系统分析

1、可行性分析

（1）、QZXing：Qt包装ZXing的解码库。ZXing是条形码处理类库，它是一个开源Java类库用于解析多种格式的1D/2D条形码。包含了联系到其他语言的端口。

（2） Opencv：OpenCV(开源的计算机视觉库)是BSD许可下发布的,因此它是免费的学术和商业用途。c++,C,Python和Java接口,支持Windows,Linux,Mac OS,iOS和Android。

2、效率分析

（1）、QXZing： 最长20秒内识别到二维码。

（2）、OpenCV：OpenCV是为计算效率而设计的，并且专注于实时应用程序用，该库采用优化的c/c++编写，可以利用多核处理。

3、准确性分析

（1）、QXZing：识别二维码效率达到98%以上。

系统设计

1、HID通讯模块 CHidCmdThread

功能：HID命令排队操作线程，HID命令添加请一律通过该类进行处理

包含类

class ProtocolUtility 通讯协议类

class HIDOpertaionUtility USB操作类

class HIDReadThread USB读线程类

类关系



2、二维码识别模块 QRCodeDetector

功能：从图片中提取二维码并识别、解析二维码数据，发送到测试线程。

包含类

class Hidopertaionutility USB操作类

class Protocolutility 通讯协议类

class Opencvutility Opencv操作类

class QZXing 二维码解码类

class CHidCmdThread HID通讯类

类关系



3、标准机亮度校正模块 StandardBrightness

功能：输入标准机灯光值，获取绿色分量曲线数据

包含类：

class CHidCmdThread HID通讯类

class Hidopertaionutility USB操作类

class Protocolutility 通讯协议类

class Opencvutility Opencv操作类

class ParamsConfiguration 读写配置文件类

class ThreadTesting 测试类

类关系：



4、目标机亮度校正模块OrdinaryBrightmess

功能：寻找目标机灯光值，使目标机灯光效果与标准机一致

包含类

class ParamsConfiguration 读写配置文件类

class CalculateGrayscale 计算绿色分量类

class CHidCmdThread HID通讯类

class ProtocolUtility 通讯协议类

class HIDOpertaionUtility USB操作类

class OpencvUtility Opencv操作类

class LineFinder 寻找直线类

类关系：



系统实现

1、HID通讯模块 CHidCmdThread

1.1功能：HID命令排队操作线程，HID命令添加请一律通过该类进行处理

1.2数据结构

struct HIDCmdData 命令类型和命令数据集合

struct HIDCmdData

{

quint16 cmdType;//命令类型

QVector<QByteArray> byteArrayVect;//命令数据集合

}

1.3私有成员变量

static CHidCmdThread\* s\_hidCmdThreadInstance;通讯模块静态对象

QQueue<HIDCmdData> m\_hidCmdDataQueue; 上位机往设备下发命令队列

HIDCmdData m\_curHIDCmdData; 目前正在执行的命令数据

DevConfigParams m\_devConfigParams; 仪器参数

1.4公有成员方法

/\*\*

\* @brief ClearCmd清空HID命令

\* @return

\*/

void ClearCmd();

/\*\*

\* @brief AddCmd 添加HID命令

\* @param hidCmdData 命令类型和命令数据集合

\* @return

\*/

void AddCmd(HIDCmdData hidCmdData);

/\*\*

\* @brief AddCloseHIDCmd 添加关闭HID命令

\* @return

\*/

void AddCloseHIDCmd();

/\*\*

\* @brief AddResetMotorCmd 添加马达复位命令

\* @param resetSpeed 复位速度

\* @return

\*/

void AddResetMotorCmd(quint16 resetSpeed);

/\*\*

\* @brief AddRotateMotorCmd 添加马达转动命令

\* @param speed 转动速度

\* @param step 转动步数

\* @param direction 转动方向

\* @return

\*/

void AddRotateMotorCmd(quint16 speed, quint16 step, quint16 direction);

/\*\*

\* @brief AddOpenLedCmd 添加开灯命令

\* @param iLedIndex 灯类型

\* @param iBrightness 灯光值

\* @return

\*/

void AddOpenLedCmd(int iLedIndex, quint16 iBrightness);

/\*\*

\* @brief AddWriteDevParamsCmd 添加写取设备参数命令

\* @param devConfigParams 设备参数结构体

\* @return

\*/

void AddWriteDevParamsCmd(DevConfigParams devConfigParams);

/\*\*

\* @brief AddCmdWithoutCmdData 添加不带命令数据的命令

\* @param qCmdType 命令类型

\* @return

\*/

void AddCmdWithoutCmdData(quint16 qCmdType);

/\*\*

\* @brief AddCmdWithoutCmdData 添加升级命令

\* @param qFilePathStr 升级文件路径

\* @return

\*/

void AddUpgradeSubControlCmd(QString qFilePathStr);

1.5私有槽

/\*\*

\* @brief AddCmdWithoutCmdData HID命令结果返回信号处理槽

\* @param cmdType 命令类型

\* @param result 命令结果

\* @return

\*/

void SlotHIDCmdComplete(quint16 cmdType ,bool result);

1.6包含类

1.6.1 class ProtocolUtility

1.6.1.1功能：封装上下位机通信协议

1.6.1.2数据结构

enum Direction 电机转动方向

1.6.1.3成员变量

//上位机设备地址：0x0100

static const quint8 PC\_ADDR\_HIGH = 0x01;

static const quint8 PC\_ADDR\_LOW = 0x00;

//下位机设备地址：0x0101

static const quint8 DEV\_ADDR\_HIGH = 0x01;

static const quint8 DEV\_ADDR\_LOW = 0x00;

//命令类型

static const quint16 CMD\_DEV\_CLOSE = 0x0000;//关闭设备HID通信

static const quint16 CMD\_OPEN\_OR\_CLOSE\_LED = 0x0001;//LED灯开关命令

static const quint16 CMD\_ROTATE\_MOTOR = 0x0002;//电机转动命令

static const quint16 CMD\_RESET\_MOTOR = 0x0003;//电机复位指令

static const quint16 CMD\_CLOSE\_ALL\_LED = 0x0004;//关闭所有灯指令

static const quint16 CMD\_CLOSE\_ALL\_LED\_AND\_STOP\_MOTOR = 0x0005;//关闭所有灯与电机立刻停止

static const quint16 CMD\_WRITE\_PARAM\_TO\_DEV = 0x0006;//写仪器参数

static const quint16 CMD\_READ\_PARAM\_FROM\_DEV = 0x0007;//读取仪器参数

static const quint16 CMD\_READ\_DEV\_VERSION = 0x0008;//读取仪器下位机软件版本

static const quint16 CMD\_ADD\_TEST\_COUNT = 0x0009;//仪器测量次数加1

static const quint16 CMD\_READ\_TEST\_COUNT = 0x000A;//仪器测量次数读取

static const quint16 CMD\_CLEAR\_TEST\_COUNT = 0x000B;//仪器测量次数清零

static const quint16 CMD\_UPGRADE\_APP\_START = 0x000C;//仪器下位机程序升级开始

static const quint16 CMD\_UPGRADE\_APP\_DATA = 0x000D;//仪器下位机程序升级数据

static const quint16 CMD\_UPGRADE\_APP\_END = 0x000E;//仪器下位机程序升级结束

//命令参数

static const quint8 COMMON\_PARAM\_HIGH = 0x00;

static const quint8 COMMON\_PARAM\_LOW = 0x01;

1.6.1.4 成员方法

/\*\*

\* @brief GetLEDCmd LED灯开关命令

\* @param ledIndex 灯类型

\* @param brightness 灯光值

\* @return

\*/

static QByteArray GetLEDCmd(quint16 ledIndex, quint16 brightness);

/\*\*

\* @brief GetMotorRotateCmd 电机转动命令

\* @param direction 方向

\* @param step 转动步数

\* @param speed 运动速度

\* @return

\*/

static QByteArray GetMotorRotateCmd(quint16 direction, quint16 step, quint16 speed);

/\*\*

\* @brief GetMotorResetCmd 电机复位命令

\* @param speed 复位速度

\* @return

\*/

static QByteArray GetMotorResetCmd(quint16 speed);

/\*\*

\* @brief GetCloseAllLEDCmd

\* 关闭所有灯指令

\* @return

\*/

static QByteArray GetCloseAllLEDCmd();

/\*\*

\* @brief GetCloseAllLEDAndStopMotorCmd

\* 关闭所有灯与电机立刻停止

\* @return

\*/

static QByteArray GetCloseAllLEDAndStopMotorCmd();

/\*\*

\* @brief GetReadParamsFromDevCmd

\* 读取仪器参数并记忆(总共包含10个包)

\* @return

\*/

static QVector<QByteArray> GetReadParamsFromDevCmd();

/\*\*

\* @brief GetWriteParamFromDevCmd

\* 写仪器参数到设备(总共包含10个包)

\* @param devConfigParams 参数结构体

\* @return

\*/

static QVector<QByteArray> GetWriteParamFromDevCmd(DevConfigParams devConfigParams);

/\*\*

\* @brief GetReadVersionCmd

\* 读取仪器下位机软件版本

\* @return

\*/

static QByteArray GetReadVersionCmd();

/\*\*

\* @brief GetAddTestTimeCmd

\* 仪器测量次数加一命令

\* @return

\*/

static QByteArray GetAddTestCountCmd();

/\*\*

\* @brief GetReadTestTimeCmd

\* 读取仪器测量次数

\* @return

\*/

static QByteArray GetReadTestCountCmd();

/\*\*

\* @brief GetClearTestTimeCmd

\* 仪器测量次数清零

\* @return

\*/

static QByteArray GetClearTestCountCmd();

/\*\*

\* @brief GetUpgradeAppStartCmd

\* 仪器下位机程序升级开始命令

\* @return

\*/

static QByteArray GetUpgradeAppStartCmd();

/\*\*

\* @brief GetUpgradeAppEndCmd

\* 仪器下位机程序升级结束命令

\* @return

\*/

static QByteArray GetUpgradeAppEndCmd();

/\*\*

\* @brief GetUpgradeAppCmd

\* 仪器下位机程序升级数据接收(包的总数根据升级文件大小而定)

\* @return

\*/

static QVector<QByteArray> GetUpgradeAppCmd(QString filePath);

1.6.1.5 私有成员方法

/\*\*

\* @brief SetCmdLength

\* 设置命令长度

\* @param dataByteArray 命令数据

\* @param

\*/

static void SetCmdLength(QByteArray& dataByteArray);

/\*\*

\* @brief DealWithCmdEnding

\* 填充命令公共收尾部分

\* @param dataByteArray

\* @param out

\*/

static void DealWithCmdEnding(QByteArray& dataByteArray, QDataStream& out);

/\*\*

\* @brief DealWithCmdHead

\* 填充命令公共头部

\* @param out

\*/

static void DealWithCmdHead(QDataStream& out);

/\*\*

\* @brief GetCmdByteArrayWithoutCmdData

\* 获取命令中不包含命令数据的命令ByteArray

\* @param qCmdType 命令类型，默认命令参数为0x0001

\* @return

\*/

static QByteArray GetCmdByteArrayWithoutCmdData(quint16 qCmdType);

1.6.2 class HIDOpertaionUtility

1.6.2.1 功能：操作USB接口

1.6.2.2 私有成员变量：

static HIDOpertaionUtility\* instance; // HIDOpertaionUtility实例

HANDLE mHidHandle;//USB hid 句柄

QThread mWorkThread;//工作线程，构造函数内使用moveToThread转到后台线程操作 Qt::HANDLE mWorkHandle;//工作线程句柄

volatile bool mIsDeviceOpened;//设备是否已打开

HIDReadThread\* mReadThread;//设备读线程

bool mAckResult;//ACK结果

bool mIsWaitForAck;//是否正在等待ACK,用于读写两个线程的信号同步

bool mReturnResult;//结果

bool mIsWaitForReturn;//是否正在等待结果，用于读写两个线程的信号同步

quint16 m\_iCmdType;//当前处理命令的命令类型

QString mDevVersion;//设备版本

DevConfigParams mParams;//设备参数

quint8\* mDevConfigParamsByte;//配置信息字节数组，用于每次从设备读取配置参数时临时存储数据

quint32 m\_qTestCount;//仪器测试次数

1.6.2.3 公有成员方法：

/\*\*

\* @brief GetInstance获取HIDOpertaionUtility实例

\* @retrun HIDOpertaionUtility实例指针

\*/

static HIDOpertaionUtility\* GetInstance();

/\*\*

\* @brief HIDWrite 目前适用于非读/写设备参数的其他控制指令

\* HID 写

\* @param writeByteArray 写入的数据

\*/

void HIDWrite(QByteArray writeByteArray);

/\*\*

\* @brief HIDOpen

\* HID 打开

\*/

void HIDOpen();

/\*\*

\* @brief HIDClose

\* 停止关闭操作比较特殊，需要立刻停止，这样之前的排队命令会被取消等待直接进行停止操作

\*

\* @param execAtHIDOpertionThread

\*/

void HIDClose();

/\*\*

\* @brief HIDRead HID读数据

\* @param recvDataBuf 存放读出来的数据地址

\* @param delaytime 读超时

\*/

bool HIDRead(quint8\* recvDataBuf, int delaytime);

/\*\*

\* @brief IsDeviceOpen

\* HID 是否打开

\* @return

\*/

bool IsDeviceOpen();

/\*\*

\* @brief ReceiveNewCmdFromDev

\* 处理从设备接收到的命令数据，处理在读取线程中

\* @param data

\*/

void ReceiveNewCmdFromDev(QByteArray data);

/\*\*

\* @brief GetVersion

\* 获取设备版本

\* @return

\*/

QString GetVersion();

/\*\*

\* @brief HIDReadDevParams

\* 读设备参数命令

\* @return

\*/

void HIDReadDevParams();

/\*\*

\* @brief HIDWriteDevParams

\* 写设备参数命令

\* @param devConfigParams

\* @return

\*/

void HIDWriteDevParams(DevConfigParams devConfigParams);

/\*\*

\* @brief HIDUpgradeSubControl

\* @param filePath

\* @return

\*/

void HIDUpgradeSubControl(QString filePath);

1.6.2.4 信号：

/\*\*

\* @brief SignalHIDOpen 打开 HID信号

\* @return

\*/

void SignalHIDOpen();

/\*\*

\* @brief SignalHIDClose 关闭 HID信号

\* @return

\*/

void SignalHIDClose();

/\*\*

\* @brief SignalHIDWrite HID写数据信号

\* @param writeByteArray 写入的数据

\* @return

\*/

void SignalHIDWrite(QByteArray writeByteArray);

/\*\*

\* @brief SignalReceiveDevVersion HID读取下位机版本信号

\* @param devVersion 版本号

\* @return

\*/

void SignalReceiveDevVersion(QString devVersion);

/\*\*

\* @brief SignalReadDevParams HID读取参数数据信号

\* @return

\*/

void SignalReadDevParams();

/\*\*

\* @brief SignalErrInfo HID状态信号

\* @param qErrorMsg 错误类型

\* @return

\*/

void SignalErrInfo(EnumTypeErr qErrorMsg);

/\*\*

\* @brief SignalOperationComplete HID命令完成信号

\* @param m\_iCmdType 命令类型

\* @param result 命令结果

\* @return

\*/

void SignalOperationComplete(quint16 m\_iCmdType,bool result);

1.6.2.5 私有槽

bool SlotOpen();// 打开USB HID

/\*\*

\* @brief Close

\* 关闭USB HID

\* @return

\*/

bool SlotClose();

/\*\*

\* @brief Write

\* @param writeByteArray

\* 写命令到USB HID

\* @return

\*/

bool SlotWrite(QByteArray writeByteArray);

/\*\*

\* @brief ReadDevParams

\* @return

\*/

bool SlotReadDevParams();

/\*\*

\* @brief WriteDevParams

\* @param devConfigParams

\* @return

\*/

bool SlotWriteDevParams(DevConfigParams devConfigParams);

/\*\*

\* @brief SlotUpgradeSubControl

\* 升级下位机

\* @param filePath 升级文件路径

\* @return

\*/

bool SlotUpgradeSubControl(QString filePath);



1.6.3 class HIDReadThread

1.6.3.1 功能：USB读线程

2、 二维码识别模块 QRCodeDetector

2.1 功能：从图片中提取二维码并识别、解析二维码数据，发送到测试线程。

2.2 数据结构

enum EnumTypeCup 杯类型

enum EnumTypeLight 灯光类型

struct InfoProject 项目信息

struct QRCodeInfo 二维码信息

2.3 私有成员变量

int m\_iQRCodePosition; 二维码定位后，距离复位位置的距离（顺时针转到复位位置的步数）

QZXing \*m\_pZxingDecoder; 二维码解析库对象指针

2.4 公有成员方法

/\*\*

\* @brief InitDevice 设备初始化

\* @return

\*/

bool InitDevice();

/\*\*

\* @brief TestGetQRCode

\* 获取二维码信息，发送获取二维码信号

\* @param

\* @return

\*/

void TestGetQRCode();

/\*\*

\* @brief locationQRCode

\* 定位和识别二维码

\* @param strQRCodeInfo 返回识别到的二维码字符串

\* @param iQRCodePosition 二维码距离复位位置的距离

\* @return

\*/

bool locationQRCode(QString &strQRCodeInfo,qint32 &iQRCodePosition);

/\*\*

\* @brief GetQRCodeImage

\* 获取二维码照片

\* @param strImagePath 输出照片保存的路径

\* @return

\*/

bool GetQRCodeImage(QString &strImagePath);

/\*\*

\* @brief GetQRCodeImageInfo

\* 识别二维码

\* @param strImagePath 输入照片保存的路径

\* strQRCodeInfo 输出解析到的二维码字符串

\* @return

\*/

bool GetQRCodeImageInfo(const QString strImagePath,QString &strQRCodeInfo);

/\*\*

\* @brief DecodeQrcode

\* 根据V2或V5版本协议 解析二维码字符串

\* @param strImagePath 输出照片保存的路径

\* qrCodeInfo 输出解析后的二维码信息

\* @return

\*/

bool DecodeQrcode(const QString strdecode,QRCodeInfo &qrCodeInfo);

/\*\*

\* @brief DecodeQrcode

\* 根据V2或V5版本协议 解析二维码字符串

\* @param strImagePath 输出照片保存的路径

\* qrCodeInfo 输出解析后的二维码信息

\* @return

\*/

bool FindQRcodeLocationRect(IplImage \*dstImg);

/\*\*

\* @brief TestLightUp

\* 开灯

\* @param type 灯类型

\* @return

\*/

//开灯 成功 0 失败-1

int TestLightUp(EnumTypeLight type);

/\*\*

\* @brief TestLightDown

\*关灯

\* @return

\*/

int TestLightDown();

/\*\*

\* @brief PackageQRCodeInfo

\* 封装二维码信息

\* @param strBatchNumber, //项目批号

\* @param strValidityData, //有效期

\* @param strCardNumber, //卡序列号

\* @param strCupType, //杯型

\* @param strAllCount //总子条数

\* @param strVersion 版本

\* @param qv\_strListitem 项目信息

\* @return

\*/

QRCodeInfo PackageQRCodeInfo(QString strBatchNumber, //项目批号

QString strValidityData, //有效期

QString strCardNumber, //卡序列号

QString strCupType, //杯型

QString strAllCount, //总子条数

QString strVersion, //版本

QVector<QStringList> qv\_strListitem); //项目信息

/\*\*

\* @brief GetProjectName

\*解析项目名称

\* @param iIndex, //项目序号

\* @return QString //项目名称

\*/

QString GetProjectName(const int iIndex);

/\*\*

\* @brief mSleep

\* 延时

\* @param msec 延时的时间（毫秒）

\* @return

\*/

void mSleep(qint32 msec);

/\*\*

\* @brief ExtractQRCode

\* 提取二维码

\* @param strSrcImage 输入照片的路径

\* @param strDesImage 输出二维码图片

\* @return

\*/

bool ExtractQRCode(QString strSrcImage,QString &strDesImage);

2.5信号

/\*\*

\* @brief SignalQRCodeLocation

\* 发送二维码图片路径信号

\* @param pathPic照片的路径

\* @return

\*/

void SignalQRCodeLocation(QString pathPic);

/\*\*

\* @brief SignalQRCodeInfo

\* 发送二维码信息信号

\* @param info二维码信息结构体

\* @return

\*/

void SignalQRCodeInfo(QRCodeInfo info);

/\*\*

\* @brief SignalErrInfo

\* 发送错误信息

\* @param eErr错误类型

\* @return

\*/

void SignalErrInfo(EnumTypeErr eErr);

/\*\*

\* @brief SignalGetQRCode

\* 发送开始获取二维码信号

\* @return

\*/

void SignalGetQRCode();



2.6包含类

2.6.1 class Hidopertaionutility

2.6.1.1 功能：操作USB接口

2.6.2 class Protocolutility

2.6.2.1 功能： 封装通讯协议类

2.6.3 class CHidCmdThread

2.6.3.1 功能：HID命令排队操作线程，HID命令添加请一律通过该类进行处理

2.6.4 class Opencvutility

2.6.4.1 功能：Opencv操作类，对摄像头和图片进行操作

2.6.4.2 私有成员变量

VideoCapture m\_VideoCapture; 视频操作类对象

static OpencvUtility\* instance; opencv操作类对象指针

2.6.4.4 公有成员方法

/\*\*

\* @brief getInstance

\* 获取 opencv操作类对象指针

\* @return

\*/

static OpencvUtility\* getInstance();

/\*\*

\* @brief OpenVideo

\* 打开摄像头

\* @param

\*

\* @return

\*/

bool OpenVideo();

/\*\*

\* @brief GetVideo

\* 录像

\* @param

\*

\* @return

\*/

bool GetVideo();

/\*\*

\* @brief GetVideoCapture

\* 拍照

\* @param strImagePath 输出图片保存的路径

\*

\* @return

\*/

bool GetVideoCapture(QString \*strImagePath);

2.6.4.5 信号

/\*\*

\* @brief SignalErrInfo

\* 发送错误信号

\* @param eErr 错误信号类型

\* @return

\*/

void SignalErrInfo(EnumTypeErr eErr);



2.6.5 class QZXing

2.6.5.1 功能：解析二维码