

Development of Open-Source Motor Controller Framework for Robotic Applications

DONGIL CHOI¹, (Member, IEEE)

Department of Mechanical Engineering, Myongji University, Seoul 17058, South Korea

e-mail: dongilc@mju.ac.kr

This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2019R1G1A1008726), and in part by the 2018 Research Fund of Myongji University. 10.13039/501100003725-National Research Foundation of Korea and 10.13039/501100002538-Myongji University.

ABSTRACT Open-source designs are being increasingly used to develop robot hardware. When motor actuators were used for robot development in the past, it was difficult to find cost-effective commercial products. Open-source motor controllers, recently developed with the collaboration of many people, have exhibited phenomenal growth as their performance is now comparable to those of commercial products and their price is cheaper. In this study, we utilized an open-source Vedder electronic speed controller (VESC), originally developed for electric skateboards, as a general-purpose motor controller for robot control. The high-speed, high-torque, and high-output VESC was hardware-upgraded to enable more accurate current control, and advanced position and speed control algorithms were developed to produce Myongji-VESC (MJ-VESC), a general-purpose motor controller for multi-axis robot systems. To evaluate its current, speed, and position control performances, MJ-VESC was applied to an open-source planetary geared actuator that can be fabricated with a 3D printer. Based on this, we propose an open-source motor controller framework that can be universally used for multi-axis robot systems.

INDEX TERMS Intelligent actuators, open-source hardware, open-source software, robot control.

I. INTRODUCTION

Open-source hardware, released in various forms, has accelerated the development and growth of entire technological fields including robotics [1]–[4]. Open-sourcing is the process of releasing the detailed information generated during the development of products or technologies, which includes source codes and hardware design documents. Its purpose is to accelerate the sharing of technology and promote unrestricted cooperation among developers. Monopolizing technologies through patents was a trend in the past. However, currently, because of opening of source codes, anyone can easily access technologies and rapidly spread them to accelerate their growth. Open-source products are also being used to secure the market and standards by expanding the range of users and developers.

The concept of open-source emerged from the software development field. A representative example is the Linux operating system (OS) [5], which released source codes from the development stage at no cost and built the system through multilateral development. Initially, Linux had many shortcomings compared with the Windows OS; however,

the benefits of being free and allowing users to contribute to its development increased the number of users and improved its performance. Currently, Linux is an established and widely used computer OS. Another example is the Android OS used in smartphones, which has been developed based on the Linux OS. The advantages of being free and open-source enabled Android to support various hardware. Consequently, the market share of Android phones in the global market is 76% (as of October 2019), which is much higher than that of iOS (22%) [6].

Open-source has a significant impact on the hardware field and on the software field. While software, being formless code, can be easily spread and shared through files, only limited design documents could be shared for hardware because it is difficult to share real objects. However, the recent development of video platforms, such as YouTube, has created an environment that allows the expertise and development methods of hardware technologies to be easily shared. Hence, it is now possible to share technologies actively using open-source in various fields, including robotics, machinery, electronics, and autonomous driving [4], [7]–[9].

Open-source motor controller technology has recently been developed rapidly, and its performance and stability are now comparable to those of commercial motor controllers.

The associate editor coordinating the review of this manuscript and approving it for publication was Aysegül Ucar¹.

Motor controller plays a very important role in the development of motor-powered robots, and the type of upper controller can be determined by the type of motor controller used. This also affects the cost and performance of the entire system. Representative open-source motor controllers include Vedder electronic speed controller (VESC) [10], [11] and ODrive [12], [13].

This study focused on the open-source VESC motor controller under GPL V3 License [14]. Introduced by Benjamin Vedder, VESC has been developed continuously since its first release in 2014 and is estimated to have thousands of users worldwide as of October 2019. Its latest hardware version is 6.4 and the firmware version is 3.61. VESC provides a convenient software environment through a graphical user interface (GUI) tool that supports not only Linux, OSX, and Windows OS, but also applications for iOS and Android OS. Originally developed for Vedder's electric skateboard do-it-yourself hobby, VESC is being widely used as a motor controller for handmade electric skateboards. Hence, its design is ideal for high-speed, high-torque, and high-output motor control. It provides current control, duty control, speed control, position control, and brake control modes and can control DC and brushless DC (BLDC) motors. For BLDC motors, VESC supports the field-oriented control (FOC) method and enables silent and smooth motor control. An example of using VESC in robotics research is the balancing electric skateboard developed by NaverLab [15].

As its characteristics are more appropriate for high-speed, high-torque control, VESC has been used for controlling wheel-type robots, and its precise position control and speed control performances have been generally evaluated to be inferior to those of ODrive, another open-source motor controller. ODrive, with the second largest number of users after VESC, was developed to produce computer numerical control (CNC) machines that require precise position control. Although both VESC and ODrive use similar hardware components, they differ in detailed tuning and software algorithms, depending on the applications. ODrive is being widely used in robotics projects that require more accurate position control performance [4], [16]–[18]. However, it is less convenient for users compared with VESC owing to the lack of a GUI tool and a small number of developers.

This study was conducted to enable more accurate position, speed, and current control when VESC is used in robotics projects. Myongji-Vedder electric speed controller (MJ-VESC) was developed by upgrading VESC software and hardware to enable more accurate current and position control. In addition, we constructed a general-purpose motor control framework capable of controlling robot systems in real time by organically connecting MJ-VESC with upper controllers, such as PC, and utilizing a robot operating system (ROS). This framework was applied to a planetary geared actuator fabricated using a 3D printer to evaluate experimentally its position and speed control performances as well as the actuator efficiency and repetition accuracy.

II. VESC

A. INTRODUCTION TO VESC

VESC is a representative open-source motor controller and its characteristics are summarized and compared with those of ODrive in Table 1. The VESC open-source motor controller was fabricated using the PCB hardware [19] and the open-source PCB artwork software Kicad, and its GUI tool [20] was created using Qt. As all the required technologies are open-source, the user can access, use, and recommend corrections or modifications through GitHub [21]. The firmware [22], also available as open-source, can be modified as desired according to the taste and purpose of the user. However, the modified code must be opened under the GPL V3 license.

TABLE 1. Specification comparison of VESC AND ODrive.

	VESC	ODrive
Model	VESC6	ODrive V3.6
Motor	BLDC, DC, GPD	BLDC
Control	BLDC, DC, FOC	FOC
Channel	1 Channel	2 Channel
Voltage	11.1 - 60 V	24 V or 48 V
Current	80 A Cont., 120 A Burst	75 A Cont., 100 A Burst
Command Mode	Position, Speed, Duty, Current, Current Brake	Goto, Position, Velocity, Torque
GUI Tool	VESC-Tool	None
Communication	USB, UART, PPM, CAN, UAVCAN	USB, UART, PPM, PWM, CAN, Step/direction
Gate Driver	DRV8301	DRV8301
MOSFET	DirectFET AUIRF7749L2TR	NTMFS4935NT1G
RTOS	ChibiOS 3.0.2	FreeRTOS
Processor	STM32F405RGT	STM32F405RGT
Current Sensor	3 Phase Shunts	2 Phase Shunts
Position Sensor	ABI, HALL, AS5047, BiSS, Resolver	AB, ABI, AS5047
PCB Design Tool	Kicad (Open-source)	Altium Designer (Commercial)
License	GPL V3	MIT

The VESC-Tool is a GUI tool that is convenient and powerful for using VESC. The real-time data view and general description is shown in Fig. 1(a). The connection between VESC-Tool and VESC is accessible in a variety of ways, including USB, Bluetooth, TCP/IP, etc. In VESC, the FOC control is recommended for BLDC motors and has an integrated function that automatically finds motor settings. The VESC-Tool can be used to find motor settings and to perform basic tests of current and duty control through the keyboard. The motor settings can be saved in XML file format and saved files can also be imported and downloaded using VESC-Tool. The motor current, speed, position, and temperature can be checked by real-time graph. The firmware updates periodically and can be upgraded using VESC-Tool. In the sampled data view in Fig. 1(b), the data within a specific time period can be graphically represented. The sampled data view provides a more detailed view of phase current and back EMF (Electro Motive Force). The VESC-Tool is available as an open-source through Github [23], allowing users to modify and upgrade their functions to suit their own convenience under the GPL V3 license.



FIGURE 1. VESC-Tool GUI: (a) Real-time data view and general description of VESC-Tool and (b) Sampled data view.

B. ORIGINAL VESC HARDWARE STRUCTURE

Fig. 2 shows the PCB structure of the original VESC. The structure of the VESC PCB can be broadly divided into six parts: MOSFET, gate driver, microprocessor, shunt resistor and current sensing circuit, CAN transceiver, and high-capacity capacitor. To withstand high current, it is important that the copper pour area of the MOSFET be connected over the widest area possible. Caution must be exercised in handling the signal lines and GND area to obtain the desired performance without noise. CAN communication is mainly used for multiaxis control. In this case, TJA1051TK3 from NXP Semiconductors was used as a CAN transceiver chip. Owing to the high current specifications, CAN transceiver chips in the past VESC hardware versions frequently failed. However, the latest version, which uses TJA1051TK3, has significantly improved in this aspect and can be used in a more stable manner.

C. VESC THIRD-PARTY HARDWARE

All the PCB drawings of VESC are open-source, allowing users to redesign PCB through modification and supplementation according to their requirements. In addition to the original VESC, there are many third-party VESCs developed through redesign. Table 2 lists the officially released third-party products.

In this study, we developed an upgraded PCB named “MJ-VESC,” which means Myongji-Vedder Electronic

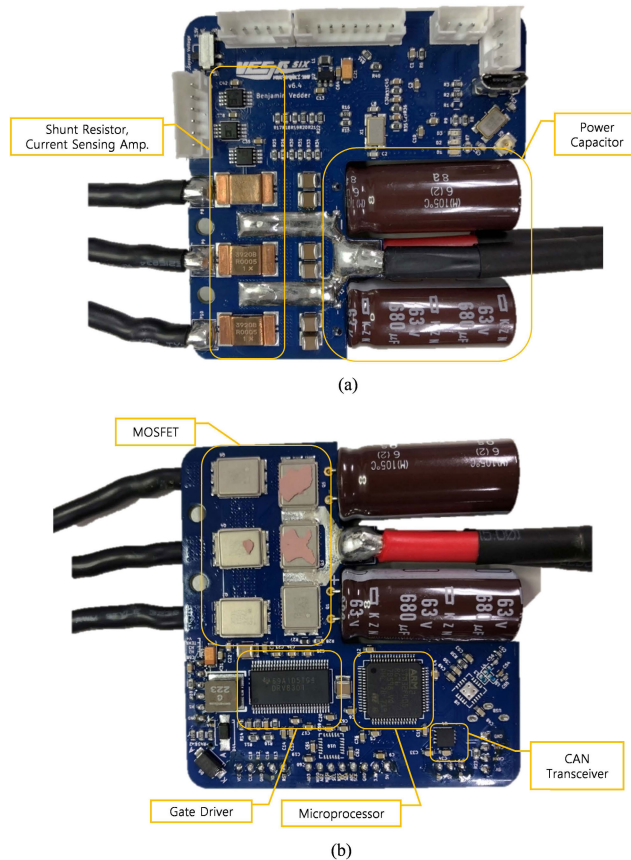


FIGURE 2. Original VESC PCB Hardware: (a) VESC PCB top side and (b) VESC PCB bottom side.

TABLE 2. VESC third-party hardware.

Model Name	Manufacturer
F5ESC 6.6	Flipsky
F5ESC 4.12	Flipsky
VESC standalone v0.1	Palta Tech
Axiom	powerdesigns
MTSVESC6.0	MAYTECH
VESC MINI 4	Anton Chromjak
BESC-G2	galpavlin

Speed Controller. The results of the study conducted using this board are detailed in the following section.

III. MJ-VESC

A. INTRODUCTION TO MJ-VESC

Fig. 3 shows the PCB structure of MJ-VESC. The MJ-VESC header pins are arranged to act as Arduino Shield. It is effective for multiaxis motor control because multiple MJ-VESCs can be interconnected in a stacked structure. Power pins and CAN communication pins, SPI communication pins are connected via board stacking, and multiaxis control can be performed by sending commands to only one master motor controller through USB. The connection can be established using Arduino Mega and Due, and commands can be sent at high speed through SPI communication. Hence, MJ-VESC is also effective for performing robot projects using Arduino.

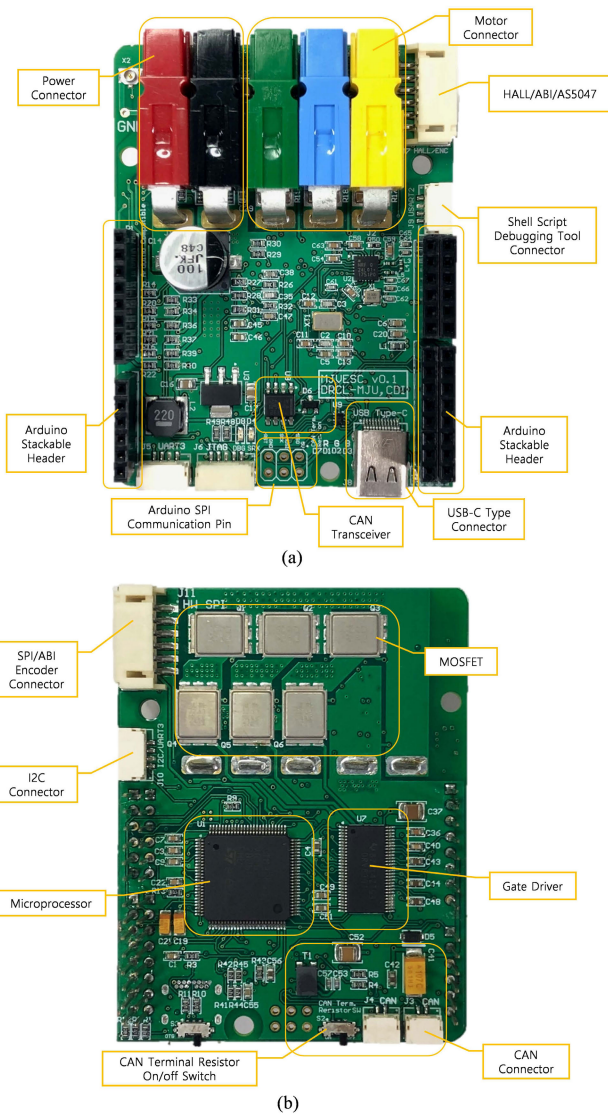


FIGURE 3. PCB hardware of MJ-VESC: (a) MJ-VESC PCB top side and (b) MJ-VESC PCB bottom side.

Fig. 4 shows the circuit diagram for CAN communication. MJ-VESC provides higher CAN communication stability than the original VESC. For CAN communication, 120 Ω terminating resistors are generally connected at both ends of the device. However, according to the SLOA101B Application Report from Texas Instrument (TI) [24], using the 60 Ω split resistor connection method can achieve more stable terminating resistance effect. Moreover, the CAN transceiver chip was protected against electrical shock using a TVS diode and a common mode choke filter. The CAN transceiver chip uses TI’s TCAN332GDR to support a maximum speed of 5 Mbps. The USB connector is generally connected to a PC to transfer motor settings or commands. While the original VESC uses the micro USB type, for MJ-VESC, we used the USB-C type connector to generate higher mating force and enable more stable use. Regarding the shunt resistor for

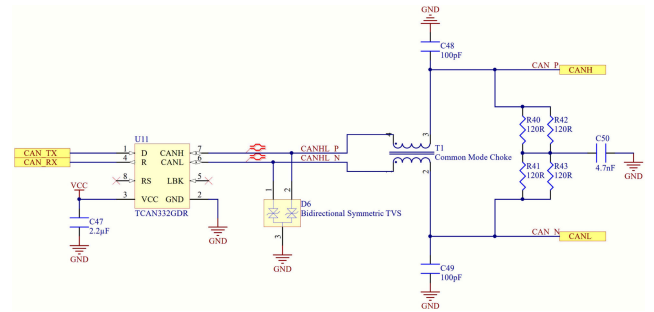


FIGURE 4. Circuit diagram of CAN communication.

current control, while the existing VESC uses 0.5 mΩ with a current measurement range of ±150 A, for MJ-VESC, we changed this to 2 mΩ to achieve a finer current control at low power than at high currents. Consequently, the current measurement noise decreases at low currents under 0.5 A, which allows finer current control. Fig. 5 compares the phase current between VESC and MJ-VESC. In both cases, changes in phase current when the motor began to rotate at standstill are plotted on a graph. We observed that the current was accurately measured with low noise when MJ-VESC was used at the standstill state of < 0.05 s.

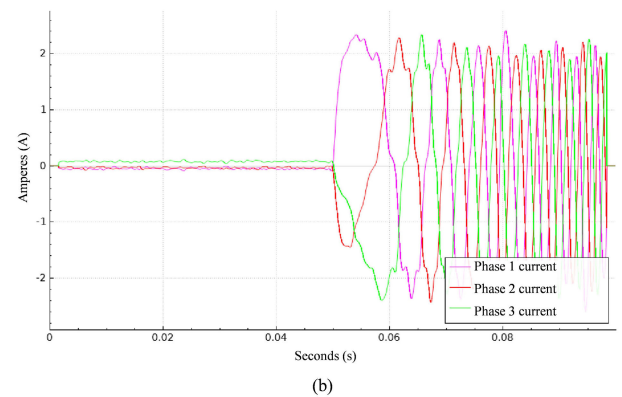
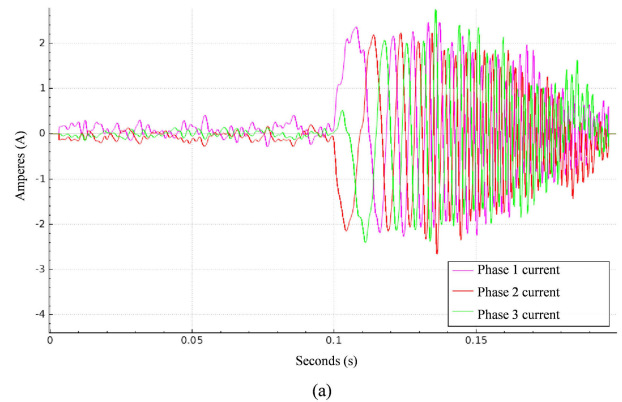


FIGURE 5. Phase current comparison of VESC and MJ-VESC: (a) Phase current of VESC and (b) Phase current of MJ-VESC.

In VESC, as the debugging environment through terminal communication is integrated in the GUI tool, the existing

debugging function cannot be used simultaneously while commands are given from the upper controller through USB communication. For MJ-VESC, an additional serial communication channel is used to enable debugging through terminal communication simultaneously while commands are given from the upper controller. The debugging environment can be used through shell-script-type commands, which enables real-time debugging, such as outputting the value of a desired variable even when the motor controller is controlled through USB or the existing UART communication. Fig. 6 shows the usage example of shell-script-type debugging tool.

```

Start VESCUino Shell Thread
VESCUino SPI RX Thread Start
SET_DPS Thread Activated
ChibiOS/RT Shell
ch>
ch> help
Commands: help exit info systime mem threads rb gpn ind pif lcd dev sdp sdt sct cdp
mpu_use mpu_param mpu_comm mpu_setup mpu_dof mpu_aub mpu_aub_use mdp sda slg smg nrf
spi sdt
spi dt: 103.809 msec, process error cnt:0
ch> info
Kernel: 3.0.1
Compiler: GCC 7.3.1 20180622 (release) [ARM/embedded-7-branch revision 261907]
Architecture: ARMv7-ME
Core Variant: Cortex-M4F
Port Info: Advanced kernel mode
Platform: STM32F407 High Performance with DSP and FPU
Board: STMicroelectronics STM32F4-Discovery
Build time: Jul 25 2019 - 18:13:08
ch> |
    
```

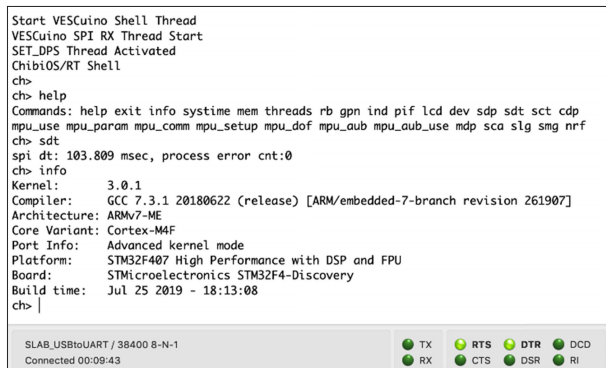


FIGURE 6. Usage example of shell-script-type debugging tool.

MJ-VESC provides additional functions while maintaining the functions of the original VESC. Table 3 summarizes the functions of the existing VESC and the added functions of MJ-VESC. Because the firmware of MJ-VESC is basically compatible with the hardware of VESC, the firmware of MJ-VESC can be uploaded and used to original VESC as well as MJ-VESC. In particular, DPS control mode and Goto control mode were developed for MJ-VESC, but were also available in original VESC, so the performance of two motor controllers was compared through experiments.

TABLE 3. Additional functions of MJ-VESC.

	Original functions of VESC firmware	Additionally Developed functions of MJ-VESC
Control Mode	Position, Speed, Current	DPS, Goto
Position Sensor	HALL, ABI, AS5047, BiSS, Resolver	HALL&ABI Hybrid
Communication	UART, CAN	SPI (1 kHz control loop up to 6 Ch)
Main Controller	Stand-alone, Arduino, PC (1 Ch)	Arduino Stackable, PC (Multi-Ch)
Debugging	GUI Tool	Shell script type

B. MJ-VESC MOTOR CONTROL FRAMEWORK

1) COMMUNICATION STRUCTURE OF MJ-VESC

The VESC motor controller is a single-channel type, i.e. one controller can control one motor. For multiaxis control, we use CAN communication and connect multiple VESCs using the daisy chain method. Connecting a Bluetooth dongle for UART communication allows checking the settings and

status of the motor controller through a smartphone application. The USB is mainly used in PCs for the connection of VESC-Tool capable of setting the motor environment and conducting a drive test, or it can be used for motor control using ROS in upper controllers, such as Linux-based PCs or single-board computers. Here, we use an ROS package called VESC-Driver, which was developed by the MIT-racecar team of MIT and can be used freely as open-source [25]. Fig. 7(a) shows an example of using the VESC motor controller.

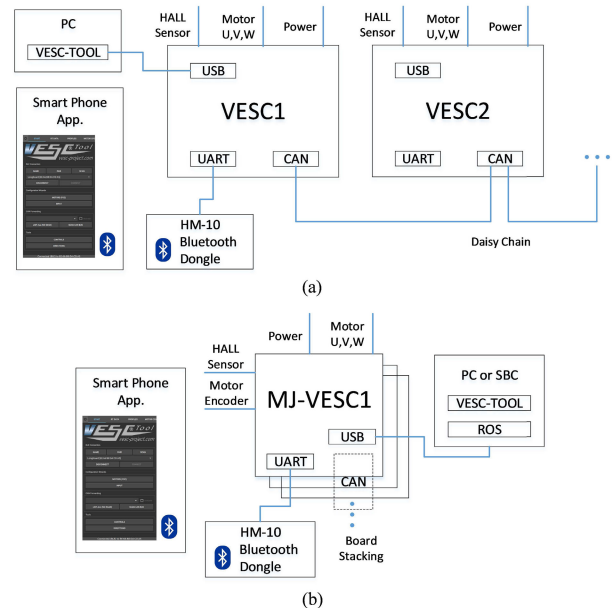


FIGURE 7. Multichannel motor control diagram: (a) VESC and (b) MJ-VESC.

MJ-VESC was designed to be more suitable for multiaxis control than the original VESC. The problem that every wire had to be connected for CAN communication was solved by adopting the board stacking method, by which the CAN communication lines are connected simultaneously with assembly. Regarding the connection of terminating resistors for CAN communication, the on/off switch was placed so that the terminating resistors could be selectively connected only to the boards at both ends. Moreover, the existing VESC-Driver package for ROS supports only the control of a single channel, but we modified it to develop a VESC-Driver package for multiaxis control with the additional CAN communication function. Fig. 7(b) shows an example of using the MJ-VESC motor controller.

2) MOTOR COMMUTATION STRUCTURE OF MJ-VESC

MJ-VESC allows the simultaneous connection and use of a hall sensor and an encoder, through which it supports hall/encoder hybrid control. This is performed to overcome the limitation that motor control cannot be performed effectively when the index phase is not found during the use of the optical encoder after the initial power connection in the FOC control of the BLDC motor. ODrive and VESC both have this limitation. To solve this problem, the ODrive user can selectively set the operation to find the index phase

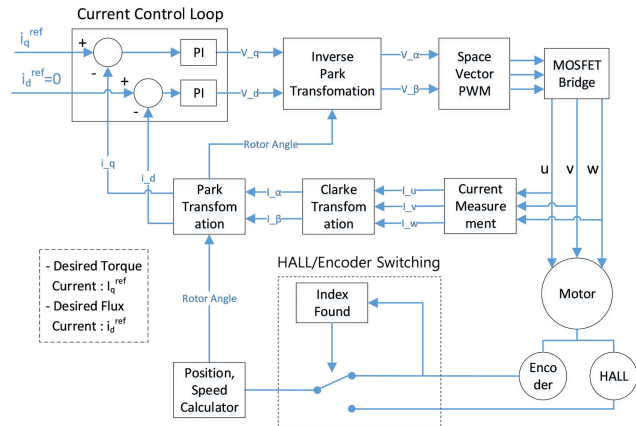


FIGURE 8. FOC current control diagram of MJ-VESC.

after the power is applied. However, this operation may be inappropriate because it can generate unwanted motion when the motor is installed and used in a robot. The BLDC motor control using a hall sensor has the advantage of performing motor control immediately with the supply of power, but it has difficulty performing accurate position control owing to the low position resolution of the hall sensor. To address this problem, MJ-VESC uses the hall sensor for control until the index phase of the optical encoder is found after the connection of power, and a seamless change is made to the encoder control mode of high resolution once the index phase is found. Fig. 8 shows the FOC current control diagram of MJ-VESC. The FOC current controller is identical to the standard FOC current control method [26]–[28]. We have added a switch between the hall sensor and the encoder depending on the detection of the index phase of the encoder. For the FOC current controller, if the value of the quadrature current, i_q^{ref} , is applied as reference, the current is subjected to feedback control by the PI controller of the current controller. The current control loop of MJ-VESC operates at 20 kHz. The PI controller has the default values for gains, i.e., a P gain of 0.03 and an I gain of 50.0.

Fig. 9 compares the initial motor rotation state between the motor operation (commutation) that uses only the Hall sensor and the Hall & Encoder hybrid commutation method. When rotation was started by the current command at standstill, we observed that the phase current was noisy when only the Hall sensor with low position resolution was used as shown in Fig. 9(a). In contrast, in the case of Fig. 9(b), after the encoder index phase was detected at 0.075 s, the mode was switched to commutation using the encoder, resulting in a very clean and stable phase current. Based on this, the initial startup performance was improved using the Hall sensor when the optical encoder was used in FOC control. We also allowed the immediate use of the encoder to achieve accurate position control.

3) MJ-VESC SPEED CONTROLLER (DPS CONTROL)

For MJ-VESC, we additionally developed the degree per second (DPS) control mode for accurate speed control based on

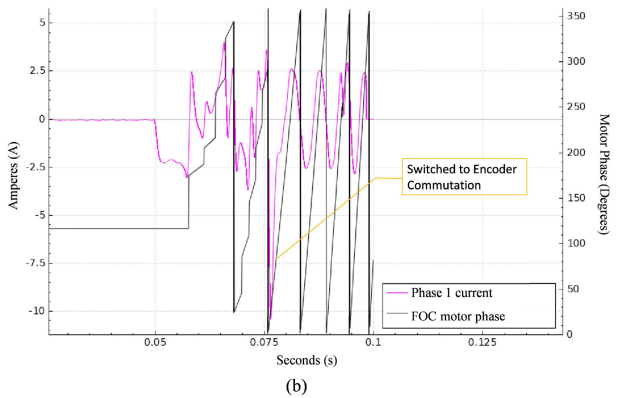
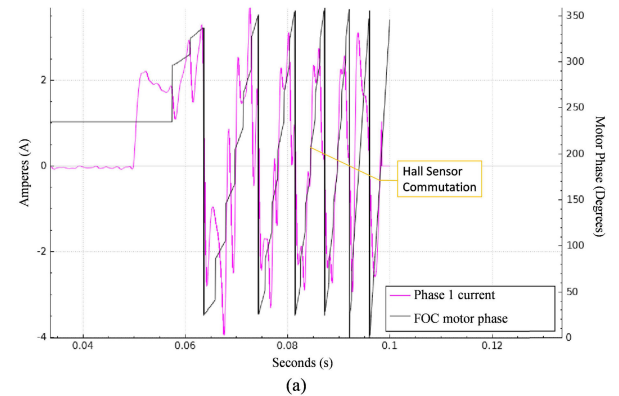


FIGURE 9. Phase current comparison of VESC and MJ-VESC: (a) Hall sensor commutation only and (b) Hall & Encoder hybrid commutation.

position control. The original VESC firmware has a position control mode. However, there is no precise speed controller. The DPS control is designed to enable accurate speed control using existing position control mode of VESC. When a speed command is sent in the DPS unit, the speed profile is tracked in the position control mode according to the speed. Fig. 10 illustrates the structure of DPS control.

In the DPS control mode, the target speed (DPS) and the maximum acceleration (A_{max}) can be received as inputs. Based on them, a speed profile of trapezoidal trajectory is created, and the rotor position for tracking the profile is determined and transferred to the position controller of MJ-VESC. At this point, proportional-integral-derivative (PID) control gains for position control can be tuned according to the motor and system environment using VESC-Tool. The default values of 0.02, 0.01, and 0.00005 were set for K_p , K_i , and K_d , respectively.

Fig. 11 shows the detailed operating principle of the profile generator that determines the position profile of the motor after receiving a speed input. The target DPS, v_{ref} , is received as an input and the acceleration value for tracking it, da , is determined by receiving and using the maximum acceleration (A_{max}) as a parameter. The speed increment, dv , and the position increment, ds , are calculated using the calculated acceleration value. These results are then used to calculate the position profile, s_{prof} . As the position of the

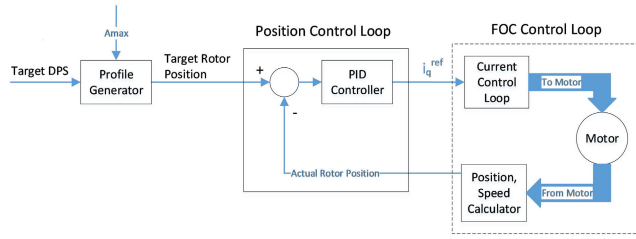


FIGURE 10. Position control based on DPS control.

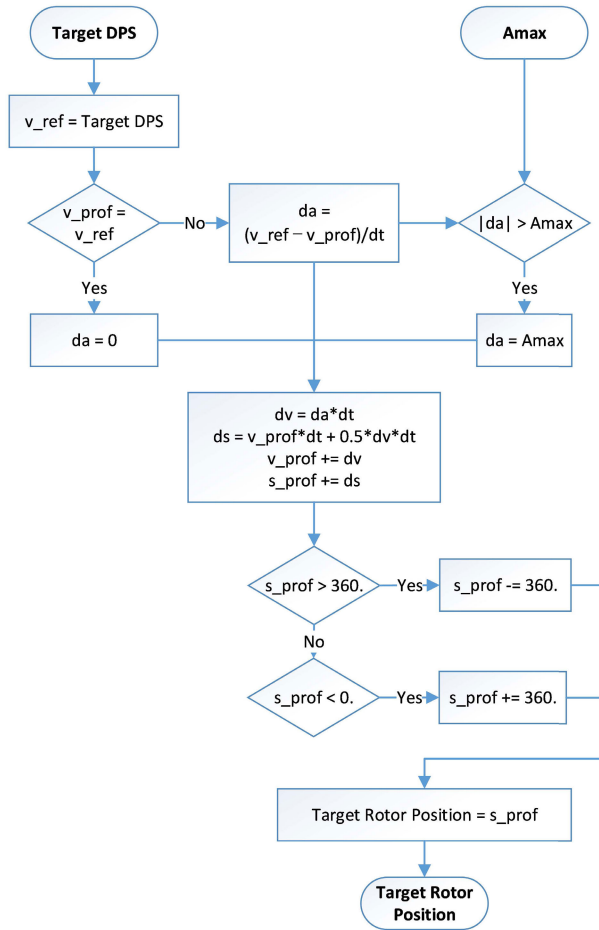


FIGURE 11. Flowchart of profile generator for DPS control.

motor ranges from 0° to 360° , s_prof is calculated accordingly. The calculated s_prof value is output as the target rotor position. The target rotor position is controlled by the PID position control loop. The control loop of the DPS controller operates at 10 kHz, and the calculated s_prof value provides a trapezoidal speed profile for tracking the target DPS value given as an integrated value from the acceleration level. This allows accurate and fast control to the desired DPS value.

4) MJ-VESC POSITION CONTROLLER (GOTO CONTROL)

In the previous section, we explained how MJ-VESC enables accurate rotation at the desired speed using DPS control. For the case in which the absolute rotation value of the motor is

given as an input, it is possible to control the absolute position of the motor if the given value is combined with DPS control. This is referred to as Goto control. Fig. 12 shows the Goto control mode.

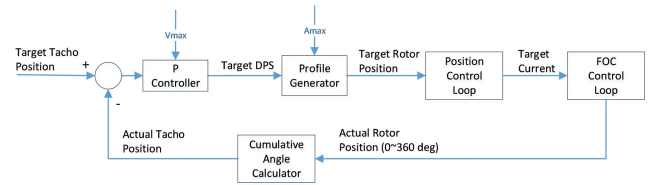


FIGURE 12. Goto controller.

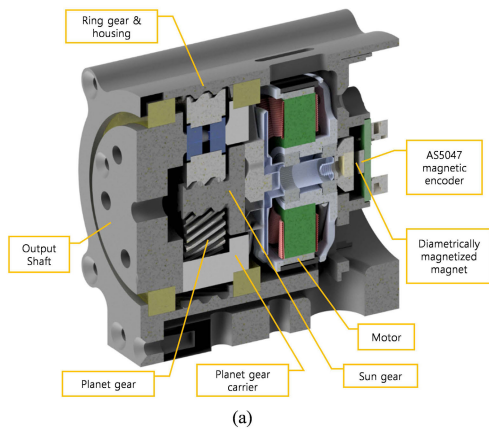
For creating the target DPS applied to the existing DPS control as an input, we added a position control using the P controller. The target DPS value is generated by multiplying the difference between the actual tacho position (the cumulative movement distance of the motor) and the target tacho position (the target movement distance) by the gain of the P controller. The speed of convergence to the desired position is determined by the gain of the P controller. In this case, the maximum speed (V_{max}) is received as a parameter and movement is controlled within the determined speed limit. The use of this controller makes it possible to move as fast as possible within the limit speed and acceleration range once the desired rotational position is specified. We set a default value of 7 for the P controller gain of the Goto controller. In the following section, we evaluate the performance of the proposed motor controller by performing experiments on the current control, DPS control, and Goto control of the actuator using the planetary geared actuator module fabricated using a 3D printer.

IV. EXPERIMENTAL VALIDATION

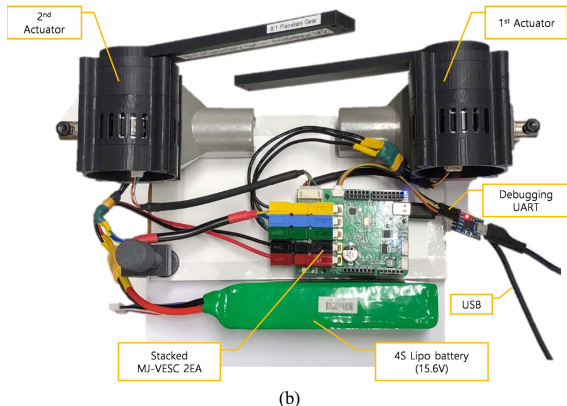
Fig. 13 shows the structure of the 3D printed planetary geared actuator and the experimental setup. The actuator had a reduction gear ratio of 8:1, and used Tarot's 4008 motor, which is mainly used in drones [29]. We used AMS's AS5047 magnetic encoder for position measurement. In the experimental setup, a 150 mm test arm was installed at the output shaft so that experiments on the rotation speed and torque could be performed. Table 4 summarizes the specifications of the actuator module. All tests were performed by switching the motor controller only under the same experimental conditions to analyze the performance differences between the original VESC and MJ-VESC. In every experiment, I used ROS as a high-level controller to control VESC and MJ-VESC and collect data.

A. CURRENT CONTROL

We compared the current control performance of original VESC and MJ-VESC motor controller. Fig. 14 shows the actual current of the motor, i_q , when 3 A was applied to the FOC controller as the reference value of quadrature current, i_q^{ref} . A three-phase shunt resistor is used for current measurement, it calculates the current value by amplifying the



(a)



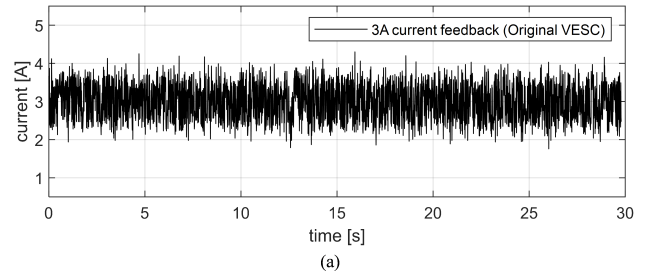
(b)

FIGURE 13. 3D Printed planetary geared actuator module: (a) Detail structure of actuator and (b) Experimental setup.

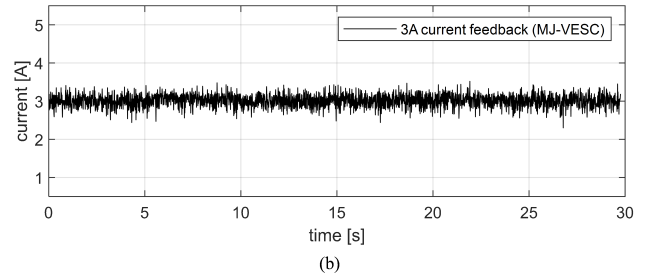
TABLE 4. 3D Printed planetary geared actuator.

Gear Ratio	8:1
Gear Module	0.65 mm
Gear Teeth	10, 30, 70 (sun, planet, ring gear)
Motor Model	Tarot 4008
Motor Speed Constant	330 kV (=rpm/volt)
Motor Torque Constant	0.029 Nm/A
Position Sensor	AS5047 (14-Bit On-axis Magnetic Encoder)
Max. Motor Speed	25000 DPS (at 15.6 V)
Max. Output Torque	9.3 Nm (at 40 A)
Cont. Output Torque	2.3 Nm (at 10 A)

very small voltage difference between the ends of the resistor. In this case, the measured current is somewhat noisy owing to the analog-digital conversion (ADC) noise, but accurate current control is possible through the digital low-pass filter and the fast current feedback of 20 kHz. The 30 second current feedback control graphs show that the result of MJ-VESC is cleaner. For a more accurate analysis, the measured current was organized in the second row of Table 5. The average current was determined by 3.005 A for VESC and 3.012 A for MJ-VESC. However, the standard deviation for MJ-VESC was about one-third of the standard deviation for VESC, confirming that the current control error for MJ-VESC was much smaller. For MJ-VESC, 2 mΩ is used for the shunt



(a)



(b)

FIGURE 14. Quadrature current measured on FOC controller under 3 A current feedback control: (a) Current graph of VESC. (b) Current graph of MJ-VESC.

resistor, whereas VESC uses 0.5 mΩ. It has been verified experimentally that the higher shunt resistance of MJ-VESC allows more accurate and fine current control in low current.

To measure the efficiency of the actuator and the motor controller, we attached a 1 kg weight to the end of the actuator as shown in Fig. 15(a), and the current to hold it in a position perpendicular to the gravity was measured. The test result is organized in the third row of Table 5. The weight was located 146 mm away from the center of rotation of the actuator. The torque generated by the weight was $1\text{ kg} \times 9.81\text{ m/s}^2 \times 0.146\text{ m} = 1.432\text{ Nm}$. In addition, the current used by the motor controller for position control was 8.359 A for VESC and 8.244 A for MJ-VESC on average. Considering the torque constant and gear ratio of the motor, the torque generated in the output shaft was $0.029\text{ Nm/A} \times 8.359\text{ A} \times 8 = 1.939\text{ Nm}$ for VESC and $0.029\text{ Nm/A} \times 8.244\text{ A} \times 8 = 1.913\text{ Nm}$ for MJ-VESC. Based on this, the torque efficiency of the motor control was calculated in Table 5. The MJ-VESC has a slightly higher torque efficiency but not much different. For the current graph as shown in Fig. 15(b) and 15(c), the current graph of MJ-VESC is less noisy. Commonly, long-term fluctuations are detected, but it can be seen that they reach a steady state, and current control performance is not necessarily degraded over time.

To test the influence of the actuator friction, the torque efficiency of the actuator was calculated by comparing the dynamic load and the current value in a situation where the weight moved at angles of $\pm 4^\circ$ in a 1 Hz sine-wave form at a position perpendicular to gravity. The Goto controller was used for this experiment. Fig. 16 shows the actual current of the motor under 1kg dynamic load and the current measurement is organized on the fourth row in Table 5. The average value of the current data measured for 15 seconds was 7.694 A

TABLE 5. Current control comparison of VESC and MJ-VESC.

Test Results	Average Current (A)		Torque Efficiency (%)		Standard Deviation Current (A)		MJ-VESC/VESC SD Ratio (%)
	VESC	MJ-VESC	VESC	MJ-VESC	VESC	MJ-VESC	
3A Current Feedback	3.005	3.012	-	-	0.482	0.158	32.812
1kg Static Load	8.359	8.244	73.854	74.881	0.416	0.288	69.307
1kg Dynamic Load	7.694	7.502	80.234	82.297	0.804	0.721	89.677

The torque efficiency was not calculated in the 3A current feedback test. The 'SD Ratio' mean the ratio of standard deviation. The SD Ratio was calculated by dividing the standard deviation of MJ-VESC by the standard deviation of VESC.

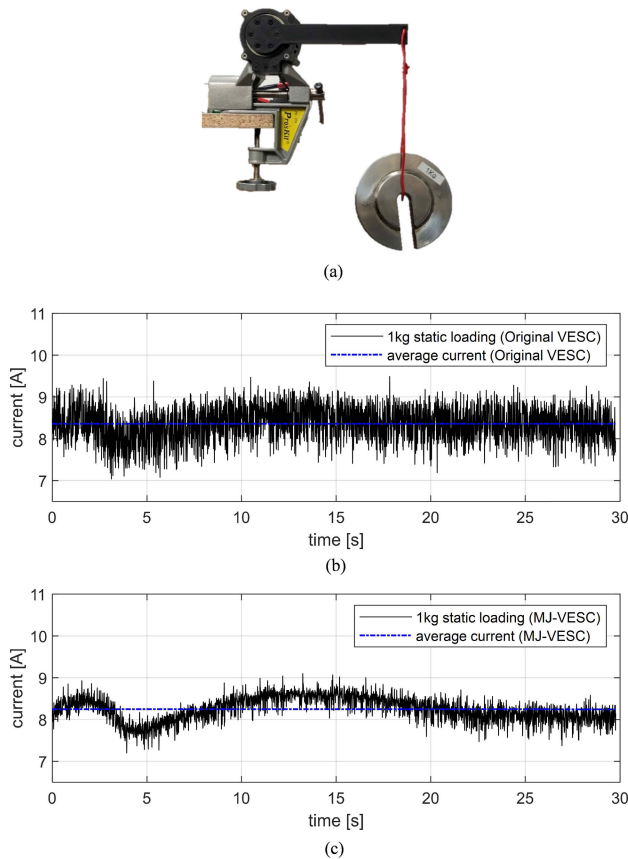


FIGURE 15. Quadrature current measured on FOC controller under 1kg static load: (a) Current graph of VESC. (b) Current graph of MJ-VESC.

for VESC and 7.502 A for MJ-VESC. Based on this, it has been confirmed in both VESC and MJ-VESC that high torque efficiency can be achieved under dynamic load conditions.

B. DPS CONTROL

The commutation of the BLDC motor with the FOC control method can be performed without difficulty at high rotation speed. However, it is challenging to achieve an accurate rotation speed at low speed [30]. Especially, when a robot arm slowly lifts a weight, the motor must be able to move with a high torque even at low speed. To test this, we conducted an experiment on rotation at an extremely low speed using the DPS controller. The DPS value of the motor was increased from 0 to 25 DPS at 5 DPS intervals, and the target DPS and the actual DPS of the motor are compared in Fig. 17. The DPS

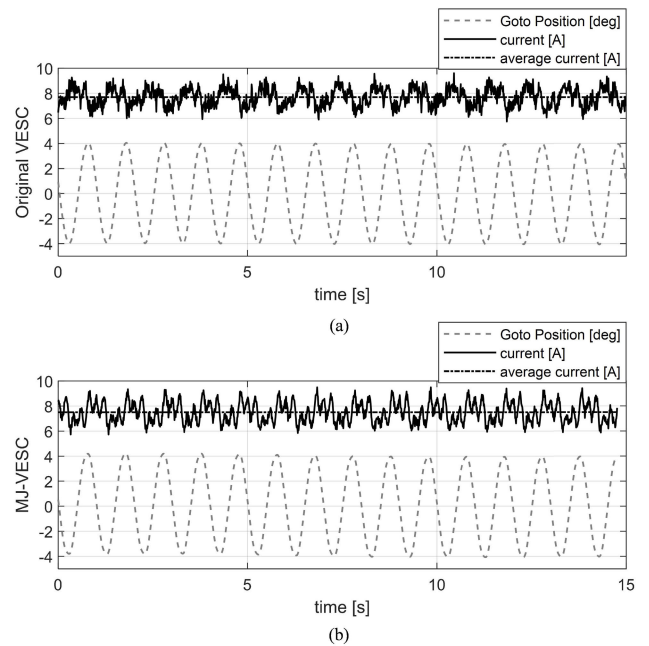


FIGURE 16. Current control experiment under 1 kg dynamic load: (a) Current graph of VESC (b) Current graph of MJ-VESC.

range used in the experiment is from 0 to 30 DPS. The DPS value required to turn at the same speed as the second hand of the clock is 48 DPS for our actuators, i.e. the DPS value taken in this test corresponds to a speed slower than the second hand of the clock. At this low speed, the actual rotation speed of the motor did not show visible vibration or discontinuity even though there were some fluctuations in the speed graph. The experimental video can be found through a link in the APPENDIX A. The average values of the actual rotation speed by section are organized in Table 6. It can be observed that the average speed accurately tracked the target speed in both VESC and MJ-VESC. However, we can confirm that more precise control is possible in MJ-VESC by checking the SD ratio and the graph in Fig. 17.

We also performed an experiment for the rotation of motors at high speed. This test was performed using MJ-VESC only. Two actuators were placed with a difference of 90° and were rotated simultaneously in opposite directions at the same speed to verify speed errors during multiaxis control as shown in Fig. 18(a). The test arm installed on the output shaft of the actuator faces the risk of collision if a speed error occurs. High-speed DPS speed control was performed while the

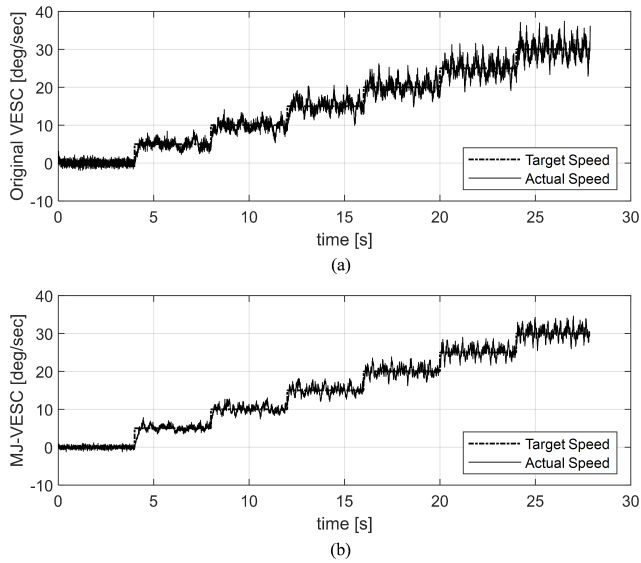


FIGURE 17. Low-speed DPS control: (a) Original VESC and (b) MJ-VESC.

TABLE 6. Low-speed dps control comparison of VESC AND MJ-VESC.

Target Speed (deg/sec)	Average Speed (deg/sec)		Standard Deviation Speed (deg/sec)		SD Ratio (%)
	VESC	MJ-VESC	VESC	MJ-VESC	
0	0.012	-0.002	0.672	0.349	51.91
5	4.846	5.010	1.183	1.010	85.34
10	9.733	10.081	1.359	1.044	76.87
15	15.116	15.086	1.732	1.190	68.71
20	20.085	20.107	1.984	1.306	65.84
25	24.912	25.212	2.182	1.408	64.50

'SD Ratio' mean the ratio of standard deviation. SD Ratio was calculated by dividing the standard deviation of VESC by the standard deviation of MJ-VESC.

speed was increased by 5,000 DPS, and the results are shown in Fig. 18(c). It was confirmed that the actual speed accurately tracked the target speed while the speed was increased from 0 to 25,000 DPS. Moreover, in the event of a cumulative speed error, the test arms will inevitably collide with each other. However, they did not collide and rotated while maintaining a difference of exactly 90°. Fig. 18(b) shows the image captured using a high-speed camera when the motor rotated at 25,000 DPS. Consequently, it was confirmed that accurate speed control was performed at both high and low speeds and the multiaxis using DPS control.

Frequency response experiments were conducted to compare the performance of DPS controllers from low to high frequencies. Fig. 19 shows a comparison of frequency response between VESC and MJ-VESC. The experiment was conducted from 0 Hz to 15 Hz and 1000 DPS amplitude sine-wave speed input was applied to the DPS controller. The experimental results showed the results of the frequency response experiment in a typical 1st-order system. The continuous-time identified transfer functions are summarized in Table 7. Although there is no significant difference between the transfer functions, the time delay of

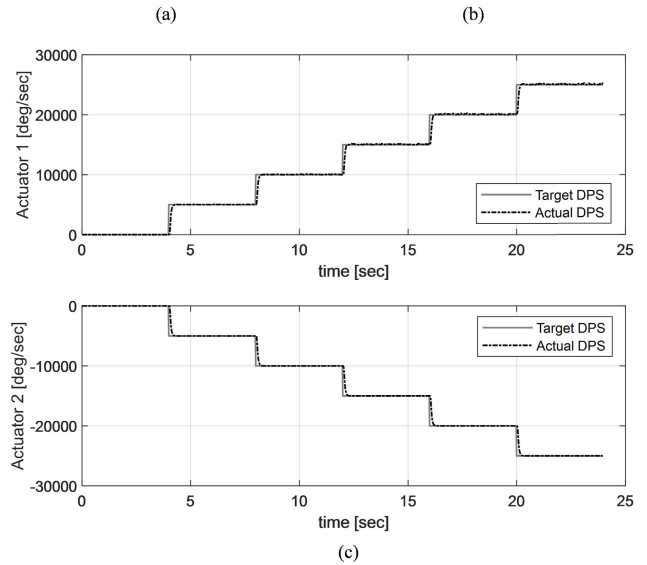
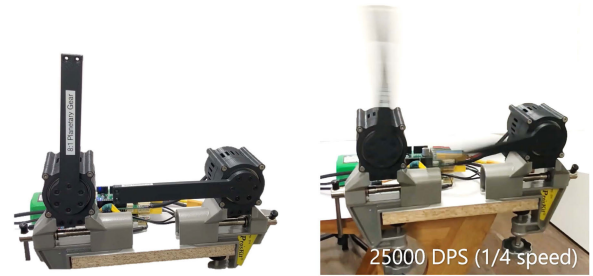


FIGURE 18. High-speed DPS control: (a) Experimental setup (b) 25000 DPS captured image and (c) Target DPS with actual DPS of Actuator 1 and Actuator 2.

TABLE 7. Transfer functions of DPS control.

Test Results	Transfer function	Accuracy of estimation
VESC	$\exp(-0.01s) \frac{22.95}{s+25.41}$	87.54 %
MJ-VESC	$\exp(-0.012s) \frac{23.34}{s+24.05}$	89.19 %

VESC is 0.01 second, whereas the time delay of MJ-VESC is 0.012 seconds. It can be interpreted that the shorter time delay of VESC is due to a smaller shunt resistance than MJ-VESC. For noise in the input output graph, VESC is measured smaller in areas above 8 Hz, which can give better control over the high frequency range if the shunt resistance is small. This showed that increasing the shunt resistance allowed precise control in low frequency, but that control performance in high frequency areas could be somewhat degraded.

C. GOTO CONTROL

The accuracy and speed of reaching the target position were tested using the Goto controller. An experiment in which about 0.7 mm was pressed using the dial test indicator and then the position of the test arm was moved by 180° in the opposite direction and quickly returned to the original position was repeated 12 times. The dial test indicator is located

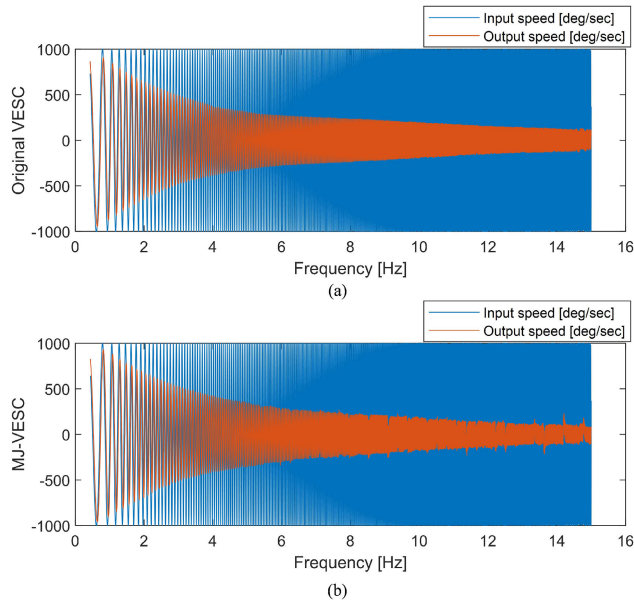


FIGURE 19. Frequency response experiment: (a) Input-Output graph of VESC. (b) Input-Output graph of MJ-VESC.

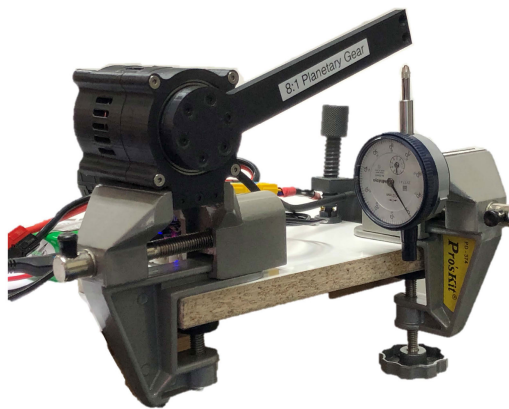


FIGURE 20. Goto control experiment setup.

135 mm from the center of the actuator. The acceleration and speed limits were set to $100,000 \text{ deg/s}^2$ and $25,000 \text{ deg/s}$, respectively. Fig. 20 shows the experimental setup.

Table 8 shows the values recorded by the dial test indicator when the test arm was returned to the set position. After the 12 trials of the experiment, the average value was 0.749 mm for VESC and 0.718 mm for MJ-VESC. The standard deviation was $3.6 \mu\text{m}$ for VESC and $1.6 \mu\text{m}$ for MJ-VESC, indicating very accurate repeatability. This standard deviation can be converted to an angle value considering the dial test indicator position, 135 mm. The converted standard deviation angle is 5.430 arcsec for VESC and 2.394 arcsec for MJ-VESC, which was an excellent result considering that the actuator was fabricated using a 3D printer. Considering that the position repeatability of a commercially available harmonic drive is approximately 10 arcsec, the 3D-printed actuator exhibited a level comparable to that of a commercial product in terms of

TABLE 8. Dial test indicator measurements.

Trials	VESC	MJ-VESC
1	0.744 mm	0.715 mm
2	0.742 mm	0.717 mm
3	0.747 mm	0.720 mm
4	0.745 mm	0.716 mm
5	0.750 mm	0.717 mm
6	0.749 mm	0.720 mm
7	0.753 mm	0.716 mm
8	0.752 mm	0.718 mm
9	0.751 mm	0.717 mm
10	0.751 mm	0.719 mm
11	0.747 mm	0.718 mm
12	0.752 mm	0.717 mm
Average	0.749 mm	0.718 mm
Standard deviation	$3.6 \mu\text{m}$	$1.6 \mu\text{m}$
Standard deviation (Angle)	5.430 arcsec	2.394 arcsec

backlash [31]. Also, the results of MJ-VESC were found to have more than twice the repeatability of using VESC. This confirmed that MJ-VESC showed higher positional accuracy performance than original VESC.

V. CONCLUSION

MJ-VESC, an open-source motor controller, was introduced and its performance was tested in this study. An attempt was made to extend the applications of original VESC, which was developed as an open-source motor controller for electric skateboards, to the robotics field by modifying and supplementing its drawbacks. In view of this, the hardware and software were upgraded for more accurate current, speed, and position control performances. The experiment compared the performance of original VESC and MJ-VESC in terms of current, speed and position, and demonstrated that the performance of the MJ-VESC was superior. As a result, an open-source motor controller with high-accuracy speed and position control performances was developed using DPS control and Goto control. Moreover, the VESC-ROS driver was modified and supplemented for controlling multiaxis motors using ROS, and a framework applicable to general-purpose robot systems was constructed. A planetary geared actuator module was developed using a 3D printer, and it was verified through experiments that two actuators could be accurately controlled with high speed. The experiment results showed that high torque efficiency and repeatability could be achieved even though the actuator module was fabricated at low cost using a 3D printer. The hardware and software source codes of MJ-VESC can be accessed on the Internet [29], [32], [33] and protected by the GPL V3 license. In the future, we will conduct follow-up studies on cooperative robots using the proposed motor controller and actuator module to develop a 6-degrees-of-freedom robot arm with the features of low cost, ease of use, and high performance.

APPENDIX A EXPERIMENTAL VIDEO

- Low-Speed DPS control test video available: <https://youtu.be/Gu5IkoJ7c6M>
- High-Speed DPS control test video available:

<https://youtu.be/Z8wtRP5yNeM>

- *Frequency response test video available:*
<https://youtu.be/cgKMLYrX4I4>
- *Goto control test video available:*
https://youtu.be/6U3oi_pJ15s

REFERENCES

- [1] M. Quigley, "ROS: An open-source robot operating system," in *ICRA Workshop Open Source Softw.*, 2009, vol. 3, no. 3.2, p. 5.
- [2] S. Kohlbrecher, J. Meyer, T. Graber, K. Petersen, U. Klingauf, and O. V. Stryk, "Hector open source modules for autonomous mapping and navigation with rescue robots," in *Robot Soccer World Cup* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2013, pp. 624–631.
- [3] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.
- [4] N. Kau, A. Schultz, N. Ferrante, and P. Slade, "Stanford doggo: An open-source, quasi-direct-drive quadruped," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, Montreal, QC, Canada, May 2019, pp. 6309–6315.
- [5] G. Hertel, S. Niedner, and S. Herrmann, "Motivation of software developers in Open Source projects: An Internet-based survey of contributors to the Linux kernel," *Res. Policy*, vol. 32, no. 7, pp. 1159–1177, Jul. 2003.
- [6] StatCounter GlobalStats. (2019), *Mobile Operating System Market Share Europa*, Accessed: Oct. 17, 2019. [Online]. Available: <https://gs.statcounter.com/os-market-share/mobile/worldwide>
- [7] L. Bulwahn, T. Ochs, and D. Wagner *Research on an Open-Source Software Platform for Autonomous Driving Systems*. Munich, Germany: BMW Car IT GmbH, 2013.
- [8] H. Darweesh, E. Takeuchi, K. Takeda, Y. Ninomiya, A. Sujiwo, L. Y. Morales, N. Akai, T. Tomizawa, and S. Kato, "Open source integrated planner for autonomous navigation in highly dynamic environments," *J. Robot. Mechatron.*, vol. 29, no. 4, pp. 668–684, Aug. 2017.
- [9] C. Moulin-Frier, P. Rouanet, and P.-Y. Oudeyer, "Explauto: An open-source Python library to study autonomous exploration in developmental robotics," in *Proc. 4th Int. Conf. Develop. Learn. Epigenetic Robot.*, Genoa, Italy, Oct. 2014, pp. 171–172.
- [10] B. Vedder (2015). *VESC—Open Source ESC*. Accessed: Nov. 27, 2019. [Online]. Available: <http://vedder.se/2015/01/vesc-open-source-esc/>
- [11] Trampa. (2018). *VESC 6 Plus VESC 75/300*. Accessed: Sep. 18, 2019. [Online]. Available: <https://www.trampaboards.com/vesc-6-in-cnc-t6-silicone-sealed-aluminium-box-with-genuine-xt90-connectors-vedder-electronic-speed-controller-trampa-special-copy-p-27456.html>
- [12] O. Weigl. (2017). *ODrive*. Accessed: Nov. 10, 2019. [Online]. Available: <https://odriverobotics.com>
- [13] O. Weigl. (2017). *ODrive*|*Hackaday.io*. Accessed: Nov. 10, 2019. [Online]. Available: <https://hackaday.io/project/11583-odrive-high-performance-motor-control>
- [14] B. Smith. (2018), *A Quick Guide to GPLv3, Free Software Foundation* Accessed: Sep. 18, 2019. [Online]. Available: <https://onovo.ua/wp-content/uploads/2018/07/quick-guide-gplv3.pdf>
- [15] iF World Design Guide. (2018), *Personal Last-mile Mobility/Electric skateboard*. Accessed: Oct. 11, 2019. [Online]. Available: <https://ifworlddesignguide.com/entry/237341-personal-last-mile-mobility>
- [16] J. Geating. (2019). *Prometheus Arm*|*Hackaday.io*. Accessed: Sep. 16, 2019. [Online]. Available: <https://hackaday.io/project/167296-prometheus-arm>
- [17] T. Wilkinson. (2019). *3D Printed Robot Joint with Active Compliance*|*Hackaday.io*, Accessed: Sep. 16, 2019. [Online]. Available: <https://hackaday.io/project/165653-3d-printed-robot-joint-with-active-compliance>
- [18] L. Dabin. (2019). *MACI Cobot*|*Hackaday.io*. Accessed: Sep. 16, 2019. [Online]. Available: <https://hackaday.io/project/165307-maci-cobot-6dof-5kg-850mm>
- [19] B. Vedder. (2015). *Vesc bldc Hardware*|*Github.com*. Accessed: Sep. 18, 2019. [Online]. Available: <https://github.com/vedderb/bldc-hardware>
- [20] B. Vedder. (2017). *VESC Project*. Accessed: Nov. 27, 2019. [Online]. Available: <https://vesc-project.com/>
- [21] L. Dabbish, C. Stuart, J. Tsay, and J. Herbsleb, "Social coding in GitHub: Transparency and collaboration in an open software repository," in *Proc. Conf. Comput. Supported Cooperat. Work (CSCW)*, 2012, pp. 1277–1286.
- [22] B. Vedder. (2015). *VESC BLDC Firmware*|*Github.com*. Accessed: Sep. 18, 2019. [Online]. Available: <https://github.com/vedderb/bldc>
- [23] B. Vedder. (2015). *Vesc Tool*|*Github.com*. Accessed: Sep. 18, 2019. [Online]. Available: https://github.com/vedderb/vesc_tool
- [24] S. Corrigan (2016), *Introduction to Controller area Network (CAN), Application Rep. SLOA101B*. Accessed: Sep. 16, 2019. [Online]. Available: <http://www.ti.com/lit/an/sloa101b/sloa101b.pdf>
- [25] Mit-Racecar. (2015). *Vesc Driver*|*Github.com*. Accessed: Sep. 19, 2019. [Online]. Available: https://github.com/mit-racecar/vesc/tree/master/vesc_driver
- [26] Texas Instruments Europe. (1998). *Field Orientated Control 3-Phase AC-Motors, Literature Number BPRA073*. Accessed: Sep. 16, 2019. [Online]. Available: <http://www.ti.com/lit/an/bpra073/bpra073.pdf>
- [27] H. Kubota and K. Matsuse, "Speed sensorless field-oriented control of induction motor with rotor resistance adaptation," *IEEE Trans. Ind. Appl.*, vol. 30, no. 5, pp. 1219–1224, 1994.
- [28] R. Gabriel, W. Leonhard, and C. J. Nordby, "Field-Oriented Control of a Standard AC Motor Using Microprocessors," *IEEE Trans. Ind. Appl.*, vols. IA-16, no. 2, pp. 186–192, Mar. 1980.
- [29] D. Choi. (2019). *3D Printed Planetary Gear Actuator Module*|*Hackaday.io*. Accessed: Oct. 07, 2019. [Online]. Available: <https://hackaday.io/project/167656-81-3d-printed-planetary-gear-actuator-module>
- [30] H. Madadi Kojabadi and L. Chang, "Model reference adaptive system pseudoreduced-order flux observer for very low speed and zero speed estimation in sensorless induction motor drives," in *Proc. IEEE 33rd Annu. IEEE Power Electron. Spec. Conf.*, vol. 1, Jun. 2003, pp. 301–308.
- [31] R. Slatter and R. Degen, "Miniature zero-backlash gears and actuators for precision positioning applications," in *Proc. 11th Eur. Space Mech. Tribol. Symp. (ESMATs)*, 2005, pp. 9–16.
- [32] D. Choi. (2019). *Dongilc*|*github.com*. Accessed: Oct. 07, 2019. [Online]. Available: <https://github.com/dongilc>
- [33] D. Choi. (2019). *MJ-VESC Arduino Shield Type Ver 0.1*|*Hackaday.io*. Accessed: Oct. 07, 2019. [Online]. Available: <https://hackaday.io/project/167706-mj-vesc-arduino-shield-type-ver-01>



DONGIL CHOI (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in mechanical engineering from the Korea Advanced Institute of Science and Technology (KAIST) in 2005, 2007, and 2012, respectively. He was a Postdoctoral Fellow with the Robotics Institute, Carnegie Mellon University, in 2013. He participated in the DARPA Robotics Challenge Trials. He joined the Korea Aerospace Research Institute (KARI), as a Senior Researcher, in 2014. He worked with Naver Labs as a Principal Research Engineer, in 2016. He has been a Professor with Myongji University, since 2018. His research interests include the design and control of autonomous robot, legged robot, and mobile manipulator.

• • •