**AI1000设计文档**

|  |  |  |  |
| --- | --- | --- | --- |
| 修改人 | 版本 | 内容说明 | 日期 |
| 李翰臣 | 0.1 | 创建文档，并根据实际情况添加内容 | 2017-11-08 |
| 李翰臣 | 0.2 | 根据于瀚尧修改的第二版设计修改 | 2017-11-17 |
| 李翰臣 | 0.3 | 根据于瀚尧修改的第三版设计修改 | 2017-11-22 |
| 李翰臣 | 0.31 | 根据于瀚尧修改的第三版设计进行开会讨论进行修改 | 2017-11-28 |
| 李翰臣 | 0.32 | 增加代码注释说明和控件说明，更多UI参考UI设计图 | 2017-12-14 |
| 何颖醛 | 0.33 | 完善管理特征数据线程的说明，与代码保持一致 | 2018-02-23 |
| 何颖醛 | 0.34 | 完善舵机控制线程的说明，与代码保持一致 | 2018-02-24 |

注: 暂未完成，仅供参考

目录

[一、说明 4](#_Toc501096643)

[二、UI界面 4](#_Toc501096644)

[2.1屏保模式 4](#_Toc501096645)

[2.2虹膜识别界面或人脸识别界面 5](#_Toc501096646)

[2.2.1状态栏 5](#_Toc501096647)

[2.2.2 正中人脸区域 5](#_Toc501096648)

[2.2.3 左侧操作区域 6](#_Toc501096649)

[2.2.4 右侧操作区域 6](#_Toc501096650)

[2.2.5 后台运行线程 6](#_Toc501096651)

[2.2.6 定时器 6](#_Toc501096652)

[2.3识别结果界面 8](#_Toc501096653)

[2.4管理界面 9](#_Toc501096654)

[2.5管理员验证界面 10](#_Toc501096655)

[2.5.1虹膜验证登陆 10](#_Toc501096656)

[2.5.2人脸验证登陆 11](#_Toc501096657)

[2.5.3密码验证登陆 11](#_Toc501096658)

[2.5.4后台运行线程 12](#_Toc501096659)

[2.5.5定时器 12](#_Toc501096660)

[2.6用户管理界面 12](#_Toc501096661)

[2.6.1 用户管理主界面 13](#_Toc501096662)

[2.6.2用户管理搜索界面 13](#_Toc501096663)

[2.6.3用户管理删除界面 14](#_Toc501096664)

[2.6.4用户管理添加界面 16](#_Toc501096665)

[2.6.5用户管理详情界面 19](#_Toc501096666)

[2.6.6用户管理的各种提示界面 22](#_Toc501096667)

[2.7系统设置界面 24](#_Toc501096668)

[2.7.1 基本设置 25](#_Toc501096669)

[2.7.2 接口设置 28](#_Toc501096670)

[2.7.3 验证设置 29](#_Toc501096671)

[2.7.4 通信设置 31](#_Toc501096672)

[2.7.5 数据管理 32](#_Toc501096673)

[2.7.6 初始化设置 36](#_Toc501096674)

[三、后台线程或无界面类 37](#_Toc501096675)

[3.1 虹膜识别线程ThreadIdent 37](#_Toc501096676)

[3.2 注册线程ThreadEnroll 38](#_Toc501096677)

[3.3管理特征数据线程（包括数据的加载和删除） 38](#_Toc501096678)

[3.4数据导入导出线程 40](#_Toc501096679)

[3.5RGB采图线程 41](#_Toc501096680)

[3.6数据库访问类 43](#_Toc501096681)

[3.7控制舵机运动线程 43](#_Toc501096682)

[3.8 人脸检测线程 43](#_Toc501096683)

[四、定时器 43](#_Toc501096684)

[五、依赖库 44](#_Toc501096685)

[5.1 自编动态库 44](#_Toc501096686)

[5.2 自编静态库 46](#_Toc501096687)

[5.3 Qt库 46](#_Toc501096688)

[5.4其他系统库 46](#_Toc501096689)

[六、数据库 47](#_Toc501096690)

[6.1 device数据表 47](#_Toc501096691)

[6.2irisdatabase数据表 47](#_Toc501096692)

[6.3 irisalgo数据表 48](#_Toc501096693)

[6.4person数据表 49](#_Toc501096694)

[6.5reclog数据表 49](#_Toc501096695)

[七、配置文件 50](#_Toc501096696)

[7.1 应用程序配置文件 50](#_Toc501096697)

[八、代码注释说明 50](#_Toc501096698)

[8.1 文件的概述说明 50](#_Toc501096699)

[8.2 h文件说明 50](#_Toc501096700)

[8.3 cpp文件说明 51](#_Toc501096701)

[九、其他 51](#_Toc501096702)

[9.1 界面名称 51](#_Toc501096703)

[9.2常用控件命名 52](#_Toc501096704)

# 一、说明

本文档是为了AI1000的Linux版本软件编写的设计说明，供开发、测试人员参考。

硬件参数：分辨率720x1280，竖屏，但在使用的时候是横着用，因此软件要对界面进行一个旋转。

目前暂定不需要进行联网，本地保存普通用户数最多10000人，管理员最多20人，总人数上限10020。

# 二、UI界面

## 2.1屏保模式

设备运行中无论处于哪个界面,如果超过30秒没有动作或者没有检测到人经过或者识别虹膜后，进入该界面，而且进入屏保模式后，如果检测到有人经过或者有人触摸屏幕,无论原来是哪个界面都会切换到识别界面。



图1

根据人脸RGB图像或者虹膜图像从算法SDK测距，还可以触摸屏幕或者点击按钮进入识别状态。

## 2.2虹膜识别界面或人脸识别界面

### 2.2.1状态栏

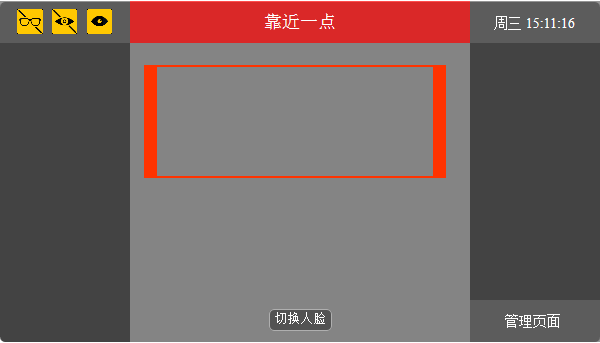


图2

界面最上方是状态栏,显示不戴眼镜，不戴美瞳，睁大眼睛等三个图标。

状态栏中间显示正在进行的功能的提示文字，按照设计UI，最多显示12个中文字。

状态栏右侧显示星期和时间。

### 2.2.2 正中人脸区域

显示摄像头采集的RGB或者虹膜图像，最下面有切换按钮。

显示虹膜图像时，用带颜色的矩形框显示出眼睛预览位置，让用户将眼睛放入到该矩形区域才可以正常识别，同时矩形框上方有适当的文字提示（见下表），矩形框也要根据不同提示用颜色进行区分，并有相应的声音提示。

会根据SDK算法返回人眼位置，绘制一个白色矩形，来标识人眼实际位置。如果界面RGB图像带有裁减和缩放,将对后续显示框的绘制有影响，该显示框和白色人眼实际位置框线形和四角圆的倒角不一致。

表格1

|  |  |  |  |
| --- | --- | --- | --- |
| 状态 | 提示语句 | 提示颜色 | 提示声音文件 |
| 距离合适的位置 | 距离正好 | 绿色 |  |
| 距离过远 | 靠近一点 | 红色 |  |
| 距离过近 | 远离一点 | 蓝色 |  |
| 用户运动过快 | 请保持稳定 | 颜色保持 |  |
| 其他 |  | 颜色保持 |  |
| 按键音 |  |  |  |

### 2.2.3 左侧操作区域

在识别界面的左侧没有控件。

### 2.2.4 右侧操作区域

该区域暂时只有一个“管理页面”按钮，点击后进入管理界面，放在该区域最下面。

### 2.2.5 后台运行线程

人脸识别线程、虹膜识别线程。其中人脸识别线程根据后面的实际配置进行启动，如果没有配置人脸识别，那么人脸识别和虹膜识别的切换按钮都不显示，也不会启动人脸识别线程。

### 2.2.6 定时器

管理上述几个线程的启动停止等，具体待定。

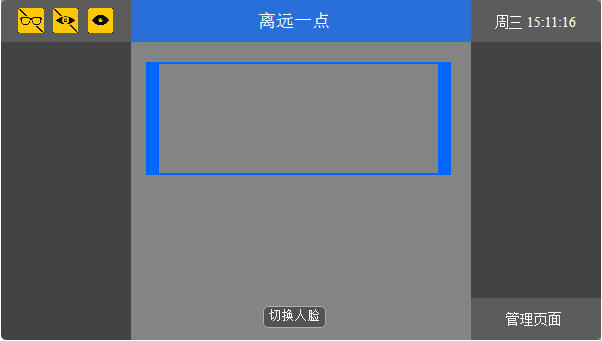


图3

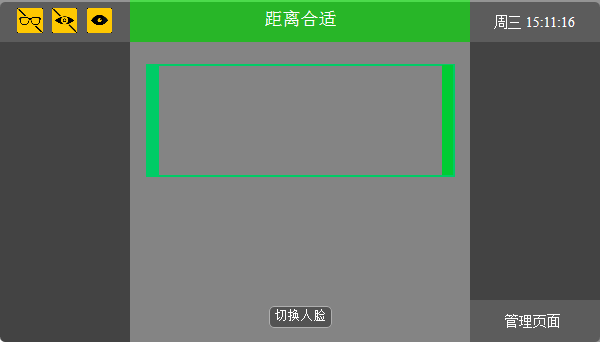


图4虹膜识别界面



图5

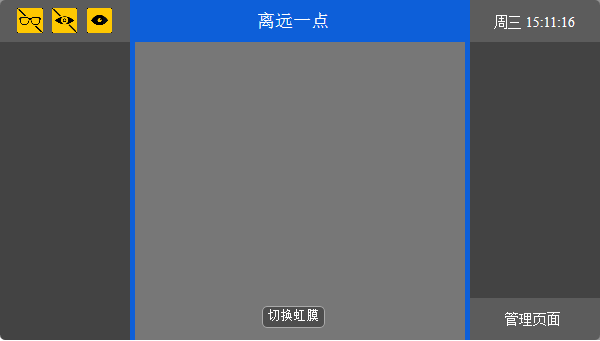


图6

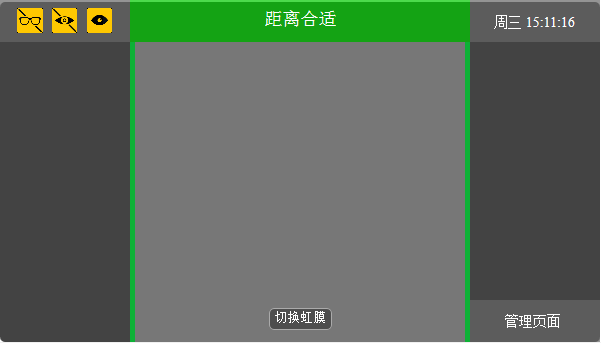


图7

人脸识别界面

## 2.3识别结果界面

无论是否识别成功均有提示。按照设计要求，识别3次不通过提示验证未通过，并拍照记录人脸。

如何算是三次，待定，需要连续三次，但如果是中间很快换人了，该如何判断？通过SDK返回的识别失败三次进行判断。

识别成功，会显示出姓名和工号（卡号？）。

识别不成功，提示验证未通过。

识别结果显示时间暂定3秒，然后返回到识别界面。

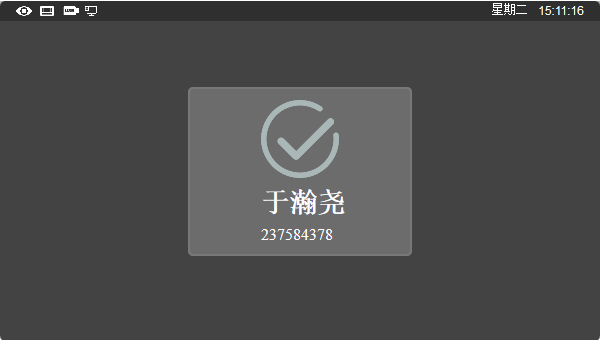


图8



图9

## 2.4管理界面

在点击人脸或虹膜识别界面的“管理页面”按钮后，会切换到管理界面。



图10

此时，无论点击“用户管理”还是“系统设置”，均要进行验证，界面切换到管理员验证界面。

点击左上角的“识别”按钮，返回到识别界面。

在此界面，红外测距线程以及虹膜识别线程要进入暂停状态，减少不必要的系统动作。

## 2.5管理员验证界面

管理员验证界面类似于正常验证界面，分别有虹膜验证、人脸验证、密码登陆和刷卡登陆共计四种方式进行管理员验证。

该界面和前面的2.2节介绍的识别界面类似，但在屏幕最下方的操作有些区别。

### 2.5.1虹膜验证登陆

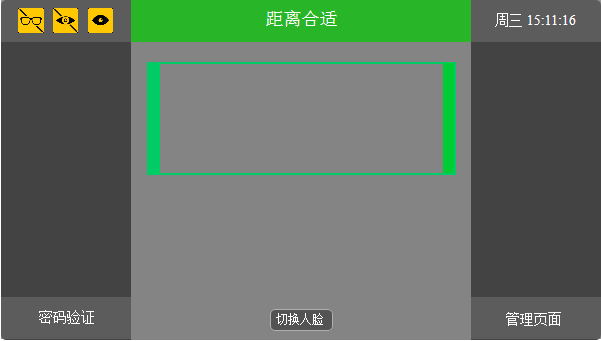


图11

在此界面进行虹膜验证，验证通过后进入用户管理界面；不通过则返回管理员登陆界面。

### 2.5.2人脸验证登陆



图12

在此界面进行人脸验证，验证通过后进入用户管理界面；不通过则返回管理员登陆界面。

### 2.5.3密码验证登陆

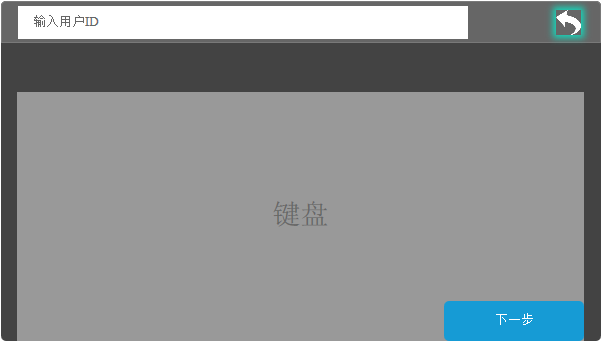


图13

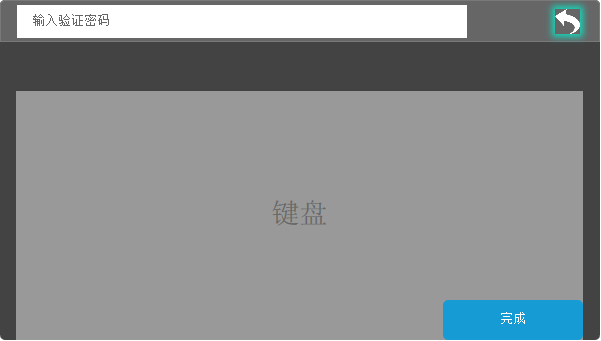


图14

以上验证结果后，显示界面同前面的第二部分识别结果界面。

注：管理员验证界面可以和一般打卡识别界面进行复用，相应按钮在相应的显示界面出现，同时后台运行的线程和定时器也可以复用。

### 2.5.4后台运行线程

人脸识别线程、虹膜识别线程。其中人脸识别线程根据后面的实际配置进行启动，如果没有配置人脸识别，那么人脸识别和虹膜识别的切换按钮都不显示，也不会启动人脸识别线程。

### 2.5.5定时器

管理上述几个线程的启动停止等，具体待定。

注：管理员识别和普通人员识别类似，只是部分按钮有区别，考虑复用。

## 2.6用户管理界面

### 2.6.1 用户管理主界面

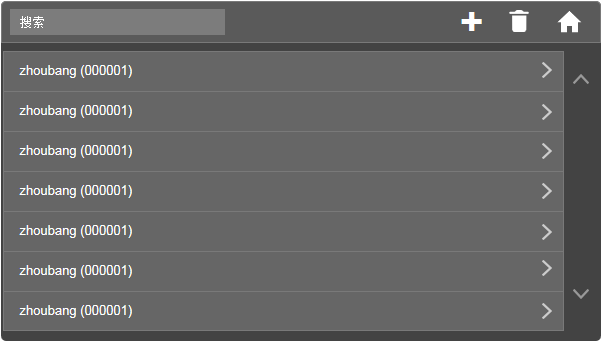


图15

该界面最上面有四个操作，最左侧是搜索输入框，输入搜索关键词，即可进行模糊搜索。

右侧的加号表示添加用户。

右侧的垃圾筒符号表示删除用户。

右侧的小房子符号表示返回到2.4节介绍的界面。

每个人员最右侧的大于号，也是可以点击，点击后进入用户详情。

人员数量超过每页显示数量时，最右侧有上下滑动条可以进行控制。

### 2.6.2用户管理搜索界面

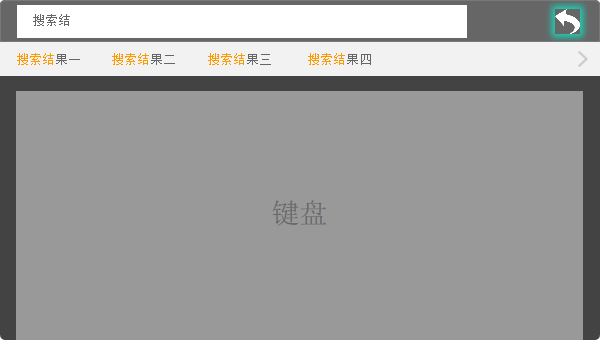


图16

在此界面，输入想要搜索的关键字（支持用户名和id搜索），搜索结果长度过长则显示右箭头，否则不必显示。显示后可以进行点击查看，如下图所示。

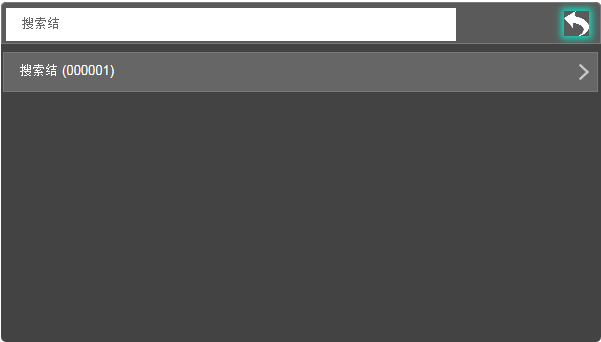


图17

该图显示详细信息？

### 2.6.3用户管理删除界面



图18

在用户管理界面点击垃圾桶按钮后，界面切换，每个人员后面显示“删除”按钮，点击后从界面上删除，但是并不进行实际删除操作，直到点击“完成”按钮才进行真实操作。

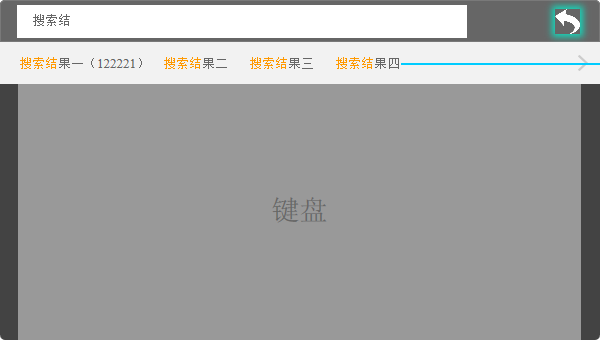


图19

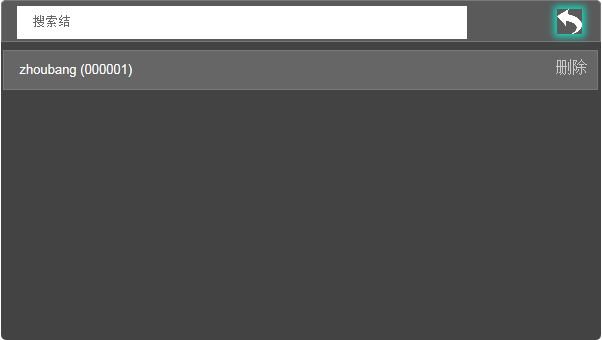


图20

搜索删除如何确认

### 2.6.4用户管理添加界面



图21

1. 用户ID在注册时要进行查重操作，以保证唯一性；
2. 虹膜、卡或者密码注册一种即可保存信息；
3. 保存信息时，要判断注册的信息是否满足验证设置中的验证条件，如果不满足，要进行提示；
4. 所有项输入完成后参考“用户详情页”；
5. 点击”填写更多”，弹出图22；



图22

1. 输入方式如下图

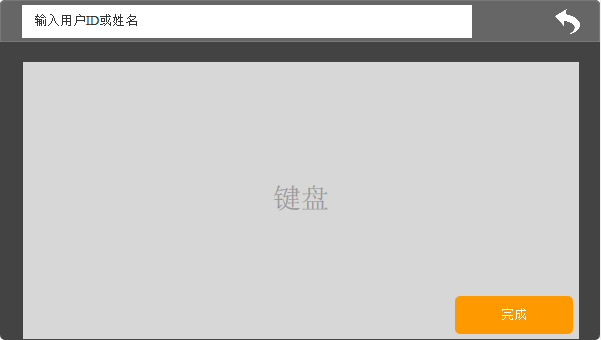


图23

注册卡分两种情况，一种是客户有实体卡，只需要在读卡器上刷卡即可，另外一种是没有实体卡，需要在输入界面进行输入。具体哪种现实需要根据配置来进行显示。



图24

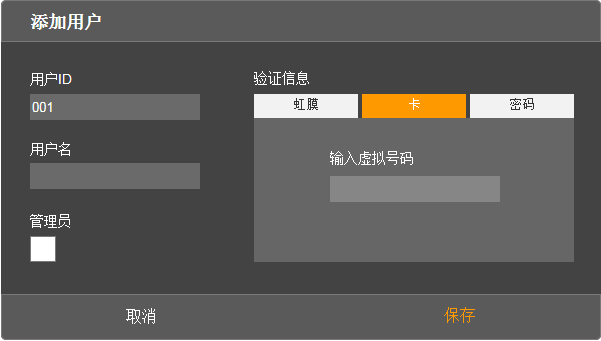


图25

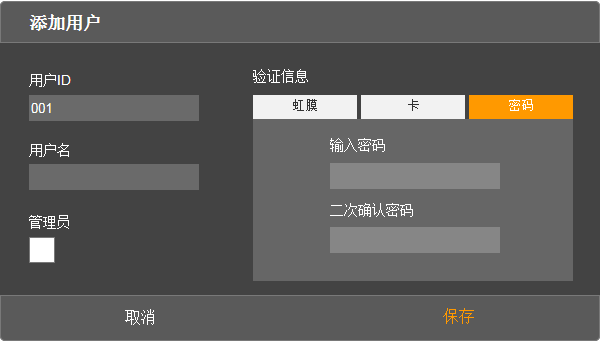


图26

管理员用户可以设置密码，普通用户无需设置密码。分别如图25和图26所示。



图27

### 2.6.5用户管理详情界面

用户详情页面是点击用户后进入的页面，用于展示或者编辑用户信息。其页面右上角有编辑和返回两个按钮。点击编辑后，对应可编辑内容都可以进行修改。

如何修改虹膜和卡，没有详细图例

![C:\Users\Administrator\AppData\Roaming\Tencent\Users\645533713\QQ\WinTemp\RichOle\]Y~J_7FG3B](_NGP$7T$C66.png](data:image/png;base64,)

图28

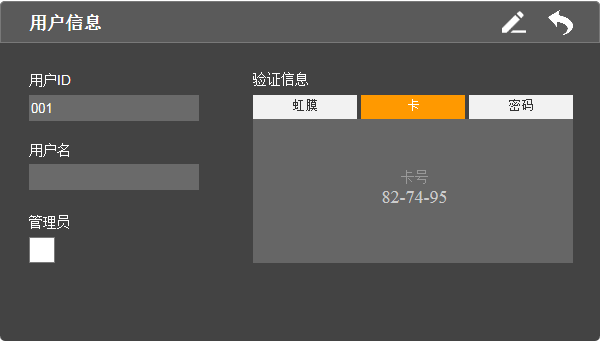


图29



图30



图31

点击编辑按钮后，目前只有密码的修改界面，点击“修改密码”后，会出现原密码输入框，新密码输入框，新密码确认框，用于修改密码。



图32



图33

### 2.6.6用户管理的各种提示界面

虹膜注册时，验证信息重复会提示如下对话框。其中点击取消按钮后，返回修改用户信息界面。

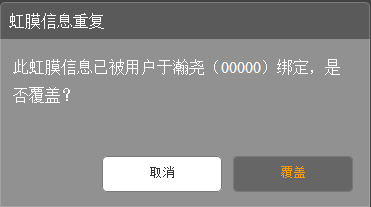


图34

用户ID重复，提示框。

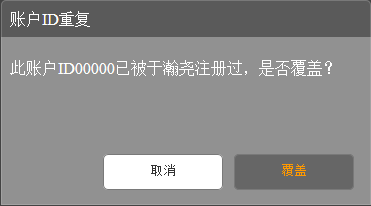


图35

验证信息为绑定是什么情况？

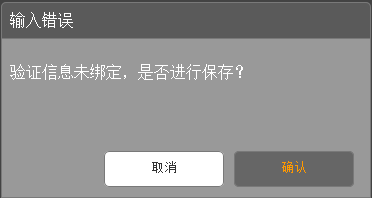


图36

延时5秒退出

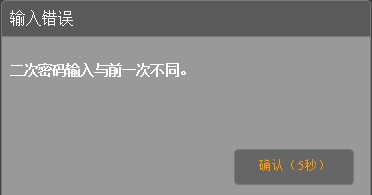


图37

容量将满在什么情况下提示？添加新人员时？剩余容量多少将要提示？

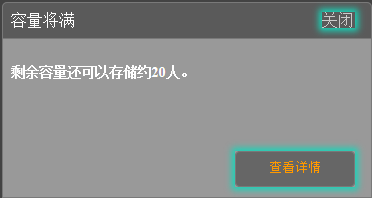


图38

需要注册多种验证方式时，如果没注册全就点击“保存”按钮，需要提示。提示内容要根据情况修改。

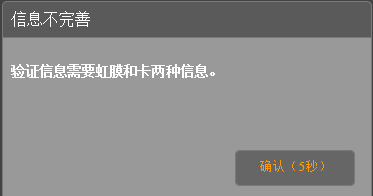


图39

修改人员成功提示



图40

删除二次确认界面

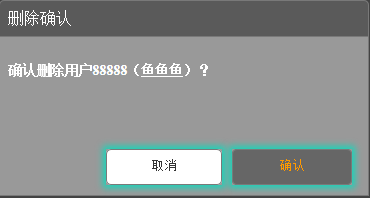


图41

## 2.7系统设置界面

系统设置界面分为基本设置、接口设置、验证设置、通信设置、数据管理和初始化设置共六个功能。



图42

### 2.7.1 基本设置

1.声音

按键声音：按下按键是否播放提示声音。

语音提示：采集及验证过程是否有语音提示。

音量设置：设置提示语音声音的音量大小，分为0-10，其中0是静音，10是最大音量。



图43



图44

2.日期时间

联网自动更新开关：设置为开，则定时与服务器进行时间同步，后面的日期及时间不能进行修改。

日期：显示或修改当前日期，当联网自动更新开启时，此项不可修改。

时间：显示或修改当前时间，当联网自动更新开启时，此项不可修改。



图45

修改日期和时间界面类似下图结构。



图46

3.系统信息

系统信息下面分为系统容量和设备信息

系统容量：显示已注册用户数量、管理员数量、显示总容量以及使用百分比。

设备信息：显示设备的版权信息、软件版本号、序列号、MAC地址、公司地址以及联系电话。（联系电话是售后还是售前、是否可以加上公司的售后支持邮箱地址）



图47



图48

系统容量计算是根据硬盘分配空间比例以及存储单人需要的空间进行计算，其中硬盘分配比例要考虑设备日志空间，避免出现日志和存储虹膜信息共用空间情况。



图49

### 2.7.2 接口设置

该界面用来设置输出信号。

1. 韦根设置

有韦根26和韦根34两种选择。

1. I/O开关设置

选择是否输出I/O信号，用来控制继电器开关。



图50

### 2.7.3 验证设置

1.用户验证方式设置

验证通过方式（模式）：虹膜或卡、虹膜和卡、只有虹膜和只有卡，共四种方式。

虹膜验证方式：任意眼或双眼，共两种方式。

操作模式，点击选项右侧的“>”符号，会在屏幕下面弹出选择列表，同时原有界面变成半透明状，选择某一选项后，返回本界面。

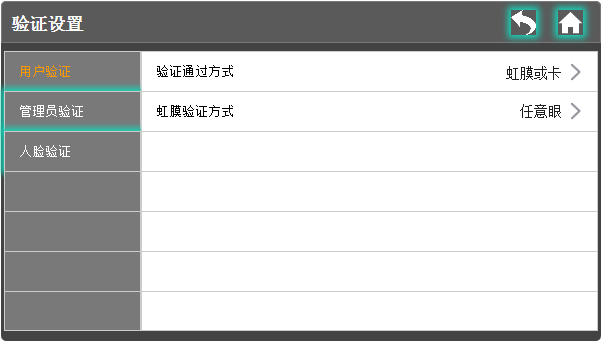


图51

![C:\Users\Administrator\AppData\Roaming\Tencent\Users\645533713\QQ\WinTemp\RichOle\1S](H691T7E254ZX)S9GEAR.png](data:image/png;base64,)

图52



图53

在选择了虹膜和卡的验证方式后，要验证数据中所有人员是否都有卡和虹膜信息，如果有人没有就要显示如下提示。

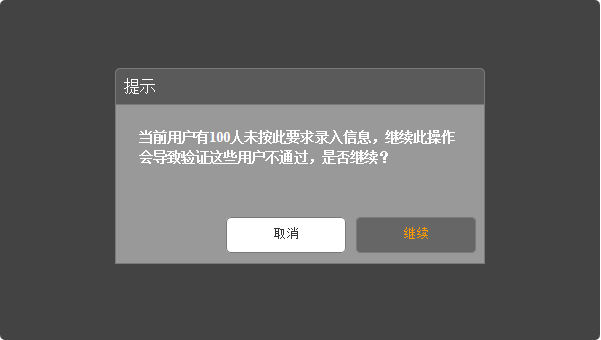


图54

2.管理员验证设置

设置管理员的验证方式，与普通用户一样，多出一个“是否允许密码验证”，选上该选项之后就允许管理员用密码登陆。



图55

3.人脸验证设置

设置人脸识别是否与虹膜识别同时开启，在管理员登陆页面可以切换两种识别方式。

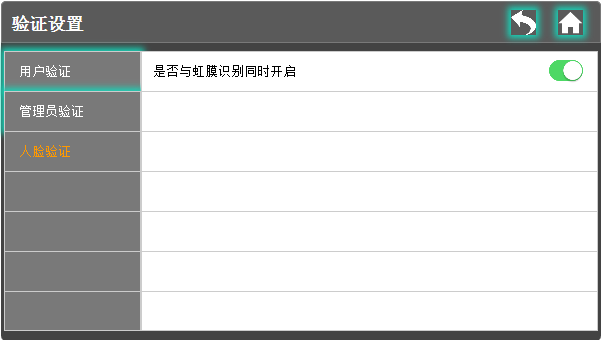


图56

### 2.7.4 通信设置

1.设备端网络设置



图57

设置内容包括设备的IP、网关和子网掩码。

2.服务器连接设置

设置设备需要连接的服务器IP和端口号。



图58

### 2.7.5 数据管理

1. 人员的导入导出

按照约定格式保存人员信息到excel文件里，将该文件保存到U盘上，将U盘插入到设备上，点击“导入”按钮，完成人员导入。

将U盘插入到设备上，点击“导出”按钮，会将人员信息按照约定格式保存到U盘上，文件格式为excel。

操作无论成功或失败都会有提示。

导出或导入内容和6.4节person数据表内容一致。

进度的计算

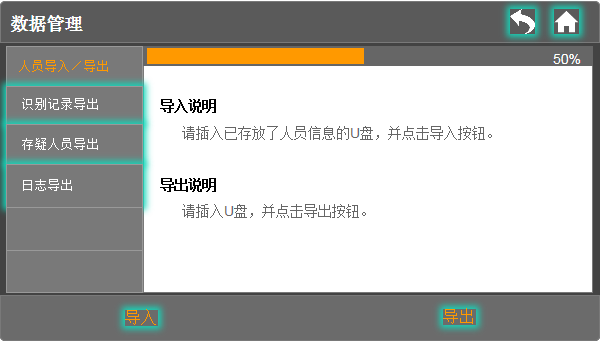


图59

1. 识别记录导出

将设备上的识别记录数据导出到U盘，导出内容和6.5节reclog数据表内容一致。导出格式，未定。

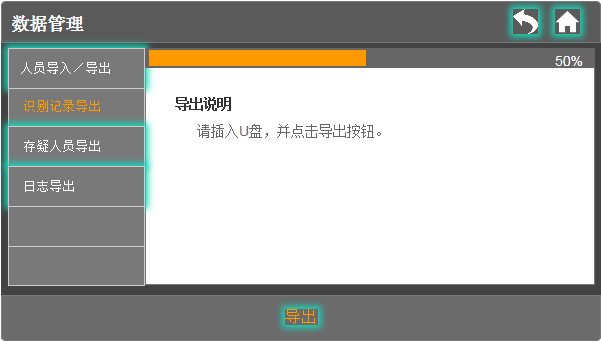


图60

1. 存疑人员导出

验证失败的人员，拍摄虹膜图像和人脸图像各一张记录为存疑。图像格式为jpg格式。保存文件名分别按照iris+日期时间、face+日期时间来设置，日期时间格式为YYYYMMDDhhmmsss。保存目录分开，分别在主程序下iris和face目录内。对于保存目录，每周进行压缩备份，避免出现目录内文件过多造成程序崩溃。

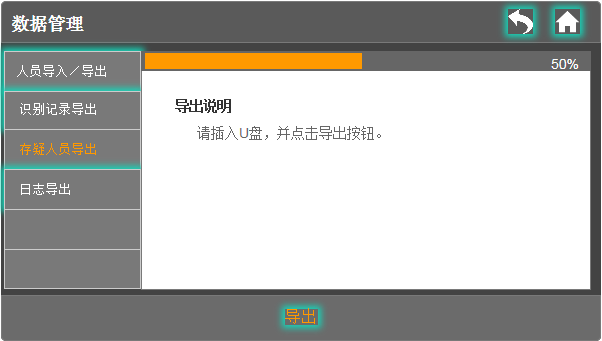


图61

1. 日志导出

将应用程序生成的日志文件导出到U盘。参考CMI的日志是识别记录，而我们说的日志是程序运行记录。

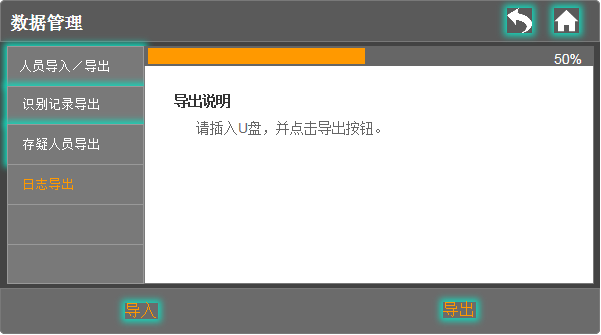


图62

1. 虹膜数据的导入导出

导出数据格式参考6.2节irisdatabase数据表和6.3节irisalgo数据表，导出文件格式为db格式（sqlite**）**

6.操作提示

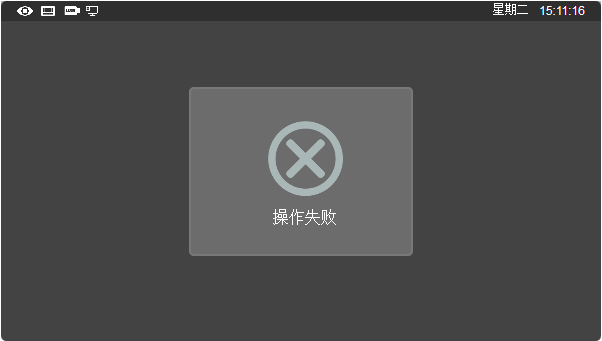


图63



图64

### 2.7.6 初始化设置



图65

初始化设置包括两项内容

1. 重启设备

对设备进行重启，点击重启设备后，会弹出确认框，进行二次确认。在确认框上点击“确认”按钮，设备立即进行重启；点击左上角“关闭”按钮，则返回到初始化设置界面。

1. 恢复出厂设置

对设备进行恢复出厂设置，点击恢复出厂设置后，会弹出确认框，进行二次确认。在确认框上点击“确认”按钮，设备会将出厂配置文件恢复，同时清空数据库，日志文件暂时不会清除；点击左上角“关闭”按钮，则返回到初始化设置界面。

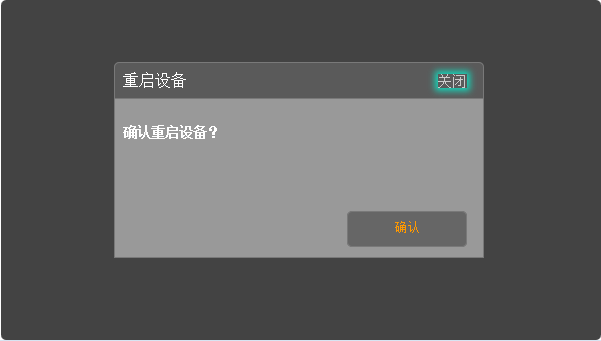


图66

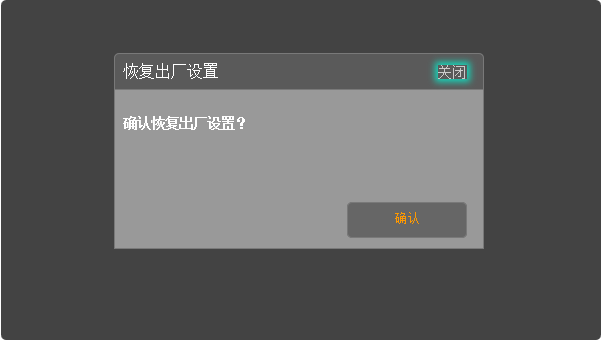


图67

# 三、后台线程或无界面类

## 3.1 虹膜识别线程ThreadIdent

识别线程设计为while(true)循环，只要在识别界面就会一直运行识别线程；当SDK识别接口返回识别结果后，识别线程发送识别结果给UI线程，同时识别线程自动退出（识别超时除外）；UI线程在接收到识别线程发送的识别结果后，进行结果的显示后，再次启动识别线程（除非退出识别界面）。

1、识别线程需要的SDK接口：

int IKUSBSDK\_StartIrisIden(参数……);

int IKUSBSDK\_StopIden();

2、线程提供的接口：

int initARG(参数…….);

int getIdentState();

int startIdent();

int stopIdent();

3、线程提供的信号

void identResult(参数…….); //发送识别结果，在SDK识别接口函数返回后发出信号。

## 3.2 注册线程ThreadEnroll

点击开始注册按钮后，启动该线程，注册完成（成功或者失败）之后，该线程停止。

是否注册回调函数，查重处理接口SDK，主线程

1. 注册线程需要的SDK接口有

Int IKUSBSDK\_StartIrisEnroll(参数……);

Int IKUSBSDK\_StopIrisEnroll();

1. 线程提供的接口

Int initARG(参数…….); // 初始化必要参数

Int getEnrollState(); // 获取注册线程状态

Int startEnroll(); // 启动注册

Int stopEnroll(); // 停止注册

1. 线程内部需要到函数

void sound(参数…….); //播放声音函数，根据参数播放不同声音。播放声音是阻塞在此。

1. 线程提供的信号

发出注册成功后获取的虹膜信息的信号

void enrollResult(IKSDKIrisInfo\* enrIrisL, int numEnrL, IKSDKIrisInfo\* enrIrisR, int numEnrR, IKResultFlag IKflag); //发出注册结果，在sdk注册接口函数返回后发出信号

注册状态变化的信号

void enrollState(int state); //0成功，1失败，2终止，3结束

启动注册的信号,让线程开始进行注册

void beginEnroll();

## 3.3管理特征数据线程（包括数据的加载和删除）

后台管理载虹膜特征数据，与识别线程启动是否有启动顺序的要求。

线程可直接访问数据库，第一次加载时，从数据库中读取；后续的操作（增加和删除），通过调用接口对特征和人员信息进行操作；管理员的特征加载是否需要单独完成？

主程序启动时从数据库里加载数据，主程序运行时从内存中直接操作，对应数据库操作由主线程管理。

内存中包括人员姓名、uuid、工号、卡号等信息。（List或者map）

每台设备只能有20个管理员，超过再注册会有提示。

1. 线程不需要SDK接口
2. 线程提供的接口
3. 初始化线程参数

int initARG(const IrisManInitArgs& args);

在成功调用initARG函数100毫秒后，线程自动根据初始化参数加载特征。

输入参数：

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| args | IrisManInitArgs | loadLocal=true，表示从本地数据库加载；sendSigBlock表示加载多少条特征后，发信号loadFeatureResult通知；  如果sendSigBlock=0，则在加载过程中不发信号，最后结束的时候发一次信号 |

结构定义：

typedef struct \_\_IrisManInitArgs

{

bool loadLocal; // 是否从本地加载

int sendSigBlock; // 加载多少条特征后发一次消息

}IrisManInitArgs, \*PIrisManInitArgs;

返回值：

E\_IRISMAN\_SUCCESS（0），成功

E\_IRISMAN\_LOADING\_NOW\_ERROR（-0x1001），已经在加载中，不做重复加载

E\_IRISMAN\_INITARG\_ERROR（-0x1002），设置参数有误，sendSigBlock<0

1. 获取当前特征加载状态

int getLoadState(bool& loading, int& totalCount, int& leftCount, int& rightCount);

输出参数：

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| loading | bool | =true，表示正在加载；=false，表示当前没有进行加载 |
| totalCount | int | 总共需要加载的特征个数 |
| leftCount | int | 左眼已加载特征个数 |
| rightCount | int | 右眼已加载特征个数 |

返回值：总共需要加载的特征个数

1. 从数据库中读取全部特征加载到内存中

int loadData(void);

如果调用了initARG，会在100毫秒后自动调用loadData，不需要重复调用。

参数：无

返回值：错误码，0表示成功

其他错误码请查看irismanager.h及msqldatabase.h中定义的错误码。

1. 释放内存中管理的所有特征

int releaseData(void);

参数：无

返回值：

E\_IRISMAN\_SUCCESS（0），成功

E\_IRISMAN\_LOADING\_NOW\_ERROR（-0x1001），正在加载中

1. 根据人员信息，删除此人及此人相关的所有虹膜

int deletePerson (const PeopleInfo& info);

输入参数：

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| info | PeopleInfo | 根据info.PersonID(uuid)，删除加载的指定人员的所有虹膜特征和此人的相关信息 |

结构定义：

typedef struct \_\_PeopleInfo

{

uuid\_t PersonID; // 人员ID，uuid

uuid\_t FeatureID; // 特征ID，uuid

std::string CardID; // 卡号

std::string UserID; // 用户号

std::string Name; // 用户姓名

std::string Depart; // 部门，暂时无用

std::string Sex; // 性别

bool IsSuper; // 是否为管理员，true是管理员

}PeopleInfo;

返回值：

E\_IRISMAN\_SUCCESS（0），成功

E\_IRISMAN\_LOADING\_NOW\_ERROR（-0x1001），正在加载中

E\_IRISMAN\_NOPERSON\_ERROR（-0x1003），没有这个人

1. 删除指定人员的指定虹膜特征

int deleteFeatureData(const InfoData & info);

输入参数：

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| info | InfoData | 根据info.PeopleData.PersonID(uuid)和info.ud，删除加载的指定虹膜特征；调用此函数时不需要设置左右眼标志 |

结构定义：

//枚举：左眼、右眼、全部、任意眼

enum class EyeFlag:short{All,Left,Right,Either};

// 虹膜特征及人员信息

typedef struct \_\_InfoData

{

uuid\_t ud; //特征uuid

EyeFlag TypeControl; // 左右眼

unsigned char FeatureData[g\_FeatureSize];

PeopleInfo PeopleData;

}InfoData,\*PInfoData;

返回值：

E\_IRISMAN\_SUCCESS（0），成功

E\_IRISMAN\_LOADING\_NOW\_ERROR（-0x1001），正在加载中

E\_IRISMAN\_NOPERSON\_ERROR（-0x1003），没有这个人

1. 增加人员及特征

int addFeatureData(const InfoData& info);

输入参数：

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| info | InfoData | 将参数中给出的人员、特征加载到内存中 |

返回值：

E\_IRISMAN\_SUCCESS（0），成功

其他错误码请查看irismanager.h中定义的错误码。

1. 修改人员信息

int updatePersonInfo(const PeopleInfo& info);

输入参数：

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| info | PeopleInfo | 将参数中给出的人员信息更新到内存中 |

返回值：

E\_IRISMAN\_SUCCESS（0），成功

E\_IRISMAN\_LOADING\_NOW\_ERROR（-0x1001），正在加载中

E\_IRISMAN\_NOPERSON\_ERROR（-0x1003），没有这个人

1. 获取虹膜特征的指针，分左右眼分别获取

bool getIrisFeature(const EyeFlag& flag, unsigned char\*&pIrisData, int &size);

输入参数：

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| flag | EyeFlag | 获取哪只眼睛的虹膜特征组针，1：左眼；2：右眼 |

输出参数：

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| pIrisData | unsigned char\*& | 对应眼睛的虹膜特征组指针 |
| size | int | 虹膜特征组中有多少条特征 |

返回值：

true，成功；

false，失败

1. 通过特征序号查找人员信息

int getPersonInfoByIndex(int index, PeopleInfo& info, const EyeFlag& type);

输入参数：

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| index | int | 特征序号，从0开始 |
| type | EyeFlag | 表示是哪只眼睛特征组中的序号 |

输出参数：

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| info | PeopleInfo | 查找到的人员信息 |

返回值：

E\_IRISMAN\_SUCCESS（0），成功

E\_IRISMAN\_NOPERSON\_ERROR（-0x1003），没有这个人

1. 线程内部需要用到的函数

数据库访问的相关函数。此线程中，只对数据库做读操作，不做写操作。

1. 线程提供的信号

当前加载进度

void loadFeatureResult(bool, int, int, int);

参数：

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
|  | bool | 当前是否正在加载 |
|  | int | 加载错误码，0为成功 |
|  | int | 数据库中总特征条数 |
|  | int | 当前已加载特征条数 |

## 3.4数据导入导出线程

导入：人员信息

导出：人员信息、识别记录、存疑人员和程序日志。

1. 线程不需要SDK接口
2. 线程提供的接口

int initARG(参数…….); // 初始化必要参数，例如数据库访问参数？导入/导出多少条数据后发一次消息？

int importPerson(int filetype, int loadtype); // 导入人员，filetype：导入文件类型，excel表格或者是sqlite数据库；loadtype：全部新加，还是有的就修改，没有的就新加

int importPersonAndIris(int filetype, int loadtype); // 导入人员和特征；filetype：导入文件类型，xml或者是sqlite数据库；loadtype：全部新加，还是人员有的就修改，没有的就新加；虹膜有的就不做操作，没有的就新加……

int exportPerson(int filetype); // 导出人员，filetype：导入文件类型，xml或者是sqlite数据库；

int exportPersonAndIris(int filetype); // 导出人员和特征；filetype：导入文件类型，xml或者是sqlite数据库；

int exportRecogRecord(int filetype, const std::string& startTime, const std::string& endTime); // 导出识别记录；filetype：导入文件类型，xml或者是sqlite数据库；startTime和endTime，导出记录的时间范围，>startTime and <=endTime

int exportFailuerPhoto (int filetype, const std::string& startTime, const std::string& endTime); // 导出存疑人员，即识别失败的人员照片；startTime和endTime，导出存疑人员的时间范围，>startTime and <=endTime

int exportLog(int type, const std::string& startTime, const std::string& endTime); // 导出log日志；type：日志类型（是否需要导出系统日志？？）；startTime和endTime，导出记录的时间范围，>startTime and <=endTime

1. 线程内部需要用到的函数

数据库访问函数、特征加载？

1. 线程提供的信号

void importResult(); // 表示当前已导入的进度，会给出共需要导入条数和当前已导入条数；现在导入的是什么内容

void importPersonData(参数……); // 导入的人员数据，主线程根据情况做出是否当时更新内存的处理；可以不是每一条都发消息，在初始化时设置

void importIrisData(参数……); // 导入的虹膜数据，主线程根据情况做出是否当时更新内存的处理；可以不是每一条都发消息，在初始化时设置

void exportResult(); // 表示当前导出的进度，会给出共需要导出条数和当前已导出条数；现在导出的是什么内容

## 3.5 RGB采图线程

用于显示RGB图像，回调函数，处理舵机运动。

获取RGB图像线程，主要作用是从5640采集图像，通过调用IKOV5640YUV\_GetMIPIImage() 来获取图像。要引用OV5640MipiApi.h，获取RGB图像是通过底层动态库完成。

1. so提供的接口

int IKOV5640YUV\_OpenMIPICamera(char\* dev,int width,int height);

//获取图像数据,data空间长度至少为ImgSize

int IKOV5640YUV\_GetMIPIImage(unsigned char \*data);

int IKOV5640YUV\_CloseMIPICamera();//关闭摄像头

int IKOV5640YUV\_SetParmMIPICamera(int framerate,int mode,int input,int rotate,int imge\_format); //设置参数

int IKOV5640YUV\_GetVersion(char \*version); //获取5640的版本号

int IKOV5640YUV\_SetBrightness(int val); //设置分量参数---亮度

int IKOV5640YUV\_SetGamma(int val);//设置分量参数---伽马

int IKOV5640YUV\_SetContrast(int val); //设置分量参数---对比度

int IKOV5640YUV\_SetSharpness(int val); //设置分量参数---清晰度

int IKOV5640YUV\_ImgProcess\_num(int val); //可不用

int IKOV5640YUV\_GetIMGParameters(int \*gamma, int \*brightness, int \*contrast, int \*sharpness); //获取分量参数

1. 线程提供的接口

Int initARG(参数…….); // 初始化必要参数

Int getThreadState(); // 获取线程状态，是已经启动还是未启动

Int startThread(); // 启动线程，开始获取图像

Int stopThread(); // 停止线程，停止获取图像

1. 线程内部需要到函数

void run(); //线程主要运行内容在此函数

////下面的是对人脸进行识别和追踪相关的，参考边检项目中的kcftracker.cpp和detecttracking.cpp这两个文件中的代码。第三方项目KCF

int detectTracking(参数);

int kdKeyImResize(参数);

int softFaceDetect(参数);

int getImage()

其他…

1. 线程提供的信号

void sendNewFrame(QImage img);//将获取到的完整图像发出。

1. 私有变量

Volatile bool stopped; ///启动或停止标识

1. 摄像头异常处理

如果未采集到或者摄像头错误，如何处理。

## 3.6数据库访问类

1.说明

数据库访问类采用单例模式

2.线程提供的接口

static msqlDataBase\* get\_Instance();//获取单例模式数据库，返回数据库单例对象指针

static void Release();//释放数据库对象

int open();//打开数据库，并判断数据库表的完整性

void close();//关闭数据库

device数据表：

int setDeviceInfo(QStringList deviceInfo);//设置设备自身硬件信息

int getDeviceInfo(QStringList deviceInfo);//获取设备自身硬件信息

irisdatabase数据表：

int insertEnroll(PIRISUserInfo faceInfo);//插入虹膜信息

int getFeatureLR(unsigned char\* feature,

std::vector<int>& UID, int L\_R);//获取虹膜信息

QString delNameFeature(int UID);//删除虹膜信息

irisalgo数据表:

int insertEnrollPara(PIRISEnrPara enrollInfo);//插入虹膜注册参数

int getEnrollPara(PIRISEnrPara enrollInfo,

int UID);//获取虹膜注册参数

QString delEnrollPara(int UID);//删除虹膜注册参数

person数据表:

int addPersonInfo(PersonInfo personInfo);//添加用户信息

int getPersonInfo(PersonInfo personInfo,int UID);//获取用户信息

QString delPersonInfo(int UID);//删除用户信息

QStringList getAllID();//获取所有用户名和ID

reclog数据表:

int addRecLog(RecLog recLog);//添加人员识别记录

int getRecLog(RecLog recLog ,int UID);//获取人员的识别记录

3.线程内部使用的函数

msqlDataBase();//构造函数，定义了数据库的数据库名和路径，用户名和密码

## 3.7控制舵机运动线程

1. 线程不需要SDK接口

2. 线程类提供单例模式访问

3. 线程提供的接口

1) 初始化线程参数

static int initARG(const PWMParam& pwmParam);

输入参数：

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| pwmParam | PWMParam | 舵机参数 |

结构定义：

typedef struct \_\_PWMParam

{

int m\_minAngle; // 最小转动角度，缺省55

int m\_maxAngle; // 最大转动角度，缺省110

int m\_homeAngle; // 初始化角度（归位）

int m\_stepAngle; // 步进角度，缺省1

int m\_msDelay; // 每次角度设置完成后的等待时间，单位：毫秒，缺省100

}PWMParam, \*PPWMParam;

返回值：

E\_PWM\_SUCCESS（0），成功

E\_PWM\_PARAM\_ERROR（-0x2008），参数有误

2) 槽函数，转动舵机到指定角度，如果指定角度超过范围，则转动到极限位置

int setSteeringAngle (int angle, bool homing);

输入参数：

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| angle | int | 目标角度 |
| homing | bool | 是否转动到初始化角度，true表示转到初始化角度，angle参数无效；  缺省值为false |

返回值：

E\_PWM\_SUCCESS（0），成功

其他错误码请查看pwmdata.h中的定义

3) 槽函数，转动舵机一个相对于当前角度的偏移的角度，如果指定角度超过范围，则转动到极限位置

int setSteeringFrame (float offsetAngle, int id);

（这样定义只是为了与之前的使用方式保持一致，如果不需要保持一致，可以修改）

输入参数：

|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 说明 |
| offsetAngle | float | 在当前角度的情况下，转动的偏移角度 |
| id | int | 舵机通路id号，这是为了和之前使用的函数保持一致，使用缺省值0即可 |

返回值：

E\_PWM\_SUCCESS（0），成功

其他错误码请查看pwmdata.h中的定义

## 3.8 人脸检测线程

FaceTrack负责从RGB采图线程采集到的图像进行人脸检测。

# 四、定时器

4.1 电源管理类中的关机、重启定时器。

4.2识别结果消息对话框显示控制时间定时器

新版要求有定时退出的对话框，该定时器需要保留。

4.3DialogLogin用的是timerEvent来保证对话框消失时间

没有登陆对话框

4.4DialogSelectLogin用的是timerEvent来保证对话框消失时间

没有选择登陆对话框

4.5IKMessageBox关闭对话框用的定时器

是否能与4.3合并？

4.6 IkNetWork用timerEvent判断是否联网

联网判断，依然需要。

4.7其他界面返回到识别界面\_timerGoBackIdent定时器

界面返回，也需要。

4.8更新时间定时器

时间的计数器，需要

4.9人员信息界面，定时更新界面显示的人员

联网才需要

4.10Shutdown类中的延时关机和重启命令的定时器

参考4.2

4.11SysSettings类中更新时间的定时器

能否与4.8合并

4.12SysSettings类中同步服务器时间的定时器

定时同步设备上的时间，联网需要，定时时间暂定8小时。

# 五、依赖库

## 5.1 自编动态库

5.1.1 libdatabase.so

动态库版本号：未知

代码版本：0.0.1

动态库MD5值：2f5c1a716a0561096beaf735741156cd

功能：数据库访问的接口类，提供访问数据库操作，特别是用到的postgresql。

5.1.2libIKxmlAPI.so

动态库版本号：未知

代码版本：1.0.0

动态库MD5值：6117352161dfb659fe44c0baafe65e22

功能：提供解析XML文件的动态库，底层调用libxml2，版本号2.9.1

5.1.3 libIKUSBSDK.so

动态库版本号：未知

代码版本：未知

动态库MD5值：83603a29e5d5497a936a79bf4c66da30

功能：虹膜sdk中的USB操作接口

5.1.4 libusbapi.so

动态库版本号：未知

代码版本：1.0.0

动态库MD5值：41774dcf721efc58bc9560e06970050f

功能：访问USB操作接口，依赖libusb.a

5.1.5 libIKAlgAPI.so

动态库版本号：未知

代码版本：未知

动态库MD5值：90ffe87eda06bd669267088dd7d01113

功能：SDK虹膜算法API库

5.1.6 libIKAlgoIR.so

动态库版本号：未知

代码版本：未知

动态库MD5值：0357ce45155773cfa1011eb747c5311b

功能：SDK虹膜算法IR库

5.1.7 libIKAlgoFR.so

动态库版本号：未知

代码版本：未知

动态库MD5值：b6745032260766cb9e19e944d24bff39

功能：SDK虹膜算法FR库

5.1.8 libFaceCamera.so

动态库版本号：未知

代码版本：1.0.0

动态库MD5值：650fc325acbd31a5e52ad1938d61ad61

功能：人脸采集摄像头库

5.1.9 libsocketcomm.so

动态库版本号：未知

代码版本：1.0.0

动态库MD5值：50ead417fad37e52f97938fb352f7308

功能：网络基础库

5.1.10 libikIOStream.so

动态库版本号：未知

代码版本：1.0.0

动态库MD5值：73729443e06517d00054d8bc503b3426

功能：对数据库进行数据的导入导出操作的库。

5.1.10libSerialMuxDemux.so

动态库版本号：未知

代码版本：未知

动态库MD5值：c382a123ac6a881f24a71d58d3f7f494

功能：对串口进行操作。

5.1.11 libInPut.so

动态库版本号：未知

代码版本：未知

动态库MD5值：195df2ccfe77fb708a5a89955ce25a7a

功能：

5.1.11 libIrisManager.so

动态库版本号：未知

代码版本：1.0.0

动态库MD5值：32207edf080f199e89a77db957ab1a0a

功能：虹膜特征管理类

5.1.12 libSDL\_mixer-1.2.so.0

动态库版本号：1.2

代码版本：1.2

动态库MD5值：629242958aeb4f5622b76ce4f7628fb5

功能：SDL声音处理动态库

5.1.13 libSDL-1.2.so.0

动态库版本号：1.2

代码版本：1.2

动态库MD5值：5a12bd74ce447117bb946d617ccd0b1e

功能：SDL基本动态库

5.1.14 libjpeg.so.8

动态库版本号：V8

代码版本：V8

动态库MD5值：23a7d55d2c51d16edbab627d371da7a5

功能：jpeg图像解析库

5.1.15liblog4cpp.so.5

动态库版本号：

代码版本：

动态库MD5值：082d2274fb441f3a3239d2db4c152af6

功能：C++的日志库

5.1.16 libsoci\_core.so.3.2

动态库版本号：3.2.2

代码版本：3.2.3

动态库MD5值：代码中提供的是34b82d44789d98bd630715f616aeb2e7

设备上是c4c8c7cacb655e6a1ff511425708393d

功能：soci核心库

5.1.17libpq.so.5

动态库版本号：5

代码版本：9.3

动态库MD5值：代码中提供的是5318a846be03e578a523db693b317eb8

设备上是5e2990e22d8e2282a93e783e54220b9e

功能：postgresql的动态库

## 5.2 自编静态库

liblog4cpp.a

## 5.3 Qt库

libQtGui.so.4、libQtCore.so.4、libQtNetwork.so.4、libQtSql.so.4、libQtDBus.so.4、libQtXml.so.4等等

## 5.4其他系统库

libuuid.so.1、linux-gate.so.1、libpthread.so.0、libstdc++.so.6、libgcc\_s.so.1、libc.so.6、libm.so.6、libmikmod.so.2、libfluidsynth.so.1、libvorbisfile.so.3、libFLAC.so.8、libmad.so.0、libasound.so.2、libdl.so.2、libpulse-simple.so.0、libpulse.so.0、libX11.so.6、libXext.so.6、libcaca.so.0、libssl.so.1.0.0、libcrypto.so.1.0.0、libkrb5.so.3、libcom\_err.so.2、libgssapi\_krb5.so.2、libldap\_r-2.4.so.2、libdbus-1.so.3、libfontconfig.so.1、libaudio.so.2、libglib-2.0.so.0、libpng12.so.0、libz.so.1、libfreetype.so.6、libgobject-2.0.so.0、libSM.so.6、libICE.so.6、libXi.so.6、libXrender.so.1、librt.so.1、libjack.so.0、libsndfile.so.1、libreadline.so.6、libvorbis.so.0、libogg.so.0、libpulsecommon-4.0.so、libjson-c.so.2、libxcb.so.1、libslang.so.2、libncursesw.so.5、libtinfo.so.5、libk5crypto.so.3、libkrb5support.so.0、libkeyutils.so.1、libresolv.so.2、liblber-2.4.so.2、libsasl2.so.2、libgssapi.so.3、libgnutls.so.26、libgcrypt.so.11、libexpat.so.1、libXt.so.6、libXau.so.6、libpcre.so.3、libffi.so.6、libvorbisenc.so.2、libwrap.so.0、libasyncns.so.0、libXdmcp.so.6、libheimntlm.so.0、libkrb5.so.26、libasn1.so.8、libhcrypto.so.4、libroken.so.18、libtasn1.so.6、libp11-kit.so.0、libgpg-error.so.0、libnsl.so.1、libwind.so.0、libheimbase.so.1、libhx509.so.5、libsqlite3.so.0、libcrypt.so.1

# 六、数据库

数据库采用sqlite，预计单表几十万记录。

## 6.1 device数据表

本表保存设备自身硬件信息，只有一条记录。

表格2

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 格式类型 | 必填 | 说明 |
| devid | Integer | 是 | 整形 |
| devname | varchar(32) | 否 | 设备名称 |
| devsn | varchar(18) | 是 | 设备sn，由生产时确定 |
| devip | varchar(16) | 否 | 设备IP |
| devmac | varchar(18) | 是 | 设备Mac |
| devsubnet | varchar(16) | 否 | 设备子网掩码 |
| devserverip | varchar(16) | 否 | 设备连接服务器IP |
| devserverport | Integer | 否 | 设备连接服务器端口号 |
| devplace | varchar(64) | 否 | 设备地址 |
| memo | text | 否 |  |

## 6.2irisdatabase数据表

本表保存的是虹膜数据。

表格3

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 格式类型 | 必填 | 说明 |
| irisdataid | varchar(38) | 是 | uuid,本表的唯一识别号 |
| personid | varchar(38) | 是 | 人员的uuid,对应5.4数据表中的personid |
| groupid | INTEGER | 是 | 组别，一个人最多4组数据，从1开始 |
| iriscode | BLOB | 是 | 虹膜特征, 1000字节 |
| eyepic | BLOB | 否 | 人眼的虹膜图像 |
| matchiriscode | BLOB | 否 | 虹膜比对特征1536字节 |
| devsn | varchar(32) | 是 | 设备sn，由生产时确定 |
| eyeflag | INTEGER | 是 | 左右眼标志，左眼1，右眼2 |
| regtime | datetime | 是 | 注册时的设备时间,对于同一个人的不同特征都是相同的 2017-10-15 10:15:10:123 |
| createtime | datetime | 是 | 本记录的创建时间 |
| updatetime | datetime | 否 | 更新时间 |

## 6.3 irisalgo数据表

本表保存注册虹膜时涉及到算法的一些参数，便于后续查问题，其中的irisdataid、personid、regtime和createtime必需与前表相同。

表格4

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 格式类型 | 必填 | 说明 |
| irisdataid | varchar(38) | 是 | uuid，本表的唯一识别号 |
| personid | varchar(38) | 是 | 人员的uuid,对应5.4数据表中的personid |
| regtime | datetime | 是 | 注册时的设备时间,对于同一个人的不同特征都是相同的 2017-10-15 10:15:10:123 |
| eyeflag | INTEGER | 是 | 左右眼标志，左眼1，右眼2 |
| groupid | INTEGER | 是 | 组别，一个人最多4组数据，从1开始 |
| createtime | datetime | 是 | 本记录的创建时间 |
| updatetime | datetime | 否 | 更新时间 |
| memo | text | 否 |  |
| pupilrow | Integer | 否 | 瞳孔圆心行,没有就填写0 |
| pupilcol | Integer | 否 | 瞳孔圆心列,没有就填写0 |
| pupilradius | Integer | 否 | 瞳孔半径,没有就填写0 |
| irisrow | integer | 是 | 虹膜圆心行 |
| iriscol | integer | 是 | 虹膜圆心列 |
| irisradius | Integer | 是 | 虹膜半径 |
| focusscore | Integer | 是 | 聚焦分数,清晰度 |
| percentvisible | Integer | 是 | 可见百分比 |
| spoofvalue | Integer | 否 | 可信值, 防伪,没有就填写0 |
| interlacevalue | Integer | 是 | 交错值 |
| qualitylevel | Integer | 否 | 质量水平,没有就填写0 |
| qualityscore | integer | 是 | 质量分数 |

## 6.4person数据表

表格5

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 格式类型 | 必填 | 说明 |
| personid | varchar(38) | 是 | 用uuid标识人员的唯一性 |
| name | varchar(256) | 是 | 姓名,用户名 |
| id | varchar(64) | 是 | 用户id，唯一 |
| superid | integer | 是 | 是否是管理员: 0不是，1是 |
| sex | varchar(4) | 否 | 性别 |
| cardid | varchar(16) | 否 | 卡号字段，此字段前面补0，凑齐8位 |
| vcardid | varchar(16) | 否 | 虚拟卡号字段，此字段前面补0，凑齐8位，如果没有实体卡，此虚拟卡号用作韦根输出。 |
| idcard | varchar(20) | 否 | 身份证号码 |
| memo | text | 否 |  |
| profession | varchar(32) | 否 | 职业 |
| hasirisdata | Integer | 是 | 是否有虹膜数据：1--有,0—没有 |
| address | text | 否 | 常用住址 |
| password | varchar(16) | 否 | 用户密码，只有superid为1（表示是管理员）才可以设置密码 |
| createtime | datetime | 是 | 用户创建的时间，用户排序等 |
| updatetime | datetime | 是 | 用户修改编辑的时间 |

保存人员信息表，

## 6.5reclog数据表

表格6

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 格式类型 | 必填 | 说明 |
| reclogid | integer | 是 | Integer |
| personid | varchar(38) | 是 | 人员表的uuid |
| irisdataid | varchar(38) | 是 | 虹膜表uuid |
| recogtime | datetime | 是 | 识别时间 |
| recogtype | Integer | 是 | 识别类别，人脸0、虹膜1、卡号2、密码3 |
| devsn | varchar(32) | 是 | 设备sn，由生产时确定 |
| memo | text | 否 |  |

保存人员识别记录， reclogid是自增字段，在填写数据时由数据库自增即可。

6.6 管理员操作数据表

保存管理员进行的每次操作，编辑员工，新加员工，删除员工，注册虹膜，删除虹膜，卡号的操作，管理员密码的操作，管理员的登陆和退出。

# 七、配置文件

## 7.1 应用程序配置文件

对应用程序进行配置，配置内容包括，界面大小的截取参数，校准设置，摄像头舵机控制配置参数，红外灯光舵机控制参数和存疑人员jpg保存路径参数等等。

部分配置文件内容是否会改到程序界面中待定。

# 八、代码注释说明

涉及到中文建议均采用utf8格式，仅针对Qt开发过程中的注释。

## 8.1 文件的概述说明

在\*.h和\*.cpp文件最上面要加入如下说明，根据实际情况编写描述内容

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Copyright: 中科虹霸

作者: lhc

创建日期: 2017-05-12

描述: 虹膜注册识别类

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

## 8.2 h文件说明

注释说明均采用单行说明，注释写在被注释的上方或右侧，举例如下：

例1：

//标识眼睛选项

enum EyeFlag{

F\_IR\_IM\_EYE\_UNDEF = 0, //Unknown left or right

F\_IR\_IM\_EYE\_LEFT = 1, //Left image

F\_IR\_IM\_EYE\_RIGHT = 2, //Right image

F\_IR\_IM\_EYE\_BOTH = 3, //get both left and right eye

F\_IR\_IM\_EYE\_NONE = -1, //NO iris image is presented

F\_IR\_IM\_Face = 4 //人脸图形

};

例2：

long int m\_img\_count; //图像数目

例3：

//获取字符串的时间，格式月-日-时-分-秒-毫秒

QString getDateTime();

## 8.3 cpp文件说明

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 获取日期时间函数

\* 函 数 名： getDateTime

\* 功 能：获取字符串的时间，格式月-日-时-分-秒-毫秒

\* 说 明：

\* 参 数：无

\* 返 回 值：时间日期的字符串QString

\* 创 建 人：zhangyp

\* 创建时间：2017-05-16

\* 修 改 人：

\* 修改时间：

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

QString CIRISEngine::getDateTime()

{……

}

# 九、其他

## 9.1 界面名称

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 界面名称 | 控件名 | 备注 |
| 1 | 虹膜识别或者人脸识别界面 | PA\_RGB | 参见2.2、2.5.1、2.5.2 |
| 2 | 屏保界面 | PA\_Screen | 参见2.1 |
| 3 | 管理界面 | PA\_Manage | 参见2.4 |
| 4 | 密码验证登陆界面 | PA\_Login | 参见2.5.3 |
| 5 | 用户管理主界面 | PA\_Users | 参见2.6.1 |
| 6 | 用户管理搜索界面 | PA\_Users\_Search | 参见2.6.2 |
| 7 | 用户管理添加界面 | PA\_Users\_Add | 参见2.6.4 |
| 8 | 用户管理填写更多界面 | PA\_Users\_AddMore | 参见2.6.4 |
| 9 | 用户管理详情界面 | PA\_Users\_Info | 参见2.6.5 |
| 10 | 系统设置界面 | PA\_Settings | 参见2.7 |
| 11 | 系统设置界面基本设置 | PA\_S\_Set | 参见2.7.1 |
| 12 | 系统设置界面修改时间 | PA\_S\_ModifyTime | 参见2.7.1 |
| 13 | 系统设置界面系统信息 | PA\_S\_SysInfo | 参见2.7.1 |
| 14 | 系统设置界面系统容量 | PA\_S\_SysCap | 参见2.7.1 |
| 15 | 系统设置界面设备信息 | PA\_S\_SysDevice | 参见2.7.1 |
| 16 | 系统设置界面接口设置 | PA\_S\_IO | 参见2.7.2 |
| 17 | 系统设置界面验证识别设置 | PA\_S\_Verify | 参见2.7.3 |
| 18 | 系统设置界面通信网络设置 | PA\_S\_Net | 参见2.7.4 |
| 19 | 系统设置界面数据管理 | PA\_S\_DataManage | 参见2.7.5 |
| 20 | 系统设置界面初始化设置 | PA\_S\_Init | 参见2.7.6 |

## 9.2常用控件命名

|  |  |  |
| --- | --- | --- |
| 类型 | 缩写 | 备注 |
| QPushButton | m\_PB\_ |  |
| QLabel | LB |  |
| QLineEdit | LE |  |
| QProgressBar | PGB |  |
| QTabWidget | TabW |  |
| QTableWidget | TableW |  |
| QTableView | TableV |  |
| QCheckBox | CB |  |
|  |  |  |