Introducing MoveIt!

One of the challenging aspects of robotics is defining a path for the motion of a robot's arms to grasp an object, especially when obstacles may obstruct the most obvious path of motion. Fortunately, a ROS package called MoveIt! allows us to plan and execute a complicated trajectory.

A video created by Rethink Robotics shows how to use MoveIt! to plan the motion of Baxter's arms and then have MoveIt! actually cause a real or simulated Baxter to execute that motion. To see the video, go to:https://www.youtube.com/watch?feature=player_detailpage&v=1Zdkwym42P4.

A tutorial is available on the Rethink wiki site at http://sdk.rethinkrobotics.com/wiki/MoveIt_Tutorial.

First, start the Baxter simulator in Gazebo:

```
$ cd baxter_ws
$ ./baxter.sh sim
$ roslaunch baxter_gazebo baxter_world.launch
```

In a second terminal window, untuck Baxter's arms and start the Python script that starts `joint_trajectory_action_server`:

```
$ cd baxter_ws
$ ./baxter.sh sim
$ rosrun baxter_tools tuck_arms.py -u
$ rosrun baxter_interface joint_trajectory_action_server.py
```

The output on the screen should be as follows:

```
Initializing node...
Initializing joint trajectory action server...
Running. Ctrl-c to quit
```

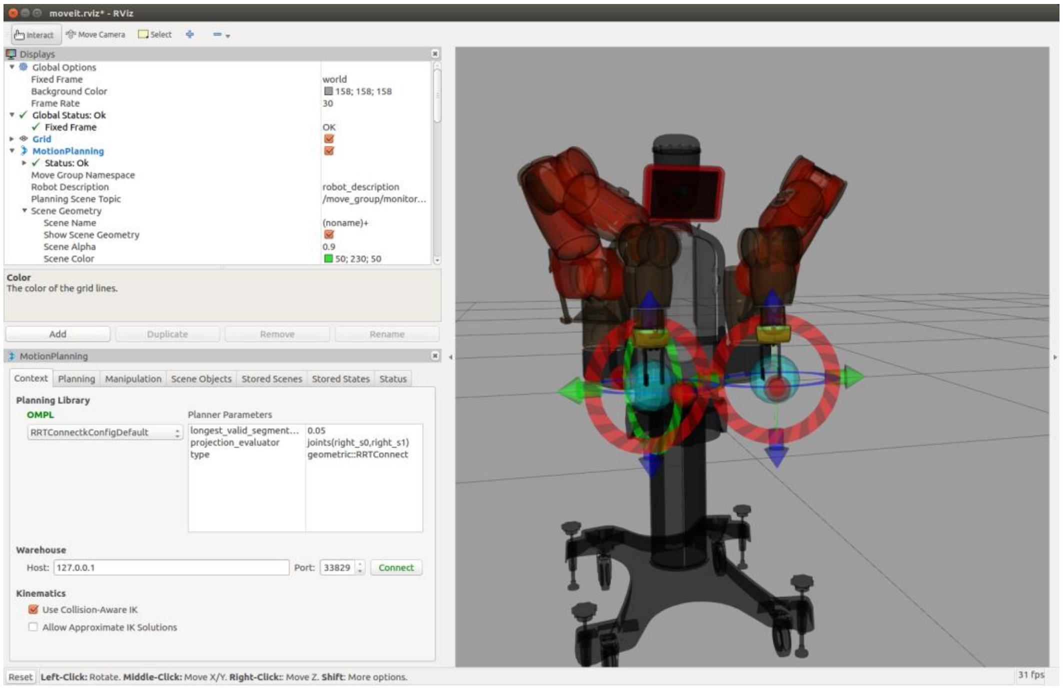
In a third Terminal, start MoveIt! and wait for the response:

```
$ cd baxter_ws
$ ./baxter.sh sim
$ roslaunch baxter_moveit_config baxter_grippers.launch
```

Look for the output:

```
. . .
You can start planning now!
```

Looking at the Gazebo window and the MoveIt! window, you'll see that Baxter looks the same in terms of the positions of itsarms:



Global Options
Fixed Frame: world
Background Color: 158; 158; 158
Frame Rate: 30
✓ Global Status: Ok
✓ Fixed Frame: OK
Grid: ☑
MotionPlanning: ☑
✓ Status: Ok
Move Group Namespace
Robot Description: robot_description
Planning Scene Topic: /move_group/monitor...
Scene Geometry
Scene Name: (noname)+
Show Scene Geometry: ☑
Scene Alpha: 0.9
Scene Color: 🟩 50; 230; 50

Color
The color of the grid lines.

Planning Library
OMPL
RRTConnectkConfigDefault

Planner Parameters

longest_valid_segment...	0.05
projection_evaluator	joints(right_s0,right_s1)
type	geometric::RRTConnect

Warehouse
Host: 127.0.0.1 Port: 33829

Kinematics
☑ Use Collision-Aware IK
☐ Allow Approximate IK Solutions

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click::** Move Z. **Shift**: More options.

MoveIt! startup

Insert Image B08463_06_18.png

In the screenshot, which is the rviz screen, the **Displays** and **Motion Planning** windows are shown on the left with the **Context** tab information showing. On the right, you can see the simulated Baxter in the starting position of MoveIt! with its arms untucked.

You can select any one of the **Displays** categories and modify the parameters. For example, the screenshot shows Baxter with a lightened **Background Color** chosen under **Global Options**.

Under the **MotionPlanning** panel, the Context/**Planning**/**Manipulation**/**Scene Objects**/**Stored Scenes**/**Stored States**/**Status**tabs are defined in the following table:

Tab	**Uses**
Context	Select the planning library and planner parameters; set collision awareness for IK solver
Planning	Set the start state, the goal state, and plan and execute moves of Baxter's arms
Manipulation	Object detection and manipulation
Scene Objects	Import or export scenes such as pillars or tabletops from a disk file or database, manipulate objects, and Publish Scene
Stored Scenes	Stored scenes on a database

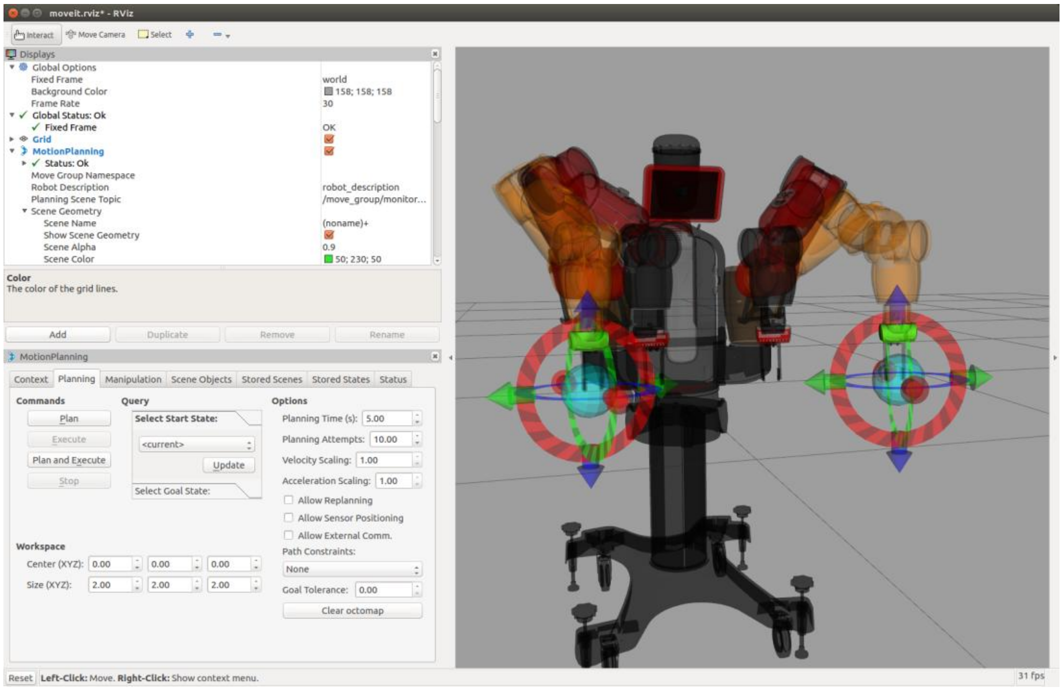
Stored States	Store and load robot states
Status	Status

Planning a move of Baxter's arms with MoveIt!

Click on the **Planning** tab under **MotionPlanning**. On the **Planning** panel, look for the **Query** field and the **Select Start State**heading. Click on **Select Start State**to reveal a menu box and set it to `<current>`. Then, click on the **Update** button.

Now to move Baxter’s arms:

1. Use arrows and rings to move Baxter's simulated arms to the desired positions. The desired goal positions should appear in orange in the simulation window.
2. Under the **Commands** area, choose the **Plan** button to see the trajectory of Baxter's arms in MoveIt!.
3. Choose the **Execute** button to see Baxter's arms move to the goal positions.
1. You should see red arms move from the start state to the final (goal) states:



Global Options
Fixed Frame: world
Background Color: 158; 158; 158
Frame Rate: 30
✓ Global Status: Ok
✓ Fixed Frame: OK
Grid: ☑
MotionPlanning: ☑
✓ Status: Ok
Move Group Namespace
Robot Description: robot_description
Planning Scene Topic: /move_group/monitor...
Scene Geometry
Scene Name: (noname)+
Show Scene Geometry: ☑
Scene Alpha: 0.9
Scene Color: 🟩 50; 230; 50

Color
The color of the grid lines.

Select Start State: <current>
Select Goal State:

Planning Time (s): 5.00
Planning Attempts: 10.00
Velocity Scaling: 1.00
Acceleration Scaling: 1.00
☐ Allow Replanning
☐ Allow Sensor Positioning
☐ Allow External Comm.
Path Constraints: None
Goal Tolerance: 0.00

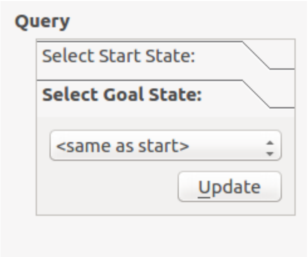
Center (XYZ): 0.00 0.00 0.00
Size (XYZ): 2.00 2.00 2.00

Left-Click: Move. **Right-Click:** Show context menu.

Baxter's arms and goal state for itsarms

Insert Image B08463_06_19.png

Next, have the arms move back to the original start positions to perform another move. To do this, under the **Query** field, click on **Select Goal State** to reveal the menu box and set it to `<same as start>`. Then, click on the **Update** button:



Query field to return arms to start positions

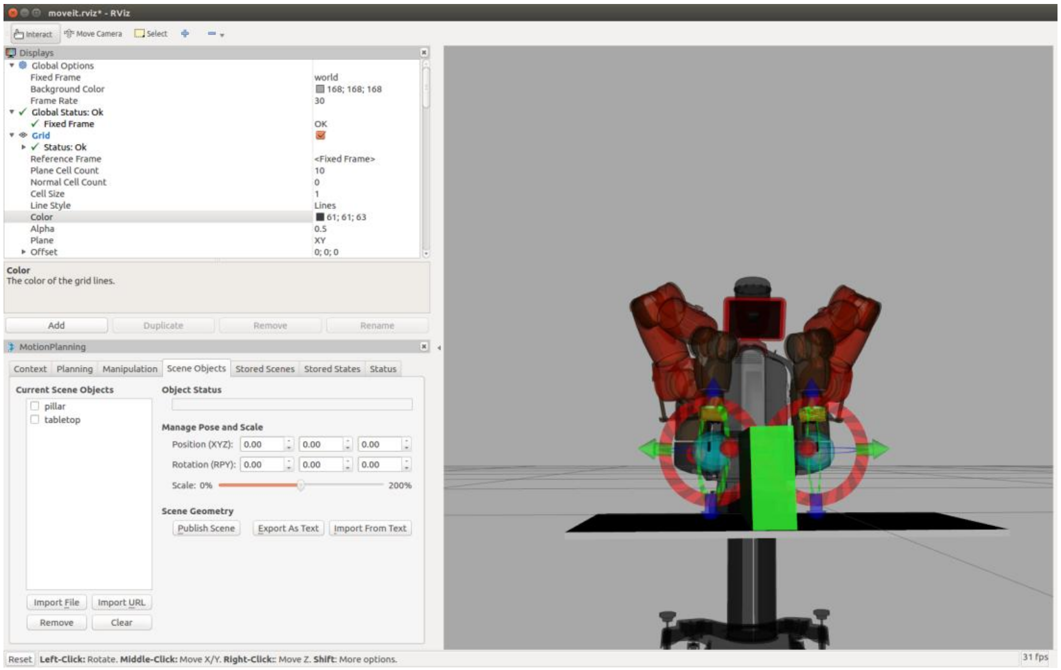
Insert Image B08463_06_20.png

Adding objects to a scene

Select the **Scene Objects** tab from the **MotionPlanning** frame. If you have scenes in your computer's directories, you can import them using the **Scene Geometry** field by selecting **Import From Text**. Alternatively, use the following scene, which we created in the `PillarTable.scene`file. Make sure that this file is saved as a text file.

```
(noname)+
* pillar
1
box
0.308 0.13056 0.6528
0.7 -0.01 0.03
0.0108439 0.706876 0.0103685 0.707178
0 0 0 0
* tabletop
1
box
0.7 1.3 0.02
0.7 0.04 -0.13
0 0 0 1
0.705882 0.705882 0.705882 1
.
```

The first line of dimensions under `box` in the file represents the height, width, and length of the pillar in meters. The next line defines the position from Baxter's origin. The third line is the pose of the pillar as a quaternion. The following screenshot shows the results of importing the scene elements:



```
▾ Global Options
    Fixed Frame            world
    Background Color       168; 168; 168
    Frame Rate             30
▾ ✓ Global Status: Ok
    ✓ Fixed Frame          OK
▾ Grid                     ☑
  ▸ ✓ Status: Ok
    Reference Frame        <Fixed Frame>
    Plane Cell Count       10
    Normal Cell Count      0
    Cell Size              1
    Line Style             Lines
    Color                  61; 61; 63
    Alpha                  0.5
    Plane                  XY
  ▸ Offset                 0; 0; 0
```

Color
The color of the grid lines.

Current Scene Objects
□ pillar
□ tabletop

Object Status

Manage Pose and Scale
Position (XYZ): 0.00 0.00 0.00
Rotation (RPY): 0.00 0.00 0.00
Scale: 0% 200%

Scene Geometry

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click:**: Move Z. **Shift**: More options.

Baxter with tabletop and pillar

Insert Image B08463_06_21

To manipulate the objects by moving or rotating them, select the object name (not the checkbox), and the arrows and rings should appear. Change their position with the green and blue arrows and rotate them with the ring. Moving the **Scale** slider will change the size of the object. Move the mouse to rotate the object, and roll the mouse wheel, if you have one, to zoom the object's size. You can save the scene (**Export As Text**) after you finish manipulating it.

Next, on the Scene Objects panel, click on the **Publish Scene** button under the **Scene Geometry** field. *This step is important and tells MoveIt! to plan around obstacles in the environment!*

Position of objects

When **pillar** is selected under **Current Scene Objects** (select the word, not the checkbox), values for its **Position (XYZ)** and **Rotation (RPY)** appear under **Manage Pose and Scale**. The position of *x*, *y*, and *z* of the centroid of the pillar is shown with respect to Baxter's origin. Baxter's *x*axis extends outward toward the viewer. The positive *y*axis is to the right in the view and the *z*axis runs upward. Note that the roll, pitch, and yaw are about the *x*, *y*, and *z*axes, respectively.

Planning a move to avoid obstacles with MoveIt!

In the followingexample, the left arm is going to move to the other side of the obstacle. MoveIt! will plan the trajectory so that Baxter's arm will not hit the pillar.

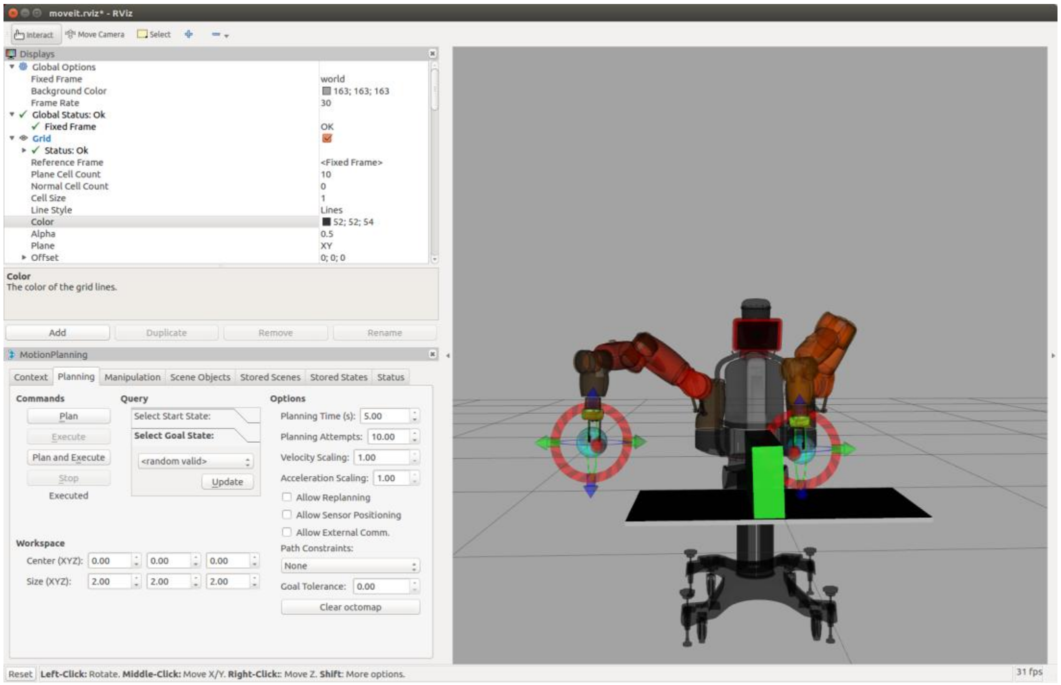
Return to the **Planning** tab on the **MotionPlanning** panel. First, move Baxter's right arm away from the pillar. **Plan**and thenExecute moving the right arm.

We can now drag our interactive markers for Baxter's left arm to move the goal state to a location on the opposite side of the pillar. Each time you click on the **Plan** button, a different arm trajectory path is shown on the virtual Baxter. Each path avoids collision with the pillar.

Caution!

We move Baxter's other arm (the right arm, in this case) out of the way to avoid any possible collisions. This is necessary if the MoveIt! trajectories are used on the real Baxter. Move the right arm with the markers and choose **Plan** and **Execute**.

The following screenshot shows Baxter prior to moving the left arm around the pillar. Notice that the right arm is moved out of the way:



▾ Global Options
Fixed Frame: world
Background Color: 163; 163; 163
Frame Rate: 30
▾ ✓ Global Status: Ok
✓ Fixed Frame: OK
▾ Grid: ☑
▸ ✓ Status: Ok
Reference Frame: <Fixed Frame>
Plane Cell Count: 10
Normal Cell Count: 0
Cell Size: 1
Line Style: Lines
Color: 52; 52; 54
Alpha: 0.5
Plane: XY
▸ Offset: 0; 0; 0

Color
The color of the grid lines.

Select Start State:
Select Goal State:
<random valid>

Executed

Planning Time (s): 5.00
Planning Attempts: 10.00
Velocity Scaling: 1.00
Acceleration Scaling: 1.00
□ Allow Replanning
□ Allow Sensor Positioning
□ Allow External Comm.
Path Constraints: None
Goal Tolerance: 0.00

Center (XYZ): 0.00 0.00 0.00
Size (XYZ): 2.00 2.00 2.00

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click::** Move Z. **Shift:** More options.

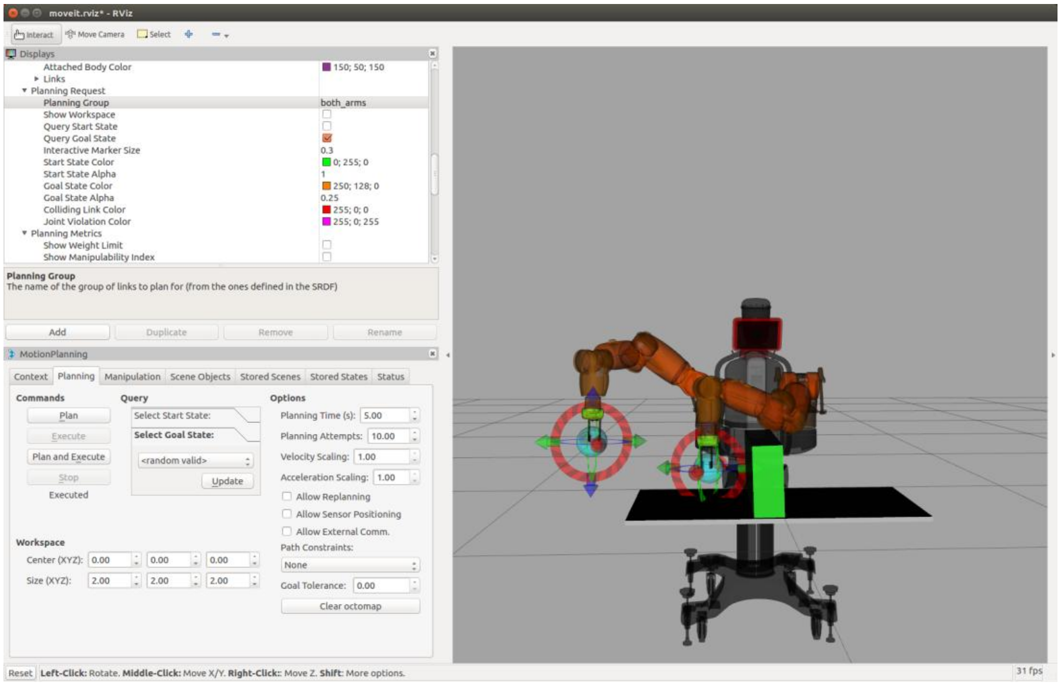
Baxter's right arm moved aside

Insert Image B08463_06_22.png

Caution!

When using MoveIt! with the real Baxter, the arms sometimes move into odd positions. If this happens, move them apart and restart MoveIt!.

Now, click on **Execute** to see Baxter's arm avoid the obstacle and move to the goal position on the other side of the pillar. The following screenshot shows that Baxter's left arm has moved around the pillar to the other side of it:



Attached Body Color	🟪 150; 50; 150
▸ Links	
▾ Planning Request	
Planning Group	both_arms
Show Workspace	□
Query Start State	□
Query Goal State	☑
Interactive Marker Size	0.3
Start State Color	🟩 0; 255; 0
Start State Alpha	1
Goal State Color	🟧 250; 128; 0
Goal State Alpha	0.25
Colliding Link Color	🟥 255; 0; 0
Joint Violation Color	🟪 255; 0; 255
▾ Planning Metrics	
Show Weight Limit	□
Show Manipulability Index	□

Planning Group
The name of the group of links to plan for (from the ones defined in the SRDF)

Commands

Executed

Query

Select Start State:

Select Goal State:

<random valid>

Options

Planning Time (s): 5.00

Planning Attempts: 10.00

Velocity Scaling: 1.00

Acceleration Scaling: 1.00

□ Allow Replanning

□ Allow Sensor Positioning

□ Allow External Comm.

Path Constraints: None

Goal Tolerance: 0.00

Workspace

Center (XYZ): 0.00 0.00 0.00

Size (XYZ): 2.00 2.00 2.00

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click::** Move Z. **Shift:** More options.

Baxter's simulated arm moved to the other side of the obstacle

Insert Image B08463_06_23.png