Table of Contents

Welcome

Project Willy

- History of Willy
- Project Willy
- Publicity
- Sponsors

Getting started

- Development Guide
- Driving Willy
- Documentation

Build of Willy

- Design history
- Requirements
- Design reference
- Physical build
- Hardware

Robotic Operating System

- Introduction to ROS
- ROS Tutorials
- Multi master

Architecture

- Software Architecture
- Hardware Architecture
- ROS topic design

Hardware nodes

- sensor node
- si node

1. Motor controller

The motor controller listens to the drive topic and controls the motors of Willy.

1.1. Repository

Windesheim-Willy/motor_controller

1.2. How P&G Pilot Plus works

The communication between the P&G (Penny and Giles) Joystick and Motorcontroller is done with a serial connection. Two variables in within the data packet contain the throttle (drive) and direction (turn). Both are signed integers with an limit of -100/100. They correspond with the 360 degrees position of the joystick.

1.2.1. Arduino reading /cmd_vel messages from ROS

The geometry twist messages broadcast by ROS will be translated by the Arduino motor controller to match the serial communication done by the Joystick controller from P&G. The data send from ROS will be multiplied with 100 and capped on their maximum value. Only two axes of the message are being used.

```
    throttle = twistMsg->linear.x * 100;
    direction = twistMsg->angular.z * 100;
```

The P&G Joystick emits a datagram package that the Willy motor controller emulates. Within the datagram there is a forward/back integer and a direction (left/right) integer. Those are calculated from the twistMsg to the respective P&G values.

1.3. Broadcasting to the emergency channel

Every iteration within the critical loop of the state of the emergency button will be broadcast. The motor controller will interrupt the circuits on a hardware level when the button is pushed. No software has been written to do any logic with the state of the button other than broadcasting its state.

1.4. How to compile?

All the dependencies (headers from ros) are copied from the Kinetic Kame release into the project. You'll need to the following to compile and flash the Arduino:

1. VSCode with Platform.io extension installed
2. Arduino MEGA

When building for the first the Platform.io will add some files in the .vscode directory that can be ignored. Its not recommended to add them to GiT. The files have hard-coded paths containing the username and are ignored. To flash the Arduino simply press Upload in the Plaform.io extension.