RELACS —
a modular software platform
for closed-loop experiments

Jan Benda
Department Biologie II
Ludwig-Maximilian
Universität München
“Traditional” Experiments

1. A set of stimuli and a more or less fixed experimental protocol are prepared
“Traditional” Experiments

1. A set of stimuli and a more or less fixed experimental protocol are prepared

2. The recordings are done on a few cells (> one week of experimental work)
   - the raw voltage trace is the only feedback
   - changes during the running experiment often involve manual manipulations
     ⇒ precious recording time is wasted
“Traditional” Experiments

1. A set of stimuli and a more or less fixed experimental protocol are prepared

2. The recordings are done on a few cells (> one week of experimental work)
   - the raw voltage trace is the only feedback
   - changes during the running experiment often involve manual manipulations
     ⇒ precious recording time is wasted

3. The data are analyzed offline
“Traditional” Experiments

1. A set of stimuli and a more or less fixed experimental protocol are prepared
2. The recordings are done on a few cells (> one week of experimental work)
   - the raw voltage trace is the only feedback
   - changes during the running experiment often involve manual manipulations
     ⇒ precious recording time is wasted
3. The data are analyzed offline
4. The stimuli and the protocol are modified
“Traditional” Experiments

1. A set of stimuli and a more or less fixed experimental protocol are prepared

2. The recordings are done on a few cells (> one week of experimental work)
   - the raw voltage trace is the only feedback
   - changes during the running experiment often involve manual manipulations
     ⇒ precious recording time is wasted

3. The data are analyzed offline

4. The stimuli and the protocol are modified

5. A new set of recordings is made
1. Present a stimulus
2. Record the response
3. Immediately analyze and visualize the data
4. Generate the next stimulus
**Closed-Loop Experiments**

*RELACS* is designed as a framework for closed-loop experiments that

- considerably speed up this traditional approach
- offer novel experimental possibilities
Simple Closed-Loop Experiments

- Online visualization of processed data:
  - General infos, e.g. quality of spike detection, sensitivity of the cell, temperature, condition of animal, ...
  - Specific results, e.g. spike raster, firing rates, spike-triggered averages, ...

⇒ Speeds up manual closed-loop
Simple Closed-Loop Experiments

- Online visualization of processed data:
  - General infos, e.g. quality of spike detection, sensitivity of the cell, temperature, condition of animal, ...
  - Specific results, e.g. spike raster, firing rates, spike-triggered averages, ...
  ⇒ Speeds up manual closed-loop

- Set stimuli relative to the neuron’s dynamic range
- Automatically control motorized electrodes
- ...
Advanced Closed-Loop Experiments

New experimental designs are possible:

- Optimal search for a neuron’s receptive field.
- Search for stimuli that drive a neuron in an "optimal" way.
- Find set’s of stimulus parameter that result in the same response (iso-response method).
- ...

Example: Optimal Stimulus Ensembles

Closed-Loop Experiments

In RELACS the closed-loop cycle can be freely programmed as a C++ plugin ("experimental protocol").

The experimental-protocol plugins

- take recorded and pre-analyzed data
- perform analysis & display results
- generate next stimulus
Modular Design

RELACS core with flexible C++ Plugins for

- hardware abstraction
- data pre-processing (filter, spike detectors)
- passive and active control
- model
Screen Shot

RELACS - Relaxed Electrophysiological data Acquisition, Control, and Stimulation: Version 0.9.6 (11/24/06)

Spikes-1
- Minimum threshold: 14.5 mV
- Threshold: 35.2 mV
- Ratio threshold / size: 40%
- Spike size: 95.8 mV

Info
- Best side: left speaker
- Best frequency: 4.0 kHz
- Best threshold: 42.5 dB SPL

Stimulus
- Mean firing rate = 101 Hz
- AM: OU-noise20Hz1228223405, Intensity: 55.0 dB SPL, Amplitude: 6.0 dB, Loop 19 of 20
**Hardware Independent Protocols**

RELACS integrates all hardware components.

*Experimental protocols* for RELACS

- are implemented independently of specific hardware
- can be used on all the different experimental setups in your lab without any modifications
- can be shared with other labs
```cpp
int Example::main( void ) {
    // some initialization ...
    OutData signal;
    signal.setTrace( "LeftSpeaker" );
    signal.sineWave( frequency, duration, amplitude );
    SampleDataD rate( 0.0, duration, 0.001 );
    for ( int counter = 0; counter < Repeats; counter++ ) {
        write( signal );
        sleep( duration + pause );
        EventData spikes( events( SpikeEvents[0] ),
                         events( SpikeEvents[0] ).signalTime(),
                         events( SpikeEvents[0] ).signalTime() + duration );
        double meanrate = spikes.rate( 0.3*duration, duration );
        spikes.addRate( rate, counter, GaussKernel( sigma ) );
        P.clear();
        P.plot( rate, 1000.0, Plot::Yellow, 2, Plot::Solid );
        P.draw();
        if ( meanrate < targetrate ) {
            amplitude *= 2.0;
            signal.sineWave( frequency, duration, amplitude );
        }
    }
}
```
Why C++

- well structured (object oriented)
- platform independent
- efficient and controllable memory usage
- very fast
Data structures (classes, container):

- **Array** — Basic 1-D vector
- **SampleData** — 1-D data vector with regularly sampled time axis
- **Map** — Sequence of $x|y$ data pairs

Algorithms:

- basic statistics (moments, quartiles, histogram)
- power spectra, coherence, transfer function
- linear fits
- non-linear fits (Simplex, Levenberg-Marquardt)
C++ Library for Data Analysis

Data structures (classes, container):

- **EventData** — Spikes and other point process data
- **EventList** — Multi-trial spike trains

Algorithms:

- firing rates (mean, PSTH binned/kernel, 1/ISI)
- CV, Fano factor, ISI correlation
- vector strength, reliability, jitter
- mutual information (lower and upper bound)
Free and Open Source Software

RELACS is open source and free software distributed under the GNU General Public License (GPL).

- No hassle with licenses of commercial software.
- Add whatever new feature you need directly to the program.
- Share the program and your specific experimental protocols with your collaborators.
- Know what the data-analysis algorithms are doing!
Meta-data

Meta-data (“data about data”) describe the context of the raw data.

Meta-data are important for:

- Data management
- Data retrieving
- Data analysis
- Data sharing
Meta-Data Acquisition by RELACS
Meta-Data Acquisition by RELACS
Meta-Data Acquisition by RELACS

RELACS records many meta-data:

- General infos about the experiment (from the dialog)
- Main characteristics of the recorded cell
- All RELACS-controlled hardware settings (e.g. sampling rate)
- All settings and version numbers of the experimental protocols
The Meta-Data Problem

Name-value (+unit) pairs for:

- Stimuli
- Experimental settings
- Cell, preparation, experimental subject
- Hardware properties
- Analysis parameter
- etc.
The Meta-Data Problem

Name-value (+unit) pairs for:

- Stimuli
- Experimental settings
- Cell, preparation, experimental subject
- Hardware properties
- Analysis parameter
- etc.

But:

- What name to choose?
- What does it mean?
- How to share meta-data?
A Proposal

We need some kind of standard!

- flexible and extensible
- downloadable and parseable for seamless integration in software
A Proposal

We need some kind of standard!

- flexible and extensible
- downloadable and parseable for seamless integration in software

An XML schema explaining

- definitions
- dependencies
- meaning of names ("tags").
A Proposal

We need some kind of standard!

- flexible and extensible
- downloadable and parseable for seamless integration in software

An XML schema explaining

- definitions
- dependencies
- meaning of names ("tags").

In collaboration with G-node and LabLog by Jan Grewe (lablog.sourceforge.net)
Closed-loop experiments
Hardware independent
Data analysis libraries
Meta-data storage
Simulation mode
Dynamic clamp
Open source, GPL, Linux
~ 120 000 lines of C++ code