Concordia University

Connecting and Controlling the Clearpath Husky and the Argo J5

Jean-Sébastien Fiset

October 1, 2019

1. Clearpath Husky A200

1.1 Technical specifications

The Husky A200 is mounted with a PC and a wireless router inside the user storage area. There is five standard aluminum extrusions (of type HFS5-2020) on top of the robot to mount other devices. As it can be seen in the picture below, the VN-100S Rugged IMU was fixed directly on one the rails.

The battery is placed at the back of the rover and can be removed to be charged. As mentioned in the user manual, it takes approximately 10 hours to fully charge the battery. Even after 10 hours, you will see in the */status* ROS topic that it is not at 100% and the four LEDs on the Husky showing the battery status will drop quickly. This is totally normal since we do not have the Lithium Ion upgrade. For more details, refer to the user manual and datasheet.

1.2 How to control

To power on the Husky, press the power button at the back and as well as the computer power button (shown in Fig. 1.1).

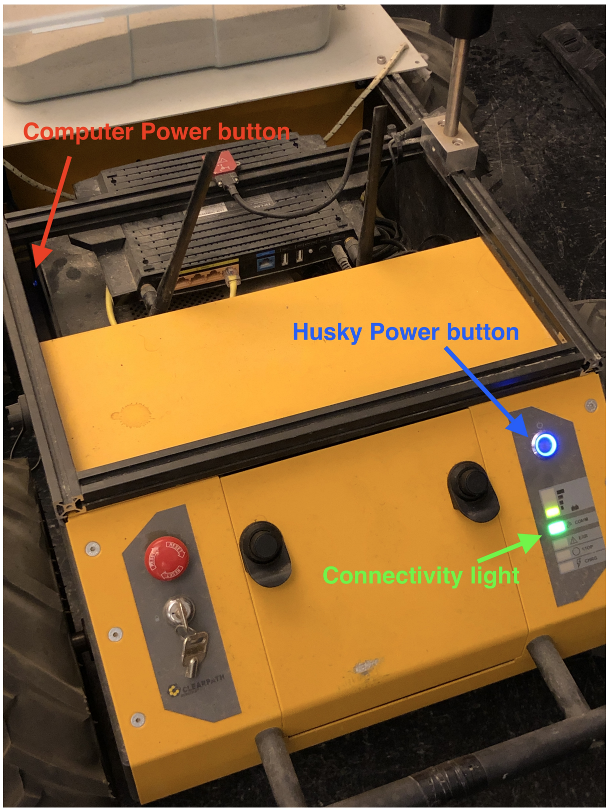


Figure 1.1: Clearpath Husky showing the computer and rover power On buttons.

1.2.1 Manually

To operate the robot manually with the controller, simply press A (for normal speed) or X (for fast speed) and control using the left joystick.

1.2.2 Using ROS

You can also control the robot by connecting to the Onboard PC and running a script with the desired inputs.

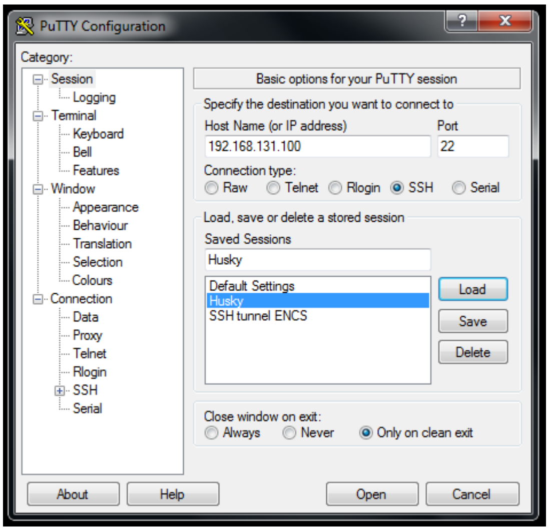
1.2.2.1 Login

When the connectivity light at the back is green, you may connect to either the *CONU01_2G* or the *CONU01_5G* network. For both of them, the password is **clearpath**.

The next step is to connect to the onboard computer, as explained below, for Windows and Mac/Linux.

On Windows

You will need to have the PuTTY application installed in order to connect to the Husky computer. In this application, you can connect to the 192.168.131.100 IP address as shown in the figure below.



Specify the destination you want to connect to

Host Name (or IP address): 192.168.131.100

Port: 22

Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

Load, save or delete a stored session

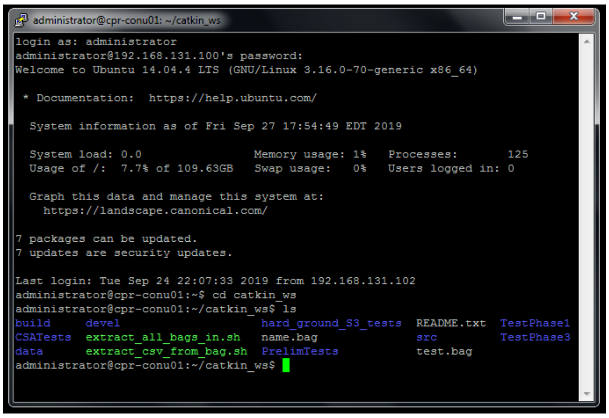
Saved Sessions: Husky

Default Settings
Husky
SSH tunnel ENCS

Close window on exit: ○ Always ○ Never ● Only on clean exit

Figure 1.2: PuTTY interface when setting the IP address to 192.168.131.100

After hitting the Open button, log in with the following credentials:
Username - administrator
Password - clearpath



administrator@cpr-conu01: ~/catkin_ws

```
login as: administrator
administrator@192.168.131.100's password:
Welcome to Ubuntu 14.04.4 LTS (GNU/Linux 3.16.0-70-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

  System information as of Fri Sep 27 17:54:49 EDT 2019

  System load: 0.0               Memory usage: 1%   Processes:       125
  Usage of /:  7.7% of 109.63GB   Swap usage:   0%   Users logged in: 0

  Graph this data and manage this system at:
    https://landscape.canonical.com/

7 packages can be updated.
7 updates are security updates.

Last login: Tue Sep 24 22:07:33 2019 from 192.168.131.102
administrator@cpr-conu01:~$ cd catkin_ws
administrator@cpr-conu01:~/catkin_ws$ ls
build     devel                     hard_ground_S3_tests  README.txt  TestPhase1
CSATests  extract_all_bags_in.sh    name.bag              src         TestPhase3
data      extract_csv_from_bag.sh   PrelimTests           test.bag
administrator@cpr-conu01:~/catkin_ws$
```

Figure 1.3: Terminal window after connecting to the Husky

On Mac/Linux

You can ssh to the PC at **192.168.131.100** by typing: `ssh administrator@192.168.131.100` followed by the password **clearpath** when prompted.

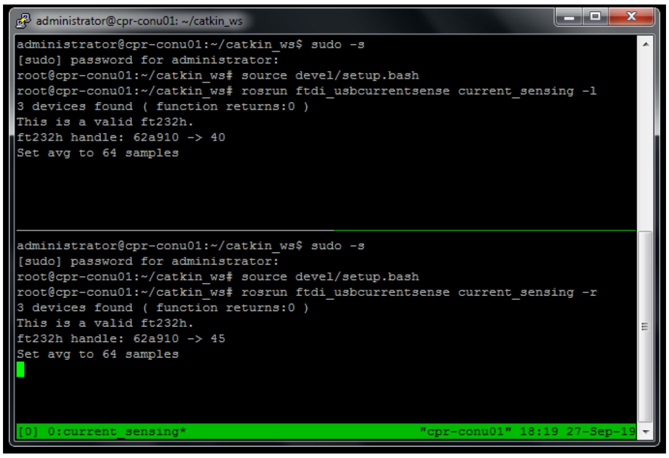
1.2.2.2 Starting the custom PCB current reading measurements

It should be noted that this subsection and the next (1.2.2.3) are only required after powering on the Husky. Also, for these sections, multiple windows are required. This can be achieve by repeating step 1.2.2.1 or by using the *tmux* command, explained in Appendix 3.

Once connected, move to the *catkin_ws* directory by typing: `cd catkin_ws`. To start the PCB current measurements, you need to be logged in as root by typing `sudo -s` and then the password: **clearpath**.

Then, while in the *catkin_ws*, first source the setup.bash file to tell ROS the working directory: `source devel/setup.bash`

You can finally run the program to start the measurements for the left (-l) and right (-r) motors by typing `rosrun ftdi_usbcurrentsense current_sensing -l` and `rosrun ftdi_usbcurrentsense current_sensing -r` respectively. Each of these lines are ran in their own window (or panel), as shown below.



```
administrator@cpr-conu01:~/catkin_ws$ sudo -s
[sudo] password for administrator:
root@cpr-conu01:~/catkin_ws# source devel/setup.bash
root@cpr-conu01:~/catkin_ws# rosrun ftdi_usbcurrentsense current_sensing -l
3 devices found ( function returns:0 )
This is a valid ft232h.
ft232h handle: 62a910 -> 40
Set avg to 64 samples
```

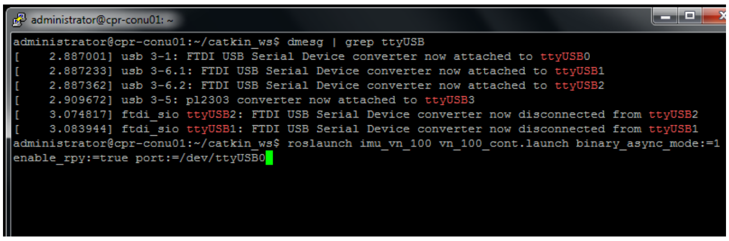
```
administrator@cpr-conu01:~/catkin_ws$ sudo -s
[sudo] password for administrator:
root@cpr-conu01:~/catkin_ws# source devel/setup.bash
root@cpr-conu01:~/catkin_ws# rosrun ftdi_usbcurrentsense current_sensing -r
3 devices found ( function returns:0 )
This is a valid ft232h.
ft232h handle: 62a910 -> 45
Set avg to 64 samples

[0] 0:current_sensing*                         "cpr-conu01" 18:19 27-Sep-19
```

Figure 1.4: Terminal window (with two panels using tmux) after starting the left and right current measurements

1.2.2.3 Starting the IMU data publishing to the */imu/rpy* ROS topic

In a new window, move to the *catkin_ws* directory and source the devel/setup.bash file (`source devel/setup.bash`). Then, verify the USB port of the IMU by typing: `dmesg | grep ttyUSB`. As shown in Fig. 1.5, the port in the one with the FTDI USB Serial Device still connected (the other two are the USB ports for the USB current sensors). In this case, the USB port is `ttyUSB0`.



```
administrator@cpr-conu01:~/catkin_ws$ dmesg | grep ttyUSB
[    2.887001] usb 3-1: FTDI USB Serial Device converter now attached to ttyUSB0
[    2.887233] usb 3-6.1: FTDI USB Serial Device converter now attached to ttyUSB1
[    2.887362] usb 3-6.2: FTDI USB Serial Device converter now attached to ttyUSB2
[    2.909672] usb 3-5: pl2303 converter now attached to ttyUSB3
[    3.074817] ftdi_sio ttyUSB2: FTDI USB Serial Device converter now disconnected from ttyUSB2
[    3.083944] ftdi_sio ttyUSB1: FTDI USB Serial Device converter now disconnected from ttyUSB1
administrator@cpr-conu01:~/catkin_ws$ roslaunch imu_vn_100 vn_100_cont.launch binary_async_mode:=1
enable_rpy:=true port:=/dev/ttyUSB0
```

Figure 1.5: Terminal window after starting the IMU publishing when connected to the `ttyUSB0` port

Once the ttyUSB port is known, run the command: `roslaunch imu_vn_100 vn_100_cont.launch binary_async_mode:=1 enable_rpy:=true port:=/dev/ttyUSB<port number>` with the correct port number.

Now that everything is set up, you may first verify that all the useful data is being published to a ROS topic. To do so, type `rostopic echo <topicname>` (and Ctrl+C to stop the process) with the following topicname:

/ftdi/motor_current/left left DC motor

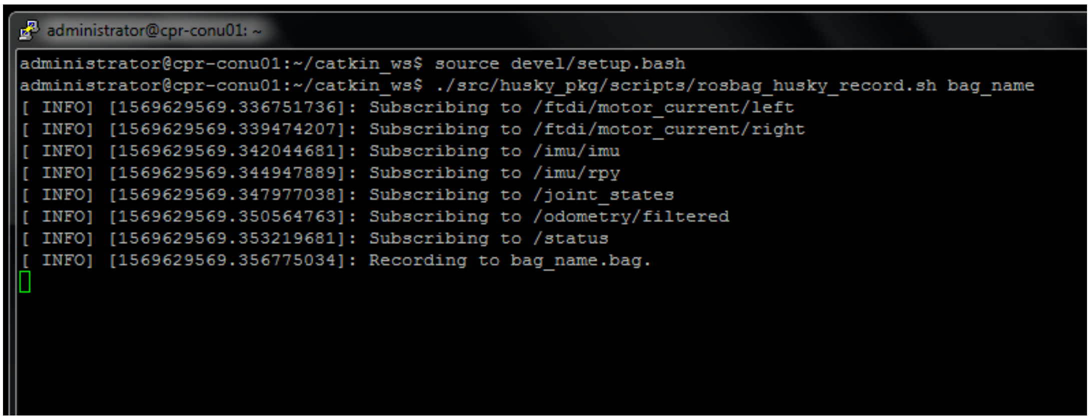
/ftdi/motor_current/right right DC motor

/imu/rpy IMU roll, pitch and yaw measurements

1.2.2.4 Start recording the data

In a new windows, move to the *catkin_ws* directory and source the setup.bash file. Then, run the *rosbag_husky_record.sh* script with a specified bag name (i.e. the test number) by typing:

```
./src/husky_pkg/script/rosbag_husky_record.sh <bag name>
```



administrator@cpr-conu01: ~

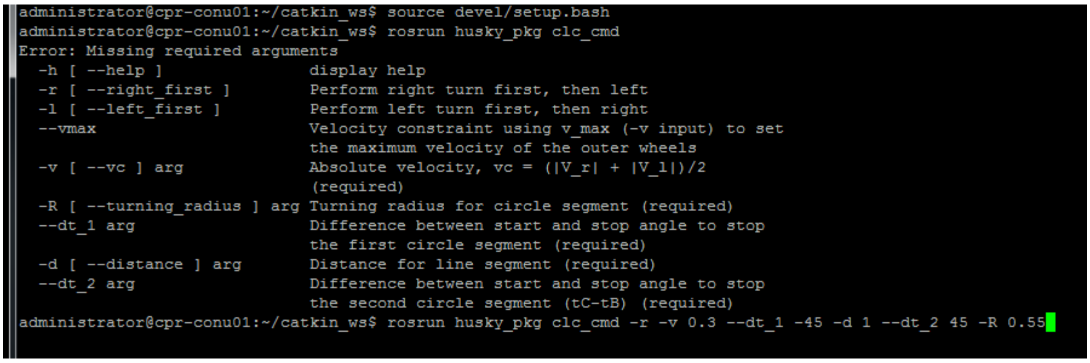
```
administrator@cpr-conu01:~/catkin_ws$ source devel/setup.bash
administrator@cpr-conu01:~/catkin_ws$ ./src/husky_pkg/scripts/rosbag_husky_record.sh bag_name
[ INFO] [1569629569.336751736]: Subscribing to /ftdi/motor_current/left
[ INFO] [1569629569.339474207]: Subscribing to /ftdi/motor_current/right
[ INFO] [1569629569.342044681]: Subscribing to /imu/imu
[ INFO] [1569629569.344947889]: Subscribing to /imu/rpy
[ INFO] [1569629569.347977038]: Subscribing to /joint_states
[ INFO] [1569629569.350564763]: Subscribing to /odometry/filtered
[ INFO] [1569629569.353219681]: Subscribing to /status
[ INFO] [1569629569.356775034]: Recording to bag_name.bag.
```

Figure 1.6: Terminal window after starting the darta recording in a bag file

1.2.2.5 Commanding a CLC/PLC/CLP/PLP path

After starting the data recording, you may now run either the `clc_cmd`, `plc_cmd` or `clp_cmd` ROS program. For example, a right-turn first 45°-10-45° PLP path with ¡R_CMD¿ can be commanded following:

```
rosrun husky_pkg clc_cmd -r -v 0.3 --dt_1 -45 --dt_2 45 -d 10 -R 0
```



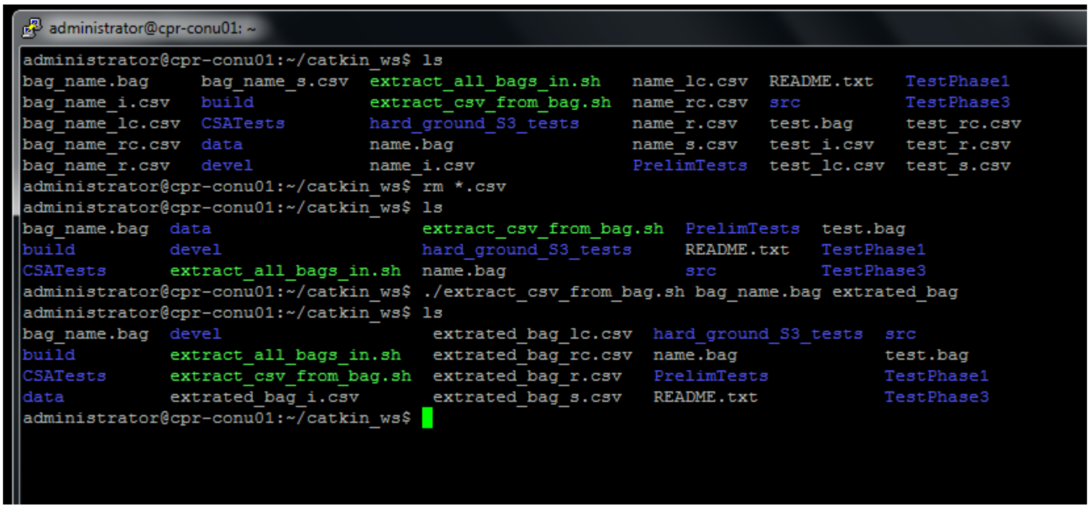
```
administrator@cpr-conu01:~/catkin_ws$ source devel/setup.bash
administrator@cpr-conu01:~/catkin_ws$ rosrun husky_pkg clc_cmd
Error: Missing required arguments
  -h [ --help ]                 display help
  -r [ --right_first ]          Perform right turn first, then left
  -l [ --left_first ]           Perform left turn first, then right
  --vmax                        Velocity constraint using v_max (-v input) to set
                                the maximum velocity of the outer wheels
  -v [ --vc ] arg               Absolute velocity, vc = (|V_r| + |V_l|)/2
                                (required)
  -R [ --turning_radius ] arg Turning radius for circle segment (required)
  --dt_1 arg                    Difference between start and stop angle to stop
                                the first circle segment (required)
  -d [ --distance ] arg         Distance for line segment (required)
  --dt_2 arg                    Difference between start and stop angle to stop
                                the second circle segment (tC-tB) (required)
administrator@cpr-conu01:~/catkin_ws$ rosrun husky_pkg clc_cmd -r -v 0.3 --dt_1 -45 -d 1 --dt_2 45 -R 0.55
```

Figure 1.7: Terminal window before commanding a CLC path (right turn - straigth - left turn) with $R_{CLC} = 0.55$ m and $\theta_{start} = \theta_{end} = 45°$ with a straight line segment of $d = 0$ m, under a $v_c = 0.3$ m/s constraint.

1.2.2.6 Extracting the data from a bag file to csv files

After the test is completed, stop the data recording (Ctrl + C). The bag file will then be in the *catkin_ws* directory. To extract the data in csv files, use the following script:

```
./extract_csv_from_bag.sh <bag name> <csv filename>
```



```
administrator@cpr-conu01:~/catkin_ws$ ls
bag_name.bag     bag_name_s.csv  extract_all_bags_in.sh   name_lc.csv  README.txt   TestPhase1
bag_name_i.csv   build           extract_csv_from_bag.sh  name_rc.csv  src          TestPhase3
bag_name_lc.csv  CSATests        hard_ground_S3_tests     name_r.csv   test.bag     test_rc.csv
bag_name_rc.csv  data            name.bag                 name_s.csv   test_i.csv   test_r.csv
bag_name_r.csv   devel           name_i.csv               PrelimTests  test_lc.csv  test_s.csv
administrator@cpr-conu01:~/catkin_ws$ rm *.csv
administrator@cpr-conu01:~/catkin_ws$ ls
bag_name.bag  data                    extract_csv_from_bag.sh  PrelimTests  test.bag
build         devel                   hard_ground_S3_tests     README.txt   TestPhase1
CSATests      extract_all_bags_in.sh  name.bag                 src          TestPhase3
administrator@cpr-conu01:~/catkin_ws$ ./extract_csv_from_bag.sh bag_name.bag extrated_bag
administrator@cpr-conu01:~/catkin_ws$ ls
bag_name.bag  devel                    extrated_bag_lc.csv  hard_ground_S3_tests  src
build         extract_all_bags_in.sh   extrated_bag_rc.csv  name.bag              test.bag
CSATests      extract_csv_from_bag.sh  extrated_bag_r.csv   PrelimTests           TestPhase1
data          extrated_bag_i.csv       extrated_bag_s.csv   README.txt            TestPhase3
administrator@cpr-conu01:~/catkin_ws$ 
```

Figure 1.8: Terminal window after extracting the data from the bag_name.bag file to csv files named extracted_bad_*.csv

2. Argo J5

2.1 Login

When the connectivity light at the back is green, you may connect to the *TP-LINK_12_173_ARGOJ5_AP* network with the password: **aloPortch4r**.

The next step is to connect to the onboard computer, as explained below, for Windows and Mac/Linux. You will need to have the PuTTY application installed in order to connect to the Husky computer. In this application, you can connect to the 192.168.0.20 IP address as shown in the figure below.

After hitting the Open button, log in with the following credentials:

Username - ugv
Password - !ugX13

On Mac/Linux

You can ssh to the PC at **192.168.0.20** by typing: `ssh ugv@192.168.0.20` followed by the password **!ugX13** when prompted.

2.2 Start the recording

There should be a similar script to the *rosbag_husky_record.sh* script on the Husky to start recording the useful data in a rosbag with a specified name. Following the test, use Ctrl+C to stop the recording session.

2.3 Commanding a CLC/PLC/CLP/PLP path

Like with the Husky, move to the *catkin_ws* and source the */devel/setup.bash* file before using the `clc_cmd`, `plc_cmd` or `clp_cmd` ROS program following the same format as the Husky programs (see Section 1.2.2.5).

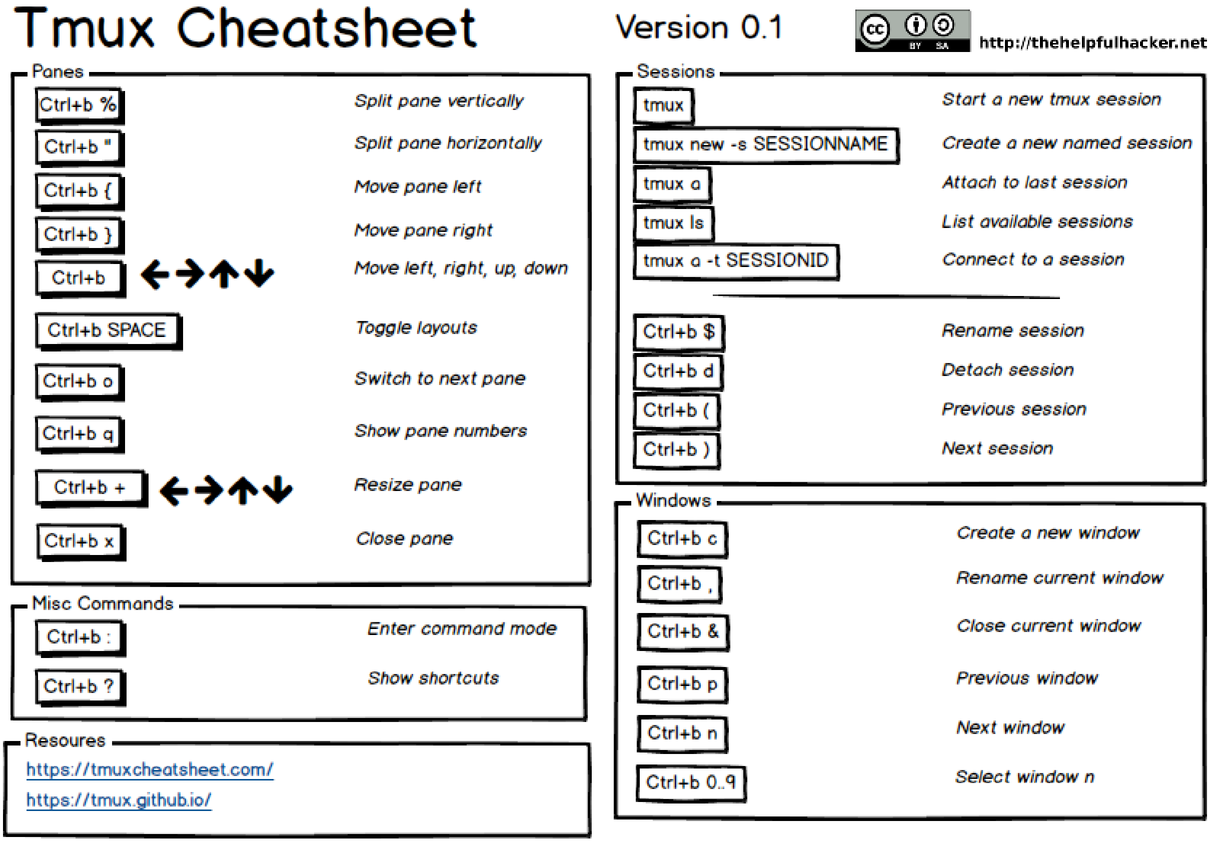
Note

This was written without access to the Argo's computer (waiting for new batteries). Therefore, there may be some slight differences in the file location or folder name but essentially, the idea is similar to the Husky software (without Section 1.2.2.2 and 1.2.2.3).

3. Appendix

3.1 Using the *tmux* command

To have multiple windows in the same session, you can first type the `tmux` command. In this environment, you can use the shortcuts in the figure below.



Tmux Cheatsheet

Version 0.1

http://thehelpfulhacker.net

Panes

Ctrl+b %	*Split pane vertically*
Ctrl+b "	*Split pane horizontally*
Ctrl+b {	*Move pane left*
Ctrl+b }	*Move pane right*
Ctrl+b ←→↑↓	*Move left, right, up, down*
Ctrl+b SPACE	*Toggle layouts*
Ctrl+b o	*Switch to next pane*
Ctrl+b q	*Show pane numbers*
Ctrl+b + ←→↑↓	*Resize pane*
Ctrl+b x	*Close pane*

Misc Commands

Ctrl+b :	*Enter command mode*
Ctrl+b ?	*Show shortcuts*

Resoures

https://tmuxcheatsheet.com/

https://tmux.github.io/

Sessions

tmux	*Start a new tmux session*
tmux new -s SESSIONNAME	*Create a new named session*
tmux a	*Attach to last session*
tmux ls	*List available sessions*
tmux a -t SESSIONID	*Connect to a session*
Ctrl+b $	*Rename session*
Ctrl+b d	*Detach session*
Ctrl+b (	*Previous session*
Ctrl+b)	*Next session*

Windows

Ctrl+b c	*Create a new window*
Ctrl+b ,	*Rename current window*
Ctrl+b &	*Close current window*
Ctrl+b p	*Previous window*
Ctrl+b n	*Next window*
Ctrl+b 0..9	*Select window n*