Electric Kart

Max-Felix Müller

2020
December

Contents

List of Figures

1 Introduction

After finding a used hoverboard at the loca trash, a rebuild of the Electric Wheelbarrow was in order. Using the new motors the output popwer could be increased. Also with two motors differential steering would be possible.

2 The Hoverboard

The hoverboard could easily be found on amazon. For more details, check the "Hoverboard Disassembly" project.

The hoverboard includes two 350 W motors.

Figure 1: Amazon Product Picture

3 O-Drive Update Problems and Pull Request

After saving the O-Drive configuration of the Electric Wheelbarrow, the O-Drive firmware should be updated. The latest version is 5.1 and on the O-Drive the last version was 3.6 so it was time for an update.

Simply calling the update was not enough though, because the flashing resulted in an error in some python module. The first intetion was unplugging the O-Drive, which was a mistake, because the O-Drive firmware was erased, but no new firmware was flashed. That meant, that the O-Drive was no longer found by the Odrivetool to update.

One possible workaround would have been flasing the firmware using an ST-Link programmer directly. Luckily there is also a DIP-switch for switching between run and dfu (device firmware update) mode on the PCB.

However, the error in the python module remained. As it turns out, the problem was the use of a deprecated function fractions.gcd. The solution for this problem was to modify the code in such a way that the function math.gcd is used instead. This is also the recommended fix.

After creating a pull request on the official O-Drive Github, I am now a contributor to the project. The pull request with the changes made can also be found in the "Links" section.

4 Testing the Motors

The reason why the hoverboard landed in the trash was unknown, so the first step was testing the motors for functionallity.

4.1 Mechanical Approach to Motor Testing

A simple test that can be performed easily is turning the motor. No blocking of the rotation is a good indication. Also there was no crunching sound, meaning there is no dust or dirt inside the motor.

4.2 Checking for Broken Wiring inside the Motor

The next test is to connect all three phases of the motor. Each phase to one another and all phases together. When two phases are connected, the motor should turn in steps, similar to a stepper motor only much coarser. Connecting all three phases should result in a continuously smooth, but harder to turn feeling. That is because the current induced by rotating the magnets on the rotator opposes the turning motion.

Another test that can be performed is using a multimeter. All three phases should measure similar resistance.

With this tests the wiring can be checked. It is possible that the motor wires are broken inside the hub motor, which could not be seen from the outside.

4.3 Powering the Motor

After testing the motor with those simple tests it is time to power it up. Using the O-Drive this can be done quite simple.

With the settings from the Electric Wheelbarrow the motors can be driven in an open control loop using the velocity control mode.

First the motors have to be calibrated for their specific resistance and inductance. With these motors, the calibration limits have to be changed. This is due to the high internal resistance of hoverboard motors.

```
odrv0.axis0.motor.config.resistance_calib_max_voltage
    = 10.0
odrv0.axis0.requested_state
    = AXIS_STATE_MOTOR_CALIBRATION
```

Both motors calibrated fine. So swithcing into the right control mode and the setting a velocity is the next step.

```
odrv0.axis0.controller.config.control_mode
    = CONTROL_MODE_VELOCITY_CONTROL
odrv0.axis0.motor.config.direction
    = 1
odrv0.axis0.requested_state
    = AXIS_STATE_SENSORLESS_CONTROL
odrv0.axis0.controller.input_vel
    = 1
odrv0.axis0.requested_state
```

```
    = AXIS_STATE_IDLE
```

For some reason, this did not quite work. At least the motor made some attempts to rotate, befor the O-Drive threw some errors. But it was enough to keep going for now.

4.4 Testing the Motor Encoder

The motor has five additional wires which connect to the hall sensors inside. The wires can be differentiated by the thinner wiring.

The five connections are:

- 5 V
- Sensor 1
- Sensor 2
- Sensor 3
- Ground

The connections to the O-Drive are described on their documentation page about hoverboard motors. That page turned out to be very useful for the whole project.

Also from that documentation the configuration for the motors can be copied:

```
odrv0.axis0.motor.config.pole_pairs
    = 15
odrv0.axis0.motor.config.current_control_bandwidth
    = 100
```

The encoder:

```
odrv0.axis0.encoder.config.mode
    = ENCODER_MODE_HALL
odrv0.axis0.encoder.config.cpr
    = 90
odrv0.axis0.encoder.config.bandwidth
    = 100
```

And the controller:

```
odrv0.axis0.controller.config.pos_gain
    = 1
odrv0.axis0.controller.config.vel_gain
    = 0.02 *
    odrv0.axis0.motor.config.torque_constant *
    odrv0.axis0.encoder.config.cpr
odrv0.axis0.controller.config.vel_integrator_gain
    = 0.1 *
    odrv0.axis0.motor.config.torque_constant *
    odrv0.axis0.encoder.config.cpr
odrv0.axis0.controller.config.vel_limit
    = 10
```

```
odrv0.axis0.controller.config.control_mode
    = CONTROL_MODE_VELOCITY_CONTROL
```

All these settings have to be made for both axis (aka both motors). Then save and reboot.

```
odrv0.save_configuration()
odrv0.reboot()
```

After those settings are done, the calibration process can be started. This time, the full calibration sequence is executed. This will calibrate the motor as well as the encoder.

```
drv0.axis0.requested_state
    = AXIS_STATE_FULL_CALIBRATION_SEQUENCE
```

After a successful calibration, the values are saved. This is done by setting both the motor and the encoder to precalibrated.

```
odrv0.axis0.motor.config.pre_calibrated
    = True
odrv0.axis0.encoder.config.pre_calibrated
    = True
```

Save and reboot.

After all this setup, it is finally time to test the motors. On a positive note: If the motors are fine, the calibration sequence and the rest of the setup will not have to be repeated.

Compared to the sensorless motor test without the encoder from the previous chapter, now the closed control loop is chosen.

```
odrv0.axis0.requested_state
    = AXIS_STATE_CLOSED_LOOP_CONTROL
odrv0.axis0.controller.input_vel
    = 1
odrv0.axis0.controller.input_vel
    = 0
odrv0.axis0.requested_state
    = AXIS_STATE_IDLE
```

Both motors were turning and reacting correctly to velocity changes. Applying a braking force by hand also wokred fine.

When the velocity was set to zero and the wheels were turned, they resisted the turning (as long as the current limit would allow it). After releasing them the wheel would spin back to their original position.

5 Build

5.1 Building a Test Platform

Holding the motors while at the same time entering the commands in the command line interface of the Odrivetool seems unexpertly. Instead a test stand was build where both motors could be mounted. The test stand was build before even testing the motors, so it should not be too complicated in case one or both of the motors turned out to be malfunctioning. At the same time, the test stand should be good enough to later integrate into the final build to avoid the double effort.

Different YouTube videos show people building a test stand comprising of aluminium extrusion where the motors are mounted. It was unclear if aluminium extrusion bars could hold the wieght of a person, so a more robust solution was necessary. Using an old wooden plank of a bed frame, both motors where mounted.

The wood was cut to a length that is about the width of the body of the wheelbarrow. The plan was to use it as some sort of seat.

To mount the motors, the original aluminium casted frame of the hoverboard was used. The center axis was removed and it's mounting points cut off. The frame was then bolted to the wood using some threaded rod and self locking nuts.

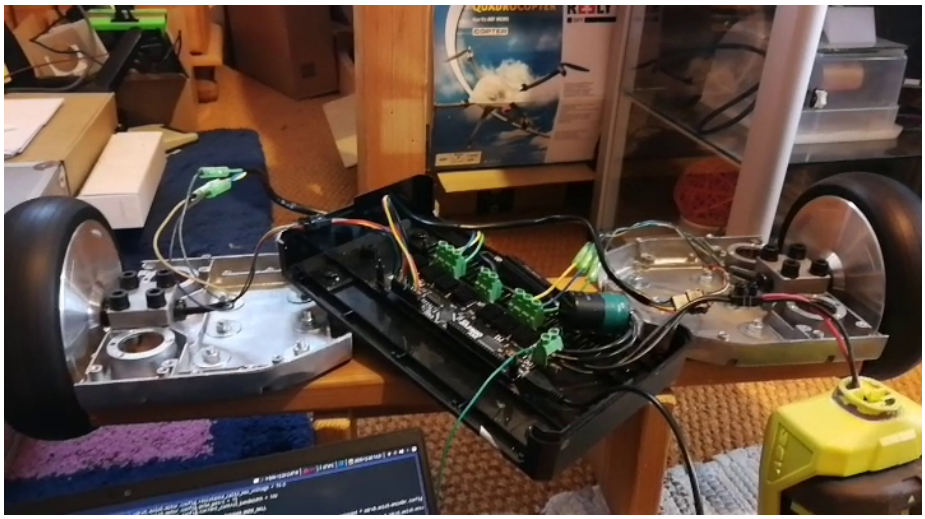

Figure 2: Test Stand for the Motors

With this test stand the motors can be tested. Later, the construction can simply be used as the front or rear axle of the kart.

5.2 Building of the Frame

For the frame of the kart, more wood was used. The longest piece was cut in half, which determined the length of the frame. It was a little longer than the body of the wheelbarrow, so balance should not be a problem.

One axle was build by mounting the test stand to those longer planks. Half of the threaded rods were replaced with longer ones. Those penetrated through both layers of wood. Addiditionally, some wodd screws were used to hold the axis to the rest of the frame. Using a right angle, the frame was buuild as straight as possible.

For the front axle, another piece of wood was cut to the same length as the rear axle. Then some swivelling rollers were mounted. The screws securing the swivelling rollers also hold the wood pieces together.

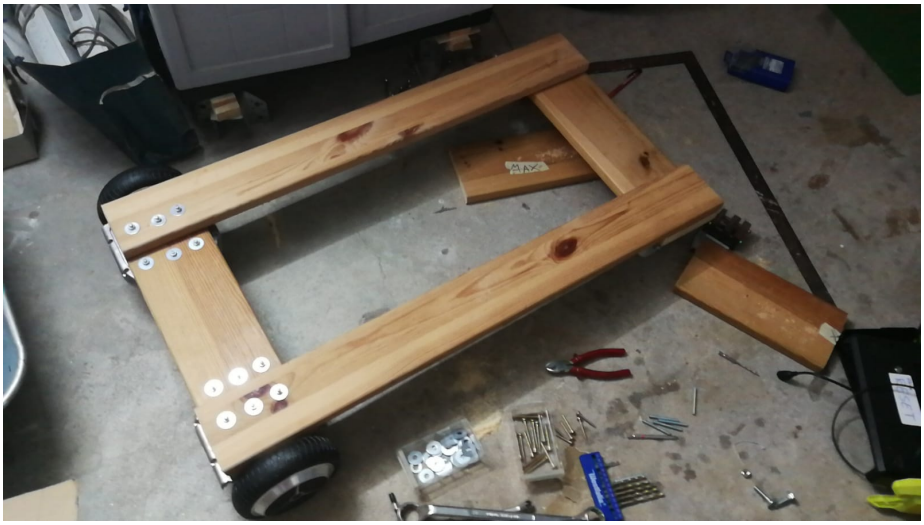

Figure 3: Frame of the Kart

5.3 Building of the Electronics

Technically the electronics were not build, but only assembled. A Raspberry Pi Zero W was used for its small form factor. The Raspberry connects to the O-Drive via the USB interface and the proprietary protocoll. It is powered directly from the 5 V supply of the O-Drive, which can deliver up to 500 mA of which the Raspberry uses less then 250 mA under full load. To program the Raspberry a VNC server is running on it. The Raspberry connects to the home WiFi and can be accessed from there. A Raspberry was used, because the interface to the O-Drive is simpler than using an Arduino. Addiditionally the bluetooth of the Raspberry was used to connect to a PS5 controller for steering. A small python script is running which converts the control inputs to velocity commands for the O-Drive.

For wiring the motor on one side, the original cables could be used. For the second side those were not long enough, so instead the standard 1.5 mm house wiring was used as power and five random wires for the encoders. The wires are guided thorugh some holes in the wood to the other side. All wires were secured with hot glue to prevent them from unplugging under the vibrations from the wheels on the street.

The most important thing was the Reset button for the O-Drive. It connects directly to the programming header, so the reset pin of the microcontroller itself. The button was mounted on the outside of the electronics enclosure, easily accessible.

Figure 4: Reset Button for the O-Drive

The reset button is very important. When the kart is drinving and the controller disconnects, anything could happen. Also it is unclear what would happen if the Raspberry shuts down. Probably, because there are no new commands, the O-Drive would just continue with the last input. To prevent a run-away (literally) the Reset button was mounted.

5.4 Final Assembly

The last step was to bring the electronics, the frame and the wheelbarrow body together. To mount the electronics the same mounting points as in the "Electric Wheelbarrow" project were used. Those were already there and tested, so no problems.

To keep the wire length short, that also fixed the orientation of the body in relation to the frame. The motor axle had to be close to the electronics. That meant that the steeper part of the wheelbarrow body was above the driven tires. The originally intended way of driving was to lean against the flatter part, because that would be more compfortable. Thus the person would sit above the swivelling rollers.

Figure 5: More or less completed Kart

The body of the wheelbarrow had a slight angle on the bottom. That would lead to it not mounting straight on the wooden frame. To compensate, some additional block of wood were used between the body and the frame.

6 Testruns

My brother driving the kart in one of the first test runs.

Figure 6: One of the first Test Runs

6.1 Heavy Drifting

On initial testruns the kart was drifting all the time, every time. No chance of driving in a straight line for more then 5 meter. Also the driving wheels with the included hub motor were slipping at every acceleration, degrading the rubber surface very quickly.

The fix was turning the whole kart around (or just the controls and the seating posiiton). That way, the driver's weight is concentrated more above the rear wheel which are the driven ones now.

6.2 Velocity Control or Position Control

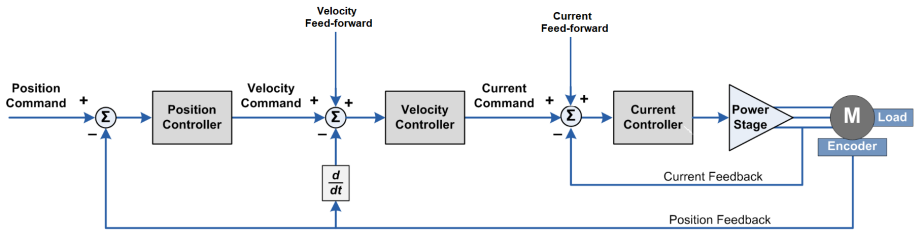

Figure 7: Block Diagram of the O-Drive Controller from the Documentation

The O-Drive documentation shows a block diagram of the motor controller. The innter control loop is the current control. It limits the current flowing from the battery to the O-Drive.

Next is the velocity control. The veolcity has to equal the input velocity. Using the velocity control control mode, the position control (which would come in front) is disabled.

The problem is, that when stopping the kart, the motors would then turn backwards for a short moment, seemingly compensating for the time it took them to come to a stop. That would mean, that the position control is still active, which it should not be.

Currently there is no solution, but the problem is known. It is no big deal, just inputting another quick forward command after breaking stops the backwards motion.

6.3 Overcurrent

During some longer runs, one of the motors would cut out and no longer be controlable via the remote control. The problem occured the first time on a parking lot away from home, so there was no way to log onto the Raspberry to troubleshoot. Because the problem kept repeating, the kart was taken back home and tested. It was not possible to recreate the error while braking by hand. When placing the kart on the ground and inputting some aggressive turns, at some point one wheel would stop responding again.

It was then possible to connect to the VNC server, kill the python script and start the Odrivetool. Dumping the errors showed an overcurrent on one of the motors. The quickest fix was allowing more current to flow.

```
odrv0.axis0.motor.config.current_lim
    = 50 // instead of 20
odrv0.axis0.motor.config.current_lim_margin
    = 15 // instead of 8
```

6.4 Battery Problems

After the overcurrent fix, the next problem occured. Now both motors would shut off. Addiditionally the battery was showing no lights when checking its charge level and all indicator lights on the O-Drive and the Raspberry would be off.

After some thinking the conclusion we drew was, that the current that was flowing was so high, that the battery voltage dropped so low, that the internal protection of the battery would cut off. That also explains why unplugging and replugging the battery would fix the problem temporarily.

As a long term fix, only a stronger battery or re-enabling the current limits were possible. Because the budget is very tight, the current limit was reduced to its original values. That also means, that the overcurrent problem was there once again.

```
dump_errors(odrv0, True)
```

A very unelegant fix, but as soon became obvious, mainstream error handling in the automotive sector, is to clear the error as soon as it appears. To do so, the python script just dumps the error in every loop and discards it. That is common practice in the automotive industry. If an error occurs, that is not significant enough, just clear it. To be professional, the error would indeed be

stored in some log file, so that later it can be checked which errors did occur, but this project does not try to be professional.

6.5 Nosie Level

The noise level of the kart is quite high. This is a result of the wheelbarrow body acting as a resonant chamber and the hard plastic wheels ratteling along the street surface. Two fixes are planned (due to the pandemic shopping is currently not possible).

First the wheels will be changed to a softer rubber compound. That should already reduce the noise level significantly.

As the next step, the wheelbarrow body will be mounted on old cloth pieces dampening it's vibrations.

7 Final Thoughts

What a fun project. I can not wait for better weather to once again drive with the kart. The noise level has to fixed as soon as possible, but I think the planned fixes will already be a large step in the right direction.

The project was a great learning opportunity. I got a deeper insight in the firmware and the programming tools of the O-Drive. Also the error handling and problem fixing at the end of the project was filled with learning opportunities.

The mechanical stability is better then expected. The father of a friend of mine tested the kart as well and I expected it to break, but it did not.

Here is another picture that would not fit elsewhere. It is from the first time I controlled the O-Drive via the PS5 controller.

Figure 8: First Time controlling with PS5 Controller

8 Links

O-Drive website: https://odriverobotics.com/
O-Drive on Github: https://github.com/odriverobotics/ODrive
O-Drive Documentation to Hoverboard Motors: https://docs.odriverobotics.com/hoverboard
My pull request: https://github.com/odriverobotics/ODrive/pull/542