Dual BLDC motor driver for arm of humanoid robot MARKO

Aleksandar Batinica, Srđan Savić, Milutin Nikolić, Mirko Raković, Branislav Borovac
Chair of Mechatronics, Robotics and Automation
University of Novi Sad, Faculty of Technical Sciences
Novi Sad, Serbia
e-mail: batinica@uns.ac.rs

***Abstract*—This paper presents the development of an open BLDC motor driver that is capable of driving two motors and interfacing to absolute and incremental encoders. The developed driver will be used for controlling the motion of arms joints of the humanoid robot MARKO. A brief description of electrical motors and their use in mobile robots is given first. After that a description about the sensors used is given, followed by the electronics of the driver shown together with explanation of the choices made. Finally the implemented regulator is described together with the control algorithm.**

Keywords—humanoid robot; BLDC motor; driver;

I. INTRODUCTION

The driver for BLDC motor is developed for controlling the arms of the humanoid robot MARKO [1]. The CAD model of the arm is shown in Fig. 1. Each arm has seven degrees of freedom (DOF), of which five are actuated by BLDC motors. The remaining two DOF are actuated by linear servomotors.

Fig. 1. CAD model of the arms of MARKO.

The need for such a driver came from the fact that most commercially available drivers are closed (they have a preprogrammed PID loop that only allows users to change the PID parameters). Another drawback of commercial drivers are high price per functionality and limited use for rare types of sensors and actuators.

The goal is to develop an open driver such that will allow implementation of different control algorithms, and not just PID control. One driver is supposed to be able to control two BLDC motors, and interface to incremental encoders with differential sinusoidal outputs, absolute encoders with a synchronous serial interface (SSI). Additional functionalities of the driver are measurement of the current of a BLDC motor and communication with other devices over a RS485 link.

II. BACKGROUND AND PREVIOUS WORK

Humanoid robots are actuated by different types of actuators. Some examples of robots powered by pneumatic and hydraulic actuators are given in [2] and [3]. Pneumatic and hydraulic actuators used in humanoid robots, and the power supply for those actuators (i.e. compressed air, and pressurized hydraulic fluid), are usually specially designed for a specific robot. Therefore, the most common actuators used in modern humanoid robots are electrical motors.

Electrical motors can be separated into two major groups depending on the type of current they are driven with: alternating current (AC) motors and direct current (DC) motors. For autonomous robots DC motors are more commonly used since battery packs supply direct current. There are three major types of DC motors commercially available: step, brushed and brushless motors.

Step motors are mostly used in open loop control which simplifies the control algorithm, but the lack of feedback is its major flaw for use in a dynamic environment. Since it can be overloaded and the control system has no feedback, there is no method for the control system to compensate for the disturbance.

Brushed DC motors are the most commonly used type of DC motors in robotics. These motors have a stator made of a permanent magnet, while the rotor is equipped with an electromagnet. Motion is made by alternating the current through the windings of the electromagnet. The process of

alternating the current through a winding is called commutation. The commutation in brushed DC motors is done mechanically via the brushes. These brushes are also the weakness of this type of motor. The brushes are mostly made from graphite material which wears down quite fast due to mechanical friction, and thus brushed DC motors need to be serviced frequently. This maintenance requires the motor to be taken apart, which lowers its lifetime. However the mechanical commutation allows very simple control of these motors, which is the reason they are so frequently used.

Brushless DC motors, as their name says, do not have brushes. Their rotor is a permanent magnet, while the stator is equipped with an electromagnet. By changing the magnetic polarity in the stator via the electromagnet the rotor follows the magnetic field rotation produced by the stator, which leads to the motion of the rotor. This type of motor most often has three phases. The current through these phases needs to be alternated which is done by a three phase MOSFET bridge. The commutation in a BLDC motor is done by turning on and off the MOSFETs in a predefined sequence. However the change in the index of the sequence is a function of the rotor position which is determined either by using Hall effect sensors embedded in the motor or by measuring the back EMF and detecting zero crossing. Also encoders can be used for this purpose but it is not common in practice.

The use of BLDC motors is increasing in different areas such as robotics, aerospace, instrumentation systems, space vehicles, electric vehicles, and industrial control applications. In such applications, conventional controllers like P, PI, and PID are being used with the BLDC servomotor drive control systems to achieve satisfactory transient and steady-state responses. Also, the development of nonconventional control algorithms for BLDC motor is progressing every day. In [4] a robust current controller of a brushless motor is introduced, upgraded with a proportional-integral velocity controller incorporating field-programmable gate array platform. Authors in [5] used the dsPIC30F4011 digital controller for control of a three phase BLDC motor, where the motor is controlled in all the four quadrants without any loss of power; i.e. the energy is conserved during the regenerative period. In [6] a fuzzy controller is designed and implemented, and its performance is compared with a PID controller to show the capability to track the error and usefulness of a fuzzy controller in control applications. Authors in [7] presented a fault-toleranttorque controller for BLDC motors that can maintain accurate torque production with minimum power dissipation.

III. ROBOT'S MOTORS AND SENSORS

The motors used for driving the joints in the arms of MARKO are BG32x20 and BG 42x15 by Dunkermotoren. Some of the important characteristics of these motors are shown in Table I. In Table I, U_N is the nominal voltage, I_N is the nominal current, I_{PMAX} is the maximum peak current, τ_N is the nominal torque and K_τ is the torque constant.

The motors are equipped with Hengstler's absolute encoder Acuro AD36. The encoder has a 25 bit resolution of which 13 bits are for single turn measurement, while the remaining 12 bits are for multi turn measurement. These 25 bits are accessed via the synchronous serial interface (SSI). 25 bits of data are additionally coded with Grey's code which needs to be decoded to its binary representation. The encoder also outputs the standard two channel incremental signals which are sinusoidal and differential. The encoders can be power by a voltage range from 10 V up to 30 V. All the input and output signals are differential so their voltage levels are not important.

TABLE I. CHARACTERISTICS OF THE MOTORS USED.

Characteristics	**Motors**	
	BG32x20	***BG42x15***
U_N [V]	24	24
Speed [rpm]	3600	3630
I_N [A]	1.13	2.24
I_{PMAX} [A]	4.5	15
τ_N [Ncm]	4.79	10.8
K_τ [Ncm/A]	4.5	5.5

As already mentioned, SSI is used to acquire data from the encoder. The encoder needs to receive a clock signal from a master, but as a differential signal. On the first rising edge the value in the data register inside the encoder is locked. On every rising edge of the clock signal the encoder shifts one bit out to the data line. The data line is also differential. After all 25 bits are shifted out the data register is unlocked again until the next readout.

IV. ELECTRONICS

The block schematic of the electronics subsystem is shown in Fig. 2. As shown in Fig. 2. the whole electronics system is separated into five blocks: power supply, microprocessor unit, communication interface, sensor interface and power electronics with current measurement.

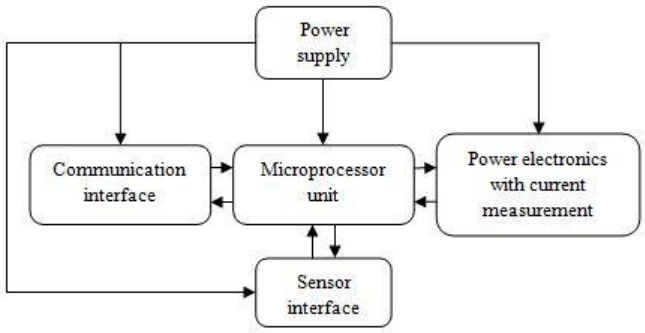

Fig. 2. Block schematic of the electronics subsystem.

A. Power supply

The power supply consists of three cascaded switching DC/DC buck converters. The input of the system is 24 V, which is used to power the absolute encoders, the motors, and the first switching converter. The output of the first converter is 15 V which is required by the gate driver to drive the power MOSFETs. The next level of conversion takes the 15 V and converts it to 5 V. The 5 V rail is used to power the Hall effect sensors in the motor and the current measurement ICs. Finally the 5 V is converted to 3.3 V which is used to power the microcontroller and the rest of the interface ICs.

B. *Microprocessor unit*

The microprocessor unit is an ARM Cortex M4 embedded in the STM32F407VG microcontroller. The STM32F407VG microcontroller has special peripherals designed to drive a BLDC motor [8]. It consists of two timers that can generate pulse width modulated (PWM) signals and its complementary value on a different output. Also the timers have a dead time insertion block embedded. There is another pair of timers that have special functionality to interface to a Hall effect sensor. Also the timers used to generate the PWM can be triggered by the timers interfacing to the Hall effect sensors which reduces the processor load significantly.

Another two timers are used to interface the incremental encoder with a square wave output. The microcontroller has three SPIs of which two are used to extract data out of the absolute encoders via the SSI. Six USARTs are embedded inside the microcontroller of which one is used for USB communication and a second one for RS-485 communication. Finally the microcontroller consists of three A/D converters with several multiplexed channels, of which two channels are used to digitalize the motor current measurement. The microprocessors can be clocked with up to 168 MHz which is important for fast execution of control algorithms.

C. *Sensor and communication interface*

The role of the sensor interface is to convert the differential signals coming from the encoder to single sided signals that the microcontroller can read and vice versa to convert the single sided signals coming from the microcontroller to differential signals that the encoder can read.

For this purpose the Texas Instruments's SN65HVD11D IC is used. This is a differential line driver/receiver. For converting the sinusoidal differential incremental encoder signals into single sided square signals a push-pull rail to rail comparator is used. By comparing the difference between sinusoidal signals a square waveform is generated. A comparator with a push-pull output was chosen to save board space and number of components. The chosen comparator is a rail to rail comparator since it is desirable for the high signal to be as close to 3.3V and for the low signal to be close to 0V. The part that was used is Microchip's MCP6562-E/SN which contains two comparators (one for each channel) in a single IC.

For communication with other devices two interfaces are available: USB and RS-485 bus. The interface itself consist of a USB to USART and RS-485 to USART converter. For that purpose MicroChip's MCP2200 and Texas Instruments's SN65HVD11D were used, respectively.

D. *Power electronics*

The purpose of power electronics module is to amplify the control signals coming out of the microcontroller so they can power a motor. As such it acts like a class D amplifier, where the motor's windings serve the purpose of a low pass filter. Controlling the high side MOSFET of the push-pull output configuration is always delicate. The junction of high side MOSFET has a variable voltage because it is connected to a motor winding, on which a voltage is induced. The value of the induced voltage is a function of the speed at which the motor is spinning.

To turn on a MOSFET, the voltage on its gate needs to be higher than the voltage on its source by at least the value of the MOSFETs threshold voltage. Since both, the junction voltage and the gate voltage are referenced to ground, it is very easy to come into a situation where the difference between the gate and junction voltage is not higher than the threshold voltage needed to turn on the MOSFET. To overcome these difficulties a monolithic gate driver with a bootstrap circuit is used. The gate driver amplifies the signal from the microcontroller to a higher level that can power on the high power MOSFET. The 3.3 V signal from the microcontroller is amplified to 15 V.

It is also worth noting that a MOSFET acts like a voltage controlled resistor. The higher the voltage on its gate is, the lower the resistance between its drain and source is, which increases the efficiency of the system. There is a limitation to this as there is a maximum voltage that a MOSFET can handle on its gate. Furthermore, the gate driver introduces a bootstrap circuit into the system. The bootstrap circuit delivers a floating supply referenced to the junction with the variable voltage. The gate driver uses this floating supply to make the signal that is aimed to control the high side MOSFET referenced to the floating supply itself. That means that the signals high value will always be higher than the source by a fixed value. The MOSFETs used are International Rectifier's IRLR8743PBF and the monolithic gate driver used is International Rectifier's IR2101S.

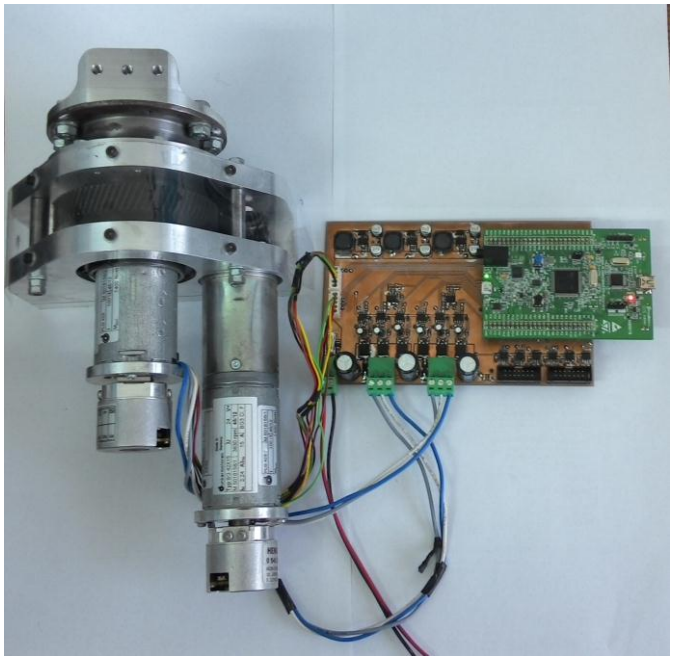

Fig. 3. The prototype board interfacing to two BLDC motors for driving MARKO's shoulder joints.

The current measurement is done by a Hall effect sensor placed in the low side of the three phase bridge. A Hall effect sensor adds very low resistance to the current path. The sensor is compact and it did not show any stability issues during testing. The sensor used is Allegro Microsystems's ACS712ELCTR-20A-T.

V. REGULATOR

At the time of writing the paper a cascaded position PI and velocity PID regulator is implemented on the driver. However as mentioned before the main reason for developing the driver is to have an open platform where different kind of regulators and control algorithms can be implemented. The block diagram of the regulator is shown in Fig. 4.

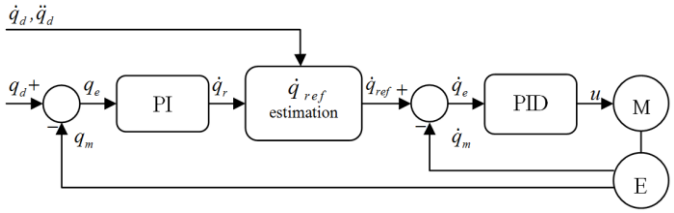

Fig. 4. Block diagram of the cascade position velocity regulator.

The input parameters for the regulator are the desired position q_d, desired velocity $\dot{q}_d$ and desired acceleration $\ddot{q}_d$. A position error q_e is formed by subtracting the measured position q_m from the desired position. The position error is the input for the position regulator. The output of the position regulator is a velocity reference $\dot{q}_r$. This reference goes to the estimation block which limits the maximum speed based on the desired velocity $\dot{q}_d$ and the maximum speed increment based on the desired acceleration $\ddot{q}_d$. The output of the estimation block is a final reference velocity $\dot{q}_{ref}$ that incorporated the desired velocity and acceleration. The reference velocity $\dot{q}_{ref}$ and the measured velocity $\dot{q}_m$ form the velocity error $\dot{q}_e$. The velocity error is the input to the velocity regulator. Finally the output of the velocity regulator is the control signal u.

The PID regulators are shown in more detail in Fig. 5.

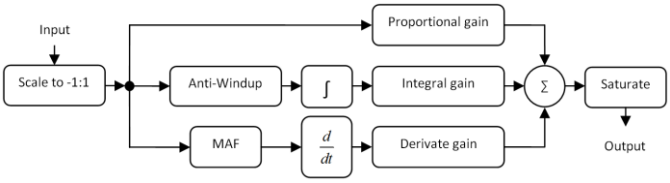

Fig. 5. Block scheme of implemented PID regulator.

As it can be seen in Fig. 5, it is a parallel PID regulator. The PID input value is scaled down to a range from -1 to 1. There is also a saturation block on the output, which is limiting the control value to have the same range as the input. This allows a simpler anti-windup block. The anti-windup is another safety mechanism which will prevent the integral error to wind up to values that would lead to oscillations of the regulator. This issue appears when the reference value is abruptly changed, and the error signal can become much bigger than the control value. The anti-windup block is introduced in such a way to allow the integral effect of the regulator to work only when the error is inside the range of the control value.

Finally the derivate effect is introduced to oppose the intensive change of the error. It is well known that the derivate effect is very sensitive to noise. Usually the noise signal is of high frequency, and it can cause the regulator to start to oscillate. To solve this issue a noise filter is introduced. It is a type of low pass filter that is calculating the average value of the error signal. A moving average is introduced, which takes into account n previous values of the error signal. This will filter out the high change caused by the noise signal. The equation of the moving average is:

$$e_{avg} = e_{avg} + \frac{e_i - e_{avg}}{n},$$

where e_i is the current measured value, n is the number of previous measurements taken into account and e_{avg} is the averaged value.

Finally as the output of the PID regulator is saturated to have the value between -1 and 1.

VI. CONTROL ALGORITHM

The control architecture of the robot's arm is planned to consist of a higher level controller (i.e. master) with the kinematic and dynamic model that will be used for motion planning and implementation of higher level control algorithms. The robot will be equipped with as many lower level slave controllers as needed for driving all the motor. The role of the master will also be the management of communication with slave controllers via the RS-485 bus. The information exchanged between master and slave controllers are the reference position, velocity and acceleration (from master to slave) as well as measurements of position, velocity and current (from slave to master).

The control algorithm is as follows. After the system initializes it enters an endless loop. In parallel with the control loop several processes and interrupts are executing in the background. First there is a system timer running that measures time. It generates an interrupt on every 100µs. Within this interrupt routine the microprocessor checks how much time has passed and rises a flag with the corresponding value. There are four flags that rise on the values of 1ms, 10ms, 100ms and 1000 ms. These flags are used inside the control loop to set up the frequency of actions.

Further, the background process for monitoring the state of the inputs coming from the Hall sensors of the motor is introduced. When a change occurs an interrupt routine is executed which determines the next state of commutation depending on whether the motion of the motor is enabled and what is the direction the motor should rotate.

Finally the background process for checking the serial communication inputs is introduced. Each time there is serial data incoming an interrupt routine executes that saves the data. The data received are later processed inside the control loop at a certain frequency.

Inside the endless loop, the algorithm checks for the time flags and when the time flag is set following actions occurs. Every 1ms the speed and position of the motor are calculated, and used for the regulator to refresh the output control variable. This means that position and velocity control loop sample time is 1ms. Every 1ms the algorithm checks if there is data

received ready to be analyzed. Every 100ms a LED toggles so that there is a visual feedback if the system is running and on every 1000ms the debug values are sent via serial communication.

VII. CONCLUSION

In this paper the design of the driver for BLDC motors that are used for arms of the humanoid robot MARKO has been described. Also the PID control algorithmhas been introduced for validation and testing. In future work different control algorithms will be implemented and tested since the driver is realized as an open platform for interfacing with different types of motors and sensors. Also the library for RS-485 communication between master and slave controllers will be developed, and the kinematic and dynamic model of robot MARKO will be implemented in the master controller. This will provide a platform for development of higher level control algorithms and sophisticated path planning methods.

ACKNOWLEDGEMENTS

This work was funded by the Ministry of education and science of the Republic of Serbia under contract III44008 and by Provincial secretariat for science and technological development under contract 114-451-2116/2011. The authors would like to thank the company Dunkermotoren for its support and equipment donation.

REFERENCES

[1] Savić, Srđan, Mirko Raković, and Branislav Borovac. "Mechanical design and control algorithm for the arm of humanoid robot MARKO." 6th PSU-UNS International Conference on Engineering and Technology ICET 2013. 2013.

[2] Narioka, Kenichi, Shinpei Tsugawa, and Koh Hosoda. "3D limit cycle walking of musculoskeletal humanoid robot with flat feet." In Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on, pp. 4676-4681. IEEE, 2009.

[3] Siyuan Feng; Whitman, E.; Xinjilefu, X.; Atkeson, C.G., "Optimization based full body control for the atlas robot," Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference on , pp.120,127, 18-20

[4] Horvat, Robert, Karel Jezernik, and Milan Curkovic. "An event-driven approach to the current control of a BLDC motor using FPGA." Industrial Electronics, IEEE Transactions on 61.7 (2014): 3719-3726.

[5] Joice, C. Sheeba, S. R. Paranjothi, and V. Jawahar Senthil Kumar. "Digital control strategy for four quadrant operation of three phase BLDC motor with load variations." Industrial Informatics, IEEE Transactions on 9.2 (2013): 974-982.

[6] Shanmugasundram, R., K. Muhammad Zakariah, and N. Yadaiah. "Implementation and performance analysis of Digital controllers for brushless DC motor drives." Mechatronics, IEEE/ASME Transactions on 19, no. 1 (2014): 213-224.

[7] Aghili, Farhad. "Fault-tolerant torque control of BLDC motors." Power Electronics, IEEE Transactions on 26, no. 2 (2011): 355-363.

[8] Zawirski, K., K. Nowopolski, B. Wicher, D. Janiszewski, B. Fabiański, and K. Siembab. "Gearless Pedaling Electric Driven Tricycle." In Analysis and Simulation of Electrical and Computer Systems, pp. 411-421. Springer International Publishing, 2015.