Introduction to Sensor Visualization with Rviz: Displaying Marker Lists

Wyatt Newman

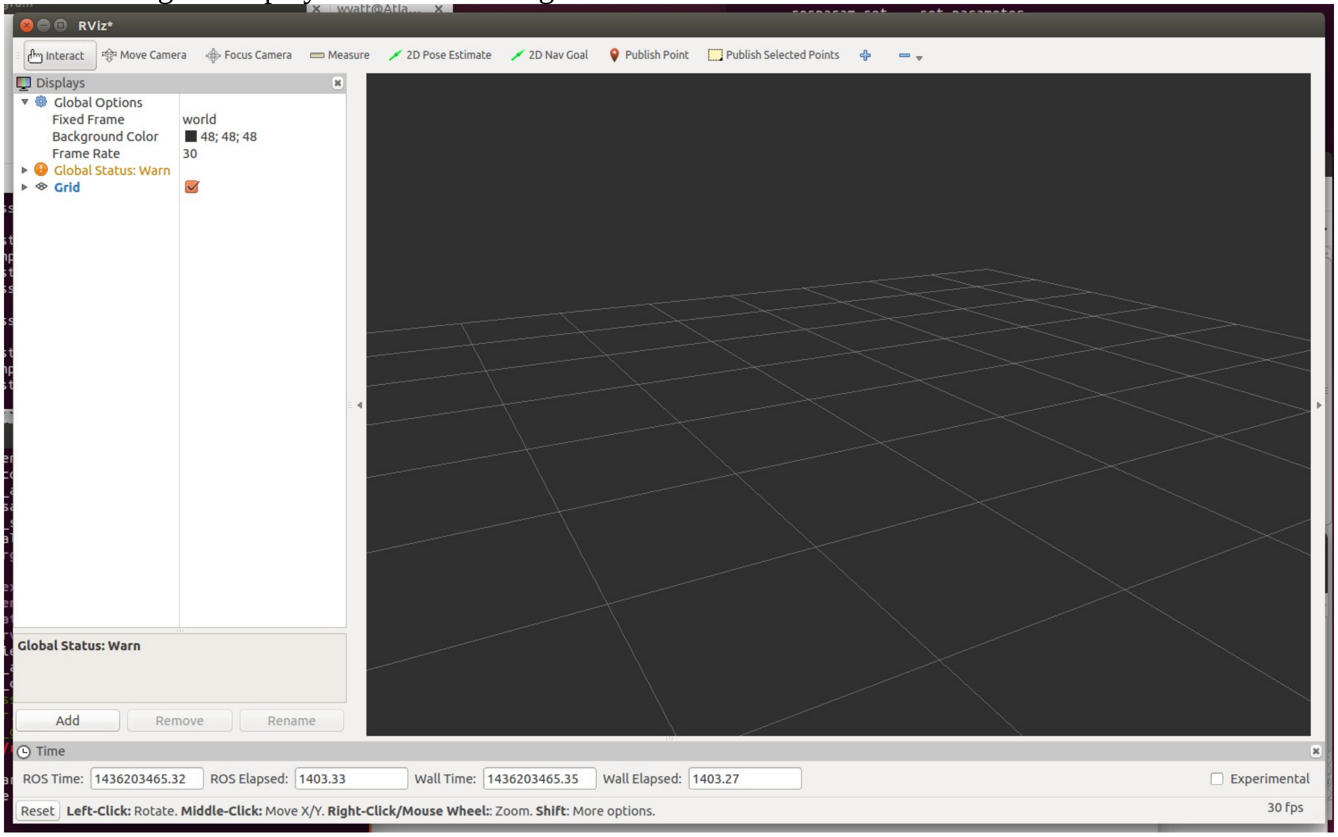
July, 2015

A valuable tool in ROS is “Rviz” (see http://wiki.ros.org/rviz). Rviz helps one to visualize sensor data in a variety of ways. Additionally, it can offer an interactive graphical user interface for valuable assistance in defining robot goals or assisting perceptual processing. Rviz has many capabilities, which will be introduced incrementally. This document starts with how to display “markers” in Rviz.

To start up Rviz, first make sure “roscore” is running. In another terminal, enter:

```
   rosrun rviz rviz
```

You should get a display like the following:



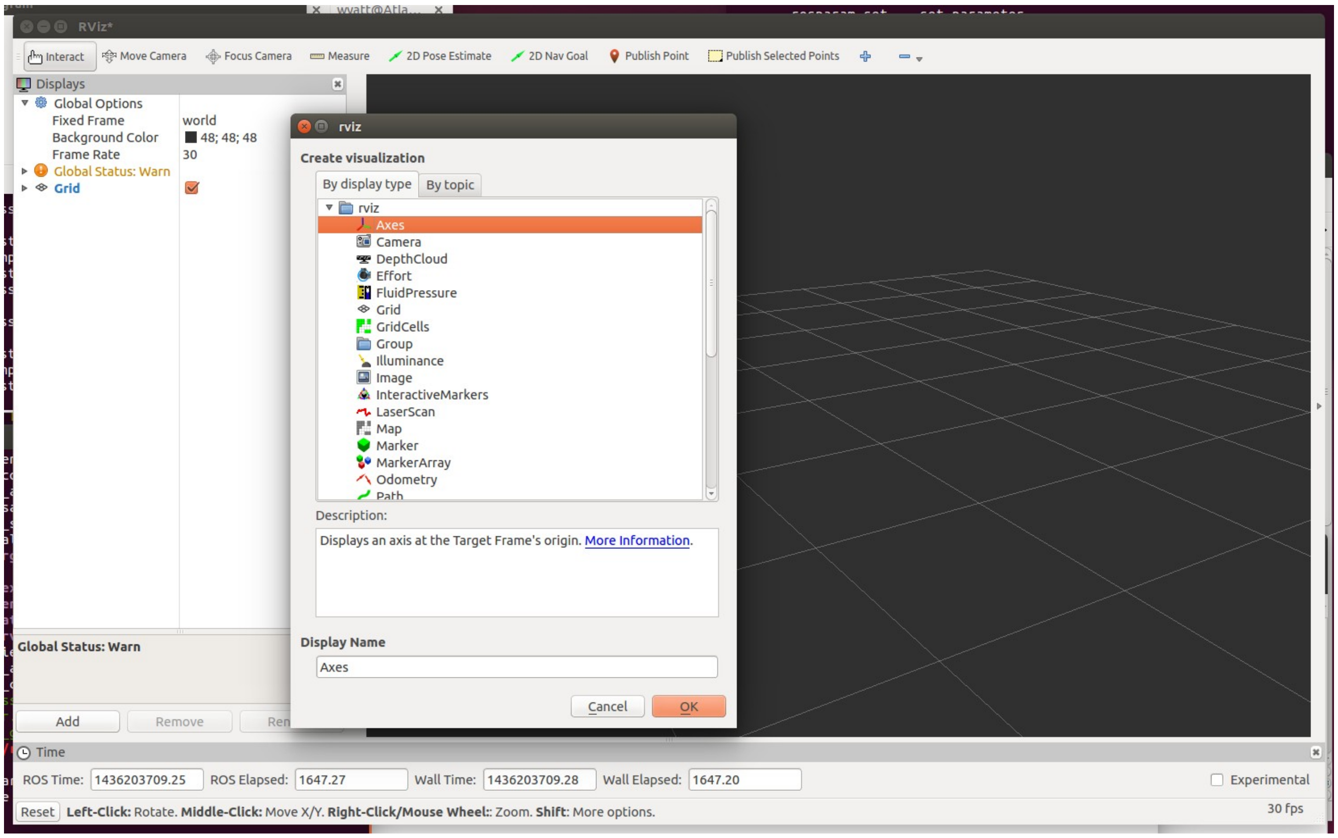
Fixed Frame: world
Background Color: 48; 48; 48
Frame Rate: 30
Global Status: Warn

ROS Time: 1436203465.32 ROS Elapsed: 1403.33 Wall Time: 1436203465.35 Wall Elapsed: 1403.27

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click/Mouse Wheel:**: Zoom. **Shift:** More options. 30 fps

The panel to the left in the above figure allows the user to select various parameters and topics to display. Beside the line “Fixed Frame”, type in: world. (For now, don't worry about the warning on “global status”).

At the lower left of this panel, click the button labelled “Add”, which will bring up a menu of options. Select “Axes”:



Global Options
Fixed Frame world
Background Color 48; 48; 48
Frame Rate 30
Global Status: Warn
Grid ✓

rviz: Axes, Camera, DepthCloud, Effort, FluidPressure, Grid, GridCells, Group, Illuminance, Image, InteractiveMarkers, LaserScan, Map, Marker, MarkerArray, Odometry, Path

Description:
Displays an axis at the Target Frame's origin. More Information.

Display Name
Axes

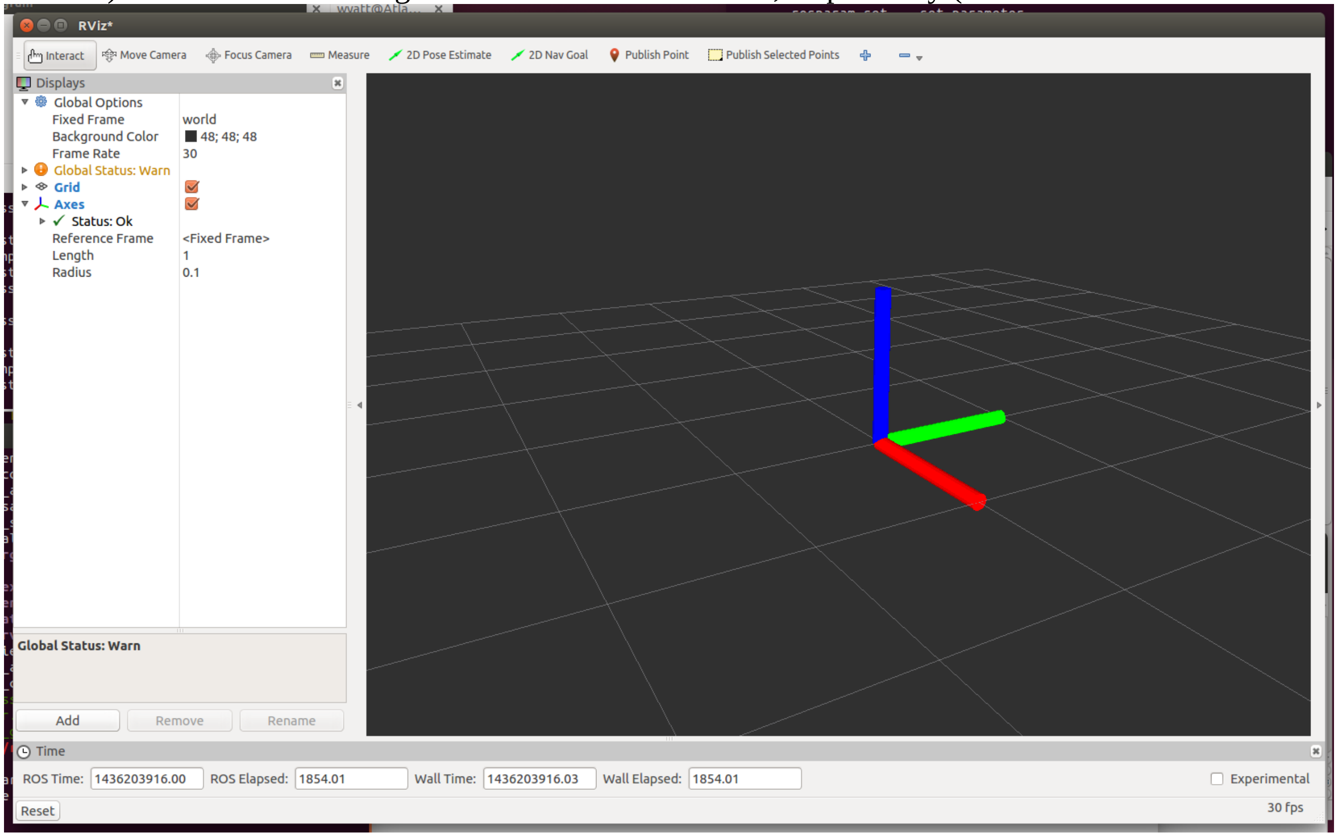
Global Status: Warn

ROS Time: 1436203709.25 ROS Elapsed: 1647.27 Wall Time: 1436203709.28 Wall Elapsed: 1647.20

Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel:: Zoom. Shift: More options.

30 fps

A new item, "Axes", will appear in the "Displays" list. Further details on "Axes" can be viewed and edited by expanding this item. The default axes display is "Fixed Frame" (which we previously set to "world"). The default axis length and radius are 1.0 and 0.1, respectively (which can be edited.



Global Options
Fixed Frame world
Background Color 48; 48; 48
Frame Rate 30
Global Status: Warn
Grid ✓
Axes ✓
Status: Ok
Reference Frame <Fixed Frame>
Length 1
Radius 0.1

Global Status: Warn

ROS Time: 1436203916.00 ROS Elapsed: 1854.01 Wall Time: 1436203916.03 Wall Elapsed: 1854.01

30 fps

The axes shown are the x, y, and z axis, shown as red, green and blue, respectively. (The axes will always be color coded in this order).

Publishing Markers to Rviz:
One can display “markers” in the Rviz view by publishing to appropriate topics. An example is in the class respository package “example_rviz_marker.” The source code of this node defines a service, called “rviz_marker_svc.” This service uses the generic “srv” message “cwru_srv/simple_float_service_message,” which expects the client to provide a single floating-point value. This service can be invoked manually, e.g., with the terminal command:

```
rosservice call rviz_marker_svc 1.0
```

When this service is invoked, the example_rviz_marker node will compute a grid of points within a horizontal plane at height 1.0 (as specified in the above example command).

This package lists dependency on “visualization_msgs” in the package.xml file. Correspondingly, the source code includes the header:

```
#include <visualization_msgs/Marker.h>
```

Within “main()” of “example_rviz_marker”, a publisher object is instantiated using this message type:

```
ros::Publisher vis_pub = nh.advertise<visualization_msgs::Marker>("example_marker_topic", 0);
```

which publishes to the chosen topic “example_marker_topic.”

The message type visualization_msgs::Marker is fairly complex, as can be seen by entering:

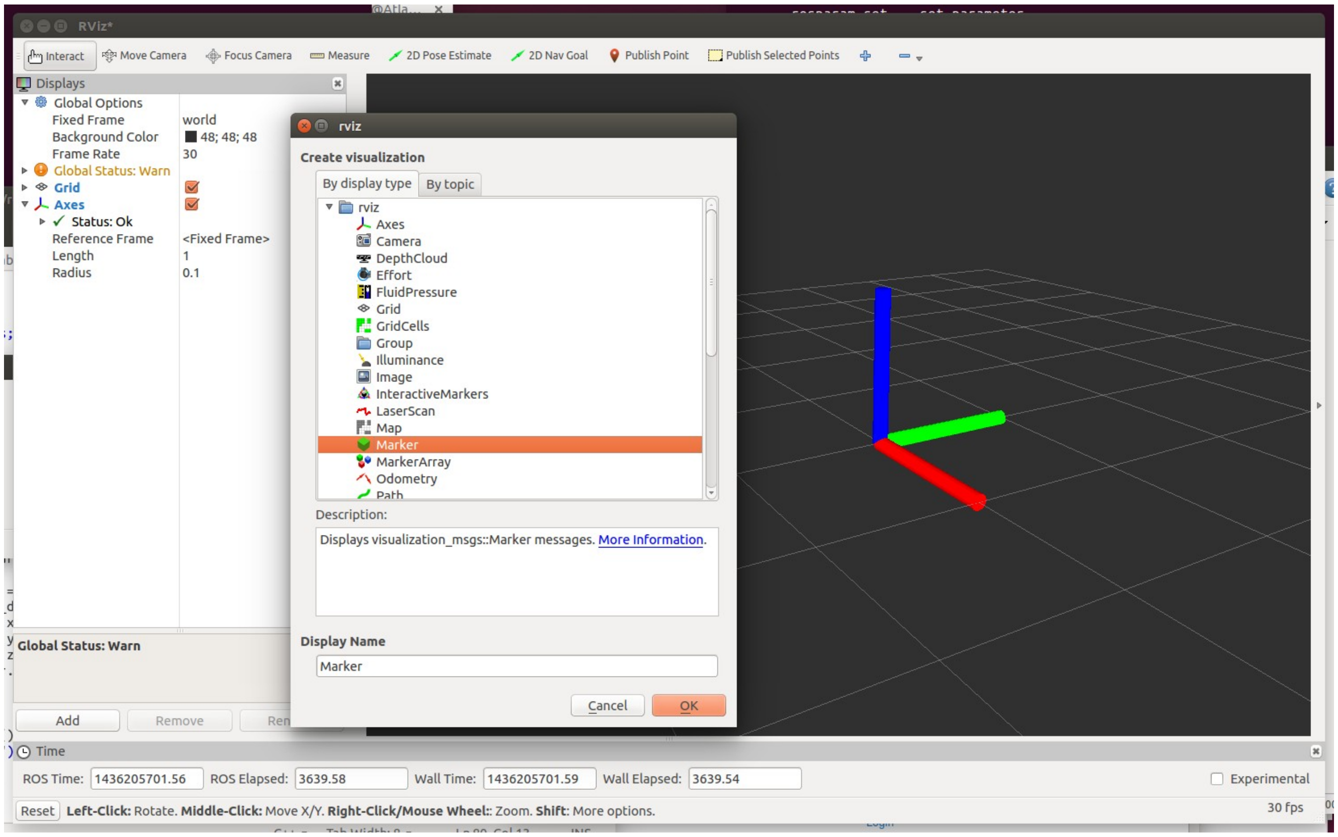
```
rosmsg show visualization_msgs/Marker
```

This message type contains 15 fields, many of which contain sub-fields. Additional details on definition and use of this message type can be found at: http://wiki.ros.org/visualization_msgs and http://wiki.ros.org/rviz/DisplayTypes/Marker. The example code populates the fields: header (including frame_id and time stamp), type, action, pose, scale, color and the vector of points (with x,y,z coordinates). In the main loop of this example node, the x,y,z coordinates of a list of points is populated and published, with the chosen shape, color and size held constant.

The example node is started running by entering:

```
rosrun example_rviz_mark example_rviz_marker
```

Upon start-up, this node will populate points (markers) to be displayed at elevation zero. Rviz will be able to display these markers graphically—after adding the display type and topic name. To do so, one must add an additional item, “Marker”, to the Rviz “Displays” list, as shown below (after clicking “Add” and choosing “Marker” from the pop-up options).



Global Options	
Fixed Frame	world
Background Color	48; 48; 48
Frame Rate	30
Global Status: Warn	
Grid	✓
Axes	✓
Status: Ok	
Reference Frame	<Fixed Frame>
Length	1
Radius	0.1

Global Status: Warn

rviz

Create visualization

By display type | By topic

rviz: Axes, Camera, DepthCloud, Effort, FluidPressure, Grid, GridCells, Group, Illuminance, Image, InteractiveMarkers, LaserScan, Map, Marker, MarkerArray, Odometry, Path

Description:

Displays visualization_msgs::Marker messages. More Information.

Display Name

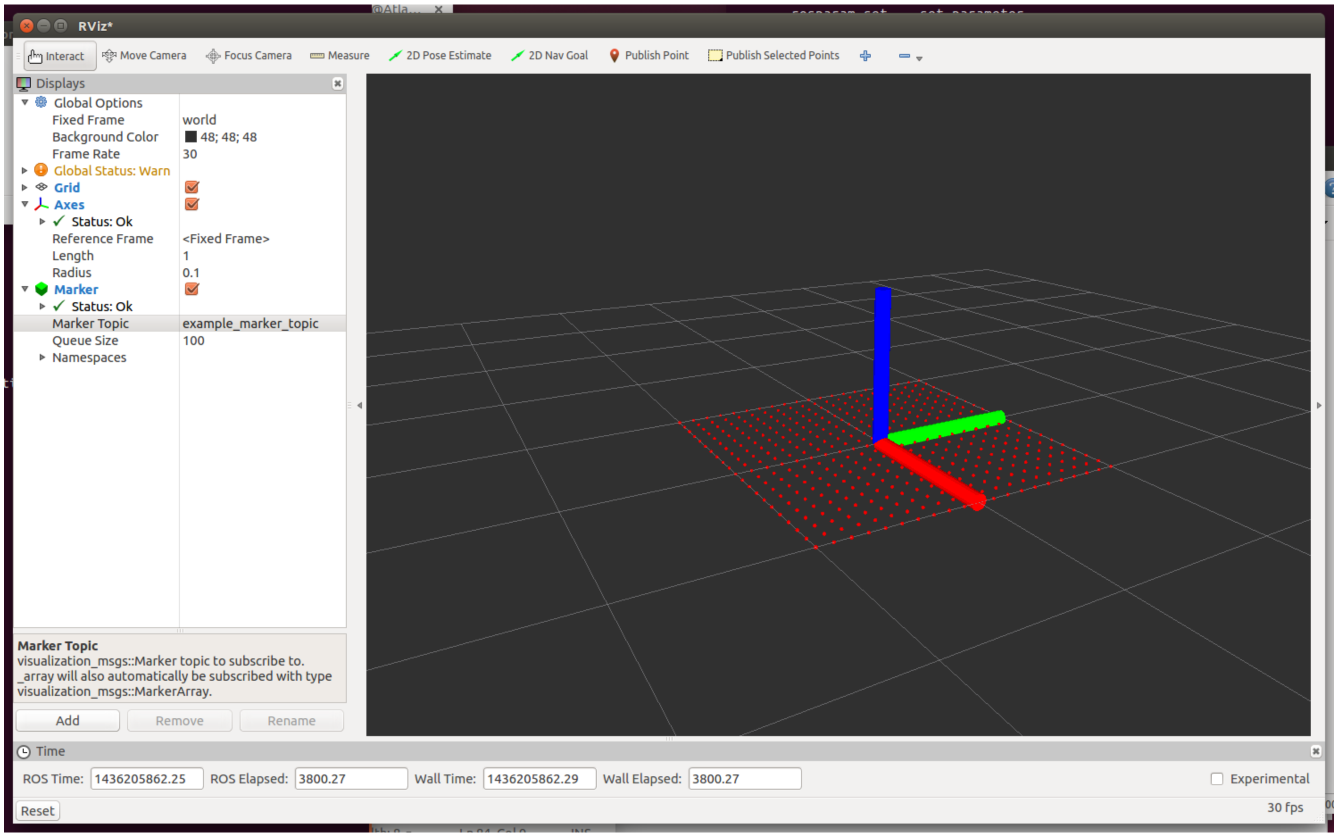
Marker

Cancel | OK

ROS Time: 1436205701.56 ROS Elapsed: 3639.58 Wall Time: 1436205701.59 Wall Elapsed: 3639.54 Experimental

Reset **Left-Click:** Rotate. **Middle-Click:** Move X/Y. **Right-Click/Mouse Wheel:**: Zoom. **Shift**: More options. 30 fps

After clicking “OK”, the display list will include the item “Marker.” Clicking on this item to expand it, one has the option of editing the topic to which this display should subscribe.



Global Options	
Fixed Frame	world
Background Color	48; 48; 48
Frame Rate	30
Global Status: Warn	
Grid	✓
Axes	✓
Status: Ok	
Reference Frame	<Fixed Frame>
Length	1
Radius	0.1
Marker	✓
Status: Ok	
Marker Topic	example_marker_topic
Queue Size	100
Namespaces	

Marker Topic
visualization_msgs::Marker topic to subscribe to. _array will also automatically be subscribed with type visualization_msgs::MarkerArray.

ROS Time: 1436205862.25 ROS Elapsed: 3800.27 Wall Time: 1436205862.29 Wall Elapsed: 3800.27 Experimental

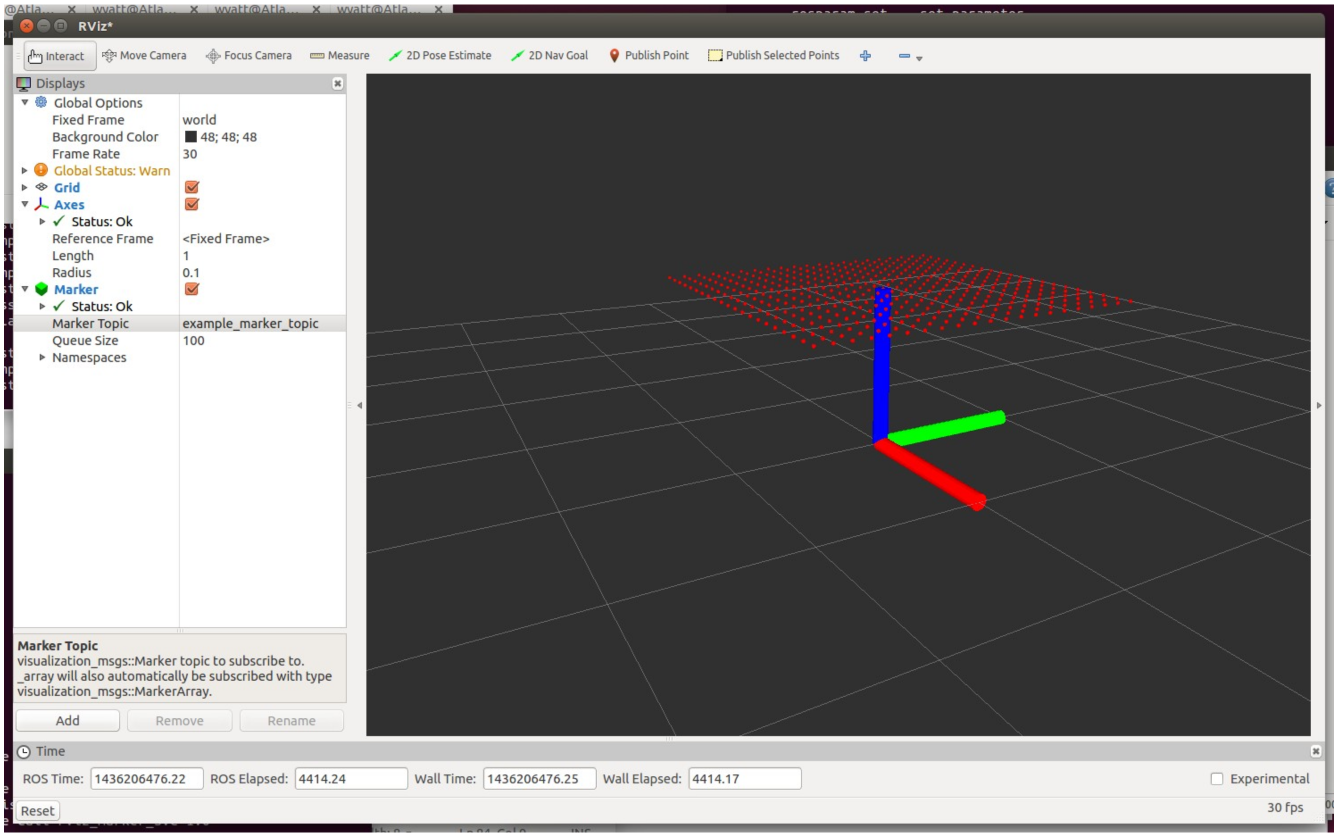
Reset 30 fps

As shown above, the topic should be edited by typing in the text “example_marker_topic”, which is the topic name chosen in our source code for “example_rviz_marker.” At this point, Rviz starts receiving messages from our example_rviz_marker node, resulting in a display of red spheres within a planar patch at height zero about the origin.

The markers will appear at height 1.0 if the example_rviz_marker node responds to a service request of:

```
        rosservice call rviz_marker_svc 1.0
```

as shown below:



Global Options	
Fixed Frame	world
Background Color	■ 48; 48; 48
Frame Rate	30
Global Status: Warn	
Grid	✓
Axes	✓
Status: Ok	
Reference Frame	<Fixed Frame>
Length	1
Radius	0.1
Marker	✓
Status: Ok	
Marker Topic	example_marker_topic
Queue Size	100
Namespaces	

Marker Topic
visualization_msgs::Marker topic to subscribe to. _array will also automatically be subscribed with type visualization_msgs::MarkerArray.

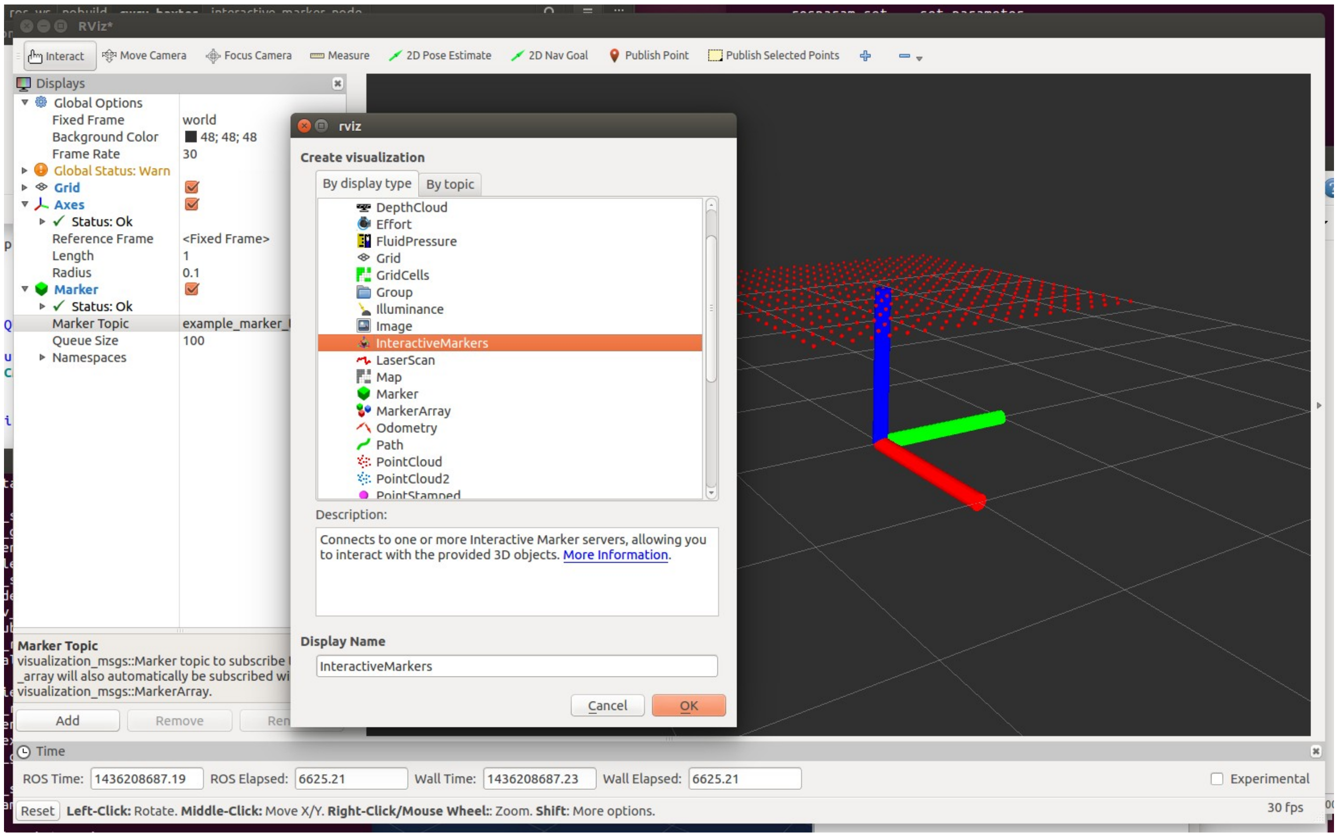
ROS Time: 1436206476.22 ROS Elapsed: 4414.24 Wall Time: 1436206476.25 Wall Elapsed: 4414.17 Experimental

Ordinarily, the service client would originate from another node—but manually stimulating the service from the command line is useful for incremental testing.

Interactive Markers:
Another very useful display within Rviz is the “interactive marker” (see http://wiki.ros.org/rviz/Tutorials/InteractiveMarkers:GettingStarted). Example code for using interactive markers is in the class repository package “example_interactive_marker.” With roscore and rviz running (optionally, additional nodes, such as example_rviz_marker), start this node with:

```
        rosrun  example_interactive_marker interactive_marker_node
```

In rviz, one must add a display and enter the appropriate topic name to visualize the interactive marker. To do so, in rviz click “Add” and choose the item “Interactive Marker”, as shown below:



Global Options
Fixed Frame world
Background Color 48; 48; 48
Frame Rate 30
Global Status: Warn
Grid
Axes
Status: Ok
Reference Frame <Fixed Frame>
Length 1
Radius 0.1
Marker
Status: Ok
Marker Topic example_marker_
Queue Size 100
Namespaces

Description:
Connects to one or more Interactive Marker servers, allowing you to interact with the provided 3D objects. More Information.

Display Name
InteractiveMarkers

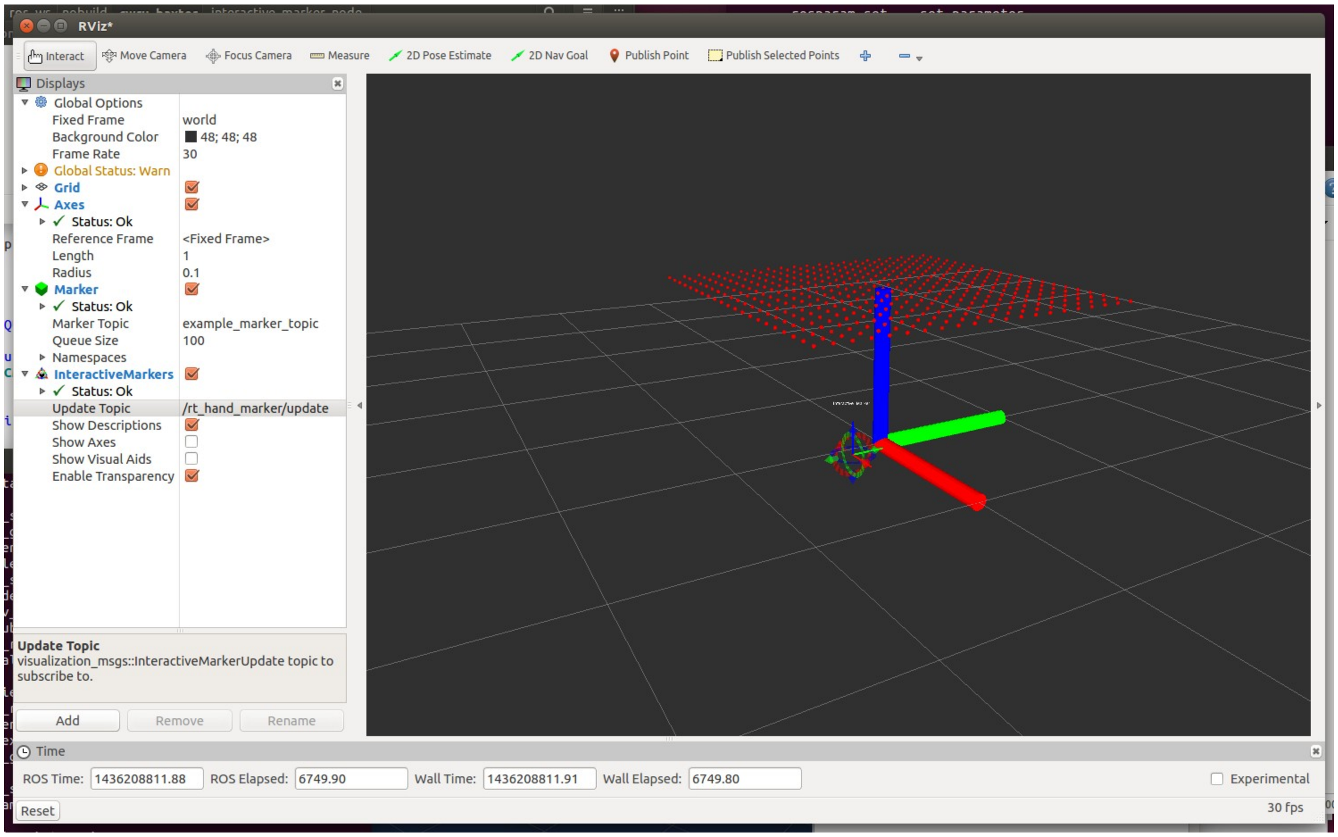
Marker Topic
visualization_msgs::Marker topic to subscribe
_array will also automatically be subscribed wi
visualization_msgs::MarkerArray.

ROS Time: 1436208687.19 ROS Elapsed: 6625.21 Wall Time: 1436208687.23 Wall Elapsed: 6625.21

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

30 fps

Expand the new InteractiveMarker item and edit the topic to: rt_hand_marker/update.



Global Options
Fixed Frame world
Background Color 48; 48; 48
Frame Rate 30
Global Status: Warn
Grid
Axes
Status: Ok
Reference Frame <Fixed Frame>
Length 1
Radius 0.1
Marker
Status: Ok
Marker Topic example_marker_topic
Queue Size 100
Namespaces
InteractiveMarkers
Status: Ok
Update Topic /rt_hand_marker/update
Show Descriptions ✓
Show Axes
Show Visual Aids
Enable Transparency ✓

Update Topic
visualization_msgs::InteractiveMarkerUpdate topic to subscribe to.

ROS Time: 1436208811.88 ROS Elapsed: 6749.90 Wall Time: 1436208811.91 Wall Elapsed: 6749.80

30 fps

As shown above, a new display will appear. This display has 9 interactive handles for moving the marker +/- x, y, z, and +/- rotation about x, y and z. In Rviz, if one hovers the mouse over one of these controls, the color of the handle will bolden, and by clicking and dragging, the marker will change its displacement or orientation. As the marker is moved interactively, its “pose” (6-D position and orientation) in space changes. The new values are published, invoking a callback response from the function:

```
processFeedback(const visualization_msgs::InteractiveMarkerFeedbackConstPtr &feedback)
```

The above function depends on the package: visualization_markers, which requires including the header:

```
#include <interactive_markers/interactive_marker_server.h>
```

In this package, the source code “IM_6dof.cpp” offers a service that accepts two request modes: one to request the current pose of the marker, and the other to request moving the marker to a specified pose. A custom service message is defined for this interaction in the cwru_srv package.

When moving the marker interactively (with a mouse), new poses of the marker are updated. However, this publication is silent with the marker is not being moved. The code in “IM_6dof.cpp” (with executable named “interactive_marker_node”), however, keeps track of all marker publications and retains the most recently published value. Upon request (via the service “IM6DofSvc”), this node will provide the current pose of the interactive marker (in the “response” field of the service request).

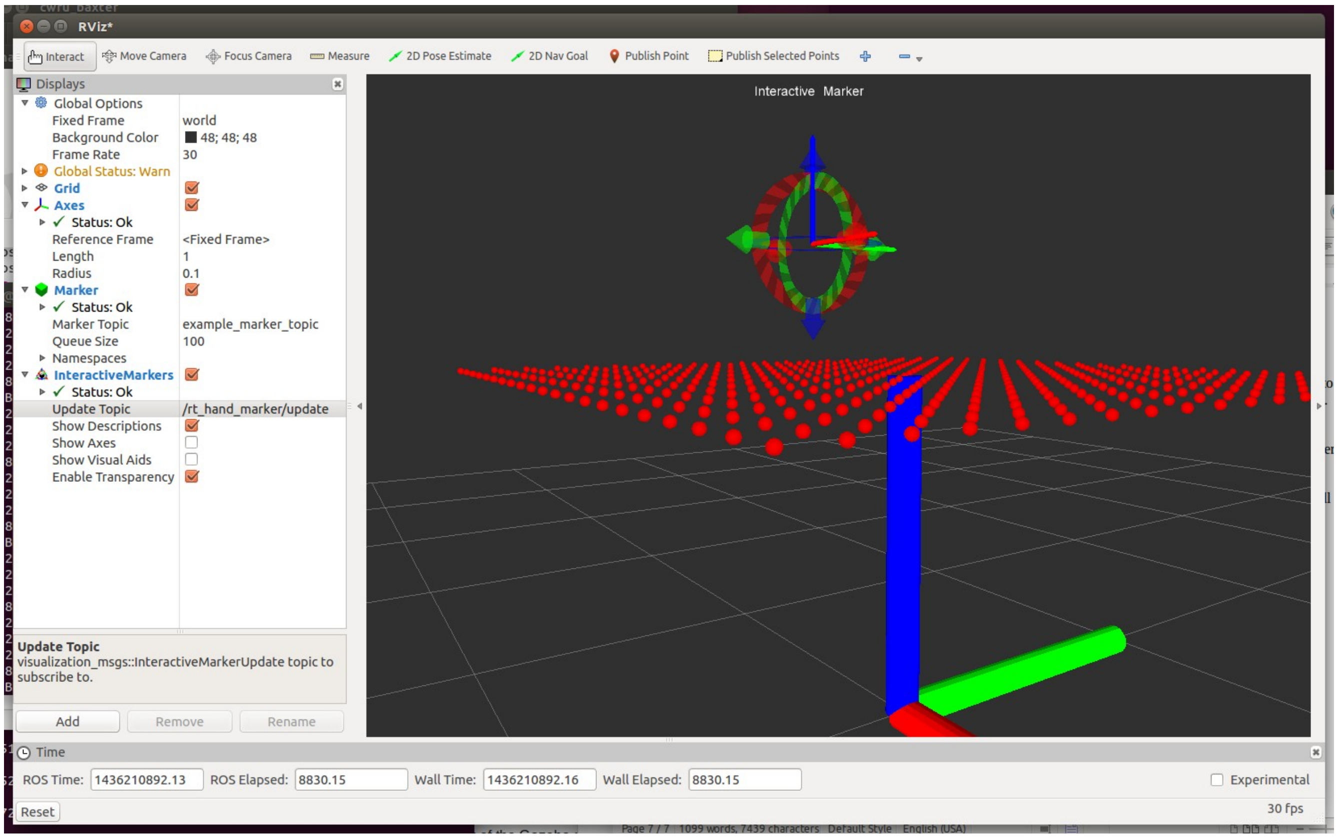
Alternatively, one can request that the marker be moved to a new pose programmatically. The same service, “IM6DofSvc”, will respond to a request command mode “1” to move the marker to the pose specified in the request field of the service call.

Use of the service IM6DofSvc is illustrated with the example node “im_6dof_svc_client_test”. The source code, IM_6dof_svc_client_test.cpp, instantiates a client of IM6DofSvc. This test routine first requests the current pose of the marker. It then adds 0.1m to the z-component of the marker origin, then requests of IM6DofSvc that the marker be moved to this new pose.

The screenshot below shows a view from rviz after the test node has been run multiple times. As a result of multiple marker-move requests from the test node, the marker elevation has increased to reside above the plane of markers at z=1.0m.

Conclusion: Markers and Interactive Markers are two very useful features of Rviz. Markers can be used to illustrate reachability (or “affordances”) or to illustrate results of perceptual processing from sensors. Interactive markers can be used to display computed candidate robot tool poses, or such markers can be moved interactively to command desired poses, as viewed by the user.

The value of these capabilities becomes more apparent once Rviz also displays sensor values and a robot model.



Global Options	
Fixed Frame	world
Background Color	■ 48; 48; 48
Frame Rate	30
Global Status: Warn	
Grid	✓
Axes	✓
Status: Ok	
Reference Frame	<Fixed Frame>
Length	1
Radius	0.1
Marker	✓
Status: Ok	
Marker Topic	example_marker_topic
Queue Size	100
Namespaces	
InteractiveMarkers	✓
Status: Ok	
Update Topic	/rt_hand_marker/update
Show Descriptions	✓
Show Axes	☐
Show Visual Aids	☐
Enable Transparency	✓

Update Topic
visualization_msgs::InteractiveMarkerUpdate topic to subscribe to.

ROS Time: 1436210892.13 ROS Elapsed: 8830.15 Wall Time: 1436210892.16 Wall Elapsed: 8830.15 ☐ Experimental

30 fps