利用socket和UDP协议进行电脑摄像头画面和图像识别结果的实时传送：分为send客户端和receive服务器端。

Send客户端：（**VideoSend**()函数）

1、获取本机的IP地址和端口号。

（1）通过用户在ui界面lineEdit组件中输入的ip地址和端口号，利用text()函数获取。

（2）将ip地址放入QHostAddress dstip中，端口号放入quint16 dstport中。

2、使用OpenCv VideoCapture类 + Qtimer定时器实现简单的摄像头调用。

（1）创建Mat frame用于存储和处理摄像头图像画面。创建VideoCapture camera用于从摄像头中读取数据。创建QTimer fps\_timer实现定时器超时处理功能。

（2）调用camera.read(frame)函数，得到单帧图像画面。

（3）调用cvtColor(frame,frame,CV\_BGR2RGB)函数，将GBR模式转为RGB模式。

3、将获取的摄像头数据利用QImage类+ QPixmap类实现在QLabel中的展示。

（1）读取frame中的数据，并将其强制转化为QImage类，并存在image中。

（2）利用setPixmap()函数（先将image从QImage类转为QPixmap类）和resize()函数将摄像头画面用Qlabel显示出来。

4、利用字节数组QByteArray类和数据缓冲区QBuffer类实现数据的存储。

（1）创建QByteArray byte作为存放在缓冲区QBuffer buff中的字节数组。

（2）调用buff.open(QIODevice::WriteOnly)函数，实现缓冲区的打开方式写。

（3）调用image.read(&buff,"JPEG")函数，将抓取的单帧图像保存在byte字节数组中。

（4）调用qCompress(byte,5)函数，将字节数组中的数据进行压缩。然后赋值给QByteArray ss字节数组。

5、利用QUdpSocket类发送UDP数据报，实现摄像画面和图像识别结果的实时传输。

（1）创建QUdpSocket类udpSocket。

（2）调用bind(QHostAddress::Any,8888)函数，绑定地址和端口。

（3）调用writeDatagram(ss,dstip,dstport)函数，发送UDP数据报。

（4）调用connect(&fps\_timer,SIGNAL(timeout()),this,SLOT(VideoSend()))函数，关联定时器和槽函数，当定时器超时时，发送摄像头画面。

（video\_receive\_show()函数）

和receive客户端的接收函数一样，用于在开启障碍物识别后返回具体的识别精度信息和识别后的画面，实现信息反馈功能。当关闭识别功能时，send客户端画面恢复识别前画面。

Receive客户端：（video\_receive\_show()函数）

1、利用QUdpSocket类接收UDP数据报，实现摄像画面的接收。

（1）创建QUdpSocket类receiver。创建QHostAddress类adrr存放ip地址，

创建quint16类port存放端口号。

（2）调用bind(QHostAddress::Any,6666)函数，绑定地址和端口。

（3）调用pendingDatagramSize()函数判断数据长度并存入quint64类size中。

（4）调用connect(&receiver,SIGNAL(readyRead()),this,SLOT(video\_receive\_show()))关联UpdSocket和槽函数。只要 UDP 数据报到达指定的地址和端口，就会触发 readyRead() 信号，此时可在槽函数中显示摄像头画面。

2、利用字节数组QByteArray类、QImageReader类和数据缓冲区QBuffer类实现数据的接收和存储。

（1）创建QByteArray byte作为存放在缓冲区QBuffer buff中的字节数组。创建QImageReader reader实现设备文件中图像以及其他图像数据的读入。

（2）调用buff.resize(size)函数，重定缓冲区的大小。

（3）调用readDatagram（buff.data(),buff.size(),&adrr,&port）函数接收数据报，并调用qUncompress（buff）函数对缓冲区内数据进行解压。解压后重新放入buff中。

（4）调用read()函数用来读取设备图像读取成功返回QImage，否则返回NULL，存入QImage类image。

3、利用QLabel进行摄像头画面的实时显示。

（1）利用setPixmap()函数（先将image从QImage类转为QPixmap类）和resize()函数将摄像头画面用Qlabel显示出来。