

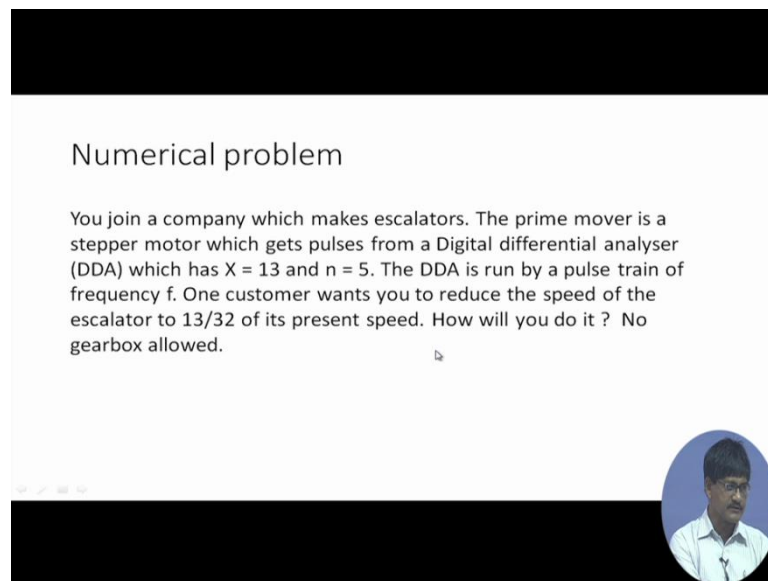
Computer Numerical Control of Machine Tools and Processes
Professor A Roy Choudhury
Department of Mechanical Engineering
Indian Institute of Technology Kharagpur
Lecture 14
Interpolators Curvilinear

Welcome to the 14th lecture on the online course “Computer control of machine tools and processes”. So here we are going to discuss about curvilinear interpolators which will include circular interpolators and also curved interpolators for that matter say if you want to move along a curve which is not a circle but definitely not a straight line. So it is much more generic approach to interpolation. So in case you have to do that, what are the “ifs” and buts and what are the things we have to be conscious about et cetera.

And before that I have also included a short discussion on some of the questions which I could not discuss in the previous lecture I will just read out those problems to you and you can do them yourself because they are extremely simple, they follow the logic of the discussion which have taken place and also the logic of the previous numerical problems which has been solved in the class. In the 15th lecture I will discuss questions and answers I mean multiple-choice questions and their answers from the last 4 lectures namely we have the last lectures that we have discussed.

They include G and M code programming; they include off-line computer off-line programming then linear interpolation of course and this particular topic of this lecture which is curvilinear interpolation. On all these 4 topics I will be discussing questions in the 15th lecture, so let us start.

(Refer Slide Time: 02:36)



Numerical problem

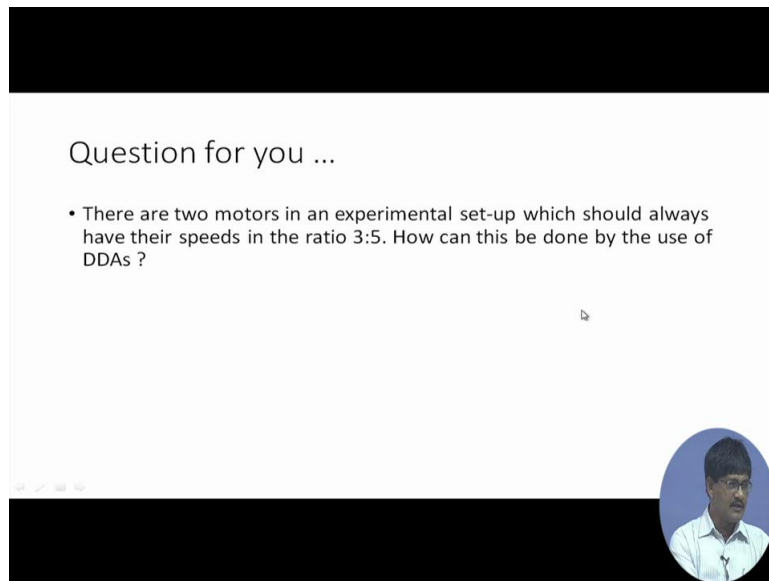
You join a company which makes escalators. The prime mover is a stepper motor which gets pulses from a Digital differential analyser (DDA) which has $X = 13$ and $n = 5$. The DDA is run by a pulse train of frequency f . One customer wants you to reduce the speed of the escalator to $13/32$ of its present speed. How will you do it? No gearbox allowed.

This was one of the problems which I wanted to discuss in the previous class, so I am going to leave this to you let me read out the question. You join a company which makes escalators; the escalators are being made where the prime mover is a stepper motor which gets pulses from a digital differential analyzer. So the stepper motor is receiving pulses from a DDA and the DDA has $X = 13$ and $n = 5$, X is the number which is put inside the counter of the DDA and the counter has 5 bits, so the maximum number it can contain is $2^5 - 1$.

So in this particular configuration it is sending pulses to the stepper motor at a definite rate and there is a pulse frequency which is input to the DDA that is F , the DDA is run by a pulse train of frequency F . One customer wants you to reduce the speed of the escalator to $13/32$ of its present speed, so the speed which it was establishing with $X = 13$ and $n = 5$, the customer wants you to further reduce it to $13/32$ of its present speed, how you do it? And no Gearboxes allowed, so you can give a solution like you can use a different DDA or you can try changing the value of X in this particular DDA or you can add yet another DDA, so let me give you the possible solution that you can provide.

Not all of them will work, but you can try one changing the number X in the present DDA, adding second DDA in series or replacing this DDA by yet another one, yet another DDA okay, please try out these 3 solutions and maybe in the next lecture I can I can design one question as an MCQ on this problem model.

(Refer Slide Time: 05:20)



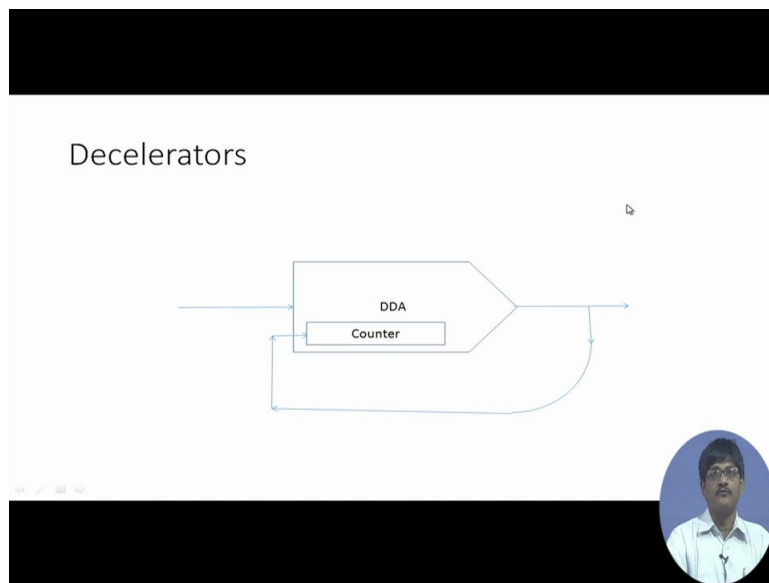
Question for you ...

- There are two motors in an experimental set-up which should always have their speeds in the ratio 3:5. How can this be done by the use of DDAs ?

Next one, there are 2 motors in an experimental setup which should always have their speeds the ratio 3:5, how can this be done by the use of DDAs? So think of it as you are having a setup of some experiment in which there are 2 motors, their rotations per minute they have a definite ratio. If you establish their rotational ratio with the help of gearbox okay if you establish the rotational ratio with the help of gearbox it is something fixed, so that you want a change in the gearbox ratio, you might have to go to the market, buy another yet another of those Gearboxes which are quite expensive and remove the previous one and put the present one in.

The problem is, such changes are very expensive and therefore in the problem we are suggesting, put another DDA sorry make use of DDAs to solve the problem so that if you require a different ratio, either in those DDAs you can make some change or you can remove these DDAs and put some other one to serve your purpose, ok so this is the problem. And if I if I can get the time, I will frame 1 MCQ to fit this particular type of problem, so with this... Last of all, I have something else to discuss also.

(Refer Slide Time: 07:11)

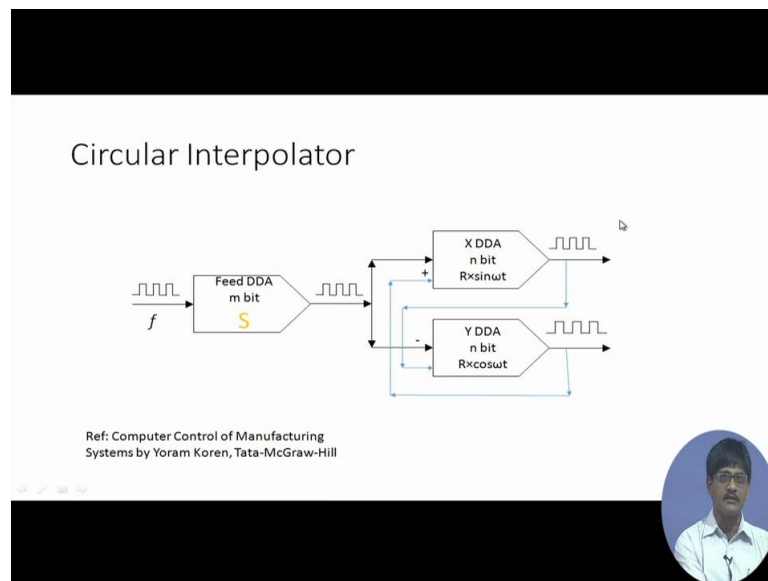


That is, from the basic DDA that we have discussed there is always a possibility to make decelerators with this sort of configuration, what is this configuration? This configuration shows that there is an input frequency coming into the DDA and the DDA is sending an output frequency corresponding to some number which has been kept in that counter okay; the number inside the counter gets added to itself due to each and every pulse coming in at the input frequency point. So in that case, if we want the rate of output pulses to automatically go down in time say exponentially.

In that case there is a provision that parallel to the output going outside to serve whatever purpose it is supposed to, parallel it can also be fed back to the input block point of the counter, in what way? Just like we have down counters, these counters inside the DDA they also have up counting point and their down counting point. At this moment it is shown to be connected at the down counting point so that each and every pulse which is coming out as an output from the DDA, that is also fed into the counter and eventually what it will do is, in this configuration we have meant it to be down counting or decrementing the value X inside the counter.

So if X goes down, ultimately the frequency of output that will also go down and it can be shown very easily that is sort of configuration will result in exponential deceleration. If we have exponential deceleration, we can achieve a number of things with that sort of a set, all right. So let us complete our discussion on circular interpolators.

(Refer Slide Time: 09:26)



Circular interpolators, they have they are somewhat different from the linear interpolators, this is their configuration. For example, we have seen that pulse rate is coming at the system at the input to the feed DDA and this was the basic configuration remains the same whatever was there for the linear interpolator, same thing is there for circular interpolators, whatever was coming in as at the system clock I mean from the system clock whatever frequency of pulses that are coming in, same pulses they are coming in here as well, so what is different?

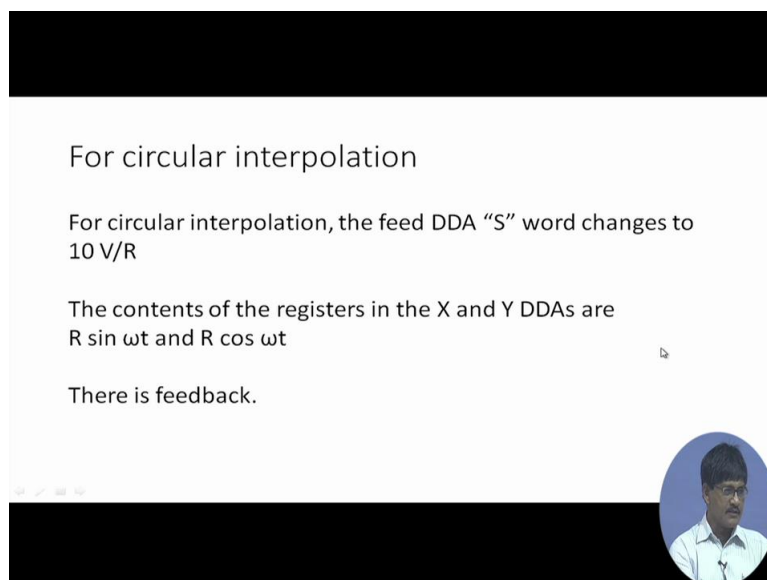
In the feed DDA previously the S word which was put inside that has not changed, what is that? That is now equal to $10 V$ by R , where R is the radius of the arc being cut if they are having an arc of a circle, it has a particular radius. If we are interpolating along that arc, in that case the radius of that arc comes into the picture and we have S word equal to $10 V$ by R in in case of circular interpolation. So S word is different, what else is different, the pulses which are output from the feed DDA and the ones which are going to X DDA and Y DDA.

The X DDA and Y DDA counter inputs where previously Delta X and Delta Y and therefore, the frequency of pulses they were proportional to the incremental distances to be covered so that we have velocity of the motors running due to these pulse trains, velocities are proportional to incremental distances and we had velocity triangle similar to the displacement triangle. In this case however, we are replacing Delta X and Delta Y by $R \Omega \sin \Omega T$, where R is the radius of the arc being cut, Ω is the angular velocity. You know, we are doing circular interpolation at constant angular velocity and therefore, $R \Omega \sin \Omega T$ replaces Delta X and $R \Omega \cos \Omega T$ replaces Delta Y.

Okay, I mean a lot of combinations are there but this is a typical combination and in addition to that, we are having feedback from the output pulses so that $R \omega \sin \omega T$ and $R \cos \omega T$ get updated time so that the rates of the pulse output from the X and Y, they also get updated in time. There if you have the same rate of pulses coming out and at X and Y, you are simply going to cut a straight line. But if you can change them sinusoidal, then you can get a circle so the pulse frequency becomes very high goes down gradually and comes to its lowest value and that way it goes on fluctuating along X and Y where circle is being cut okay.

While X becomes high Y becomes low, Y becomes high X becomes low, this is the typical combination that we have for these 2. This is achieved by the feedback which has been shown and the type of feedback is exactly the one which was shown in case of decelerator, okay. The counters are updated or updated with the pulse rates coming out from the velocity, but why so, what is the logic behind this thing, why are we doing this? These things can they can all be derived, it is already present in the reference which is mentioned at the bottom, so we are not going into the elaborate derivation of these expressions but we can work well this particular basic knowledge of these basic changes from the linear interpolator.

(Refer Slide Time: 14:04)



For circular interpolation

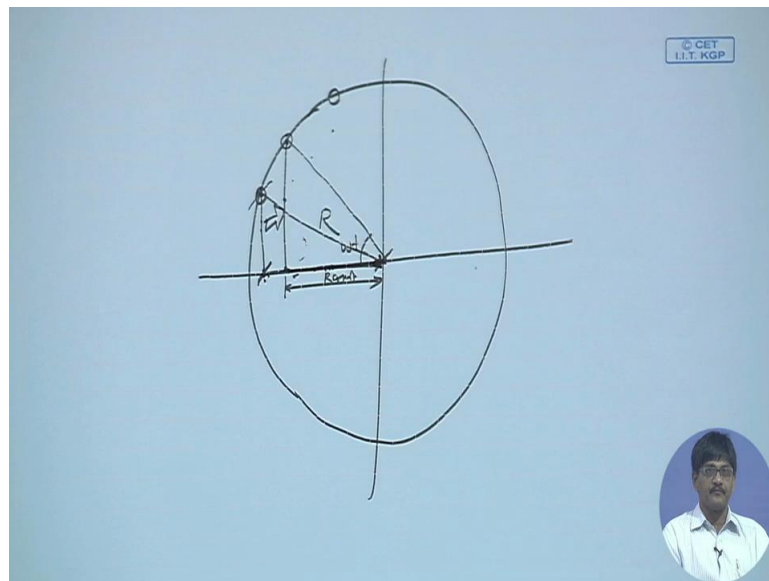
For circular interpolation, the feed DDA "S" word changes to $10 V/R$

The contents of the registers in the X and Y DDAs are $R \sin \omega t$ and $R \cos \omega t$

There is feedback.

So we have just summed up the changes that we discussed just now, 'S' word changes to V by R instead of V by L, 10 is just a constant multiplied to it, it is not affected. Contents of the registers in the X and Y DDAs are now $R \sin \omega T$ and $R \cos \omega T$ and they get updated in time and there is also feedback. Feedback is to change the values of $R \sin \omega T$ and $R \cos \omega T$. So $R \sin \omega T$, if you have a look at this particular figure.

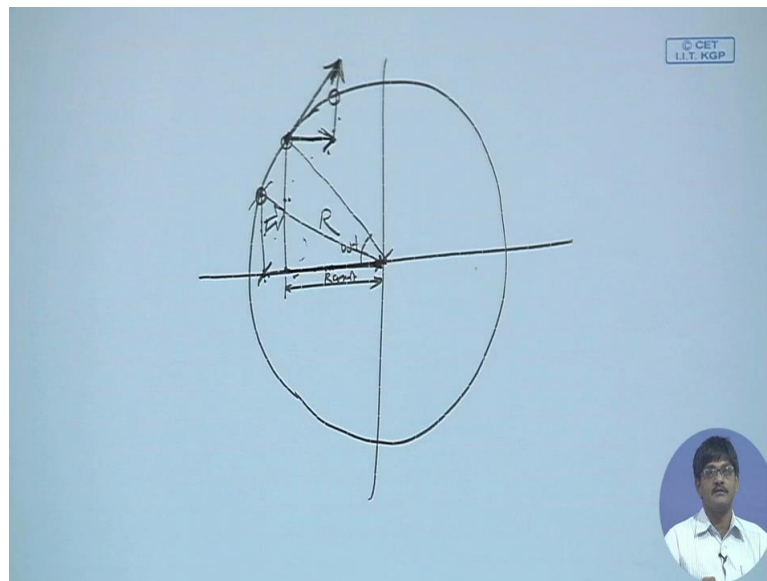
(Refer Slide Time: 14:53)



If you are starting from this point say and you are trying to reach this particular point, in that this is R and this is $\sin \Omega T$ and $R \cos \Omega T$. So these values okay in basic length units they are fed into the interpolator and they get updated in time which means that suppose you have reached this particular point therefore, this is the present value of $R \sin \Omega T$, this is the sorry ΩT . This is the present value of $R \cos \Omega T$. So while $R \sin \Omega T$ has increased, $R \cos \Omega T$ has decreased, from this value it has come to this smaller value $R \cos \Omega T$. So this may be fine that if you are cutting along a 2nd quadrant, $R \sin \Omega T$ goes on increasing while $R \cos \Omega T$ goes on decreasing.

So the same thing might not occur in this segment or this segment and while you are moving, say here we are moving counterclockwise while you are moving sorry, we are moving clockwise, so while you are moving in some other segment and maybe moving counterclockwise, you might not always get this same combination okay. But this is what happens, the number inside first we put this number and this number in basic length unit and they constantly get updated in time, so this number becomes larger and larger, etc as you move towards the target and this number becomes, $R \cos \Omega T$ becomes smaller and smaller as you approach the target okay, so this way it works.

(Refer Slide Time: 17:02)



And at any point in time when this is the point which has been reached, this is the resultant velocity and these are the 2 velocity components. And output pulse rate of that particular interpolator, they become exactly these values, okay. So output pulse rate, they are fed to the control loops, at the same time they are also fed back so that they update the values of $R \sin \Omega T$ and $R \cos \Omega T$, this is how it works. No coming back let us now discuss something which is little more general that is instead of cutting straight lines and circles, et cetera, do we do something else?

(Refer Slide Time: 17:56)

Curvilinear interpolators

- The interpolators that we have been discussing are hardware interpolators. There are software interpolators also. Some of these are simply 'software translations' of the hardware interpolators while others work on different principles.

CNC reference pulse interpolators

- The most common interpolator of this kind is simply a soft version of the hardware interpolator that you have studied. However, the problems of the hardware interpolator are identified and removed. For example – the hardware interpolator has its DDA sizes fixed and this comes from the consideration of worst cases – so that very high values of system clock frequencies are required.

Instead of simply achieving linear and circular interpolation, software interpolators can be employed for finding out points on parametric curves.

- Such Software interpolators can be 'offline' or 'online' as per system requirement.

And in addition to that, are we always using hardware interpolators? Is there anything called software interpolator? There is; software interpolators are very much there and software

interpolators for example, reference pulse interpolators in a way they mimic the the working of the hardware interpolator. Inside the hardware interpolators were having sequentially an overflow of the feed DDA going to the going to the X DDA and Y DDA and their turn they are producing overflow pulses and these overflow pulses are sent to the control loops.

In the same way, we can have the working of the software program in which the control 1st goes on doing some kind of virtual addition inside a feed subroutine, which when it overflows a certain limit, it goes to the X subroutine and Y subroutine and addition takes place there also and they produce the respective outputs and then if an overflow takes place, a pulse is sent to the X or Y motor whichever is the case. So in that case there are certain developments in case of software. In case of software the advantage is that, while hardware interpolator always has the counter size et cetera, always fixed, in the software integrators such restrictions are not there.

Like in the hardware interpolators you always have to have size of the interpolator to be 2 to the power n - 1, such restrictions are not there in case of software interpolators okay. So in many cases we can we can have lot of flexibility in the software interpolators and that way we have the reference pulse interpolator working in in software and working on the principle of the hardware interpolator. And instead of achieving just linear and circular interpolation, we can also have software interpolators for parametric curves, what is the meaning of this?

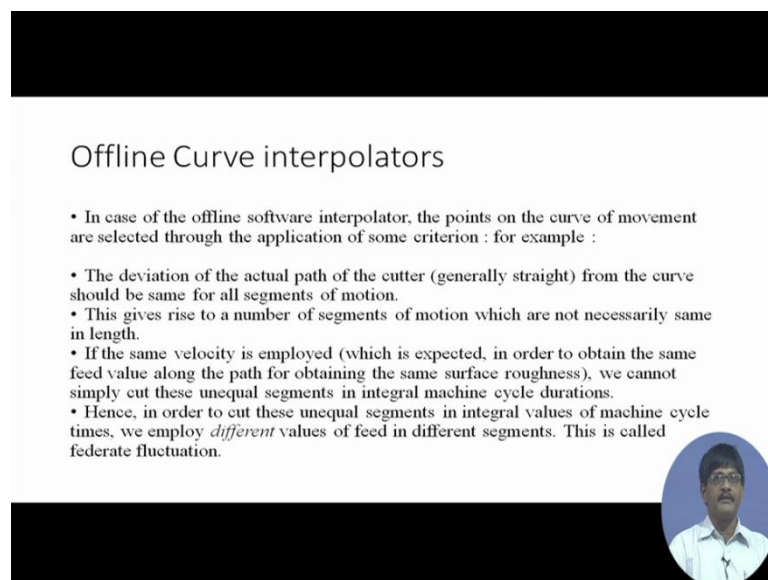
So if I have a curve which is neither a circle nor any of the common geometrical curves that we come across; ellipse, parabola, hyperbola, et cetera, but these parametric curves are free of any mathematical form, they are also sometimes called Free form curves and say the movement of the cutter has to take place along these. In such cases also we have interpolators and these interpolators can work off-line as well as online, now what does this mean? An interpolator which is working off-line it means that away from the actual point of machining, it is doing some calculations and finding out some points through which the cutter has to move and these points are then finally dumped to the machine memory.

So when these points are dumped to the machine memory, the machine simply goes through I mean leads the cutter through these points and whatever curve is being machined that machining can be done very easily, this is off-line. Online means that when I am at a particular point on the curve, I do calculation to compute the next point by some sort of you know, some sort of very fast routine. Why fast because if I moving from one point to another and the distances are typically very small, in that case I do not have much time at my disposal

to calculate the next point, so some very fast calculation is done like some Taylor series approximation is resorted to in order to find out the next point.


Once the next point is found, again the next point, calculation for the next point takes place. And if there is some provision for some error correction, because whenever you are trying to move to a particular position, in that case there is always a chance that you will be making some error and feedback might be telling you that you are away from the target by this amount, so this is the typical working of the online curve interpolator. But why at all has there been a branching of into 2 types of interpolators? So obviously, there must be some disadvantage for off-line and some disadvantage for online as well, let us have a quick look at that.

(Refer Slide Time: 22:56)



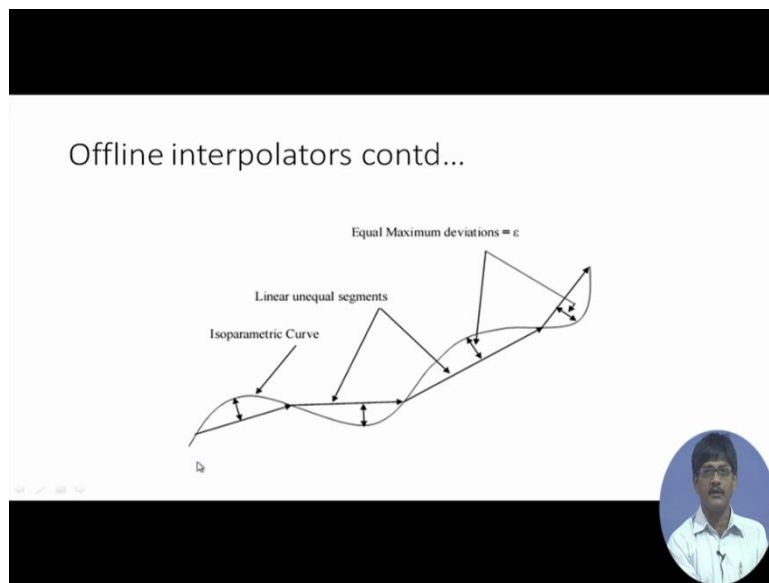
Offline Curve interpolators

- In case of the offline software interpolator, the points on the curve of movement are selected through the application of some criterion : for example :
- The deviation of the actual path of the cutter (generally straight) from the curve should be same for all segments of motion.
- This gives rise to a number of segments of motion which are not necessarily same in length.
- If the same velocity is employed (which is expected, in order to obtain the same feed value along the path for obtaining the same surface roughness), we cannot simply cut these unequal segments in integral machine cycle durations.
- Hence, in order to cut these unequal segments in integral values of machine cycle times, we employ *different* values of feed in different segments. This is called federate fluctuation.



So in case of off-line software interpolators, the point on the curve of movement are selected through the application of some Criteria, so what do we mean by that? There is a particular arbitrary curve and on that I select certain points through which I am supposed to move. For example, we might see the division of the actual path of the other okay, the cutter is always moving straight line I mean it it approximates a curvilinear motion by breaking it up into small straight lines, let me quickly draw and show you an example or do I have an example here? Yes.

(Refer Slide Time: 23:38)



This curvilinear path okay, this is the path which is the path which should ideally be taken by the cutter, it is a parametric curve, so this particular parametric curve is our goal, the tool should be moving along this particular path. But as the tool can only move in straight line, it is moving like this, this is the path of the tool. It is going this way, then is going this way, then it is going this way, these are the pathways of the tool. However, there is a deviation you cannot avoid deviation if the tool is moving along straight line segment and the path to be taken ideally is a curve. So we observe that these segments which are building up the tool path, they are unequal segments and these deviations which are shown, these are equal, why so?

This is not arbitrarily happening, but this is what we try to do at least I mean this equality of the deviations we try to achieve, once again why? This is because whenever we are accepting the fact that we have to deviate from the path, and then from formed tolerance a particular value of the maximum deviation is stated. So suppose we say that okay we accept that the tool will be deviating, but this deviation should always be equal to 50 microns. Mind you, not feed in but we are suppose saying that it should be equal to 50 microns, let us see what these leads do.

So in that case what we do is, we do some sort of calculation in off-line computing system, you have all the time to do all these calculations and what we do is, we set up a target and we decide okay, this is my point and then you make a calculation okay this is my deviation and therefore, you find that it is not equal to 50 may be it is equal to 45, so you can do some sort of iteration that means repetitively you can use the results of last calculation, update your

calculations that way okay and ultimately you will always try to achieve a deviation of 50 microns by placing the target point of the cutter for this 1st cut at some change point.

So this way say after 5 or 6 such iterations, you hit upon a deviation of 50 microns that is very good. And suppose you find that this particular segment length is equal to say X, for the next segment once again this is the maximum deviation and this is say this you equate to 50 microns by calculations and come up with another segment and therefore there is absolutely no reason why these should be equal to each other. If they are completely, they have no relation to each other, they are completely arbitrary that way and therefore we understand that these 2 these 2 segment lengths, they are not necessarily equal to each other.

So I have segment 1, segment 2, segment 3, segment 4 coming up and they are of different lengths, while I have deviation 1, deviation 2, deviation 3, et cetera equal to each other and I am satisfied. Deviations are the same as laid down by the programmer, so what is wrong if these segments are not equal? The problem is, if we have different values of these segments, these segments have to be covered in the same time intervals, why? Because velocity values can only be imparted to the machine controls at the end of a machine cycle, a machine has a definite number of cycles and at say we assume at each machine cycle start, we are setting a different value of velocity for that particular machine.

So at this point say first machine sorry here starts the 1st machine cycle and it is of T seconds and for that we set a particular velocity ratio so that this is the resultant velocity. After T seconds is over, we set up another velocity and this way at after 2T we are here and we set up another velocity ratio of the axis velocity, at 3T we set up yet another one, so far so good nothing is wrong, but the problem is, if these are different then the velocity value has to be different in order to go through these points because the time interval we are setting the same. At the end of equal machine cycle intervals in time, I am setting up different velocities to attend different directions.

But since these distances are different, they have to be I mean the velocity values have to be different. Now comes in the machining from the from the machining accuracy point of view, he will say that if you do not set up equal feed velocity in these cycles, you will have a problem, what is that problem? If you say feed fluctuation occurs, surface finish cannot be maintained to be the same therefore, if you have to have maximum deviation always occurring so that you are working with maximum efficiency, in that case you have to allow for feed rate fluctuation between these and your surface finish would be non-uniform okay.

Next, so apart from the rate fluctuation suppose you say that okay I am going to tolerate different values of this particular deviation okay, I am going to tolerate different values of this particular deviation and that way keep feed to be the same. In that case, you have different contour error all through the contour. At one place you will have 50 microns, at another place you will have 25 micron. If you have no problem with different deviation, then okay you can accept it, but if you want the deviations to be same, you have to allow some feed rate fluctuation.

Secondly, if there are wide variations of the contour; I mean if this curved portion has very high curvatures, etc, in that case contour error might be occurring because this simply cannot restrict the deviation to this particular value for a reasonable segment length okay, so their contour error will be uncontrollable. Also, if this sort of a scheme is working, in that case we have to tolerate with a large value of formed tolerance, we have to we have to accept a large value of tolerance, why? Because for an extended length of this parametric curve, we can well have huge number of points if the tolerance is small, so we have to off-line calculate all these points and that will make up a very large data file size.

So in order to keep data file size within limits, generally the amount of variation of the cut profile from the ideal profile that is very large. For large tolerance values have to be accepted if you go for off-line calculations because if you are uploading a huge file it is difficult for the machine to handle because of memory size restriction okay, thank you.