ROS-I Fanuc

Michaloski, John L. (Fed)

5/2/2017 1:34:00 PM

Fanuc LRMate 200iD

Company: Fanuc

Controller: R i30

Version: V8.3.0 P/13 7DC3/13

DOF: 6

Joint names: joint\_1, joint\_2, joint\_3, joint\_4, joint\_5, joint\_6

Joint limits:

joint velocities:

Contents

[NIST Fanuc LRMate 200iD Robot Operation Instructions 3](#_Toc481749469)

[Powerup: 3](#_Toc481749470)

[Powerdown: 3](#_Toc481749471)

[Fault Recovery 3](#_Toc481749472)

[Run ROS Fanuc demo 4](#_Toc481749473)

[Run RVIZ roslaunch with Fanuc LRMate 200id 4](#_Toc481749474)

[Launch file: 4](#_Toc481749475)

[Fanuc ROS installation tutorial - I 4](#_Toc481749476)

[Create a Joint Position Streaming Interface utilizing TCP Socket Libraries 5](#_Toc481749477)

[Joint Commands (Motion) 5](#_Toc481749478)

[Pseudo-Code 5](#_Toc481749479)

[Joint Feedback (Joint States) 6](#_Toc481749480)

[Pseudo-Code 6](#_Toc481749481)

[ROS Code 6](#_Toc481749482)

[Fanuc ROS Industrial Integration 7](#_Toc481749483)

[Installation of ROS-Industrial on Fanuc controllers 7](#_Toc481749484)

[This section guides you through an installation of the ROS-Industrial programs on Fanuc controllers. Overview 7](#_Toc481749485)

[Installation 7](#_Toc481749486)

[Prerequisites 8](#_Toc481749487)

[Creating a Workcell 8](#_Toc481749488)

[Importing 9](#_Toc481749489)

[Building 10](#_Toc481749490)

[Binary Install 10](#_Toc481749491)

[Copying the Binaries 10](#_Toc481749492)

[Next 12](#_Toc481749493)

[Notes 12](#_Toc481749494)

[References 12](#_Toc481749495)

[Configuration of ROS-Industrial on Fanuc controllers 12](#_Toc481749496)

[Server Tags 12](#_Toc481749497)

[Flags and Registers 14](#_Toc481749498)

[Maximum Concurrent Task Number 14](#_Toc481749499)

[Payload, Tool and User Frames (optional) 14](#_Toc481749500)

[KAREL and TPE Programs 15](#_Toc481749501)

[Motion Speed and Segment Termination 18](#_Toc481749502)

[Complete the Configuration 19](#_Toc481749503)

[Notes 19](#_Toc481749504)

[References 19](#_Toc481749505)

[Running the ROS-Industrial programs on your Fanuc robot 19](#_Toc481749506)

[Prerequisites 19](#_Toc481749507)

[Outline 19](#_Toc481749508)

[Troubleshooting 20](#_Toc481749509)

[Controller 21](#_Toc481749510)

[KAREL programs are invisible on the Program Select window 21](#_Toc481749511)

[MEMO-159: Convert failed in PROG 21](#_Toc481749512)

[TPIF-013: Other program is running 21](#_Toc481749513)

[KAREL driver 22](#_Toc481749514)

[HOST-144: Comm Tag error 22](#_Toc481749515)

[FILE-032: Illegal parameter 22](#_Toc481749516)

[INTP-320: (LIBSSOCK, N) Undefined built-in 22](#_Toc481749517)

[jnt\_pkt\_srlise err: ERRNO 23](#_Toc481749518)

[exec\_move err: -1 23](#_Toc481749519)

[ROS 24](#_Toc481749520)

[RViz only shows 'collision quality' models 24](#_Toc481749521)

[Interactive marker in RViz cannot be dragged around 24](#_Toc481749522)

[Robot stops at seemingly random points during trajectory execution 24](#_Toc481749523)

[Failed to load byte array 25](#_Toc481749524)

[Unable to reconnect to the robot 25](#_Toc481749525)

[fanuc\_driver\_exp 26](#_Toc481749526)

[Overview 26](#_Toc481749527)

[Installation 26](#_Toc481749528)

[Binaries 26](#_Toc481749529)

[Source build 27](#_Toc481749530)

[Copying 27](#_Toc481749531)

[Configuration 27](#_Toc481749532)

[Running 27](#_Toc481749533)

[License 27](#_Toc481749534)

# NIST Fanuc LRMate 200iD Robot Operation Instructions

## Powerup:

1. Turn on power on front of controller (keyed)
2. If auto mode, make sure teach pendant upper left corner knob is **OFF**
3. If in fault- hold deadman switch halfway, Hold [**Shift**], press [**Reset**] key
4. reset to local mode Menu -> [32] Remote/Local/... [F4] Local [Enter]
5. Start ROS programs [Teach][Select] => scroll down to ROS, number 39 hit [Enter] (starts 2 programs)
6. Cycle start Green Auto button on front controller panel, press/release, green Light should go on.

## Powerdown:

1. Kill ROS programs - DO TWICE - 2 programs running [FCNT] -> 1 -> [ENTER]   
   [FCNT] -> 1 -> [ENTER]

## Fault Recovery

If fanuc controller faulted, you have to manually reset the joint(s) to a "safe" position:

1. Turn controller box to teach pendant from auto
2. hold deadman switch half-on, [**SHIFT**] Hold, hit [**Reset**]
3. Now move robot - +/- joint key or xyz key
4. Note to increase traversal- feedoverride in green xx% field in upper right corner

## Run ROS Fanuc demo

roslaunch fanuc\_lrmate200id\_moveit\_config moveit\_planning\_execution.launch

sim:=false robot\_ip:=129.6.78.111

## Run RVIZ roslaunch with Fanuc LRMate 200id

**FIXME**

#!/bin/bash

source .../catkin\_ws/devel/setup.bash

roslaunch nist\_fanuc lrmate200id\_sim.launch

sleep 100

## Launch file:

<?xml version="1.0"?>

<launch>

<arg name="sim" default="true" />

<include file="$(find fanuc\_lrmate200id\_moveit\_config)/launch/moveit\_planning\_execution.launch">

<arg name="sim" value="$(arg sim)"/>

</include>

</launch>

## Fanuc ROS installation tutorial - I

Web stop #1:

http://wiki.ros.org/fanuc?distro=indigo

# Create a Joint Position Streaming Interface utilizing TCP Socket Libraries

These notes apply to creating the joint position streaming interface on an industrial controller using the ROS-Industrial TCP socket libraries. This interface is good for basic motion and proof of concept integration.  
  
Joint Position Streaming Interface

The controller acts as a server for all connections. All types are 32 bit types (floats and integers, byte order is "Little Endian" (see [bug write up](http://code.google.com/p/swri-ros-pkg/issues/detail?id=8&can=1) for more information).

There are two connections, one for joint commands and one of joint feedback. Here are the details:

## Joint Commands (Motion)

|  |
| --- |
| Socket type: TCP |
| Socket port: 11000 |
| [JointMessages](http://wiki.ros.org/JointMessages) (constant values shown in **bold**) |
| Client request: LENGTH(bytes-not including length specifier), **10** (MSG\_ID – JOINT MESSAGE), **2** (COMM\_TYPE - REQUEST), **0** (REPLY\_TYPE – N/A), SEQ\_NUMBER(see special values below), JOINT\_DATA[10] (in rads (floats)) |
| Server reply: LENGTH(bytes), **10**, **3**(COMM\_TYPE – RESPONSE), REPLY (1 = SUCCESS, 2 = FAILURE), UNUSED, UNUSED[10] |

SEQ\_NUMBER – The sequence number is the number assigned by ROS to each of the points (i.e. this is always 0 or positive). A value of -1 indicates the end of a trajectory, a value of -2 indicates a stop (in both cases the controller stops, **joint data is not valid for these special types**)

### Pseudo-Code

The client will send joint points one at a time. When it receives the server response it sends the next point. This allows the server (controller) to determine the rate at which points are sent and how much buffering is performed. In practice this results in slower motion than desired, but the point spacing can be adjusted in ROS to account for this.

Pseudo-code for the controller is shown below (code for specific controllers might look very different given the differences between controllers).

while(true)

if(tcp.isConnected())

{

tcp.receive(jointPositionMsg)

jointPosition = decode(jointPositionMsg)

MoveJ(jointPosition)

jointResponseMsg = encode(jointResponse)

tcp.send(jointResponseMsg)

}

else

{

tcp.connect()

}

}

## Joint Feedback (Joint States)

|  |
| --- |
| Socket type: TCP |
| Socket port: 11002 |
| [JointMessages](http://wiki.ros.org/JointMessages) (constant values shown in **bold**) |
| Server streams: LENGTH(bytes), **10**, **1**(COMM\_TYPE – TOPIC), **0** (REPLY\_TYPE – N/A), UNUSED, JOINT\_DATA[10] (in rads (floats)) |

The Joint States just stream rate is determined by the controller, 10-50Hz is an appropriate range.

### Pseudo-Code

Pseudo-code for the controller is shown below (code for specific controllers might look very different given the differences between controllers).

while(true)

if(tcp.isConnected())

{

jointCurrentMsg = encode(jointCurrent)

tcp.send(jointCurrentMsg)

sleep(rate)

}

else

{

tcp.connect()

}

}

## ROS Code

The ROS Code to support this interface can be found in the dx100 package. This code can be reused for different controllers as it was developed genercially (**TODO: Move this generic code to a package under the ROS-Industrial stack**).

# Fanuc ROS Industrial Integration

The following pages contain tutorials specific to [ROS-Industrial](http://wiki.ros.org/Industrial) [Fanuc](http://wiki.ros.org/fanuc) support. Use the below listing to continue to the correct version. This documentation was collated from the ROS-Industrial Fanuc tutorials (as I find hyperlinked documentation evil).

1. [Installation of ROS-Industrial on Fanuc controllers](http://wiki.ros.org/fanuc/Tutorials/hydro/Installation)

This section guides you through an installation of the ROS-Industrial programs on Fanuc controllers.

1. [Configuration of ROS-Industrial on Fanuc controllers](http://wiki.ros.org/fanuc/Tutorials/hydro/Configuration)

This section guides you through the configuration of the ROS-Industrial Fanuc components on the controller.

1. [Running the ROS-Industrial programs on your Fanuc robot](http://wiki.ros.org/fanuc/Tutorials/Running)

This section explains how to run the ROS-Industrial programs installed in the previous tutorial(s) in simulation and on the real hardware.

## Installation of ROS-Industrial on Fanuc controllers

### This section guides you through an installation of the ROS-Industrial programs on Fanuc controllers. Overview

This section guides you through all the steps necessary to install the ROS-Industrial programs onto a Fanuc Robotics controller. After installation, the programs and controller will need to be configured, which will be described in the next section.

All of the steps outlined in this tutorial can be used for setting up a real controller as well as a simulated one in Roboguide. For a source install, the only difference is in the [Copying the Binaries](http://wiki.ros.org/fanuc/Tutorials/hydro/Installation#Copying_the_Binaries) step: Roboguide automatically loads the binaries onto the virtual robot controller as part of the build process, so no additional action is needed. For a binary install, the copy destination is the virtual controller instead of the real one.

Note that these tutorials only cover the installation and setup of those programs of the [fanuc\_driver](http://wiki.ros.org/fanuc_driver) package that are supposed to run on the controller. For help on installing ROS-Industrial in general, please see the ROS-Industrial [installation](http://wiki.ros.org/Industrial/Install) pages.

Finally: this tutorial assumes the use of an iPendant. For users with older (legacy) pendants, this may mean that key sequences given in the text, screenshots and menus will not match completely. Due to the lower resolution of legacy pendants, many key sequences need either an additional NEXT inserted, or (sub)menus should be opened using TYPE.

### Installation

There are two options for installation: from source or using a set of precompiled binaries. In general, a binary install is recommended, as it is much faster and less involved (note: there are currently no KAREL binaries available for the Indigo version of the driver, thus a source install is always necessary for that version). After choosing a compatible version, only a small number of files have to be copied to the controller.

If it is not possible to use the provided binaries, a source install is the only option. This may be the case if the target controller runs an incompatible version of the KAREL runtime, if it cannot support access to KAREL Vars or if there is a requirement to use customised driver code. A source install is more complicated, as it necessitates the creation of a Roboguide workcell, and the compilation of several source files.

The same configuration procedure is used for both installation types.

### Prerequisites

For a binary install, the following is needed:

* a compatible binary distribution of the KAREL and TP binaries[1](http://wiki.ros.org/fanuc/Tutorials/hydro/Installation#fnref-dde6292df22fb0a09dfff784c141410b8b2772f9)
* a method of transferring the necessary files to the controller (USB stick, a network connection)

For a source installation, additional requirements are:

* a copy of the [fanuc\_driver](http://wiki.ros.org/fanuc_driver) package, version 0.2 or newer
* an installation of Fanuc Roboguide[2](http://wiki.ros.org/fanuc/Tutorials/hydro/Installation#fnref-44087a7991e33fe2bf15fd2cf3e7715ffbc0301c)
* a Windows PC or VM (as Roboguide does not run on Linux)

In all cases, the following software options should be present on your robot in addition to basic networking and TCP/IP support:

* R632 - KAREL
* R648 - User Socket Messaging

Information on the installed software options can be found by consulting the documentation that came with your robot. Alternatively, the Version ID submenu of the STATUS screen can be used. On the TP: Menu→NEXT→STATUS→Version ID, then CONFIG (or F3). The same information is available on the Summary Configuration/Status webpage, which should be accessible if your controller has the web server option installed.

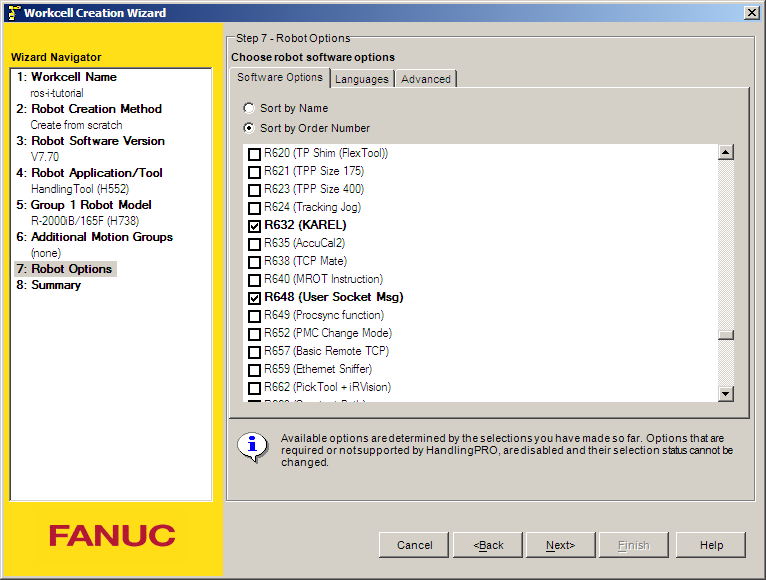
Make sure your controller has a correctly configured and working network setup. Verify you can ping the controller from your ROS pc. Refer to the relevant documentation for instructions on how to configure networking on a Fanuc industrial robot controller.

If you've chosen to perform an installation using the precompiled binaries, please proceed to the [Binary Install](http://wiki.ros.org/fanuc/Tutorials/hydro/Installation#Binary_Install) section. For an installation from source, continue with the next section.   
  
Source Install

Before the ROS-Industrial programs can be transferred to the robot controller, they need to be compiled into binaries. We'll use Fanuc Roboguide for this.

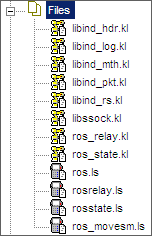
### Creating a Workcell

First, create a new workcell in Roboguide, either by using the wizard or by creating a copy from your actual robot controller. When using the wizard, be sure to select the correct Application package, as well as the correct robot type. On the software options screen, also make sure to select at least options R632 - KAREL and R648 - User Socket Messaging.



After completing the wizard, you should end up with an empty workcell with just your robot in the middle of it. If you chose to create a copy of the controller, the Cell Browser might show programs and files that were already present on the robot under the Programs and Files sections. As long as they do not conflict with the ROS-Industrial KAREL programs, they can be ignored.

### Importing

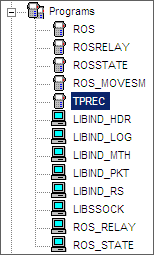
****In the Cell Browser, import the KAREL and TPE sources into the workcell using the Project→Add Files menu option. Browse to your local [fanuc\_driver](http://wiki.ros.org/fanuc_driver) package and select all \*.klfiles in the karel subdirectory not ending in \_h.kl or \_t.kl.

For [fanuc\_driver](http://wiki.ros.org/fanuc_driver) version 0.3.0 and up, just select all .kl files in the karel subdirectory.

Click the Open button to add them to the Cell Browser.

Now add the \*.ls files in the tpe subdirectory as well. The Files section in the Cell Browser should contain at least the files shown.

## Building

****Now build all files by selecting Project→Build→All in Roboguide (if you have other source files under Files, build only those shown in the [Importing](http://wiki.ros.org/fanuc/Tutorials/hydro/Installation#Importing) section). No errors or warnings should be reported, and the Programs section in the Cell Browser should show the compiled binaries (TPRECis a default program and is not a part of ROS-Industrial).

Now continue with [Copying the Binaries](http://wiki.ros.org/fanuc/Tutorials/hydro/Installation#Copying_the_Binaries).

### Binary Install

A Roboguide workcell is not needed for a binary install: all that is required is a copy of the necessary binaries. They should be compatible with the software installed on the target controller, as well as with the installed version of fanuc\_driver (so version 0.2.0 of fanuc\_driver must be used with version 0.2.0 of the precompiled binaries).

From the [Binaries](http://wiki.ros.org/fanuc_driver/hydro#Binaries) section on the fanuc\_driver page, download the correct version and extract it in some temporary location. You should end up with a directory containing several .pc and .tp files. Make sure the names correspond to those listed in the [Building](http://wiki.ros.org/fanuc/Tutorials/hydro/Installation#Building) section.

We can now continue to copy the extracted files to the target controller.

### Copying the Binaries

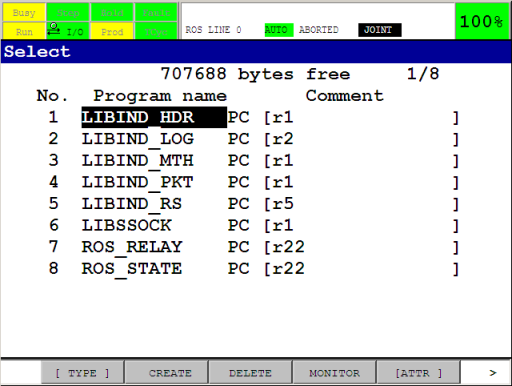
Finally, the binaries need to be transferred to the controller. This can be done by direct transfer over a network connection (FTP) or by using a mass storage device (such as a USB stick or PCMCIA memory card). Refer to [[1, Chapter 8]](http://wiki.ros.org/fanuc/Tutorials/hydro/Installation#References) for information on how to copy files using removable media. Refer to the Roboguide help files for information on how to perform network transfers of files to robots[3](http://wiki.ros.org/fanuc/Tutorials/hydro/Installation#fnref-fd56541f852dca8b9b8180c7cac57f8e0516535a).

Updating older versions of the ROS-Industrial KAREL programs may result in a MEMO-159 or VARS-014 error. See [MEMO-159: Convert failed in PROG](http://wiki.ros.org/fanuc_driver/Troubleshooting#MEMO-159:_Convert_failed_in_PROG) in that case. TP programs are always overwritable, provided none of them are loaded as the current program (ie: loaded through the Program Select window).

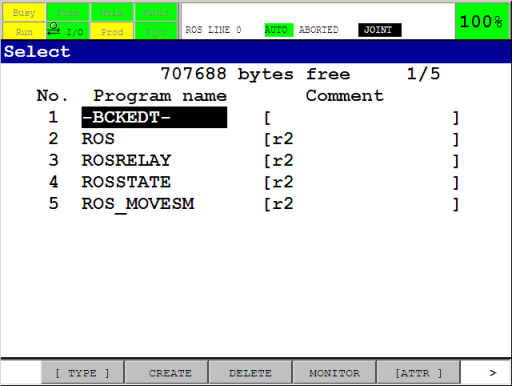
After copying the binaries onto the robot controller, open the Program Select window by pressing the Select button on the TP, then [TYPE]→KAREL Progs.

If the KAREL programs do not show up on the TP, or you cannot change the display TYPE to KAREL Progs, make sure the KAREL support has been properly setup on your controller. See [KAREL programs are invisible on the Program Select window](http://wiki.ros.org/fanuc_driver/Troubleshooting#KAREL_programs_are_invisible_on_the_Program_Select_window) on the [Troubleshooting](http://wiki.ros.org/fanuc_driver/Troubleshooting) page.

Make sure the following programs are listed (note that the revision numbers may be different):



Also check to make sure the necessary TPE programs are present by pressing [TYPE]→TP Programs. You should see the following programs (the -BCKEDT- program is not part of ROS-Industrial):



The ROS-Industrial programs are now installed on the controller.

### Next

Now that the necessary libraries and programs have been installed they need to be configured. We will do this in the next section.

### Notes

1. There is a certain amount of backward compatibility in KAREL, so it may be possible to use the provided binaries, even though the runtime version specified does not match that of the controller. The TP programs do not use any non-standard options, so they should be compatible with most controllers. ([1](http://wiki.ros.org/fanuc/Tutorials/hydro/Installation#fndef-dde6292df22fb0a09dfff784c141410b8b2772f9-0))
2. As an alternative to Roboguide, WinOLPC / OlpcPRO or even the KCL console could be used to compile the KAREL sources into p-code binaries. These alternatives are not documented in this tutorial, and only the use of Roboguide is described. ([2](http://wiki.ros.org/fanuc/Tutorials/hydro/Installation#fndef-44087a7991e33fe2bf15fd2cf3e7715ffbc0301c-1))
3. If you're setting up a virtual controller, FTP to localhost can be used, or you can copy the binaries to the Robot\_N\MC directory (where N is the index of the robot controller). This directory is a subdirectory of the workcell directory under My Workcells. For a workcell named ROS-I Test, this directory would by default be at %USERPROFILE%\(My) Documents\My Workcells\ROS-I Test\Robot\_N\MC. ([3](http://wiki.ros.org/fanuc/Tutorials/hydro/Installation#fndef-fd56541f852dca8b9b8180c7cac57f8e0516535a-2))

### References

1. FANUC Robot Series, R-30iA, Handling Tool, Operator's Manual, B-82594EN-2/02
2. FANUC Robot series, R-30iA/R-30iA Mate Controller, KAREL Function, Operator's Manual, B-83144EN/01
3. FANUC Robotics SYSTEM, R-30iA Controller, KAREL Reference Manual, MARRC75KR07091E Rev C

# Configuration of ROS-Industrial on Fanuc controllers

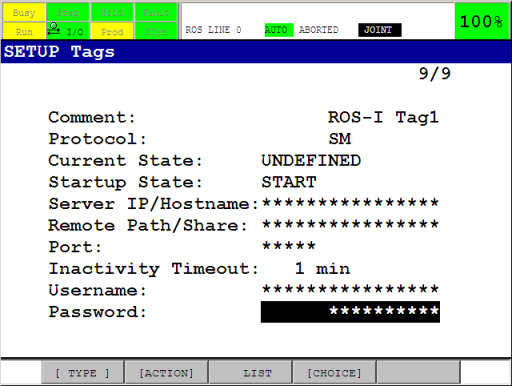
This section guides you through the configuration of the ROS-Industrial Fanuc components on the controller.  
Overview

Before the ROS-Industrial programs can be started, some additional configuration of both the controller and the programs is required.

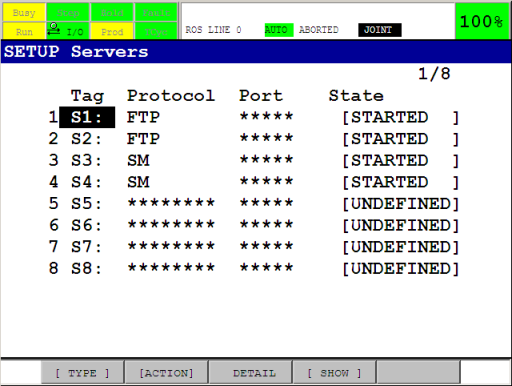
### Server Tags

The ros\_state and ros\_relay programs make use of the User Socket Messaging (USM) software option, and expect two Server Tags to be available. By default ROS-Industrial uses 3 and 4, and they must be properly configured before use (1 and 2 are typically used by the built-in FTP server).

To check Tag availability and status, open the Host Comm setup screen (Menu→SETUP→Host Comm), and then go to the SETUP Servers listing using [SHOW]→Servers. Pick two available Tags and write down their numbers (we'll need those later, when updating the configuration of the ROS-Industrial programs). Finally, make sure the Tags are configured as follows (the comment is not required):



Go back to the SETUP Servers listing and verify that the chosen Tags are in state STARTED. If not, start each tag with [ACTION]→DEFINE, then [ACTION]→START, or restart the control cabinet. Make sure they have been started by re-opening the SETUP Servers screen (if necessary) and checking the Current State of both tags: it should say STARTED.



For more information on User Socket Messaging and tags, see [[3, Chapter 11]](http://wiki.ros.org/fanuc/Tutorials/hydro/Configuration#References) and [[3, Section 11.3.2]](http://wiki.ros.org/fanuc/Tutorials/hydro/Configuration#References).

### Flags and Registers

The ros\_relay and ros\_movesm programs make use of several registers and flags. By default, flags 1 & 2, integer registers 1 & 2 and position registers 1 & 2 are used. Make sure these are not used by any other programs that may be running concurrently with the ROS-Industrial programs on the controller[1](http://wiki.ros.org/fanuc/Tutorials/hydro/Configuration#fnref-961e2784bfbafc8455e9dfb23d43b8f799f7782e). If any of the flags or registers are currently being used, find some alternatives, and note their numbers. We'll use those in the next section.

All flags and integer registers can be used, but the position registers need two registers at consecutive locations (ie: posreg 34 **and** 35).

It is good practice to set a descriptive comment on the used flags and registers. This will make it easier later in this tutorial to verify that the correct flags and registers are used in the ROS-Industrial programs.

### Maximum Concurrent Task Number

By default, the controller can only run a small number of concurrent user tasks, which could prevent the ROS-Industrial programs from starting. The currently configured number of allowed tasks is stored in the $SCR.$MAXNUMTASK system variable. Any number above 3 should be enough to start the KAREL programs, but higher may be necessary if the controller also starts other tasks.

Changing system variables can result in an unstable or non-functioning system. Before making any changes, be sure to have a recent backup of the controller and consult with the robot operator in case you are unsure about any of these steps.

The current value can be found in the SYSTEM Variables screen (Menu→NEXT→SYSTEM→Variables). Scroll down to $SCR, press ENTER, then scroll to the $MAXNUMTASK entry.

To change the number, start the controller using a controlled start. Then open the Program Limits screen using Menu→NEXT→PROGRAM SETUP. Then input a new value for the User Tasks entry. Now restart the controller using Fctn→START (COLD).

Refer to [[1, Section B.1.3]](http://wiki.ros.org/fanuc/Tutorials/hydro/Configuration#References) for more information on how to perform a controlled start.

### Payload, Tool and User Frames (optional)

The following system settings may influence the behaviour of the robot when used with ROS-Industrial, however for basic operation, changes to these settings are not absolutely necessary.

Changes to payload schedules and / or selected user and tool frames may not be apparent to other users of the robot and can potentially influence all other programs on the controller. Be sure to have a recent backup and / or write down their current values in order to be able to restore them if necessary.

For the payload schedule, input the correct information using the TP on the Motion/Payload Set setup screen (Menu→NEXT→SYSTEM→Motion). Be sure to configure and select the correct schedule. For more information, see [[1, Section 9.14]](http://wiki.ros.org/fanuc/Tutorials/hydro/Configuration#References).

Configured user and tool frames are (currently) ignored in the ROS Industrial nodes as motion trajectories are specified in joint space. This is also true for robot state reporting. The use of a user frame coincident with the world frame is however recommended as it helps in debugging and verifying the motion of the arm and the location of the TCP using the [tf](http://wiki.ros.org/tf) tools available in ROS.

Tool frames configured at the robot controller side are also not taken into account by the ROS motion planners, as tools should be represented by an appropriate URDF. The combination of the arm and tool URDFs provides similar information to the planning libraries (see also the [Create a URDF for an Industrial Robot](http://wiki.ros.org/Industrial/Tutorials/Create%20a%20URDF%20for%20an%20Industrial%20Robot) tutorial). It is however recommended to manually verify the correct tool frame setup on the TP, as the robot controller also uses this information.

Go to Menu→Setup→Frames and use the [OTHER] function key to switch between the Tool and User Frame listings. Disable the user frame: use the Next button, then press the CLRIND function key (alternatively, configure an all zeros user frame) [2](http://wiki.ros.org/fanuc/Tutorials/hydro/Configuration#fnref-d1ed32ae8608d184b1126b1dd3011e19696fc8ef). Verify the tool frame setup in the Tool Frame screen.

For more information on user and tool frame setup, see [[1, Section 3.9]](http://wiki.ros.org/fanuc/Tutorials/hydro/Configuration#References).

### KAREL and TPE Programs

With the information gathered in the previous sections, we can now proceed to configure the KAREL and TP programs on the controller. Even if you do not have to change any of the defaults, you still need to follow the steps in this section.

#### KAREL Programs

The default configuration of the ros\_relay program is shown in Table 1. Table 2 shows the default configuration of the ros\_state program.

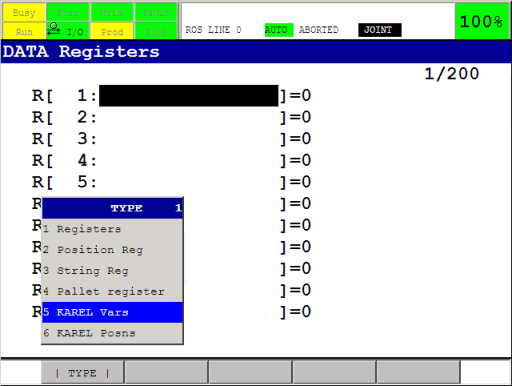
| **Name** | **Type** | **Default** | **Unit** | **Description** |
| --- | --- | --- | --- | --- |
| checked | boolean | False | - | Configuration has been completed by user |
| f\_msm\_rdy | integer | 1 | - | movesm i'face: 'ready/ack' signal flag |
| f\_msm\_drdy | integer | 2 | - | movesm i'face: 'data ready' signal flag |
| loop\_hz | integer | 42 | Hz | Main loop update rate |
| move\_cnt | integer | 50 | % | CNT to set with each joint motion instruction |
| move\_speed | integer | 20 | % | Joint speed to set for all trajectory points |
| pr\_move | integer | 1 | - | movesm i'face: position register for next trajectory point |
| r\_move\_spd | integer | 1 | - | movesm i'face: integer register for motion speed |
| r\_move\_cnt | integer | 2 | - | movesm i'face: integer register for CNT value |
| s\_tcp\_nr | integer | 11000 | - | TCP port to listen on |
| s\_tag\_nr | integer | 4 | - | Index of the Server Tag to use |
| um\_clear | boolean | True | - | Clear user menu on start |

* Table 1: default configuration of ros\_relay.

| **Name** | **Type** | **Default** | **Unit** | **Description** |
| --- | --- | --- | --- | --- |
| checked | boolean | False | - | Configuration has been completed by user |
| loop\_hz | integer | 42 | Hz | Main loop update rate |
| sloop\_div | integer | 10 | - | Divider for robot\_status reporter loop |
| s\_tcp\_nr | integer | 11002 | - | TCP port to listen on |
| s\_tag\_nr | integer | 3 | - | Index of the Server Tag to use |
| um\_clear | boolean | True | - | Clear user menu on start |

* Table 2: default configuration of ros\_state.

The data in these two tables will need to be entered into the configuration structures of the respective programs. If the above defaults cannot be used, update the relevant entries with the new values. If for instance Server Tag 3 is not available, use the index number of a Tag that can be used. 



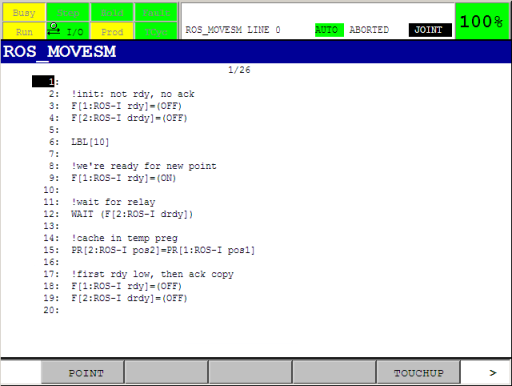
To access the configuration structures, open the Program Select window by pressing the Select button on the TP, then [TYPE]→KAREL Progs. Now select ros\_state (or ros\_relay), and press ENTER. Press the Data button, then [TYPE]→KAREL Vars. You should now see the cfg\_ variable, press ENTER.

If you selected ros\_state, the next screen should show the contents of Table 2. To change the server tag used, navigate to the s\_tag\_nr entry, press ENTER and input the new value. Press ENTER again when finished. Do this for all the entries in the structure.

Go back to Program Select and repeat this procedure for the ros\_relay program (now use Table 1).

#### TPE Programs

Unfortunately, the TPE programs cannot use the configuration data of the KAREL programs, so they must be manually updated every time we change the used flags, integer or position registers. Fortunately, only the ros\_movesm program needs to be updated.



On the TP, open the ros\_movesm program, and update all statements referencing the flags, integer or position registers with the new values. Make sure they correspond with the values you entered in the KAREL configuration in the previous section. If you've set comments on the registers and flags in the [Flags and Registers](http://wiki.ros.org/fanuc/Tutorials/hydro/Configuration#Flags_and_Registers) section, the TP program should show these in the appropriate places.

Incorrect configuration of the flags, integer or position registers can result in damage to your robot, you and / or your workcell. Please make sure they are free to use, before continuing with the next steps. If you are unsure about register or flag usage by other programs, ask the operator of the robot to assist you.

### Motion Speed and Segment Termination

By default, the ros\_relay program uses a CNT value of 50% and a fixed joint velocity of 20% for all motion commands (Table 1). Depending on the precision required during trajectory execution, these values can be changed. The relevant entries are move\_cnt and move\_speed in ros\_relay, and can be changed using the procedure explained at the beginning of this section.

For more information on the influence of these two values on trajectory execution, see the [Improving ROS-Industrial motion on an Industrial Robot](http://wiki.ros.org/Industrial/Tutorials/Improving_Motion_on_an_Industrial_Robot) tutorial.

As all MoveIt motion plans are planned using the real velocity limits found in the URDF, the MoveIt Trajectory Execution Manager will frequently abort the execution (resulting in Trajectory stopmessages on the User Menu on the TP). Execution monitoring can either be disabled, or the allowed time can be increased by setting the relevant parameters. See the The Trajectory Execution Manager section on the [Executing Trajectories with MoveIt!](http://moveit.ros.org/wiki/Executing_Trajectories_with_MoveIt!#The_Trajectory_Execution_Manager) page on the MoveIt wiki for more information.

The Troubleshooting subpage of [fanuc\_driver](http://wiki.ros.org/fanuc_driver) also has an entry on this: see [Robot stops at seemingly random points during trajectory execution](http://wiki.ros.org/fanuc_driver/Troubleshooting#Robot_stops_at_seemingly_random_points_during_trajectory_execution).

### Complete the Configuration

As the last step, complete the configuration by setting the checked entry in each of the configuration structures to TRUE.

### Notes

1. Note that the list of flags can be accessed via Menu→I/O→NEXT→Flag. The F[] array on the TP corresponds to the FLG[] array in the KAREL programs. ([1](http://wiki.ros.org/fanuc/Tutorials/hydro/Configuration#fndef-961e2784bfbafc8455e9dfb23d43b8f799f7782e-0))
2. Depending on the configuration and mounting of the arm, an 'all-zeros' user frame may not be the best choice. In those situations, be sure to configure a more suitabe frame. ([2](http://wiki.ros.org/fanuc/Tutorials/hydro/Configuration#fndef-d1ed32ae8608d184b1126b1dd3011e19696fc8ef-1))

### References

1. FANUC Robot Series, R-30iA, Handling Tool, Operator's Manual, B-82594EN-2/02
2. FANUC Robot series, R-30iA/R-30iA Mate Controller, KAREL Function, Operator's Manual, B-83144EN/01
3. FANUC Robotics SYSTEM, R-30iA Controller, KAREL Reference Manual, MARRC75KR07091E Rev C

## Running the ROS-Industrial programs on your Fanuc robot

This section explains how to run the ROS-Industrial programs installed in the previous tutorial(s) in simulation and on the real hardware.  
  
Overview

This page provides instructions on how to verify correct installation of the ROS-Industrial programs on your Fanuc robot and on how to run them for use with the fanuc\_X\_support and fanuc\_M\_moveit\_config packages provided through the ROS-Industrial repositories.

Whenever this page refers to packages, please replace X with the robot series, and M with the robot variant. See [Working with ROS-Industrial Robot Support Packages](http://wiki.ros.org/Industrial/Tutorials/WorkingWithRosIndustrialRobotSupportPackages) for more information.

### Prerequisites

In order to be able to complete this tutorial, you should have access to the following:

* the [fanuc\_driver](http://wiki.ros.org/fanuc_driver) package and the appropriate support package for your robot (fanuc\_X\_support)
* the appropriate [moveit](http://wiki.ros.org/moveit) or [arm\_navigation](http://wiki.ros.org/arm_navigation) (deprecated) package for your specific robot variant (fanuc\_M\_moveit\_config)
* a Linux PC to run the ROS nodes and RVIZ
* an installation of Fanuc Roboguide (only used for simulation)
* a Windows PC or VM for Roboguide (only used for simulation)

### Outline

Running in Auto mode means only the e-stop (on the TP or on the controller cabinet) can be used to stop the robot in an emergency. Always adhere to safety regulations and take proper precautions when working with the real robot.

/!\ This section is unfinished, and assumes familiarity with ROS-Industrial robot support and [MoveIt](http://wiki.ros.org/MoveIt) configuration packages. See [Working with ROS-Industrial Robot Support Packages](http://wiki.ros.org/Industrial/Tutorials/WorkingWithRosIndustrialRobotSupportPackages) for more detailed information on how to work with these packages.

Steps:

1. Test fanuc\_X\_support nodes with Roboguide. On the TP, start the rosstate TPE program. On the Linux PC, run:

roslaunch fanuc\_X\_support robot\_state\_visualize\_M.launch robot\_ip:=IP\_OF\_ROBOGUIDE use\_bswap:=false

Verify J2-J3 coupling by jogging the robot in Roboguide. RVIZ should show same movements.

When done, ABORT the rosstate program. ctrl+c the ROS session.

1. Test fanuc\_X\_support nodes with real robot. On the TP, start the rosstate TPE program. On the Linux PC, run:

roslaunch fanuc\_X\_support robot\_state\_visualize\_M.launch robot\_ip:=IP\_OF\_ROBOT

Again, verify correspondence RVIZ and real robot by jogging

When done, ABORT the rosstate program. ctrl+c the ROS session.

1. Test fanuc\_X\_support and fanuc\_M\_moveit\_configuration nodes with real robot in mode T1. On the TP, start the ros TPE program. On Linux PC, run:

roslaunch fanuc\_M\_moveit\_config moveit\_planning\_execution.launch sim:=false robot\_ip:=IP\_OF\_ROBOT

from the [MoveIt](http://wiki.ros.org/MoveIt) configuration package. Now perform simple motions (see the [MoveIt! RViz Plugin Tutorial](http://docs.ros.org/hydro/api/moveit_ros_visualization/html/doc/tutorial.html) for how to work with the [MoveIt](http://wiki.ros.org/MoveIt) RViz plugin). This mode requires the operator holding the TP and depressing the deadman.

When done, ABORT the ros program. ctrl+c the ROS session.

1. Test fanuc\_X\_support and fanuc\_M\_moveit\_configuration nodes with real robot in mode T2 or Auto. On the TP, start the ros TPE program. On Linux PC, run:

roslaunch fanuc\_M\_moveit\_config moveit\_planning\_execution.launch sim:=false robot\_ip:=IP\_OF\_ROBOT

again from the [MoveIt](http://wiki.ros.org/MoveIt) configuration package and plan and execute some motions. Remember that in T2 mode, the operator still needs to depress the TP deadman.

When done, ABORT the ros program. ctrl+c the ROS session.

# Troubleshooting

This page lists some of the possible errors and issues encountered when using the [fanuc\_driver](http://wiki.ros.org/fanuc_driver) package. The Remedy subsections provide possible solutions.

If you encounter an error not listed on this page or the provided remedy does not seem to work, please contact the developers using the [ROS-Industrial](https://groups.google.com/forum/?fromgroups#!forum/swri-ros-pkg-dev) Google group (direct mail: [ROS-Industrial](mailto:swri-ros-pkg-dev@googlegroups.com)).

Make sure to check the [Known Issues](http://wiki.ros.org/fanuc/indigo/known_issues) page before sending a message to the mailing list.

## Controller

### KAREL programs are invisible on the Program Select window

After copying the KAREL programs onto your controller, you do not see them on the Program Select window, nor can you change the displayed program TYPE to KAREL Progs.

**Cause**: Listing and accessing KAREL programs is not enabled on the controller.

**Remedy**: Enable the listing of KAREL programs by setting the $KAREL\_ENB system variable to 1. Open the SYSTEM Variables screen (Menu→NEXT→SYSTEM→Variables), and scroll down to $KAREL\_ENB. Press ENTER, and change the value to 1. The modification is immediate, you do not need to restart the controller.

### MEMO-159: Convert failed in PROG

While trying to copy a new or updated version of any of the ROS-Industrial KAREL programs, the TP shows the following error:

Duplicate creation TYPE mismatch

MEMO-159 Convert failed in PROG

VARS-014 Create type - KL\_TYPE failed

Where PROG is any KAREL program, and KL\_TYPE is a KAREL type used in PROG.

**Cause**: The type or structure of a variable used in PROG has changed in the updated version, and the KAREL runtime system cannot convert the variable to the new type. This error can occur even after removing PROG.PC file from the controller, as the related .vr file (or variable record, the file that stores information on variables) is not automatically deleted upon deleting the p-code file.

**Remedy**: Remove the PROG.VR file from the controller. This can be done from the Program Select window, as well as by using the Menu→FILE menu (switch to the MD: device). If the copy attempt left a PROG.PC file, remove it as well. The copy operation should now succeed.

In some cases, other (ROS-Industrial) KAREL programs or libraries have references to the old type as well and these will have to be removed too. Always first delete the .PC file, then the .VR.

**Note**: Removing the .VR file will result in loss of the values of the variables stored in the .VR. Backup important data before deleting the files. The PROG.VA file should provide you with an ASCII rendering of the .VR file.

### TPIF-013: Other program is running

Upon trying to (re)start the ros TP program, the TP shows the following error:

TPIF-013 Other program is running

**Cause**: This is caused by another TPE or KAREL program that is still running on the controller. You cannot start programs while others are still running.

**Remedy**: Make sure no other user programs are running or paused on the controller before trying to start the ROS programs. Fctn→ABORT ALL can be used to abort any running programs.

**Note**: You may need to use ABORT ALL twice for programs that have network sockets that are in the ACCEPT state. For the ros\_relay and ros\_state programs, this is when the TP shows the following on the User Menu:

12345 I RSTA Waiting for ROS state prox

12345 I RREL Waiting for ROS traj relay

and the ROS PC is not connected to the controller. Select Fctn→ABORT ALL twice, then try again.

## KAREL driver

### HOST-144: Comm Tag error

The TP shows the following on the User Menu:

12345 E PROG cfg err, TAG idx: N

Where PROG is any of RSTA or RREL, and N is a Server Tag index number.

**Cause**: the configured Server Tag does either not exist, is not correctly configured, or is not in the correct state.

**Remedy**: make sure you have configured the Tag correctly, and that the configuration of the respective ROS-Industrial program uses the correct tag number. Refer to the [Server Tags](http://wiki.ros.org/fanuc/Tutorials/hydro/Configuration#Server_Tags) section of the installation tutorial for information on Tag setup.

### FILE-032: Illegal parameter

The TP shows the following on the User Menu:

12345 E PROG cfg error: -1

12346 E PROG check cfg

Where PROG is any of RSTA or RREL.

**Cause:** the configuration of the respective program is either uninitialised, or has not been completed.

**Remedy:** make sure the configuration of all ROS-Industrial programs has been completed, and that the checked entry is set to TRUE. Refer to the [KAREL Programs](http://wiki.ros.org/fanuc/Tutorials/hydro/Configuration#KAREL_Programs) section of the installation tutorial for information on the setup of the KAREL and TPE programs.

### INTP-320: (LIBSSOCK, N) Undefined built-in

Nothing is shown on the User Menu, and all ROS-Industrial programs (ros\_state, ros\_relay) have been aborted.

**Cause**: this is most likely caused by attempting to start the programs on a controller that does not have all required options installed. For libssock, this is most likely option R648 (User Socket Messaging) (resulting in the missing MSG\_\* built-ins).

**Remedy**: make sure the controller has all required options installed (and activated). Refer to the [Prerequisites](http://wiki.ros.org/fanuc/Tutorials/hydro/Installation#Prerequisites) section of the installation tutorial for information on which options are required.

If the controller has the required options installed, please report the issue to the mailinglist and / or the [ros-industrial/fanuc](https://github.com/ros-industrial/fanuc/issues) repository. Do not forget to include any relevant information (ROS release, version of [fanuc\_driver](http://wiki.ros.org/fanuc_driver), controller type and software and used manipulator).

### jnt\_pkt\_srlise err: ERRNO

The TP shows the following on the User Menu:

12345 E PROG jnt\_pkt\_srlise err: ERRNO

12345 I PROG Waiting for ROS ..

Where PROG is any of RSTA or RREL, and ERRNO is a negative error number (possible values: -67213).

**Cause**: The ROS client disconnected the TCP socket connection at the moment the KAREL program tried to write to or read from it.

**Remedy**: This is not an error. The state proxy and trajectory relay will handle the disconnect and wait for a new connection.

### exec\_move err: -1

The TP shows the following on the User Menu:

12345 E RREL exec\_move err: -1

12345 I RREL Waiting for ROS ..

The ROS console shows:

[ERROR] [1234567890.123456789]: Socket sendBytes failed, rc: -1. Error: 'Broken pipe' (errno: 32)

[ERROR] [1234567890.123456789]: Failed to connect to server, rc: -1. Error: 'Transport endpoint is already connected' (errno: 106)

(the same error repeated)

[ERROR] [1234567890.123456789]: Failed to connect to server, rc: -1. Error: 'Transport endpoint is already connected' (errno: 106)

[ERROR] [1234567890.123456789]: Timeout connecting to robot controller. Send new motion command to retry.

Sending a new motion command does not always work.

**Cause**: The ROS client asked the trajectory relay to move to a point in the current trajectory that the robot is unable to reach (violation of joint limits). The relay aborts execution of the trajectory and disconnects the ROS client in that case.

**Remedy**: Verify that the joint limits configured on the Fanuc controller correspond to those in the used urdf (in the support package). The [MoveIt](http://wiki.ros.org/MoveIt) motion planners depend on these limits to generate valid trajectories. Update the limits in the xacro macro if there are any differences. Be sure to regenerate the urdf and any [MoveIt](http://wiki.ros.org/MoveIt) packages that depend on the it.

**Note**: In some cases it may be necessary to restart the ros TPE program on the controller.

## ROS

### RViz only shows 'collision quality' models

After starting RViz (either by using the test\_...launch files in the support packages, or when running any of the [MoveIt](http://wiki.ros.org/MoveIt)planning\_execution launch files), the visualisation only shows collision detail models, even when Visual Enabled is selected.

**Cause:** the support packages only include the collision quality models, as we do not currently have permission from Fanuc to distribute the detailed CAD models.

**Remedy:** none. A possible work-around would involve re-exporting the support package meshes from SolidWorks or any other suitable program. A complicating factor is that correct display of the model depends on proper origins of the meshes, which are not necessarily identical to those of the original models. Additionally, users would need access to the original models.

### Interactive marker in RViz cannot be dragged around

After starting demo.launch or moveit\_planning\_execution.launch from any of the Fanuc [MoveIt](http://wiki.ros.org/MoveIt) configurations, the 6D interactive marker cannot be dragged from the starting position, and the terminal window shows the following error:

[ERROR] [1234567890.123456789]: Exception caught while handling end-effector update: ikfast exception: /path/to/ikfast/plugin/solver.cpp:LINE\_NR: polyroots8: Assertion 'rawcoeffs[0] != 0' failed

**Cause:** the manipulator is most likely a singular configuration, for which the IK plugin cannot find any valid solutions. demo.launch (and moveit\_planning\_execution.launch without the sim:=false argument) does not use real joint state data, but a simple joint space interpolator, for which the initial state puts all joints at their 'zero position'. For some models this is a singular configuration.

**Remedy:** if possible, use a different initial state, command the manipulator to move away (in joint space) from the singular configuration, or (if in simulation) use the random valid option for the Goal State and click Plan and Execute button. As soon as the manipulator is out of the singular configuration, dragging the interactive marker should start working again.

### Robot stops at seemingly random points during trajectory execution

The TP shows the following on the User Menu:

12345 I RREL Trajectory stop

The ROS console shows:

[ERROR] [1234567890.123456789]: Controller is taking too long to execute trajectory (the expected upper bound for the trajectory execution was X.Y seconds). Stopping trajectory.

[ INFO] [1234567890.123456789]: MoveitSimpleControllerManager: Cancelling execution for

[ INFO] [1234567890.123456789]: Execution completed: TIMED\_OUT

Where X.Y is some duration in seconds.

**Cause:** the MoveIt Trajectory Execution Manager detects a trajectory execution overrun, and aborts execution of the trajectory. This is often due to the physical robot not being able to attain the speeds specified in the trajectory. The Fanuc KAREL trajectory relay overrides specified motion speed, resulting in the quoted error message.

**Remedy:** increase the MoveIt duration scaling parameter, or disable duration monitoring completely. See the The Trajectory Execution Manager section on the [Executing Trajectories with MoveIt!](http://wiki.ros.org/fanuc/Troubleshooting/MoveIt%20Trajectory%20Execution%20Manager) page on the MoveIt wiki or [How do I disable execution\_duration\_monitoring ?](http://answers.ros.org/question/196586/how-do-i-disable-execution_duration_monitoring/) on ROS Answers for more information.

### Failed to load byte array

The TP shows the following on the User Menu:

12345 I RSTA Connected

12345 I RREL Connected

The ROS console shows:

[ERROR] [1234567890.123456789]: Set buffer size: 1060, larger than MAX:, 1024

[ERROR] [1234567890.123456789]: Failed to load byte array

**Cause:** one possible cause is a difference in endianness between the robot and the PC running ROS. The Fanuc stack includes preconfigured launch files taking the endianness of the specific robot controller into account. If the actual controller uses a different endianness, the communication will not work properly.

**Remedy:** try to provide the launchfile with a different value for the use\_bswap argument. For most controllers, this is set to true. If appending use\_bswap:=false solves the problem, please report the issue to the mailinglist and / or the [ros-industrial/fanuc](https://github.com/ros-industrial/fanuc/issues) repository. Do not forget to include any relevant information (ROS release, version of [fanuc\_driver](http://wiki.ros.org/fanuc_driver), controller type and software and used manipulator).

### Unable to reconnect to the robot

The TP shows the following on the User Menu:

12345 I RSTA Waiting for ROS state prox

12345 I RREL Waiting for ROS traj relay

12345 I RSTA Connected

or

12345 I RREL Connected

The ROS console shows:

[ERROR] [1234567890.123456789]: Failed to connect to server, rc: -1. Error: 'Connection timed out' (errno: 110)

[..]

[ERROR] [1234567890.123456789]: Timeout connecting to robot controller. Send new motion command to retry.

**Cause:** this error can be caused by an incomplete and / or unexpected shutdown of the ROS-Industrial nodes on the PC, leaving the ROS-Industrial programs on the controller in an inconsistent state. In such a case, the socket on the controller is still waiting for a timeout to occur and blocks any new connection attempt in the meantime.

**Remedy:** either wait for the timeout on the controller to occur, or forcibly stop the ROS-Industrial programs on the controller using Fctn→ABORT (ALL). Now restart the programs on the controller, then restart the ROS nodes on the PC.

# fanuc\_driver\_exp

A new - experimental - Fanuc robot driver for ROS-Industrial.

## Overview

Main differences with fanuc\_driver:

* tries to adhere to commanded velocity constraints on trajectory segments (but will most likely fail)
* adds out-of-the-box support for linear joints / tracks
* 125 Hz / 83 Hz joint state reporting (controller maximum, depending on ITP)
* decouples network traffic processing from trajectory execution
* proper support for trajectory goal abort (ie: motion cancel using a SKIP condition), but see [issue 3](https://github.com/gavanderhoorn/fanuc_driver_exp/issues/3)
* adds a configurable length point-buffer to minimise influence of network latency between controller and ROS PC
* significantly more robust and flexible network code
* extensive use of rossum build infrastructure support (modularity, code reuse)

## Installation

Installation using a binary distribution is preferred, but building from sources is also supported.

## Binaries

Determine the system software version running on the target controller, then download the latest binary distribution from the [releases](https://github.com/gavanderhoorn/fanuc_driver_exp/releases) page.

Extract the zip file to a suitable location and follow the steps under [Copying](https://github.com/gavanderhoorn/fanuc_driver_exp/blob/master/README.md#copying).

## Source build

Note: the below steps use Dirk Thomas' [vcstool](https://github.com/dirk-thomas/vcstool) to clone all required repositories with a single command (vcstool can be installed under Windows using pip). This can also be done manually: just git clone .. each of the repositories listed in the repos.repos file.

In a [rossum](https://github.com/gavanderhoorn/rossum) workspace (on a Windows machine with Roboguide):

wget https://github.com/gavanderhoorn/fanuc\_driver\_exp/blob/master/repos.repos

vcs import --input repos.repos

mkdir build

cd build

rossum -r ..\fanuc\_driver\_exp\robot.ini ..

ninja

Now continue with [copying](https://github.com/gavanderhoorn/fanuc_driver_exp/blob/master/README.md#copying) the necessary files.

## Copying

Copy the following files to the controller:

* libindlog.pc
* libmathex.pc
* librosfanuc.pc
* libsm000a.pc
* libsm000b.pc
* libsm000d.pc
* libsm\_hdr.pc
* libssock.pc
* ros\_state.pc
* ros\_traj.pc
* ros.ls
* ros\_movesm.ls

## Configuration

Configuration is similar to that of fanuc\_driver. See [fanuc\_driver/wiki/cfg](http://wiki.ros.org/fanuc/Tutorials/hydro/Configuration) for more information (but note that fanuc\_driver\_exp has additional configuration entries which have not been documented yet).

Note that upon first start, both ROS\_STATE as well as ROS\_TRAJ will setup defaults. It is recommended to run the ROS TP program at least once, and only then edit the configuration of the Karel programs.

## Running

Start the ROS TP program, then follow the normal fanuc\_driver workflow.

## License

Unless stated otherwise, all code under the Apache 2.0 license.

Gijs latest Fanuc experimental release is '0.1.0-r1b':

> [1] https://github.com/gavanderhoorn/fanuc\_driver\_exp

> [2] https://github.com/gavanderhoorn/fanuc\_driver\_exp/releases

MOVEIT TEST WITH FANUC:

https://github.com/ros-industrial/fanuc\_experimental.git

1 cd /path/to/catkin\_ws/src

2

3 # retrieve the latest development version of industrial\_core. If you'd rather

4 # use the latest released version, replace 'indigo-devel' with 'indigo'

5 git clone -b indigo-devel https://github.com/ros-industrial/industrial\_core.git

6

7 # retrieve the latest development version of fanuc and fanuc\_experimental

8 # NOTE: you MUST use the '-devel' version of fanuc

9 git clone -b indigo-devel https://github.com/ros-industrial/fanuc.git

10 git clone -b indigo-devel https://github.com/ros-industrial/fanuc\_experimental.git

11

12 cd /path/to/catkin\_ws

13

14 # checking dependencies

15 rosdep install --from-paths src --ignore-src --rosdistro kinetic

16

17 # building

18 catkin\_make

19

20 # source this workspace (only if you don't have any others)

21 source /path/to/catkin\_ws/devel/setup.bash

roslaunch fanuc\_lrmate200id\_moveit\_config moveit\_planning\_execution.launch

sim:=false robot\_ip:=129.6.78.111