User manual

sig_ros package

June 8, 2015

Contents

1 Generalities **3**
1.1 Goal 3
1.2 For who? 3
1.3 Install 3

2 Usage **5**

3 Topics **7**

4 Services **11**

5 FAQ **13**

Bibliography **14**

1 Generalities

1.1 Goal

This package aim to provide a tool for using SIGVerse[1] though ROS without knowledge of SIGVerse or limited knowledge.
Using sig_ros package will allow you to send topics and call services directly to SIGVerse.

1.2 For who?

This package is intended for ROS users or SIGVerse users who want to use SIGVerse in a different way.
For using this package you previously need basic knowledge of ROS, that means at least the beginner level of the ROS tutorials page[4], running a node, publishing and subscribing to a topic, calling a service...is the minimum requiered.

1.3 Install

First of all, you have to install SIGServer[2] and SIGViewer[3] like explained in the SIGVerse wiki page[1].

Create a catkin workspace:

```
mkdir -p ~/catkin_ws/src
```

Initialize the workspace:

```
cd ~/catkin_ws/src
catkin_init_workspace
cd ..
catkin_make
source devel/setup.bash
```

Clone the git repository:

```
git clone https://github.com/GG31/sig_ros.git
```

Change the name of sig_ros folder you've just cloned by src, so you have the tree:

```
|-- catkin_ws
    |-- src
```

```
        |-- sig_ros
        |-- user
    |-- devel
    |-- build
```

Change the absolute links on `catkin_ws/src/user/xml/CleanUpDemo2014.xml` **there is 5, on** `catkin_ws/src/sig_ros/src/ros_controller.cpp` **there is one and on** `catkin_ws/src/sig_ros/CMakeLists.txt`
Create libsig_ros:

```
mkdir ~/catkin_ws/devel/lib/libsig_ros
```

2 Usage

The repository `https://github.com/GG31/sig_ros.git` contains two package sig_ros and user. sig_ros is the package who make the interface between SIGVerse and ROS and user is an example of package who contains severals nodes. These nodes send messages and call services who reproduce the clean up task demo.
On the directory `~/catkin_ws/src/user/xml` there are the all xml file needed by the clean up task.

Go to the directory `~/catkin_ws/src/user/xml` and run the ros_controller node of the sig_ros package with:

```
cd ~/catkin_ws/src/user/xml
rosrun sig_ros ros_controller CleanUpDemo2014Robot.xml
```

The SIGServer is launched automatically and you and see the number of the port.
Find the IP address with ifconfig.
Then open the SIGViewer and write the IP adress and the port. Click on "Connect". It is the step 1 in the figure 2.1.
After that, you can see the world defined by the xml files, if the camera is not well positionned, do not hesitate to move it with the mouse and the keys Ctrl, Alt and Maj.

Start the simulation, the all topics and services are created at the same time. This is the step 2 in figure 2.1. After that, you will be able to publish, subscribe and call a service.
You can see figure 2.1 a sum up of the three steps. During the third step you can create all the node you want and communicate with SIGVerse.
For example, in the package user, there are severals node which can be started, "RobotCommand", "ModeratorCommand",...
Start the "RobotCommand" node.

```
rosrun user RobotCommand
```

The robot will begin to move.
If you start the service "Referee" and the "ModeratorCommand" node, the score will be counted.

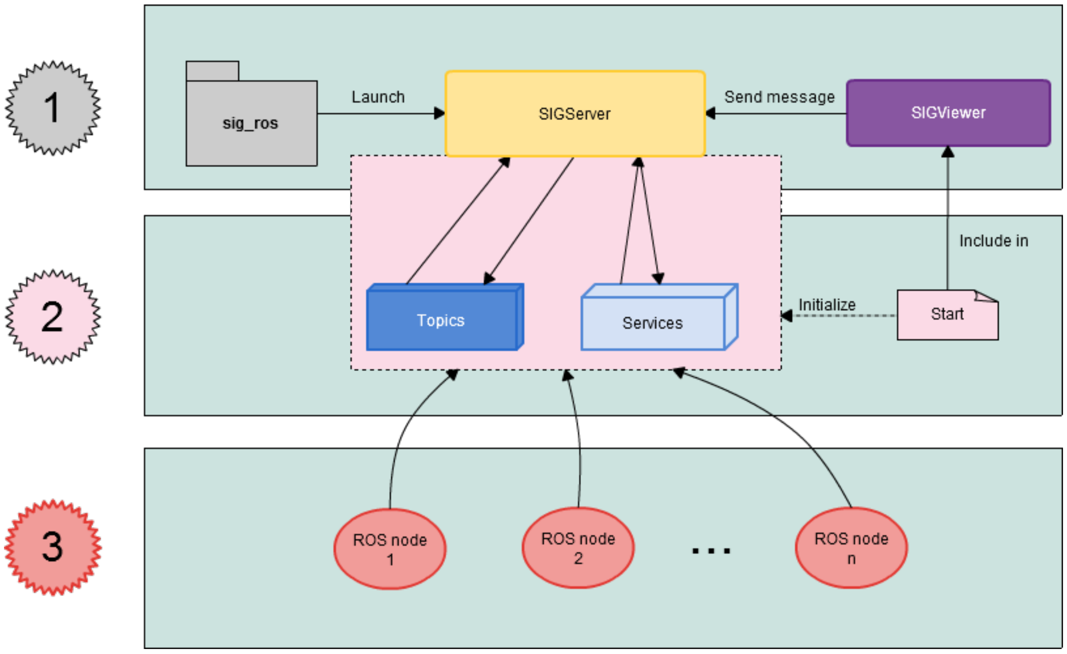

Figure 2.1: Usage of the package

3 Topics

For all the topics, if there is a parameters called "name", that means it refers to an entity. For example, if we have the topic robot_000_setPosition if the parameter "name" is filled by "trashbox_0", the topic will set the position to the trashbox, but if the parameter "name" is an empty string, then it will be the "robot_000". For the services, the parameter "name" works as well.

Topic name	Message	Description
_onRecvMsg	**sender** : string **content** : string	The "Controller" send the message received by the SIGViewer.
_onCollisionMsg	**name** : string **part** : string	The name of the agent which one is in collision with are sent to this topic. If there is severals collision at the same time, severals messages are sent.
_setWheel	**wheelRadius** : double **wheelDistance** : double	Publish the radius and the distance in a message and they will be applied to the robot.
_setWheelVelocity	**leftWheel** : double **rightWheel** : double	Publish the velocity for the left and the right wheel and it will be applied.
_setJointVelocity	**jointName** : string **angularVelocity** : double **max** : double	jointName, angular velocity, max ???
_releaseObj	**arm** : string	Publish the part which you want to release an object and it will be done.
_setAxisAndAngle	**name** : string **axisX** : double **axisY** : double **axisZ** : double **angle** : double	Set the axis defined by "axisX", "axisY" and "axisZ" and set the angle "angle" to the entity called "name", if no name is provided, the main entity of the topic will be set.
_setPosition	**name** : string **posX** : double **posY** : double **posZ** : double	Set the position "posX", "posY" and "posZ" to the entity called "name", if no name is provided, the main entity of the topic will be set.
_setAccel	**name** : string **x** : double **y** : double **z** : double	Set the acceleration to the entity

_setAngularVelocity	**name** : string **x** : double **y** : double **z** : double	Set angular velocity to the entity name (only in Dynamics ON)
_setTorque	**name** : string **x** : double **y** : double **z** : double	Set the torque.
_setVelocity	**name** : string **x** : double **y** : double **z** : double	Set Velocity to the entity.
_setCollisionEnable	**name** : string **flag** : boolean	Set if the collision is enable, true, false otherwise.
_setGravityMode	**name** : string **boolean** : boolean	Set the gravity mode, true if enable, false otherwise.
_setJointAngle	**name** : string **jointName** : string **angle** : double	Set the angle of the joint (only in Dynamics OFF).
_setJointQuaternion	**name** : string **jointName** : string **qW** : double **qX** : double **qY** : double **qZ** : double **offset** : boolean	Set the quaternion of joint (only in Dynamics OFF).
_setMass	**name** : string **mass** : double	Set the mass of the entity .
_addForce	**name** : string **x** : double **y** : double **z** : double	Add force to a body using absolute coordinates (only in Dynamics ON).
_setForce	**name** : string **x** : double **y** : double **z** : double	Set the force applied to the entity (only in Dynamics ON).

_addForceAtPos	**name** : string **x** : double **y** : double **z** : double **posX** : double **posY** : double **posZ** : double	Add force to a entity using absolute coordinates at specified absolute position (only Dynamics ON).
_addForceAtRelPos	**name** : string **x** : double **y** : double **z** : double **posX** : double **posY** : double **posZ** : double	Add force to a entity using absolute coordinates at specified relative position (only Dynamics ON).
_addRelForce	**name** : string **x** : double **y** : double **z** : double	Add force to a entity using relative coordinates (only Dynamics ON).
_addRelForceAtPos	**name** : string **x** : double **y** : double **z** : double **posX** : double **posY** : double **posZ** : double	Add force to a entity using entity-relative coordinates at specified absolute position (only Dynamics ON).
_addRelForceAtRelPos	**name** : string **x** : double **y** : double **z** : double **posX** : double **posY** : double **posZ** : double	Add force to a entity using entity-relative coordinates at specified relative position (only Dynamics ON).

_setDynamicsMode	**name** : string **boolean** : boolean	Enable (true) or disable (false) gravity mode.
_setRotation	**name** : string **qW** : double **qX** : double **qY** : double **qZ** : double	Set the entity orientation.

4 Services

Service name	Request	Response	Description
_get_time		**time** : double	Get the simulation time.
_get_obj_position	**name** : string	**posX** : double **posY** : double **posZ** : double	Get the position of the object named name, if name is empty, return the position of the agent which the service's name start with.
_get_parts_position	**name** : string **part** : string	**posX** : double **posY** : double **posZ** : double	Get the position of the part in parameter.
_get_rotation	**axis** : string	**qW** : double **qX** : double **qY** : double **qZ** : double	Get the rotation of ...
_get_angle_rotation	**axis** : string **x** : double **y** : double **z** : double	**angle** : double	Get the angle of ...
_get_joint_angle	**name** : string **nameArm** : string	**angle** : double	Get the angle between the joint.
_grasp_obj	**name** : string **obj** : string	**ok** : bool	Grasp the object "obj" with the part "part"
_get_entities		**entitiesNames** : string[] **length** : int	Get the names of the entities in the simulator.
_check_service	**serviceName** : string	**connected** : bool	Check if the service "serviceName" is connected.
_connect_to_service	**serviceName** : string	**connected** : bool	Connect the "serviceName", true if it is connected, false otherwise.
_get_collision_state _of_main_part		**collisionState** : bool	Get the collision state of the main part.
_is_grasped	**entityName** : string	**answer** : bool	True if "entityName" is grasped, false otherwise. If no entity name is provided, it will return the answer for the agent which is asked

_get_collision_state	**name** : string **part** : string	**collisionState** : boolean	If part="main" return getCollisionOfMainPart.
_check_service	**serviceName** : string	**connected** : boolean	Check if the service called "serviceName" is connected.
_connect_to_service	**serviceName** : string	**connected** : boolean	Connect the service "serviceName", return false if it fails, true otherwise.
_send_msg_to_service	**name** : string **msg** : string	**ok** : boolean	Send the message "msg" to the service called "name", return true if it is done, false otherwise.
_get_all_joint_angles	**name** : string	**jointName** : string[] **angle** : double[]	Get the angles for each joints.
_get_joint_position	**name** : string **jointName** : string	**posX** : double **posY** : double **posZ** : double	Get the position of the joint.
_get_mass	**name** : string	**mass** : double	Get the mass of the entity called "name".

5 FAQ

You don't see the robot on the world

Try changing the position of the camera with the keys Ctrl, Maj and/or Alt and the mouse.

SIGViewer has crashed

Don't worry, restart the viewer, it will work.

I can't publish to a topic

Have you started the roscore? If not tape on a terminal:

```
roscore
```

If you have started it, have you made a source? If not, tape:

```
source ~/catkin_ws/devel/setup.bash
```

fatal error: Controller.h: No such file or directory

If this error occurs, verify it the link to sigserver on the CMakeLists is correct. It should be `/home/<user>/sigverse-<version>/include/sigverse/home/<user>/catkin_ws/src/sig_ros/src/`

Bibliography

[1] SIGVerse wiki page :
`http://www.sigverse.org/wiki/en/index.php?Tutorial`.

[2] SIGServer wiki page :
`http://www.sigverse.org/wiki/en/index.php?Tutorial%2FInstallation%20of%20SIGVerse%20server`.

[3] SIGViewer wiki page :
`http://www.sigverse.org/wiki/en/index.php?Tutorial%2FInstallation%20of%20SIGViewer`.

[4] ROS wiki page :
`http://wiki.ros.org/ROS/Tutorials`.

[5] SIGVerse wiki page ROS integration tutorial :
`http://www.sigverse.org/wiki/en/index.php?ROS%20integration`.