nifti_teleop Documentation

nifti_teleop: nifti_teleop

Teleoperation package for the NIFTi robot. Currently only proposes a node to teleoperate the robot with a joystick.

- Homepage: http://ros.org/wiki/nifti_teleop

nifti_teleop is the package of teleoperation of the NIFTi robot.

The main node, **nifti_teleop_joy**, allows the user to control the robot with a joystick. This node just instantiates the **nifti_teleop_joy::NiftiTeleopJoy** class.

This package also provides the **nifti_mux_control** (**nifti_mux_ctrl::NiftiMuxCtrl**) and **priority_wrapper** (**priority_wrapper::PriorityWrapper**) nodes to handle the priorities for the different /cmd_vel publishers.

Additionally, **nifti_teleop_helper** eases the process of taking a single 3D scan (see **nth_action::ScanningService**) and controling flipper configuration (see **nth_action::FlipperPosture**).

The launch file to be launched is nifti_ugv_teleop.launch on the robot or nifti_joy.launch on any remote computer with a joystick.

Controls

The **nifti_teleop_joy.py** node allows to pilot the robot with a joystick. All commands require one of the **deadman** buttons to be pressed to be taken into account. All buttons and axes can be changed via parameters (for different joysticks typically, see below).

- deadman button: '2' or 'A' (secondary: button of the left thumbstick)
- driving: left thumbstick: up/down for linear velocity and left/right for angular velocity;
- driving faster: pressing '1' or 'X' while driving;
- moving flippers: up and down with the left cross for the selected flippers:
    - front left: '5' or 'LB',
    - front right: '6' or 'RB',
    - rear left: '7' or 'LT',
    - rear right: '8' or 'BT';
- asking for the flippers to go in their flat position: '4' or 'Y';
- changing scanning speed: left/right on the left cross (if you decrease again after stopping it will come back to the center);
- re-enabling motor controller: '10' or 'Start' (useful when a flipper is disabled typically);
- toggling the brake of the differential: '8' or 'Back';
- taking a 3D scan: '3' or 'B'.

You can also use the right thumbstick to move the flippers into predefined positions by moving it then pressing its button:

- up: 'driving position': rear flippers vertical and front flippers folded for maximum field of view;

- right: 'forward obstacle': rear flippers flat and front flippers 45 degrees up to climb an obstacle;
- left: 'backward obstacle': front flippers flat and rear flippers 45 degrees up to climb down from an obstacle or to climb backwards;
- down: 'convex edge': front and back flippers 40 degrees into the ground to smoothly overcome an edge.

Parameters

The parameters of the **nifti_teleop_joy** node can be seen in the **nifti_teleop_joy::NiftiTeleopJoy** class. They can be broken down in 3 groups.

Motion parameters

The parameters to define the motion limits:

- *max_linear* (`0.3` m/s): maximum linear velocity of the standard motion,
- *max_angular* (`0.6` rad/s): maximum angular velocity of the standard motion,
- *max_linear_run* (`0.6` m/s): maximum linear velocity with the run buttun,
- *max_angular_run* (`1.24` rad/s): maximum angular velocity with the run button,
- *max_scanning_speed* (`1.2` rad/s): maximum angular velocity of the laser,
- *tracks_distance* (`0.397` m): distance between tracks (used to not ask for too high velocities),
- *steering_efficiency* (`0.41`): steering efficiency parameter of the kinematic model (used to not ask for too high velocities),
- *~scanning_speed_increment* (`0.2` rad/s): increment in scanning speed control,
- *~flipper_increment* (`20*pi/180` rad): increment in the flippers motion,
- *~scanning_once_speed* (`1.2` rad/s): speed for a single 3D scan.

Joystick parameters

These parameters define which button or axis for each command mode (take care that the numbers start from 0 here and from 1 in the section above):

- *~axis_linear* (`1`): axis of the linear velocity command,
- *~axis_angular* (`0`): axis of the angular velocity command,
- *~flipper_axis* (`5`): axis of the flipper command,
- *~scanning_speed_axis* (`4`): axis of the scanning speed command,
- *~right_ts_lr_axis* (`2`): left-right axis of right thumbstick,
- *~right_ts_ud_axis* (`3`): up-down axis of right thumbstick,
- *~deadman_buttons* (`[1, 10]`): deadman buttons,
- *~run_button* (`0`): to move faster,
- *~enable_button* (`9`): enables all motor controllers,
- *~differential_brake_button* (`8`): toggles the differential brake,
- *~flipper_reset_button* (`3`): moves flipper in flat position,
- *~front_left_flipper_button* (`4`): select the front left flipper to move,
- *~front_right_flipper_button* (`5`): select the front right flipper to move,
- *~rear_left_flipper_button* (`6`): select the rear left flipper to move,
- *~rear_right_flipper_button* (`7`): select the rear right flipper to move,
- *~scanning_once_button* (`2`): button to launch a single scan,

- *~right_ts_button* (`11`): button of right thumbstick for flipper command.

Misc parameters

- *~cmd_vel_topic* (`/teleop_joy/cmd_vel`) the topic to publish velocity commands.

Priority control

The handling of priorities is done by the **nifti_mux_control** (**nifti_mux_ctrl::NiftiMuxCtrl**) node that commands a standard **mux** node. In order to publish velocity commands, one must end a request to that node (on the `/mux_cmd_vel/acquire` service) and can then publish on its own defined topic. When a node is done sending commands, it must release the control (so that lower-priority processes can control the robot) using a second service (`/mux_cmd_vel/release`).

The topics are, with decreasing priority:

- /local_joy/cmd_vel (for the joystick on board the robot)
- /teleop_joy/cmd_vel (for all other joysticks)
- /teleop_ocu/cmd_vel (for the OCU)
- /nav/cmd_vel (for autonomous navigation)

For external packages, the **priority_wrapper** (**priority_wrapper::PriorityWrapper**) node can be used to do this acquisition and release automatically. The drawback in that case is that the node sending the commands is not aware of whether or not his commands are passed.

In order to know which is the topic currently having access to the robot, one can listent to the `/mux_cmd_vel/selected` topic.