Evaluation of OSMC open source motor driver for reproducible robotics research

Elijah Alabi[1], Fanta Camara[1,2] and Charles Fox[1]

[1] School of Computer Science, University of Lincoln
[2] Institute for Safe Autonomy, University of York.

Abstract. There is a growing need for open source hardware (OSH) subcomponents to be evaluated. Most robotic systems are ultimately based upon motors which are driven to move either to certain positions, as in robot arms, or to certain velocities, as in wheeled mobile robots. We evaluate a state of the art OSH driver, OSMC, for such systems, and contribute new Open Source Software to control it. Our findings suggest that OSMC is now mature enough to replace closed-source motor drivers in medium-size robots such as agri-robots and last mile delivery vehicles.

1 Introduction

Robotics research currently struggles with reproducibility. While open source software (OSS) such as ROS enables software reuse, this is not so for hardware, which is proprietary and differs between labs. Researchers thus waste significant time porting software to run on their different hardware platforms. Researchers in developing counties often cannot afford the proprietary robots used by other labs. Open source hardware (OSH) is hardware whose designs and build instructions are made public, easy, and low-cost so that anyone may build and modify, enabling large community collaborations to grow. Fully open software and hardware stacks would allow any researcher to download, build, exactly replicate, then extend the published work which they read.

Recent 'shallow' definitions of OSH such as the 2020 CERN-OSH licences [4] do not require designs to be made up of OSH subcomponents, but allow closed source subcomponents if available on the open market. 'Deep' definitions such by Open Source Ecology (OSE) [7] further restrict subcomponents, recursively, to be all OSH, so that entire designs are open down to the level of ISO standard nuts, bolts, resistors and transistors. 'Deeper OSH' has been proposed [6] as the process of successively replacing lower level subcomponents of shallow OSH designs with OSH alternatives, working towards deep OSH. To enable this process, there is a need for standard subcomponents to be not only created as OSH but also to be publicly evaluated to scientific standards. A recurring difficulty is that while some OSH designs are created and published through academic peer-review, others emerge from the maker community outside the academic system. Maker designs may be of high quality but additional peer-reviewed evaluations (sometimes known as 'blessings') are then needed to create specifications and/or create trust that the specifications are met.

Motor drivers are a common subcomponent so an important target for deeper OSH. We here perform the evaluation of a maker community OSH motor driver, the OSMC (Open Source Motor Controller; *sic.*—it is a driver not a controller), for this purpose. OSMC is a brushed DC driver designed by hobbyists in 2001 for use in *Robot Wars* style, manually remote controlled combat robots. OSMC as been used [8] for a telecontrolled service weeding robot in outdoor environment; in an underwater vehicle [1]; in the RoboCup Rescue challenge [2]; and in the manual control [9] "Battle-Bot" combat robot. However these projects are not open source hardware and do not provide evaluations of the OSMC other than reporting its use in the larger systems. OSMC has been released as a hardware design but does not include standard software to control it. This paper aims to address these gaps by evaluating OSMC, and by releasing new open software to control it, available at `https://github.com/Pelex-a/osh-motor-control`.

2 Methods

We evaluate OSMC for use in both 'small-size' (capabale of motion and information gathering) robots such as RC cars, and 'medium sized' (capable of actuating physical work) robots such as agri-robots and last-mile delivery vehicles [3].

The small-size evaluation setup is made up of the L298N driver module, a DC motor, an incremental encoder and a means of serial port communication, with new Arduino software for PID control as shown in fig. 1.

The medium-size evaulation uses OSMC with two example motors. The motors used were a 73ZYT-155-24-MCP4-25:1 rotary DC motor (Motion Control Product Ltd) shown in Fig. 2 and a GLA750-P 12V DC linear actuator with potentiometer feedback shown in Fig 3. The two main wires of the motor are connected to the 'MOT+' and 'MOT-' leads on the OSMC driver. For directional and PWM control, AHI, BHI, ALI, BLI and Disable pins are connected to digital pins of an Arduino board. The OSMC is then connected to the DC power source. An external incremental encoder is connected to the Arduino. The linear actuator has the same connection from the OSMC to the microcontroller board. However for control, we connected the potentiometer wiper, which is one of the three leads of the potentiometer in our linear actuator, to the analogue pin of the Arduino board. The two other leads, the potentiometer reference, are connected to the ground and reference voltage pin on the Arduino board.

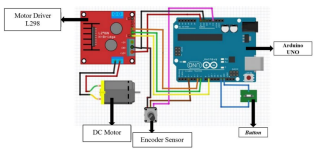

Fig. 1: Hardware wiring

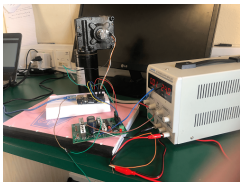

Fig. 2: Rotary setup

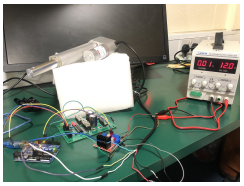

Fig. 3: Linear setup

3 Results

An open-loop test was conducted to observe the system response of the small-scale set-up by monitoring the revolution made per minute with a constant input voltage in PWM. The system rises initially from rest and settles at a specific RPM over time. It is noticed that the open loop system does not react to disturbance or change in condition. For this reason, a system with feedback – a closed system – is needed to efficiently carry out the desired task and to ensure the targeted value is reached.

Proportional-Integral-Derivative (PID) control is widely used in closed loop systems. We test PID control to evaluate system performance responding to angular position and speed requests. Videos of the tests are at `https://bit.ly/3qrC4JL` and `https://bit.ly/3waN9zG`. They show high degrees of positional and speed accuracy ($Accuracy(\%) = |\frac{set\ value - observed\ position}{set\ value}|\ X\ 100$) as seen in Tables 1 and 2. Each of the three control techniques has varying effect on the system as shown in Table 3, 4 and 5.

Table 1: Rotary performance

Set Value (mm)	Motor Position (mm)	Accuracy (%)
90°	91°	98.89%
180°	182°	98.89%
270°	272°	99.23%
360°	363°	99.17%

Table 2: Linear performance

Set Value (mm)	Actuator Pos (mm)	Accuracy (%)
10	7	70
25	23	92
40	39	97.5
70	68	97.1
85	83	97.6

Increasing the proportional (P) action in the control system reduces the steady-state error. Adjusting the integral (I) constant effect changes by eliminating the residual steady-state error. Although, the system becomes unstable and vibrate aggressively when set too high. Lastly, the derivative (D) action controls overshooting and rise time. These recordings (`https://bit.ly/3EN65HP` & `https://bit.ly/3wvRv4L`) shows how the system performed.

4 Discussion

The evaluation suggests that OSMC is ready for practical use in medium-size robots such as last-mile delivery vehicles and agri-robots. We have released new open source Arduino control software for OSMC, which now makes OSMC plug-and-play. This enables other community members to focus next on creating, building, and releasing new OSH robots using OSMC as a subcomponent. For example, OSMC is now being integrated into a general purpose medium sized robot control board [5]. Future work could extend OSMC with regenerative braking,

Table 3: System Response, P

Kp	**Ki**	**Kd**	Rise Time	Steady-state error
2	0	0	-	-
3	0	0	-	-
5	0	0	5.2	26
8	0	0	4.1	19
10	0	0	3.7	12

Table 4: System Response, I

Kp	**Ki**	**Kd**	Rise Time	Over-shoot	Steady-state error
10	0	0	3.7	-	12
10	2	0	3.4	-	0
10	3	0	3.2	4	0
10	4	0	2.9	11	3
10	5	0	2.8	19	5

Table 5: System Response, D

Kp	**Ki**	**Kd**	Rise Time	Over-shoot	Steady-state error
10	4	0	2.9	11	3
10	4	2	2.85	7	1
10	4	3	2.81	5	0
10	4	4	2.77	5	3
10	4	5	2.75	3	2

as found in many closed drivers. OSMC requires some closed subcomponents – including the Intersil H-bridge driver chip – which could be replaced with deeper OSH alternatives. OSMC is a brushed driver, but maker OSH Brushless drivers such as ODrive[1] v3.5 (but not later versions) and OpenBLDC[2] are similar in power to OSMC and could be similarly evaulated.

References

1. Aldehayyat, Y., Dahan, R., Fayyad, I., Martin, J., Perkins, M., Sharples, R.: Ocra-xi: An autonomous underwater vehicle. OCRA AUV Team (2009)
2. Angeles, D., Arroyo, G., Culebro, J., Espinoza, J.L., Gutierrez, H., Lara, M., Minami, Y., Muñoz, S., Ramos, G.: Robocup rescue 2014, team UNAM.
3. Camara, F., Waltham, C., Churchill, G., Fox, C.: OpenPodcar: an open source vehicle for self-driving car research. Jounral of Open Hardware (in press, preprint arXiv:220504454)
4. CERN: CERN Open Hardware Licence `https://cern-ohl.web.cern.ch/`
5. Gaikwad, K., Soni, R., Fox, C., Waltham, C.: Open source hardware robotics interfacing board. In: TAROS (2023)
6. Henry, A., Fox, C.: Open source hardware automated guitar player. ICMC (2021)
7. Jakubowski, M.: Open Source Ecology. `www.opensourceecology.org/` (2003)
8. Khan, M.I.: A framework for telecontrolled service robots. Master's thesis (2008)
9. Warren, J.D., Adams, J., Molle, H.: The Battle-Bot, pp. 513–562. Apress (2011)

[1] `www.github.com/madcowswe/ODriveHardware/blob/master/v3/v3.5docs/schematic_v3.5.pdf`

[2] `www.open-bldc.org/wiki/Open-BLDC`