Move it configuration for the Arm and the Robot.

PART 1 - One time Configuration - This has to be done only once to the Files of the mobile Robot

Adding the controller in yaml

1. In the robot_sim/robot_control/config Folder add the controller to the list of controllers ie: summit_xl_sim/summit_xl_control/config

below the tag summit_xl:

```
arm_controller:
        type: position_controllers/JointTrajectoryController
        joints:
          - arm_shoulder_pan_joint
          - arm_shoulder_lift_joint
          - arm_elbow_joint
          - arm_wrist_1_joint
          - arm_wrist_2_joint
          - arm_wrist_3_joint
constraints:
       goal_time: 0.6
       stopped_velocity_tolerance: 0.05
       arm_shoulder_pan_joint: {trajectory: 0.1, goal: 0.1}
       arm_shoulder_lift_joint: {trajectory: 0.1, goal: 0.1}
       arm_elbow_joint: {trajectory: 0.1, goal: 0.1}
       arm_wrist_1_joint: {trajectory: 0.1, goal: 0.1}
       arm_wrist_2_joint: {trajectory: 0.1, goal: 0.1}
       arm_wrist_3_joint: {trajectory: 0.1, goal: 0.1}


stop_trajectory_duration: 0.5
state_publish_rate:  25
action_monitor_rate: 10
```

the type should be consistent with the the hardware interface mentioned in the So we have to change the transmission_interface in the urdf file of the robot. In our case we do it to the urdf of the UR_arm or a combined urdf generated for the robot and the arm together.

Changing the Hardware interface in URDF

2. In the robot_description/urdf folder check the transmission interface in the robot urdf file. If not consistent with the one being mentioned in the controller then change it. ie. summit_xl_description/robots/summit_xl_hls.urdf
or   universal_robot/ur_description/urdf

If it matches then its ok. Or either change it.

for
type: position_controllers/JointTrajectoryController
 <hardwareInterface>PositionJointInterface</hardwareInterface>

and for type: effort_controllers/JointEffortController
the interface shoud be
<hardwareInterface>EffortJointInterface</hardwareInterface>

Adding the controller in launch file
3. In the robot_sim/robot_control/launch/ robot_control.launch file add the controller in the <load the controllers> tag along with other controllers.
ie. summit_xl_sim/summit_xl_control/launch/ summit_xl_hls_control.launch

```
<!-- load the controllers -->
  <node name="controller_spawner" pkg="controller_manager" type="spawner"
respawn="false"
    output="screen" ns="/summit_xl" args="--namespace=/summit_xl
                    joint_blw_velocity_controller
                    joint_brw_velocity_controller
                    joint_frw_velocity_controller
                         .....................................
                    arm_controller(our controller)
```

PART 2 – Every time configuration – This configuration changes to the Moveit package have to be done every time a new Moveit configuration is generated.

1. Copy the Planning_Execution.launch file from the UR arm Moveit launc folder or make one and copy the following text into it.

```
<launch>
  <arg name="sim" default="false" />
  <arg name="limited" default="false"/>
  <arg name="debug" default="false" />

  <!-- Remap follow_joint_trajectory -->
  <remap if="$(arg sim)" from="follow_joint_trajectory"
to="summit_xl/arm_controller/follow_joint_trajectory"/><!-- -->


   <remap if="$(arg sim)" from="/joint_states" to="/summit_xl/joint_states" />

 <!-- Launch moveit -->
  <include file="$(find summit_moveit_config_2)/launch/move_group.launch">
    <arg name="limited" default="$(arg limited)"/>
    <arg name="debug" default="$(arg debug)" />
  </include>
</launch>
```

The line in Red is crucial to add since it publishes the current joint states. Other wise without the Arm will not work, in simualtion.

2. Change the controller manager.xml file generated by moveit in the Movit/launch. It will be empty and copy the text from the controller manager file from the UR Moveit.

```
<launch>
  <rosparam file="$(find summit_moveit_config)/config/controllers.yaml"/>
  <param name="use_controller_manager" value="false"/>
  <param name="trajectory_execution/execution_duration_monitoring"
value="false"/>
  <param name="moveit_controller_manager"
value="moveit_simple_controller_manager/MoveItSimpleControllerManager"/>
</launch>
```

3. Change planning_context.launch file in the Moveit/launch.

Either ...
 Add the text <arg name="limited" default="false"/> below the load robot and copy the following text

```
 <!-- Load universal robot description format (URDF) -->
  <group if="$(arg load_robot_description)">
    <param unless="$(arg limited)" name="$(arg robot_description)" command="$
(find xacro)/xacro.py '$(find
summit_xl_description)/urdf/robot/summit_xl_hls_omni.urdf.xacro'" />
    <param if="$(arg limited)" name="$(arg robot_description)" command="$(find
xacro)/xacro.py '$(find
summit_xl_description)/urdf/robot/summit_xl_hls_omni.urdf.xacro'" />
  </group>
```

OR

just set <arg name="load_robot_description" default="true"/>

4. Change move_group.launch in Moveit/launch.

Add the text

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

<include file="$(find summit_moveit_config)/launch/planning_context.launch" >
          <arg name="limited" value="$(arg limited)" />
 </include>
```

5. change the controller.yaml file in the Moveit/Config folder
Add the controller with the name of the robot and the controller. For exmp.

```
controller_list:
          - name: /summit_xl/arm_controller
```

```
      action_ns: follow_joint_trajectory
     type: FollowJointTrajectory
     joints:
      - arm_shoulder_pan_joint
      - arm_shoulder_lift_joint
      - arm_elbow_joint
      - arm_wrist_1_joint
      - arm_wrist_2_joint
      - arm_wrist_3_joint
```

Launching the Simulation

Now launch the following launch files in the sequence

Gazebo launch
1. roslaunch summit_xl_gazebo summit_xl_hls_omni.launch

Moveit Launch with the controllers
2. roslaunch summit_moveit_config summit_xl_moveit_planning_execution.launch sim:=true limited:=true

Moveit Rviz launch
2. roslaunch summit_moveit_config moveit_rviz.launch config:=true

Note:

We had a problem with our planning_execution.luanch file. The robot wrm did not move. We then came across this fix to publish the joint states to make the arm work in Gazeebo as mentioned in the link
(source: http://answers.ros.org/question/252114/failed-to-validate-trajectory-couldnt-receive-full-current-joint-state-within-1s-error/)

but the problem was that the robot arm was going back to its initial position again and again. ie. The position with which it was launched in gazeebo. It would go from the initial to the next goal planned from the Moveit Rviz. It would not start from the current pose as in Moveit Rviz, but go back to Pose zero and then start the move from that pose to the Goal.

We solved this problem by removing the above fix and then added the Tag to the planning_execution.launch file. This tag publishes the current joint states.

Adding the tag

```
<remap if="$(arg sim)" from="/joint_states" to="/summit_xl/joint_states" />
```

After this the robot starts in gazeebo from the current pose to the next goal.