Training: software on the NIFTi platform

Francis Colas

ETHZ – ASL

30 May 2011

Introduction

Robot and drivers:

- tracks and flippers: `nifti_ros_drivers`,
- brake, status, laser motion: `nifti_ros_drivers`,
- laser: `LMS1xx`,
- omnicam: ???,
- IMU/GPS: `xsens_mtig`,
- joystick: `joy` (optional).

ROS nodes

Drivers

- `nifti_robot_node` (from `nifti_ros_drivers`[1]),
- `joy_node` (from `joy`),
- `LMS100` (from `LMS1xx`[2]),
- omnicam?
- `xsens_mtig` (soon on the `svn`);

Utilities

- `nifti_teleop_joy` (from `nifti_teleop`[3]),
- `mux` (from `topic_tools`).

---

[1] on the svn: `https://subversion.dfki.de/nifti/code/drivers`

[2] `$ git clone git://github.com/RCPRG-ros-pkg/RCPRG_laser_drivers.git`

[3] on the svn: `https://subversion.dfki.de/nifti/code/ros/stacks/nifti_misc/trunk/nifti_teleop`

nifti_robot_node

Generalities:

- handles controllers,
- outputs status,
- embeds lowlevel library,
- can embed rover simulation library.

nifti_robot_node

Topics

```
$ rosnode info nifti_robot_node
--------------------------------------------------------------------------------
Node [/nifti_robot_node]
Publications:
 * /rosout [roslib/Log]
 * /tf [tf/tfMessage]
 * /flippers_state [nifti_ros_drivers/FlippersState]
 * /robot_status [nifti_ros_drivers/RobotStatus]

Subscriptions:
 * /scanning_speed_cmd [std_msgs/Float64]
 * /flippers_cmd [nifti_ros_drivers/FlippersState]
 * /brake [std_msgs/Bool]
 * /cmd_vel [geometry_msgs/Twist]
 * /enable [std_msgs/Bool]
```

nifti_robot_node

FlippersState Message:

```
$ rosmsg show nifti_ros_drivers/FlippersState
float64 frontLeft
float64 frontRight
float64 rearLeft
float64 rearRight
```

nifti_robot_node

RobotStatus Message:

```
$ rosmsg show nifti_ros_drivers/RobotStatus
float64 battery_level
int32 battery_status
bool brake_on
float64 scanning_speed
nifti_ros_drivers/ControllersStatus controllers_status
  int32 core
  int32 track_left
  int32 track_right
  int32 flipper_front_left
  int32 flipper_front_right
  int32 flipper_rear_left
  int32 flipper_rear_right
```

nifti_teleop_joy

Controls

- b2 deadman button: nothing happens if not pressed down,
- a0–1 (left joystick) velocity control,
- b1 run mode: moves faster when pressed down,
- b5–8 flipper select: if pressed the flipper can be moved with:
  - a5 (cross up/down) move selected flippers up or down,
  - a4 (cross left/right) change laser rolling speed,
  - b9 enable button: send an enable command to the controllers (soft reset to go out of fault mode),
  - b10 brake button: toggle differential brake.

Mux

Purpose

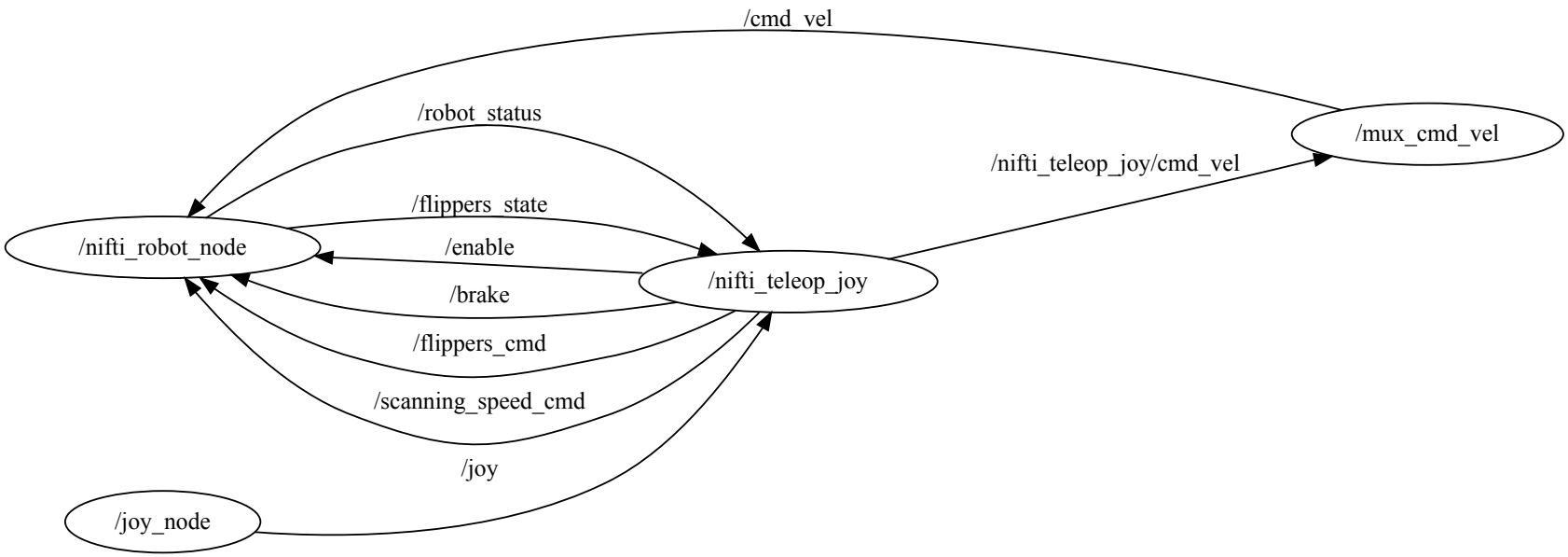
- having both some joystick control and navigation,
- multiplexes both velocity topics,
- choice based on deadman button;

Implementation

- `topic_tools/mux` node,
- output: `/cmd_vel`,
- inputs: `/nifti_teleop_joy/cmd_vel` and `/nav/cmd_vel`,
- selection by the `nifti_teleop_joy` node.

rxgraph

```
$ roslaunch nifti_teleop nifti_teleop_joy.launch
```



Getting documentation

Doxygen

- `$ rosrun rosdoc rosdoc nifti_ros_drivers`
- open `doc/nifti_ros_drivers/html/index.html`
- `$ rosrun rosdoc rosdoc nifti_teleop`
- open `doc/nifti_teleop/html/index.html`
- code is fairly commented: read it!

Conclusion and outlook

Current state:

- works both in simulation and on the robot,
- tested nearly 2 hours on the robot,
- documented;

To do:

- checking controller errors,
- implementing diagnostics (deciding what),
- parameters?
- (opt.) better flippers management?
- diamondback support, tf2?, ...