**Technical Document**

**Loading the software modules**

**Versions to load:**

Nengo: 2.2.0

Nengo\_GUI: 0.4.5

Nengo\_Spinnaker: 0.4.0

Rig: 2.4.1

**OS :**

Linux (Ubuntu 18.04)

**Requirements:**

Python 2.7 minimum

Pip

-For Python > 3, make sure you have pip3

*Note:* If you have both python versions, it is not a problem.

Try to load the above softwares with Python 2, rather than 3. It makes interfacing with ROS easier.

**Two ways to load the softwares:**

**One:**

- Download the open access tar files of the required versions from PyPI.

- Extract the tar.

- Run the *setup.py* as below in the terminal:

$**sudo** python setup.py install

$**sudo** python setup.py build

Both the above commands should run without any errors.

*Note:* sudo (or super user) is important, since without a sudo command the paths to Nengo, Nengo\_Spinnaker and Nengo\_GUI would not be set correctly.

GitHub of the above softwares contain the latest code updates and point to the latest versions.

However, they can be unstable and may not work.

**Two:**

- Do a pip install of the required software versions as below:

$**sudo** pip install nengo==2.2.0

$**sudo** pip install nengo\_gui==0.4.5

$**sudo** pip install nengo\_spinnaker==0.4.0

-Rig gets automatically installed with nengo\_spinnaker. If not, try:

$**sudo** pip install rig==2.4.1

**Sanity Check:**

- Go to */usr/local/lib/python2.7/dist-packages*

- Do a

$ls -lta

Should see something similar to this:

*-rwxrwxrwx 1 root staff 46 Oct 29 12:46 nengo-spinnaker.egg-link*

*drwxrwsrwx 2 root staff 4096 Nov 15 13:48 nengo-2.2.0.dist-info*

*drwxrwsrwx 7 root staff 4096 Nov 15 13:48 nengo*

*drwxr-sr-x 4 root staff 4096 Nov 15 17:23 nengo\_gui-0.4.5.dev0-py2.7.egg*

*-rwxrwxrwx 1 root staff 33 Nov 15 17:25 easy-install.pth*

*drwxr-sr-x 9 root staff 4096 Nov 15 17:26 rig*

*drwxr-sr-x 2 root staff 4096 Nov 15 17:26 rig-2.4.1.dist-info*

*drwxr-sr-x 2 root staff 4096 Nov 15 17:26 nengo\_spinnaker-0.4.0.dist-info*

*drwxr-sr-x 10 root staff 4096 Nov 15 17:26 nengo\_spinnaker*

*drwxr-sr-x 5 root staff 4096 Nov 25 13:30 nengo\_extras*

*drwxr-sr-x 2 root staff 4096 Nov 25 13:30 nengo\_extras-0.4.0.dist-info*

*Note:* sudo (or super user) is important, since without a sudo command the paths to Nengo, Nengo\_Spinnaker and Nengo\_GUI would not be set correctly.

**Uninstalling any of the software versions**

- Make sure the package folder and the *.egg* and/or *.dist-info* files get removed from path

*/usr/local/lib/python2.7/dist-packages*

- If nengo\_spinnaker is removed, rig will be removed.

- Running a *setup.py* file for nengo\_spinnaker from the github, will not load the required rig version with it. Rig will need to be installed separately.

- Make sure Rig is present in *path /usr/local/lib/python2.7/dist-packages*, otherwise nengo will not be able to read the connection to the spinnaker board.

It will throw a boot error originating from the *simulator.py* file in nengo.

**Configuring nengo\_spinnaker:**

- Run

$nengo\_spinnaker\_setup

- Switch your Spinn-5 board on.

- Follow the terminal instructions.

- Click on the option of automatically detect.

The .conf file gets generated.

- Make sure the .conf file contains these:

*hostname: 192.168.240.1*

*width: 8*

*height: 8*

- If you want to override the file, use $nengo\_spinnaker\_setup - - force

- Linux machine ethernet settings:

- Go to Settings→ Network → IPv4 → IPv4 method → Manual

IP address : 192.168.0.250

Subnet Mask: 255.255.0.0

Gateway: 0.0.0.0 (optional, generally left blank in settings)

**Testing if Nengo works:**

- Type

$nengo

- The gui should open a browser tab and you should be able to see a default network model.

- Click on the Play button on the bottom right.

The network should simulate and start running.

- Right click on the neuron clusters or node to get the spikes, voltages or X-Y values.

- Re-click on the Play button to restart the simulation again.

**Testing if Nengo\_Spinnaker works:**

**One:**

- Take a model example from the nengo\_spinnaker git.

- Copy the code into a .py file.

- Switch on your Spinn-5 board.

- Type

$nengo filename.py -b nengo\_spinnaker

- The gui should open with the network layout. Click on play.

See if the lights in the Spinn-5 board start blinking.

- To test if its actually running on the Spinnaker board, turn off power of the board.

The simulation in the gui should stop with a connection error.

**Two:**

- Type

$nengo

- In the gui, click on the wrench on the top left corner.

- Go to Configure Settings.

- Change the backend from nengo to nengo\_spinnaker.

The simulation should now use nengo\_spinnaker and will require a spinnaker board to be connected.

**Issues and how to fix them:**

- These versions might not start working and give errors like:

*ensemble.sample() not found*

- The way to fix it is to open the github of the nengo\_spinnaker.

This git will contain the latest code.

- Check the error the from the Nengo\_gui, spot the file and the line.

- Go to  */usr/local/lib/python2.7/dist-packages*

- Go to nengo\_spinnaker folder

- Give write permissions to all the files.

$sudo chmod 777 \*

- Find the file with the error.

- Do a

$gedit filename.py

- Go to the error line inside the file.

- Open the same file and try to find same error line in the github file. The error line number will not be same in the git file due to the changes in the code. Easiest way to find the error line is to look for the function which throws the error. It should be mentioned in the Nengo\_gui error message.

- Compare the git file code with the open filename.py code.

If you find any differences, comment out the code lines in filename.py which are different from the git version and copy the git version of the code in the same place in filename.py.

So comment the filename.py lines with error/different, paste the git file version of the code there.

For instance, in the latest code version, ensemble.sample is no longer used.

Instead get\_samples() is used.

So replace ensemble.sample lines with get\_samples() lines found in the git, if you face an error with it.

- This technique needs to be followed in case you face any other errors as well.

- Make sure you do not replace the entire installed code in the file with the git version of the same.

This will lead to more errors. So only copy and replace the portions which log an error.

**Interfacing with ROS**

- Include the following lines in your bashrc file

*source /opt/ros/melodic/setup.bash*

*source ~/Workspace/whiskeye\_ws/devel/setup.bash*

*source ~/Workspace/whiskeye\_ws/devel/setup.sh*

*export GAZEBO\_MODEL\_PATH=$HOME/Workspace/whiskeye\_ws/src/whiskeye\_description/:$ {GAZEBO\_MODEL\_PATH}*

*export GAZEBO\_RESOURCE\_PATH=$HOME/Workspace/whiskeye\_ws/src/whiskeye\_gazebo/ worlds/:${GAZEBO\_RESOURCE\_PATH}*

*export*

*GAZEBO\_PLUGIN\_PATH=${GAZEBO\_PLUGIN\_PATH}:/home/samia/Workspace/ whiskeye\_ws/src/whiskeye\_description/whiskeye\_robot*

Workspace should be your Workspace folder name set up while creating a workspace for the vitaslam or openratslam codes

- Source the ~/.bashrc file after adding the above lines.

**Spiking RatSLAM**

- Download spiking ratslam from <https://github.com/bjkomer/spiking-ratslam>

You may do a download and extract or a git clone.

- If cloning, do it directly inside /home/samia/Workspace/whiskeye\_ws/src

If not, extract git and copy the folder to the above path.

- The important files from this repo are:

ratslam/posecell\_network.py

ratslam/nengo\_posecell\_network

ratslam/nengo\_posecells\_ci.py

ratslam/spinnaker\_nengo\_posecells\_ci.py

File ratslam/nengo\_posecells\_ci.py cannot be deployed on spinnaker.

File ratslam/spinnaker\_nengo\_posecells\_ci.py is meant for the spinnaker deployment.

**Running the nengo Posecell models**

**Link it to ViTaSLAM**

- Go to /home/samia/Workspace/whiskeye\_ws/src/vitaslam/launch

- Open and edit whiskeye\_slam.launch. Save the changes.

- Comment this line out like

<!--<node name="RatSLAMPoseCells" pkg="vitaslam" type="vitaslam\_pc" args="$(find vitaslam)/config/config\_whiskeye.txt.in" cwd="node" required="true" output="screen">

</node>-- >

**Changes in the Spiking RatSLAM code:**

- In the posecell\_network.py file, make sure you are listening to CombinedTemplateMessage message from ROS and publishing to the TopologicalAction message.

- This line should be there along with other imports.

from vitaslam.msg import CombinedTemplateMessage, TopologicalAction

**Run the nengo simulation (nengo as backend):**

- Download a bag file from the shared files.

- Do a ROS launch

*roslaunch master\_node bagfile.launch bagfile\_path:="path/to/your/bagfile.bag"*

This should start the roscore automatically and play the bag file.

You should see View cell and Experience Map windows open.

The view cell template window will display whiskers and images. However, the experience map will only build once nengo simulation is started.

- Open a separate terminal and go to the spiking ratslam directory and inside ratslam.

- Type

$nengo nengo\_posecells\_ci.py

- Click on the play button.

It should build and run.

You should be able to see the spikes from the posecell ensembles in the network when you right click on the ensembles in the gui.

**Running on spinnaker (nengo\_spinnaker as backend):**

- Download a bag file from the shared files.

- Do a ROS launch

*roslaunch master\_node bagfile.launch bagfile\_path:="path/to/your/bagfile.bag"*

This should start the roscore automatically and play the bag file.

You should see View cell and Experience Map windows open.

The view cell template window will display whiskers and images. However, the experience map will only build once nengo simulation is started.

- Open a separate terminal and go to the spiking ratslam directory and inside ratslam

- Type

$nengo spinnaker\_nengo\_posecells\_ci.py -b nengo\_spinnaker

- Click on the play button.

It should build and run.

As of now, spikes cannot be visualized in the gui due to the core failure issues that we get.

But the spinnaker cores running with code should blink and the experience map should start building.

**Link it to OpenRatSLAM**

- When you want to work with a different RatSLAM code in ROS, make sure you change these lines in ~/.bashrc to point to the new ratslam code directory now.

*source ~/Workspace/whiskeye\_ws/devel/setup.bash*

*source ~/Workspace/whiskeye\_ws/devel/setup.sh*

- Source the ~/.bashrc after the changes.

- Go to /home/samia/Workspace/src/ratslam\_ros/launch

- Open and edit irataus.launch. Save the changes.

- Comment out the line like

*<!--<node name="RatSLAMPoseCells" pkg="ratslam\_ros" type="ratslam\_pc" args="$(find ratslam\_ros)/config/config\_irataus.txt \_image\_transport:=compressed" cwd="node" required="true" />-- >*

**Changes in the Spiking RatSLAM code:**

- In the posecell\_network.py file, make sure you are listening to CombinedTemplateMessage message from ROS and publishing to the TopologicalAction message.

- This line should be there along with other imports.

from ratslam\_ros.msg import ViewTemplate, TopologicalAction

**Run the nengo simulation (nengo as backend):**

Download a bag file from the shared files.

- Do a ROS launch

*roslaunch ratslam\_ros irataus.launch*

- Play the bag file.

Go to the directory where the bagfile is downloaded in terminal.

*rosbag play bagfile.bag*

This should start the roscore automatically and play the bag file.

You should see View cell and Experience Map windows open.

The view cell template window will display whiskers and images. However, the experience map will only build once nengo simulation is started.

- Open a separate terminal and go to the spiking ratslam directory and inside ratslam.

- Type

$nengo nengo\_posecells\_ci.py

- Click on the play button.

It should build and run.

You should be able to see the spikes from the posecell ensembles in the network when you right click on the ensembles in the gui.

**Running on spinnaker (nengo\_spinnaker as backend):**

- Download a bag file from the shared files.

- Do a ROS launch

*roslaunch ratslam\_ros irataus.launch*

- Play the bag file.

Go to the directory where the bagfile is downloaded in terminal.

*rosbag play bagfile.bag*

This should start the roscore automatically and play the bag file.

You should see View cell and Experience Map windows open.

The view cell template window will display whiskers and images. However, the experience map will only build once nengo simulation is started.

- Open a separate terminal and go to the spiking ratslam directory and inside ratslam.

- Type

$nengo spinnaker\_nengo\_posecells\_ci.py -b nengo\_spinnaker

- Click on the play button.

It should build and run.

As of now, spikes cannot be visualized in the gui due to the core failure issues that we get.But the spinnaker cores running with code should blink and the experience map should start building.

You can store the spikes in a file with nengo.Probes and then later use matplotlib or seaborn to plot raster plots or convert them into firing rates.

**Gazebo**

- See if Gazebo is installed.

- Make sure you have these files:

*"/home/samia/ratslam/whiskeye\_ws/src/whiskeye\_description/whiskeye\_robot/ 2019\_08\_12\_CBS\_libwhiskeye\_gazebo.so  
 /home/samia/ratslam/whiskeye\_ws/src/whiskeye\_description/whiskeye\_robot/ 2019\_08\_22\_libwhiskeye\_gazebo.so"*

- Rename this file  *2019\_08\_22\_libwhiskeye\_gazebo.so* to *libwhiskeye\_gazebo.so*

- In your ~/.bashrc file , make you you have this line

*source ~/Workspace/whiskeye\_ws/devel/setup.sh*

**Logging data from Gazebo Simulation:**

- Go to *datalogger.py*inside  */home/samia/ratslam/whiskeye\_ws/src/datalogger/src*

- Change line 41 to point your directory path:

*dataset\_location ="/home/samia/ratslam/whiskeye\_ws/src/datalogger/dataset"*

- Create a folder *dataset*inside

***/****home/samia/ratslam/whiskeye\_ws/src/datalogger/*

- Create a folder*tactile*inside **#**

***/****home/samia/ratslam/whiskeye\_ws/src/datalogger/dataset/tactile*

- Give write permissions to these folders from the terminal.

- Go to */home/samia/ratslam/whiskeye\_ws/src/master\_node/launch*

- Make sure this line is uncommented:

*<!-- launch vitaslam -->*

*<include file="$(find vitaslam)/launch/whiskeye\_slam.launch"/>*

*</launch>*

- Make sure only one Launch Gazebo is uncommented. More than one uncommented line inside ‘Launch Gazebo’ heading will raise an error.

- Do a

*roslaunch master\_node master.launch log:=true*

- Go to */home/samia/ratslam/whiskeye\_ws/src/experiments/CBS2019/metrics*

- Do a

*python plot\_paths.py --logs="../cbs19\_logs/tactile\_2.csv" –save=False*

This is display the Energy and Experience Map metric values for the simulation.

If you need to compare metrics of Posecell in ROS and Nengo/Nengo-Spinnaker, you only need to uncomment/comment the Posecell network line in whiskeye\_slam.launch and run the simulations.