**开发者文档 v1.2**

Latest Update: 2021/12/01

1. **使用流程**
   * 1. 主GUI：选择protocol，设定面板上需要的参数
     2. 子GUI：设定其他参数

主GUI在SET PARAMS按键回调函数中调用子GUI以设置protocol需要的在主GUI面板以外的参数

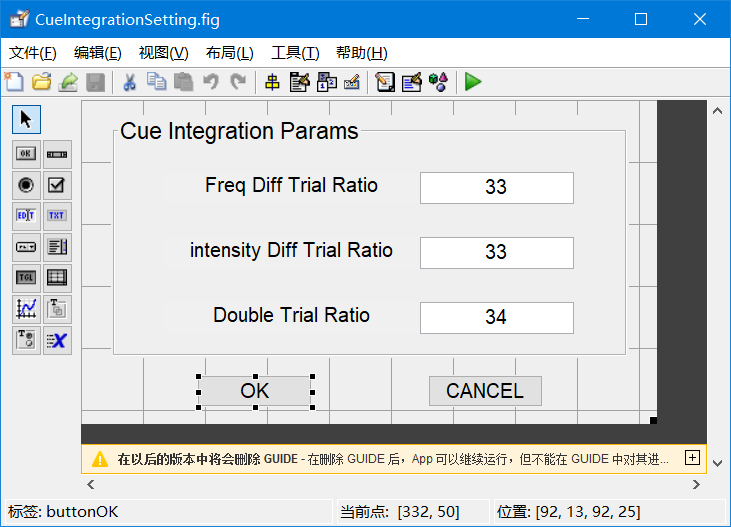
* + 1. 主GUI：按下START启动
    2. 等待训练/记录完成
    3. 主GUI：按下STOP结束

1. **新增protocol**
2. 在protocolList.mat中添加新的protocol信息，包括：

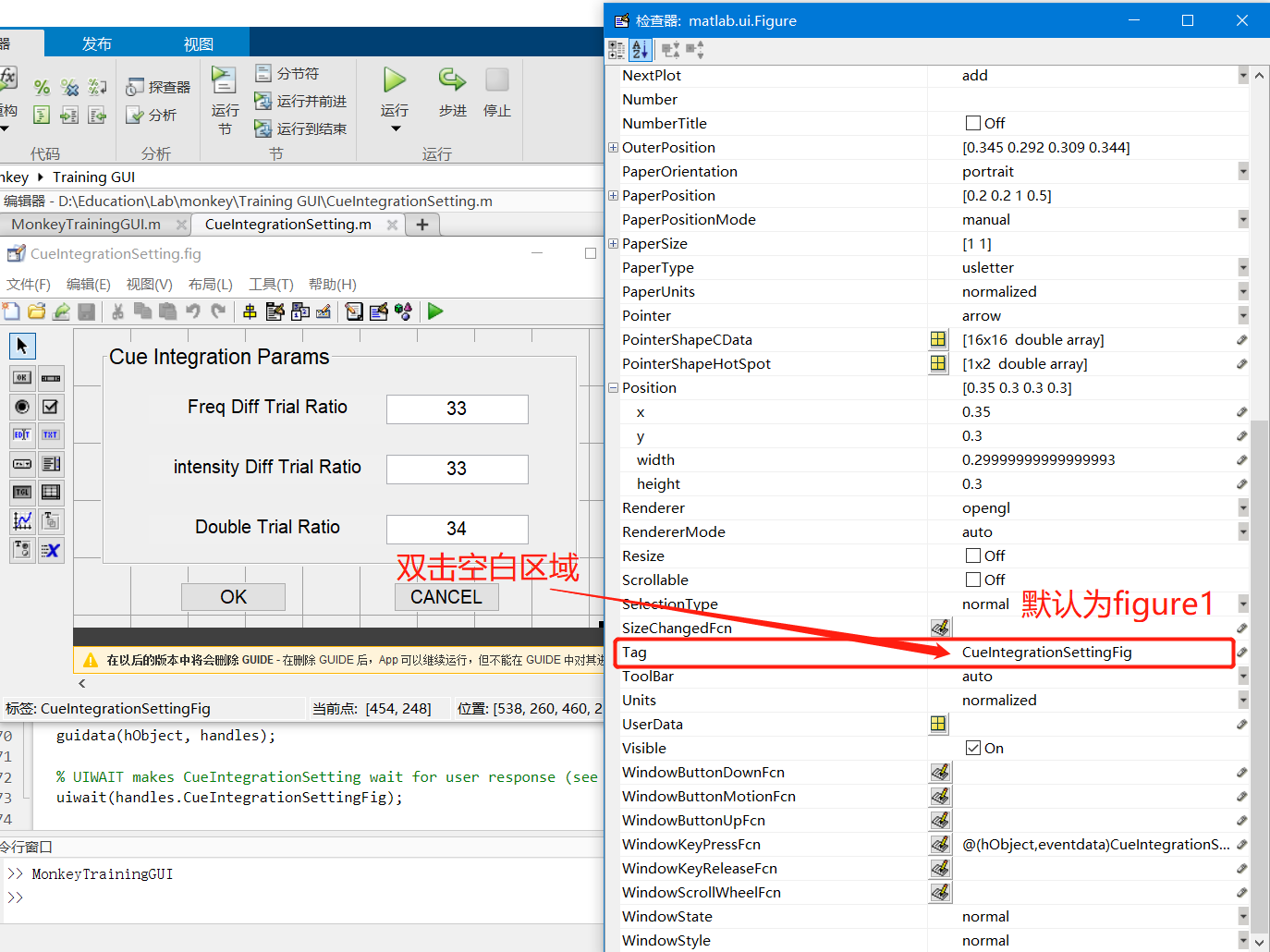
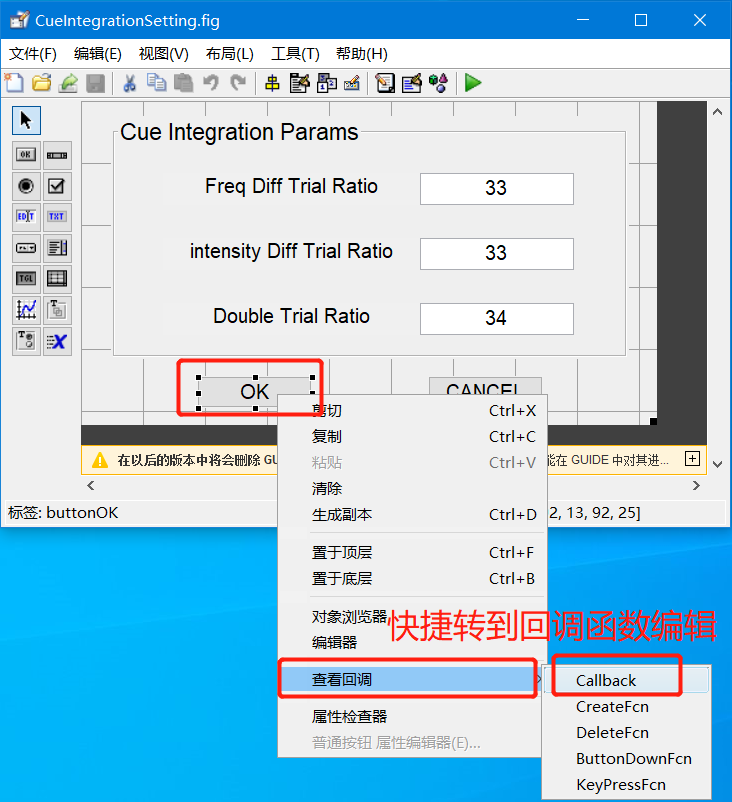
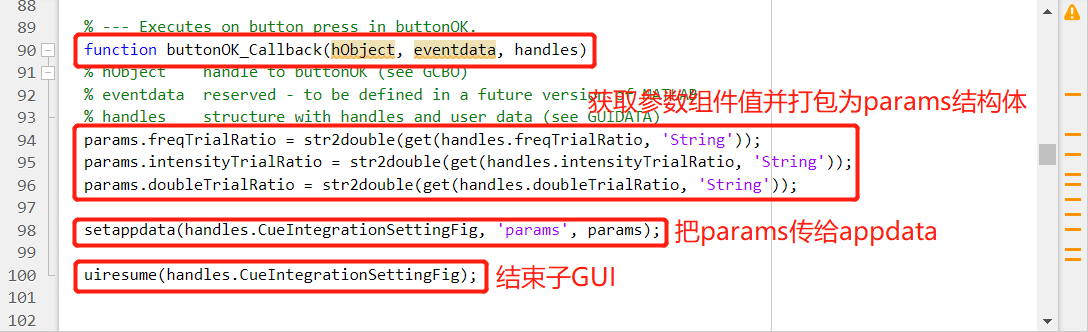
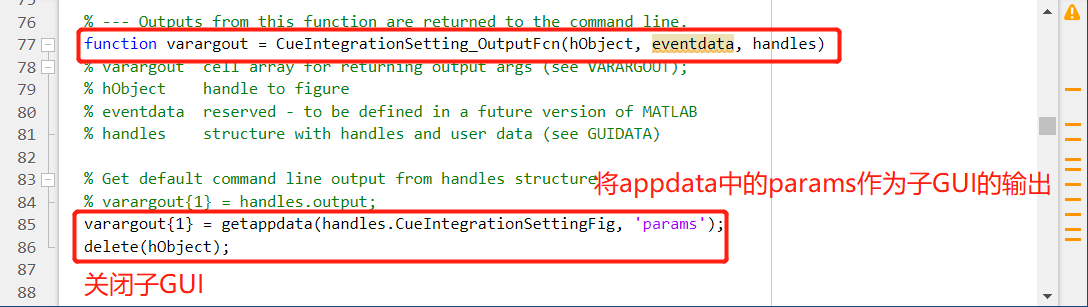
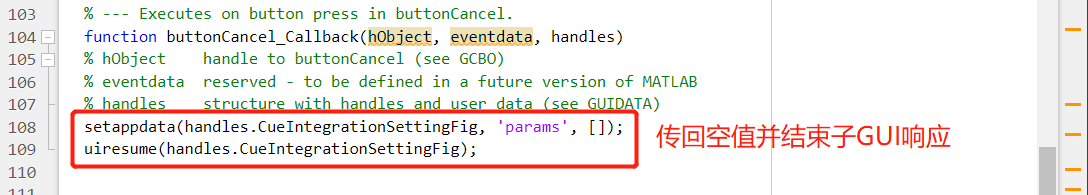
|  |  |  |
| --- | --- | --- |
| **字段名** | **含义** | **示例** |
| **protocolCode** | protocol编码，唯一标识符 | 1010101 |
| **protocolName** | protocol名 | PEOddBasicPassiveTone |
| **protocolDetail** | protocol详情信息 | 7-10纯音被动 |
| **settingFcn** | protocol专用参数设置子GUI函数句柄 | @CueIntegrationSetting |
| **stimuliFcn** | 刺激函数句柄 | @CueIntegrationStiFcn |
| **serialCallbackFcn** | 串口回调函数句柄 | @GeneralSerialFcn |

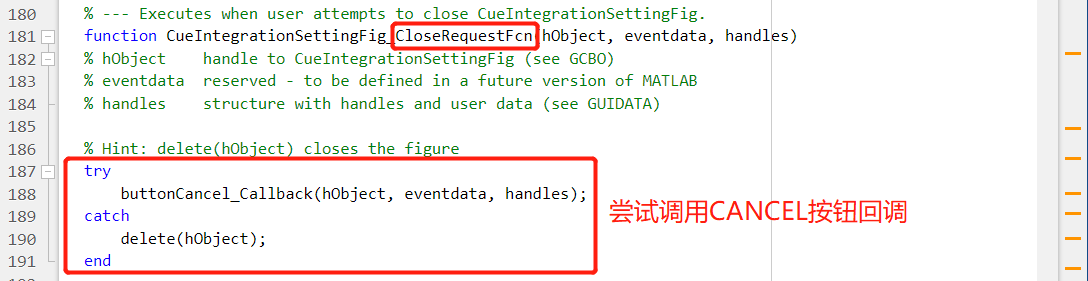
1. 编写settingFcn对应的setting的GUI（如果需要额外参数输入）

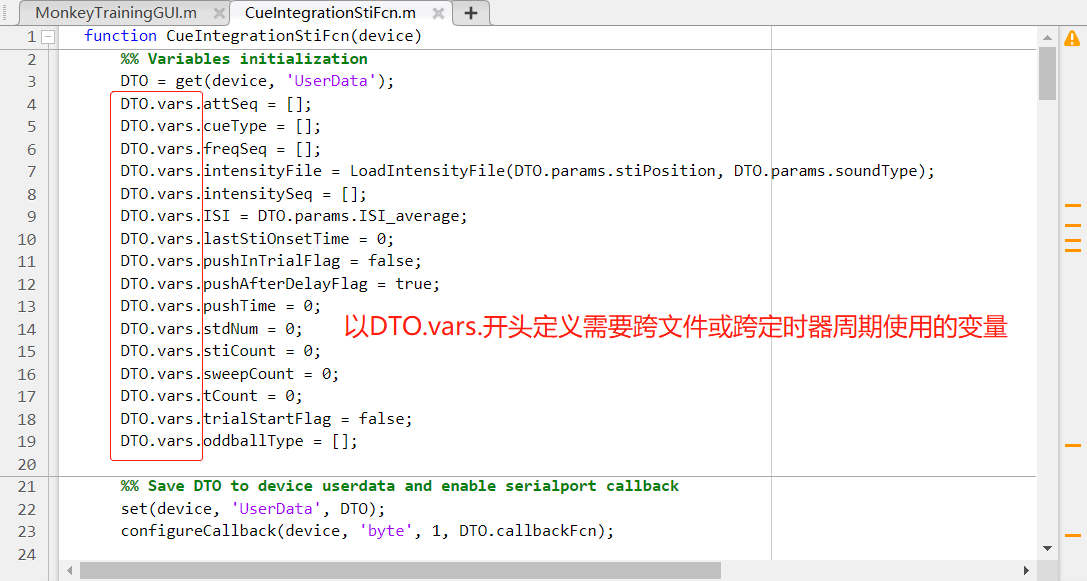
以protocol名p1为例，使用GUIDE创建作为子GUI的p1Setting.m和p1Setting.fig

1. 在\*.fig中配置自己的参数获取组件，如edit和checkbox，并添加OK按键
2. 对参数获取组件的tag属性进行**标准化命名**，更新变量文档的params区

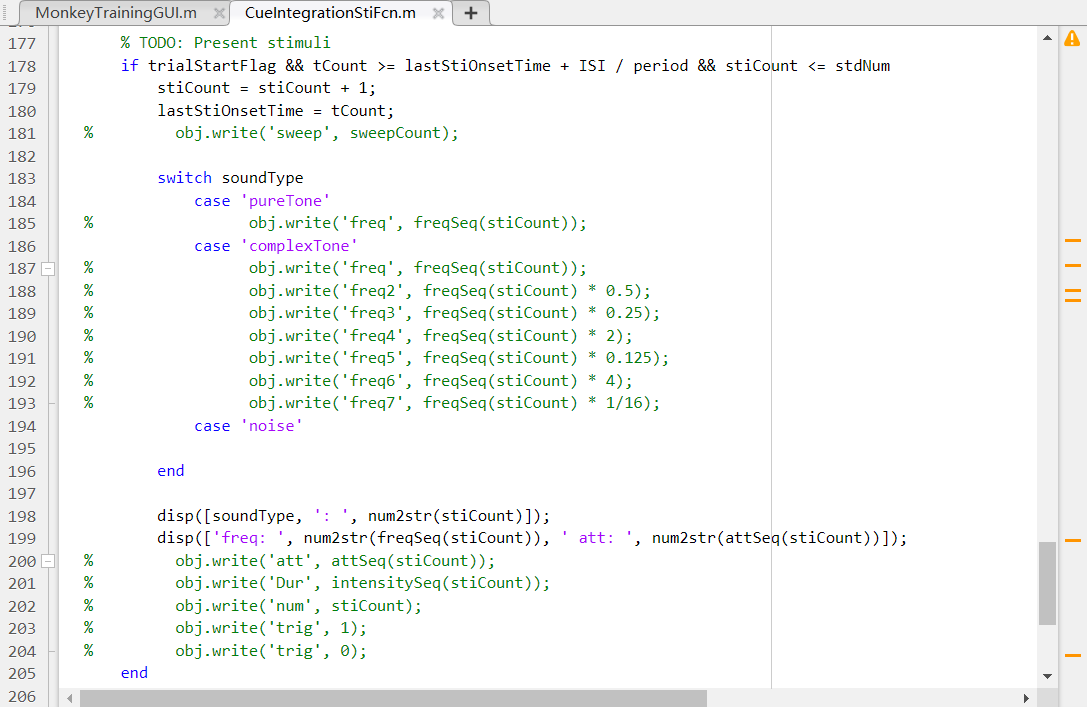
* 命名标准：
  + 变量：lowerCamelCase，首字母小写，后面单词首字母大写，要求见名知意，严禁使用单字母和不明意义的命名。好的命名如：frequencyStd, intensityDev；不好的命名如i, kkk, stdaz
  + 函数名：UpperCamelCase，所有单词首字母大写

1. 在p1Setting.m中找到p1Setting\_OpeningFcn函数，这是GUI生命周期函数，在GUI创建时执行。取消uiwait(handles.figure1); 这一行的注释，目的是让主GUI打开子GUI时等待子GUI响应，直到遇到uiresume再执行之后的语句
   * + - **选做：**子GUI通过paramsProtocol = p1Setting(paramsLoad)打开，在这里可以通过paramsLoad = vargarin{1}; 获取主GUI传进来的预加载的参数。在这里初始化一些变量并以set(handles.component1, ‘string’, paramsLoad.component1); 这类方式改变组件显示值，实现为子GUI导入文件参数的功能
2. GUI传参方式统一为setappdata(handles.figure1, ‘paramName’, paramValue); 和getappdata(handles.figure1, ‘paramName’); 其中figure1为GUI最外层figure的tag，默认为figure1，可酌情修改
3. 编辑OK按键的回调函数，获取组件中的参数并打包为params结构体，然后执行uiresume，这会转到OutputFcn。具体组件的参数获取和传递方式参考DemoSetting.m
4. 找到p1Setting\_OutputFcn函数，将varargout{1}设置为打包好的params，然后执行delete(hObject); 关闭子GUI并将参数传出，主GUI接收参数后存在自己的appdata里面
5. 选做：添加CANCEL按钮和窗口关闭事件（右上角×关闭子GUI）



1. 编写stimuliFcn对应的刺激函数p1StiFcn.m，请修改带TODO注释位置
2. 确定需要的变量并在开头初值化，更新变量文档的vars区



1. 在定时器回调函数mTimerFcn中编写自己的刺激序列生成，可以按注释一项项写也可以自己添加新的，但注意用到的所有要跨定时器周期传递或需要传给串口程序的变量要在前面的DTO.vars中定义
   * + - 刺激程序和串口程序的数据传递使用device的UserData属性，存储的结构体名为DTO（Data Transfer Object），device是一个serialport对象，DTO除了vars以外不需要编辑
2. 编写产生刺激所需要发送给TDT的变量
3. 编写serialCallbackFcn对应的串口回调函数p1SerialFcn.m

如果没有特殊需求，请使用@GeneralSerialFcn作为serialCallbackFcn

1. **维护**

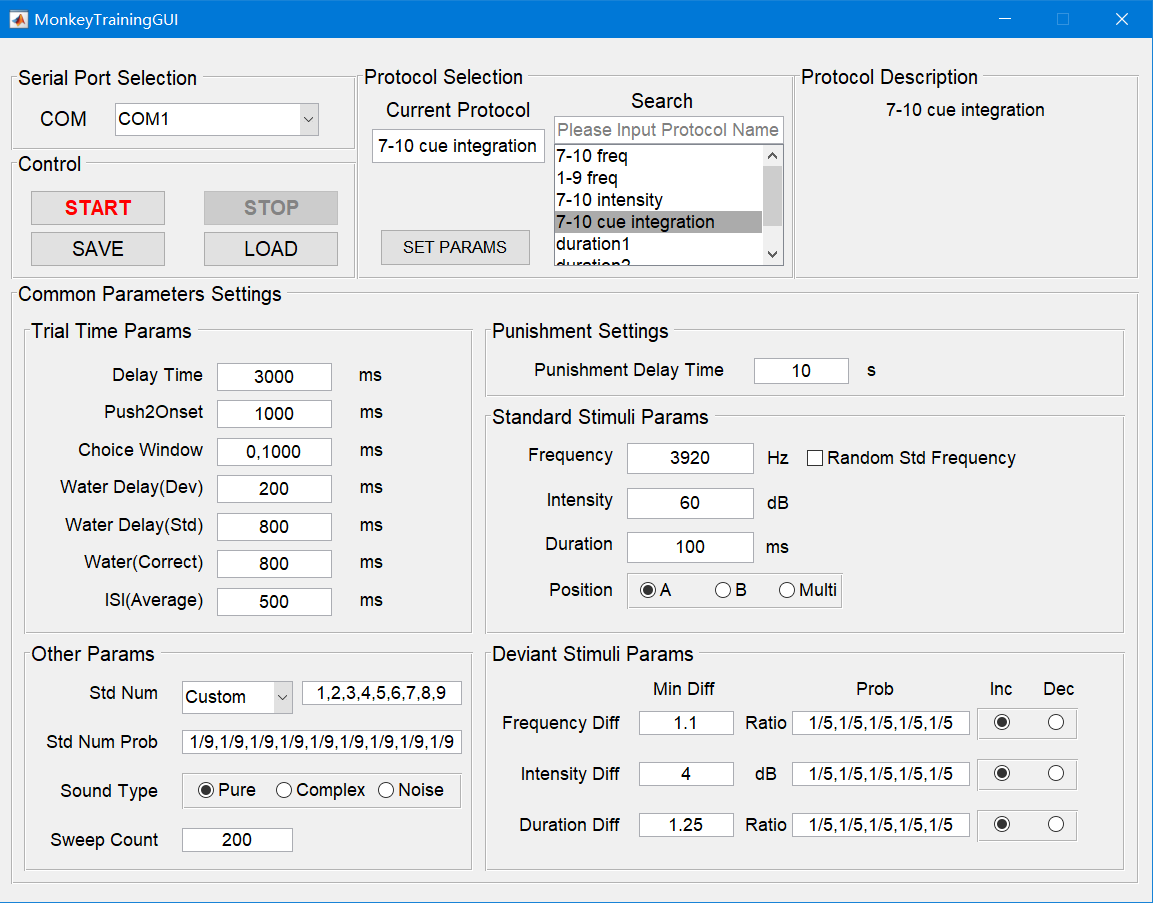
写在前面：这个项目相当于一个平台，保留基本功能以外可以通过接口集成其他GUI的功能，核心在于对传输对象的打包处理，标准化了数据传输过程，取消了global这类不安全的传输方式。

——Xu, 2021/12/01

* 1. 程序组成

分类打包放在主目录下的文件夹中。

* + 1. MonkeyTrainingGUI.m, \*.fig

主GUI，包括串口选择、按钮控制面板、protocol检索选择和详情、通用参数设置、protocol专用参数设置（接口，集成在SET PARAMS回调中，函数ProtocolSetParams.m）

* + - 1. 串口选择：COM1至COM8可选，可拓展功能（未实装）：通过serialportlist查看可用串口自动选择
      2. Protocol检索选择和详情
         1. 检索逻辑：将keyword分割为单个字符，只要protocol名中按顺序包含这些字符就纳入检索结果
         2. 选择逻辑：单击列表protocol名在右侧显示详情；双击则选中作为current protocol并弹出相应的参数设置子GUI（如果有）；单击选中+按下SET PARAMS=双击
      3. 通用参数设置：见变量文档
      4. 按钮控制面板
         1. START：开始，根据当前protocol调用相应的刺激程序
         2. STOP：停止，关闭串口回调（不清除串口对象）、清除定时器
         3. SAVE：将当前所有参数（包括主GUI和子GUI中的）保存在指定目录下，默认命名为当前日期\_params.mat
         4. LOAD：将指定\*.mat中的参数导入主GUI，并将所有文件参数存在appdata的paramsLoad中，供子GUI调用时预加载
    1. SearchKeywordInProtocolList.m

Protocol检索逻辑，对象为protocolName，搜索用的正则表达式为[‘\S\*’, keyword(1), ‘\S\*’, keyword(2), ‘\S\*’, … , ‘\S\*’]

* + 1. ProtocolSetParams.m

根据选择的protocol无参数输入执行其对应的settingFcn函数句柄，返回值为paramsProtocol结构体，将paramsProtocol中的变量添加到主GUI的params变量中再存入appdata

* + 1. GetParams.m

获取主GUI界面中的参数并打包为params存入appdata

* + 1. LoadParams.m

将选择的\*.mat文件中的参数设置到主GUI界面中显示

* + 1. LoadIntensityFile.m

预加载声强文件

* + 1. CalAttenuation.m

计算声强衰减值序列，输入频率序列和目标声强序列，返回衰减值序列，需要根据喇叭位置、声音类型设置对应的声强文件路径

* + 1. GeneralSerialFcn.m

通用串口回调函数

* + 1. ValidateParams.m

校验参数合法性

* + 1. SetProtocolUI.m

选定protocol时主GUI的响应事件，如改变某些参数值

* + 1. numstrcat.m

工具函数，将double array转为对应的用connector连接的分数形式的string

* + 1. 其他刺激程序和串口程序
  1. 数据传输
     1. GUI内部传输

统一使用appdata作为传输工具，appdata为figure的一个对象属性

* + - 1. setappdata(handles.figureTag, ‘data’, data);
      2. getappdata(handles.figureTag, ‘data’);
    1. 跨文件传输

使用Serialport对象的UserData的属性，程序用到的对象名为device，由于UserData不能重复set多个字段的变量，因此打包为一个DTO结构体（Data Transfer Object），其中包括了params，vars，period，callbackFcn，obj这几个属性字段。调用和更改通过set和get方法完成

* + - 1. 在StiFcn和SerialFcn文件开始通过DTO = get(device, ‘UserData’); 获取DTO，然后将其中的params和vars中的变量提取出来放到workspace中方便调用，在结束位置再把vars中的变量更新并重新set(device, ‘UserData’, DTO);
      2. DTO属性意义
         1. params: 从UI获取的常量参数
         2. vars: 用户在StiFcn.m开头定义的跨定时器周期或跨文件传输的变量
         3. period: 定时器周期，默认使用0.02s，尽量保证时间参数是period的整数倍
         4. callbackFcn: 当前protocol串口回调函数句柄
         5. obj: TDT对象
  1. 主要逻辑