Progressive Regenerative Braking using Real Time Speed Sensing

Mike Peterson
Roboteq, Inc
Scottsdale, AZ

One of the most interesting features of electric motors when used in drive train application is that they can also behave as generators, and thus recharge the vehicle's battery while braking. Roboteq's motor controllers can easily be programmed to take advantage of this characteristic to create regenerative braking in a controlled and progressive manner. This article describes the theory behind a simple and very effective technique that uses the motor speed sensing. It describes a practical example using a brushless motor connected to a Roboteq controller.

Experimental electric scooter capable of progressive regenerative braking using Roboteq Brushless motor controller

Motor and generator at the same time

The simplified model of a motor is a resistor in series with an inductor and a voltage generator. The resistor and inductor are simply the resistance and inductance of the electromagnets inside the motor. The voltage generator represents the voltage that is created by the motor while it is

turning and isreferred as Back EMF, or BEMF. The BEMF voltage is a fixed ratio of Volts per RPM.

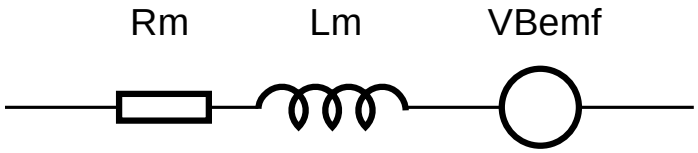

Fig1. Model for electrical motors

When the motor is locked mechanically and a voltage is applied, the model is essentially a resistor across the battery and the current is measured as I=VBat/Rm. The inductance only affects the current the instant the voltage is applied and its effect disappears if the voltage remains constant.

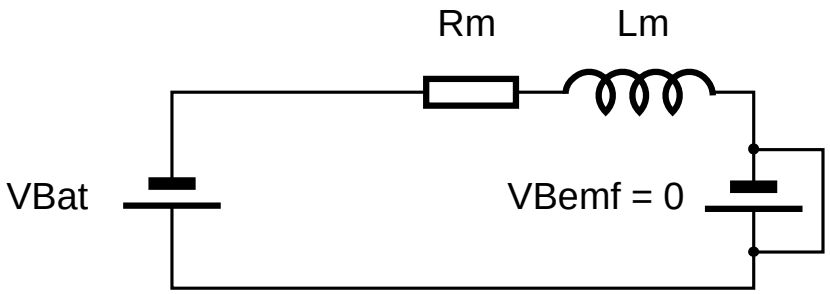

Fig 2: Equivalent circuit when motor is stopped or stalled

If the motor is allowed to spin, then it will generate a BEMF voltage proportional to its rotation speed. The model therefore is now a resistor with generators on both sides. The resulting voltage at the resistor is now the battery voltage minus the BEMF, and the current is I = (VBat - VBemf) / Rm. Practically, this means that as the motor accelerates, the current decreases.

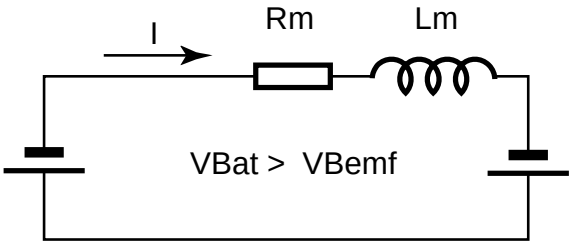

Fig 3: Acceleration condition

If the motor could spins fast enough for the backEMF to equal the battery voltage, the two voltage sources would cancel each other and the equivalent voltage at the resistor would be 0. Therefore no current would flow from the battery. In practice, this cannot happen because this would mean that the motor would have no torque at all, while some is needed to overcome the friction.

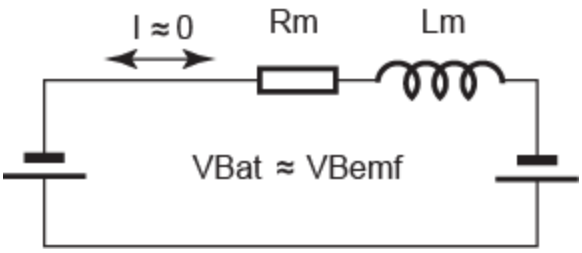

Fig 4: Stable speed without load or friction

In practice, the motor speed will stabilize when the BEMF is such that the current from the battery, which in turn creates torque, is sufficient to overcome the friction and motor mechanical load.

However, if the motor is driven by an external force (the vehicle going downhill, for example), the rotation can cause the BEMF to be exactly equal to VBat and no current at all will flow.

If now, the motor is spining faster and the BEMF becomes larger than the battery voltage, we can see that current flows from the motor to the battery. We are now in a regeneration situation.

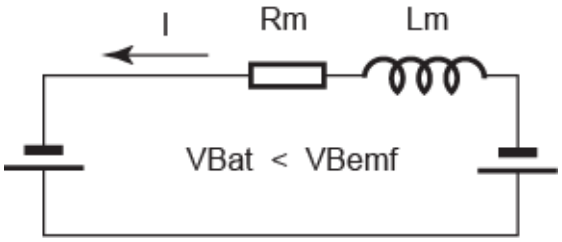

Fig 5: Regenerative braking condition

Effect of PWM Switched Voltage Source

So far we have assumed a fixed voltage battery that is directly connected to a motor, and we have seen that at constant load, speed is controlled by varying the battery voltage.

In modern controllers, varying the voltage is achieved by turning on and off the power to the motor at a very fast rate around 20kHz using power MOSFET transistors arranged in half bridge (unidirectional) or full bridge (bidirectional) configuration. The bridge has a bottom MOSFET and a top switch which are activated in a complementary manner (bottom if on while top is off, bottom is off while top is on). When combined with the inductance of the motor, this switching as has the net effect of making the controller behave like an adjustable voltage source that is proportional to the switch on/off duty cycle. For instance at 50% on and 50% off, the circuit is equivalent to a generator of half the battery voltage.

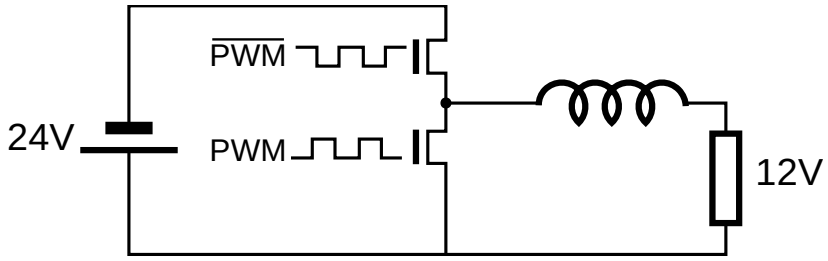

Fig 6: Step down conversion at 50% PWM

This effect works in the opposite direction as well. If no battery is connected and the motor is being driven to generate a voltage, the PWM switches and motor inductance act as a voltage booster that doubles the voltage at 50% PWM. This boosting effect explains how regeneration can occur even as the motor spins slowly and the Back EMF is lower than the battery voltage.

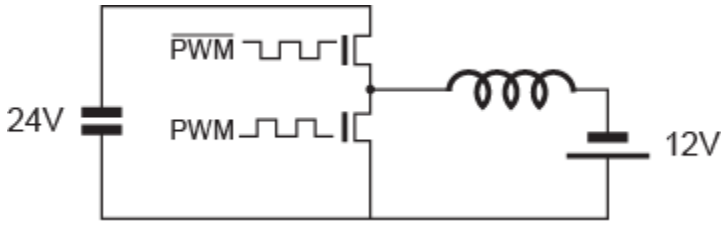

Fig 7: Step up conversion at 50% PWM

Controlling the amount of Regenerative Braking

We have seen that because of the BEMF, for any supply voltage there is a speed at which the motor will draw no current at all. The ratio of BEMF/RPM is a constant that is sometime published in the motor datasheet. It can also be easily be measured by spinning the unconnected motor at a known speed and measure the voltage at its leads.

Then, provided that the motor speed can be measured, the controller can compute the motor's BEMF be made to do any of the following:

1- Match the motor's BEMF, in which case the motor neither accelerates nor brakes. If it is stopped is stays stopped. If already moving, it maintains that speed.

2- Exceed the motor's BEMF, in which case the motor will accelerate

3- Be lower than the motor's BEMF, in which case the motor will brake and regenerate current. The greater the difference between the motor's BEMF and the controller's output voltage, the stronger the regeneration and braking will be.

Assuming that the battery level will remain fairly constant, a simplified approximation of the above theory can be done by figuring the ratio of the controller's power output PWM level and the resulting speed at that level.

Practical example using Speed-adjusted PWM

Roboteq motor controllers are capable of sensing the rotation speed of the motor. On brushless motors, this is done without any additional hardware, by monitoring the changes of the motor's hall sensors. On brushed DC motor, speed can be sensed using an optical encoder mounted on the motor shaft. Applying 50% power and recording the speed of the unloaded motor will give us the ability to know what power level is necessary to near the motor's BEMF.

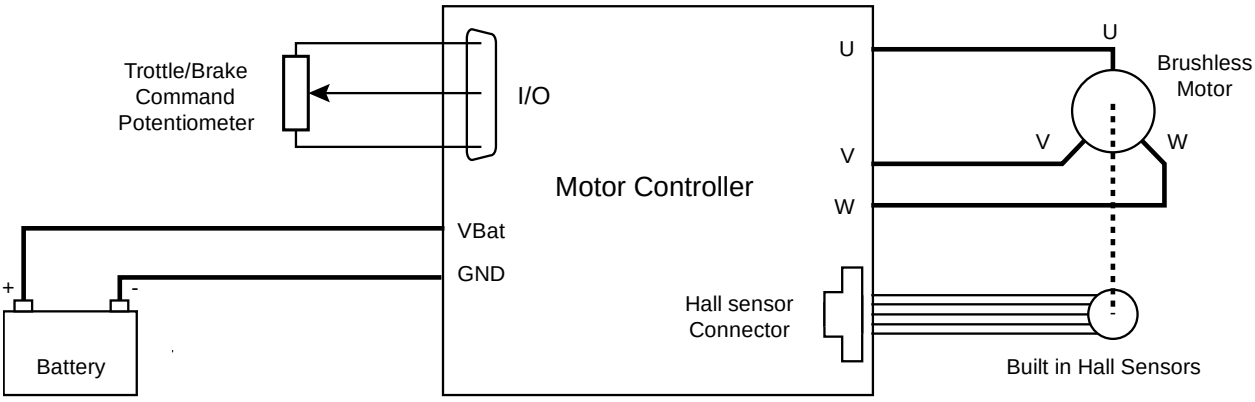

Fig 8: Motor connection of Roboteq motor controller

Using the controller's Microbasic scripting language, we can now easily write a script that will measure the motor's current speed and compute the power level that must be output to accelerates, brake, or maintain the current speed.

For example, consider the case of a vehicle where we measured that at 50% PWM, the motor turns at 2500 RPM when the wheel is not touching the ground.

If now the vehicle is on a slope where the motor is spinning at 2500 RPM, the controller will measure the RPM and it will apply 50% power to it, and so it will consume practically no current (current will be the no load current at that speed). This is because when on a slope that makes the motor run naturally at 2500, the electromechanical conditions are exactly the same as when the wheel is not touching the ground.

Now, if we apply more than 50% power while the motor already spins at 2500 RPM, this will give extra power and the motor will try to run faster

If we apply less than 50% power while at 2500 RPM, there will be regenerative braking. The lower the controller output is 50%, the stronger the braking.

The technique is therefore to let the controller measure speed, compute the power output to apply to it to neither accelerate or brake at that RPM. Then use the throttle command to increase or

lower that power level in order to either accelerate or brake.

The script below is the complete listing needed to achieve controlled regeneration. It uses a single user command for throttle and brake. It is written to operate the motor only in the forward direction.

```
option explicit

dim RPMat50pct as integer ' Measured RPM at 50% power level during bench test
dim CommandStrenghtPct as integer ' Desired strenght of acceleration and braking

dim MeasuredSpeed as integer ' Measured MeasuredSpeed
dim NeutralPower as integer ' Computed Level so the motor neither accel or decelerates
dim AppliedPower as integer ' Power that will be applied to the motor
dim ThrottleCommand as integer ' User command

' Change these constant to meet your requirements
RPMat50pct = 6000 ' Measured RPM at 50% power level and no load
CommandStrenghtPct = 50 ' Strenght of throttle and braking effect in %

top:

' Read read operating values from controller
ThrottleCommand = getvalue(_CIA, 1) ' Read user command from Analog Joystick on in 1

' Uncomment on of the two following lines depending on motor type and sensor
MeasuredSpeed = getvalue(_BS, 1) ' Measure current speed using Hall sensor
'MeasuredSpeed = getvalue(_S, 1) ' Measure current speed using Encoder

' Compute Power Level needed to maintain current speed
NeutralPower = (MeasuredSpeed * 500) / RPMat50pct

' Add power to accelerate or remove power to decelerate
AppliedPower = NeutralPower + ((ThrottleCommand * CommandStrenghtPct) / 100)

' Only allow positive commands and cap to +1000
if (AppliedPower < 0)
       AppliedPower = 0
elseif (AppliedPower > 1000)
       AppliedPower = 1000
end if

setcommand(_G, 1, AppliedPower) ' Apply power to the motor

' Optional log printed on the console for monitoring or debug
print("C= ", ThrottleCommand,"\tS= ",MeasuredSpeed,"\tN= ", NeutralPower,"\tP= ",
AppliedPower,"\r")

wait(10) ' Wait 10ms
goto top ' Repeat loop forever
```

Configuring, testing and improving the script

To use this script, it is first necessary to measure the RPM reported by the controller at 50% power level. Then enter this value as a constant in the top part of the script. Then with the command joystick centered and the script running, the motor will remain idle. Turning the motor

shaft by hand will result in power to be applied to the motor and it will have almost no resistance in the forward direction.

Then applying the joystick command in the forward direction, the motor will accelerate, and brake when the joystick is in the reverse direction. The amount of acceleration and braking for a given joystick position can be adjusted by changing another constant in the script.

The script assumes a stable voltage is present at the battery so that a given power level will always result in the same output voltage equivalent. For example with a 24V, 50% RPM will result in 12V at the motor. If it is expected that batteries voltage will fluctuate significantly, the script should be modified to take account the battery's current voltage. For example, to still have 12V at the motor while the battery is 18V instead of 24V, the power level must now be 75% instead of 50%.

The script can also easily be modified to use a separate throttle and brake pedal by capturing two analog inputs and computing a resulting command depending how hard each is pressed, and prioritizing the brake over the throttle, for example. The strength of the braking can also be made to be higher or lower than this of the acceleration.

Then very strong braking is required, the script can be modified to activate an electromechanical brake connected to one of the controller's digital outputs.

Benefit compared to other controlled regenerative braking techniques

For all practical purposes, this technique that uses the measured the motor's actual speed to compute its BEMF results in drive characteristics that are similar to torque mode drive, where the controller will attempt to feed the motor with constant current.

Compared to torque mode, the method described here is more accurate because it is typically simpler and more accurate to measure speed than it is to measure motor current. Most motor controllers, including Roboteq's, measure the battery current whereas it is the motor current that must be regulated in torque mode. Motor current is approximated from the measured motor current and will be very inaccurate at low power levels. Motor current cannot be measured at all when the power level is 0.

In addition, in torque mode, the controller uses a regulator, typically a PID, that will continuously adjust the power level in order to move the measured current to the desired value. The PID needs tuning which is not always simple given the very variable load condition of a motor in transportation applications. The control loop continuous "guessing" is not as quick or accurate as the immediate computation of the motor's BEMF from the measured speed.