

**Fundamentals of Electric Vehicles: Technology and Economics**  
**Professor L. Kannan**  
**Professor of Practice**  
**Indian Institute of Technology, Madras**  
**Lecture - 56**  
**Field-Oriented Control**

(Refer Slide Time: 00:17)



## 6.6 Field-oriented Control (FOC)

ADC, Signal Conditioning, Low-pass filter, PI control, Protection

6.6


Fundamentals of Electric Vehicles: Technology & economics

57

We will now apply whatever we learned so far to actually control the motor. We are now getting out of the motor per se, and getting into the controller which will control the motor according all the physics about the motor that we have learned so far.

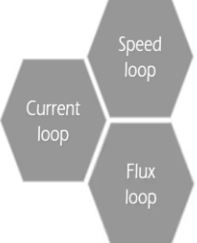
And the best possible control, there are different methods of control that are used different context. The best one that we would recommend for an electric vehicle is what is called field-oriented control.

(Refer Slide Time: 00:52)



## What do we really control?

- ? What is in our control?
  - The voltage applied to the motor: amplitude, frequency, phase
- ? What do we want to control?
  - The speed (from Hall sensor, encoder, resolver...)
  - The torque component  $I_q$  (inferred from the measured current)
  - Flux component  $I_d$  (inferred from speed & current)



6.6 Fundamentals of Electric Vehicles: Technology & economics 58

And before getting into the details of it, let us understand what is it, what is it that we want to control, what is in our control, what are we trying to control. This word control is used in a very ambiguous way, vague way.

See, all that we can do to the motor, supposing there is a motor lying on my table, all that I can do is I can apply a voltage to it. There is really nothing else under my control. If I apply a voltage it will draw the current that it wants, it will rotate in the way it wants, all depending upon its internal physics. When I say it wants, depending on its internal design, construction, physics, and everything.

It will deliver the kind of torque that is capable of delivering. There is nothing else I can do to it other than change the voltage that is being applied. So that is the only thing but it is an AC voltage so we can, when I say I am applying a voltage, I can decide what is the amplitude of the voltage, what is the frequency of the voltage, what is the phase of the voltage. All of these is in my control. This is the, this is all there is in my control, nothing else.

But what do we want to control? We are not interested in controlling the voltage itself, it is only a means. What we want to control is the speed. How fast is it going? I want it to go at 2800 rpm that is what I want to control.

And whenever we want to control something, first we should be able to measure it. We will come to why in the next slide but anything that I want to control, I must be also able to measure it to know there is under control, otherwise, I do not know if it is under control.

And speed can be measured through any number of devices. Very commonly, we use hall sensors, we can also use encoder, we can use something called resolvers, these are all different instruments that will tell us what is the speed.

The other thing that we want to control is the torque. We want to control the speed, we want to control the torque. And torque we know is related to the current. So if I can measure the current, since I have already measured the parameters of the motor like back EMF constant and other things, I can do my arithmetic and say, if this is a current, this is the torque. Fine? So I can measure the torque indirectly by measuring the current.

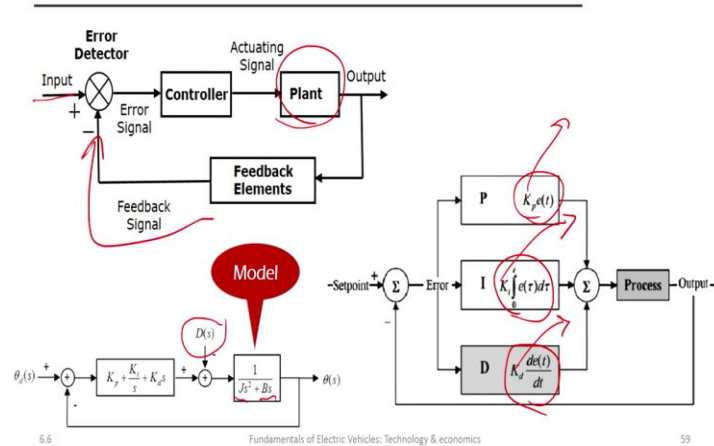
And then, when we are doing flux weakening, we want to measure the  $I_d$  because we want to increase that. And an indirect way of measuring that will be inferring it from speed and the current using the voltage equations.

So if I know the basic parameters like what is the resistance, what is the  $L_d$  and  $L_q$ , and what is the back EMF constant, if I know these things then I can infer and I can back-calculate what current should be going in. And for that current to go in what voltage I should apply. And then, by measuring the parameters that I am interested in, I can see whether my calculation is correct or is there any deviation. And normally, there will be a deviation.

I will apply a voltage thinking that it will make the motor run at 2800 rpm but when I actually measure, I may find it is running at only 2650 rpm, it is 150 rpm short. Then what should I do? I should slightly increase the voltage. So by measuring what I want to control and comparing it with my desired value, the difference between them I will call it as error and I will compensate that error by changing whatever is in my control, which is the voltage.

(Refer Slide Time: 05:26)

## Conventional closed-loop control



So this is called closed-loop control. The commonest way it is done is what I am describing. Again, there are different approaches that different people take but the most common way of achieving closed-loop control is what I am discussing here.

The most generic name given to whatever the thing that we want to control; I can say it is a vehicle, it is a motor, it is a chemical plant where some chemicals are flowing through pipe whatever. Whatever is we want to control is generically called as the plant or the process, any name one can give.

Now, I am giving an input. Let us say there is an operator. For the moment, we will think that the controller is a human operator with a knob in front of him and I am telling him that I want the motor to run at 2800 rpm. So he turns the knob and sets it to a certain level, which is the voltage, and says, okay, this should give 2800 rpm.

And then what happens, I have got some sensors. They are giving me a feedback that the rpm is only 2650. So when I subtract 2650 from the set value of 2800, the difference, the plus, and minus is the way of doing difference, then next input that goes to the operator who is called the controller is 150. What it means is increase the speed by 150 more that is the error.

Now, based on that information he has to increase the knob to the extent of 150. How does 150, the error, translate into the input change? Should I turn it by 10 degrees, 12 degrees, or 8 degrees, how much more should I turn is the decision that the controller has to make.

So there are three ways in which the controller will interpret the error. One is if I say that the error is 150 then I will turn it by an angle which is proportional to 150. The other thing I will do is okay, right now, the error is 150 but 1 second ago what was the error; one more second ago what was the error. If I add up all the historical errors what is it? And proportional to that historical error I will add another term. The controller will add another angle which is proportional to sum of all the errors that happens in the history.

And the third thing that the controller will try to analyze is, okay, the error is now 150 but what is the trend of this error, is this error increasing or is it decreasing. The rate of change of the error. And based on the rate at which the error is changing, it will sort of anticipate that the future error is going to be like, this let me compensate for that also now itself.

So there are three terms of error. One is simply proportional to the error, another is proportional to the integral of the error. And another is proportional to the rate of change of the error. So whatever we are measuring here as the error is translated into three different terms. And based on some sort of a judgment, the controller will assign different weightages to the three errors.

The weightage it gives are called the loop gains.  $K_p$  is the weightage that is being accorded to the term which is proportional to the error.  $K_i$  is the weightage for the integral of all the errors and  $K_d$  is the weightage given to the rate at which the error is changing. And after assigning these weightages when I add up everything, I will get some value 17.3. So I will turn the knob by 17.3 degrees in response to the statement that the error is 150.

So what is very critical is that the weightages must be correct. How do we know what weightage is correct? If we change the weightage, this angle will change and the entire system may become unstable; from 2650, the rpm will shoot up to 3000, and then I will try to decrease it, it will come back to 100 and it will go wild. That is called unstable behavior.

So assigning the correct weights, the correct value of  $K_p$ ,  $K_i$ , and  $K_d$  is very important and the way it is done is by understanding how the entire system responds. To the extent we understand the inner dynamics of the motor itself, the inner dynamics of the any mechanical system will be broadly depended on an inertia term, which is given here is  $J$ , and a friction term given by  $B$ .

And I can create a rough model of how this thing is. And I can have  $K_p$ , which is proportional to the error, the  $K_i$  is proportional to the integral error; in the Laplace transform, it will become 1

by  $s$ . And the  $K_d$  is proportional to the differential of the error; in the Laplace transform, it will become multiplied by  $s$ . And I can create a model like this and solve it using some tool like MATLAB or something to arrive at what is the optimum value of  $K_p$ ,  $K_i$ , and  $K_d$ .

Or I can just do it by trial and error in experiments. I will do a number experiments by using different combinations of  $K_p$ ,  $K_i$ , and  $K_d$  and see which one is working. If I do it like that, experimentally, that is called tuning; it is called PID tuning. Or I could solve. Again, solving assumes that I know the value of  $J$ , value of  $B$  and most importantly, there is another term here called  $D$  which is disturbances.


The disturbances will be force coming from the wind, the blowing from the side-wards direction, suddenly there is an up and down in the road, all of those are disturbances. So it is not possible to perfectly model everything. But to the extent we understand it, we can model it and we can come to a  $K_p$ ,  $K_i$ ,  $K_d$  combination which is optimum. And the job of the  $K_p$ ,  $K_i$ ,  $K_d$  terms is because we do not understand the physical model perfectly, there will be a deviation. And whenever there is a deviation, correct it; course correct it.

So this is the way in which we do control. So you understand now that to be able to control anything, I must measure it. If I do not measure it, I cannot get a feedback, therefore, I do not know if there is an error or not.

(Refer Slide Time: 13:26)



## Hygiene items

ADC	SOFTWARE FILTER (LOW-PASS)	SIGNAL CONDITIONING
<ul style="list-style-type: none"> <li>Measured values are always analog</li> <li>These have to be converted into digital values for further storage and processing</li> <li>For example: a voltage signal may be proxy for speed. This value from the instrument is converted into DC</li> </ul> $V_{\text{analog}} \rightarrow V_{\text{digital}}$	<ul style="list-style-type: none"> <li>Often, the measurement picks up noise – typically of much higher frequency</li> <li>This can be removed using a simple software filter</li> </ul> 	<ul style="list-style-type: none"> <li>The digital values are converted into meaningful values that represent entities of interest</li> <li>For example: The DC voltage has to be 'interpreted' to represent the number that denotes speed</li> </ul> $(V_{\text{digital}} - 834) * 2.3 = \text{rpm}$

So this is as far as control is concerned. Before we implement control in software, as I said we have to measure. And whatever we measure will normally come to us as an analog value, there will be a certain voltage coming from a wire from the sensor.

But for me to manipulate it in my processor, I have to convert it into a digital value. And the device that is used is called an ADC, Analog to Digital Converter. So it will take any analog value and give it out as a digital value of certain resolution. So this is the first step.

So I get a voltage which is proportional to the speed from the speed sensor and the ADC will take that voltage and convert it into a digital number. The other thing that often happens is that whatever the sensor is, it is measuring something but there are a lot of other disturbances in the environment because of which it also picks up noise.

So what it gives me is the measurement which is also mixed up with a lot of noise. The characteristic feature of this noise is that the noise keeps going up and down very frequently. It is of high frequency compared to the rate at which the signal itself will change. What is measured, like speed, will change gradually but the noise in the speed will keep going up and down very rapidly. So I can remove that noise by using what is called a software filter.

What is the software filter, we will see in the next slide. It will actually be an assignment for you. But you see that the raw signal coming from the sensor as what is shown here as the blue lines, every instant that it is giving me, it is giving a different value. So the actual signal is around 1500 but the value I am getting is maybe plus or minus 100. Within that range, it is just fluctuating wildly.

And then the speed is being increased to 2000 rpm. All along the way, the noise is persisting and after I reach 2000 also, the noise is persisting. And in fact, the noise is not even dependant on the level. Whether I am at a lower speed or at a higher speed, noise is roughly the same. And when I apply a filter, what I get is the brown line, and that brown line is very clean and smooth. And that is what I will now take for all my further processing in the controller.

And finally, whatever value I get is just a number and I have to interpret it or translate it to represent the entity that I am interested in. For example, I get a voltage which has a relationship with rpm but the number itself is not equal to rpm. So I may have to do some manipulation of it to convert it into rpm.

So usually it involves a scale shifting, level shifting, and then a scaling. A combination of these two things will give me the value that I am interested in. So having done all of this, now I have the measured value in my memory location in the processor which I can start using for the purposes of control.

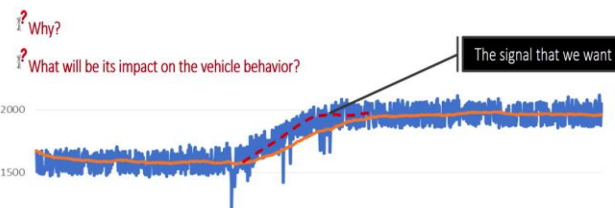
(Refer Slide Time: 17:21)



## Assignment 6.6.1

The low-pass software filter on the throttle signal works like below: Every 'new' signal is damped (to say, 10% of its value) and added to the previous value with a complementary weight (90%). When we compare the raw signal and the filtered signal, we notice two things:

- o The filtered signal is smooth. This is what we like!
- o But when the signal changes, the filtered signal changes with a delay.



66

FUNDAMENTALS OF ELECTRIC VEHICLES: TECHNOLOGY & ECONOMICS

61

So here is an assignment about what we just now discussed. So I have a throttle. When I turn it, a voltage proportional to how much I am turning goes to the controller. And let us say, the extent by which I am turning is a proxy for the speed at which I want to go.

If I have slightly turned it, it means I want to go at 20, 20 kilometers per hour. If I am turning it more, it means I want to go at 35 kilometers per hour. If I really turn it high, it means I am trying to go at 60 kilometers per hour. So depending upon how fast I want to go I am turning it.

This message has to go to the controller. It goes through a sensor which may simply be a potentiometer, which is then giving a voltage proportional to how much I am turning. But because the wire is long and it is coming in contact with many things along the way, it also picks up a lot of noise which is what we saw in the previous slide. And then we do a software filter. After converting it to a DC using the ADC, we filter out the noise and we find that the filtered signal is very smooth.



The normal way in which this filtering is done is that supposing, I am taking a reading every millisecond, every successive instant of time is 1 millisecond apart, and I have a value like let us say 1500 now. The next millisecond it has become 1580 because of the noise.

Instead of taking its value to be 1580, I will take only one-tenth of that value, 1580 into 10 percent, which is like 158. And the value I already have which is 1500, I will give it a complementary weight of 90 percent. So 1500 into 90 percent is 1350 plus 158, what does it add up to, 1350 plus 158, 1508.

So instead of 1500 jumping up to 1580, the next value has only jumped up by 8. I have suppressed the noise component which was 80 to 8; 90 percent suppression of the noise has happened. If I want to even more aggressively, I can say that I can only give 1 percent weight to the new reading and 99 percent weight to the older reading. And I will keep on doing it every instant. So all the fluctuation will get damped out.

This is how the software is implemented. And it is easy to implement in software because you are not making any extra wires and resistances or anything and you can just do it by computation. And what you like is the fact that it is smooth.

But what I want you to notice is that when the signal changes from about 1700 or something, it is going to 2000 and there is a transition that is happening. When the transition is happening, the filtered signal is responding with a delay. It should have responded like the red line because that is the true way in which the transition happened. But the filtering has a downside which is that it causes a delay. Why does this delay happen and what will be its effect on the vehicle behavior, this is what I want you to figure out.

(Refer Slide Time: 21:33)



## Protection Boundaries

### Over-current

Excess current leads to excess heat generation and thus equipment damage and other risks.

### Over-voltage

Being over-voltage can increase the current and in turn heat that drastically reduces the life of the equipment.

### Under-voltage

Being under-voltage could cause switching devices to enter unstable states, leading to burn-out.

### Temperature protection

Any abnormal temperatures have to be monitored in order to turn off the circuit before any components are damaged.

These boundaries are defined in software and in hardware



6.6

Fundamentals of Electric Vehicles: Technology & economics

62

And finally, before we get into the heart of what the controller does which in some sense, we have already covered in the previous lessons, the additional detail that I want to say is that no matter what happens, we want the device to be safe, the passengers to be safe, the vehicle to be safe. So we create a lot of fences around. That is also there in the controller to safeguard everything.

So typically there will be 4 kinds of ways these fences. One is protecting overcurrent; if for some reason there is a sudden surge of current, in a matter of a fraction of a second that excess current will cause localized heating of some component in just fraction of a second. Because these are very tiny components and that component will blow. And the consequence of that blowing is that something else will happen somewhere and also sort of things and go berserk. We do not know what will happen.

So we want to limit, we want to protect the entire thing against any runaway increase in current even before anything get damaged, the fact is the current going dangerously high we want take action and protect the device.

And other thing is overvoltage. Again, if the voltage want to suddenly rise unpredictably, it will actually lead to an overcurrent situation. And then there are also devices which have a specification of the voltage. Even if the no current is flowing, if the voltage goes above a certain

level, the component will break down. And then it will suddenly start conducting and that will then lead to an inrush of current.

The other thing that we have to monitor is under voltage. Under voltage, you may think is very safe nothing will happen, nothing can get damaged. But actually in all switching devices particularly, I say that if the voltage is less than 2 volts, it is off, if it is more than 7 volts, it is on.

But if the voltage is 4 or 5 volts, it is it is undefined state and you can never predict what will happen. Very peculiarly it will start conducting or not conducting and behave in a unpredictable manner. So under voltage can be very dangerous and it will lead to a very unpredictable behavior. That is another thing we have to monitor carefully.

And then, of course, we will have a few temperature sensors like earlier there was discussions about temperature sensors to monitor temperature in a battery. Inside the motor, inside the controller also we have temperature sensors. If the temperature is rising, we have to take corrective action to protect everything.

But remember that this temperature protection is not a very helpful thing for other failure conditions like this because when there is an over-current, the overall temperature in the place where I have kept the temperature sensors will not increase. But because high current is flowing through some track and going to some component and things like that, in those localized places there will be a sharp rise in the temperature which will not be detected by the temperature sensor.

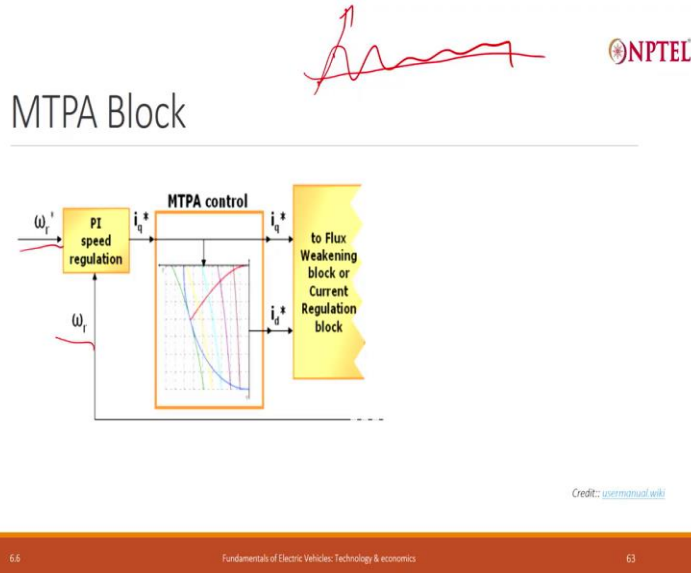
So having a temperature sensor does not mean that you do not need over-current protection and the other things. So that is the reason why all of these are there. Although the failures are finally all thermal, only certain thermal failure can be detected by the temperature sensor. For other things we use, we use the proxy of current or the voltage to identify failures modes.

And these fences, we can draw them in software and in hardware. If I do it in software, it is convenient because I can configure and flexibly change things. Even during the course of a ride, I will say, temporarily I will increase this limit there, et cetera, et cetera, I can do; it is flexible.

But with software, there could be some bug and in some condition, that boundary may fail. If that happens, we still want to have a final hardware check on the whole thing which is not

affected by any of the software misbehaviors and things like that. So these boundaries are applied both in the software and in the hardware.

(Refer Slide Time: 26:14)



So we have seen so far, what are things that we want to control which is speed, torque, and flux in the case of flux weakening; how we can measure them and using our earlier knowledge about the parameters of the motor, we can convert our desired parameter of torque and speed and flux into some voltage to be applied.

And what is the knowledge that we have used to do that? First is what we have already learnt about MTPA, maximum torque per ampere. That is implemented in a block of code called the MTPA block. There is a set speed and there is a measured speed. Based on the difference between the two, I know what is the error in speed.

There is a PI controller. Again, another point I forgot to mention. I was talking about PID but normally, we do not use D, can anybody tell me why? We almost never use the D, we just set the  $K_d$  to be 0. It is only  $K_p$  and  $K_i$  that we use. The answer is related to the noise problem that we saw the signals.

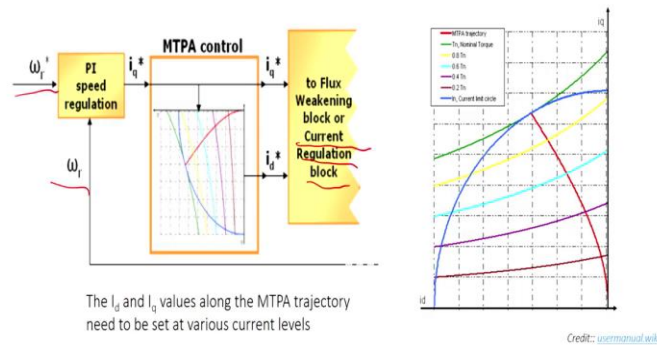
If the signal is moving up and down like this then during any two successive instances, if I have a signal like this then between here and here, actually the true signal is only like this but because of this noise, I will think that the error is going up like this. And if this is 1-millisecond interval

and I extrapolate it to 1 second, I will say next second the error is so big. So I will violently overcorrect.

So unless I have perfectly noise-free environment, it is very dangerous to use the D term and extrapolate from the D term. So always we avoid using the D term, we only use P and I. So there is a PI regulation. Let me erase this.

(Refer Slide Time: 28:30)

### MTPA Block

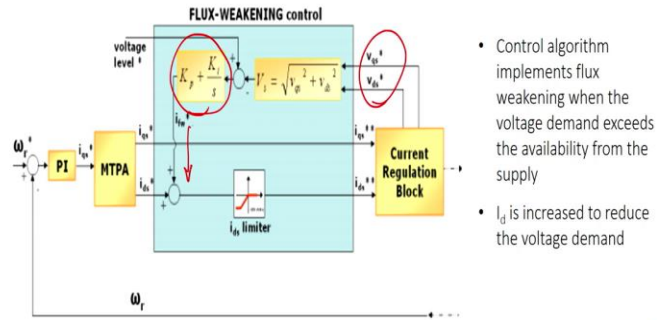


And based on the PI regulation I know what is the current that is needed to supply the torque. And then I have MTPA algorithm which tells me that if you want maximum torque for a given value of current, then you divide the current into two components; one is  $I_q$  and one is  $I_d$ . So based on that algorithm which is like a lookup table, using that lookup table, I will add an  $I_d$  component so that I get maximum possible torque for the given amount of current.

And then send it to another block which is here written as the current regulation block. All that this does is it converts the current into voltage. Whatever we saw in the voltage discussion earlier, it will use the  $I_d$  and  $I_q$  and all the parameters of the motor that are known and convert that into  $V_d$  and  $V_q$  and apply it on the motor except when I want to do flux weakening.

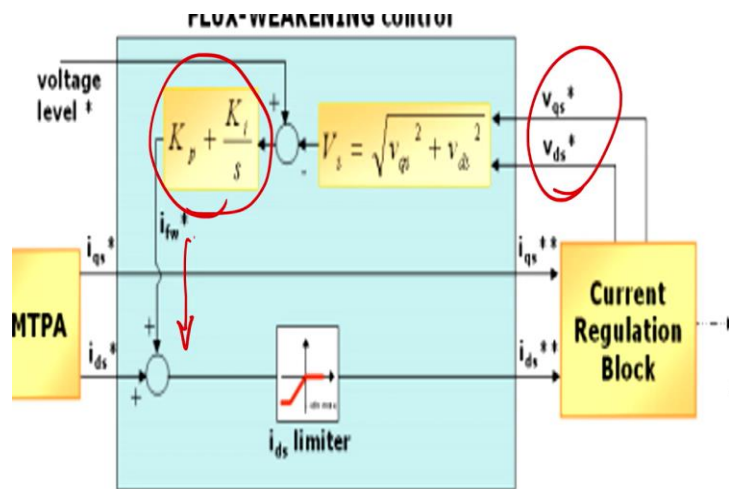
(Refer Slide Time: 29:51)

## Flux Weakening Block



- Control algorithm implements flux weakening when the voltage demand exceeds the availability from the supply
- $i_d^*$  is increased to reduce the voltage demand

Credit: yoramansari/wiki



(a)

If I am trying to run at a speed greater than the rated speed then instead of directly converting it into the voltages, I will go through an extra block called the flux weakening block. Then the voltages are calculated here. I told you the  $V_d$  and  $V_q$  are calculated in the current regulation block.

If that voltage is greater than what the battery can supply, we earlier discuss and the example that the limit is 34 volts, if the pair of  $I_d$  and  $I_q$  coming from the MTPA block results in a voltage which is greater than 34 volts then we enter the flux weakening loop you see that there is a condition.

If the sum of the square of  $V_d$  and  $V_q$  is greater than the limit then we enter the flux weakening loop. And the flux weakening loop has its only PI for the flux, another PI loop which is called the flux loop, and that will add an additional component called  $i_{fw}$  in this picture. I do not know if it is clear, I can zoom in if you want. So there is an additional component called  $i$  flux weakening which is added to the  $I_d$ .

So that the  $I_d$  that is coming from the MTPA block is augmented with an additional component of  $I_d$  which is meant to further weaken the flux and reduce the back EMF. And then that pair goes into the current regulation block. It will again recalculate the voltage to see whether it is below the desired limit. If it is, then it will supply that voltage to the motor. And if it is not, it will again be going in this loop till comes down.

And these loops are executed in microseconds. So fraction of a millisecond, so very rapidly it will do number of iterations and bring the voltage down to an appropriate level by adding sufficient  $I_d$  before it is getting applied to the motor. So this increased  $I_d$  is to reduce the voltage demand by reducing the back EMF.

(Refer Slide Time: 32:22)



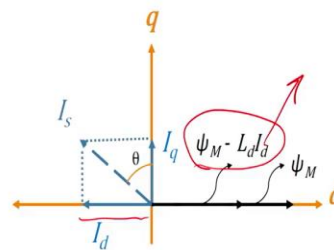
## Work around BEMF for high speed

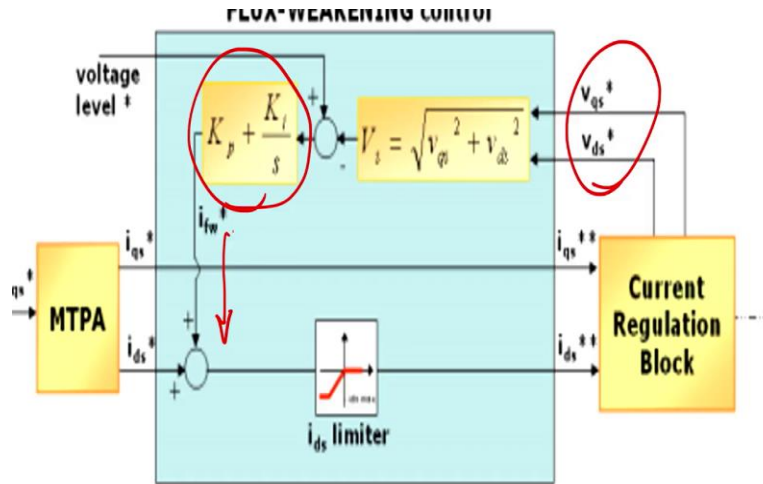
The problem with getting to high speeds is the back-EMF... What if we weaken it?

**How?**

- Send a strong current that opposes it ( $I_d$ )
- That would leave us with less available 'budget' for torque producing current, though – since the overall current limit cannot be exceeded

**Would that really extend the speed range?**

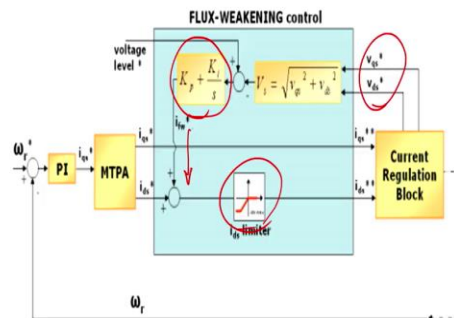




(a)



## Flux Weakening Block



Credit: [usermanual.wiki](#)

65

Fundamentals of Electric Vehicles: Technology & economics

64

This pretty much covers what the controller, as in, all the control logic does. If you have questions about this, please ask me. Yeah?

Student: Sir, actually the rotor has permanent magnet, right? So we cannot change, reduce the flux, or alter the flux.

Professor: So the question is the rotor has a permanent magnets, how can we reduce the flux? We are not reducing the flux of the magnet itself but we are adding additional flux in the opposite direction so that the resultant flux is lower. The magnet continues to give the same flux that it was giving before.



If I revert to a quickly take you through the  $\psi M$  is a fixed number, it is not change at all; I cannot change it. But the total flux on the d-axis is  $\psi M$  minus  $L_d I_d$ . And  $L_d$  also is fixed number, I cannot change it. Only thing that I can change is  $I_d$ . If I increased  $I_d$ ,  $L_d I_d$  will also become more, therefore,  $\psi M$  minus  $L_d I_d$ , which is the resultant flux which is actually being seen by the battery will come down. Fine?

I am only weakening the flux but I am not weakening the magnet itself. And now that you brought this up, I will also tell you another important but subtle point that I did not mention. You can see here that when I add an additional  $I_d$ , the total  $I_d$  will increase because it will be  $I_d$  plus the additional  $I$  due to flux weakening. The sum of the two, if it is become very large then the negative flux will become so much that it may permanently weaken the magnet.

If you recollect what we talked about negative magnetomotive force, magnetic intensity causing the flux to deplete where we discussed about the coercivity and all that. So there is a certain limit beyond which if I increased the  $I_d$  it will permanently weaken the magnet it would not recover. It will become a much weaker magnet then the future performance of the motor will be poorer, it will deliver less torque. So we set a limit

After I add the additional flux weakening thing, I tried to check whether it is exceeding the limit. That is another protection boundary that is there in the software and if it is within it, normally, it will be within it then I allow it. If it is so high, then I will not allow it.

It means I will not reach that speed. If I am trying to reach 20000 rpm then maybe, I will exceed this speed and that will not be allowed because that means a magnet will get permanently damaged.