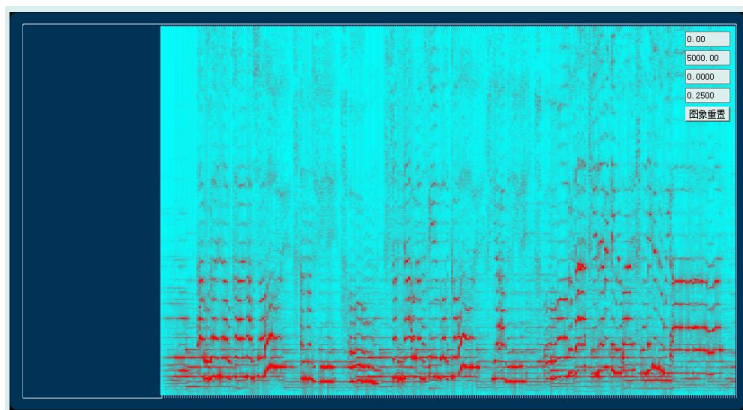


DRMeshFast 实时语谱图显示控件用法

DRMeshFast 实时语谱图显示控件主要用于显示随信号实时变化的语谱图。因需要耗费较多的显示资源，该控件只支持 pyqt 版本；另外，该控件是为教学中示例自定义控件设计方法而做，并不完善，仅用于展示自定义控件的设计概念。



DRMeshFast 实时语谱图显示控件显示效果图

在 Python 程序 pyqt GUI 界面中创建一个 DRMeshFast 实时语谱图显示控件的命令和语法格式如下：

```
hl= DRMeshFast(win,x,y,w,h,option=value, ...)
```

其中输入参数 win 为 pyqt 库创建的 GUI 窗口句柄，[x,y,w,h]四个参数为该控件插入 GUI 窗口 win 时的左上角坐标[x,y]和该控件的宽、高[w,h]。另外是一组可选的输入参数，包括：

cb: 背景色，默认值='#003355'

cb: 背景色，默认值='#ffffff'

M: 语谱图中的谱图数量，默认值=400

xlow: Y 轴(频率轴)最小值，默认值=0Hz

xhigh: Y 轴(频率轴)最大值，默认值=10000Hz

ylow: Z 轴(幅值，色彩表示)最小值，默认值=0

yhigh: Z 轴(幅值，色彩表示)最大值，默认值=1

返回值 hl 则是该控件创建后的句柄变量名，可以通过 hl 对创建后的控件进行进一步的设定。控件创建时。

DRMeshFast 实时谱阵显示控件有 1 个属性函数，其功能定义如下：

hl.setValue2D(t,Amp): 添加当前频谱到谱阵中，t 和 Amp 为一维数组，对应频谱曲线的频率和幅值数据。

下面是一个 DRMeshFast 实时谱阵显示控件创建实例：

```
hl=dr.DRMeshFast(self.win,77,118,1000,400,M=400,xlow=0,xhigh=5000,ylow=0,yhigh=0.25)
```

通常，可以选择设定部分选择参数的形式进行设定，除基本参数 win、[x,y,w,h]和用变量名设定的可选参数外，其余可选参数采用默认值。创建后可以通过返回的控件句柄对其属性函数进行操作。

当采用 DRPython 进行图形化编程时，控件的创建代码和直连绑定代码的生成均可以由机器实现；控件创建只需要将所需控件拖拽到 GUI 布局区，而直连绑定则需要在 GUI 布局区的控件间进行连线操作。DRMeshFast 控件为输入控件，在连线状态下当上一控件已选输出函数后，在控件上点击鼠标，则弹出连线选择菜单；选择完成后，若输出端和输入端类型一致，则完成连线。输入端口(setValue2D)收到的是频谱图数据。



下面是 DRMeshFast 实时语谱图控件的创建和应用示例小程序。功能是用一个信号发生器连续生成信号，然后显示和计算信号的波形与幅值谱，并将当前幅值谱添加到 DRMeshFast 实时语谱图控件中，实现语谱图的实时显示。示例小程序代码如下：

```
import sys
from PyQt5.QtWidgets import *
from PyQt5.QtGui import *
from PyQt5.QtCore import *

import numpy as np
import drvi.drviDSP as dsp
import drvi.drviQTControls as dr

#==Start of MainClass=====
class drviMain(QMainWindow):
    def __init__(self):
        super(drviMain, self).__init__()
        self.initGUI()

#==Init GUI=====
def initGUI(self):
    self.setWindowTitle("APP Frame by DRPython")
    self.resize(1100, 720)
    self.win=QWidget()
    self.win.setStyleSheet("background-color : #ddeeee; ")
    self.setCentralWidget(self.win)

    #Add controls
    self.mButton1=dr.DRButton(self.win, 20, 20, 100, 40, txt='运行/停止')
    self.mHSlider=dr.DRHSlider(self.win, 150, 20, 900, 30, '#444444', '#880000', '#008822', 8, 20, 0, 1000, 20)
    self.mGenerator3=dsp.DRGenerator(2, 5000, 4096, 0.6, 100, 0, 250)
    self.mPlot4=dr.DRPlotFast(self.win, 20, 70, 530, 160, xlow=0, xhigh=0.05)
    self.mPlot5=dr.DRPlotFast(self.win, 560, 70, 530, 160)
    self.mSpectrum6=dsp.DRSpectrum()
    self.mPlot7=dr.DRMeshFast(self.win, 20, 240, 1070, 430, cb='#003355', cf='#ffffff', M=400, xlow=0, xhigh=1500, ylow=0, yhigh=0.5)
    self.mButton1.addCallbackSingle(self.mGenerator3.setVSOonOff)
    self.mHSlider2.addCallbackSingle(self.mGenerator3.setVSFre)
    self.mGenerator3.addCallback2D(self.mPlot4.setValue2D)
```

```

        self.mGenerator3.addCallBack2DX(self.mSpectrum6.setValue2DX)
        self.mSpectrum6.addCallBack2DAmp(self.mPlot5.setValue2D)
        self.mSpectrum6.addCallBack2DAmp(self.muserArrayTwo)

    def muserArrayTwo(self, tArray, xArray):
        self.mPlot7.setValue2D(tArray, np.sqrt(np.sqrt(xArray)))

#==Main Loop=====
app=QApplication(sys.argv)
app.setStyle("Fusion")
win=drviMain()
win.show()
sys.exit(app.exec())

```

可参考实验“003_QT 媒体文件信号谱阵分析.py”和“005_QT 麦克风谱阵分析.py”。