

oym8CHWave 无线肌电数据采集

快速使用手册

V0.1

2020 年 8 月 6 日

一、启动 oym8CHWave.exe

将 gForce-Dongle 插入 Windows 10 电脑的 USB 插口。运行 oym8CHWave.exe

☐ 名称

- ☐ bearer
- ☐ iconengines
- ☐ imageformats
- ☐ platforminputcontexts
- ☐ platforms
- ☐ qmltooling
- ☐ scenegraph
- ☐ styles
- ☐ translations
- ☐ virtualkeyboard
- ☐ gforce32.dll
- ☐ gforce64.dll
- ☒ oym8CHWave.exe
- ☐ qt.conf
- ☐ Qt5Core.dll
- ☐ Qt5Gui.dll
- ☐ Qt5Network.dll
- ☐ Qt5Qml.dll
- ☐ Qt5Quick.dll
- ☐ Qt5SerialPort.dll
- ☐ Qt5Svg.dll
- ☐ Qt5Widgets.dll
- ☐ zh_CN.qm



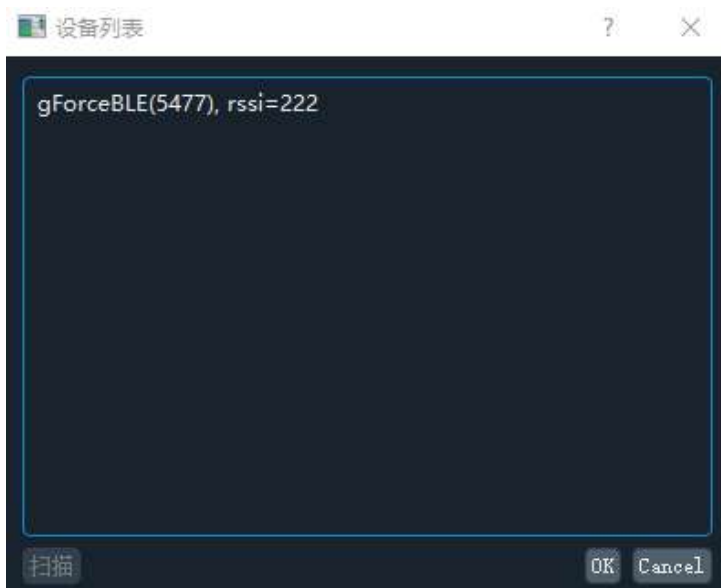
二、连接 gForce-Pro 肌电臂环或 gForce-Octopus 无线肌电采集仪



点击左上角“菜单”，选择“连接到 gForce”。



打开 gForce-Pro 或者 gForce-Octopus 电源（轻按设备上的按钮，绿色 LED 灯慢速闪动），点击弹出的“设备列表”对话框左下角“扫描”按钮，进行设备查找。如果设备没有找到，可等“扫描”按钮变亮后重新点击，如果问题依旧，可关闭程序，拔插 gForce-Dongle，重新从第一步开始。



点击选择需要连接的设备，点击下方“OK”按钮进行连接。

注：本程序只能连接到傲意的产品。

三、选择数据位数和采样率



设备支持 8bit 和 12bit 两种 ADC 采样模式。

在 12bit 模式下，采样率设置最高为 500Hz。

选择模式和，点击下方“OK”按钮，进行设备连接。

正常情况下，几秒后程序和 gForce 成功建立无线连接，数据在页面上实时以波形形式呈现。



注 1: gForce 没有佩戴情况下 (传感器悬空), 数据极易受环境影响, 此时数据是无效的。请确保 gForce 已经佩戴妥当。

注 2: 肌电是一种非常弱的生物电信号, 极易受交流电源的工频干扰。在数据采集过程中, 为降低数据干扰和避免电击危险, 用户严禁在使用中插上 USB 充电使用。

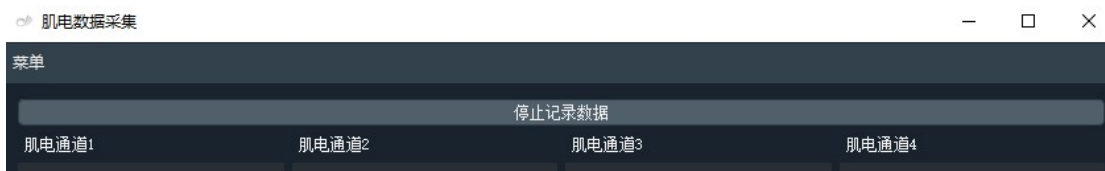
下图为佩戴 gForce 后, 肌肉不用力时, 所有的通道应接近直线 (DC 电平), 数据在 120 左右 (此为信号采集过程中加入的 offset, 使用中需先进行 DC 去除)。



四、数据采集保存



点击波形上方的“记录到文件”按钮, 开始数据采集和保存, 按钮变化为“停止记录数据”。



当用户需要停止数据采集时, 点击“停止记录数据”, 弹出“文件已保存”窗口, 询问用户是否要将数据另存到其他位置。



根据实际需要选择“Yes”或者“No”。

注: 默认的文件保存在\data 目录下, 文件名为以 EMG_开始, 后跟采集时的日期和时间, 后跟 8bits 或者 12bits 属性, 最后是采样率属性。

五、连接其他设备或者修改采样属性

点击左上角“菜单”, 选择“连接到 gForce”, 系统询问是否要断开当前连接:



选择“Yes”, 重新进行设备扫描、参数设置、设备无线连接。

如果 gForce 设备查找不到, 请关闭 gForce 电源 (长按按钮 5 秒后释放), 再打开 gForce 电源 (轻点按钮), 进行设备查找。如此问题持续出现, 轻关闭程序, 拔插 gForce-Dongle, 重新运行。

六、数据格式

12bit 模式:

00000000	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00000000	79	07	65	07	6d	07	7c	09	d3	0e	d7	0e	d9	0e	9e	0c
00000010	72	07	50	07	5c	07	7a	08	d9	0e	dd	0e	df	0e	27	0a
00000020	6f	07	5c	07	68	07	6d	03	73	05	00	00	00	00	10	02
00000030	5e	07	53	07	5e	07	1b	04	06	00	01	00	00	00	00	00
00000040	5a	07	4c	07	39	07	f6	04	05	00	00	00	00	00	70	00
00000050	70	07	55	07	89	07	73	06	a2	01	00	00	00	00	f3	05
00000060	74	07	59	07	6f	07	d9	0b	e5	0e	a9	07	bb	05	ce	0e
00000070	71	07	57	07	6f	07	e4	0a	e8	0e	d6	0e	c8	0e	cf	0e
00000080	72	07	5a	07	6f	07	8d	09	e8	0e	d2	0e	c9	0e	76	0c
00000090	6b	07	56	07	68	07	92	08	e8	0e	d8	0e	cb	0e	c7	09
000000a0	71	07	52	07	5b	07	72	03	ac	04	00	00	00	00	74	01
000000b0	5f	07	54	07	5d	07	15	04	07	00	01	00	00	00	00	00
000000c0	56	07	53	07	47	07	ee	04	05	00	00	00	00	00	98	00
000000d0	7e	07	63	07	92	07	bf	06	a1	01	00	00	00	00	9a	06
000000e0	71	07	58	07	6f	07	d5	0b	e3	0e	9e	08	bc	06	de	0e
000000f0	77	07	61	07	71	07	b2	0a	cf	0e	d2	0e	d7	0e	e2	0e
00000100	77	07	62	07	70	07	70	09	cf	0e	d3	0e	d6	0e	69	0c
00000110	64	07	4d	07	5f	07	48	08	e1	0e	e5	0e	e6	0e	bd	09

所有通道的数据以 16bit (word) 方式依次存放。所有数据都是正数。

通道 0: byte[1], byte[0] 构成 16bit 有效数据 (word), 其中 bit[15:12]==0.

通道 1: byte[3], byte[2] 构成 16bit 有效数据 (word), 其中 bit[15:12]==0.

...

通道 7: byte[15], byte[14] 构成 16bit 有效数据 (word), 其中 bit[15:12]==0.

依次再是通道 0~通道 7, 通道 0~通道 7.

8bit 模式:

00000181	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00000000	76	75	ed	78	75	76	6b	68	76	75	ed	78	75	76	73	8e
00000010	77	75	00	74	75	76	86	85	76	73	00	6e	75	76	7a	7b
00000020	76	74	00	6e	75	76	74	76	76	75	00	72	75	77	74	74
00000030	76	75	ed	87	77	77	71	66	76	75	ee	7e	76	77	6c	65
00000040	76	75	ed	78	75	77	6c	69	77	75	ee	78	75	76	75	8e
00000050	77	75	00	74	75	76	85	84	76	73	00	6e	75	76	79	7a
00000060	77	76	00	6f	75	76	74	76	76	75	00	72	75	76	74	74
00000070	76	75	ee	87	76	76	71	65	75	75	ed	7e	76	77	6b	65
00000080	76	75	ed	78	75	76	6b	68	76	74	ed	78	75	76	79	8e
00000090	77	75	00	74	75	76	84	83	77	74	00	6d	74	76	78	79
000000a0	76	75	00	6f	75	76	74	75	76	76	00	73	75	76	73	73
000000b0	76	75	ed	87	77	77	71	66	75	75	ee	7d	76	76	6a	65
000000c0	76	75	ee	79	76	76	6b	69	77	75	ed	78	75	76	7b	8e
000000d0	77	75	00	74	75	77	84	84	77	74	00	6e	75	76	78	79

所有通道的数据以 8bit (byte) 方式依次存放。所有数据都是正数。

通道 0: byte[0]构成 8bit 有效数据。

通道 1: byte[1]构成 8bit 有效数据。

...

通道 7: byte[7]构成 8bit 有效数据。

依次再是通道 0~通道 7, 通道 0~通道 7.