张玉林,马千里,陈立强,等. 承德台数字化观测仪器数据实时监控系统研发[J]. 华北地震科学,2018,36(2):76-80.

承德台数字化观测仪器数据实时监控系统研发

张玉林[1],马千里[2],陈立强[1],郭亚亚[1],宋晓冰[3],孙贵成[1]

(1. 河北省地震局承德中心台,河北 承德 067000;
2. 北京世扬软件开发有限公司,北京 100000;3. 河北省地震局张家口中心台,河北 张家口 075000)

摘要:通过 NIM 语言实现数据的实时采集、传输,并通过对显卡的编程操作实现绘图,同时用 C 语言对 NIM 语言进行重新编译,实现了承德中心台 7 套"十五"仪器原始数据的实时采集、传输及绘图。将 12 h 内的第 1 个数据作为坐标原点,进行实时坐标转换,绘制 12 h 内的原始数据实时曲线,进而实现对"十五"仪器原始数据的实时监控,方便值班人员随时查看。

关键词:数据实时监控系统;承德中心台;"十五"仪器

中图分类号:P315-391　　**文献标志码**:A　　**文章编号**:1003－1375(2018)02－0076－05

doi:10.3969/j.issn.1003－1375.2018.02.013

0　引言

承德中心台的地形变观测、地磁观测开始于 20 世纪 70 年代,起初的模拟数据观测需要每天定时检查仪器,防止仪器超限或故障,这对仪器的正常运行产生较大的人为干扰。随着"十五"项目的实施,自 2002 年 1 月 1 日至今,承德台中心台形变观测、地磁观测等测项中的所有仪器陆续更新为数字化观测仪,如承德台、丰宁台 VP 型宽频带倾斜仪、FHD-2B,丰宁地震台 SQ-70D 数字水平摆,宽城地震台新水管仪、新伸缩仪等。因观测仪器数目较多、测项各异,又需要经常检查仪器运行状态,及查看原始数据记录,承德中心台改进工作模式,积极探讨新方式新方法,研究适合承德中心台地形变、地磁等所有数字化观测仪原始数据进行一体化的实时采集、显示系统,以避免观测数据超限或受到其他因素干扰的影响,能更直观地显示仪器工作状态,更简洁地查看原始数据观测曲线,减少工作人员分别查看仪器运行情况,有效提高工作效率。

1　系统架构

MVC 全名是 Model View Controller,是模型(model)－视图(view)－控制器(controller)的缩写,是一种软件设计典范,用业务逻辑、数据、界面显示分离的方法组织代码,将业务逻辑聚集到一个部件里面,在改进和个性化定制界面及用户交互的同时,不需要重新编写业务逻辑。本系统采用 MVC 架构及并行设计,分为主调度模块、界面模块、绘图模块、数据获取模块、数据模型(图 1)。系统中除绘图模块设计为界面模块的子模块外,其他模块均在不同的线程运行,它们之间依靠消息传递来实现通信与交换数据(图 2)。

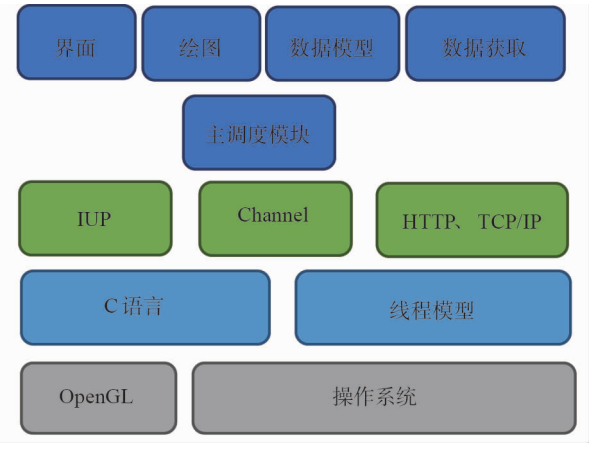


图 1　系统总体构成

1.1　视图部分(V 部分)

界面模块和绘图模块属于绘图部分,主要用于系统 UI 界面及数据的绘图显示。

收稿日期:2017-09-29
基金项目:2014 年河北省地震局星火计划项目(DZ20140714051)
作者简介:张玉林(1981—)男,河北行唐人,工程师,主要从事地震监测工作. E-mail:ylzhang2006@163.com

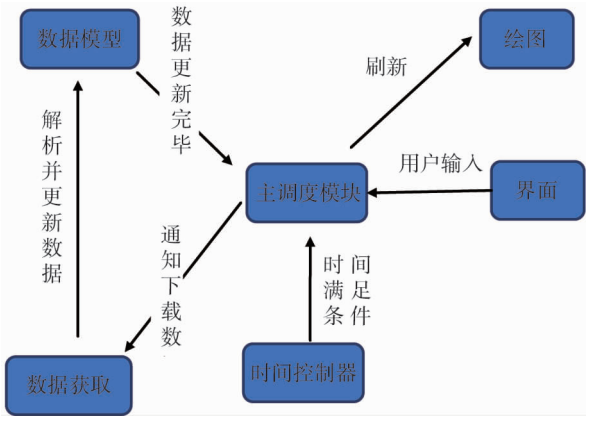


图 2　技术流程图

本系统采用 OpenGL4.3 对数据进行绘制。系统保存 12 h 内的仪器原始数据,在显卡内开辟空间,当绘制某测项数据时,将该部分数据动态复制到显存中。数据到达显存,通过对显卡编程来实现数据绘制工作。

1.2　控制器部分(C 部分)

该部分为主调度模块,负责系统运行中逻辑跳转与数据流的传送。

1.3　模型部分(M 部分)

数据获取模块,这部分是系统实现的主要支撑,系统中绝大部分工作都是由模型部分完成的。

该模块负责完成与 7 套仪器的通信和数据获取,并通过使用数据模型将数据解析、发送至界面模块。因为是与网络交互,该部分采用异步处理。本系统采用独立线程完成计时、下载、解析等工作。

数据模型模块,对各数据进行建模,实现对这类数据的存储、解析、绘制等内容。

系统每分钟刷新一次,每套仪器实时数据显示于界面顶端,通过仪器选择实现对相应仪器的实时数据绘图,把 12 h 内第一个数据作为坐标原点,并将 12 h 内原始数据的最大值与最小值之差作为纵坐标,通过实时下载的最新数据"排挤"首位数以实现对坐标系的实时转换,从而达到对原始数据的实时监控。

2　软件开发

2.1　软件开发环境

软件开发环境如表 1。

2.2　关键技术

2.2.1　Nim 语言

Nim 是一个新型的静态类型、命令式编程语言,支持过程式、函数式、面向对象和泛型编程风格而保持简单和高效。Nim 从 Lisp 继承来的一个特殊特性——抽象语法树(AST)作为语言规范的一部分,可以用作创建领域特定语言的强大宏系统。

表 1　软件开发环境

硬件		软件	
CPU	i7 2820QM	操作系统	Windows 8.1
GPU	英伟达 540M	编程语言	Nim
内存	8G	编译环境	MinGW
		集成开发环境	Sublime Text 3

Nim 是一个编译型的具有垃圾收集的系统编程语言,有着极其卓越的生产/性能比。Nim 的设计集中在 3 个 E 上:即效率(efficiency)、表达能力(expressiveness)和优雅(elegance)。Nim 主要特点是程序代码可以翻译为 C 语言、Object－C、C＋＋或者 Javascript,然后再使用相应语言编译工具进行编译。

2.2.2　Channel

Channel 是 Nim 语言实现以消息传递代替共享内存来完成线程同步的多任务并行机制。类似的技术还有 go 语言的 routine,Erlang 语言的消息传递技术等等。

2.2.3　Nim 语言和 Channel 在系统中的应用

界面模块中,Nim 语言封装了 C 语言的界面库——IUP 库。Nim 语言先编译为 C 语言代码,再由 C 语言编译工具生成可执行文件,且 Nim 语言封装 C 语言库非常方便,可以达到"无缝衔接"。

绘图模块中,Nim 语言本身提供了官方的对 OpenGL 封装库,并且 IUP 本身也支持 OpengGL,因此,Nim 使用 OpenGL 绘图十分方便。

数据获取模块使用 Nim 语言提供的 HttpClient API,可以像浏览器一样访问服务器,更有利于数据下载。

主调度模块是最复杂的模块。首先,界面和绘图模块运行在一个线程中,并且使用 Channel 与其它线程交换数据;其次,数据获取模块大部分时间都不运行,主调度模块会在需要时启动它,每台仪器一个线程,因此,如果其中某台仪器的数据获取出了问题,也不会影响其他仪器的数据获取以及下次数据的获取,数据获取之后使用 Channel 将数据交给数据模块保存,并通知主调度模块本次数据获取结束;第三,主调度模块本身占用一个线程,并且自身处在

一个循环之中。它不断地处理各个模块通过 Channel 传来的信息,将软件各个模块关联在一起。

2.3 主要代码

2.3.1 数据下载

```
## 使用 http 协议下载
Proc httpDownload(url,target:string; parseFunc:proc(body:string):Realtime | Ssy | Sgy | Fhd) =
    # 不同的数据的下载与解析放入相同线程,减少数据复制;
    # 不同数据种类放入不同的线程处理,分别由各个数据类处理。
  var response:httpclient.Response
  response.body= nil
  try:
    response=httpclient.request(url,sslContext=nil)  # 发送请求
  except:
 if response.body== nil:
    return
 var data = parseFunc(response.body) # 解析响应数据
## 使用 Socket 下载
proc socketDownload(ip:string; port:int; requestStr,target:string; parseFunc:proc(body:string):Realtime|Ssy|Sgy|Fhd) =
  var client = newSocket()
  try:
    client.connect(ip, Port(port)) # 建立 Socket 连接
    client.send(requestStr) # 发送请求
    var context= ""
    while len(context) == 0 :
      var datlength= 0
      var line= ""
      var doloop= true
      while doloop:    # 轮询等待,接收数据;收到数据后停止轮询
    client.readLine(line,500)
if line != nil and len(line)>0:
var data= parseFunc(context) # 解析收到的数据
## 数据下载
proc downloadData() =
  var doLoop= true
  while doLoop: # 轮询等待,达到间隔时间之后开始新一轮数据下载
  # 接收发给 downloadData 的消息
  let msg = timerChannel.recv()
   if msg == 0:
    doLoop = false
  else:
    if int(msg and 1) == 1:
      spawn httpDownload("http://仪器 IP/cgi-bin/realtime.cgi", "CDczb", realtime.parse) # 分别启用不同的线程来下载数据
      if int(msg and 2) == 2:
        spawn httpDownload("http:// 仪器 IP /cgi-bin/realtime.cgi", "FNczb", realtime.parse) # 分别启用不同的线程来下载数据
```

2.3.2 数据流向控制

“”“”“”“”‘,“length“”“”“”“”“”“”

数据流控制一方面是通过下载函数,在下载数据时将各个仪器分开处理,另一方面是通过以下函数实现的:

```
## 在 UI 无事件时,执行此函数操作
proc idleActionCB():cint {.cdecl.} =
  let (avai,msg) = uichannel.tryRecv() # 接受主调度模块传来的绘图数据
  if avai :
    let (t,data) = msg
    case t
    of "CDczb":
  uiUpdateCDczb(cast[Realtime](data)) # 更新数据
    else:
 os.sleep(100)
  return iup.IUP_DEFAULT
```

2.3.3 绘图

不同种类的仪器在绘图上有一些差异,下述是水管仪的绘图代码:

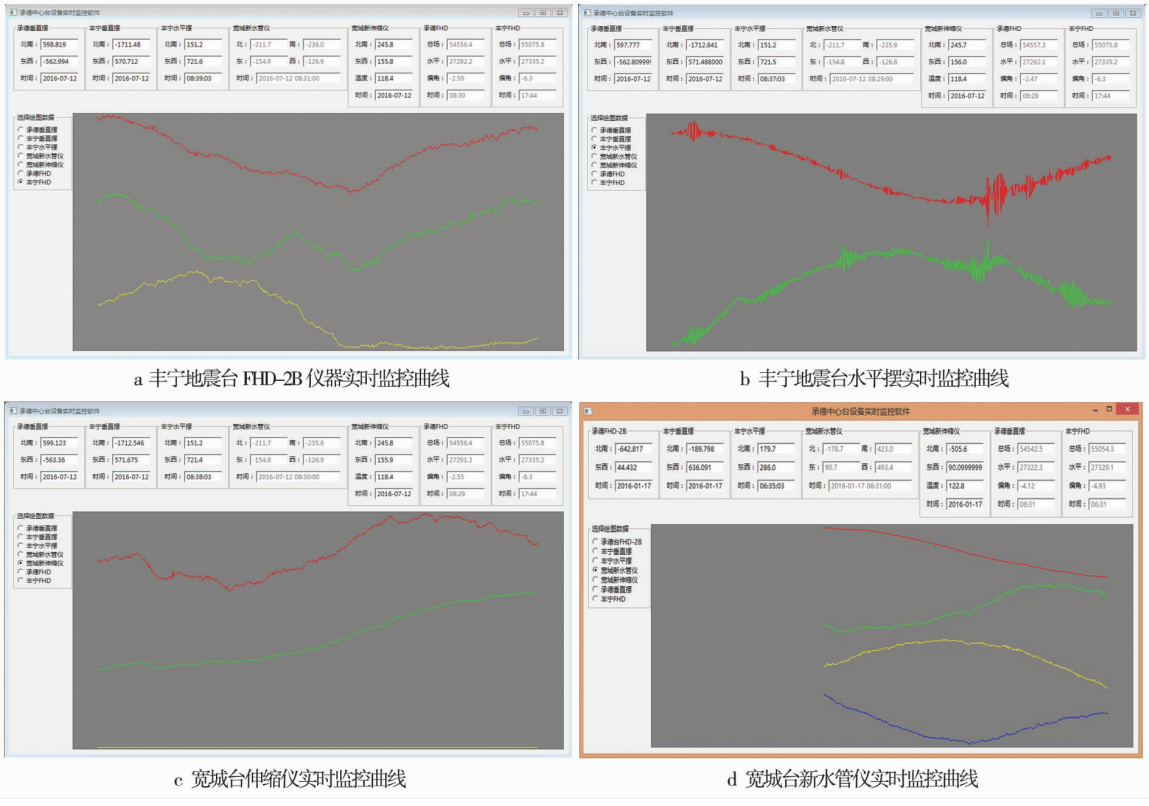
```
proc render * (shaderInfo:ShaderInfo, vaos:seq[GLuint], data:var seq[seq[float32]]) {.procvar.} =
  var minmaxArray = [[3.40282e+038'f32, -3.40282e+038'f32],[3.40282e+038'f32, -3.40282e+038'f32],[3.40282e+038'f32, -3.40282e+038'f32],[3.40282e+038'f32, -3.40282e+038'f32]]
```

```
glBindVertexArray(vaos[0]) # 绑定 OpenGL VAO
for i in 0..3:
    var gpuDataPoint = cast[ptr array[toInt(NUM_VERTEX_ARRAY/4), float32]](glMapBufferRange(GL_ARRAY_BUFFER, cast[GLintptr](i * toInt(NUM_VERTEX_ARRAY/4) * sizeof(float32)), cast[GLsizeiptr](toInt(NUM_VERTEX_ARRAY/4)), GL_MAP_WRITE_BIT))
    if gpuDataPoint != nil:
      for j in countup(0, toInt(NUM_VERTEX_ARRAY/8-1)):
        gpuDataPoint[j*2] = toFloat(j)
        gpuDataPoint[j*2+1] = data[i][j]
        if data[i][j] > -3.40282e+038'f32 :
          if data[i][j]<minmaxArray[i][0]:
            minmaxArray[i][0] = data[i][j]
          if data[i][j]>minmaxArray[i][1]:
            minmaxArray[i][1] = data[i][j]
        if glUnmapBuffer(GL_ARRAY_BUFFER) == false:
          echo "glUnmapBuffer didn't success."
    var red=[1.0f, 0.0f, 0.0f, 1.0f] # 找出最大最小值,绘图

    glUniform4fv(shaderInfo.uniformInfos["vColorValue"], 1, addr(red[0]))
    glUniform1f(shaderInfo.uniformInfos["minValue"], minmaxArray[0][0])
    glUniform1f(shaderInfo.uniformInfos["spanValue"], minmaxArray[0][1] - minmaxArray[0][0])
    glUniform1f(shaderInfo.uniformInfos["translationY"], 0.525f32) # 给显卡发送数据
    glDrawArrays(GL_LINE_STRIP, 0, cast[GLsizei](toInt(NUM_VERTEX_ARRAY/8)))
    # 数据绘图
```

3　系统主要作用及使用效果

原始数据实时采集、显示系统用于对承德中心台数字化观测仪器原始数据实时监控,可以在界面上直观看到各仪器实时原始数据,同时还可以通过选择仪器查看 12 h 内的原始数据曲线,部分仪器曲线显示如图 3。



a 丰宁地震台 FHD-2B 仪器实时监控曲线　　b 丰宁地震台水平摆实时监控曲线

c 宽城台伸缩仪实时监控曲线　　d 宽城台新水管仪实时监控曲线

图 3　部分仪器实时监控曲线

4 结论

本文采用 MVC 架构，利用 NIM 语言进行程序研发，以分为单位进行数据采集，利用 channel 技术进行数据流向控制，通过对显卡编程，实现数据绘图。系统有以下优点：

1)实现一个软件对不同台站、不同学科、不同仪器原始数据同时进行实时监控，减少了台站工作量，提高台站工作效率。

2)实现以 12 h 内第一个数据作为坐标原点，实现坐标系实时转换，更好地展示原始数据变换形态。

3)目前，该软件设计了 2 个版本，基于 Windows32 位系统与 Windows64 位系统各一套，但尚存不足，如某些仪器记录数据因网络问题出现连接不上的情形。该软件未设置数据中断立即报警的功能，在日常工作中可能就会出现一些失误，因此，在未来软件设计过程中，需要针对目前软件运行中所出现的问题做进一步完善。

参考文献：

[1] 程岩，汤永佐，刘岩. 基于 VC++实现的实时数据监控和显示方法[J]. 山东科学，2010(2)，83-85.

[2] 秦虹. 气象实时数据监控程序的设计与研发[J]. 安徽农业科学，2014(42)：9434-9439.

[3] 彭宏伟，朱文成，胡治国. 无人值守台站电源远程监测系统的设计[J]. 防灾科技学院学报，2010(3)：44-47.

Development of A Real-time Data Monitoring System for Digital Instrument at Chengde Centeral Seismic Station

ZHANG Yu-lin[1], MA Qian-li[2], CHEN Li-qiang[1], GUO Ya-ya[1], SONG Xiao-bing[3], SUN Gui-cheng[1]

(1. Chengde Central Seismic Station, Hebei Earthquake Agency, Chengde 067000, China;

2. Beijing Shiyang Software Development Co., Ltd., Beijing 100000, China;

3. Zhangjiakou Central Seismic Station, Hebei Earthquake Agency, Zhangjiakou 075000, China)

Abstract: In this study, through the NIM language recompiled by the C language, using the programming operation of the graphics card, a data real-time monitoring system is established and put into use on the 7 sets of instrument built during the China's 15th Five-Year Plan at Chengde central seismic station. The system take the first data of 12 hours as a coordinate origin making real-time coordinate conversion, and rendering 12 hours of raw data real-time curve, and then realizes data real-time monitoring for attendant to view at any time.

Key words: real-time data monitoring system; Chengde central seismic station; instrument built during the "Tenth Five Year Plans" of China