Current-Limiting Motor Control

Chris Gerth - Jeremy Lee - FRC 1736 Robot Casserole

v0.2

Abstract

A method for limiting motor voltage commands is presented. The target controlled variable is system voltage - too much current draw from the motors will lower system voltage to an unacceptable level. This limiting is achieved through calculations involving an adaptive observer plant model which estimates battery parameters in real time, and projections of current draw from motors.

Contents

Part I
Problem Statement, Constraints, and Assumptions

1 Introduction

A brownout is the undesired shutdown of various electrical components due to low system voltage. We will show an algorithm to prevent brownouts. The algorithm is based in the underlying physics of the electrical components, and should provide optimum performance of the drivetrain.

2 Overview

Most brownouts occur due to transient current spikes from collisions or sudden changes in driver commands. These current draw spikes in turn drop system voltage to an unacceptably low level. Our algorithm involves calculating crucial parameters which define battery performance under load. This is combined with estimated motor current to limit the load applied to a battery. Our method allows for optimal mechanical design of the robot, but actively ensures system voltage does not drop below a preset minimum threshold (referred to herein as $V_{sys,min}$). By limiting the system voltage drop in software, the robot can be designed to be arbitrarily powerful and aggressively-geared - the software will maintain best-cases performance without over-taxing the battery. In the ideal case, the algorithm only requires a preset $V_{sys,min}$ limit (along with a few well-known physical parameters of the motors). It is guaranteed to always run the hardware and electronics at their physical limits, and not impose additional "tuning" constraints seen by other algorithms.

3 Basic Steps of the Algorithm

Limiting system voltage drop involves a multi-step process. The basic algorithm is:

1. Estimate time-varying battery performance parameters.
2. Measure motor speeds from each large motor.
3. Determine the driver requested voltages for each large motor.
4. Estimate the current draw **if** the driver demanded voltages were sent to the motors on this time-step.
5. Estimate the system voltage given the estimated current draw.
6. **If** the estimated system voltage is below the preset threshold $V_{sys,min}$, calculate a scaling factor for the driver-demanded voltages which will hold V_{sys}at $V_{sys,min}$.
7. **Else,** set the driver-demanded voltage to the motor (no limiting)

4 Existing Solutions

Mechanical-based solutions for preventing brownouts involve using fewer large motors, minimizing stall conditions via lowering coefficients of friction, or increasing the gear ratio to slow the whole system down. All of these mitigation can reduce overall robot performance.

Multiple software solutions to reduce the effects of system voltage drop also exist. The most common perhaps is to simply limit the maximum rate-of-change of a motor command, effectively low-pass filtering the signal. Fewer extreme changes in applied motor voltage tends to reduce current draw. Another low-bar software solution for

limiting system voltage drop is to monitor either the system voltage or battery current draw, and reducing all motor commands as a preset limit is neared. Any of these may work in many cases. We seek to provide an improved algorithm which would work in more cases. Additionally, the physics-based nature of the algorithm should push toward the maximum possible performance of the robot (constrained by a minimum allowable system voltage).

5 Typical Electrical System Description

During FRC robotics competitions, there are many constraints on the set of electrical components which may be used. The construction of an ever-more-powerful drivetrain is inherently limited by these component constraints, as arbitrarily more-powerful motors and energy sources cannot be used. A common combination of components involves the FR801-001 2.5" CIM motor and the MK ES17-12 battery. There are a maximum of six CIM motors used to power all mechanisms. The motors are fed by electronic PWM controllers, where voltage is the commanded parameter. CIM motors can pull upwards of 120A at stall. Six stalled CIM motors is more than enough to cause controller brownout due to current-draw-induced voltage drop. Even at non-stall conditions, allowing the motors to draw excessive current can quickly pull down system voltage.

For the purposes of this paper, a six-CIM tank drive platform is assumed. This means two sets of three-motor drivetrains with a single gear ratio from motor to wheel, which apply force on the left and right sides of a rigid frame. Rotation is accomplished by driving the wheels at different speeds. Other current draw sources are negligible compared to the drivetrain. However, the algorithms described herein can be adapted to other drive platforms.

6 Technical Prerequisites

Understanding the algorithms described in this paper should require some basic knowledge of circuit analysis (Kirchhoff's & Ohm's laws), and algebra. An introductory E&M Physics course should suffice.

This paper presumes discrete-time math is being used. Therefor, bracketed notation is used to indicate time varying variables. For example:

$$V_{sys}[n]$$

Indicates that V_{sys} is a variable which varies over time, and we are specifically referring to the value of V_{sys} at time-step n. Our time-step variable will start at 0 and advance by 1 each software control loop, always maintaining an integer value. In general a software loop rate of 20ms was presumed, but further discussion of this is beyond the scope of this paper.

Part II
Current Limiting Algorithm

The Current Limiting Algorithm is responsible for establishing a maximum motor voltage command to each set of mechanically-linked motors. The amount of limiting is calculated to be as small as possible, while still preventing system voltage from dropping below the preset $V_{sys,min}$ limit.

7 Data Sources

The key time-varying external input needed for current estimation is the motor's present rotational speed. This may be derived from an encoder attached somewhere on each side of the drivetrain. The motor's signed rotational speed at time-step n in rad/sec can be calculated as

$$\omega_m[n] = K_{enc\_ratio} * \omega_{enc}[n] \tag{1}$$

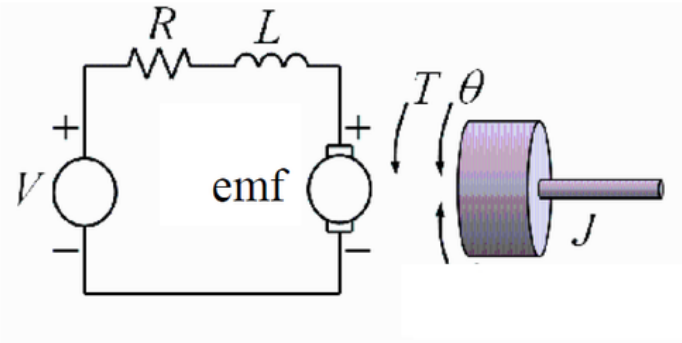

Figure 1: Classical model of a DC Motor

Here, ω_{enc} is the encoder's rotational speed in radians/sec at time-step n, and K_{enc_ratio} is the gear ratio between the motor, and the measuring point of the encoder. If the encoder is attached directly to the motor, K_{enc_ratio} is simply 1. If the encoder is attached on another gear, it will be something other than 1.

Additionally, we will need to know the voltage applied to the motor. This can be calculated based off the present system voltage and the software command to the motor. The software command is normalized to the range $[-1, 1]$, where -1 means "full reverse" and 1 means "full forward". The present system voltage can come from a number of places: For simplicity it can be assumed to just be 12V. It could also be taken from a filtered measurement of the present system voltage (which will be needed anyway in part III). Finally, it could also be derived from the previous loop estimation from this current limiting algorithm. The first solution does not account for decaying battery voltage over the match, and the final one involves math implications beyond the scope of this paper. Therefore, for this analysis, the filtered measurement of present system voltage will be used.

$$V_m[n] = V_{sys}[n] * Cmd[n] \tag{2}$$

8 Motor Model

The classic electrical model for a DC motor involves a two-terminal device circuit, where the two terminals are linked by a resistance and a variable voltages source in series. The resistance represents the electrical resistance of the motor winding and communicator. All inductive effects of the wrapped wire are ignored. The voltage source represents the "back-EMF" induced by the fact that a magnet is spinning with respect to a coil of wire. The current through the series circuit is proportional to the torque of the motor. The speed of the motor shaft is proportional to the back-EMF effect of the voltage source.

From Figure 1, presuming L to be zero, we can apply Kirchhoff's Voltage Law and Ohm's Law around the circuit to arrive at the following relationship:

$$V_m[n] = I_m[n]R_m + V_{emf}[n]$$

For $V_m$being the driving voltage of the motor, $R_m$being the resistance of the motor winding, $I_m$being the current draw of the motor, and V_{emf} being the back-EMF from the output shaft rotation.

Re-arranging, we find the current draw from one motor on one side of the drivetrain is:

$$I_m[n] = \frac{V_m[n] - V_{emf}[n]}{R_m}$$

Again from the classical DC motor model, we assume that V_{emf} is linearly proportional to the rotational speed of the motor. We will call this constant of proportionality K_i. This yields the final relationship

$$I_m[n] = \frac{V_m[n] - K_i\omega_m[n]}{R_m} \tag{3}$$

We have now reduced the total current from one motor to an equation made up of known inputs and constant values.

9 Determining Motor Constants

The needed motor constants are not always explicitly spelled out in a motor's datasheet, but stall and free-wheel currents and speeds usually are. We will derive the needed constants for a CIM motor as an example.

For the CIM motor, we know a few crucial facts: Stall Current is 133A, and Free-wheel speed is 5310 RPM while drawing 2.7A[1]. In both cases, the supply voltage is 12V. Using these relationships, we can solve for the constant parameters in the motor model. Starting with Stall condition, we know $\omega_m = 0$. Therefor:

$$I_m[n] = \frac{V_m[n] - K_i\omega_m[n]}{R_m}$$

$$113 = \frac{12 - K_i * 0}{R_m}$$

$$R_m = \frac{12}{113}$$

$$R_m = 0.1062\Omega \tag{4}$$

Now, using this number, we plug in information for Free-Wheel Speed. Remember to convert RPM to rad/sec

$$I_m[n] = \frac{V_m[n] - K_i\omega_m[n]}{R_m}$$

$$2.7 = \frac{12 - K_i * 5310 * \frac{\pi}{180}}{0.1062}$$

$$K_i = 0.1263 \tag{5}$$

Remember these constants are for CIM motors only. Different motors will need this section to be re-calculated for their parameters.

10 Total Estimated Current Draw

Since all parameters of the motors are now known, we can use the speed from either side of the drivetrain to determine total drivetrain current draw. Again, note this assumes a 6-CIM tank drive setup, with subscript r and l indicating left and right sides of the drivetrain:

$$I_{dt}[n] = 3 * I_{mr}[n] + 3 * I_{ml}[n]$$

$$I_{dt}[n] = 3 * \frac{V_{mr}[n] - K_i\omega_{mr}[n]}{R_m} + 3 * \frac{V_{ml}[n] - K_i\omega_{ml}[n]}{R_m} \tag{6}$$

Again, note this equation for total current draw produces the current at time-step n using only known inputs (V and ω) and constant parameters.

At this point, limiting could be applied based around a maximum desired current draw from the drivetrain. However, since brownouts are caused by system voltage drops, a further step in this algorithm is taken to estimate the system voltage drop induced by this current draw.

[1] See http://content.vexrobotics.com/docs/217-2000-CIM-motor-specs.pdf

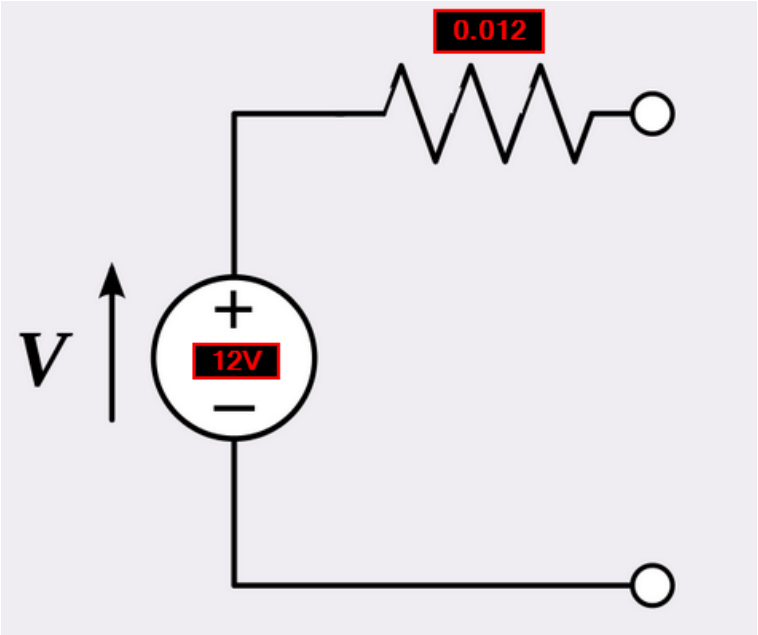

Figure 2: Classical model of a lead-acid Battery

11 Battery Model

Before continuing determining if limiting is actually needed, we must establish what our battery behaves like under load. A battery is classically characterized as an ideal voltage source in series with a resistor. As more current is drawn from the battery, the voltage drop across the resistance causes the output voltage of the battery to drop. As the battery discharges, the series resistance tends to increase, while the open-circuit voltage decreases slightly. This makes a dead battery's output voltage sag much lower when put under a given constant load. See figure 2.

From section 10, we know during every control loop what the current draw from the drivetrain will be if we apply the driver-desired voltage to the motors at the drivetrain. Given this current draw $I_{dt}[n]$, we can then estimate the system voltage via the following equation:

$$V_{sys}[n] = V_{oc} - R_{bat} I_{dt}[n] \tag{7}$$

For V_{sys}being the system voltage at time-step n, V_{oc}being the open-circuit voltage of the battery (the ideal voltage source in figure 2), R_{bat}being the battery's internal resistance, and I_{dt}being the current draw from the robot. This is assumed to be drivetrain-only, all other current sinks on the robot are negligible. Note the battery's open-circuit voltage and internal resistance are considered constants for this part of the analysis. For reference, they are usually around $V_{oc} = 12$ and $R_{bat} = 0.012\Omega$ for a healthy battery.[2] However, later in the analysis, it will be shown how to calculate them over time.

12 Limiting Method

Recall the steps of the algorithm from section 3. Every step except 6 has been demonstrated already. The crucial task now is determining a scaling factor (which we will call γ) to apply to the driver-demanded voltages.

Estimating the system voltage drop can be done by combining equations 6 and 7:

$$V_{sys,est}[n] = V_{oc} - R_{bat}\left[3 * \frac{|V_{ddr}[n] - K_i\omega_{mr}[n]|}{R_m} + 3 * \frac{|V_{ddl}[n] - K_i\omega_{ml}[n]|}{R_m}\right] \tag{8}$$

[2]See http://www.mkbattery.com/images/ES17-12.pdf

For $V_{sys,est}$being the estimated system voltage for driver-demanded motor voltages $V_{ddr}$and V_{dl}. Note the introduction of absolute-value signs to account for the fact that the current direction is inverted at the motor controller, so the battery always sees current drawn out of it (even when going in reverse)[3].

If scaling is needed (when $V_{sys,est} < V_{sys,min}$), we plug in the scaled values to equation 8 and solve for the scaling value γ. Starting from equation 8 and plugging in the known values for the "scaling needed" condition:

$$V_{sys,min} = V_{oc} - R_{bat}\left[3 * \frac{|\gamma V_{ddr}[n] - K_i\omega_{mr}[n]|}{R_m} + 3 * \frac{|\gamma V_{ddl}[n] - K_i\omega_{ml}[n]|}{R_m}\right]$$

$$V_{oc} - V_{sys,min} = R_{bat}\left[3 * \frac{|\gamma V_{ddr}[n] - K_i\omega_{mr}[n]|}{R_m} + 3 * \frac{|\gamma V_{ddl}[n] - K_i\omega_{ml}[n]|}{R_m}\right]$$

$$\frac{V_{oc} - V_{sys,min}}{3R_{bat}} = \frac{|\gamma V_{ddr}[n] - K_i\omega_{mr}[n]|}{R_m} + \frac{|\gamma V_{ddl}[n] - K_i\omega_{ml}[n]|}{R_m}$$

$$\frac{R_m\left(V_{oc} - V_{sys,min}\right)}{3R_{bat}} = |\gamma V_{ddr}[n] - K_i\omega_{mr}[n]| + |\gamma V_{ddl}[n] - K_i\omega_{ml}[n]|$$

Due to the double absolute value symbols, we now have four possible solutions for γ. Doing some nifty algebra [4], we find all four solutions for gamma:

$$\gamma = \begin{cases} \frac{-\left(\frac{R_m\left(V_{oc}-V_{sys,min}\right)}{3R_{bat}}\right)+(K_i\omega_{mr}[n])-(K_i\omega_{ml}[n])}{V_{ddr}[n]-V_{ddl}[n]} \\ \frac{\left(\frac{R_m\left(V_{oc}-V_{sys,min}\right)}{3R_{bat}}\right)+(K_i\omega_{mr}[n])-(K_i\omega_{ml}[n])}{V_{ddr}[n]-V_{ddl}[n]} \\ \frac{-\left(\frac{R_m\left(V_{oc}-V_{sys,min}\right)}{3R_{bat}}\right)+(K_i\omega_{mr}[n])+(K_i\omega_{ml}[n])}{V_{ddr}[n]+V_{ddl}[n]} \\ \frac{\left(\frac{R_m\left(V_{oc}-V_{sys,min}\right)}{3R_{bat}}\right)+(K_i\omega_{mr}[n])+(K_i\omega_{ml}[n])}{V_{ddr}[n]+V_{ddl}[n]} \end{cases} \tag{9}$$

All four possible values for γ would be computed at run-time. From the four possible values, it is known that gamma can be in the range $[0, 1]$ since the scaling cannot exceed the physical limits of what was requested or is possible with the motor controllers. Among all the solutions which are in this range, the largest should be chosen (as the $\gamma = 1$ case is where the driver-demanded voltage is honored). If no solution is within the $[0, 1]$ range, default to $\gamma = 0$. This assumes current cannot be back-driven through an active controller, and friction in the system will push the system back to steady-state with zero control effort.

Once a suitable γ is found, the motor voltages for each motor should be applied as such:

$$V_m = \begin{cases} V_{dd} & V_{sys,est} \geq V_{sys,min} \\ \gamma V_{dd} & V_{sys,est} < V_{sys,min} \end{cases} \tag{10}$$

Part III

Battery Parameter Estimation Algorithm

Up until now, it has been assumed that the battery parameters V_{oc} and R_{bat}. are constants. This is valid for some batteries, but will induce significant error as batteries discharge over a match, and age over many seasons. While the aforementioned numbers are good starting points, it is worthwhile attempting to determine these parameters on the fly, to account for different conditions. Since they are slowly-changing compared to driver inputs and motor speeds, the previous calculations will be unaffected (as the assumption that the battery parameters are "constant" is

[3]Note this is a presumption of a behavior of specific speed controllers. Depending on the speed controller used, your mileage may vary.

[4]See www.wolframalpha.com

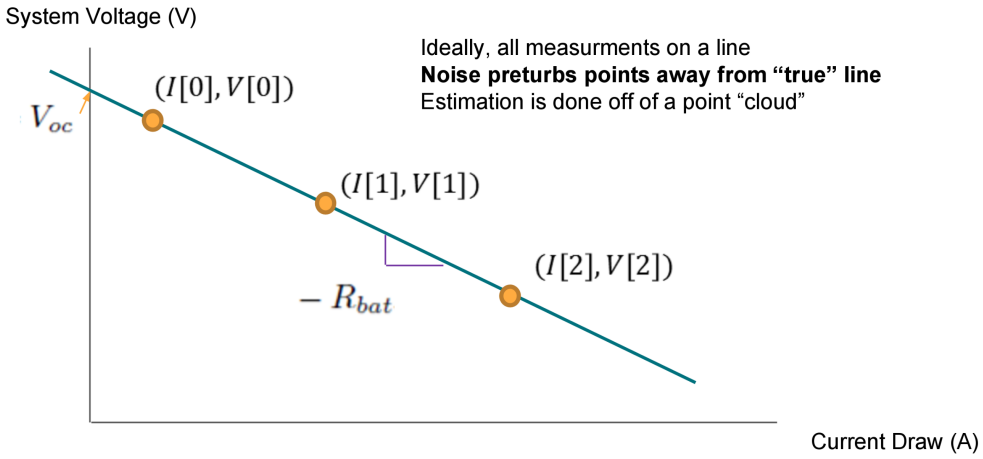

Figure 3: IV curve for ideal estimation of battery parameters

still fairly true with respect to the other time-varying signals). The basic estimation method will use known voltage and current values over multiple time-steps to calculate parameters of the simplified battery model presented in figure 2.

13 Data Sources

The battery model has two degrees of freedom (Open-Circuit voltage and internal series resistance), so two sources of data are needed to fully constrain the system. Starting in 2015, a CAN-enabled Power Distribution Panel was made standard for FRC. Using the API's associated with it, it is possible to read the total current from the battery, and the present system voltage. As mentioned already, these signals are lower-bandwidth and are not the best for measuring sharp transient conditions. However, outside of extreme transients, their values can be filtered down to fairly accurate values for the voltage and current just at the battery's connection to the robot.

The measured current and voltage values for the system form points on the I-V plot for the battery. It should be noted that the slope of this plot represents the equivalent series resistance (R_{bat}), and the y-intercept is the open-circuit battery voltage (V_{oc}). See figure 3. The basic method for determining these two desired parameters is to take many system current and voltage measurements, and use them to create a best-fit line. The slope and y-intercept of this best-fit line are then used to calculate the $V_{oc}$and R_{bat}parameters needed in the current-limiting algorithm. To get an accurate best-fit line, it is required that the current and voltage measurements be spread out over the I-V plane. In other words, a steady-state robot sitting still will not produce meaningful measurements. A changing current draw from the battery is required. See figure 5 R_{bat} in particular is sensitive to steady-state error due to noise. While the current is changing, the estimated R_{bat}parameter can be trusted, but a sample-and-hold methodology must be used when it is not changing much.

14 Estimation Method

Figure 2 is referenced as the model for the battery, with the measured system voltage and measured system current present at its output ports.

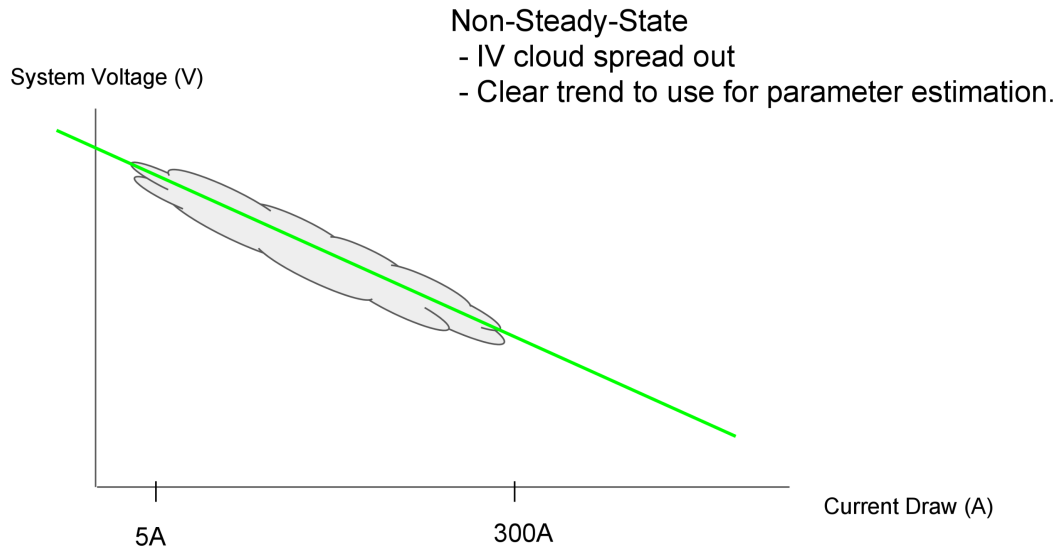

Figure 4: IV curve for estimation with many samples, including noise

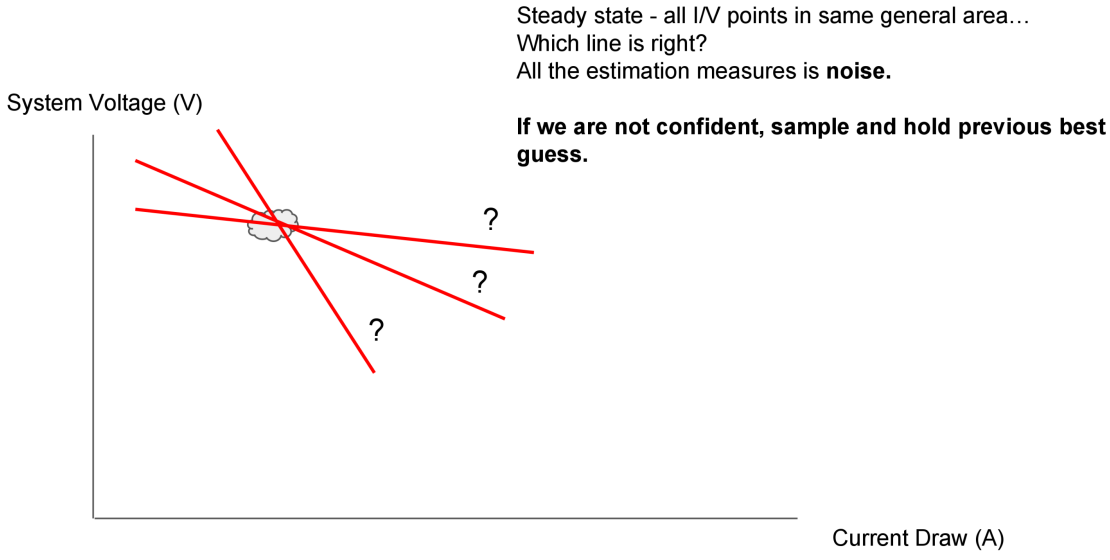

Figure 5: IV curve for many samples of similar current draw

The estimation method can be summarized as follows:

- Read in the present system voltage and current.
- Average some amount of previous data to help reject noise in the measurement.
  - Simulated experiments showed 300ms of data averaging was a good starting point. Make the length of the averaging tunable, and adjust to taste on the actual robot.
    - At 10ms, this equates to the past 30 samples.
  - This will generate two filtered signals - one for the system current, one for the system voltage.
- Consider a window of the filtered data stretching from the present into the past for a given length of time.
  - Simulated experiments showed 1000ms for a window length worked well.
- Estimate the equivalent series resistance of the battery from this window of samples.
  - See section 14.1.2 for the algorithm.
- Calculate the spread of current draw readings in the window under consideration.
  - Here, we define spread as the standard deviation of the points within the window.
- If the spread is above a constant, tunable threshold, use the estimated ESR in all other calculations, assuming it to be correct.
  - Additionally, if this is the biggest spread seen since the last time spread went above the threshold, record the spread and ESR values for later use.
- If the Spread is below the threshold, use the previous best-spread ESR. (sample and hold most-confident value).
  - Reset the "largest spread" threshold to zero as well in prep for the next time spread goes above the threshold.

The results of such an algorithm can be seen in the provided figures: Note that when confidence goes to "1", the percent error goes to nearly zero and the estimated ESR "catches up" with the actual one. When information is sparse in-between large current draw changes, the best last known-good value for ESR is held. This is a reasonable approximation, as percent error stays mostly below 10% for both ESR and Voc. The figures illustrate the algorithm working in the presence of noise in the readings for system voltage and current.

14.1 Calculations

14.1.1 Averaging Filters

The algorithm refers to utilizing an averaging filter to eliminate some of the noise from the measured input $I_{sys}$and V_{sys}readings. An averaging filter has one input and one output - the output is always equal to the average (or arithmetic mean) of the last N inputs, where N is said to be the "length" of the filter. For example, on the system voltage,

$$V_{sys}[n] = \frac{\sum_{i=0}^{N-1} V_{meas}[n-i]}{N}$$

Where V_{sys}ends up being the filtered voltage value at time n, and $V_{meas}$is the voltage value read from the PDB sensor via a function call in software. N's value can be determined by the length of averaging needed and the sample time. If it is desired to average 2 seconds of data, and the sample time is 10 ms, you will need 200 points ($N = 2/Ts = 200$).

Use averaging filters on both the read-in system current and voltage. It may also be used on the ESR calculated.

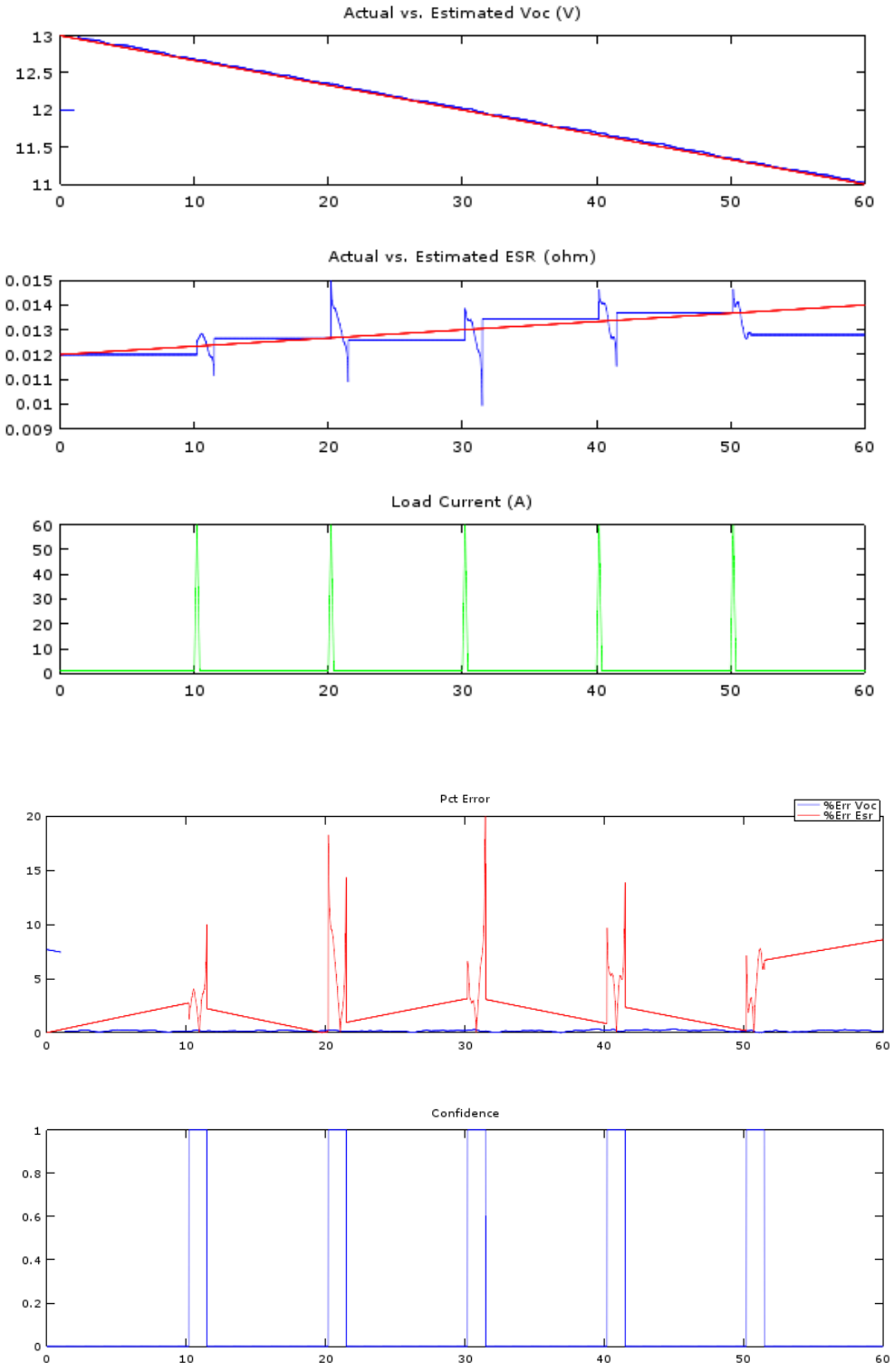

Figure 6: Performance of Algorithm on a pulsed-current-draw waveform with a discharging battery

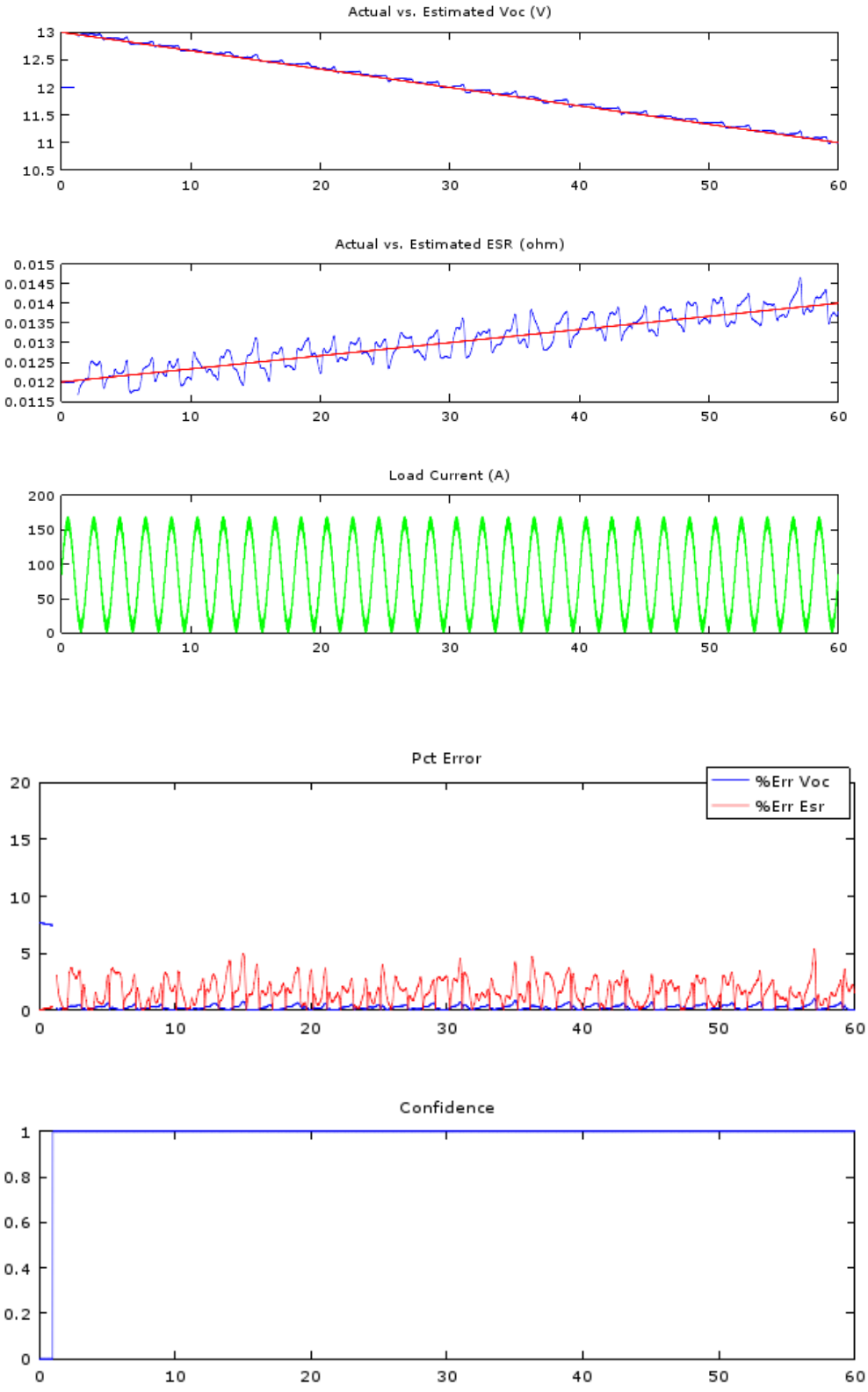

Figure 7: Performance of Algorithm on a sine waveform current draw with a discharging battery

14.1.2 R_{bat}Calculation using Least Mean Squares on a Set of Points

From http://faculty.cs.niu.edu/~hutchins/csci230/best-fit.htm.

The algorithm used to calculate $R_{bat}$from a window of current and voltage measurement points is based on the "Least-Mean-Squares" algorithm. LMS seeks to define a line with a slope such that the distance from the line to any point in the set is as small as possible. That is to say, the line is the closest to the average of all the points as much as possible. The algorithm is derived by first defining this "best fit" condition mathematically, and then working backward to the line equation. That proof is not covered in this paper. Instead, the final algorithm is presented.

Define first $N_{lms}$to be the number of points in the window of I-V readings to consider. This is referred to as the "window size", and is numerically the same as the number of current-voltage pairs plotted on the I-V graph from which we will attempt to apply a best-fit line.

Define a few more variables:

$$S_v[n] = \sum_{i=0}^{N_{lms}-1} V_{sys}[n-i]$$

$$S_I[n] = \sum_{i=0}^{N_{lms}-1} I_{sys}[n-i]$$

$$S_{IV}[n] = \sum_{i=0}^{N_{lms}-1} (I_{sys}[n-i] * V_{sys}[n-i])$$

$$S_{I^2}[n] = \sum_{i=0}^{N_{lms}-1} \left(I_{sys}[n-i]^2\right)$$

$$\bar{V}[n] = \frac{S_V}{N_{lms}}$$

$$\bar{I}[n] = \frac{S_I}{N_{lms}}$$

Then, using these variables, the ESR estimate for timestep n may be calculated:

$$R_{bat}[n] = -\frac{S_{IV}[n] - \left(S_I[n] * \bar{V}[n]\right)}{S_{I^2}[n] - \left(S_I[n] * \bar{I}[n]\right)}$$

14.1.3 Confidence Calculation and Filtering

As stated, the confidence we have in the estimated $R_{bat}[n]$ is based off of the standard deviation of the set of current measurements within the window being considered. For reference, this standard deviation (or "spread") is:

$$\sigma_{conf}[n] = \sqrt{\frac{1}{N_{lms}} \sum_{i=0}^{N_{lms}-1} \left(\left(I_{sys}[n-i] - \bar{I}[n]\right)^2\right)}$$

A tunable constant σ_{min}determines the minimum spread needed before the R_{bat}estimation is trusted. If $\sigma_{conf}[n]$ is larger than the minimum, use $R_{bat}[n]$. Otherwise, re-assign $R_{bat}[n]$ to the value during the highest-$\sigma_{conf}[n]$ loop during the most recent period of trustable-R_{bat}numbers. A snippet of GNU Octave code is provided illustrating this logic:

```
% If the spread is above a tuned minimum threshold, we may use this window
% for the ESR calculation.
if(spread > min_spread_thresh_A)
        confidence(i) = 1;
        % Additionally, if this is the larges spreaad we've seen so far,
        % save the ESR value for when the spread is no longer
        % big enough to trust the calculation.
        if(spread > prev_best_spread)
                prev_best_spread = spread;
                prev_best_esr = ESR_est_raw(i);
        end

else
        %When the spread isn't big enough, we don't trust our calculation
        % We also reset the "best spread" to zero since we are no longer in a
        % "confidence = 1" region.
        confidence(i) = 0;
        prev_best_spread = 0;
end

%If we didnt' trust this window's calculation, use the previous best calculation
if(confidence(i) == 0)
        ESR_est_raw(i) = prev_best_esr;
end
```

14.1.4 V_{oc} Calculation

Regardless of whether the R_{bat} calculation was trusted or not, the open-circuit voltage can always be calculated via this formula:

$$V_{oc}[n] = \bar{V}[n] + R_{bat}[n] * \bar{I}[n]$$

15 Usage of Estimated Parameters

The two battery parameters have just been estimated from measured system voltage and current values. This calculation should be done as part of the first step of the limiting algorithm. The determined numbers for the open-circuit voltage and internal resistance of the battery may then be used in the other calculations.

Part IV
Implementation Details

In this section, we will cover a handful of the implementation details that were discovered while putting this current limiting algorithm onto FRC Team 1736's 2016 robot.

We had a shifter drivetrain with two possible ratios between wheel speed and motor speed. Since the encoders measured speed at the output of the gearboxes, we had to add logic to account for the fact the motor speed to encoder speed ratio (K_{enc_ratio}) changed.

The equations presented above are in a verbose form to demonstrate their physical meaning. Simplifying them is very possible to reduce the number of computations done at run-time.

Additional voltage drops and resistances may need to be added to model the losses in the wires and motor controllers. For example, we found a fresh battery, including wiring, connectors, and circuit breaker, had closer to 0.025 ohms of series resistance. Or motors also exhibited an extra 0.051 ohms of resistance in wiring and connectors which we accounted for.

Filtering the input voltage and current properly for battery estimation was fairly tricky (especially to avoid time-delay). A filter length of 5 (at 20ms sample rate) provided a good trade-off between delay and noise rejection. Post-filtering the ESR also helped make the reading more accurate - the post-filter operation was done with a 20-point averaging filter (again at a 20ms rate).

It has been mentioned that the algorithm for estimating battery parameters should sample-and-hold the R_{bat} when the spread of drawn current is too small. We experimentally found that sampling new R_{bat} only when the standard deviation of the measured current went above 7A provided good noise rejection.

Most fully-charged batteries start at a voltage above 12 volts. We found that an initial guess of 13V for the open-circuit voltage was better than 12.

It is highly recommended to keep a log of pertinent internal information. Included in this is the measured system voltage and current, estimated system voltage and drivetrain currents, the estimated battery parameters, the utilized scaling factor γ, as well as the driver-demanded and commanded voltages. These logs can be used to validate proper software behavior, and diagnose erratic motions. Additionally, any times when limiting was applied can be used to show why exactly the algorithm was needed.

A complete solution for scaling factor γ was provided in this paper. However, on the actual robot, we opted instead to utilize an iterative solver. When the estimated system voltage for certain driver commands was sensed to be too low, the solver would iterate over smaller and smaller values of γ until a suitable one was found.

Finally, it is to be noted that this algorithm will necessarily reduce the apparent responsiveness of the drivetrain. Hopefully the 6-CIM drive is powerful enough that the reduction from this algorithm will not be missed. However, it is ideal to have it in place well before the drive team begins practice, as adding it partway in may be perceived as an undesired limitation. In our implementation experience, the downgrade in performance was negligible to the drivability of the robot.

Part V
Results

16 Experimental

After the 2016 season, we evaluated some of the properties of the robot. We never experienced a brownout during any match or during normal practice. Additionally, we ran a series of experiments to determine the full effectiveness of the system on a real robot.

The most telling of these experiments was run as follows:

1. Put a fully-charged, older (˜2014) battery into the robot
2. Disable the motor command limiter
3. Run the robot hard until brownouts start to occur (about a minute and a half)
   (a) "Hard" means forward/reverse cycles at full speed
4. Stop and enable the current limiting algorithm
5. Observe brownouts cease for at least 30 seconds

Figure 8 shows our results from this experiment. System voltage remains above a threshold as soon as limiting is applied, and brownouts cease.

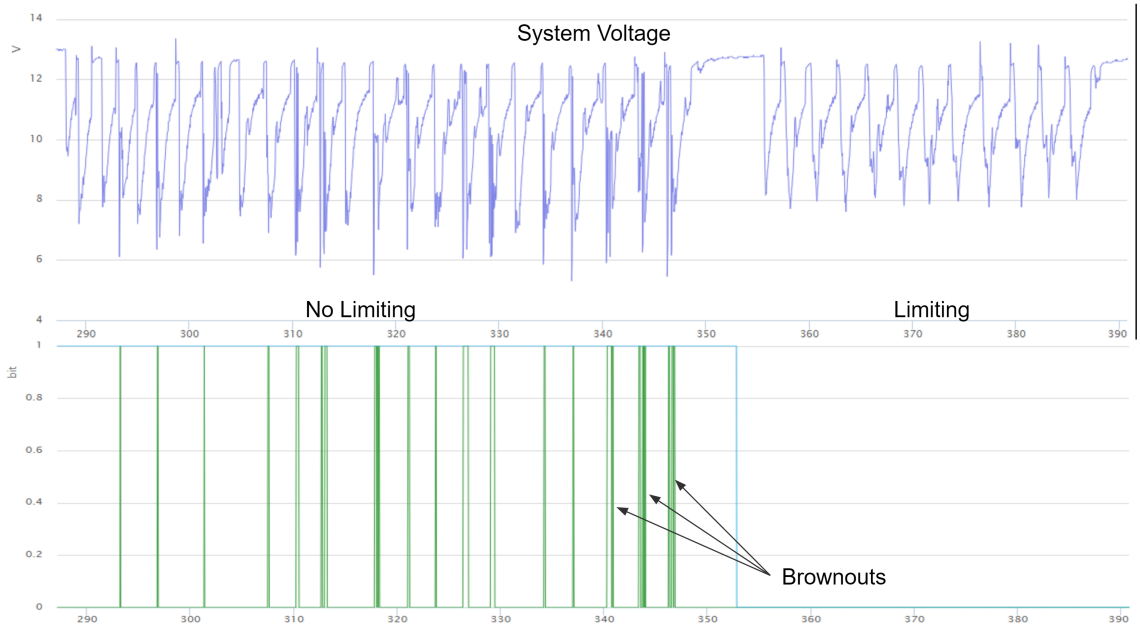

Figure 8: Results of running the limiting algorithm ($V_{sys,min} = 7.5V$)

17 Conclusion

We have shown an algorithm which limits motor commands based around a minimum system voltage criteria, utilizing physical models of batteries and motors. It provides a "next-level" approach to solving brownout issues which can hinder FRC robot performance. The reader should note that many simpler methods exist, and simpler methods will suffice in many cases. However, at a minimum, this method drives a deeper understanding of how the electronic components of the robot interact.