水下机器人上位机的研究

集美大学海洋信息工程学院

通信工程专业　2018届 李昕凯　学号：201821112051

# 摘 要

如今人类对陆地的探索趋于完善，对海洋的了解却知之甚少。

# 引言

浩

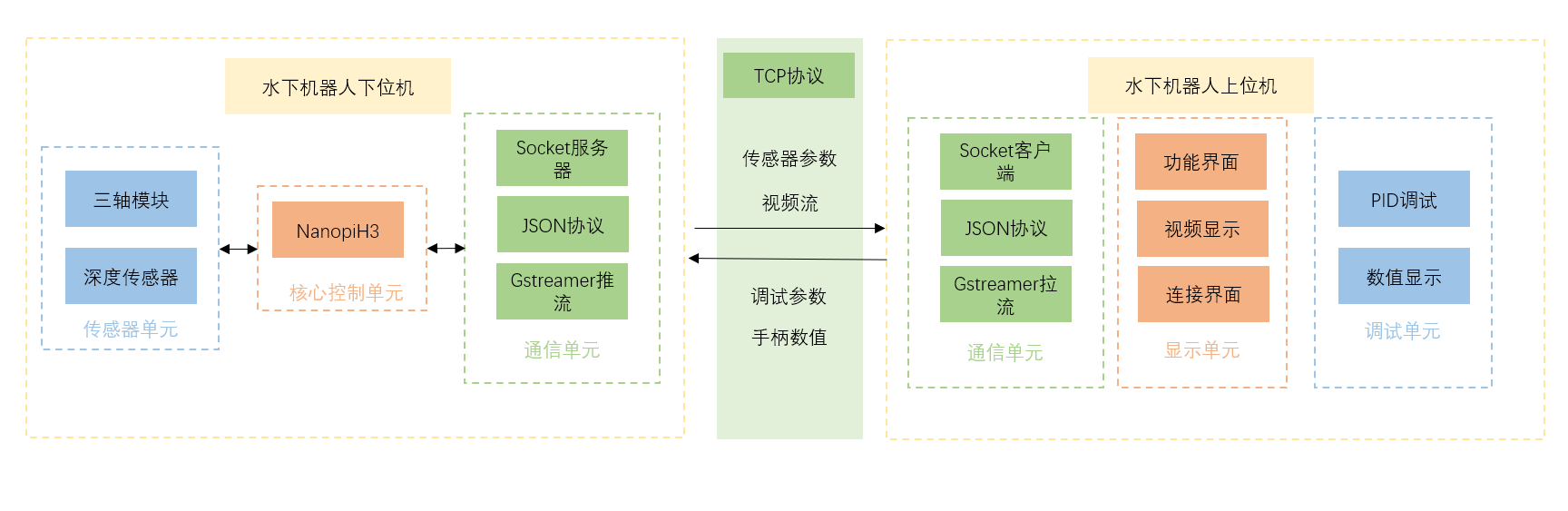
2.1 总体方案设计

1. 以QT的图形控件为显示界面框架而开发的，它为具有技术专业背景的程序提供GUI组件和一组实用类，其目标是以基于2D方式的窗体部件来显示数据， 数据源以数值，数组或一组浮点数等方式提供，提供美观，简介图形组件的同时，使用上手门槛也比较低。

2.基于JSON协议（是一种用于数据交换的轻量级数据结构）。它基于ECMAScript的子集，以完全独立于编程语言的一种文本数据格式存储和代表数据。 这种设计方法能够很好地处理复杂且庞大的数据结构以及对时间要求非常高的应用需求，并且具有良好的扩展性。此外，由于使用了简单直观的数据类型描述方式，所以更容易理解和操作。 层次结构的简单和清晰使JSON成为数据交换的理想语言。 它完全独立于任何一种语言，不论C还是Python都能高效的解析与申明该数据类型；还具有良好的扩展性和可配置性。它允许用户在不需要任何编程基础的情况下直接调用。 便于开发者阅读，易于解析和生成，有效提高开发者同时分组不同部分工作协同的效率，尤其在不同设备上的协同开发。 该协议通常建立在TCP/IP协议上，因为Json协议具有轻量级的特性，为带宽有限或需要随时更改协议定义的远程设备提供实时、可靠、可扩展的消息服务。 因此JSON协议常常被应用于各类物联网设备及嵌入式系统中。TCP协议是一种非常有效的应用层传输协议你，除开少量功能位保证通信建可靠性，每个包可含括1460个字节。 TCP数据采用端到端的校验方法，保证了接收端接收到的数据的准确性。对于每一组数据，它等待接收端对ACK信号作出响应，接收到ACK的确认信号才继续接收数据，确保下位机能够正确地接收到所有正确数据。 JSON通信协议可以保证上位机和下位机之间通信的灵活性，TCP校验机制可以保证数据包传输的正确性，并避免传输中的错误或更改。

2.GStreamer 是一款用于构建流媒体应用的[开源](https://baike.baidu.com/item/%E5%BC%80%E6%BA%90/246339)[多媒体框架](https://baike.baidu.com/item/%E5%A4%9A%E5%AA%92%E4%BD%93%E6%A1%86%E6%9E%B6/4639266)，其目标是要简化音频与视频应用程序的开发，通过 [GStreamer](https://gstreamer.freedesktop.org/) 对 [RTP](https://wikipedia.org/wiki/Real-time_Transport_Protocol) 视频流进行多种格式的实时编解码，减小视频传输对带宽的要求，以增加视频流的显示质量与完整性。

3.上位机系统方案包括通信功能、下位机数据解析功能，实时播放视频功能，调试功能等。



**2.2程序UI开发**

用户交互界面（简称UserInterface，简称UI），界面设计家有美感之称，通过软件界面美观度的设计等，可以为人们提供一个界面简洁、动态美观、大方、使用方便的窗口。 漂亮的用户UI设计往往会创造舒适的感官体验，部分原因是缩短了人与机器之间的距离，人们更有可能使用。 因此，一个优秀的上位机UI界面需要根据工科要求来设计界面，作为一款电子控制类软件，界面需要简介明了，易于使用，提示清晰，功能效果突，“傻瓜式”使用体验。

**2.2.1用控件应用**

QT出于为开发者提供上手难度简单，门槛低的图形控件。为开发者提供了大量可以直接调用，操作方便的函数接口。只需要在进行项目开发前对这些控件做一定的了解，稍作练习熟悉接口的效果，理解每个API的属性、显示效果、使用方法。根据应用场合调整参数及如何添加事件等，就能在设计过程中能够按照自己需要熟练的操作和使用它们，减少复杂BUG的数量。

QT是一个跨平台的图像库，内含有可以直接使用window的函数库，对于一个window上位机程序。经常使用的API接口有 文本框-TextBox（）函数，按钮-Button（）API函数，标签-Label（）函数，面板-Panel（）函数，报表-CrystalReport（）函数等。

以下会详细介绍四类常用的API，具体介绍API的属性，用法以及效果，并加以自己开法过程中的一些理解，一些不常用的API接口便没有介绍。

窗体 Form 控件：

下表介绍了窗体Form控件中常用的一些属性和方法：

表 2.1 Form 窗体主要属性及方法

|  |  |
| --- | --- |
| 属性或方法 | 用途 |
| Name | 窗体的名称 |
| WindowState | 表示 Windows 窗口打开后呈现的状态，默认 Normal |
| Text | 窗体标题栏中显示的文本 |
| BackColor | 背景颜色 |
| Fond | 用于选择窗体内要显示的内容的字体类型[13] |
| Location | 用于设定窗体的左上角相比于整个界面的左上角的坐标点[14] |
| Visible | 用于设定能否让用户看到该窗体，若值等于 false，则用户不能看  到该窗体 |
| Hight | 获取或设置窗体的高度 |
| Width | 获取或设置窗体的宽度 |
| Start Position | 用于设定当我们执行程序时该窗体在屏幕上的相对位置，可以选择居中，系统默认，或者当前居中等 |
| Minimize Box | 用于设定该窗体是否能够以最小的形式显示，如果可以，那么在窗体的右上角出的最小化图标可以点击；如果不能够以最小化形  式显示，那最小化的按钮为灰色显示，处于不可用状态 |
| Maximize Box | 类似于最小化窗口属性，该属性用于设定该窗体是否能够以设计时的最大窗口显示，如果可以，那么在窗体的右上角出的最大化图标可以点击；如果不能够以最大化形式显示，那最大化的按钮  为灰色显示，处于不可用状态[15] |

2.Button 控件：

Button API控件是图形API控件最常使用的按钮API，在window上位机的界面的开发中经常使用的Button控件，常常使用Button去实现一些具体点击反馈的操作。其中最长使用API控件的方法和属性定义如下表所示。

|  |  |
| --- | --- |
| 属性 | 说明 |
| Enabled | 使按钮处于能够点击使用或者不可用无法点击的状态 |
| Click(方法) | 单击时用以实现一定的指令操作 |

按钮API库中经常用是 Click 方法，该方法能够在点击按钮之后进行相应反馈，在点击反馈中加入写好的函数，从而满足功能开关的效果。

3.TextBox 控件：

TextBox 是QT图新控件最经常使用的API控件，通常用于获取输入变量和显示提示信息以及文本信息，QT提供的TextBox 控件可以满足绝大部分场景，合理搭配使用可实现大部分需求。TextBox 控件的常见API和方法参照下表的说明。

表 TextBox 类基本功能和方法

|  |  |
| --- | --- |
| 属性 | 说明 |
| Text | 检索在控件中输入的内容 |
| MaxLength | 可在文本框中输入的最大字符个数[17] |
| Multiline | 表示能否在该控件中显示若干行的文本 |
| ReadOnly | 文本框的文本为只读 |
| AppendText | 用于追加字符。用法同 Text.Add(str)类似，但两者并  不完全相同 |
| Clear（方法） | 删除文本框内现有的所有文本 |
| Show（方法） | 用于显示一个控件 |
| KeyPress（事件） | 当我们按任意一个键时会执行该事件函数内的一些  操作 |
| TextChanged（事件） | 在文本框内的内容产生变化的情况下会执行一定的  操作 |

基本属性：

Text 属性：

Text属性是文本框函数最基本的属性之一，它用于显示用户可以看到的文本框内容。 编程人员可以通过使用Text Bow控件来实现对多个文件进行操作。 但有时由于一些原因，如用户对文档的要求不同等因素,Text属性可能会被修改或取消。需要根据具体情况具体设置。 如果要增加或删除某个文本框控例中的字符，则需要先将该控件添加到Multi Lce属性上，然后再把它放入true,此时就能实现32KB以上的字符了。

Text属性通常有三种设置方式：一种是在编写代码时修改名称，也可以直接在文本框的“属性”列中输入内容，可以根据输入的变量更改属性中内容；或者用户可以直接按键输入字符串，并通过在程序初始化阶段或其他触发事件中获取Text中的字符串。

MaxLength 属性：

MaxLenth用于设置文本框限制的输入的最大字符数，合理的设置可以避免许多不必要的字体错位，格式不统一的问题。将MaxLenth设置成0，即是解除了输入长度的限制。

MultiLine 属性：

MultiLIne用于设置文本框多行输入开关，在标题或提示文本中经常进行更改设置，为了减少显示bug或者错位的问题。

基本方法：

(1)AppendText 方法：

AppendText用于文本拼接。与.add()函数用法相似，但实现的效果不完全相同，该方法会在原有文本字符串的后面直接添加文本数据，并且可以任意次数使用。

(2)Clear方法：将控件内的内容全部删除。

该方法调用的一般方式如下：

在需要删除的文本函数后，使用Clear（），该函数没有参数。

Focus方法：

为文本框函数在特定像素位置设置一个焦点，常用于文本框的定位，是文本框根据界面的大小试试调整自己的位置，可自适应不同分辨率的设备一般以文本对象.Focus()的格式进行使用;

TextChanged 事件：

程接口（如Foundry,JAVA等）访问文件，就可以通过调用该接口来实现这种功能。这就是我们所熟悉的Text Box技术。 当开发人员可在更改文本框的内容时触发此事件，更改TextBox的Text属性或添加其他函数达到效果。

Show 方法：

通过Show函数，我们调整控件的显示属性。但在使用此方法的名称作为控件名时。 Label函数可以作为实现某些操作的提示功能的最简单控件提供对Text值的访问，并用于显示用户无法编辑的图像和文本。  在许多应用系统中,Label2是很重要的工具之一。它能使应用程序更加简洁和美观，并且可以为用户带来更多的便利。因此，使用非常广泛。 Label控件中的常见属性与方法参照表 2.4 的说明。

表 2.4Label 控件主要属性和方法

|  |  |
| --- | --- |
| 属性 | 说明 |
| BackColor | 获取或设置控件的背景颜色 |
| BorderStyle | 用来设置或返回边框 |
| Enabled | 设置或返回控件的状态 |

BackColor 属性：

BackColor（）函数用于获取或设置Label的背景颜色。上位机中自定义系统的背景颜色的方法有三种。 其中，系统对用户输入的参数进行分析并根据分析结果来自动地对环境的背景颜色进行设定；而系统则是通过手动方式来实现。系统中有很多选项，可以根据功能或美感设置任何颜色，选择阴影等效果。

BorderStyle 属性：

用于设置 Label 的边框。此时有几种边框风格： 系统默认情况下为

BorderStyle.None，没有边框。 通过使用该方法可将任意大小的图像都显示出来，同时还能在屏幕上自由地移动鼠标和键盘等操作。用户只需选择需要查看的内容就能得到想要的结果。 固定的固定单边框,以设置属性值FixedSingle.

(3)Enabled 属性：

Enabled可设置控件的点击属性，当值设置为false时,Label控件为暗灰色,无法直接边境。合理设置该属性能减少大量不必要的bug，增加开发的效率。

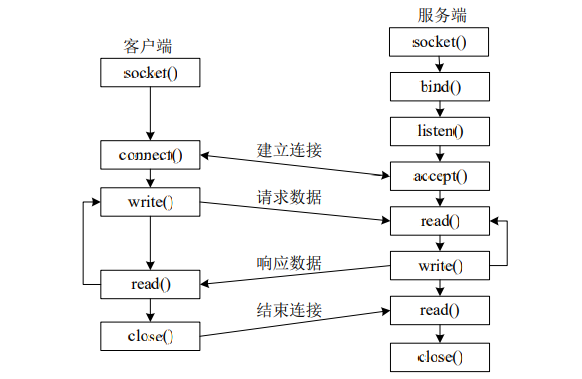
如果要修改Enabled属性，可以通过调用如下方法：

nameLab.Enabled="字符串";//设置显示内容

**2.3.1 Socket 技术**

Socket用于描述IP地址和端口，是一个通信链的句柄。socket套接字技术用于建立终端与终端通信链路，或多个终端设备间的稳定连接。它是一种基于TCP/IP协议，处于应用层与传输层之间的的通信技术。由于该数据传输技术采用了“点对点”传输模式，相比串口、IIC等直接传输数据的协议，具有很高的可靠性。目前全球的数据网络都由IPv4协议作为网络的识别号，世界上有着数以万记，因此有着许多IP地址，每个地址可以挂载数万个端口作为API接口。因此我们通常使用“IP+端喊”方式来实现套接字的传输。当一台主机发生故障时，其他主机能够迅速地从其它主机获取该主机的地址信息来恢复其正常运行状态。本文主要研究如何利用套接字机制提高网络通信效率。主机一般运行了多个服务软件，同时提供几种服务。每种服务都打开一个Socket，并绑定到一个端口上，不同的端口对应于不同的服务（应用程序），因此，在网络协议中使用端口号识别主机上不同的进程。该数据交互方式通过套接字描述符区分不同的端口与应用线程，一张网卡就可以分出多个虚拟仲端，实现多TCP连接的数据并发服务，节省了计算机资源并降低了硬件成本。

一个 Socket 嵌套字连接服务，由服务端和客户端两个端口组成，基于QT的网络控件，实现 Socket 配置的示意图



服务器被动接受连接请求，不会主动发送，而是作为监听器，监听特定IP+端口，等待客户端发送的socket连接请求并确认。 利用socket控件创建服务端的过程为：

1）socket()创建套接字；

2）bind()绑定端口；

3）listen()侦听队列；

4）accept()接受连接；

5）close()关闭连接。

客户端是socket网络连接请求的发起者，通过目标主机IP地址与非指定的空闲端口发送连接请求。利用socket控件创建客户端连接请求的过程为：

1）socket()创建套接字；

2）connect() 发起连接；

3）close()关闭连接。

在Socket连接中， 数据请求是发送数据包来进行交互， 数据按信道配置顺序尽心传输，就像水流过管道一样。它的优点是不会重复和没有错误。基于TCP协议的流量控制，为了防止过多的数据造成缓存拥塞，Socket限制数据包的大小，并使用ACK应答的校验方式来保证数据传输。此外，Socket是面向连接的数据交互方式，依照TCP协议的三次握手，优先确保连接的可靠性，在服务端接受客户端请求之后将持续保持与客户端连接。嵌入式系统的设计与开发。 Socket是指通过建立通讯连接来实现数据传输的方法，即两台以上的主机之间可以相互通信并完成数据交互，它不需要额外的通讯规约和接口。 基于上述特点,Socket可以应用于多种工业环境下的实时测控领域。本文介绍的水下机器人监控系统就是一个典型的远程控制和监测平台。它由单片机控制器作为主控部分。 由于嵌入式系统中没有串口,IIC等标准的数据传输协议，在传输过程中易出现误码或数字丢失的现象。 因此，该系统将采用Socket和JSON数据格式实现水下机器人下位机之间的数据交互。

**2.4 JSON 数据格式**

JSON协议是一种结构简单、利用率高、轻量级的文本格式的数据结构，是一种独立所有变成语言之外的数据结构，其具有诸多优点：

1）有效信息描述十分简洁，体量轻量级的数据交换格式；

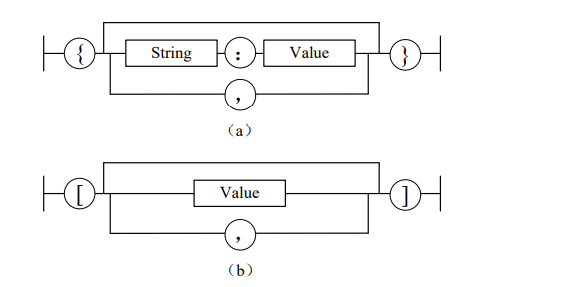
2）信息直观，十分易于开发者进行阅读、编写和维护；

3）易于计算机生成和解析；

4）独立于编程语言，不论是C， Java、 Python还是go语言都能十分便捷的使用JSON协议等。

5）JSON语法结构

JSON协议的数据结构分为对象类和数组类两种结构类型，JSON 对象类数据结构表示方法Key/Value对来表示并保存。而JSON协议的数组类则是将Key/Value对作为一个Value对象。数组类结构是一种嵌套结构以key值为数组名，value为新的对象结构，JSON 对象类和数组类的数据结构的语法结构分别如图



“{”表示开始读取对象；

“}”表示结束读取对象；

“[”表示开始读取数组；

“]”表示结束读取数组；

“：”表示分隔“名称-值”中的名称和值

“，”表示分隔对象中的“Key-Value”元素或数组中的“Value”元素。 例如，JSON 对象：{ “城市”:“厦门” , “age”: 22 , “name”: lixinkai}； JSON 数组：{ “厦门” , 22 , “李昕凯”}。

（2）JSON 序列化

在使用JSON的过程中，没申明一个key/value,实际上申请了以链表结构。封装JSON数据的过程，其实就是创建链表和向链表中添加节点的过程，其中各个Key:value对的地址并不连续。JSON序列化则是将不连续的链表合并与转换为字符串的过程，客户端发送和存储的数据必须进行JSON链表的序列化。将信息按照 JSON 语法编写的顺序与逻辑合并为固定格式的字符串。在QT之中，JSON 可以直接调用CJSON库函数直接序列化，这种用法的优点是能够及时直观的验证语法鱼内容的正误。JSON序列化功能函数中常用的有 JavaScriptSerializer，下位中则使用cJSON\_Print（）函数。

（3）JSON 反序列化

JSON 反序列化则是将具有特定格式的字符串转换为JSON链表结构的过程。客户端JSON.parse()函数将接收的 JSON 格式字符串转化为key-value对的链表嵌套结构，对传输数据进行解析，然后使用cJSON\_GetObjectItem（）根据需要key值提取对应value值。JSON 虽然只是一种文本格式，但与串口协议、IIC协议、SPI协议有本质区别，通过TCP协议与socket嵌套字程序的支持，可以实现数据交互过程中的稳定传输。使用JSON 协议进行通信交互，在不影响传输速率的基础上简化了自定义协议中功能位定义流程与数据流程。开发人员不需要理解JSON通讯协议或自定义协议具体每个字节的作用，只需要按照规定提取 JSON 对象元素中key-value对就能解析或修改传输的数据内容。基于JSON数据格式的特点以及优势，本系统将在控制、调试等多个模块的数据交互中应用。

**2.5 视频编码标准**

**2.5.1 主流视频编码标准介绍**

视频原理利用人眼反应的余辉效应，造成连续的错觉。所以视频是有一张张的连贯的图片组成的。而这一张张连贯的图片，往往每一张图片中有许多相同的的数据。这对数据的存储与解析是不要的冗余数据。例如一组连贯的图片的数据可能基本完全一致，使图片的数据存储产生空间冗余，视频播放产生了不要的时间冗余和编解码资源浪费。为了避免不必要的资源浪费，提高效率，就需要消除这组图像的空间冗余。主流的做法则是对视频的图像帧进行图像预测，消除多余的信息，减少冗余。还可以通过相邻像素预测来消除图像中亮度和色度相近的图像像素信息。最后本文会利用已有的技术探索实时视频的效果。在ITU-T和ISO等优秀编码技术的推广下，已经对视频编解码技术进行数个版本的迭代。各技术被大量应用于实时视频，监控录制，图像数据存储与传输等应用中。其中最为主流的视频编解码标准 有MPEG-4、H.265以及H.264。

（1）经过多个版本的迭代H.264视频编解码技术已经成熟，历代的升级极大提升了H.264的编码效率。提出了先进的4\*4DCT变换算法，极大降低了算法复杂度。优秀的帧内预测算法与多模型帧间补偿与预测相结合，提高了编解码的压缩比，在理想情况下最高能减少70%的压缩数据。改进的码流结构模型提高了信道的利用率。相比上一代的H.263，相同环境与条件下能够减少50%左右的带宽，大大提高了视频的实时性与图像质量的同时，减小的对带宽的要求，从而减少硬件成本。应用的领域也更加广泛。

**2.5.2 H.264 的技术特点和选择说明**

因为先进的编码方案和优秀的编码算法H.264不仅具备优秀的编码效率，也兼具出色的网络传输能力。因此H.264是开发者最常使用的视频编码方式。H.264领先于其他编码算法在于先进的算法思想和优越的系统设计方案。H.264的系统设计中，按功能分为了两块。一块专门负责视频编码，对视频中的冗余信息进行预测并剔除。其中适用多种先进算法，包括运动多模型补偿算法、图像预测算法，定义了新的编码特性。在此基础上,提出了一个通用的视频编码标准框架,并给出了其详细实现步骤。因为视频传输所需要的带宽比较大，通常会占满整个带宽，所以不会设置校验位，对数据的可靠性的要求往往比较低。因此视频编码通常使用UDP这种无需建立的网络协议。采用新的理论进行设计的编解码在编码特性上有着卓越的编码效果，H.264通过帧间预测将临近属性相似的像素点进行算法优化，配合运动补偿算法消除视频流编码的时间冗余。结合变换编码与运动补偿算法特点的混合编码方案消除空间冗余。H.264的主要编码特性包括：

（1）帧内预测编码：H.264通过使用当前图像块左边和上侧的图像对当前图像块作出估计并对现实值和预期值的差异编号,能够有效地减小空间冗余率。H.264根据宏块尺寸提出了灵活多样的帧内预测编码模型,具有很强的应用适应性。

（2）帧间预测编码：通常在视频流信息中，每一帧图像包含的数据与数据与前后图像间存在一定的联系。这种物体的运动关系可以用较少的函数关系进行表达，也就成了时间上的冗余。而这可以预测的运动相关性造成的时间冗余远远大于空间冗余，也是影响视频实时性的关键。H.264在在传统的PBI帧预测方案基础上，加入了 多参考帧间预测与融合算法，使压缩效率提高了50% 。

（3）量化与整数DCT变异在完成转换编码时,H.264采用了整数DCT转换来逼近离散余弦变换特征,以整数为基本变异算法不但大大降低了统计量和复杂性,而且反转换毫无误差,物质运动的分析也更为准确,控制了块效果和振铃效应。

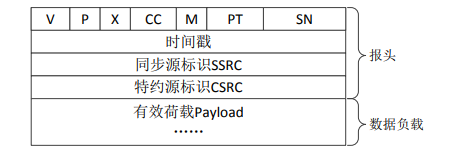
（4）在编码过程红，编码算法会按照既定分割方式将图像分成宏块，再对宏块中的子块进行预测、变换和量化编码，而以宏块的为单位进行的预测编码结构会导致图像中产生块效应。因此需要对产生块效应的数据进行去块滤波。

（5）熵编码编码在统计意义上对数据冗余进行编码。 它能有效地降低传输过程中因丢失信息而导致图像质量下降的问题。为了进一步提高压缩比和减少计算量，熵编码被引入到视频编码器当中。目前常用的熵编码方法有多种。 H.264采用两种可选的熵编码模式，一种是基于上下文的自适应可变长编码的CAVLC模型，另一种是基于内容的自适应二进制算数编码的CABAC模型。 前者根据编码的语法元素，动态调整代码中的码表，实现自适应能力。 然而，由于采用静态概率来估计码表，CAVLC编码效率受到限制。 CABAC解决了这个问题,可以实时更新概率,更好地适应高码率. 相对于其他视频编码方案，还是网络状况的适用性上， 都十分出色。尽管HEVC作为新一代的编码标准，压缩性能优于H.264，但是HEVC更复杂的 算法会降低编解码的计算速度，在强调实时性的视频监控系统中，H.264作为压缩性能和算法复杂度的折衷考虑，无疑是更好的选择。并且目前而言，H.264的软硬件支持都优于HEVC。 因此本文选择了H.264编码作为本文系统的视频编码方案。

**2.6 流媒体传输协议**

**2.6.1 RTP 传输协议**

实时传输协议（RTP）是由Internet专家任务组IETF针对多媒体实时数据传输而提出的新型应用层协议。该协议基于单播或多播网络为用户提供端到端的流媒体数据传输服务[32]。RTP仅负责数据的实时阐述，通过提供时间戳信息来保持收发端的数据同步，而传输中的差错控制和网络QoS则依赖同族协议RTCP来完成。 RTP协议一般运行在UDP之上，一个RTP协议帧的结构主要由12个字节组成的固定报头和有效载荷两部分构成，如图2.3所示。其中报头又包括了以下10个部分：



（

**2.6.2 RTCP 传输协议**

作为RTP的传输控制协议部分，RTCP主要负责数据传输中的流量控制和拥塞控制。 由于RTCP具有良好的扩展性，可以很方便地扩展到各种应用场景之中,RTCP已经成为网络传输中一种重要的协议形式。它是一个简单而又复杂的过程。 RTCP在数据包两端定期发送包含已发送和丢失的统计信息的包，来反馈数据在网络中的传输情况。RTCP协议同样使用UDP来发送数据，包含四种分组类型：

（1）发送端功能位SR SR用于描述发送端的发送和接收的计算数据。SR报文主要包括NTP时间戳、RTP时间戳、SSRC标识符、发送字节数以及发送包数等信息。

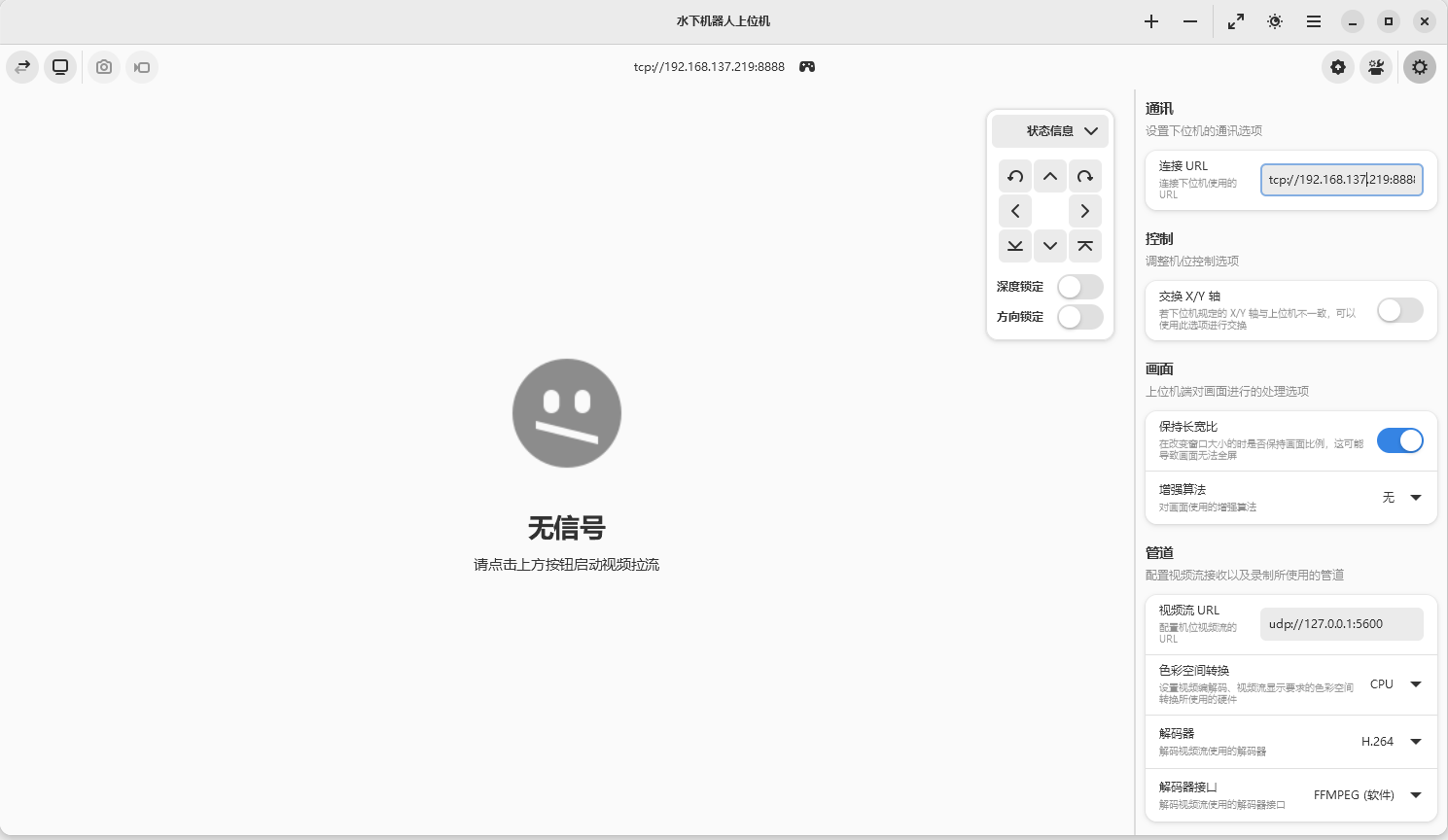
（2）接收端功能位RR RR用于描述接收端的计算数据。RR报文由接收端来发送，提供了同步源标、识符、累计丢包数、接收最后一个SR的时间、到达时间抖动、接收最后一个SR的延迟等信息。

（3）文本信息来描述会话用于源描述报文SDES、SDES，包括终端标识、号码、用户、名称、电话地理、位置电子邮箱、等源描述项，为获取发送端的报文及数据包提供了方便。

（4）通知离开报文BYE，用于通知成员的离开。当发送端要结束发送时，会发离开报文BYE标志位示意接收端结束接收。

**第三章 上位机软件的实现**

**3.1上位软件的功能与实现方法**

该上位机是基于QT作为主机主窗口类的设计. 该系统主要分为三个界面：主界面、菜单显示界面以及数据存储与调试界面。 主窗口包含主菜单栏、工具栏、状态栏，操作简单，可扩展性好。 将QWidget类添加到主窗口中，主窗口是所有用户界面对象QWidget基类。您可以添加各种界面组件，如QLabel、QpushButton、QlineEdit等。 同时，使用QTabWidget组件进行分页处理、不同功能的接口切换，上位机软件的主界面如图所示。

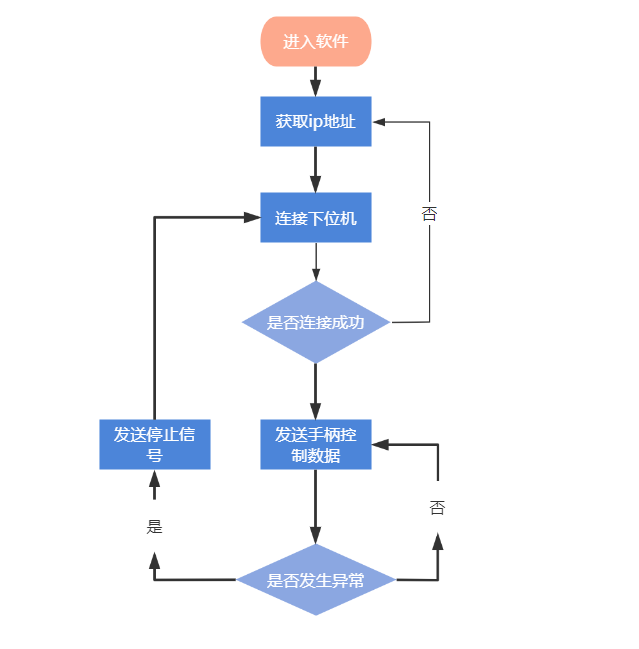
该上位机软件主要由主界面与调试界面组成：主从机的登录通信模块以及机器人当前话题列表和运动信息显示界面、可视化界面以及机器人启动设置和控制模块显示界面。可根据需要修改配置，具有较强的灵活性与通用性。上位机软件所实现的主要功能与实现方法如下：

### 3.1.1通信功能

建立上位机与下位机之间的通信：将启动上位机的电脑与下位机的嵌入式系统设置为静态IP地址（上位机与下位机静态IP地址需要在同一网段下，测试之中下位机设置为192.1468.137.219，上位机设置为192.168.137.50），在上位机通信模块中的连接URL输入下位机的静态IP地址及端口号，再通过TCP协议让上位机机登录下位机控制并获取下位机的状态数据及视频推流，上位机持续发送手柄控制数据或调试界面的调试数据，从而完成登录通信过程。

**实现方法：**调用socket库函数搭建客户端，使用QT的输入文本框API在主界面中设置连接目标的输入框。获取目标的IP地址及端口号传入connect（）函数请求连接下位机。

异常处理：除开TCP自带的异常处理，上位机与下位机发生连接异常时，上位机会先做出反应，向下位机发送异常警告与停止信号。下位机收到异常警告与停止信号时或者连接断开信号时，会停止水下器机器人的一切运动（包括推进器、机械臂、云台），以防止不受控制而暴走。



### 3.1.2 Xbox手柄控制

上位机使用市面使用率最高的Xbox手柄作为控制器，使用微软提供的Xbox手柄通信协议库获取手柄参数，按照需要映射成下位机控制数据发送至下位机，也可由上位机控制界面直接控制。手柄可控制下位机进已下动作。

1. 运动\t停止
2. 前后\左右\上浮下潜\左旋右旋
3. 机械臂抓紧\放开
4. 探照灯亮\灭
5. 摄像头变焦、对焦
6. 摄像头云台的上升\下降



**实现方法：**

1. 获取Xbox官方库

git clone git@github.com:WanderROS/Xbox360GamePad\_Qt.git

1. 对协议上图拟定的按键功能对xbox按键进行功能映射

//连接设备

QGamepad \*m\_gamepad = **new** QGamepad(0, **this**);

...

//显示手柄按下的按键的名称

connect(m\_gamepad, &QGamepad::buttonAChanged, **this**, [=](**bool** pressed){

    showCurrentBtnName(pressed, "buttonA");

});

connect(m\_gamepad, &QGamepad::buttonBChanged, **this**, [=](**bool** pressed){

    showCurrentBtnName(pressed, "buttonB");

});

...    //中间省略映射功能

//显示摇杆的值

connect(m\_gamepad, &QGamepad::buttonL2Changed, **this**, [=](**double** value){

    showBtnVaule(1, value);

});

connect(m\_gamepad, &QGamepad::buttonR2Changed, **this**, [=](**double** value){

    showBtnVaule(2, value);

});

1. 发送下位机

### 视频显示

水下机器人前段装有一颗H.264编码1080p的摄像头通过插件Gstreamer进行视频推流，上位机接收并显示在主界面中

**实现方法：**

1. 安装好Gstreamer的相关插件

yum install gstreamer-devel gstreamer-plugins-base-devel gstreamer-plugins-base-tools gstreamer-plugins-espeak gstreamer-plugins-base

gstreamer-python gstreamer-rtsp gstreamer-plugins-bad-free-devel gstreamer-tools gstreamer-plugins-bad-free gstreamer-plugins-good gstreamer-plugins-ugly

1. 下位机视频推流

进入下位机系统/etc/rc.local自启文件加入下列指令。其中根据插入的摄像头的设备号自行选择设备号输出。以H.264编码协议30帧帧率1080p格式发送至ip为192.168.137.50端口号为8888的设备上。

gst-launch-1.0 -v v4l2src device=/dev/video1 ! 'video/x-h264,width=1920,height=1080,framerate=30/1' ! rtph264pay ! udpsink clients="192.168.137.50:8888"

…

1. 在上位机中嵌入Gstreamer的拉流指令，其中以H.264方式解码，1080p播放视频流。

gst-launch-1.0 rtspsrc location=rtsp://username:passwd@ipaddr:port  latency=0 ! rtph264depay  !  capsfilter caps="video/x-h264"  ! h264parse  ! vaapidecode  !  vaapipostproc  width=1920 height=1080  !  vaapisink sync=false

### JSON通信协议

上位机与下位机之间使用 [JSON](https://wikipedia.org/wiki/JSON) 数据包通过 [TCP](https://wikipedia.org/wiki/Transmission_Control_Protocol) 协议进行通信，一个数据包可以包含任意数量的有效命令，基本框架如下：

{

    "command1" : null,

    "command2" : "arg",

    "command3" : ["arg1", "arg2"],

    "command4" : { "key1": "value1", "key2": "value2" }

    // ...

}

当前有效的发送命令如下：

|  |  |
| --- | --- |
| 命令 | 描述 |
| x | 控制机器人前进或后退 |
| y | 控制机器人左右平移 |
| z | 控制机器人的上浮或下沉 |
| set\_propeller\_parameters | 设置推进器参数 |
| set\_control\_loop\_parameters | 设置控制环参数 |
| save\_parameters | 保存参数 |
| load\_parameters | 请求读取参数 |
| firmware\_update | 进行固件更新 |
| feedbacks | 反馈数据 |
| set\_propeller\_parameters | 设置推进器参数 |

自定义协议包根据功能分为控制指令数据包与调试指令数据包

1. 控制指令数据包

{

    "x"                : 0.0,   // 左右平移

    "y"                : -0.5,  // 前进后退

    "z"                : 0.8,   // 上浮下沉

    "rot"              : 0.1,   // 左右旋转

    "depth\_locked"     : **false**, // 深度锁定

    "direction\_locked" : **true**   // 方向锁定

}

1. 调试指令数据包

{

    "set\_propeller\_parameters": {    // 推进器参数

        "back\_right": {              // 推进器名称

            "deadzone\_upper" : 8,    // 死区上限

            "deadzone\_lower" : -12,  // 死区下限

            "power"          : 0.75, // 动力百分比

            "enabled"        : **true**  // 启用/禁用推进器

        }

        // ...

    },

    "set\_control\_loop\_parameters": { // 控制环参数

        "depth\_lock": {              // 控制环名称

            "p" : 1.0,               // 比例调节

            "i" : 2.0,               // 积分调节

            "d" : 0.5                // 微分调节

        }

        // ...

    }

}

实现方法：

按照上位机与下位机定好的协议，将key值与value对应打包。

# 第四章上位软件的功能的整体测试与分析

本文所使用的实验设备为第二章中所提到的自己设计的控制板。其核心控制芯片为全志科技的Nanopi H3。该控制板上搭载着MS5837深度传感器、DCM250B三轴传感器、PCA9685 PWM扩展芯片。具体实验步骤如下：

## 4.1 测试环境配置

1. 下位机环境
2. 使用以下指令打开/etc/network/interface.conf文件

sudo nano /etc/network/interfaces

1. 向文件中添加以下指令，将IP地址修改为静态IP：192:168:137:219

allow-hotplug eth0

iface eth0 inet **static**

address 192.168.137.219

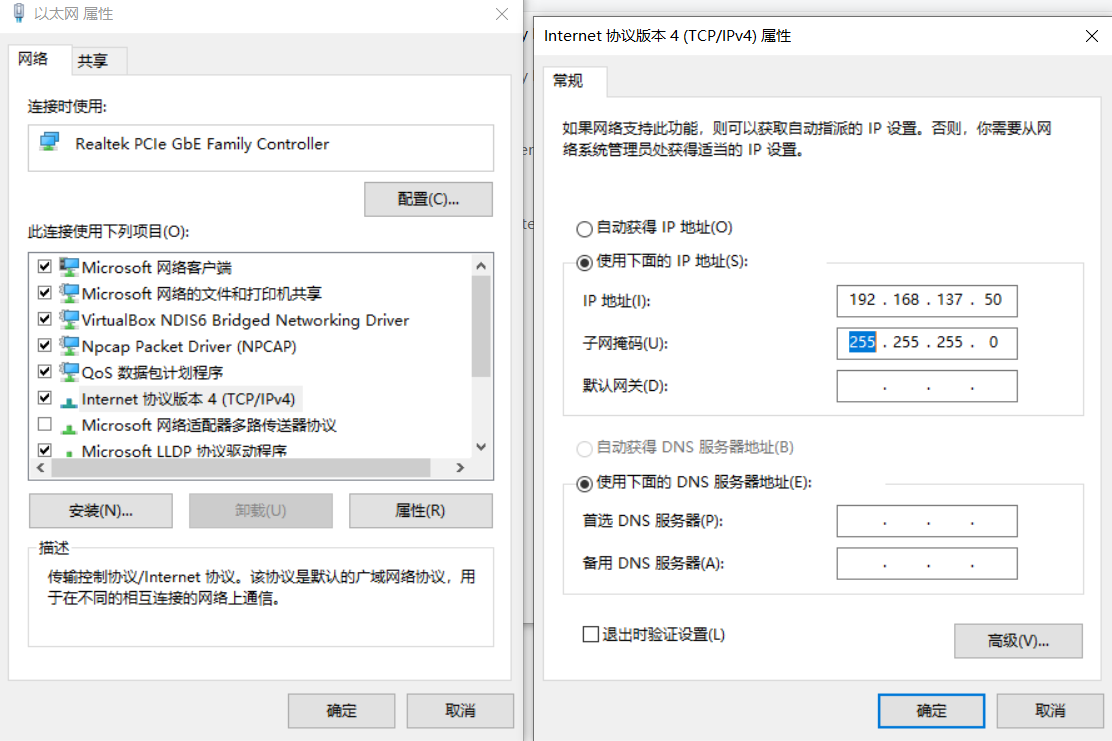
netmask 255.255.255.0

gateway 192.168.137.1

dns-nameservers 192.168.137.1

1. 上位机环境

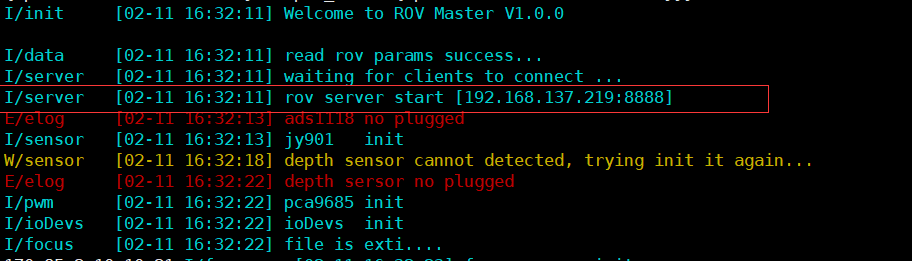
打开电脑网络适配器，将IPV4网络IP地址配置为192:168:137:50



## 4.2通信功能测试

### 4.2.1测试流程

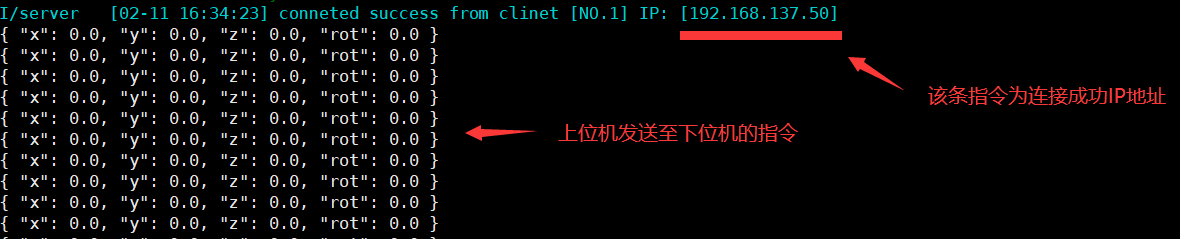
1.启动下位机程序，下位机显示当前IP为192.168.137.219:8888



2.有上节环境变量配置，在上位机通信模块中输入tcp://192.168.137.219:8888如下图。



1. 下位机显示收到IP：192.168.137.50连接请求，并连接成功。并且成功收到上位机发送的停止运动指令。



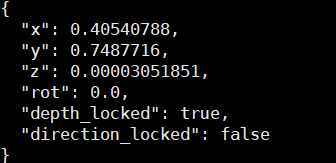
### 4.2.2 结论

上位机与下位机通信功能正常且上位机可以连接任意IP地址的主机。

## 4.3 手柄控制测试

4.3.1 测试流程

1.上位机成功识别到Xbox手柄的连接，并且下位机接收到手柄的映射数据包。



4.3.2 结论

手柄控制功能正常，且数据未产生冲突

## 4.4视频显示

4.4.1测试流程

1.下位机启动Gstreamer推流指令。

gst-launch-1.0 -v v4l2src device=/dev/video1 ! 'video/x-h264,width=1920,height=1080,framerate=30/1' ! rtph264pay ! udpsink clients="192.168.137.1:8888"

2.上位机点击拉流按键，上位机主界面能够显示水下机器人实时拍摄的画面。画面清晰流畅，没有模糊或失真。

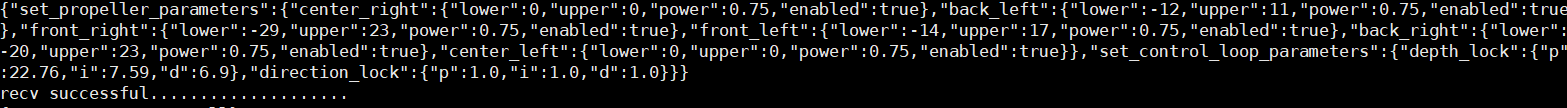
4.4.2结论

视频显示功能正常

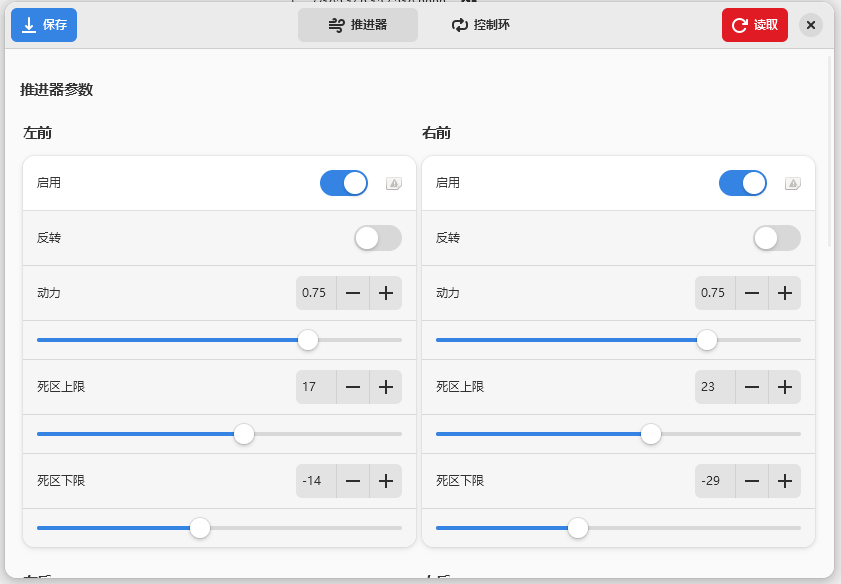
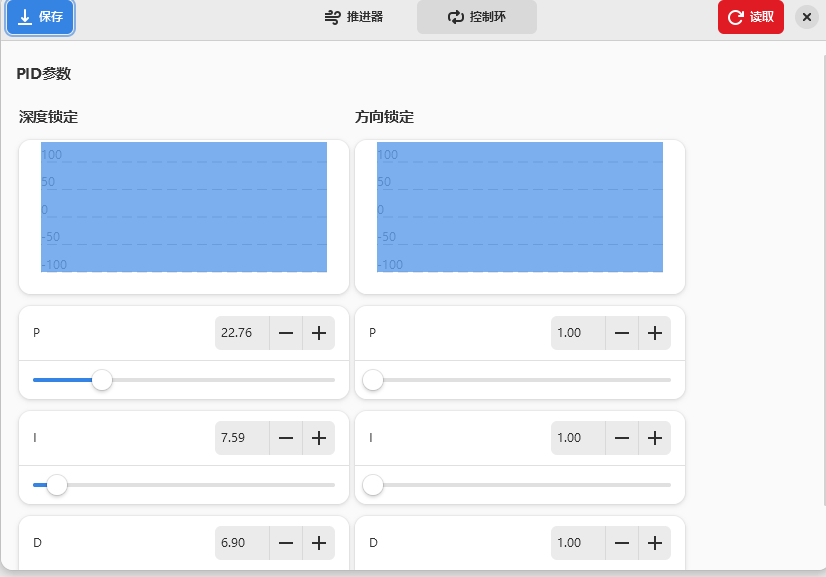
## 4.5 参数调试功能

### 4.5.1测试流程

1.点击调试界面的保存键时，下位机成功据保存至JSON文件中。



1. 点击调试界面的读取键时，上位机成功将数据显示至上位机的调试界面中

### 4.5.2结论

调试功能正常

### 结论

本文就水下机器人控制与调试不便捷这一问题，以跨平台Qt图形库框架为基础，用面向对象的C＋＋编程语言设计并实现了一个界面简洁、功能齐全、简单实用的以可靠的TCP协议作为通信协议的上位机软件。本文首先介绍了上位机软件的设计架构、TCP通信原理、JSON通信机制和Gstreamer的推流原理，然后基于原理详细的阐述了设计各功能模块的设计与实现的过程，并以最终的测试结果为实验用例来论证该上位机实现效果。并且实现下位机的状态的可视化，实验清晰的反映上位机的功能效果，验证了上位机所描述的所有功能的有效性。同时该上位机预留了控制、调试、图像增强算法等接口，增加其了扩展性与灵活性，使后续对机器人的开发更加简单。与此同时，可用用于其他嵌入式设备的控制与调试。

# 致谢语

但是在我的导师杜勇教授的帮助下，让我最终顺利完成了该课题的设计。在这期间，遇到很多难题，依靠自身难以解决，杜勇教授都在百忙之中抽出时间为我讲解，帮助我解决难题。在此，我非常感谢我的导师杜勇教授。胡教授毕生心血都花在科研事业上，为科学技术的发展奉献了很大的力量，虽然平常事情很多，但在我们遇到难题时还是抽出时间为我仔细，耐心的讲解，以严谨，认真的科学态度教导我。不仅是我的专业技术能力得以提高，还让我将技术与实际相结合，也让我对科学技术的态度更神往，更严谨。在此，我再一次感谢杜勇教授，感谢他的热心付出和辛苦栽培。

同时我还要感谢吴一亮老师，在我遇到难题的时候吴老师也给我很多帮助，并且教会了我很多的基础知识，在思路和方法不对是吴老师会给我适当的提醒，让我不至于弄错思路与方法，使我锻炼了自学能力和遇到问题的独立思考能力。非常感谢吴老师的悉心指导。我还要感谢在设计过程中同学们的帮助，让我节省了很大一部分时间，非常感谢写老师和同学们！

最后，我要感谢一直在被后默默付出不求回报的父母，在大学四年的路上给了我很大的鼓励与支持。