构体  */
    short          ageConf;/*  年龄识别信赖度(0~1000  低~高) */
    short          genConf;/*  性别识别信赖度(0~1000  低~高) */
    short          age; /*  年龄*/
    short          gen; /*  性别*/
} PFD_AGR_INFO;

typedef struct _pfd_agr_detect {
    short          num;     /*  图片中人脸数(最大:PFD_MAX_FACE_NUM) */
    PFD_AGR_INFO       info[PFD_MAX_FACE_NUM];       /*年龄性别*/
} PFD_AGR_DETECT;
```

gen:

```
#define PFD_MALE          (0)          /* 男性   */
#define PFD_FEMALE        (1)          /* 女性   */
```

Age：

年龄的返回值	可能的年龄范围
3	0 到 5 岁
8	6 到 10 岁
13	11 到 15 岁
18	16 到 20 岁
23	21 到 25 岁
28	26 到 30 岁
33	31 到 35 岁
38	36 到 40 岁
43	41 到 45 岁
48	46 到 50 岁
53	51 到 55 岁
58	56 到 60 岁
63	61 到 65 岁
68	66 到 70 岁

4.1.4 表情识别功能

使用图片作为参数使用表情识别函数。

函数声明：

```
int PFD_SmileRecogImg (unsigned char* bmpData
                       PFD_ SMILE _DETECT* smileInfo,
                       short faceRote)
```

参数：

unsigned char*	bmpData	位图数据
PFD_SMILE_DETECT	smileInfo	表情信息
short	faceRote	输入图像的人脸角度信息

返回值：

PFD_STATUS_OK	正常结束
PFD_STATUS_NG	异常结束
PFD_STATUS_INVALID	认证失败

表情信息结构体

```
/*  表情构造体*/
typedef struct _pfd_smile_info {
    PFD_FACE_POSITION faceInfo;   /*  人脸坐标结构体  */
    short          smile;   /*  笑脸程度(0~100  低~高)*/
} PFD_SMILE_INFO;

typedef struct _pfd_smile_detect {
    short          num;     /*  图片中人脸数(最大:PFD_MAX_FACE_NUM) */
    PFD_SMILE_INFO    info[PFD_MAX_FACE_NUM];       /*笑脸信息*/
} PFD_SMILE_DETECT;
```

4.1.5 人脸朝向识别功能

使用图片作为参数使用人脸朝向识别函数。

函数声明：

```
int PFD_DirectRecogImg (unsigned char* bmpData
                        PFD_ DIRECT _DETECT* directInfo,
                        short faceRote)
```

参数：

unsigned char*	bmpData	位图数据
PFD_DIRECT_DETECT	directInfo	方向信息
short	faceRote	输入图像的人脸角度信息

返回值：

PFD_STATUS_OK	正常结束
PFD_STATUS_NG	异常结束
PFD_STATUS_INVALID	认证失败

人脸方向信息结构体

```
/*  人脸方向构造体*/
typedef struct _pfd_direct_info {
    PFD_FACE_POSITION faceInfo;   /*  人脸坐标结构体  */
    short          pitch;    /*  抬头低头的角度(-180~180度，抬头是正值)*/
    short          yaw;      /*  摇头角度(-180~180度，向左摇头是正值)*/
    short          roll;      /*  倾斜角度(-180~180度，顺时针是正值)*/
} PFD_DIRECT_INFO;

typedef struct _pfd_direct_detect {
    short          num;      /*  图片中人脸数(最大:PFD_MAX_FACE_NUM) */
    PFD_DIRECT_INFO   info[PFD_MAX_FACE_NUM];/*人脸方向信息*/
} PFD_DIRECT_DETECT;
```

4.1.6 闭眼睁眼判断功能

函数功能：使用图片作为参数使用闭眼判断函数。

函数声明：

```
int PFD_BlinkRecogImg (unsigned char* bmpData
                       PFD_ BLINK _DETECT* blinkInfo,
                       short faceRote)
```

参数：

unsigned char*	bmpData	位图数据
PFD_ BLINK_DETECT	blinkInfo	闭眼信息
short	faceRote	输入图像的人脸角度信息

返回值：

PFD_STATUS_OK	正常结束
PFD_STATUS_NG	异常结束
PFD_STATUS_INVALID	认证失败

人脸方向信息结构体

```
/*  人脸方向构造体*/
typedef struct _pfd_blink_info {
```

```
    PFD_FACE_POSITION   faceInfo;   /*  人脸坐标结构体  */
    short           lb;         /*  左眼闭眼程度，数值越大代表眼睛越大*/
    short           rb;         /*  右眼闭眼概率，数值越大代表眼睛越大*/
} PFD_BLINK_INFO;

typedef struct _pfd_blink_detect {
    short           num;    /*  图片中人脸数(最大:PFD_MAX_FACE_NUM) */
    PFD_BLINK_INFO    info[PFD_MAX_FACE_NUM];       /*闭眼信息*/
} PFD_BLINK_DETECT;
```

4.1.7 人脸特征值的取得和对比功能

函数功能：自动取得一张图像内所有的人脸特征值。

函数声明：

```
int PFD_GetFeature (unsigned char * bmpData,
                    PFD_FACE_DETECT* faceInfo,
                    unsigned char ** feature,
                    short faceRote
                    )
```

参数：

unsigned char*	bmpData	位图数据
PFD_FACE_DETECT*	faceInfo	人数和人脸位置信息
unsigned char **	feature	识别出的特征值数组
short	faceRote	输入图像的人脸角度信息

返回值：

PFD_STATUS_OK	正常结束
PFD_STATUS_NG	异常结束
PFD_STATUS_INVALID	认证失败

```
/*  人脸情报保存结构体  */
typedef struct _pfd_detect_info{
        PFD_FACE_POSITION faceInfo; /*  人脸坐标结构体  */
        short ageConf;              /*  年龄识别可信度  (0~1000  低~高) */
        short genConf;              /*  性别识别可信度  (0~1000  低~高) */
        short age;                  /*  年龄  */
        short gen;                  /*  性别  */
        short smile;                /*  笑脸程度(0~100  低~高)*/
        short pitch;                /*  抬头低头的角度(-180~180度，抬头是正值)*/
```

```
        short yaw;                  /* 摇头角度(-180~180度，向左摇头是正值)*/
        short roll;                 /* 倾斜角度(-180~180度，顺时针是正值)*/
        short lb;                   /* 左眼闭眼概率，数值越大代表眼睛越大*/
        short rb;                   /* 右眼闭眼概率，数值越大代表眼睛越大*/
        short flen;                 /* 特征值长度 */
}PFD_DETECT_INFO;

/* 图像内的人脸信息结构体 */
typedef struct _pfd_face_detect {
    short          num;    /* 图像中人脸数(最大:PFD_MAX_FACE_NUM) */
    PFD_DETECT_INFO  info[PFD_MAX_FACE_NUM];/* 人脸的位置和情报信息 */
} PFD_FACE_DETECT;
```

注意：

1：在调用此函数前，需要首先创建一个指针数组，为指针数组中的每一个指针分配 2048 个字节的内存。

2：函数运行后，会将实际的每个人脸的特征值长度保存在 info 中的 flen 字段中。

3：步骤 1 申请的内存请自行释放。

函数功能：手动设置范围取得特征值

函数声明：

```
int PFD_GetFeatureByManual (unsigned char * bmpData,
                            PFD_FACE_POSITION* faceInfo,
                            unsigned char * feature,
                            short faceRote,
                            short* fsize
                            )
```

参数：

unsigned char*	bmpData	位图数据
PFD_FACE_POSITION *	faceInfo	人脸的眼睛和边框位置信息
unsigned char *	feature	识别出的特征值数组
short	faceRote	输入图像的人脸角度信息
short*	fsize	返回的特征值大小

返回值：

PFD_STATUS_OK	正常结束
PFD_STATUS_NG	异常结束
PFD_STATUS_INVALID	认证失败

人数和人脸信息结构体：

```
/*  人脸坐标结构体  */
typedef struct _pfd_face_position {
    short       conf;     /*  识别信赖度(0~1000  低~高) */
    short       rect_l;   /*  人脸最左侧坐标，可省略*/
    short       rect_r;   /*人脸最右侧坐标，可省略*/
    short       rect_t;   /*人脸最上侧坐标，可省略*/
    short       rect_b;   /*人脸最下侧坐标，可省略*/
    short       eye_lx;   /*人脸左眼x坐标*/
    short       eye_ly;   /*人脸左眼y坐标*/
    short       eye_rx;   /*人脸右眼x坐标*/
    short       eye_ry;   /*人脸右眼y坐标*/
} PFD_FACE_POSITION;
```

注意：

1：在调用此函数前，需要首先创建一个指针数组，为指针数组中的每一个指针分配 2048 个字节的内存。

2：步骤 1 申请的内存请自行释放。

3：可以省略人脸的边框坐标，4 个边框坐标都设置为 0，会自动根据眼睛坐标计算出边框坐标。

4：如果设置了边框坐标，会使用眼睛和边框的组合来取得特征值，但是有可能取不出来，推荐只使用眼睛坐标取得特征值。

函数功能：对比特征值

函数声明：

```
int PFD_FeatureMatching(short fLen1,
                        unsigned char* feature1,
                        short fLen2,
                        unsigned char * feature2)
```

参数：

short	fLen1	第一个特征值大小
void*	feature1	第一个特征值
short	fLen2	第二个特征值大小
void*	feature2	第二个特征值

返回值：

0 到 100	匹配结果，一般大于 80 认为相似度极高
PFD_STATUS_NG	异常结束
PFD_STATUS_INVALID	认证失败

4.1.8 使用内置数据库进行人脸相似度对比

本识别库内置了人脸特征值数据库，只要程序不中止，数据库的内容一直保存在内存中。可以实现相对高速的检索速度。但是因为数据全部保存在内存中，所以最大保存数量推荐在 50 万以内。并且在程序退出后，数据库中的内容不会保存。推荐尽量使用内置数据库进行检索。

本库可以对应多个 PDB，目前最多可以支持 4 个。用户在新建数据库后可以通过返回的编号来使用数据库。

函数功能：新建数据库

函数声明：

```
int PDB_AddDataBase(int maxFaceNum)
```

参数：

int	maxFaceNum	数据库内保存的最多人脸数

返回值：

>0	数据库的编号
PFD_STATUS_NG	异常结束
PFD_STATUS_INVALID	认证失败

函数功能：删除数据库

函数声明：

```
int PDB_DeleteDataBase(short dbId);
```

参数：

short	dbId	需要删除的数据库编号

返回值：

PFD_STATUS_OK	正常结束
PFD_STATUS_NG	异常结束
PFD_STATUS_INVALID	认证失败

函数功能：重置数据库

函数声明：

int PDB_ResetDataBase(short dbId);

参数：

short	dbId	需要重置的数据库编号

返回值：

PFD_STATUS_OK	正常结束
PFD_STATUS_NG	异常结束
PFD_STATUS_INVALID	认证失败

函数功能：保存特征值

函数声明：

```
int PDB_StoreFeature (short dbId,
                      short fsize,
            unsigned char* feature)
```

参数：

short	dbId	数据库的编号
short	fsize	特征值大小
void*	feature	指向特征值数组的指针

返回值：

>=0	数据库中的 ID
PFD_STATUS_NG	异常结束
PFD_STATUS_INVALID	认证失败

函数功能：删除特征值

函数声明：

int PDB_DeleteFeature (short dbId,int usid)

参数：

short	dbId	数据库的编号
int	usid	数据库中的 ID，PDB_StoreFeature 的返回值

返回值：

PFD_STATUS_OK	正常结束
PFD_STATUS_NG	异常结束
PFD_STATUS_INVALID	认证失败

函数功能：修改特征值

函数声明：

```
int PDB_DeleteFeature (short dbId,
                       int usid,
                       short fsize,
                       unsigned char* feature)
```

参数：

short	dbId	数据库的编号
int	usid	需要修改的 ID
short	fsize	特征值大小
unsigned char*	feature	特征值指针

返回值：

>=0	数据库中的 ID
PFD_STATUS_NG	异常结束
PFD_STATUS_INVALID	认证失败

函数功能：检索特征值

函数声明：

```
int PDB_MatchFeature (short dbId,
                      unsigned int threshold,
                      short fsize,
                      unsigned char* feature,
                      unsigned short num,
                      int* candidate,
                      short* score)
```

参数：

short	dbId	数据库的编号
unsigned int	threshold	检索结果的最小匹配度(0~100)
short	fsize	特征值大小
unsigned char*	feature	特征值指针
unsigned short	num	检索结果的最大个数
int*	candidate	检索出的结果 ID
short*	score	检出结果的每一个 ID 的得分(0~100)

返回值：

>=0	匹配的记录个数
PFD_STATUS_NG	异常结束
PFD_STATUS_INVALID	认证失败

函数说明

1：threshold 表示检索结果中包含的最小匹配度，合适的输入是 85。过高可能无输出结果，过低可能会输出大量的不相同人脸。

2：candidate 和 score 需要事先分配内存，数组的个数是 num 个

4.1.9 SDK 注销

函数声明

int PFD_Exit()

返回值

PFD_STATUS_OK	正常结束
PFD_STATUS_NG	异常结束
PFD_STATUS_INVALID	认证失败

函数说明：在退出时必须调用，否则会产生内存泄漏的问题。

5 限制事项及性能指标说明

5.1 限制事项

照明环境、太阳镜、口罩以及人脸倾斜角度会多少影响到人脸识别的精度。

5.1.1 最小可识别人脸宽度

可以采用以下公式，计算出最小可识别人脸宽度，单位：pixel。

1：→IMG_SIZE_VGA

最小可识别人脸宽度=MAX(MAX(图像宽度/640，图像高度/480)*20, 20)

表 5.1 →IMG_SIZE_VGA 最小可识别人脸宽度

图像尺寸（宽）	图像尺寸（高）	最小可识别人脸宽度
1920	1080	60
1280	960	40
1280	720	40
640	480	20
640	360	20
320	240	20
320	180	20

2：→IMG_SIZE_FULLHD

最小可识别人脸宽度=MAX(MAX(图像宽度/1920，图像高度/480)*20, 20)

表 5.2 →IMG_SIZE_FULLHD 最小可识别人脸宽度

图像尺寸（宽）	图像尺寸（高）	最小可识别人脸宽度
8192	4096	86
4096	2048	43
1920	1080	20
1280	720	20
640	480	20
320	180	20

5.1.2 人脸识别角度

人脸识别角度约为上下 30 度，左右 45 度。

5.1.3 最小可识别人眼间距

最小可识别人眼间距约为 10 像素。

5.1.4 同时最大识别人脸数

一张图像里可同时识别的最大人脸数为 20。

5.1.5 输入图像要求

最大图像尺寸：8192*4096 像素

最小图像尺寸：40*40 像素

图像格式： 24 位 RGB 的 BMP 格式。

5.2 性能指标

5.2.1 人脸匹配正确率

实时匹配时，由于人脸的角度，表情等是实时变化的，会影响到匹配的正确率。通常情况下，第一帧图片的匹配率达到 90%以上，10 帧的匹配率可以达到 100%。

5.2.2 人脸识别、匹配速度

操作系统的不同（如：Windows、Android）,计算机的性能的不同（如：CPU、RAM）等都会影响人脸识别和匹配的速度。

下面是在 Windows 7、CPU 2.5GHz、RAM 4.0GB、IMG_SIZE_VGA 设定下的测试结果，仅供参考。

表 5.3 人脸识别、匹配速度

分辨率	一张人脸处理速度速度（ms）	一秒钟处理人脸数
QVGA(320*240)	140	7
VGA(640*480)	240	4
720P(1280*720)	290	3
1080P(1920*1080)	330	3

5.2.3 服务器端人脸识别性能

从网络摄像机取得实时图像在服务器端进行人脸识别，可提高系统效率，节省许可证费用。但是受服务器性能，相机分辨率等的影响，一个开发平台可同时处理的相机数量有限制。

下面是在 Windows 7、CPU 2.5GHz、RAM 4.0GB、集成显卡、IMG_SIZE_VGA 设定下的测试结果，仅供参考。

表 5.4 实时图像显示，人脸识别匹配速度

分辨率	连接台数	线程数	内存使用率（GB）	CPU 使用率	1 秒处理画像数
QVGA(320*240)	5	5	1.3	50%～70%	35
VGA(640*480)	5	5	1.5	50%～70%	20
720P(1280*720)	3	3	1.0	50%～60%	9
1080P(1920*1080)	2	2	0.6	50%～60%	6

5.2.4 人脸识别的最大场景宽度

分辨率不同，最小可识别人脸宽度不同，最大场景宽度也就不同。下面给出一部分分辨率下，最大场景宽度的概算。用户可以根据最大场景宽度来设置相机位置和选用合适的镜头。

表 5.5 IMG_SIZE_VGA 设定人脸识别最大场景宽度

分辨率（pixel）	最小可识别人脸宽度（pixel）	最大场景宽度（m）
1920*1080	60	6.4
1280*960	40	6.4
1280*720	40	6.4
640*480	20	6.4
640*360	20	6.4
320*240	20	3.2
320*180	20	3.2

表 5.6 IMG_SIZE_FULLHD 设定人脸识别最大场景宽度

分辨率（pixel）	最小可识别人脸宽度（pixel）	最大场景宽度（m）
1920*1080	20	19.2
1280*960	20	12.8
1280*720	20	12.8
640*480	20	6.4
640*360	20	6.4
320*240	20	3.2
320*180	20	3.2