How to run dVRK

Andrej Simeunović
simeunovic.1@osu.edu

Contents

1 Preliminaries

This document describes startup processes for the dVRK system. There are several ways to run the system. Methods using Qt Widget graphical user interfaces (GUIs) and the ROS middleware (Fig. 1) are described here.

Each method has its own section below and they are all standalone. You do not need to reference previous sections when jumping to the method you want to use.

The following software is required to run the system.

- dVRK software developed by JHU. To verify this is installed go the `/local/storage` folder and verify there is a `catkin_ws` folder.
- A good terminal. This guide uses Terminator.
- (For ROS) ROS Melodic.
- (For Section 3) Matlab. R2020a is recommended given the state of compatibility with R2020b and later.
- (For cameras) Pylon Viewer.

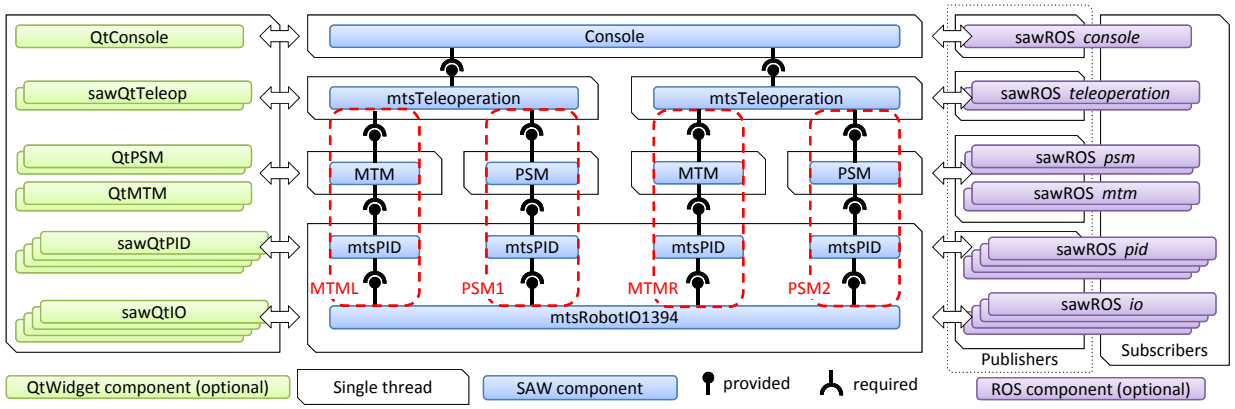


Figure 1: dVRK can be run via Qt Widget or ROS.[1]

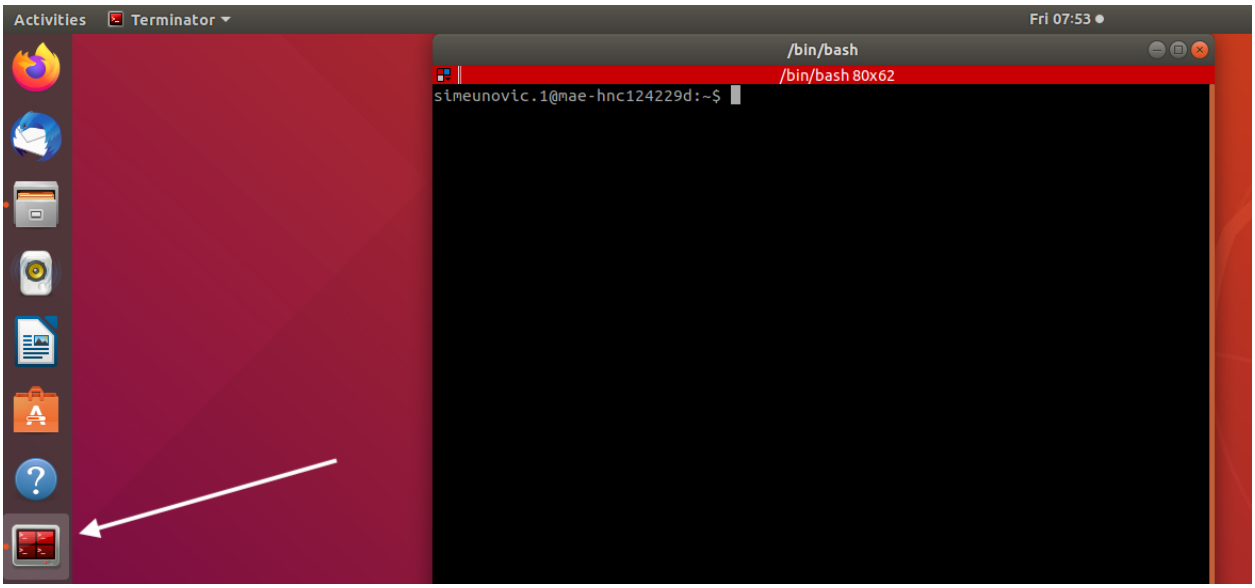
Further information is found on the dVRK wiki.

[1]Figure reprinted with permission from P. Kazanzides, Z. Chen, A. Deguet, G. S. Fischer, R. H. Taylor, and S. P. DiMaio, "An open-source research kit for the da Vinci surgical system," in IEEE Intl. Conf. on Robotics and Automation (ICRA), 2014, pp. 6434–6439.

2 Qt Widgets

2.1 PSM

1. Open terminator.



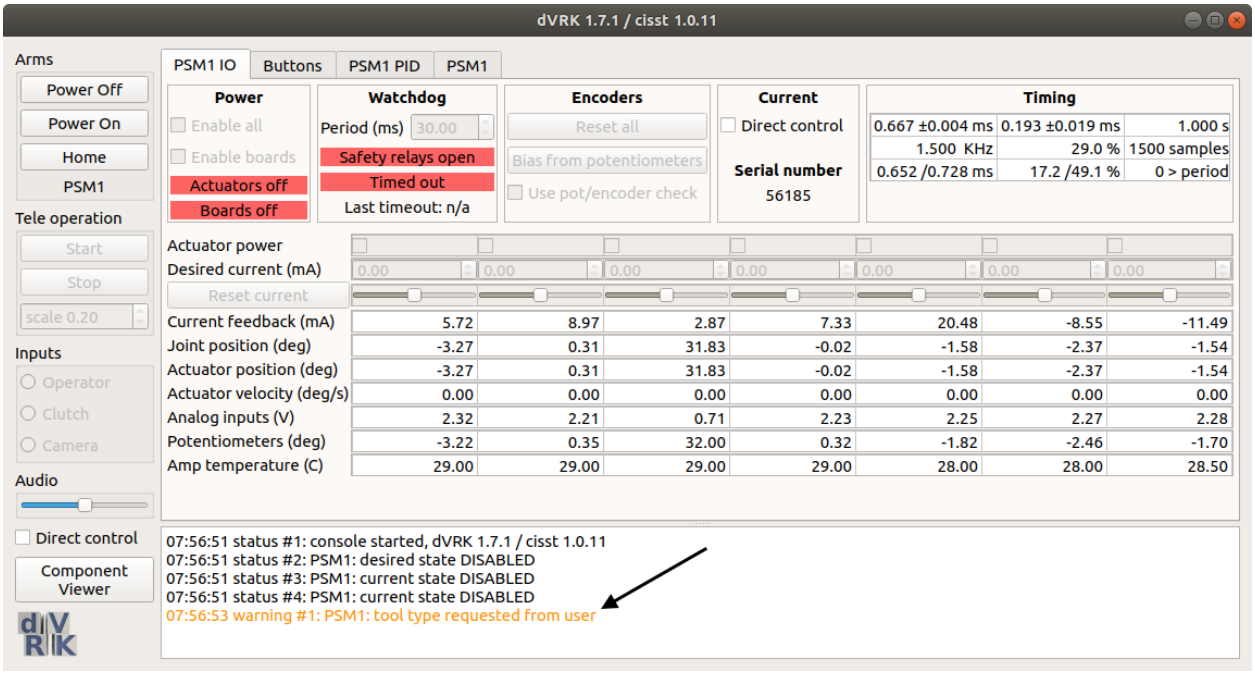
2. Go to the osu-dVRK folder by typing:

   ```
   $ cd /local/storage/catkin_ws/src/cisst-saw/sawIntuitiveResearchKit/share/osu-dVRK/
   ```

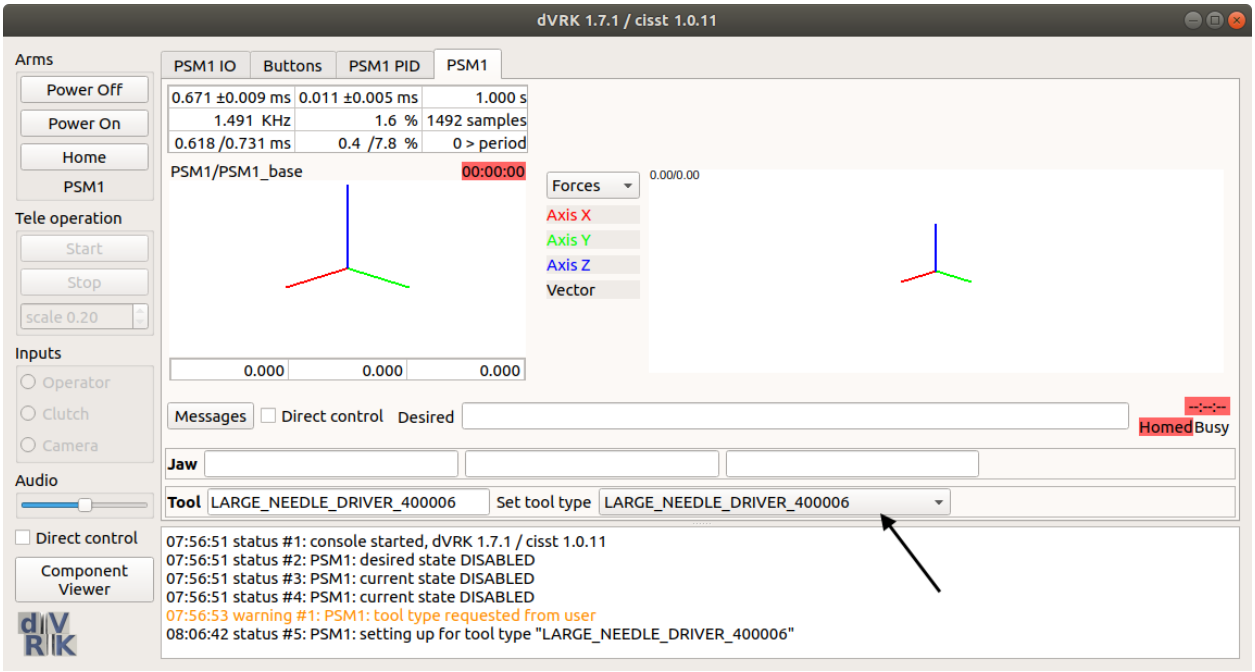
3. Open the PSM Qt Widget by typing:

   ```
   $ sawIntuitiveResearchKitQtConsoleJSON -j console-PSM1.json
   ```

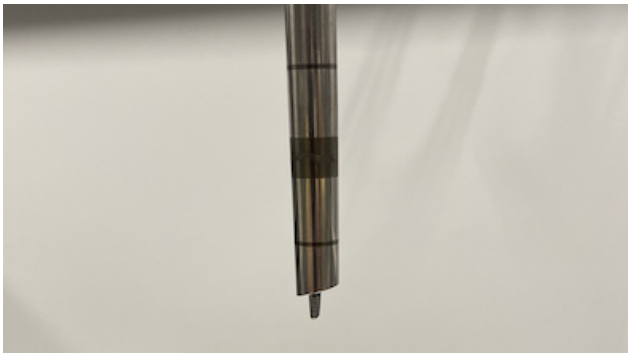
4. The PSM QT widget will launch and request a tool type selection from the user.



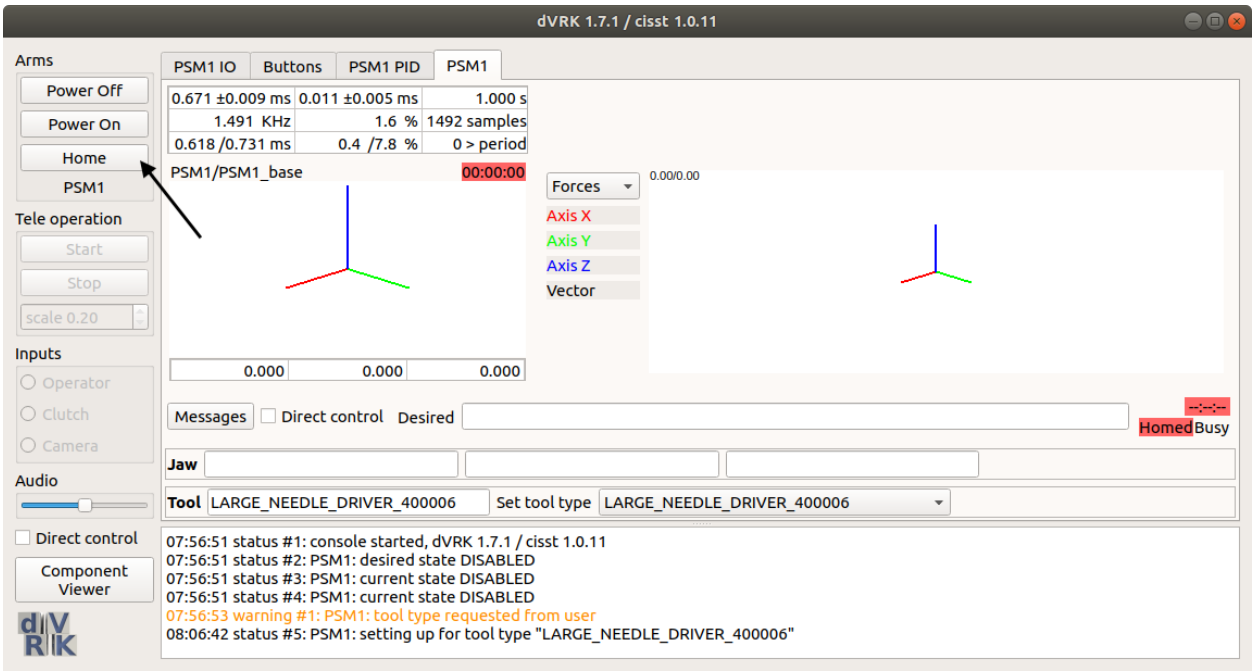
5. Select the appropriate tool in the PSM1 tab.



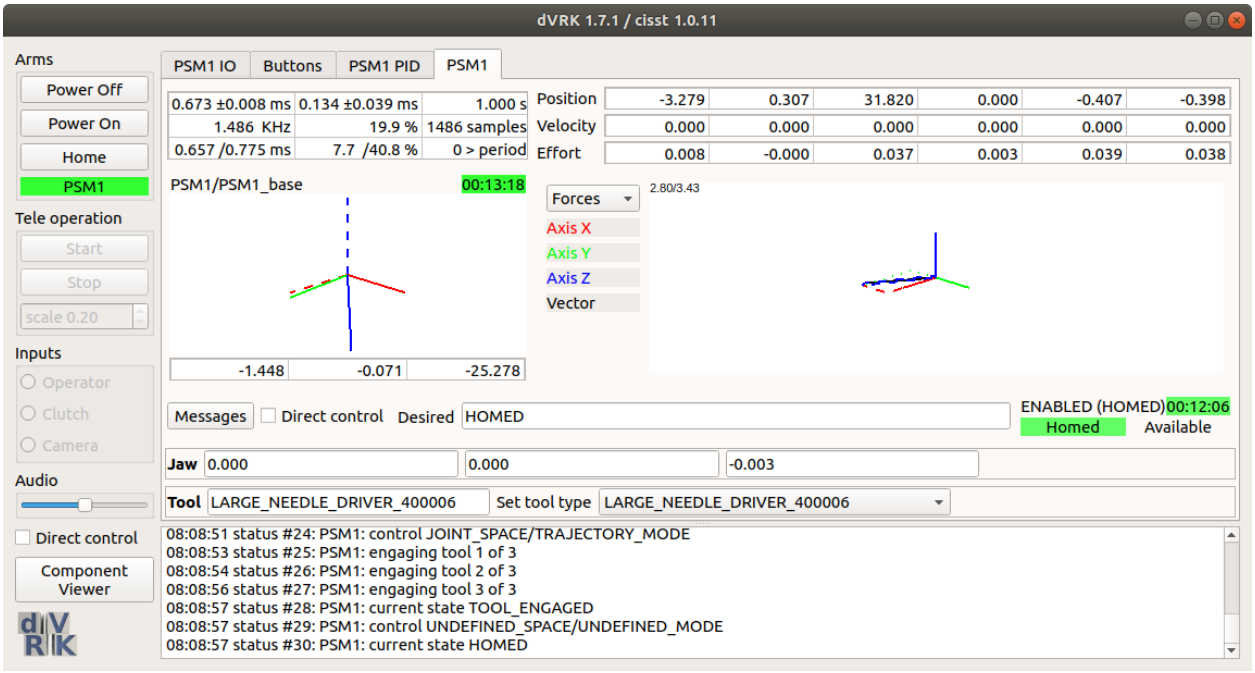
6. Manually retract the tool until the end effector is inside the cannula.



7. Press the `Home` button in the Qt Widget.

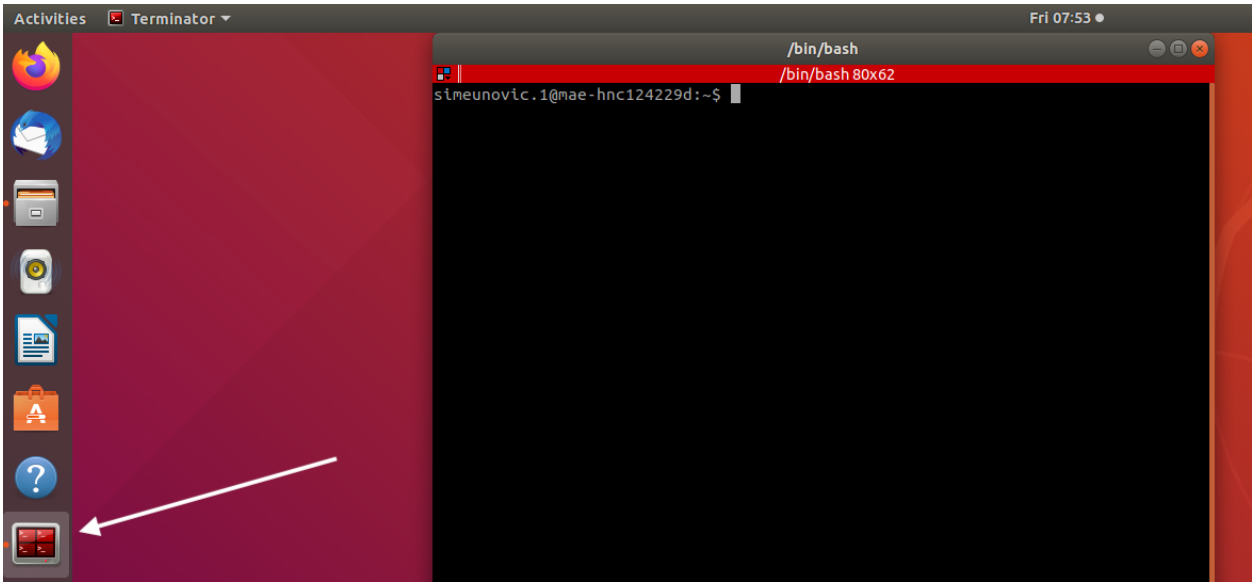


Homing will power on the system and initiate the homing routine. Once it completes, the system is ready to use. You can turn system power on/off using the provided buttons. You do not need to re-home the system unless you close the Qt Widget or power cycle the system.



2.2 MTM

1. Open terminator.



2. Go to the osu-dVRK folder by typing:

   ```
   $ cd /local/storage/catkin_ws/src/cisst-saw/sawIntuitiveResearchKit/share/osu-dVRK/
   ```

3. Open the appropriate Qt widget.

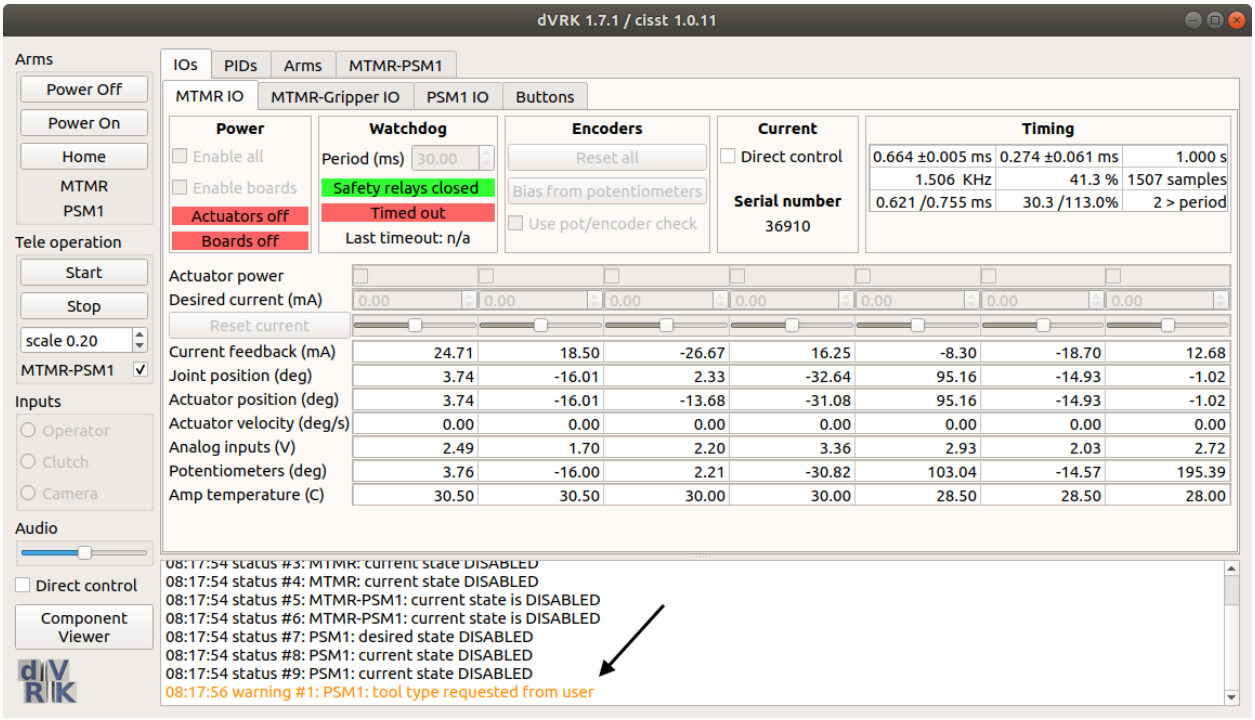
   For TeleOp, type:

```
$ sawIntuitiveResearchKitQtConsoleJSON -j console-MTMR-PSM1-Teleop.json
```

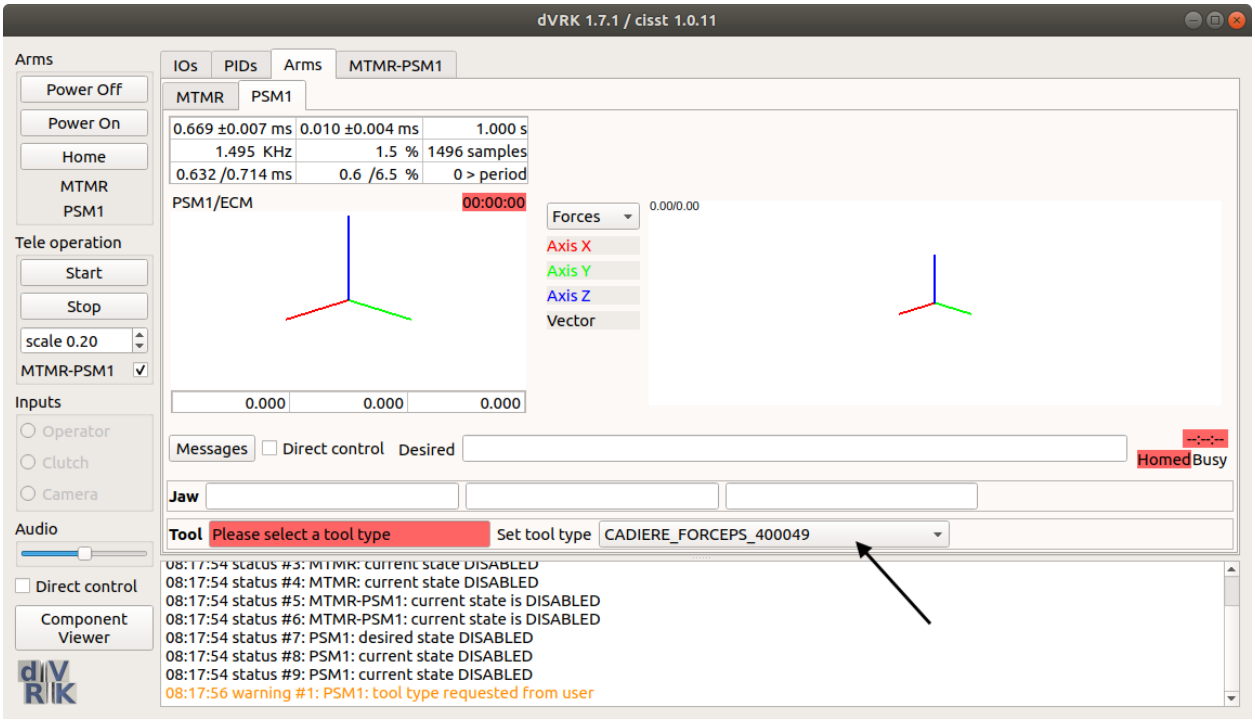
For MTM only, type:

```
$ sawIntuitiveResearchKitQtConsoleJSON -j console-MTMR.json
```

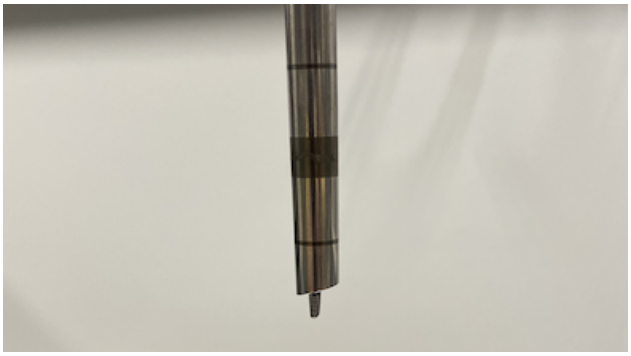
4. The chosen widget will launch and request a tool type selection from the user.



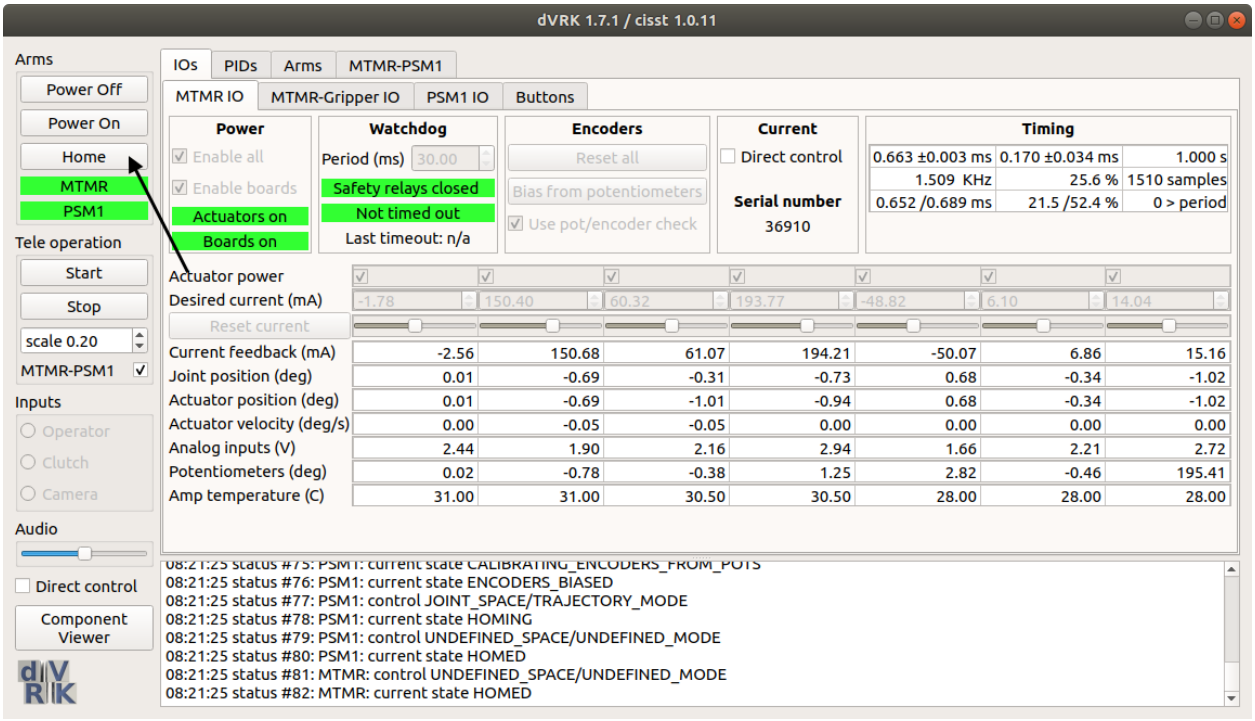
5. Select the appropriate tool in the PSM1 tab.



6. Manually retract the tool until the end effector is inside the cannula.

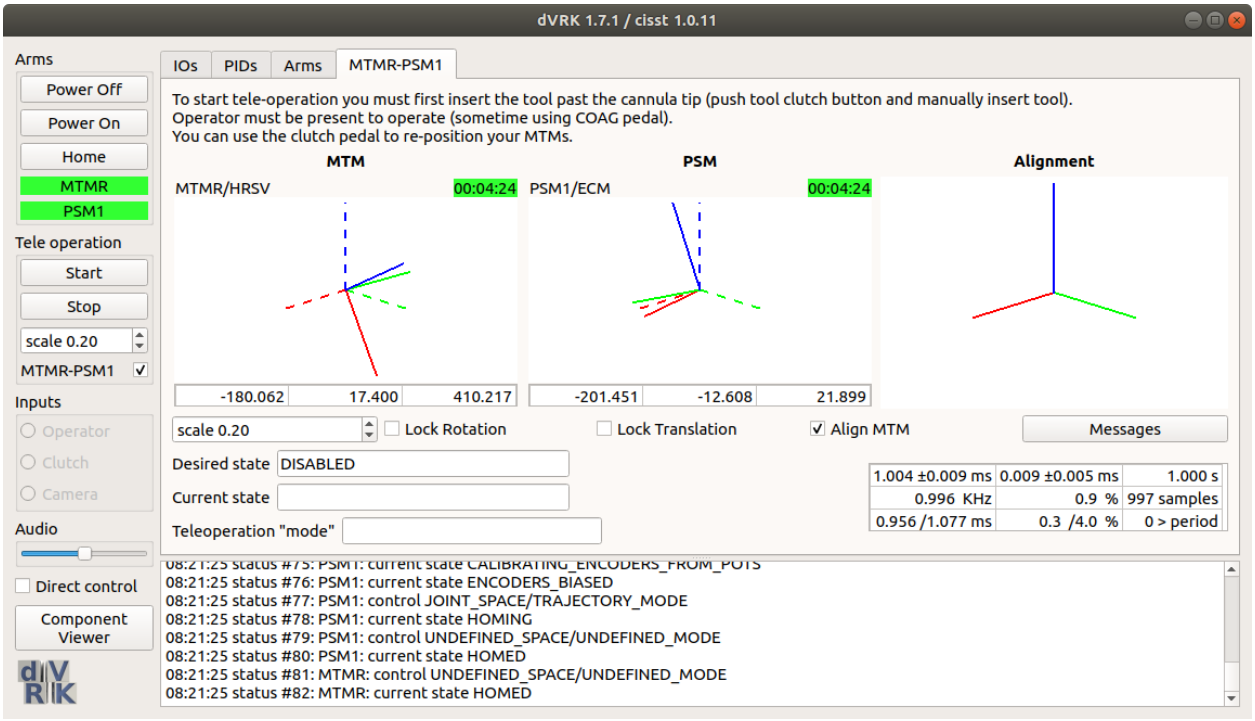


7. Press the `Home` button in the Qt Widget.



Homing will power on the system and initiate the homing routine. Once it completes, the system is ready to use. You can turn system power on/off using the provided buttons. You do not need to re-home the system unless you close the Qt Widget or power cycle the system.

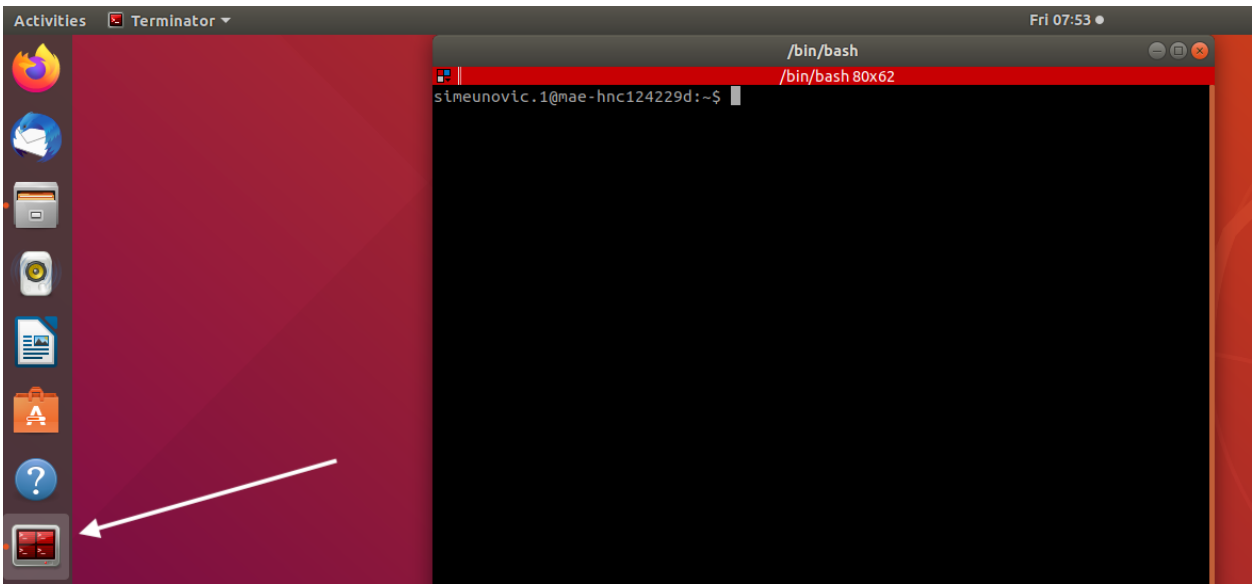
Make sure to follow additional instructions in the Qt Widget if your configuration requires it (see example below).



3 Matlab via ROS

3.1 ROS via Matlab

1. Open terminator. Right click to split the window into 3 terminals horizontally or vertically.

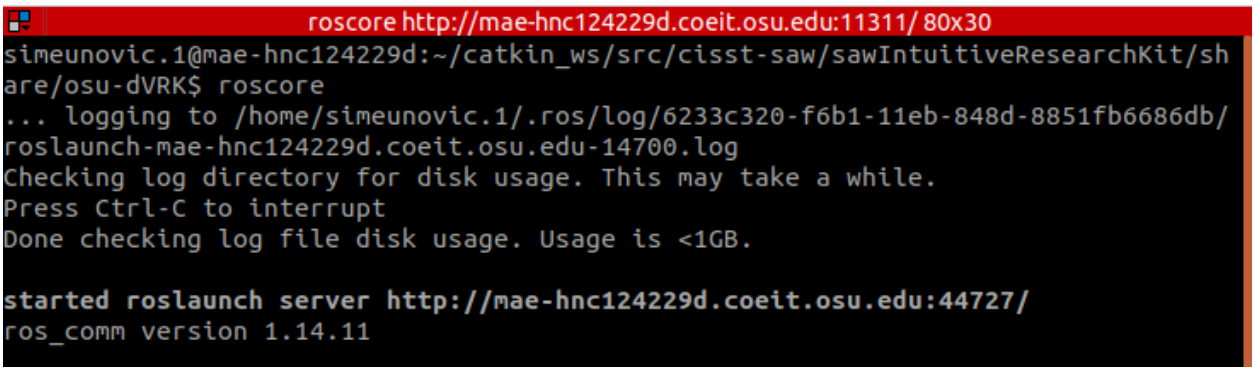


2. In the first terminal, go to the osu-dVRK folder by typing:

```
$ cd /local/storage/catkin_ws/src/cisst-saw/sawIntuitiveResearchKit/share/osu-dVRK/
```

3. In the second terminal, start a ROS core by typing:

```
$ roscore
```



4. In the third terminal, open Matlab by typing:

```
$ matlab
```

5. In the first terminal, open the appropriate Qt widget.

   For PSM, type:

```
$ rosrun dvrk_robot dvrk_console_json -j console-PSM1.json
```

   For Teleop, type:

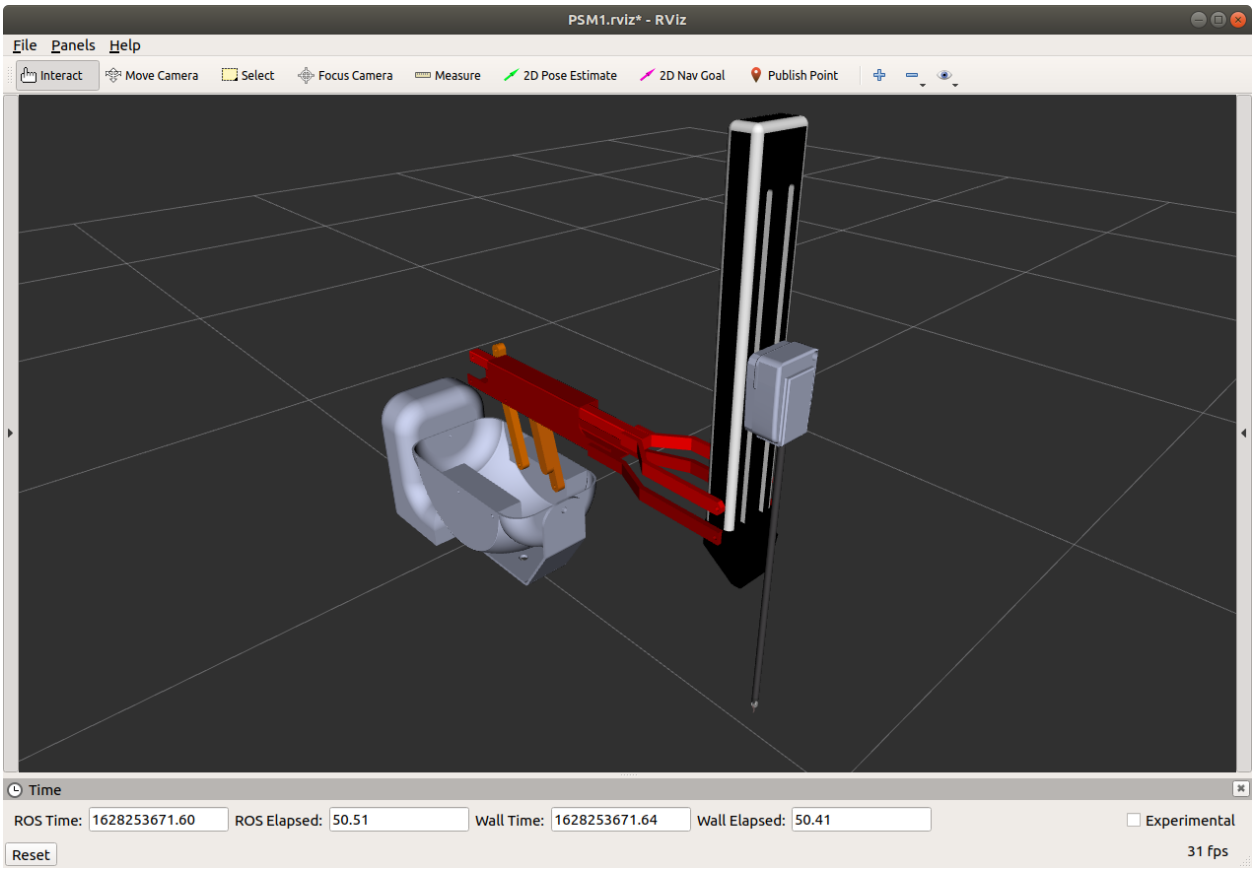
```
$ rosrun dvrk_robot dvrk_console_json -j console-MTMR-PSM1-Teleop.json
```

   For MTM, type:

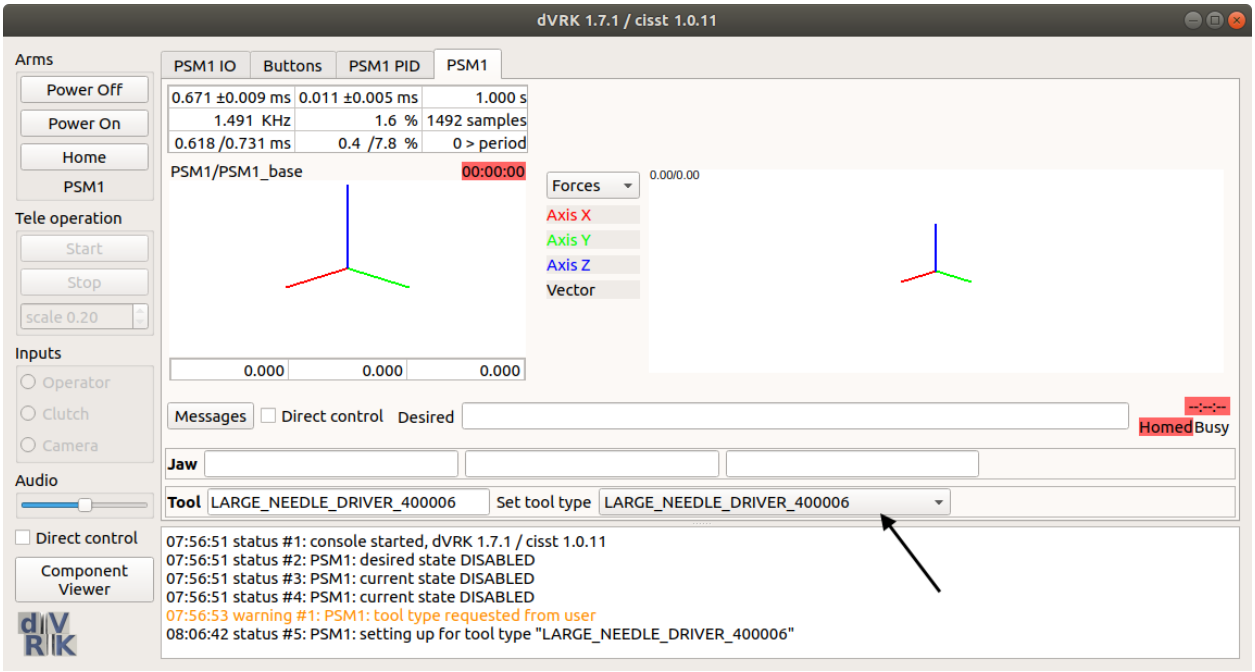
```
$ rosrun dvrk_robot dvrk_console_json -j console-MTMR.json
```

   For the dVRK simulation in ROS rviz, type:

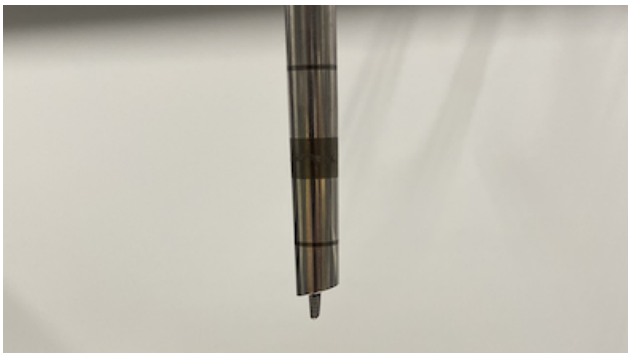
```
$ roslaunch dvrk_robot dvrk_arm_rviz.launch arm:=PSM1
```



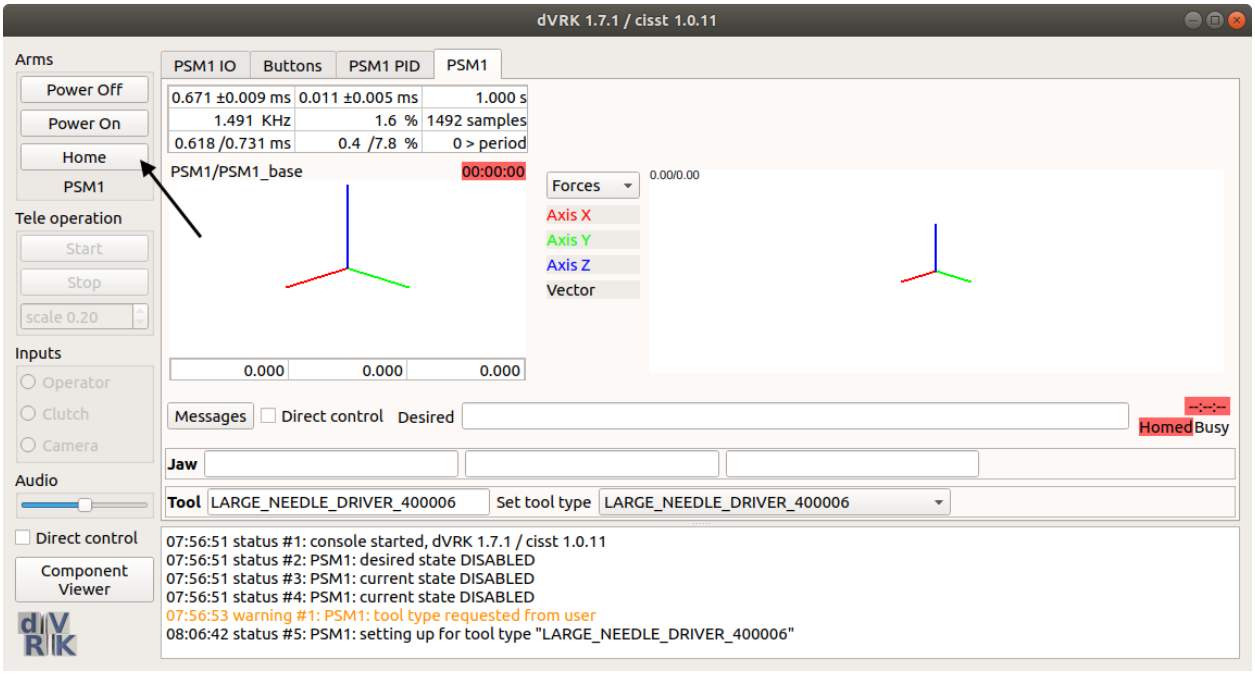
6. Select the appropriate tool in the PSM1 tab.



7. Manually retract the tool until the end effector is inside the cannula.



8. Press the `Home` button in the Qt Widget.



9. In the Matlab command window, type:

   ```
   rosinit
   ```

10. To verify Matlab and dVRK are on the ROS network, in the command window type:

    ```
    rosnode list
    ```

    ```
    >> rosnode list
    /PSM1/joint_state_publisher
    /PSM1/robot_state_publisher
    /dvrk_PSM1_node
    /matlab_global_node_65601
    /rosout
    /rviz
    ```

11. Create an arm object in the command window. For example, to assign the PSM as an object, type:

    ```
    r = dvrk.arm('PSM1')
    ```

    If you execute the command without a suppressing semicolon you will see a list of available ROS topics for your chosen arm.

```
>> r = dvrk.arm('PSM1')

r =

  arm with properties:

              body: [1×1 dvrk.arm_cf]
           spatial: [1×1 dvrk.arm_cf]
             local: [1×1 dvrk.arm_local]
       setpoint_cf: @(varargin)self.setpoint_cf(varargin{:})
          servo_cp: @(varargin)self.servo_cp(varargin{:})
       measured_js: @(varargin)self.measured_js(varargin{:})
           disable: @(varargin)self.disable(varargin{:})
          servo_cf: @(varargin)self.servo_cf(varargin{:})
           move_cp: @(varargin)self.move_cp(varargin{:})
           move_jr: @(varargin)self.move_jr(varargin{:})
            enable: @(varargin)self.enable(varargin{:})
       measured_cp: @(varargin)self.measured_cp(varargin{:})
   operating_state: @(varargin)self.operating_state(varargin{:})
       measured_cv: @(varargin)self.measured_cv(varargin{:})
       setpoint_js: @(varargin)self.setpoint_js(varargin{:})
           is_busy: @(varargin)self.is_busy(varargin{:})
          servo_jp: @(varargin)self.servo_jp(varargin{:})
       measured_cf: @(varargin)self.measured_cf(varargin{:})
          servo_jr: @(varargin)self.servo_jr(varargin{:})
     wait_for_busy: @(varargin)self.wait_for_busy(varargin{:})
          servo_jf: @(varargin)self.servo_jf(varargin{:})
       setpoint_cp: @(varargin)self.setpoint_cp(varargin{:})
     state_command: @(varargin)self.state_command(varargin{:})
           move_jp: @(varargin)self.move_jp(varargin{:})
              home: @(varargin)self.home(varargin{:})
```

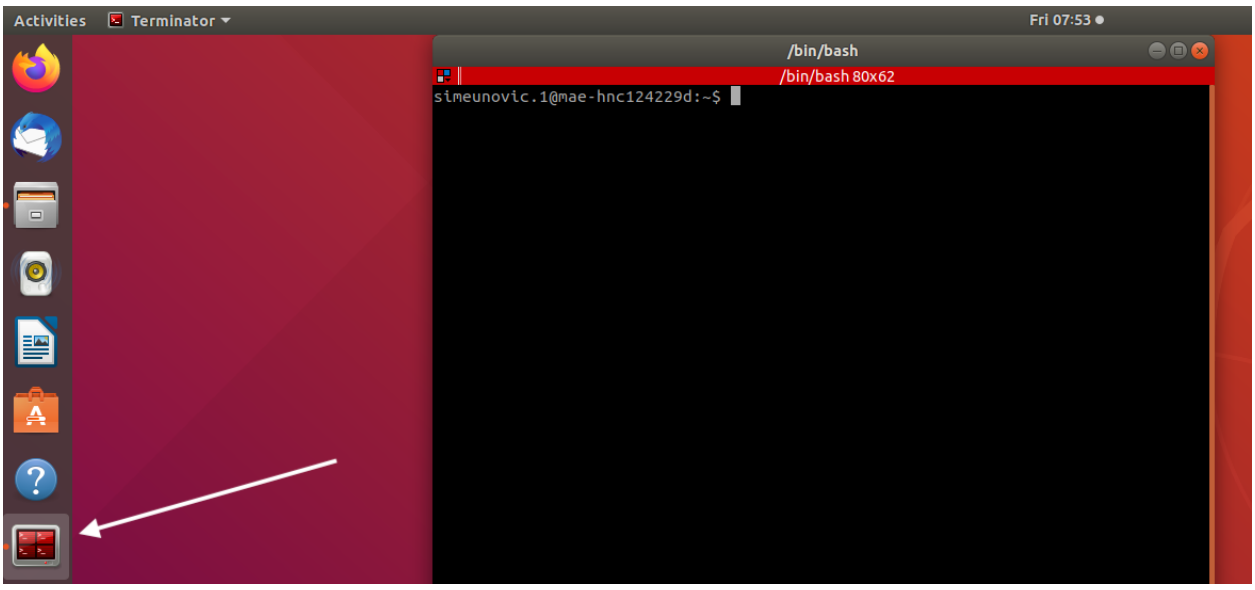
12. Interact with any of the ROS topics from the command window using the arm object.

```
>> r.setpoint_cp()

ans =

   -0.0000    1.0000   -0.0000    0.0000
    1.0000    0.0000    0.0000    0.0000
    0.0000    0.0000   -1.0000   -0.1135
         0         0         0    1.0000

>> r.measured_cp()

ans =

   -0.0000    1.0000   -0.0000    0.0000
    1.0000    0.0000    0.0000    0.0000
    0.0000    0.0000   -1.0000   -0.1135
         0         0         0    1.0000
```

13. When done, close all applications except Terminator.
14. In the first terminal, kill the dVRK process by pressing 'Ctrl+C'.
15. In the second terminal, shutdown the ROS core by pressing 'Ctrl+C'.

3.2 Matlab GUI

1. Open terminator. Right click to split the window into 3 terminals horizontally or vertically.

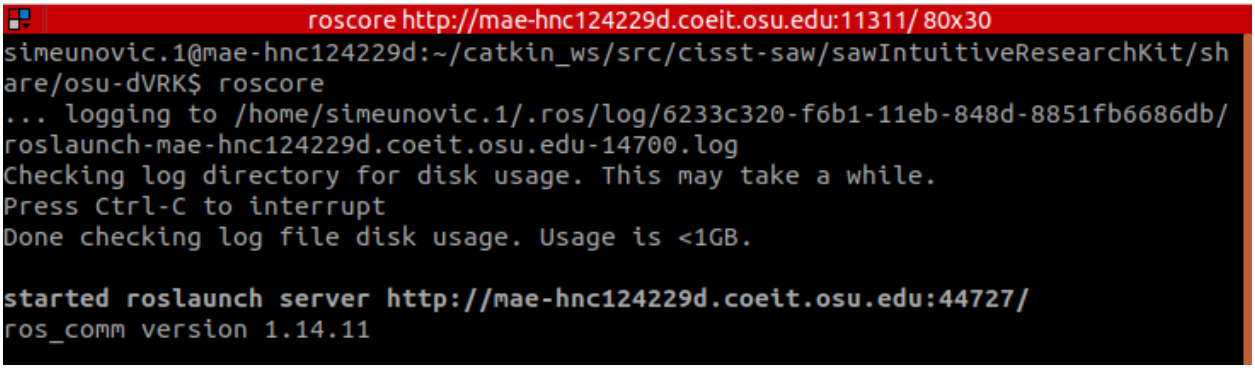


2. In the first terminal, go to the osu-dVRK folder by typing:

```
$ cd /local/storage/catkin_ws/src/cisst-saw/sawIntuitiveResearchKit/share/osu-dVRK/
```

3. In the second terminal, start a ROS core by typing:

   ```
   $ roscore
   ```



4. In the third terminal, open Matlab by typing:

   ```
   $ matlab
   ```

5. In the first terminal, open the appropriate Qt widget.

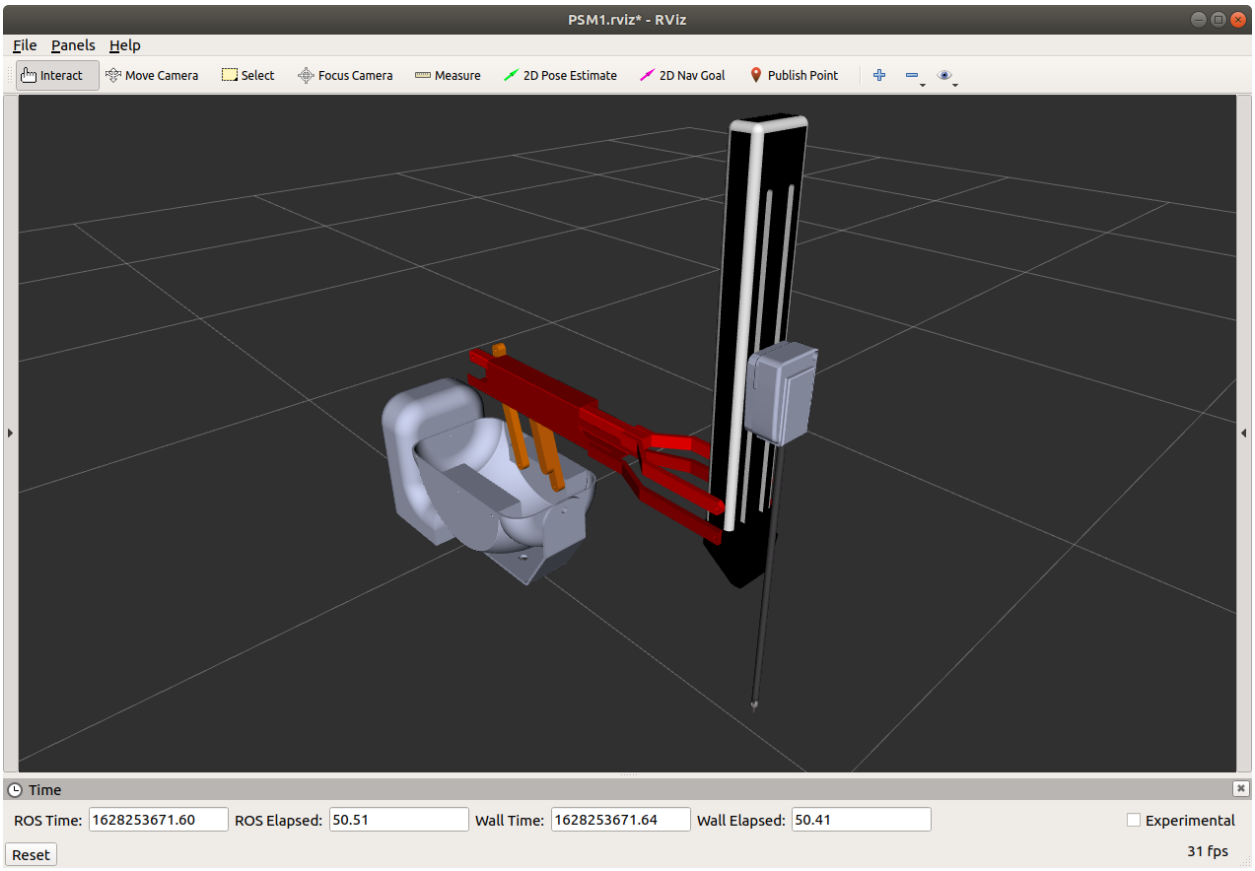
   For PSM, type:

   ```
   $ rosrun dvrk_robot dvrk_console_json -j console-PSM1.json
   ```

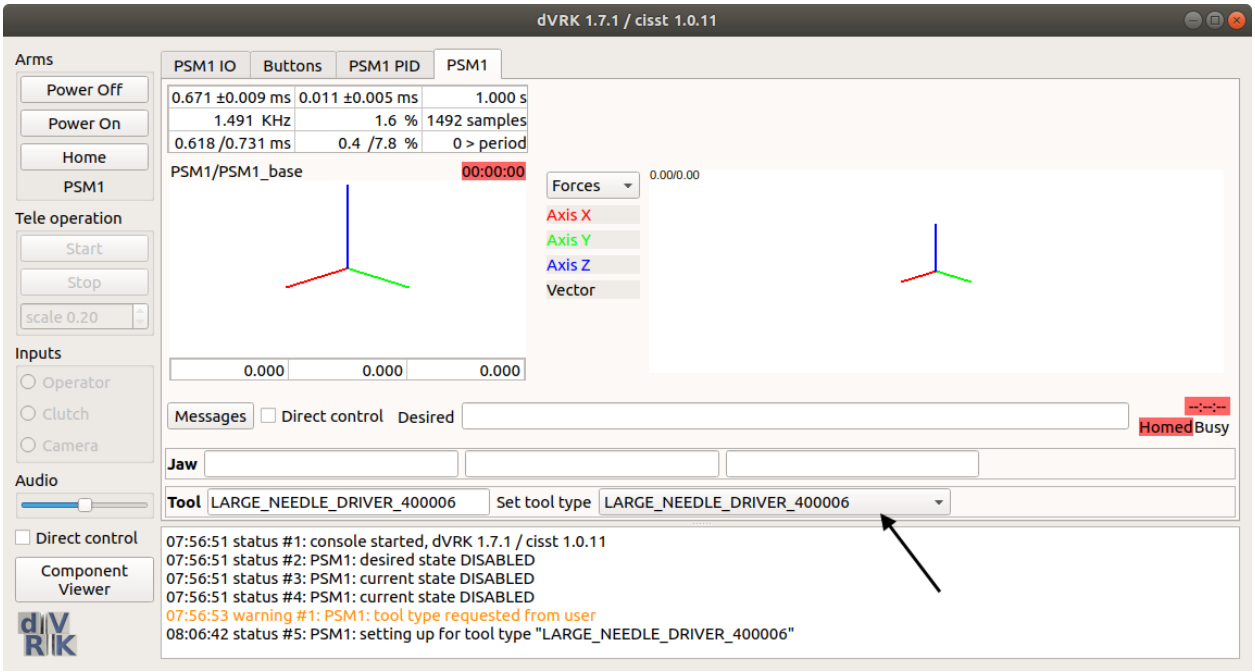
   For the dVRK simulation in ROS rviz, type:

   ```
   $ roslaunch dvrk_robot dvrk_arm_rviz.launch arm:= PSM1
   ```

   MTM and TeleOp are not currently supported via Matlab GUI.

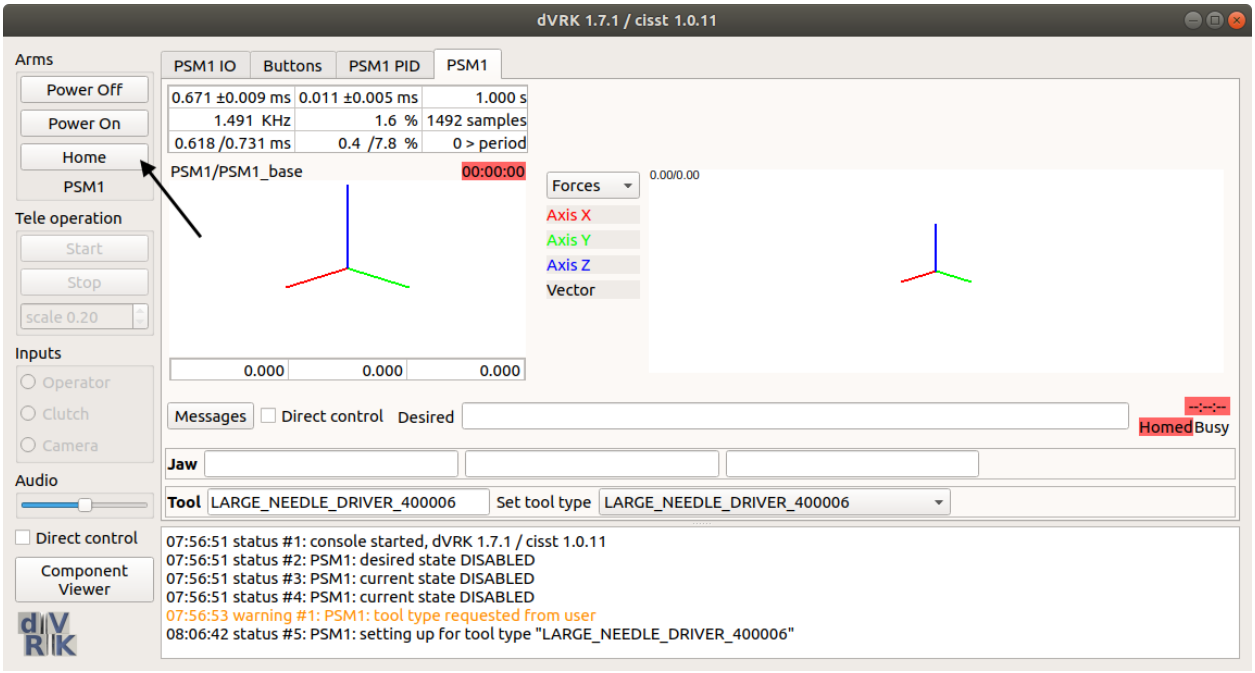


6. Select the appropriate tool in the PSM1 tab.



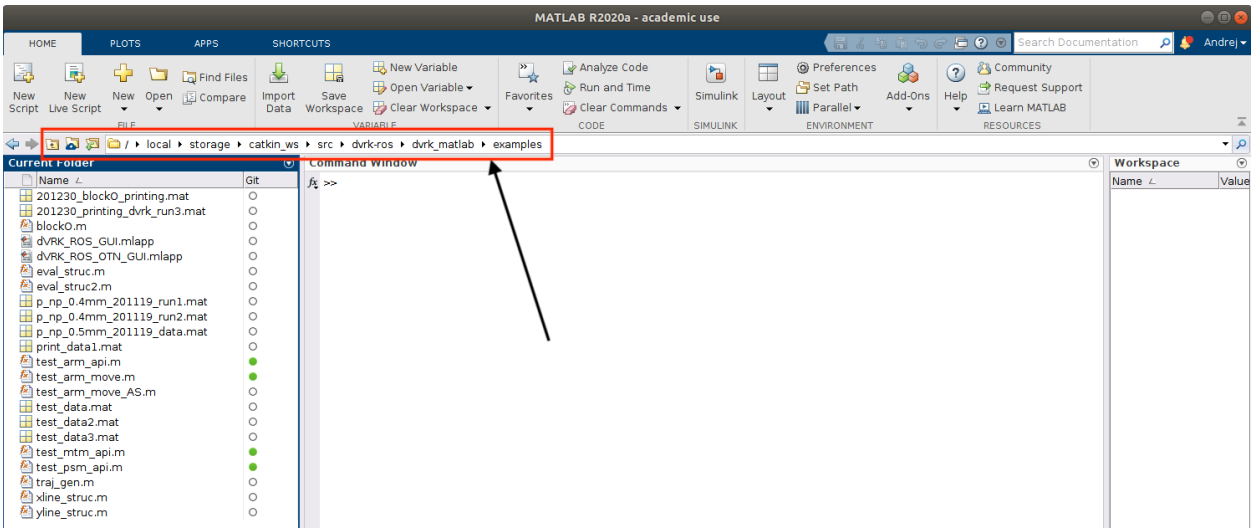
7. Manually retract the tool until the end effector is inside the cannula.

8. Press the `Home` button in the Qt Widget.



9. In the Matlab, navigate to the folder

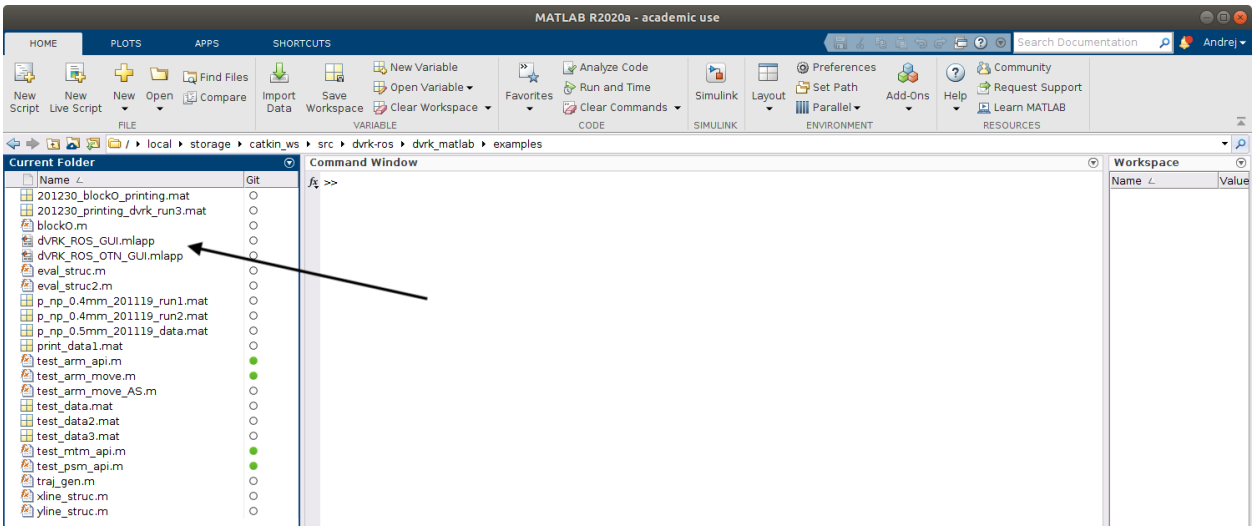
   `/local/storage/catkin_ws/src/dvrk-ros/dvrk_matlab/examples`



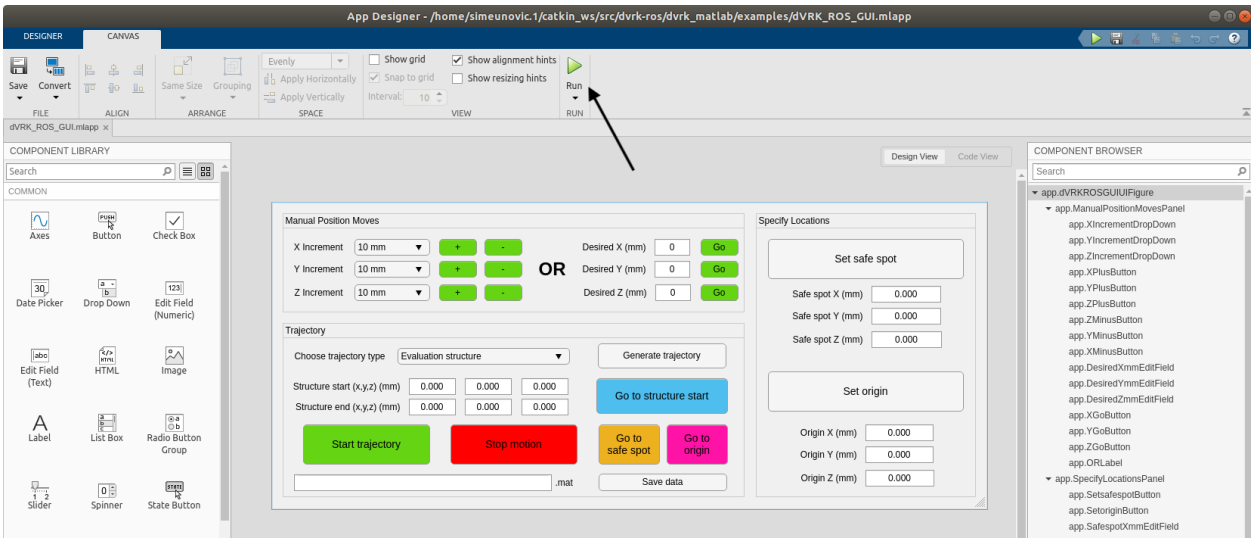
10. In Matlab open a GUI by double clicking the appropriate `.mlapp` file in the Current Folder list.

    Open `dVRK_ROS_GUI.mlapp` for normal operation.

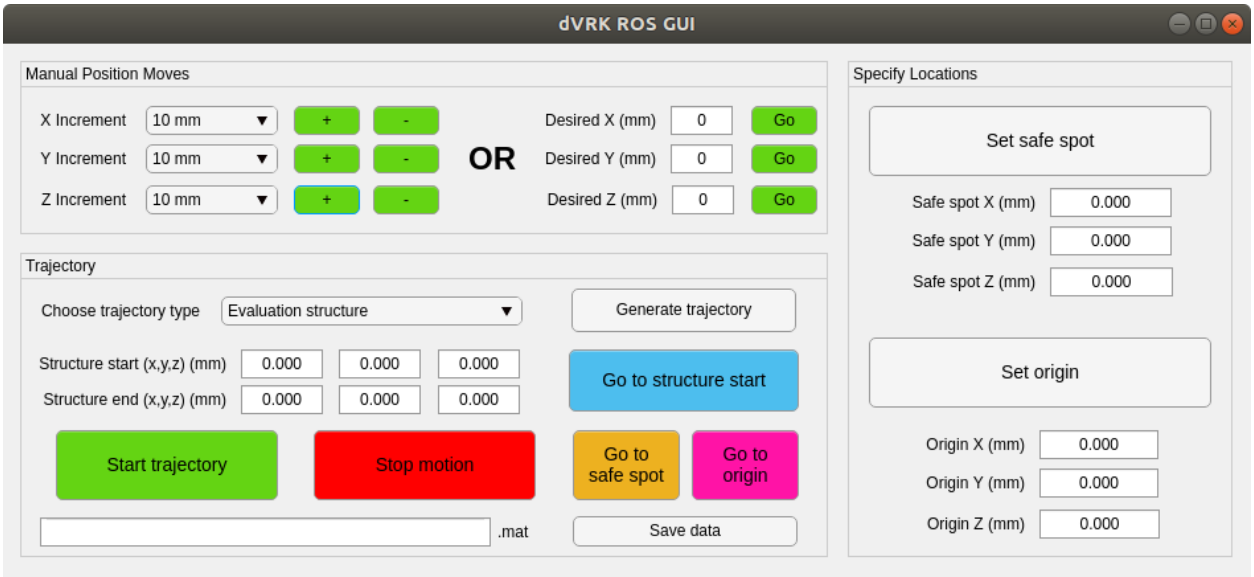
    Open `dVRK_ROS_OTN_GUI.mlapp` for printing.



11. In the App Designer window, click `Run` to launch the app.



12. You can control the system from either the Qt Widget or the Matlab GUI.



13. When done, close all applications except Terminator.
14. In the first terminal, kill the dVRK process by pressing 'Ctrl+C'.
15. In the second terminal, shutdown the ROS core by pressing 'Ctrl+C'.

4 FAQ

1. How do I get rid of an error related to powering the system?

   Open a new Terminator window.

   Type:

   ```
   $ qlacloserelays
   ```

2. How do I view the ROS network?

   Open a new Terminator window.

   Type:

   ```
   $ rosrun rqt_graph rqt_graph
   ```

3. How do I view the cameras?

   Open a new Terminator window.

   Type:

   ```
   $ pylonviewer
   ```