Modern Robotics (Hitechnic) CORE Motor Controller.

Motor control characteristics.

By Phil Malone. Updated 7/12/2015

1.0 Scope.

This documents attempts to characterize the control modes of the new Modern Robotics Core Motor controller. This study was performed primarily to see if the new Modern Robotics controller exhibits the same operational flaw that was reported in the 2011 version of the original Hitechnic Motor controller.

A test bed was created that utilizes the new standard FTC robot controller setup: That is, a ZTE Speed phone connected to a Core Power Distribution Module and a Core Motor controller. A freshly charged 12V DC battery was used for each test (Approximate voltage was 14.0V)

This setup was used to run both a TETRIX 12V DC motor, and an AndyMark NeveRest 12V DC motor. Both motors were stepped through 0% to 100% power under all three control modes:

- "A" RUN_WITHOUT_ENCODER
- "B" RUN_USING _ENCODER
- "C" RUN_TO_POSITION.

The motors were unloaded, and so they exhibited No-Load speed responses.

2.0 Summary.

Of the three run modes, mode "B" (RUN_USING_ENCODERS) was the only mode that produced the desired result. The other two modes had significant flaws, which produces unexpected and generally less than desirable results.

Mode "B" (RUN_USING_ENCODER) was the most predictable, linear and consistent across both motors and over the full range of "power" from 0 to 100%. Caveat: given that the AndyMark shaft-encoder produces 22% less pulses per revolution, that motor reached its top speed sooner in the test sequence, but this was in complete accordance with expectations.

Mode "A" (RUN_WITHOUT_ENCODERS) produced the least predictable and consistent results. The MATRIX motor has an initial dead-band, where the output voltage was very low with no motor rotation, then this was followed by a sudden jump to 60% of full voltage and RPM with only 20% power requested. The AndyMark motor did actually start rotating at the initial low power setting, but then proceeded to follow the same sudden rise to 60% voltage and RPM. Neither the voltage output, nor the motor response, was remotely linear, even with no-load. It's also intriguing that the measured voltage for 0-15% power was different for the two motors. Apparently the motor windings and internal protection devices are affecting the PWM of the applied voltage.

Mode “C” (RUN_TO_POSITION) exhibited exactly the same clipped response that was reported on the original TETRIC Motor controller. When being asked to run to a very large encoder value, both motors performed in a linear fashion (RPM vs Pwr) up until approximately 65% power. At that point, the drive voltage and resultant RPM did not increase and further. This was despite the increasing power request. Some factor within the motor controller is clipping the maximum “run to position” speed to 65% of full speed. So “Run To Position” is suitable for slow motor control, but not for time-critical Autonomous driving.

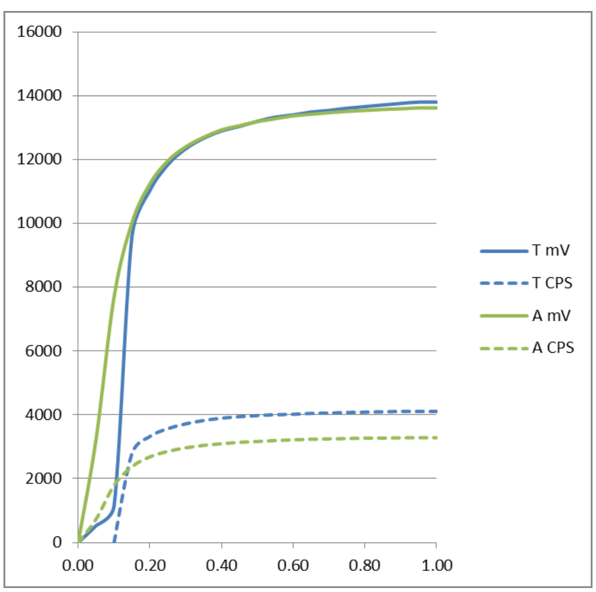
The motor response to each of the three control modes is described in the following analysis section.

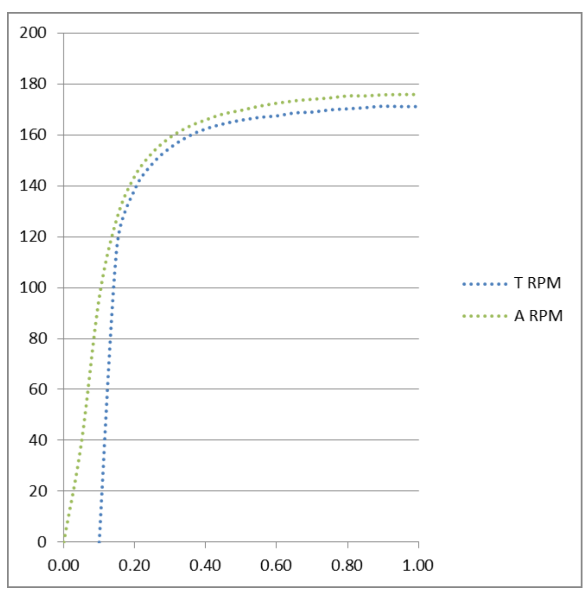
3.0 Analysis

The following charts show the performance of both motors under the three control modes. In all cases, the TETRIX motor (T) is in Blue, and the AndyMark motor (A) is in Green.

3.1 Open Loop (RUN_WITHOUT_ENCODER)

The open loop controller performance is somewhat perplexing. The driver voltage output is extremely non-linear. Most noticeable is that the controller output voltage jumps up to 11V before the requested power is even at 20%. This leaves very little headroom for the remaining 80% power range. With this extremely non-linear output, it will be very difficult to achieve controlled slow-speed actions without the aid of encoders. Note the lower encoder count rates (CPS) for the AndyMark motors in green. This is expected based on 1120 CPR vs 1440 CPR.





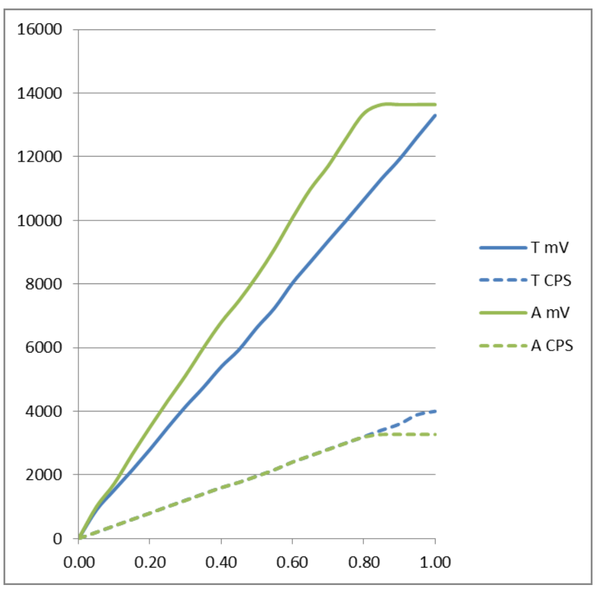
3.2 Closed Loop Control (RUN_USING_ENCODERS)

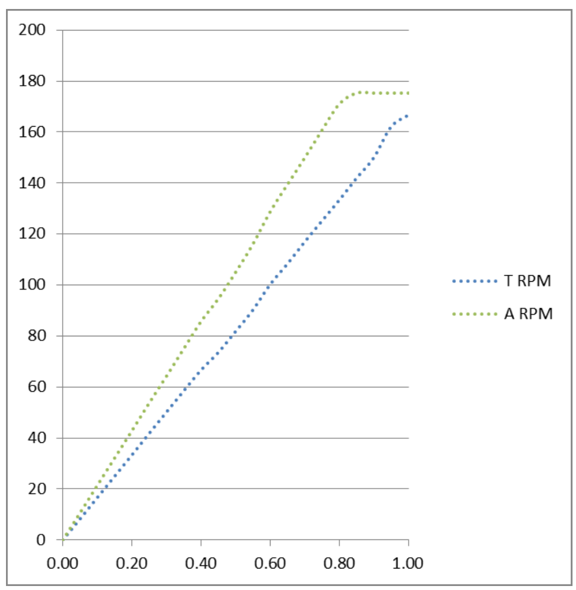
In this mode, the encoder is used to measure the rotational speed of the motor, and the controller adjusts the applied voltage to obtain the desired speed.

From the charts below it can be seen that the Controller does an excellent job of regulating the output voltage to produce the desired speed characteristic. Several other things should be noted.

1. The Controller is optimized for the encoder counts of the TETRIX motor. This motor runs from 0 to 100% speed in a very linear fashion, and achieves full speed at 100% “power”. The AndyMark motor has fewer pulses per revolution, so the controller increases the applied voltage faster to get the desired pulse rate. This means that full speed is obtained at about 78% “power” on the NeveRest motor.
2. Both motors are able to reach, and maintain, the full closed-loop speed. This max speed has not been limited artificially to provide any extra power overhead.
3. The applied voltage ends up being approximately proportional to the output speed, so there is no clear reason why a proportional voltage is not being used in the motor in Open Loop mode.

Note: It is unfortunate that the software API does not discriminate between setting the desired ***Power***, vs. the desired ***Speed***. Historically SetPower implies Open Loop, and SetSpeed implies Closed Loop. This convention is ignored in the current API. There is no mention of “Speed” anywhere in the current API.

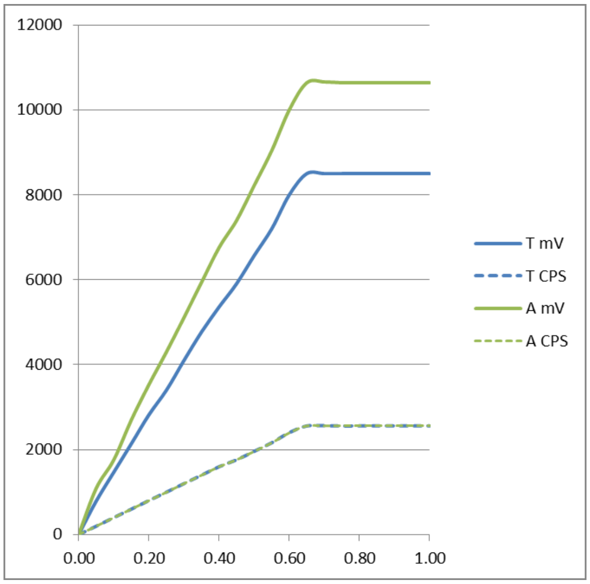


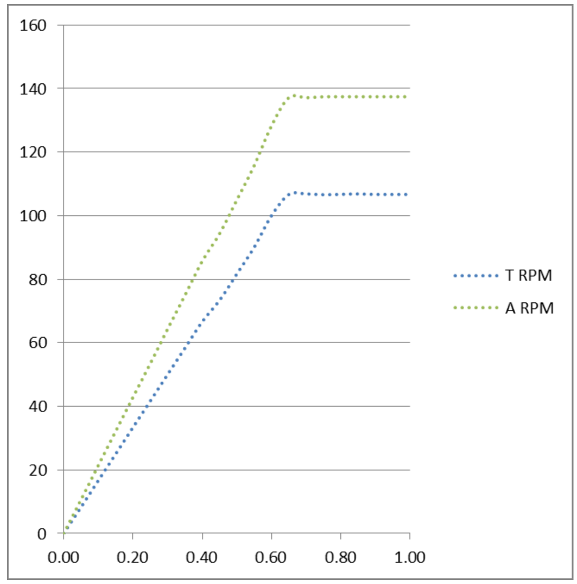


3.3 Run to Position (RUN_TO_POSITION)

To test the “Run to Position” mode speed characteristics, the controller was directed to run to a very large encoder count, so it should have run at the requested “power” for the full duration of each 10 second speed step. Instead it can clearly be seen from the charts below that controller stops increasing the applied voltage once 65% power has been reached. It’s unclear whether this is a bug, or an intentional action. From the earlier tests it can be seen that the controller is capable of regulating the speed up to the full 100% so there doesn't seem to be any need to artificially clip the max speed.

The only good news here is that since the AndyMark motor is reporting less encoder counts, it runs closer to its full speed before the controller clips (clipping at 10.6V rather than 8.6V)





It's worth noting that this 65% clipping action was also present in the previous Hitechnic controller (as reported in 2011), so if it is a bug, it's 4 years old.

4.0 Conclusion.

While very capable in the "Closed Loop" (RUN_USING_ENCODERS) control mode, the new Modern Robotics Motor Controller seems to be flawed in both the "Open Loop" and "Run to Position" modes. Anyone expecting to achieve a smooth (and full) range of motion control on their robot should definitely plan to use encoders on all motors, but should avoid using the "Run to Position" mode.

It's unclear as to what other enhancements were made by Modern Robotics for this new FTC controller, but in terms motor control capabilities, the new devices appear to be just a repackaged version of the 6 year old Hitechnic controller, with a faster USB interface added. Considering the amount of time since its predecessor was released, and the fact that the deficiencies have been known for a long time, this new controller feels a lot like opening the case of a new iPhone 6, and finding an iPhone 4 inside.

Given the guaranteed (captive) FTC market for this new unit, and the likely long product lifecycle, it would have been nice for them to have cleaned up the control logic somewhat.