Manual for the protocol generator

Table of contents

Manual for the protocol generator....1
Introduction....3
Acknowledgements....3
Preparation and installation....4
Preparation for Presentation *.log file before acquiring data....4
Installation....5
Using the protocol generator....6
Activating the plugin....6
Main menu....7
Option 1: generating a protocol file....8
After activating the plugin....8
Computation of the interval values in a condition....9
Begin of an interval....9
End of an interval....10
Generating the color values for each condition....11
Results....11
Option 2: changing the offset....13
Option 3: reordering the conditions....15
Appendix....16
Generating the plugin from source code....16
Generating the API documentation....20
History....21
Changes in the plugin....21
version 0.8....21
version 0.7....21
version 0.6....21
Changes in this document....21
version 0.3....21
version 0.2....21
version 0.1....21

Introduction

The protocol generator creates a BrainVoyager QX Protocol file (*.prt) in milliseconds from a Presentation Log file (*.log). Each Picture or Sound event will result in a single condition.
Latest version of the plugin: 0.8.

Acknowledgements

Thanks to Aline de Borst for providing Presentation *.log and BrainVoyager stimulation protocol (*.prt) files.

Preparation and installation

Preparation for Presentation *.log file before acquiring data

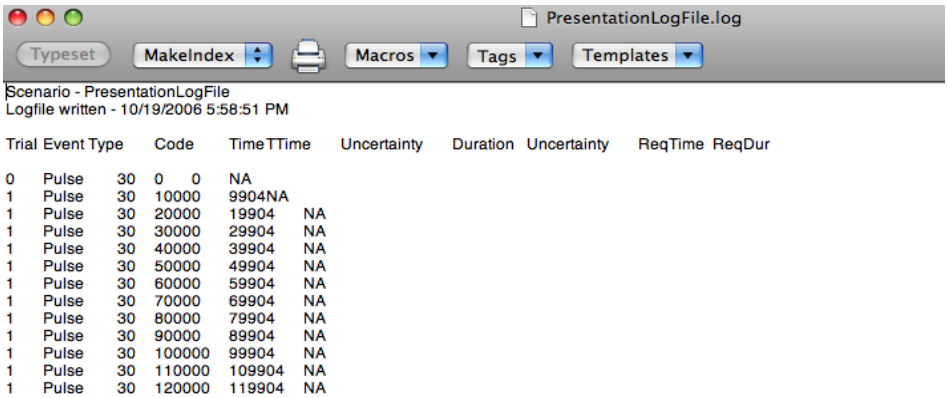
When using the protocolgenerator, it is important to have a regular output. Although in the heading more fields are announced, these values are only for when an event occurs written to the Log file.
For a pulse or response event type, the generator expects 6 columns to appear on each line:

```
Trial, EventType, Code, Time, TTime, Uncertainty, Duration
```

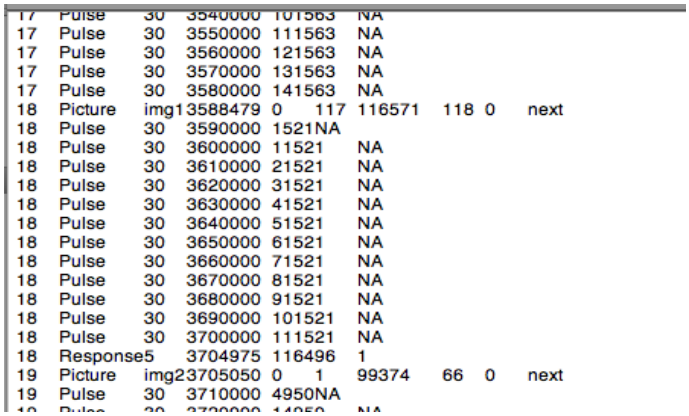
or

```
Subject, Trial, EventType, Code, Time, TTime, Uncertainty, Duration
```

For all other types (e.g. Pulse or Response), usually less values are written in the Log (see figure below)



In the figure below is shown that in the Presentation Log file 10 columns are written for a Picture event, and 6 values for a Pulse and Response. This is the expected format.



Please note that for event types, spaces in the name are not allowed. So for an event with the name `img nr2` please change the name to `imagenr2` (for example via the 'Replace all' function of a text editor).

Installation

Put the protocolgenerator.dll/*.dylib in /(My) documents/BVQXExtensions/Plugins/ folder.

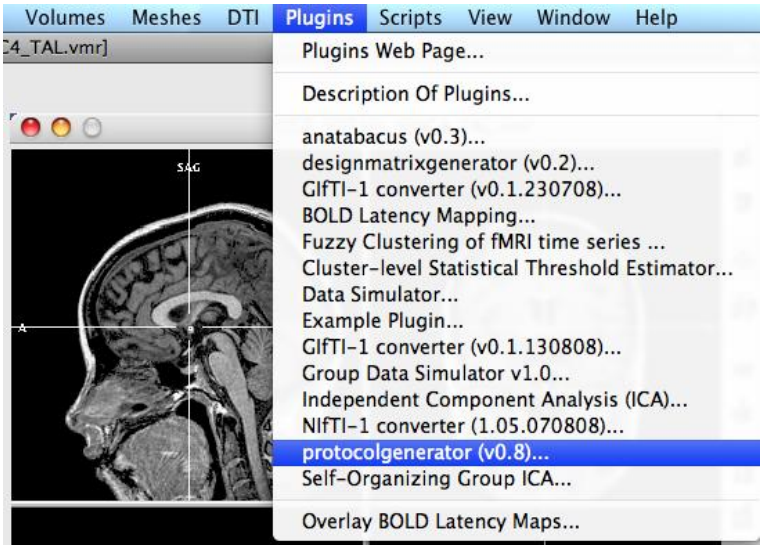
In case you have a operating system than the binaries available, use the source code to compile a version for your own platform (for instructions on compilation, please consult the Appendix).

Using the protocol generator

Open a BrainVoyager VMR project, so that conditions can be assigned to the new protocol. In case the offsets need to be changed, also a stimulation protocol should be linked to the *.vmr.

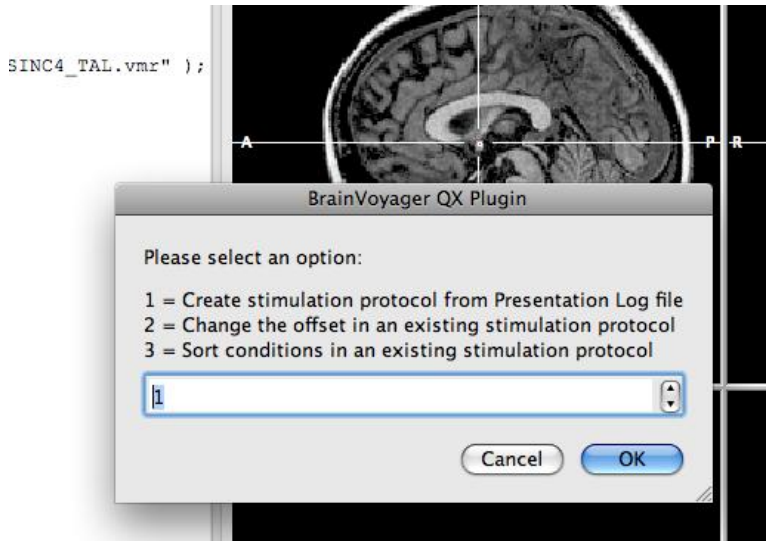
Activating the plugin

Start BrainVoyager QX and activate the protocol generator via the “Plugins” menu:



Main menu

After activating the generator via the Plugins menu, an input dialog shows the options of the protocolgenerator (see figure below).

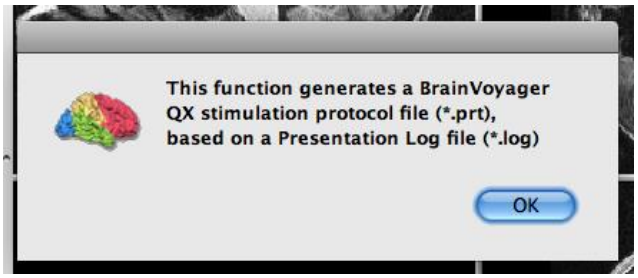


Option 1: generating a protocol file

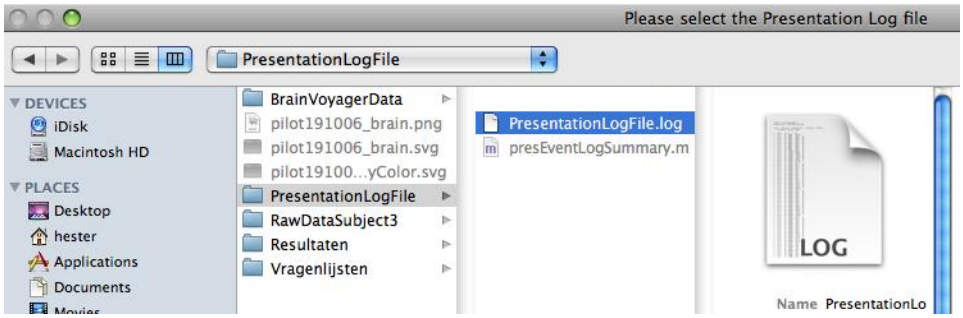
Before starting the plugin, open an anatomical image (*.vmr). It is not necessary to link a *.vtc file in order to use the plugin.

After activating the plugin

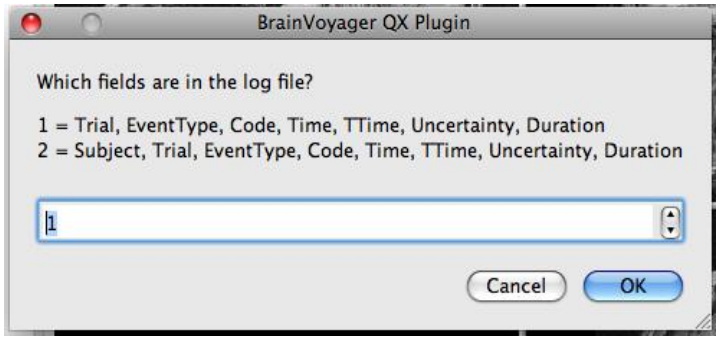
When the plugin is activated and option 1 of the menu has been selected, the following dialog will announce the functionality of menu option 1:



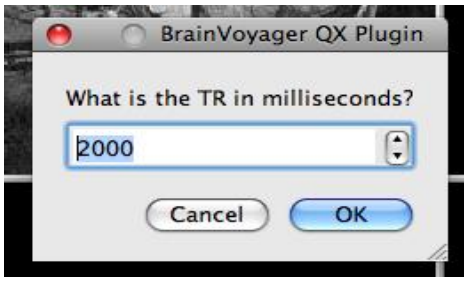
Select the Presentation Log file (*.log):



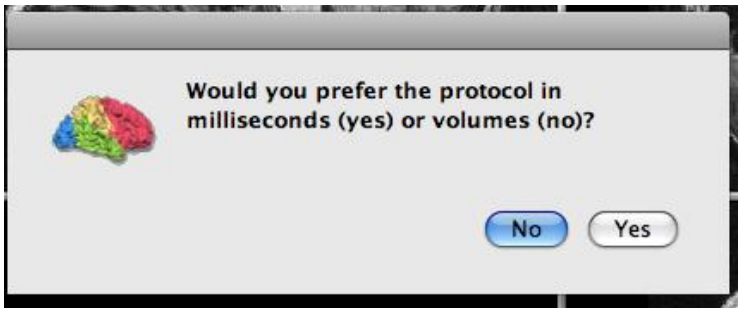
Now, one can choose the columns in the Log file that are (only) written for each event. For more exotic formats, please contact the plugin developer for change requests.



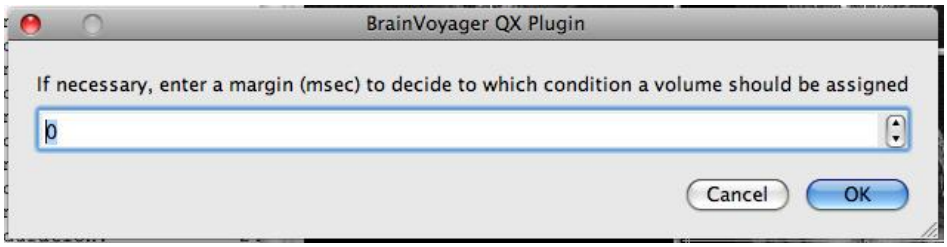
The TR can be entered in the following dialog (is necessary to compute the volume for volume resolution).



Then, the resolution can be chosen, which can be milliseconds or volumes. The volume option is not well tested; it is recommended to select the milliseconds option, because it is always possible in BrainVoyager to change to volumes afterwards.



Thanks to a suggestion of Henk Jansma, it is possible to select a margin that defines to which condition a volume between two conditions should be assigned (in test phase, comments welcome). This dialog only appears when one chooses for volume resolution.



Computation of the interval values in a condition

For millisecond resolution, the interval begin is simply the Ttime in the log file divided by 10. The interval end is the interval begin plus the duration divided by 10.

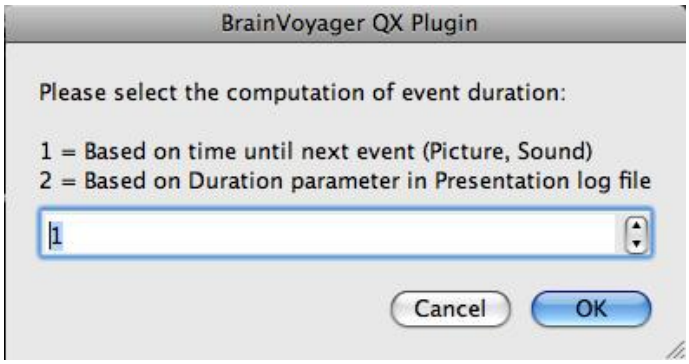
Begin of an interval

For volumes resolution, the begin is assumed to be:

```
mod = (begin % TR);
if (mod > margin) {
      from = int(ceilf(begin/TR));
} else {
      from = int(floorf(begin/TR));
}
```

End of an interval

The end of an interval can be computed in different ways.
Via the first way, the interval is computed by subtracting the start time of a previous picture from the next picture. Via the second method, simply the Duration parameter in the Presentation log file is used (*.log).



When the end interval is based on the Duration parameter in the Presentation log file (*.log), the duration might not be completely until the next Picture event. For example when a Response event is being logged, the Duration parameter indicates the duration from a Picture until response from a subject. See the figure below for a precise indication of the elements in the *.log file.

method 1: start Picture 2 – start Picture 1

```
17  Pulse     30  3540000  101563  NA
17  Pulse     30  3550000  111563  NA
17  Pulse     30  3560000  121563  NA
17  Pulse     30  3570000  131563  NA
17  Pulse     30  3580000  141563  NA
18  Picture   img1 3588479  0    117  116571  118  0    next
18  Pulse     30  3590000  1521NA
18  Pulse     30  3600000  11521   NA
18  Pulse     30  3610000  21521   NA
18  Pulse     30  3620000  31521   NA
18  Pulse     30  3630000  41521   NA
18  Pulse     30  3640000  51521   NA
18  Pulse     30  3650000  61521   NA
18  Pulse     30  3660000  71521   NA
18  Pulse     30  3670000  81521   NA
18  Pulse     30  3680000  91521   NA
18  Pulse     30  3690000  101521  NA
18  Pulse     30  3700000  111521  NA
18  Response5 3704975  116496  1
19  Picture   img2 3705050  0    1    99374   66   0    next
19  Pulse     30  3710000  4950NA
```

method 2: start Picture 1 + duration Picture 1

```
17  Pulse     30  3540000  101563  NA
17  Pulse     30  3550000  111563  NA
17  Pulse     30  3560000  121563  NA
17  Pulse     30  3570000  131563  NA
17  Pulse     30  3580000  141563  NA
18  Picture   img1 3588479  0    117  116571  118  0    next
18  Pulse     30  3590000  1521NA
18  Pulse     30  3600000  11521   NA
18  Pulse     30  3610000  21521   NA
18  Pulse     30  3620000  31521   NA
18  Pulse     30  3630000  41521   NA
18  Pulse     30  3640000  51521   NA
18  Pulse     30  3650000  61521   NA
18  Pulse     30  3660000  71521   NA
18  Pulse     30  3670000  81521   NA
18  Pulse     30  3680000  91521   NA
18  Pulse     30  3690000  101521  NA
18  Pulse     30  3700000  111521  NA
18  Response5 3704975  116496  1
19  Picture   img2 3705050  0    1    99374   66   0    next
19  Pulse     30  3710000  4950NA
```

When using the 'Duration' parameter, the interval end is computed in the following way:

```
int initialto = start + int(floor((duration/10)));
mod = (initialto % TR);
if (mod > margin) {
      to = int(ceilf(initialto/TR));
} else {
      to = int(floorf(initialto/TR));
}
```

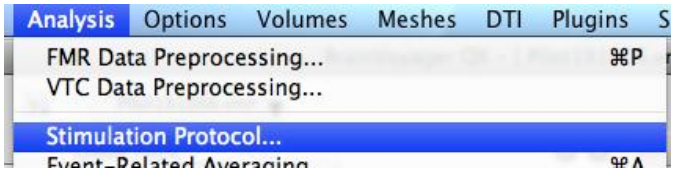
Otherwise the start of the new event is used minus 3 milliseconds, so that the conditions do not overlap.
Comments are welcome.

Generating the color values for each condition

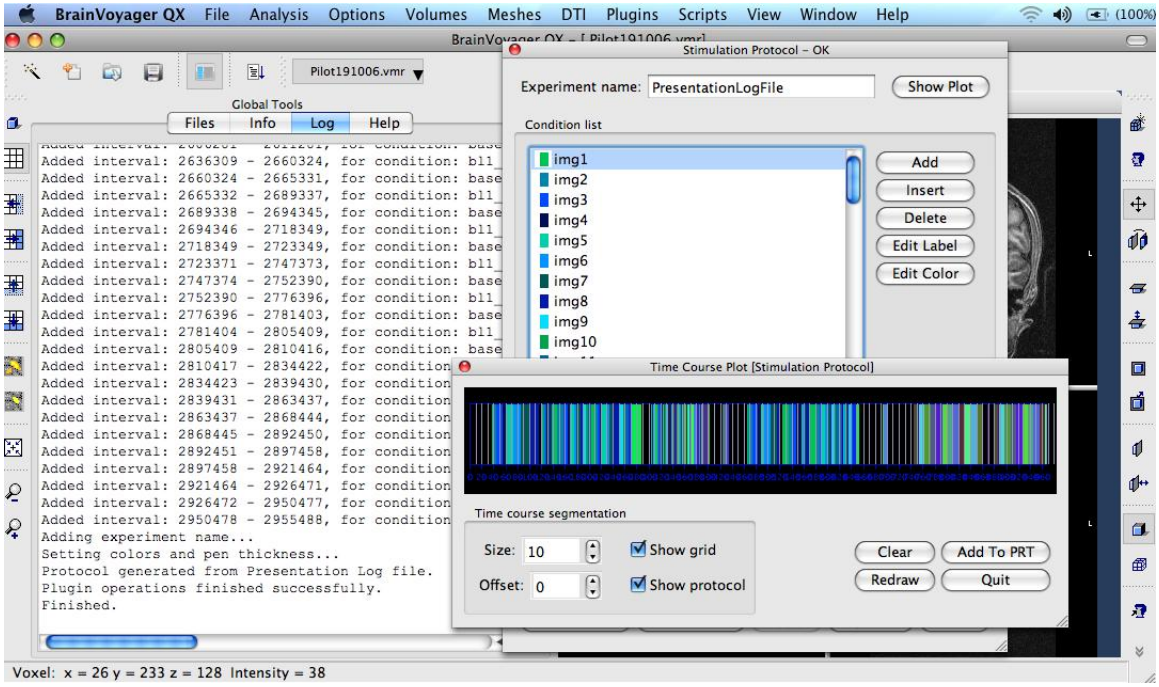
The colors are generated automatically. In future the color generating function might have some randomization feature, so that adjacent conditions can be better distinguished.

Results

The protocol file is written to the same directory as where the Presentation Log file is located. The conversion process is printed to the BrainVoyager QX Log tab.

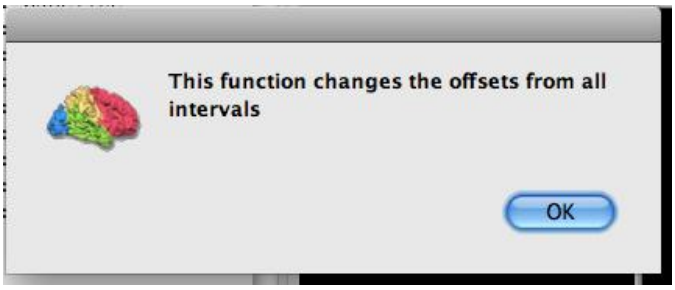


Because the BrainVoyager QX Plugin Access functions are used, the results are directly visible via the “Stimulation Protocol...” option of the BrainVoyager “Analysis” menu.

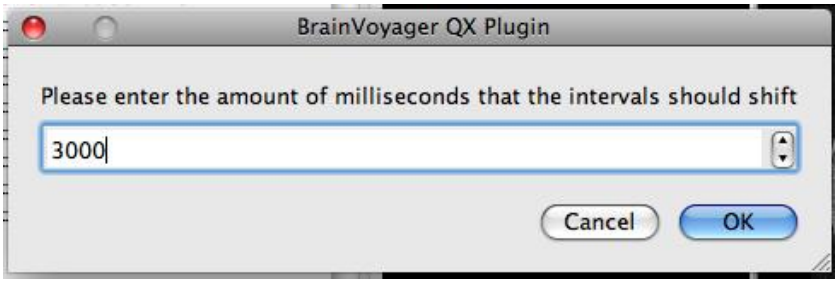

BrainVoyager QX File Analysis Options Volumes Meshes DTI Plugins Scripts View Window Help
Pilot191006.vmr
Global Tools
Files Info Log Help
Added interval: 2636309 - 2660324, for condition: bl1_
Added interval: 2660324 - 2665331, for condition: base
Added interval: 2665332 - 2689337, for condition: bl1_
Added interval: 2689338 - 2694345, for condition: base
Added interval: 2694346 - 2718349, for condition: bl1_
Added interval: 2718349 - 2723349, for condition: base
Added interval: 2723371 - 2747373, for condition: bl1_
Added interval: 2747374 - 2752390, for condition: base
Added interval: 2752390 - 2776396, for condition: bl1_
Added interval: 2776396 - 2781403, for condition: base
Added interval: 2781404 - 2805409, for condition: bl1_
Added interval: 2805409 - 2810416, for condition: base
Added interval: 2810417 - 2834422, for condition
Added interval: 2834423 - 2839430, for condition
Added interval: 2839431 - 2863437, for condition
Added interval: 2863437 - 2868444, for condition
Added interval: 2868445 - 2892450, for condition
Added interval: 2892451 - 2897458, for condition
Added interval: 2897458 - 2921464, for condition
Added interval: 2921464 - 2926471, for condition
Added interval: 2926472 - 2950477, for condition
Added interval: 2950478 - 2955488, for condition
Adding experiment name...
Setting colors and pen thickness...
Protocol generated from Presentation Log file.
Plugin operations finished successfully.
Finished.
Stimulation Protocol - OK
Experiment name: PresentationLogFile
Show Plot
Condition list
img1
img2
img3
img4
img5
img6
img7
img8
img9
img10
Add
Insert
Delete
Edit Label
Edit Color
Time Course Plot [Stimulation Protocol]
Time course segmentation
Size: 10
Offset: 0
Show grid
Show protocol
Clear
Add To PRT
Redraw
Quit
Voxel: x = 26 y = 233 z = 128 Intensity = 38

Option 2: changing the offset

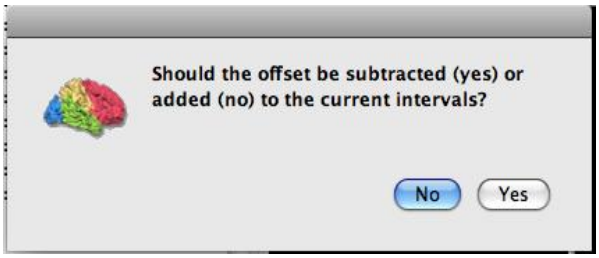
When the start of the protocol should be earlier or later, it is possible to change the offset via this function.
For example, when a few volumes are skipped, one might want to decrease all values by a certain amount. In this case one can subtract this amount via the following procedure.
This amount is then subtracted from all interval beginning and endings.



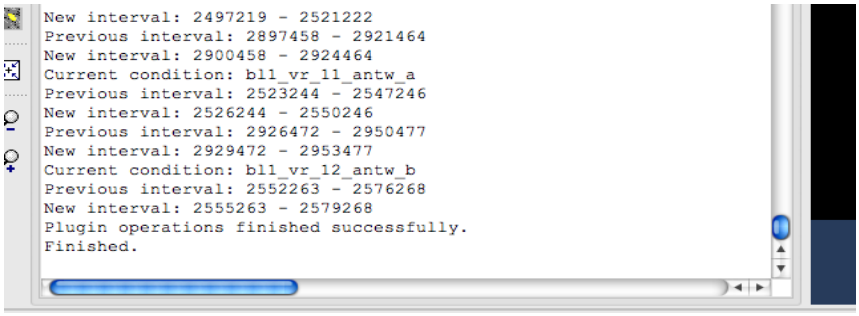
The amount of milliseconds or volumes can be indicated via a dialog (see figure below).



Via the following message box one can indicate whether that amount should be subtracted or added (see figure below).

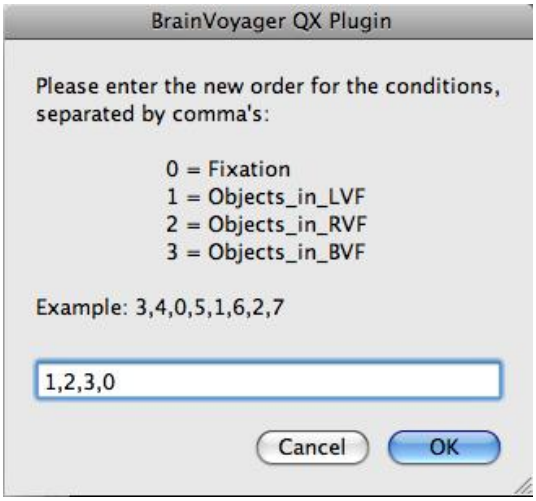


The previous and new interval values are written to the BrainVoyager QX Log tab. The values are changed in the current protocol file.

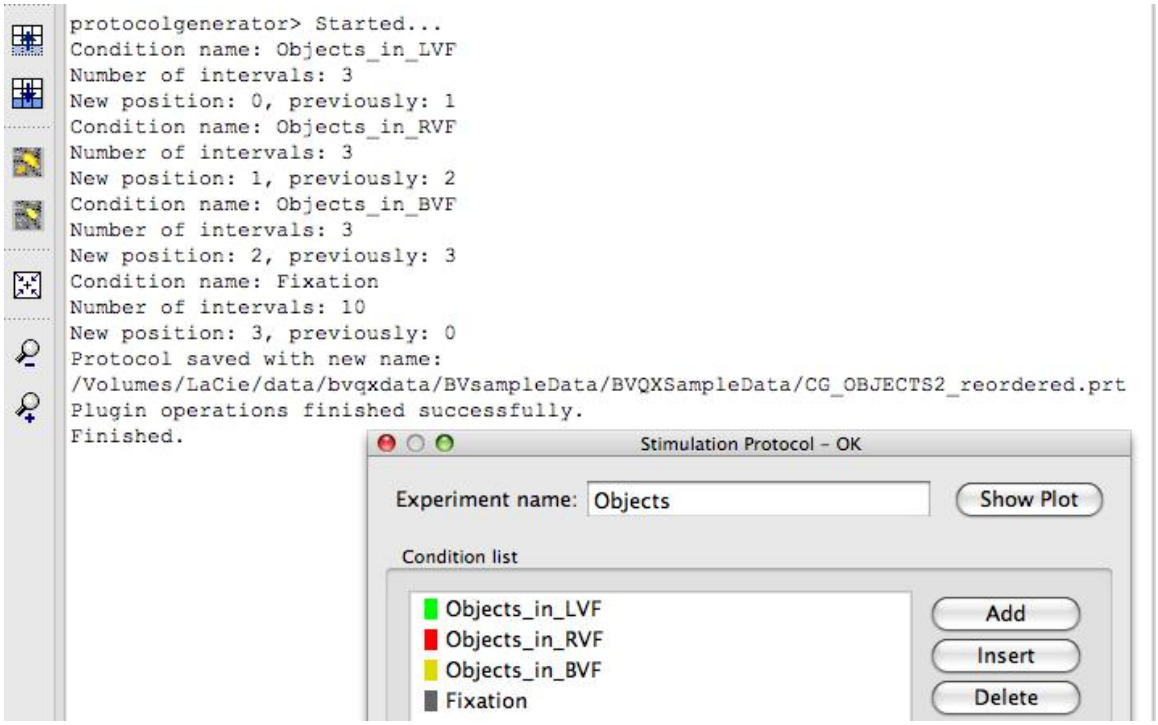

New interval: 2497219 - 2521222
Previous interval: 2897458 - 2921464
New interval: 2900458 - 2924464
Current condition: bll_vr_ll_antw_a
Previous interval: 2523244 - 2547246
New interval: 2526244 - 2550246
Previous interval: 2926472 - 2950477
New interval: 2929472 - 2953477
Current condition: bll_vr_l2_antw_b
Previous interval: 2552263 - 2576268
New interval: 2555263 - 2579268
Plugin operations finished successfully.
Finished.

Option 3: reordering the conditions

When the final goal is to perform a statistical analysis with multiple studies, the order of the conditions needs to be similar in each protocol file, so that the columns (predictors) in the design matrix will correspond in case an automated tool like the designmatrixgenerator plugin is used.



When selecting this option, an input dialog appears which shows the current order of options (see figure above). In the input box one can specify the new order, separated by commas. A stimulation protocol file with the new order of conditions will be saved to disk with the suffix "*_reordered.prt". A report of the process is printed to the BrainVoyager QX Log tab (see figure below).



Appendix

Generating the plugin from source code

Windows XP

For Windows XP, the Visual Studio 2005 project that is included in the source code zip file, can be used to build the protocolgenerator_vxxx.dll.

Mac OS X

For Mac OS X, open the XCode project and click the 'Build' button to generate a protocolgenerator_vxxx.dylib.

Linux

```
##############################################################################
# Makefile for building: protocolgenerator.dylib
##############################################################################

####### Compiler, tools and options

CC       = cc
CXX      = c++
LEX      = flex
YACC     = yacc
CFLAGS   = -pipe -Wall -W -O2 -fPIC -DMAKE_PLUGIN
CXXFLAGS = -pipe -Wall -W -O2 -fPIC -DMAKE_PLUGIN
LEXFLAGS =
YACCFLAGS= -d
INCPATH  = -I.
LINK     = g++
LFLAGS   = -shared -Wl,-fPIC
LIBS     =
AR       = ar cqs
RANLIB   =
TAR      = tar -cf
GZIP     = gzip -9f
COPY     = cp -f
COPY_FILE= cp -f
COPY_DIR = cp -f -r
INSTALL_FILE= $(COPY_FILE)
INSTALL_DIR = $(COPY_DIR)
DEL_FILE = rm -f
SYMLINK  = ln -sf
DEL_DIR  = rmdir
MOVE     = mv -f
CHK_DIR_EXISTS= test -d
MKDIR    = mkdir -p

####### Output directory

OBJECTS_DIR = obj/
```

```
####### Files

HEADERS =     \
        pres2prot.h \
        pres2protfuncs.h \
        PRES_Header.h \
        colortools.h \
        iofuncs.h \
        BVQXPluginInterface.h \
        global.h \
        Plugin_CMP_Header.h \
        Plugin_ICA_Header.h \
        Plugin_MTC_Header.h \
        Plugin_POI_Header.h \
        Plugin_PRT_Header.h \
        Plugin_SMP_Header.h \
        Plugin_SRF_Header.h \
        Plugin_SSM_Header.h \
        Plugin_VMP_Header.h \
        Plugin_VOI_Header.h \
        Plugin_VTC_Header.h


SOURCES = \
        pres2prot.cpp \
        BVQXPluginInterface.cpp \
        pres2protfuncs.cpp \
        colortools.cpp \
        iofuncs.cpp


OBJECTS = \
        obj/pres2prot.o \
        obj/BVQXPluginInterface.o \
        obj/colortools.o \
        obj/pres2protfuncs.o \
        obj/iofuncs.o \


DESTDIR  =
TARGET   = protocolgenerator.so
TARGETD   = protocolgenerator.so

first: all

####### Implicit rules

.SUFFIXES: .c .o .cpp .cc .cxx .C

.cpp.o:
        $(CXX) -c $(CXXFLAGS) $(INCPATH) -o $@ $<

.cc.o:
        $(CXX) -c $(CXXFLAGS) $(INCPATH) -o $@ $<

.cxx.o:
```

```
	$(CXX) -c $(CXXFLAGS) $(INCPATH) -o $@ $<

.C.o:
	$(CXX) -c $(CXXFLAGS) $(INCPATH) -o $@ $<

.c.o:
	$(CC) -c $(CFLAGS) $(INCPATH) -o $@ $<

####### Build rules

all: Makefile  $(TARGET)

$(TARGET): $(OBJECTS)
	-$(DEL_FILE) $(TARGET)
	$(LINK) $(LFLAGS) -o $(TARGET) $(OBJECTS) $(LIBS)

clean:
	-$(DEL_FILE) $(OBJECTS)


####### Sub-libraries

distclean: clean
	-$(DEL_FILE) $(TARGET) $(TARGET)


FORCE:

####### Compile

obj/pres2prot.o: \
	pres2prot.cpp \
	pres2prot.h\
	BVQXPluginInterface.h \
	colortools.h \
	iofuncs.h \
	pres2protfuncs.h \
	PRES_Header.h \
	Plugin_CMP_Header.h \
	Plugin_ICA_Header.h \
	Plugin_MTC_Header.h \
	Plugin_POI_Header.h \
	Plugin_PRT_Header.h \
	Plugin_SMP_Header.h \
	Plugin_SRF_Header.h \
	Plugin_SSM_Header.h \
	Plugin_VMP_Header.h \
	Plugin_VOI_Header.h \
	Plugin_VTC_Header.h
	$(CXX) -c $(CXXFLAGS) $(INCPATH) -o obj/pres2prot.o pres2prot.cpp

obj/BVQXPluginInterface.o: BVQXPluginInterface.h BVQXPluginInterface.cpp
	$(CXX) -c $(CXXFLAGS) $(INCPATH) -o obj/BVQXPluginInterface.o
BVQXPluginInterface.cpp

obj/colortools.o: \
```

```
	colortools.h colortools.cpp\
	BVQXPluginInterface.h \
	Plugin_SRF_Header.h
	$(CXX) -c $(CXXFLAGS) $(INCPATH) -o obj/colortools.o colortools.cpp

obj/pres2protfuncs.o: \
	pres2protfuncs.cpp\
	pres2protfuncs.h \
	iofuncs.h \
	BVQXPluginInterface.h \
	PRES_Header.h
	$(CXX) -c $(CXXFLAGS) $(INCPATH) -o obj/pres2protfuncs.o pres2protfuncs.cpp


### Protocol generator
```

Generating the API documentation

The functions in the header files have been documented using Apple's HeaderDoc markup language. It is possible to generate your own Application Programmer's Interface (API) documentation in the way described below.

Run `headerdoc2html` and `gatherheaderdoc` for the directory where the header files reside:

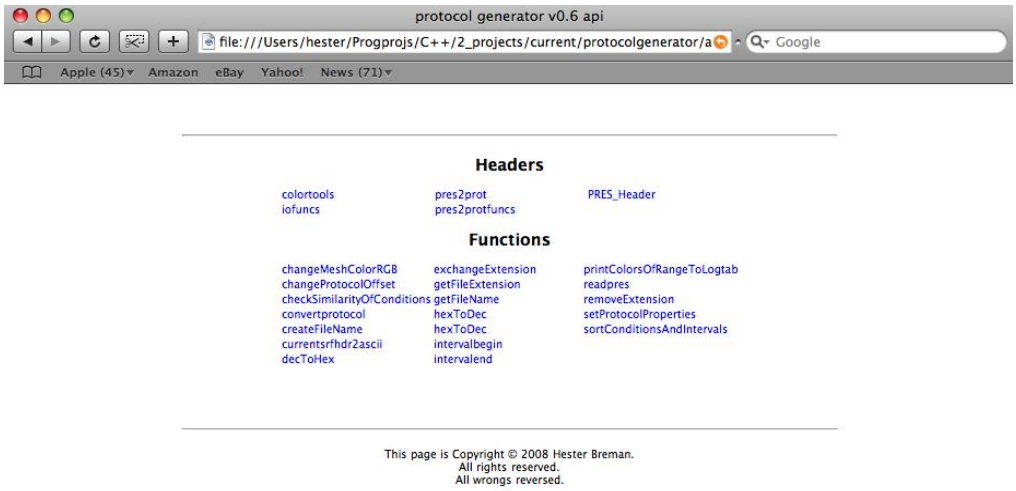
```
headerdoc2html /Users/hester/Progprojs/C++/2_projects/current/protocolgenerator/api/
gatherheaderdoc /Users/hester/Progprojs/C++/2_projects/current/protocolgenerator/api/
```

It is also possible to first create a batch file, so that the commands do not need to be retyped each time:

1. add `#!/bin/bash` to an empty text file and add the lines above, save.
2. in the Terminal, type ~ `chmod a+x <name batch file>`
3. to execute the commands in the file, type ~ `./<name batch file>`

For reuse of the batch file, just run (3) in the Terminal (shell).

This will generate the modest Application Programmer's Interface documentation:



History

Changes in the plugin

version 0.8

August, 2008: In this version a third option is added, to reorder the conditions. Also, when shifting the protocol timing, the file is now saved with a suffix: “_min/plus<ms>.prt”. Changed text color and time course color to white.

version 0.7

July, 2008: This version of the plugin can compute the end of an interval in two ways.

version 0.6

April, 2008: In this version a function that recognizes intervals of the same condition. The function documentation for generating an API has been changed to Apple's HeaderDoc markup.

Changes in this document

version 0.3

July, 2008: The document has been reorganized, the preparation part moved to the installation section. Also, the different ways to compute the end of an interval in a condition has been described.

version 0.2

March, 2008: The document has been reorganized, the compilation part has moved to the appendix.

version 0.1

November 2006: Creation of document.