COSIVINA
Compose, Simulate, and Visualize
Neurodynamic Architectures

Summer School Neural Dynamics for Cognitive Robotics
September 2013

Sebastian Schneegans
sebastian.schneegans@ini.rub.de
Motivation

• quickly create and simulate DNF architectures
• create online visualizations and interactive controls of model parameters without hassle
• standardize implementations for different architectures and usages

➢ class library for Matlab
Examples
Structure for DNF Architectures

- Feature space $x$
- Activation $u(x)$
- Surround inhibition
- Local self-excitation
- External inputs
- Feature space $x$
Structure for DNF Architectures

- MexicanHatKernel1D
- NeuralField
  - GaussStimulus1D
  - GaussStimulus1D
Simulator class manages elements, provides functions to initialize and simulate whole architecture.

Elements implemented as handle classes, can have handles to other elements that provide input to them.
NeuralField < Element

properties

label = 'field u'

% meta-data
parameters
components

% inputs
nInputs
inputElements % handles to other elements
inputComponents

% parameters
size = [1, 100]
tau = 20
...

% components
activation
output
Connections between Elements

Neural Field

... inputElements ...

inputComponents ...

GaussStimulus1D

... % components output ...

Element Classes

NeuralField < Element

methods

NeuralField(...) % constructor, creates element object with given parameters

init() % initializes the element

step(time, deltaT) % updates element for one Euler step

close() % for classes that connect to hardware

... % some auxiliary functions to manage inputs etc.
Elements implemented as handle classes, can have handles to other elements that provide input to them.

Simulator class manages elements, provides functions to initialize and simulate whole architecture.
Simulator

% general parameters
t
tZero
deltaT

methods

init() % initializes all element

step() % performs one Euler step

close() % closes all elements

run(tMax, ...) % runs simulation until specified time

... % a lot of methods to access elements and parameters
Graphical User Interfaces
Connection between GUI and Simulator

StandardGUI
- Control
- Visualization

Simulator
- Element
  - parameter1
  - parameter2
  - component
- Element
  - parameter1
  - parameter2
  - component
• “offline” mode without the GUI: run simulator (perhaps for many trials), plot or analyze resulting activities manually

• “offline” mode with GUI: run and view a simulation with specified time course of stimuli etc.

• “online” mode: GUI controls the simulation, runs continuously, change parameters through control elements

• and change back and forth between these modes of operation...
Resources

- get the cosivina toolbox at https://bitbucket.org/sschneegans/cosivina/
- documentation available as pdf or as wiki on bitbucket (the wiki is a bit more up to date)
- Matlab help for all elements, controls and visualization