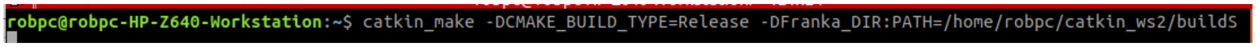
Git clone the tas_controllers into the **src** folder of the **catkin:**

```
Unset
git clone https://github.com/tasbolat1/tas_controllers.git
```

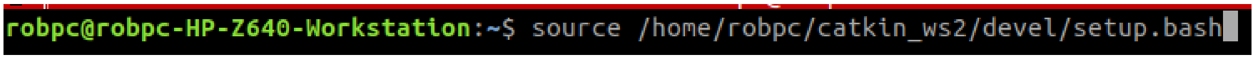
After that, we do the catkin make this way:

```
Unset
catkin_make -DCMAKE_BUILD_TYPE=Release
-DFranka_DIR:PATH=/home/user/catkin_ws/build

source /devel/setup.bash
```



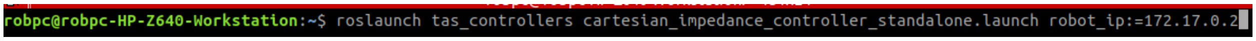
```
robpc@robpc-HP-Z640-Workstation:~$ catkin_make -DCMAKE_BUILD_TYPE=Release -DFranka_DIR:PATH=/home/robpc/catkin_ws2/buildS
```



```
robpc@robpc-HP-Z640-Workstation:~$ source /home/robpc/catkin_ws2/devel/setup.bash
```

After the catkin_make we can do the roslaunch

```
Unset
roslaunch tas_controllers cartesian_impedance_controller_standalone.launch
robot_ip:= <ip>
```



```
robpc@robpc-HP-Z640-Workstation:~$ roslaunch tas_controllers cartesian_impedance_controller_standalone.launch robot_ip:=172.17.0.2
```

Now lets break down the **cartesian_impedance_controller_standalone.launch**

Pass the robot’s IP address and the load the gripper

Unset

```
<arg name="robot_ip" />
<arg name="load_gripper" default="true" />
```

Include the Franka launch and pass the robot IP and the gripper loading to it

Unset

```
 <include file="$(find franka_control)/launch/franka_control.launch" >
    <arg name="robot_ip" value="$(arg robot_ip)" />
    <arg name="load_gripper" value="$(arg load_gripper)" />
```

Set the parameters that are defined in the.yaml file for the Franka joints

Unset

```
<rosparam command="load" file="$(find
tas_controllers)/config/tas_controllers.yaml" />
```

Nodes to launch:

First launch the controller_manager node type “spawner” to automatically load and start the controllers. Also, pass the .cpp file for the cartesian_impedance_controller

Unset

```
<node name="controller_spawner" pkg="controller_manager" type="spawner"
respawn="false" output="screen" args="cartesian_impedance_controller"/>
```

Launch Rviz, with marked Franka:

Unset

```
 <node pkg="rviz" type="rviz" output="screen" name="rviz" args="-d $(find
tas_controllers)/launch/rviz/franka_description_with_marker.rviz"/>
```

Launch the marker with which the user can interact with the manipulator, and fix it to the end-effector of the Franka:

Unset

```
 <node name="interactive_marker" pkg="tas_controllers"
type="interactive_marker.py" required="true" output="screen">
    <param name="link_name" value="panda_link0" />
  </node>
```

Launch the RQT to control the positional, angular and null position stiffness of the robot.

Unset

```
  <node name="rqt_reconfigure" pkg="rqt_reconfigure" type="rqt_reconfigure"
required="false" />
```

Cartesian_impedance_controller.launch:

First we define argument *stopped* and set to false

Unset

```
<arg name="stopped" default="false" />
```

Load the franka via the .yaml file

Unset

```
<rosparam command="load" file="$(find
tas_controllers)/config/tas_controllers.yaml" />
```

Next, we use the argument *stopped* to launch the node using the if statements, the first node will be launched unless it is stopped, thus it runs until the robot stops, and the second node will launch if it is stopped. “--stopped” sent to controller to define its state

Unset

```
<node unless="$(arg stopped)" name="controller_spawner_ci"
pkg="controller_manager" type="spawner" respawn="false" output="screen"
args="cartesian_impedance_controller"/>
  <node if="$(arg stopped)" name="controller_spawner_ci"
pkg="controller_manager" type="spawner" respawn="false" output="screen"
args="--stopped cartesian_impedance_controller"/>
```

Launch the RQT to control the positional, angular and null position stiffness of the robot:

Unset

```
<node name="rqt_reconfigure" pkg="rqt_reconfigure"
type="rqt_reconfigure" required="false" />
```