DC motors and servo-motors controlled by Raspberry Pi 2B

Michal Šustek[1,*], *Miroslav* Marcaník[2], *Pavel* Tomášek[2] and *Zdeněk* Úředníček[1]

[1]Tomas Bata University, Faculty of Applied Informatics, Department of Automation and Control Engineering, 76005 Zlín, Czech Republic
[2]Tomas Bata University, Faculty of Applied Informatics, Department of Informatics and Artificial Intelligence, 76005 Zlín, Czech Republic

Abstract. The expanding capabilities of today's microcontrollers and other devices lead to an increased utilization of these technologies in diverse fields. The automation and issue of remote control of moving objects belong to these fields. In this project, a microcontroller Raspberry Pi 2B was chosen for controlling DC motors and servo-motors. This paper provides basic insight into issue of controlling DC motors and servo-motors, connection between Raspberry and other components on breadboard and programming syntaxes for controlling motors in Python programming language.

Introduction

Nowadays, there is a high demand for wireless devices controlled by Wi-Fi, GSM or Bluetooth [1]. These technologies offer a simplification of our lives in home automation or entertainment in the form of Radio Controlled (RC) models. At the same time, the possibilities of microcontrollers have risen in many applications [2].

Current control systems are based on special purpose devices. However, little attention is given to universal microcontrollers that are able to perform a wide variety of tasks [1, 2]. The remote control via microcontrollers is more popular among researchers and "RC fans" than among the public [3].

The performance growth of microcontrollers has led to their ability to manage complex applications. At the same time, there is a variety of manufacturers of microcontrollers with many diverse types of processors and performance. Raspberry and Arduino belong to the most widely used microcontrollers [4]. Both of them provide high performance, and they can be used in many challenging automation applications.

The second generation of the Raspberry microcontroller provides enough performance to replace a standard PC in some audiovisual and automation applications [5]. With the use of the wireless extension, which can be achieved by a simple antenna, the device becomes in a complex control station.

Section 1 describes a microcontroller Raspberry Pi and its performance. Section 2 shows a basic functionality of DC motors and servo-motors. In section 3 is insight into issue of connection between components. In addition, section 4 describes basic programming in language Python for one component of each type (1 motor and 1 servo-motor).

1 Microcontroller Raspberry

A Raspberry Pi 2B is a small single-board device, which supports Linux-based operating systems, USB connections for mouse, keyboard, Ethernet adapter, and other devices. On Raspberry board are HDMI connector for attaching a monitor and general-purpose inputs and outputs [6]. A MicroSD card is used a storage device and Python is used for programming.

In history Raspberry Pi was created in UK as a low cost platform for teaching computer basics, in particular Python [8]. The first generation of Raspberry was introduced in February 2012. Since this time, a wide variety of Raspberry microcontrollers has been created. In addition, each of these variants have different performance and parameters.

Raspberry Pi 2B was chosen for the purpose of this research. It contains 4-core 900 MHz processor, 1 GB of RAM, 4 USB 2.0 ports, HDMI video output, 40-pin GPIO (General Purpose Input/Output) header.

Fig. 1. Microcontroller Raspberry PI 2B [8].

Raspberry can be used as an unpretentious and cheap replacement of a PC; however, it has wide use in home automation and remote control systems. This is

* Corresponding author: sustek@fai.utb.cz

caused by an opportunity to use GPIO pins, which function is programmable.

1.1 GPIO pins

General-Purpose Input/Output are generic pins on integrated circuit of Raspberry. Behavior of these pins is not defined and they can be programmed according to users' needs. Each pin can be configured as input or output; it can be enabled and disabled; input value can be readable and output value can be readable or writable [8]. In some applications are GPIO pins used as maskable interrupt (IRQ).

Ability of using the GPIO pins is provided by external Python module RPi.GPIO. This module must be imported into the main control program.

2 DC motors and servo-motors

DC motors and servo-motors are main actuators part in each robotic system. These components provide movement and rotation around desired axis. In deeper context, this component provide ability of movement. DC motors and servo-motors are crucial in all robotic projects. [12]

2.1 DC motors

Direct current motors are composed of three main parts (rotor, stator, and commutator). The stator circumferentially is provided with regularly spaced and mutually oppositely oriented main magnetic poles and commutation poles. The poles of the same polarity follow the poles of the given polarity in the direction of rotation of the anchor (rotor). The rotor has coils in the grooves and these coils are connected to a mechanical commutator. The commutator provides the supply of a correctly oriented current to the coils of the rotating anchor so that all currents of the flowing coil side form a torque of the same direction in the magnetic field of the main poles [12].

On magnetic neutral place between main poles of commutator, are placed carbon brushes. The number of the brushes is the same as the number of the main poles.

Current, which flows through anchor windings, creates reactionary magnetic field that deforms and weakens magnetic field around main poles and has effect on commutator magnetic field. A compensating winding is used to suppress the reactionary magnetic field. Today a stator and a rotor are created from isolated dynamo-sheet of metal in modern motors.

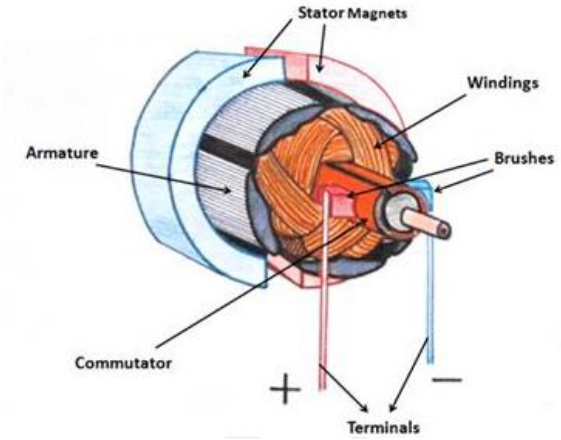

Fig. 2. Brushed DC motor [14].

Brushless DC motors are the second variant of DC motors. Brushless motors provide electrical commutation with permanent magnet rotor and stator with a sequence of coils. A permanent magnet rotates and current carrying conductors are fixed in this type of motor. Transistors or rectifiers at the correct rotor position switch the armature coils in such a way that the armature field is in space quadrature with the rotor fields [12].

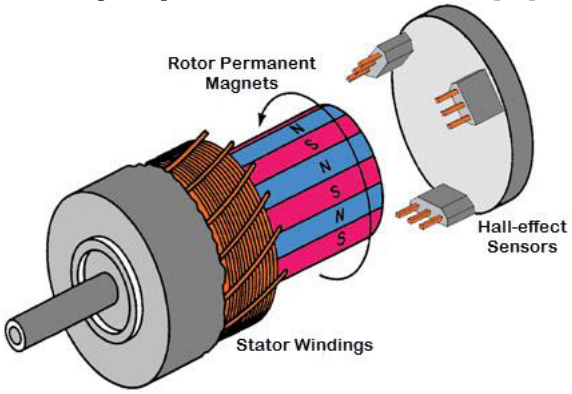

Fig. 3. Brushless DC motor [15].

2.2 Servo-motors

A servo-motor (servo) is a motor which uses a feedback to correct the output of the motor. The feedback is based on information from a sensor or from a sensors and external circuitry. The servo itself is consisted from a DC motor, a potentiometer and a control circuit. They are small but very energy-efficient devices, which are controlled by electrical impulses. Gears for controlling a shaft attach the motor. The power supply of the motor is stopped, when the motor shaft is at the desired position. If it is not in desired position, then it is turned in the appropriate direction. The motor speed is proportional to the difference between desired and actual position [12].

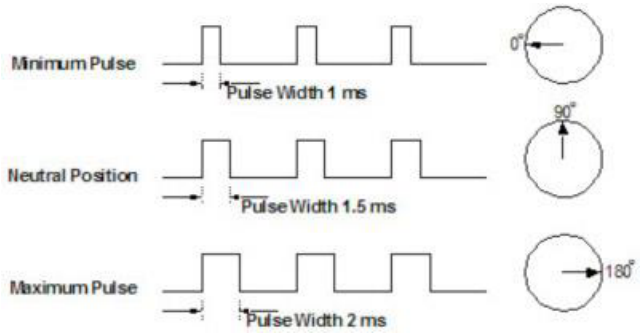

Fig. 4. Servo-motors controlling pulses [8].

Signals are send as electrical pulses with variable width, or a Pulse Width Modulation (PWM) is used. A minimal and a maximal value of pulse and repetition rate is usually distinguished. Usually servos can turn 90º in each direction for total 180º.

The neutral position is defined as the position, where the motor have amount of potential rotation equal to direction in both side (clockwise and counter-clockwise). The servo expects a signal every 20 milliseconds and the length of the electrical pulse determines how far the motor turns. When the servo moves on position, it will hold that position. If any external force try to push against the servo, while it holding a position, the servo will resist from moving out of the position. The maximum amount of force which a servo can resist is called the torque rating. The position will not be held forever, the position pulse must be repeated to instruct servo to stay in desired position.

There are two types of servos [13], AC and DC. AC servos can work with higher current surges and are used in industrial applications. On the other hand, DC servos are not designed for high current surges and are better usable for smaller applications. There are also servos for continuous rotation.

In real application servos are used in RC models (airplanes, walking robots), service robots and operating grippers.

3 Motor connection

A small 6V DC motor PERMAX 280, a microcontroller Raspberry Pi 2B, and an H-bridge L293D were used in this project. The H-bridge is a simple circuit, which contains switching elements. These elements are usually Field-Effect Transistors (FET) transistors. All switches can be turned off or on independently. The H-bridge can be used to switch a direction of a motor depending on a current flow. Table 1. presents how switch state can effect the behavior of a motor.

Tab. 1. H-bridge switch combination [8].

S-1	S-2	S-3	S-4	Motor behavior
1	0	0	1	Clockwise turns
0	1	1	0	Counter-clockwise turns
0	0	0	0	Stop
1	1	-	-	Short circuit
-	-	1	1	Short circuit
1	0	1	0	Braking
0	1	0	1	Braking

Where **1** means the switch is closed, **0** means the switch is opened, and – means it does not matter on the state of a switch.

The H-bridge L293D contains 2 H-bridges, so it is useful for 2 DC motors. The main advantages of this chip are:

- Thermal protection
- Capable with Raspberry logic (3V)
- Voltage range of 4.5 to 36V for motors
- Motor's peak current of 1.2A
- Continuous motor current of 600 mA

In this work, all the components are connected on a breadboard as depicted in Figure 5. The main advantage of using L293D with Raspberry (it has 3V logic) is that the control of pins need only a little current to control motors.

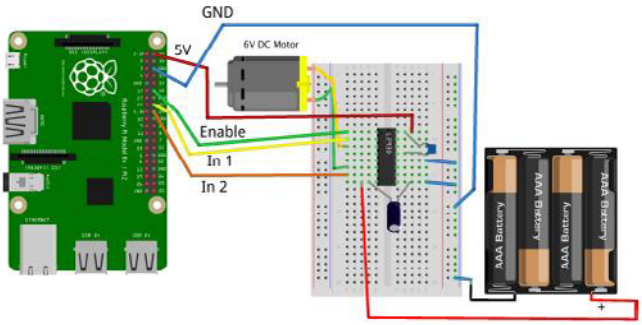

Fig. 5. DC motor wiring with Raspberry Pi 2B [8].

4 Software solution

Python has been chosen as the programming language in this work. Python is a higher programming language, which does not need to have strictly defined variables, unlike C++. In the deeper context, it is a hybrid dynamically interpreted language from a group of scripting languages. Graphic User Interface (GUI) IDLE, which is well organized, was used to develop the software itself. A module called Rpi.GPIO has been used together with basic libraries.

4.1 Solution for DC motor

At the beginning, there is a need to import a module for controlling GPIO pins, in particular RPi.GPIO, which can be downloaded from www.python.org.

```
import RPi.GPIO as GPIO
import time
```

Then the mode for GPIO pins must be set up. BOARD and BCM setup can be chosen. BOARD option specifies referring of the pins to the plug. BCM option means referring of the pins by the Broadcom SOC channel.

```
GPIO.setmode(GPIO.BCM)
```

There is also a need to select GPIO pins, which will be used. The H-bridge L293D is used as a motor driver.

Therefore, pin number 18 must be enabled to control the speed of the motor.

```
enable_pin = 18
pin_1 = 23
pin_2 = 2
```

All used pins will be configured as outputs. These pins help to control direction of the motor. We also define PWM analog output, where 500 is PWM frequency. The initial value of duty cycle is set to 0% of the frequency.

```
GPIO.setup(enable_pin, GPIO.OUT)
GPIO.setup(pin_1, GPIO.OUT)
GPIO.setup(pin_2, GPIO.OUT)
pwm_motor=GPIO.PWM(enable_pin,500)
pwm_motor.start(0)
```

In the following block of code, the movement function for each direction (forward, backward) and stop function are defined. Pin 1 is responsible for forward movement, on the other hand pin 2 is used for reverse movement.

```
def forward(duty_cycle):
    GPIO.output(pin_1, True)
    GPIO.output(pin_2, False)
    motor_pwm.ChangeDutyCycle(duty_cycle)

def reverse(duty_cycle):
    GPIO.output(pin_1, False)
    GPIO.output(pin_2, True)
    motor_pwm.ChangeDutyCycle(duty_cycle)

def stop():
    GPIO.output(in_1_pin, False)
    GPIO.output(in_2_pin, False)
    motor_pwm.ChangeDutyCycle(0)
```

The following part of the program represents the main loop which prompts the user for a command and calls direction functions and the stop function.

```
try:
    while True:
        direction = raw_input('w – forward, x – reverse, t - stop')
        if direction[0]=='t':
            stop()
            else:
                duty_cycle= input('Duty cycle (0-100%)')
                if direction [0]=='w':
                    forward(duty_cycle)
                elif direction [0]=='s':
                    reverse(duty_cycle)

finally:
    print("Cleaning up")
    GPIO.cleanup()
```

This is elementary program for control DC motor with the microcontroller Raspberry Pi 2B. This program is written for one motor, which is wired to the microcontroller by L293D motor driver.

4.2 Solution for servo-motor

The module RPi.GPIO must be used for GPIO control as in the previous case.

```
import RPi.GPIO as GPIO
import time
```

The number of GPIO pin on Raspberry Pi 2B has to be chosen. In addition, every servo needs a slightly different length of pulse to maximize its range of angles, two constant are used to set the pulse duration between angles 0º and 180º. Values of these variables represent duration of pulse in milliseconds. Frequency is set up to 50 Hz, therefore it is giving a pulse every 20 milliseconds.

```
servo_pin=18
0_deg = 0.5
180_deg = 2.5
frequency= 50.0
```

Some calculations, which are related to the length of pulse, were made for easier future modifications. Period is 1000 milliseconds and is divided by frequency (50 Hz in our case) so the result is 20 millisecond. In addition, if there is a need to change a duty cycle, an interval between 0 and 100 must be used and the constant k can be used to scale an angle to duty value. To convert the pulse for 0º to a corresponding value of duty between 0 and 100, is length of pulse multiplied by constant k. The value of range of duty is calculated by multiplying the span of pulse length by the constant k.

```
period = 1000/frequency
k = 100 / period
0_deg_duty = 0_deg*k
pulse_range = 180_deg-0_deg
duty_range=pulse_range*k
```

The part, which initialize the GPIO pin, is similar to the variant for DC control by Raspberry Pi 2B.

```
GPIO.setmode(GPIO.BCM)
GPIO.setup(servo_pin,GPIO.OUT)
pwm=GPIO.PWM(servo_pin, frequency)
pwm.start(0)
```

In angle definition, we convert angle into duty cycle value, and then we call *ChangeDutyCycle* to set the new pulse length.

```
def set_angle(angle):
    duty_cycle=0_deg+(angle/180.0)*duty_range
    pwm.ChangeDutyCycle(duty_cycle)
```

The main loop of control program is written below. In the following part, a value of angle between 0º and 180º is set. It could be used for steering, when 90º is forward direction, 0º is turning left and 180º is turning right.

```
try:
    while True:
        angle = input("Angle 0º to 180º")
```

```
        set_angle(angle)

finally:
    print("Cleaning up")
    GPIO.cleanup()
```

Conclusion

The high demand for wireless devices, which can simultaneously perform a wide variety of tasks has led to significant use of microcontrollers. This paper provides elementary insight into the issue of microcontrollers and the controlling of DC motors and servo-motors.

The microcontroller Raspberry Pi 2B has been chosen as the brain of this project. This microcontroller provides enough performance to run control system for DC motors and servo-motors. 6V DC motor PERMAX 280, motor driver L293D and breadboard have been chosen for creation of this system. This connection is only experimental and it is not used in real robotic platform yet. The controlling software is written in programming language Python; however, we also need to use GPIO pins. The module RPi.GPIO must be imported into the program. The submitted syntaxes in programs are the elementary way to control DC motors and servo-motors. It can be modified for controlling more motors and servo-motors.

The next work is planned to be focused on extending the control system on a cell phone as a control device, in particular, by using LTE (Long Term Evolution) technology. The main difference in LTE system will be in control software (Android, iOS application). An LTE modem must be added to the entire system. Another further work can be aimed at implementation of this control system into a 4-wheeled robotic chassis.

Acknowledgment

This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic within the National Sustainability Programme project No. LO1303 (MSMT-7778/2014) and also by the European Regional Development Fund under the project CEBIA-Tech No. This work was also supported by Internal Grant Agency of Tomas Bata University under the project No. IGA/FAI/2017/004.

References

1. A. A. Asadi, S. Bagheri, A. Imam, E. Jalayeri, W. Kinsner, and N. Sepehri. *Institute of Electrical and Electronics Engineers Inc.* (2016)
2. K. Chaitanya, G. Karudaiyar, C. Deepak, S. B. Reddy, *1st International Conference on Data Engineering and Communication Technology*, **469** Springer Verlag. (2017)
3. I. Lobachev, E. Cretu *Institute of Electrical and Electronics Engineers Inc.* (2016)
4. A.C. Martinez, *International Conference on Human Factors and System Interactions,* **497** Springer Verlag. (2016)
5. P. Membrey, D. Veitch, R. K. C. Chang. *Association for Computing Machinery*, (2016)
6. M. Šustek, M. Opluštil, Z. Úředníček, 6th *International Masaryk Conference for Ph.D students and young researchers*, (2015)
7. M. Vanitha., M. Selvalakshmi, R. Selvarasu. *Institute of Electrical and Electronics Engineers Inc.* (2016).
8. S. Monk. *Make: action: Movement, light, and sound with Arduino and Raspberry Pi.* San Francisco, CA: Maker Media. (2016)
9. H.L. Dai, L. Wang. *Communications in Nonlinear Science and Numerical Simulation* **46**: 116-125 (2016)
10. K. Premkumar, K. G. J. Nigel. "Smart Phone Based Robotic Arm Control using Raspberry Pi, Android and Wi-Fi."*Institute of Electrical and Electronics Engineers Inc.* (2015)
11. D. Sanchez-Benitez, J. M. de la Cruz, G. Pajares, D. Gu. "Visual Control of a Remote Vehicle." *Intelligent Robotics and Applications*, Pt Ii 7102: 579-588 (2011)
12. M. Ghosh, S. Ghosh, P.K. Saha, G.K. Panda, *IEEE Transactions on Industrial Electronics*, **64** (2017)
13. R.S. Ashok, Y.B. Shtessel, *American Control Conference (ACC)* (2016)
14. Todd H. Hubing, Clemson University [online], Available from: http://www.cvel.clemson.edu/auto/actuators/images/chaudhary-DCmotor.png (2017.5.18)
15. Electrical Technology, UK, Birmingham, Available from: http://www.electricaltechnology.org/2016/05/bldc-brushless-dc-motor-construction-working-principle.html (2017.5.18)