Optimal Control, Guidance and Estimation

Prof. Radhakant Padhi
Department of Aerospace Engineering
Indian Institute of Science, Bangalore

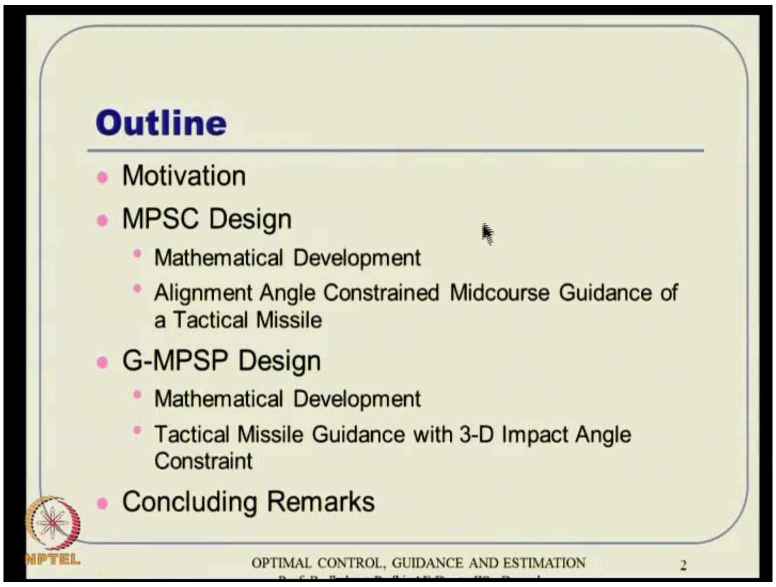
Lecture No. # 25

Model Predictive Spread Control (MPSC) and Generalized MPSR (G-MPSP) Designs

Hello everybody. Let us continue with our lecture series on these optimal control guidance and estimation course. So, last two lectures you have seen some of these advance concept or recently developed concept rather as we call as model predictive static programming and followed by its usage in in various aerospace guidance application. We will continue that line of thought in this lecture as well, and probably give some of these extension ideas on on top of this basic idea basically.

So, here we will talk about something called model predictive spread control, very close to static programming, but we will also see a slightly different version, parameters version and all that. Also, we will see this something very recent development rather as of today, something called generalized MPSP. That means, we do not want to deal in discrete domain, but we want to develop this entire procedure and that continuous time domain where discrete time thing turns out to be kind of a special case actually.

(Refer Slide Time: 01:20)



Outline

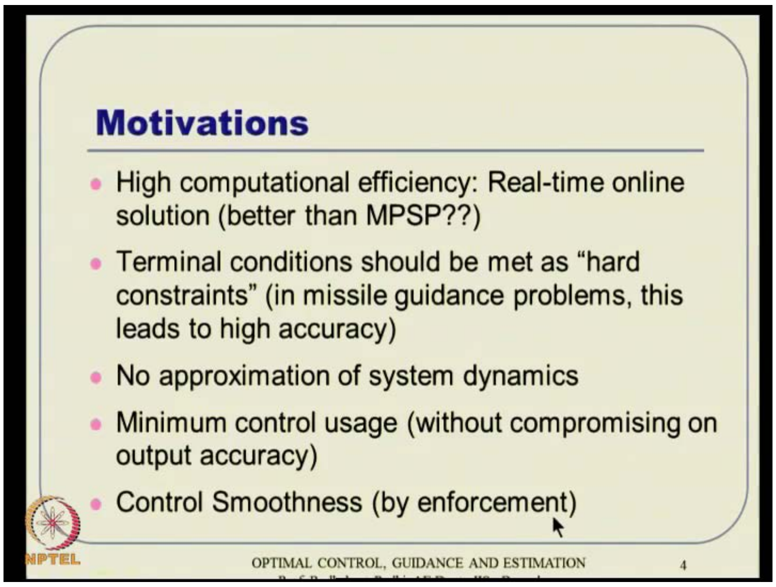
- Motivation
- MPSC Design
  - Mathematical Development
  - Alignment Angle Constrained Midcourse Guidance of a Tactical Missile
- G-MPSP Design
  - Mathematical Development
  - Tactical Missile Guidance with 3-D Impact Angle Constraint
- Concluding Remarks

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 2

So, all right. So, let us get going. The outline of this particular lecture is little bit on motivation followed by this MPSC design what you call, model predictive static control you can state control basically, model predictive spread control design followed by the this. I mean this will be. Within this frame work we will talk about mathematical development and the alignment angle constrained midcourse guidance of a tactical missile.

That will be followed by this general MPSP design. Again, mathematical development followed by another tactical missile guidance application with the 3-D impact angle constraint. Essentially, the same application that we discussed in the last lecture basically. But it will be solved in the G-MPSP procedure. That is all. Then, we will have some some concluding remarks while we talk basically. So, let us see. This first thing is model predictive spread control or parameterized MPSP. You can think about that way also basically all right.

(Refer Slide Time: 02:23)



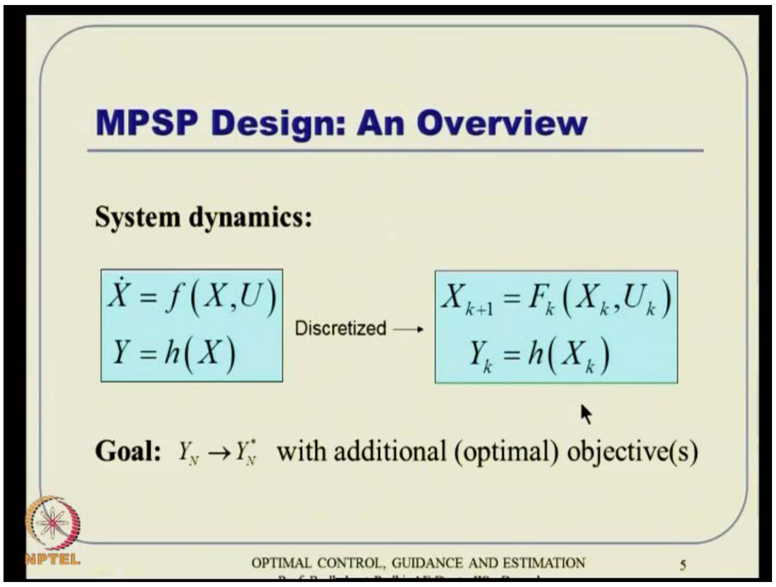
Motivations

- High computational efficiency: Real-time online solution (better than MPSP??)
- Terminal conditions should be met as "hard constraints" (in missile guidance problems, this leads to high accuracy)
- No approximation of system dynamics
- Minimum control usage (without compromising on output accuracy)
- Control Smoothness (by enforcement)

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 4

So, let us see what the motivation here is. First thing is a high computational efficiency. We do not want to compromise with that particular expect of MPSP actually. So, we want to retain that. In fact, you are asking one more question that can it be still better than MPSP. So, is it possible still? Second is again the same thing that terminal conditions should be met met as hard constraints and especially, in missile guidance problems, this leads to a very high accuracy basically and no approximation of system dynamics and then, minimum control usage. They are all similar to MPSP what you have seen before and in addition to that, we are telling, can the computational efficiency is slightly better and the last one we are talking is can control smoothness be guaranteed by enforcement? It should not be kind of luck or by enforcement, we should be able to guarantee that basically.

(Refer Slide Time: 03:22)



MPSP Design: An Overview

System dynamics:

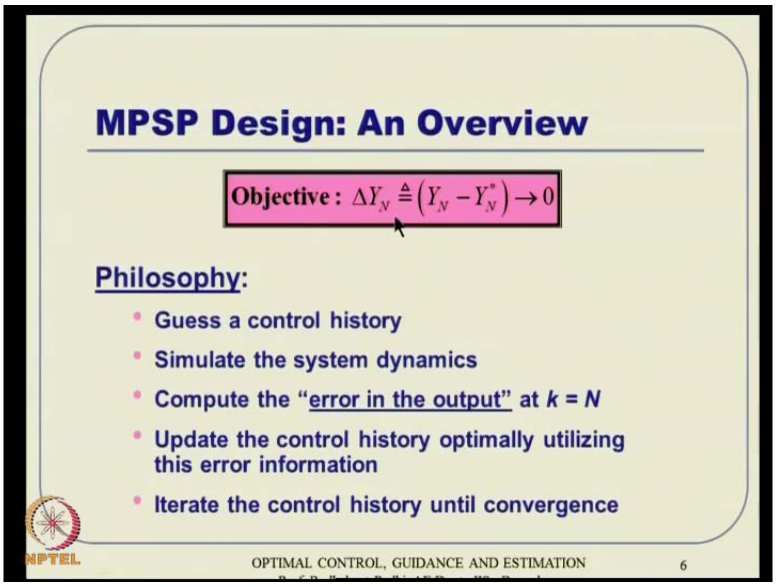
$$\boxed{\begin{aligned}\dot{X} &= f(X,U)\\ Y &= h(X)\end{aligned}} \xrightarrow{\text{Discretized}} \boxed{\begin{aligned}X_{k+1} &= F_k(X_k,U_k)\\ Y_k &= h(X_k)\end{aligned}}$$

Goal: $Y_N \rightarrow Y_N^*$ with additional (optimal) objective(s)

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 5

So, it is possible to do that. So, let us see this is how we proposed to do this, say MPSP design. What happened is, we had a system dynamics something like X dot is f of X U and Y is h of X and we discretized it to this this form, X of k plus 1 is F k of X k U k and then, Y k h of X k. That is the first thing to do there. Then, the objective is Y N should be a Y N star with some additional or optimal objective sort of thing actually.

(Refer Slide Time: 03:54)



MPSP Design: An Overview

Objective: $\Delta Y_N \triangleq \left(Y_N - Y_N^*\right) \to 0$

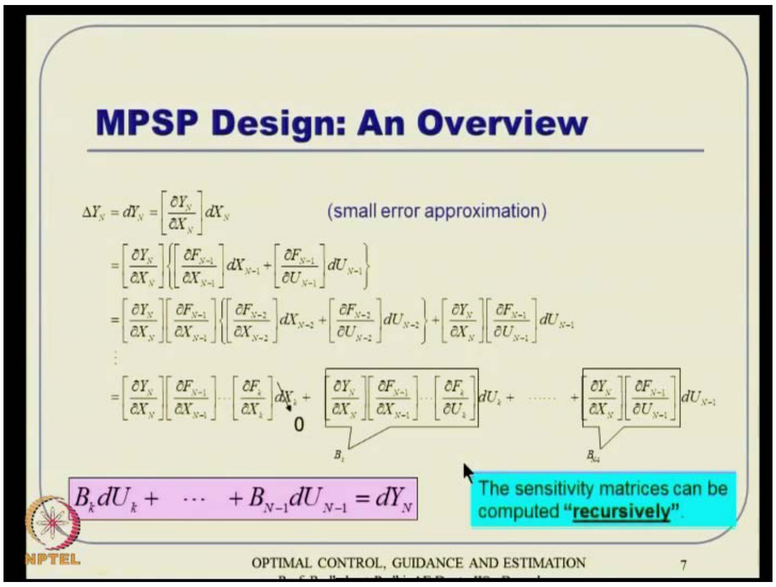
Philosophy:

- Guess a control history
- Simulate the system dynamics
- Compute the "error in the output" at $k = N$
- Update the control history optimally utilizing this error information
- Iterate the control history until convergence

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 6

So, the process of analysis what we have told is ok that the delta Y N should go to 0, Y N minus Y N star, but how? What is the philosophy? Your design we have to guess a control history and then, simulate the system dynamic, you compute the error in the output at the final time and then, update the control history optimally utilizing this error information and then iterate the control history until convergence. That is what the basic philosophy of MPSP design actually.

(Refer Slide Time: 04:20)



MPSP Design: An Overview

$$\Delta Y_N = dY_N = \left[\frac{\partial Y_N}{\partial X_N}\right] dX_N \qquad \text{(small error approximation)}$$

$$= \left[\frac{\partial Y_N}{\partial X_N}\right]\left\{\left[\frac{\partial F_{N-1}}{\partial X_{N-1}}\right] dX_{N-1} + \left[\frac{\partial F_{N-1}}{\partial U_{N-1}}\right] dU_{N-1}\right\}$$

$$= \left[\frac{\partial Y_N}{\partial X_N}\right]\left[\frac{\partial F_{N-1}}{\partial X_{N-1}}\right]\left\{\left[\frac{\partial F_{N-2}}{\partial X_{N-2}}\right] dX_{N-2} + \left[\frac{\partial F_{N-2}}{\partial U_{N-2}}\right] dU_{N-2}\right\} + \left[\frac{\partial Y_N}{\partial X_N}\right]\left[\frac{\partial F_{N-1}}{\partial U_{N-1}}\right] dU_{N-1}$$

$$\vdots$$

$$= \left[\frac{\partial Y_N}{\partial X_N}\right]\left[\frac{\partial F_{N-1}}{\partial X_{N-1}}\right]\cdots\left[\frac{\partial F_k}{\partial X_k}\right] \underset{0}{dX_k} + \underbrace{\left[\frac{\partial Y_N}{\partial X_N}\right]\left[\frac{\partial F_{N-1}}{\partial X_{N-1}}\right]\cdots\left[\frac{\partial F_k}{\partial U_k}\right]}_{B_k} dU_k + \cdots\cdots + \underbrace{\left[\frac{\partial Y_N}{\partial X_N}\right]\left[\frac{\partial F_{N-1}}{\partial U_{N-1}}\right]}_{B_{N-1}} dU_{N-1}$$

$$\boxed{B_k dU_k + \cdots + B_{N-1} dU_{N-1} = dY_N}$$

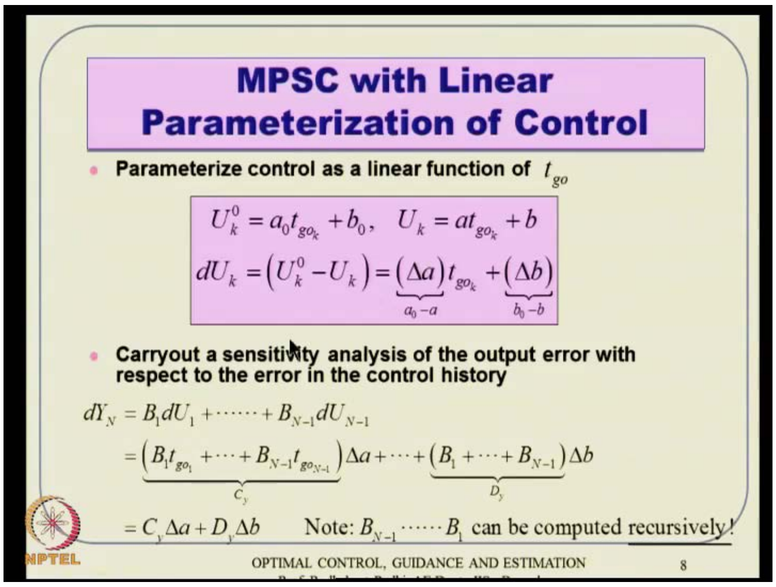
The sensitivity matrices can be computed "**recursively**".

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 7

So, in the process of analysis, we told this is our error and we approximate this approximation sort of thing and then, this dY N can be expanded that way, where d X N can be expanded that in terms of previous state and previous control. Previous control we keep it but previous state we expand in terms of another previous state and previous control, things like that, but ultimately we will end up of this kind of a expression for dY N and then, define this coefficient as something like sensitivity matrices and this sensitivity matrices can be computed recursively.

We also told that initial condition is known precisely. So, this error in initial condition should turn out to be 0 basically and then, what we really get is some linear expression something like this. That is ok. After this, the idea was to minimize the control history everywhere. In other words to minimize some sort of a quadratic performance in that actually in terms of this errors and control essentially, but here we will depart a little bit in this design and tell ok that is, that form is ok, but can you really think of something like a parameterize version of the control.

(Refer Slide Time: 05:29)



MPSC with Linear Parameterization of Control

- Parameterize control as a linear function of t_{go}

$$U_k^0 = a_0 t_{go_k} + b_0, \quad U_k = a t_{go_k} + b$$

$$dU_k = \left(U_k^0 - U_k\right) = \underbrace{(\Delta a)}_{a_0 - a} t_{go_k} + \underbrace{(\Delta b)}_{b_0 - b}$$

- Carryout a sensitivity analysis of the output error with respect to the error in the control history

$$dY_N = B_1 dU_1 + \cdots\cdots + B_{N-1} dU_{N-1}$$

$$= \underbrace{\left(B_1 t_{go_1} + \cdots + B_{N-1} t_{go_{N-1}}\right)}_{C_y} \Delta a + \cdots + \underbrace{\left(B_1 + \cdots + B_{N-1}\right)}_{D_y} \Delta b$$

$$= C_y \Delta a + D_y \Delta b$$

Note: $B_{N-1} \cdots\cdots B_1$ can be computed recursively!

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 8

To begin with, let us talk about something like a linear parameterization. In other words, we tell that the previous control history including the guess control is in the form of linear history in terms of time to go. So, the t go or t f. By definition something like we know t go is nothing, but t f minus t, wherever current time is actually. So, you can parameterize with respect to t also, no harm, but typically in guidance problems, we parameterize with respect to t go basically. So, that is compatible with guidance literature basically.

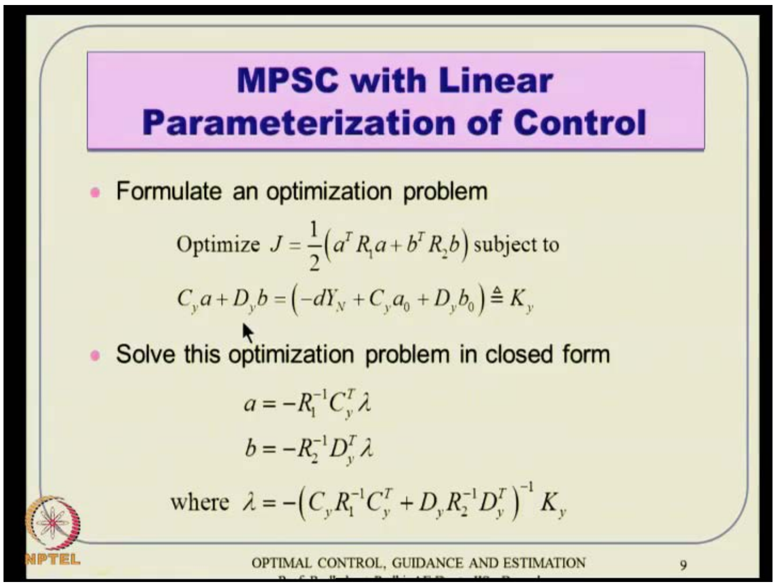
Anyway, so this is a linear expression in terms of t go and this is also the updated control which has to be linear approximation in terms of t go as well. However, what happens is, here the parameters are a naught and b naught. That is like a linear state line equation sort of thing and here, it is a and b. So, the idea here is, how do we compute a and b, given the fact that we know a 0 and b 0. So, you with that now knowing this this U k0 and U k, we define this this dU k is something like what we did before. However, this particular expression can be, once you substitute this expression here and here, you substitute U k naught with this thing and U k with this expression. Then, it turns out that dU k, then we approximated as delta a times t go plus delta b, where delta a and delta b are defined as these arrows actually.

Now, what you see here is the entire expression of control dU k is expressed in terms of delta a and delta b, no matter what value of k is actually. So, for various values of k, this t go k will change, but delta a and delta b will remain same and taking advantage of that, when you go back to this expression here, we write this this expression again will turn out to be something like this. You substitute all these expressions what you get here and it turns out to be something like this ok.

So, what you define is, the entire thing can be defined as something like C y and the entire thing can be defined as D y. Then, it turns out that dY N can be expressed as some sort of a linear combination of delta a and delta b, ok. So, the whole idea here is the entire flexibility (()) down to only in terms of delta a and delta b. In other words, we can simply update this a and b coefficients and you are done actually. So, the number of pre-variables in this particular case reduces to only 2. That is because we have actually parameterized in terms of linear expressions and linear expressions typically, I mean straight line expression should be very exact. So, this straight line expression can have only 2 flexible parameters actually. So, because of that, this dY N has become a function of only delta a and delta b

As we know this sensitivity matrixes can always be computed recursively. That you can see some previous lectures too, how you do that and all that. All right. The point is, it is now a linear expression in terms of delta a and delta b. Now, what happens here is, we we formulate this optimization problem now telling that ok optimize this quadratic cost function subject to this expression. What you got? This simply got it here actually dY N is C y times delta a plus D y times delta b, but you can always substitute what is delta a and what is delta b and solve it in terms of a and b basically. So, if you just substitute that and then, rewrite this same expression turns out to be this expression actually ok.

(Refer Slide Time: 08:55)



MPSC with Linear Parameterization of Control

- Formulate an optimization problem

$$\text{Optimize } J = \frac{1}{2}\left(a^T R_1 a + b^T R_2 b\right) \text{ subject to}$$

$$C_y a + D_y b = \left(-dY_N + C_y a_0 + D_y b_0\right) \triangleq K_y$$

- Solve this optimization problem in closed form

$$a = -R_1^{-1} C_y^T \lambda$$

$$b = -R_2^{-1} D_y^T \lambda$$

$$\text{where } \lambda = -\left(C_y R_1^{-1} C_y^T + D_y R_2^{-1} D_y^T\right)^{-1} K_y$$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 9

So, what you are telling here is, we optimize this cost function subject to this linear constraint actually. Now, what is the idea behind optimizing this quadratic cost function? In other words, I mean we are interested in a solution with a and b both being minimum actually. Now, if you go to a straight line, go to a straight line expression, what do the x, what do they mean basically? Now, if you see this expression, it is something like a and b will represent something like y intercept and slope actually ok.

All right. So, if you, I mean if you go back to this expression, this a naught and b naught and a and b, what you see here will represent a is nothing, but this slope in terms of t go and b is nothing, but the y intercept actually ok. So, if you plot this as function of t go, this is nothing, but a is a slope and b is the y intercept sort of thing actually ok. So, by demanding that the slope should be minimum and y intercept should be minimum, what you are telling is, it should be almost like a constant function, not necessarily constant unless a is exactly 0, but subject to the optimization procedure here. If a is very small or close to 0 rather, then the control profile that you are interested is almost constant and because b is also 0, what you are telling is the constant value is almost close to 0 basically, so that the value of the constant should also become close to 0. So, that is the whole idea why we want to kind of minimize this cost function actually.

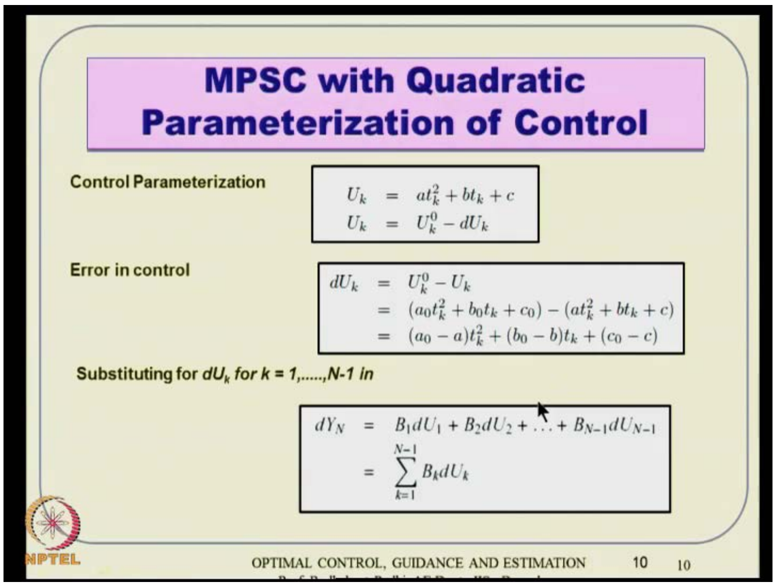
Again, how much you want to play with the slope and how much you want to play with the intercept, it all depends upon the values of R 1 and R 2. If you select both to be same, then we are giving equal importance. Otherwise, if one is relatively higher than the other, then I mean the importance of that particular expression becomes higher actually. In other words, if R 1 is higher compared to R 2, then the solution will tell that a become smaller than b actually. All these things we have discussed in LQR class also basically.

Anyway, so coming back. This is objective to optimize or minimize this cost function subject to this linear constraint. Obviously, it is a static optimization problem again and it is a very simpler problem actually. So, the solution of that again you to formulate a j bar and then, take derivative of j bar with respect to a. I mean, del j bar b f del a equal to 0, del j bar by del b equal to 0 and del j bar by del lambda also equal to 0 ok. Then, we will turn out to the solution tools out to be something like this actually, a and b will turn out to be like this, where lambda can be computed that way. So, again in other words, you got a solution in terms of a static lambda again and the solution turns out to be only two variables actually ok.

No matter how long is the control history and things like that, the solutions turns out to be just two. So, in a way, it also addresses the problem of this cursive dimensionality in some sense because it now does not define on what is the length of your control application time. So, even if your control application time turns out to be high or the number of grid point are high, then actually it does not matter with a limit of that actually ok. So, in that sense, it kind of addresses that problem as well. Anyway, so this is what it is. Now, remember this optimization can be done only when the the dimension of this equation turns out to be smaller than the dimension of a and b, I mean a and b vector together. I mean the dimension of this constraint equation turns out to be smaller than 2. In other words, the objective is just one actually ok.

So, if you talk about kind of a missile gradients problem, if you talk about only mist is turns out of thing, then this is perfect actually. Otherwise, what happens is the constraint, if the constraint turns out to be of equal dimension; that means, number of variable are same as numbers of constraints, then you simply solve it. In other words, there is no flexibility of optimizing can be varied and that is obvious.

(Refer Slide Time: 13:51)



MPSC with Quadratic Parameterization of Control

Control Parameterization

$$\begin{aligned} U_k &= at_k^2 + bt_k + c \\ U_k &= U_k^0 - dU_k \end{aligned}$$

Error in control

$$\begin{aligned} dU_k &= U_k^0 - U_k \\ &= (a_0t_k^2 + b_0t_k + c_0) - (at_k^2 + bt_k + c) \\ &= (a_0 - a)t_k^2 + (b_0 - b)t_k + (c_0 - c) \end{aligned}$$

Substituting for dU_k *for k = 1,.....,N-1 in*

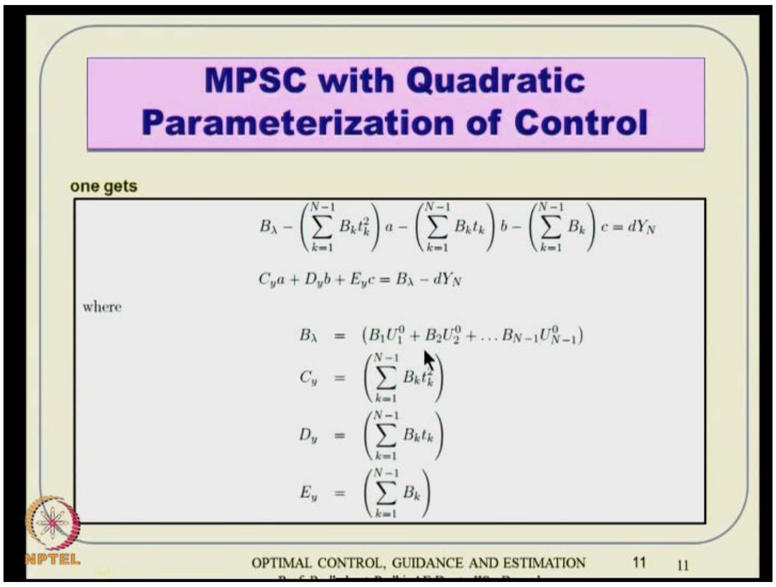
$$\begin{aligned} dY_N &= B_1dU_1 + B_2dU_2 + \ldots + B_{N-1}dU_{N-1} \\ &= \sum_{k=1}^{N-1} B_k dU_k \end{aligned}$$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 10 10

Now, coming to this control parameterization, suppose somebody does not to want this linear parameterization or it turns out to be very restrictive, it does not give you a solution and thing like that, so it is time to think about something like a little more general. So, this turns to be a quadratic function and in terms of t or t y either way. The choice is yours actually. So, there we did with respect to t go and somebody can always think about ok I do not want it. I will parameterize with respect to simply t. So, it turns out to be U k is a function of t, say t square plus b t k plus c sort of things. This is a quadratic expression now and U k again, this control update has to happen this way.

So, dU k is by definition these 2 below and hence, it is again you can do this and all the time it nicely turns out that the error in parameter turns out to be linear actually. The coefficients if you look at it, it turns out to be something like linear expression actually. The coefficients if you look at it, it turns out to be something like a linear expression actually. So, again you go back to this expression and turn ok. This can be done that way and then, we substitute all that expression to get something, something similar to what you have done before. Now, the parameters of the 3 parameters rather in terms of a, b and c.

(Refer Slide Time: 14:56)



MPSC with Quadratic Parameterization of Control

one gets

$$B_\lambda - \left(\sum_{k=1}^{N-1} B_k t_k^2\right) a - \left(\sum_{k=1}^{N-1} B_k t_k\right) b - \left(\sum_{k=1}^{N-1} B_k\right) c = dY_N$$

$$C_y a + D_y b + E_y c = B_\lambda - dY_N$$

where

$$B_\lambda = (B_1 U_1^0 + B_2 U_2^0 + \dots B_{N-1} U_{N-1}^0)$$

$$C_y = \left(\sum_{k=1}^{N-1} B_k t_k^2\right)$$

$$D_y = \left(\sum_{k=1}^{N-1} B_k t_k\right)$$

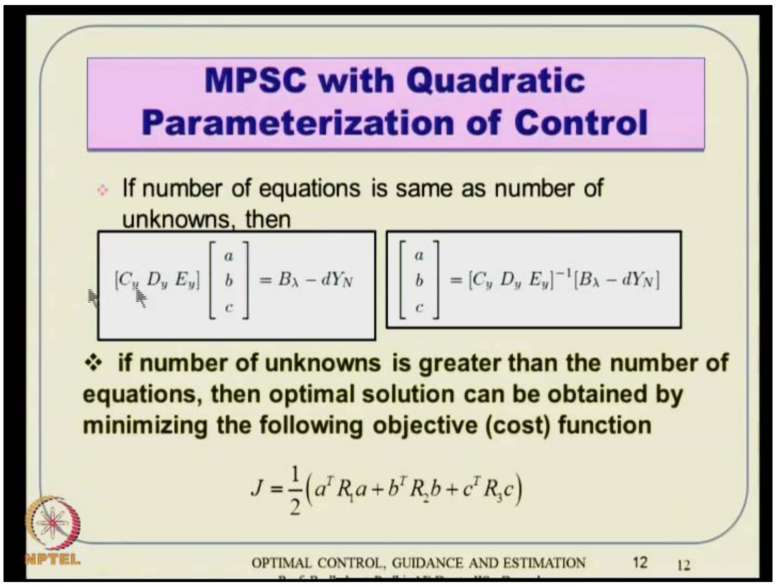
$$E_y = \left(\sum_{k=1}^{N-1} B_k\right)$$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 11 11

So, the equation again we write it as some set of a linear expression, in terms of parameter variables something like this, C y D y E y b lambda and d y and all that. You can define properly and then, tell this equation turns out to be some sort of a equation of a well, equation of a plane actually in 3-D ok.

So, you have 3 variables only to solve for. Again, no matter how whatever is the control application length, you can restrict it in terms of parameters a b c basically. All right. So, now, what happens here is we go back to this and the tell what is the cost function to optimize again.

(Refer Slide Time: 15:57)



MPSC with Quadratic Parameterization of Control

- If number of equations is same as number of unknowns, then

$$[C_y\ D_y\ E_y]\begin{bmatrix} a \\ b \\ c \end{bmatrix} = B_\lambda - dY_N \qquad \begin{bmatrix} a \\ b \\ c \end{bmatrix} = [C_y\ D_y\ E_y]^{-1}[B_\lambda - dY_N]$$

- **if number of unknowns is greater than the number of equations, then optimal solution can be obtained by minimizing the following objective (cost) function**

$$J = \frac{1}{2}\left(a^T R_1 a + b^T R_2 b + c^T R_3 c\right)$$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 12 12

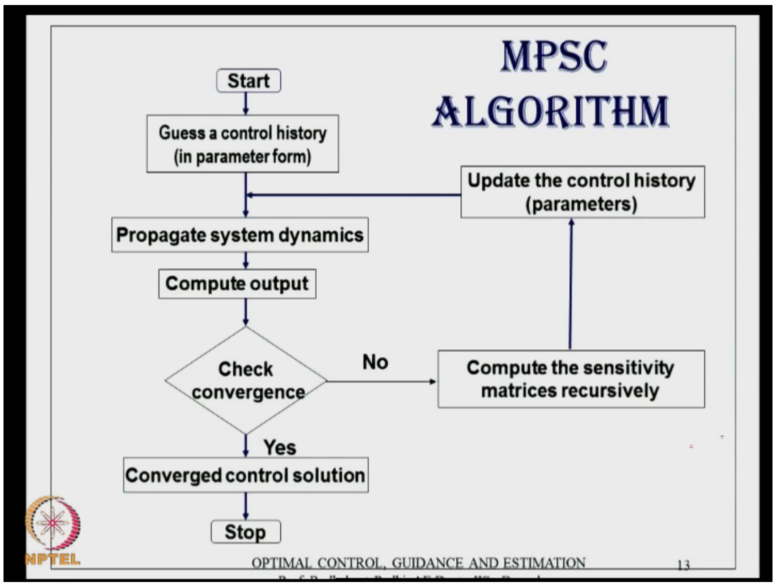
Resume that the dimension of this is equal to dimension of the number of equations, then there is no optimization. You can directly solve it something like this. It turns out to be this. This will turn out to be a square matrix in that case. The same equation what you get here is written in terms of hector matrix thing here and it turns out to be a square matrix provided the number of equations are 3, a b c basically ok. Then, in that case, you can solve it in terms of this, directly in terms of matrix inversion and all that. Remember this has to be 3 by 3 matrix. So, the matrix inversion is not computational intensive either actually.

The dimension that will depend on again the number of 3 variables we are taking in terms of parameters actually ok. Now, if it is not equal, in other words, the number of unknowns is greater than the number of equations, then there is a scope for optimization and again motivated by our previous discussion, you can think of optimizing this cost function subject to this this constant equation actually.

So, again the idea of this cost function is if you take a quadratic, I mean quadratic variable, it turns out to be something like a quadratic variable like this. Then, I mean the c again represents Y intercept. Remember this is t, this is whatever control and this is quadratic parameterization. So, c represents the Y intercept, b represents the slope, a represents the curvature, things like that. So, again by optimizing this, what you are telling is curvature has to, maximum slope has to be minimum and y intercept has to be minimum.

Again we are interested in some sort of a constant control history throughout which will give us the result; I mean what we are desired for. Even though it is mathematically speaking, it is going to be quadratic actually, but what you are interested is somewhat close to, I mean somewhat close to constant actually. That is what I am telling by the optimizing this actually, all right. So, this is the idea behind this parameterize for some of MPSP or you can name as mean that is a MPSC. Somebody can always tell it is parameterize MPSC, MPSP actually anyway. So, this is what it is.

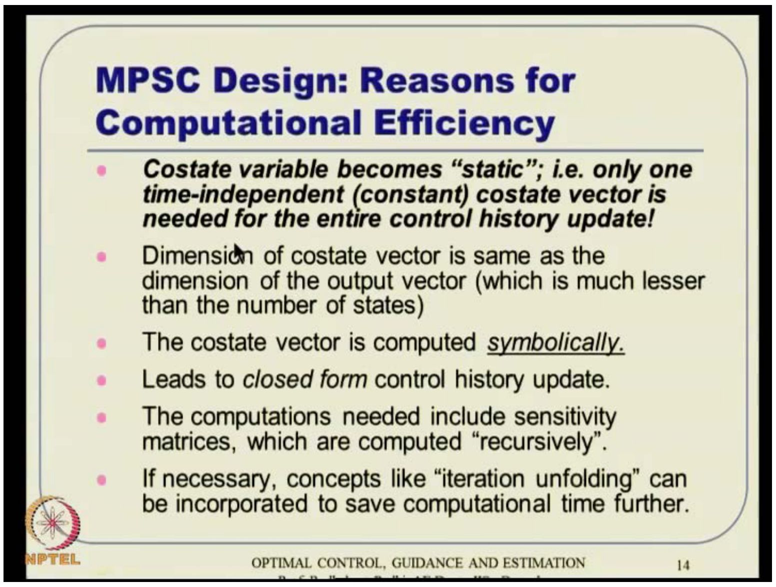
(Refer Slide Time: 18:12)


MPSC ALGORITHM
Start
Guess a control history (in parameter form)
Update the control history (parameters)
Propagate system dynamics
Compute output
Check convergence
No
Compute the sensitivity matrices recursively
Yes
Converged control solution
Stop
OPTIMAL CONTROL, GUIDANCE AND ESTIMATION
13

So, algorithm sense, it is the same, very similar to what you have done before. The only difference is we have to guess a control history in terms of control parameterization. In other words, the (()) guess itself has to be in the form of same parameterization. Suppose, we start with straight line parameterization, then the guess also needs to be straight line quadratic parameterization, guess is also quadratic like that. Then, propagate the system dynamics, compute the output, check convergence. If it is converged already, take it and stop it actually. Otherwise, you can compute the sensitivity matrixes same thing as what you done in MPSP, but then update the control history using parameters updates actually. Using the sensitivity matrixes, you compute the updated parameters and then, proceed forward further actually. This is the only difference here ok.

Now, MPSC design if you think about, why it is computational efficiency and all it turns out to be very similar. The ideas are very similar to MPSP. So, hence the regions are also very similar actually and in other on top of that, that we also get a little better computational efficiency because in terms of numbers of free variables, we are reducing the dimension quite a lot actually. In MPSP, we had all sort of grid, number of grid points multiplied by dimension of the control vector. That means a few variables are there.

(Refer Slide Time: 19:01)



MPSC Design: Reasons for Computational Efficiency

- ***Costate variable becomes "static"; i.e. only one time-independent (constant) costate vector is needed for the entire control history update!***
- Dimension of costate vector is same as the dimension of the output vector (which is much lesser than the number of states)
- The costate vector is computed *symbolically.*
- Leads to *closed form* control history update.
- The computations needed include sensitivity matrices, which are computed "recursively".
- If necessary, concepts like "iteration unfolding" can be incorporated to save computational time further.

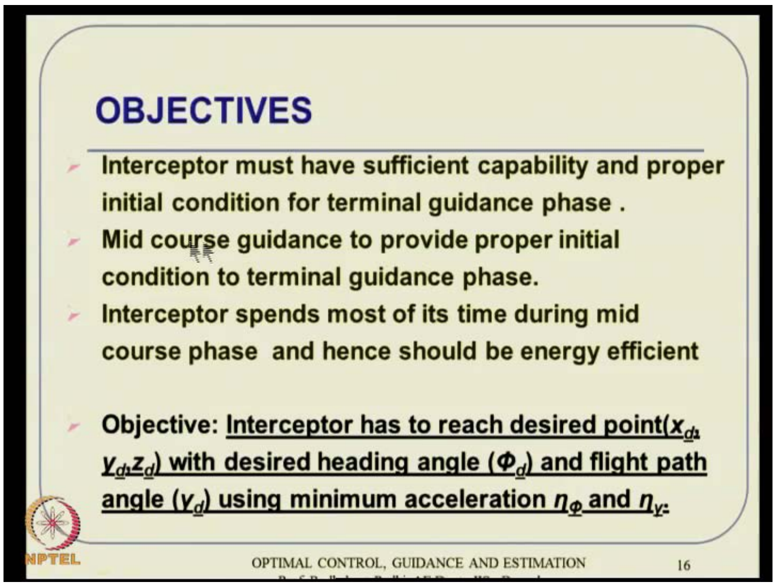
OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 14

Now, here it turns out to be number of control into number of 3 parameterization for each each dimension or each element of the control vector. So, if you have let say two controls and you have got about quadratic parameterization; that means, U 1 is a quadratic function and U 2 is also quadratic function. Then, the total number of 3 variables is 2 into 3 and that is 6 actually.

So, it kind of max it is completely independent of number of grid points and hence, it is independent of, I mean in the limited sense it addresses the issue of I mean this cursive dimensionality. However, also remember that we also need to compute the sensitivity matrices and that is a function of number of grid points. So, we really do not get rid of cursive dimensionality in that sense, but to some extent, it addresses that issues basically all right.

Now, moving on. We will see some applications of this and this problem turns out to be an angle constraint midcourse guidance using this technique actually.

(Refer Slide Time: 20:40)



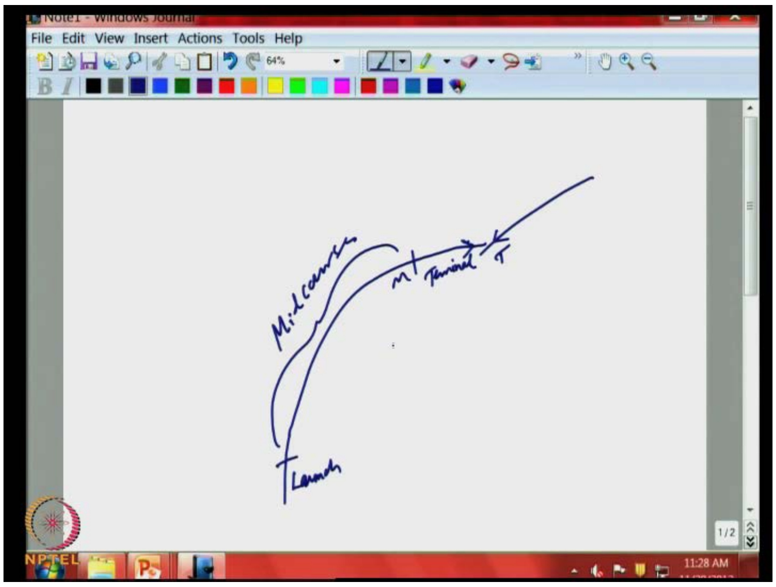
OBJECTIVES

- Interceptor must have sufficient capability and proper initial condition for terminal guidance phase .
- Mid course guidance to provide proper initial condition to terminal guidance phase.
- Interceptor spends most of its time during mid course phase and hence should be energy efficient
- Objective: Interceptor has to reach desired point(x_d, y_d, z_d) with desired heading angle (Φ_d) and flight path angle (γ_d) using minimum acceleration η_Φ and η_γ.

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 16

So, the objective here is the interceptor must have sufficient capability and proper initial condition for terminal guidance phase and mid course guidance. The whole idea here is I mean, I think I talked about that before as well. So, let me talk a little bit here. So, if you got a target which is coming or running away, whatever it is, finally, you you launch a missile. It it goes there and finally, the idea is, is to go and somewhere it has to (()) actually. So, this is the target.

(Refer Slide Time: 21:01)

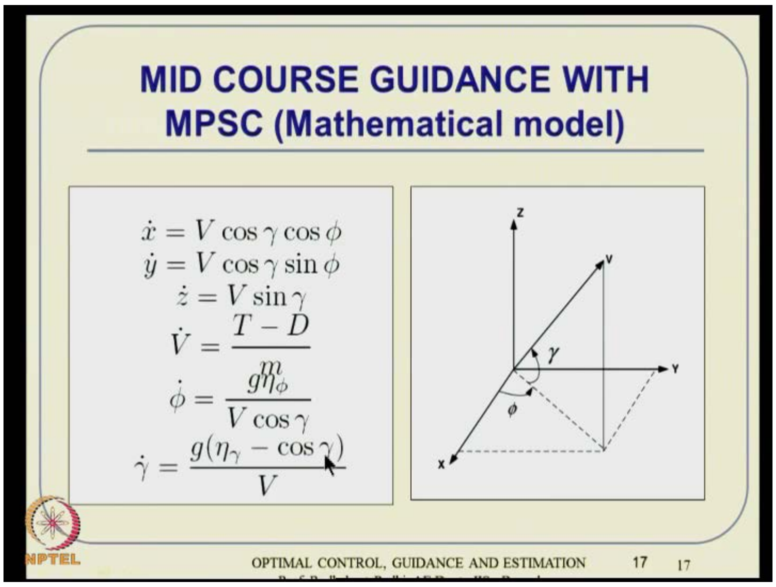


Now, this this final, this entire phase of guidance can be divided into 3 segments. One is this is launch phase or boost phase, something as called and this terminal phase, the last phase of engagement and the large part turns out to be just a midcourse guidance actually ok.

So, that is where the last part of the trajectory of the missile lies and this is the problem that we are talking about here actually ok. All right. So, interceptor spends most of its time during mid course phase and hence, should be energy efficient. That is more important actually. However, what you are telling here is see finally to go and engage with the target. So, ultimately the terminal phase dictates, so what is a mid distance angle constraint and things like that.

So, to have a proper initial condition for the terminal phase, we also need some sort of a angle constrained at the end of the mid course guidance phase actually and that is what the problem is actually. Interceptor has to reach the desired point that X d Y Z d will be given to it with desired heading angle phi d and desired flight path angle also gamma d basically and that should happen using this minimum acceleration or minimum normal acceleration actually, which is n phi and n y sort of thing.

(Refer Slide Time: 22:40)



MID COURSE GUIDANCE WITH MPSC (Mathematical model)

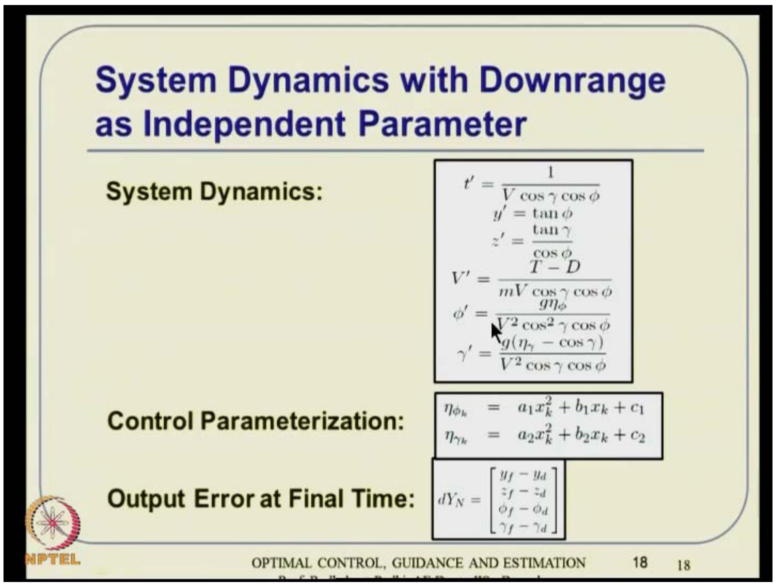
$$\dot{x} = V\cos\gamma\cos\phi$$
$$\dot{y} = V\cos\gamma\sin\phi$$
$$\dot{z} = V\sin\gamma$$
$$\dot{V} = \frac{T-D}{m}$$
$$\dot{\phi} = \frac{g\eta_\phi}{V\cos\gamma}$$
$$\dot{\gamma} = \frac{g(\eta_\gamma - \cos\gamma)}{V}$$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 17 17

So, this is the problem that we are talking here. This is I mean if you visualize these problems that way; this is the 3 dimensional plane where these dynamics is given like this, x dot y dot z dot, something like V cosine gamma which is this component times cost phi. This is your x dot x dot component. Similarly, V cost gamma sin phi is a y dot component and V sin gamma is a z dot component and then, we have phi dot and gamma dot also. How to change it phi dot and gamma dot? They can be derived something like this actually ok.

Now, the whole idea here is, how do I come up with this now, these accelerations n phi and n gamma. So, that I will accept the objective basically and here, the little bit idea was see, remember our MPSP or MPSC, the demand final time actually ok.

(Refer Slide Time: 23:25)



System Dynamics with Downrange as Independent Parameter

System Dynamics:

$$\boxed{\begin{aligned} t' &= \frac{1}{V\cos\gamma\cos\phi} \\ y' &= \tan\phi \\ z' &= \frac{\tan\gamma}{\cos\phi} \\ V' &= \frac{T-D}{mV\cos\gamma\cos\phi} \\ \phi' &= \frac{g\eta_\phi}{V^2\cos^2\gamma\cos\phi} \\ \gamma' &= \frac{g(\eta_\gamma-\cos\gamma)}{V^2\cos\gamma\cos\phi} \end{aligned}}$$

Control Parameterization:

$$\boxed{\begin{aligned} \eta_{\phi_k} &= a_1x_k^2+b_1x_k+c_1 \\ \eta_{\gamma_k} &= a_2x_k^2+b_2x_k+c_2 \end{aligned}}$$

Output Error at Final Time:

$$\boxed{dY_N = \begin{bmatrix} y_f-y_d \\ z_f-z_d \\ \phi_f-\phi_d \\ \gamma_f-\gamma_d \end{bmatrix}}$$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 18 18

So, the final time typically is not known because again that depends on the trajectory to be followed and it depends on the resolutions that we apply n phi and n gamma. The point is, we do not know it how to apply actually. So, how do you get it there, but the whole idea here is suppose, we want to go to a point something like x naught y naught z naught, that is what the point at a x d y d z d. That is what we want to go there. So, that particular point if you think about in this plane, the typical the guidance phase happens in terms of something called down range and cross range. That means, if the target is somewhere in this direction and the velocity turns out to be like that, it will go somewhere at that direction. So, this value is something like a down range plane and then, anything happens perpendicular to that happens to be cross range plane actually.

So, what you are receiving here is x is the kind of a down range plane. That is how that x is system is defined. X is the down range plane and y and z whatever happens in that happens to be the cross range plane sort of things. So, what you are assuming here is the down range is monotonic and hence, you can re-write this equation in terms of these variables actually. So, this is d t by d x. This d y by d x, this is d z by d x sort of thing. So, in other words, taking x is a free variable, we rewrite this equation because ultimately we know the value of this x d now. When x goes to x d and x be an monotonic variable, it will go to x d because x d is greater than x naught obviously.

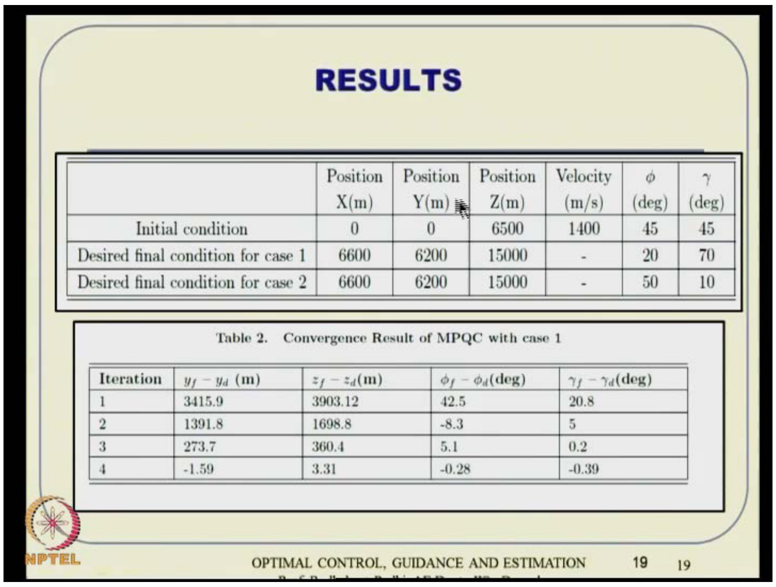
So, when x goes to x d, the problem formulation demands that y should go to y d, z should go to z d, phi should go to phi d and gamma should go to gamma d actually. This is when x goes to x d, all other things, y should go to y d, z should go to z d and phi and gamma should go to the corresponding values. Note that we are not putting any restrictions on v because our aim is not to control V.

In other words, the final impact velocity can be anything. We did not so much bothered about it and this is not really an impact velocity because we are talking about end of mid course guidance and not end of terminal phase. The idea here is typically fishy mid goes guidance literature, how do you maximize this v f and that is also important. Ultimately, we want to have the either (()) enhancement or you talk about the impact velocity wing high, but here we are not talking about that. What we are interested in is we want to guide the vehicle to a particular desired point and at that desired point; the phi and gamma should also have some desired values basically. So, that is the problem.

Now, let us this is the system dynamics. Now, these are the control variables. So, we followed this is a we do that in terms of quadratic parameterization a. Remember the free variable is not t, but x is actually here the down range. So, in terms of down range, we parameterize and phi and gamma n and then, the error functions tools have to be like this y z phi and gamma a n and then, we apply here MPSC series also. Whatever we have seen before actually this kind of logic and initial value of this parameter values a 1, b 1, a 2, b 2, c 2, they were found using this augmented p m guidance, which does not talk about angle constraint, but it will take you there. Position constant will be met; angle constraint will not be met.

So, you generate a control history like that and using list square feet, you can do this initialization of a 1, b 1, c 1 and a 2, b 2, c 2. So, from there you start and then, update this this coefficient values using this algorithm. Then, there is a result (()) you can see this this kind of thing, you have this position and this x y z.

(Refer Slide Time: 26:37)



RESULTS

	Position X(m)	Position Y(m)	Position Z(m)	Velocity (m/s)	ϕ (deg)	γ (deg)
Initial condition	0	0	6500	1400	45	45
Desired final condition for case 1	6600	6200	15000	-	20	70
Desired final condition for case 2	6600	6200	15000	-	50	10

Table 2. Convergence Result of MPQC with case 1

Iteration	$y_f - y_d$ (m)	$z_f - z_d$(m)	$\phi_f - \phi_d$(deg)	$\gamma_f - \gamma_d$(deg)
1	3415.9	3903.12	42.5	20.8
2	1391.8	1698.8	-8.3	5
3	273.7	360.4	5.1	0.2
4	-1.59	3.31	-0.28	-0.39

NPTEL OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 19 19

So, this is your initial condition. This is our initial condition and finally, we have to go there. For case 1, you have go there and case 2 also, the position remains same, but angle is different actually. For case 1, we want this angle and for case 2, we want that angle actually.

You can see very quickly how it is able to go there. (()) see something using this p m guidance error was something very high sort of thing. Even then the correction of the error is very quick, only only 4 iterations you can see the reduction of them actually, reduction of the separation distance basically. So, 3 kilometers are almost like 4 kilometers separation. It reduces to something like well, 1 meter and 2 meters sort of thing actually, I mean 1 meter, 3 meters like that.

So, that is the position part of it. What about the angle? An angle also is a 42 starting from 42 degrees 20 degrees and all it is very quickly is able to give, I mean give us very small errors in the desired value. This phi f minus phi d turns out to be the errors actually what you get ok. So, no matter whether it is case 1 or case 2, it is able to, sorry no matter the case 1 and case 2, what you are seeing here is is for case 1 actually and anyway is able to reduce very quickly actually, just 4 iterations. That is all ok. So, this also you can see how this iteration proceeds from trajectory to trajectory. Initially, it was something like this and first iteration, second iteration, third and fourth are almost top of each other actually. So, that is the final desired value.

(Refer Slide Time: 28:45)

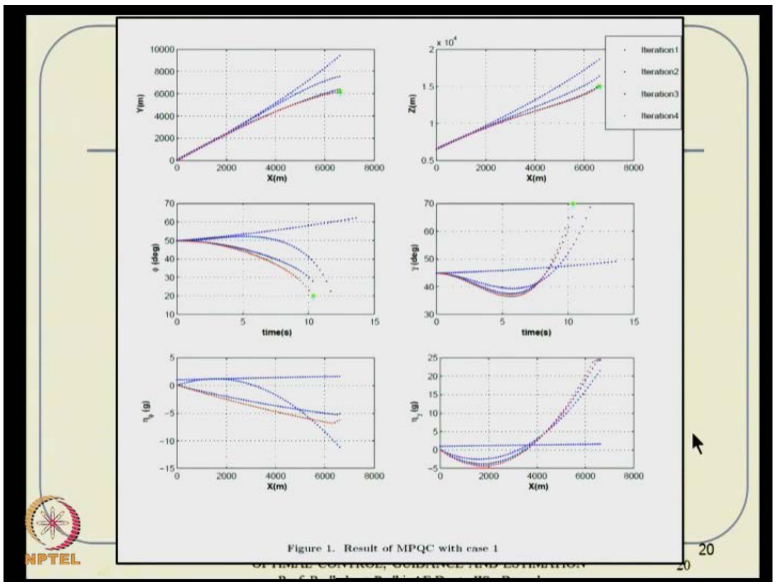
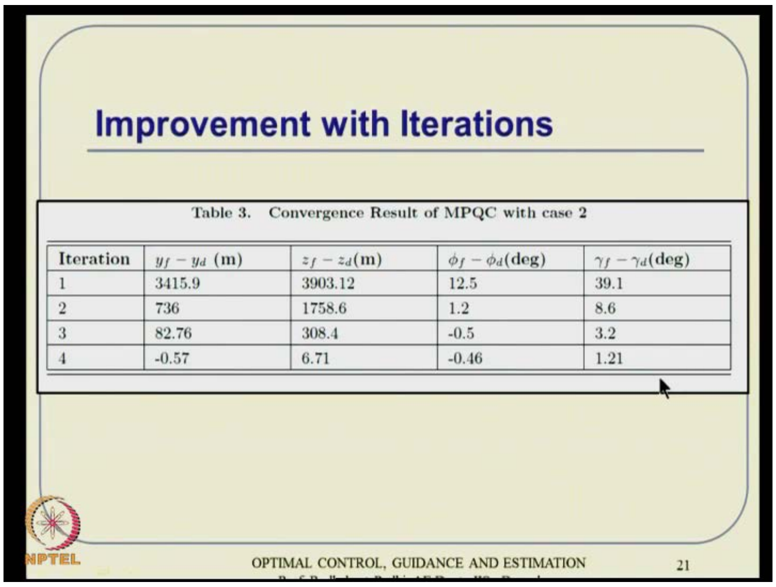


Figure 1. Result of MPQC with case 1

Similarly, on z first iteration, this is your guess and the first I mean, this iteration 1, 2 and 3, 4 basically. So, like that actually. If all the variables there how they converge and converse quickly also. Now, the angle angle value it was going somewhere that way. First iteration, second iteration, third and fourth, it is there what you want actually like that.

(Refer Slide Time: 29:24)



Improvement with Iterations

Table 3. Convergence Result of MPQC with case 2

Iteration	$y_f - y_d$ (m)	$z_f - z_d$(m)	$\phi_f - \phi_d$(deg)	$\gamma_f - \gamma_d$(deg)
1	3415.9	3903.12	12.5	39.1
2	736	1758.6	1.2	8.6
3	82.76	308.4	-0.5	3.2
4	-0.57	6.71	-0.46	1.21

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 21

So, this is for case 2 and if you see how how it improves that iteration, again you can see that starting from 3.4 kilometer and again almost like 4 kilometer miss the separation distance, reduces to something like half a meter and some 6-7 meters which is not a terminal face. We are not talking about missed distance, but whatever is the desired distance with very close to that we all, I mean whatever the desired point were going very close to that. Then, at that that point, the angle constraints are also getting (()) to very good starting from 12.5 degrees and almost like other 39 degrees, we are able to reduce it to very small values actually. In fact, if somebody wants you can continue 1 or 2 more iterations and these values will be even better actually, but we thought of coming, looking at practical aspect, there is no point of doing too much actually.

Alright. So, that is what it is. Again, similarly, you can see case 2. It reduces first iteration, second and third. I mean second, third and fourth are here. First iteration, second third and fourth are here. Like that actually it proceeds everywhere. All right. Also sometimes somebody can think about what is this MPQC. Sometimes, we thought of like we define it as model predictive quadratic control because this is all in terms of quadratic parameterization basically, all right. So, it is all various name basically.

(Refer Slide Time: 30:52)

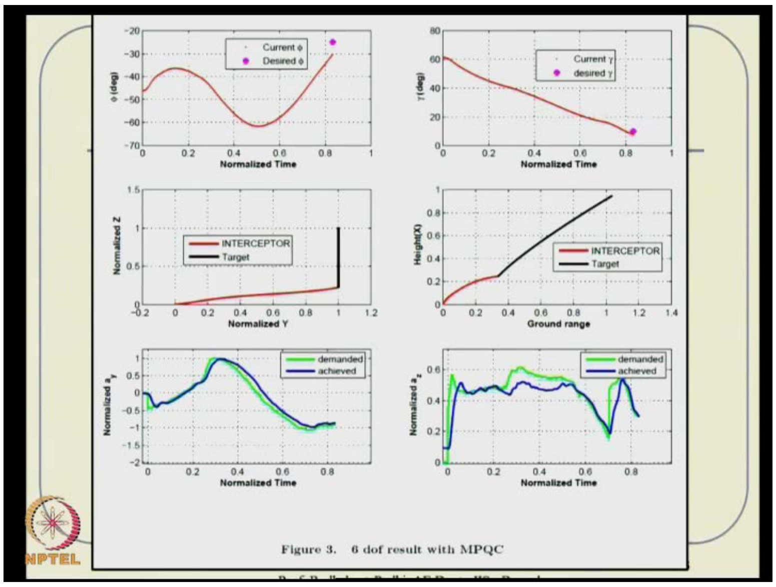


Figure 3. 6 dof result with MPQC

All right then, we thought of to validate it using 6 dof simulation and all like the 6 degree of freedom model using a control in the loop and things like that. Also see that, even if you put a auto pilot loop or control synthesis loop using full 6 degree of freedom non-linear equations and all that and the inner loops are designed based on this dynamic inversion ideas actually. So, there also you can see that I mean this guidance is able to do bizarre basically.

Ultimately, what you see is something like this and in this case, you have actually closed the loop. In other words, the tunnel phase is also there, then what actually. So, you can see this trajectory sense. These 2 trajectories give us the picture z and y direction and x and I mean, this x and z direction sort of thing actually ok. You can see that missile goes and engages with the target actually. These are some of these results with 6 dof control in x n actually.

(Refer Slide Time: 31:50)

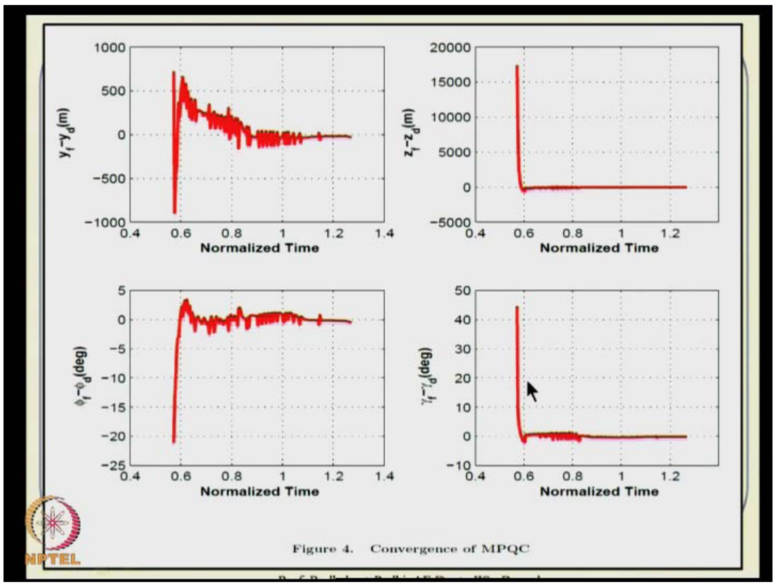
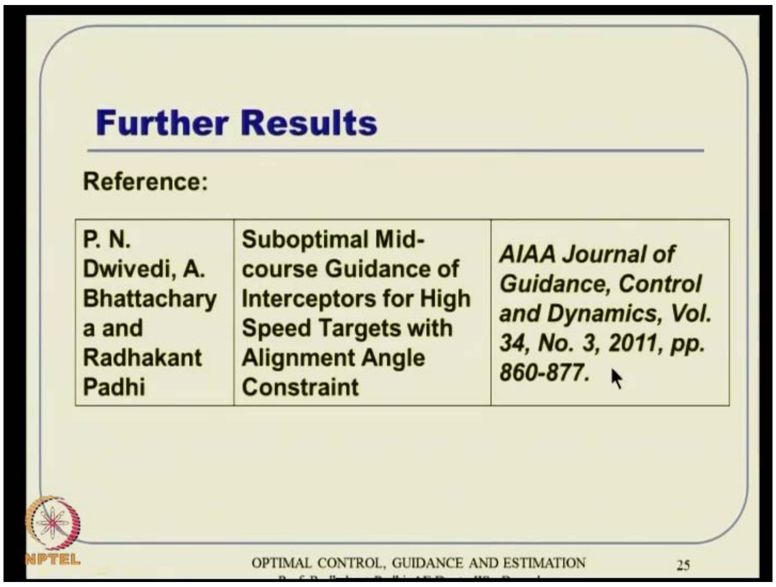


Figure 4. Convergence of MPQC

So, that means not to go through the details. What you can also see that how even with a with 6 dof simulation, you see that your final error is very close to 0 and final error in z is also very close to almost all 0 and final phi and gamma are also 0. That means, with the validation here, I mean the level of validation is 1 degree high. I mean much more higher than this guidance. Just guidance validation if you do that is fine, but many times, it is required that you also put control in the loop and then, verify whether the things are all right or not actually.

(Refer Slide Time: 32:29)



Further Results

Reference:

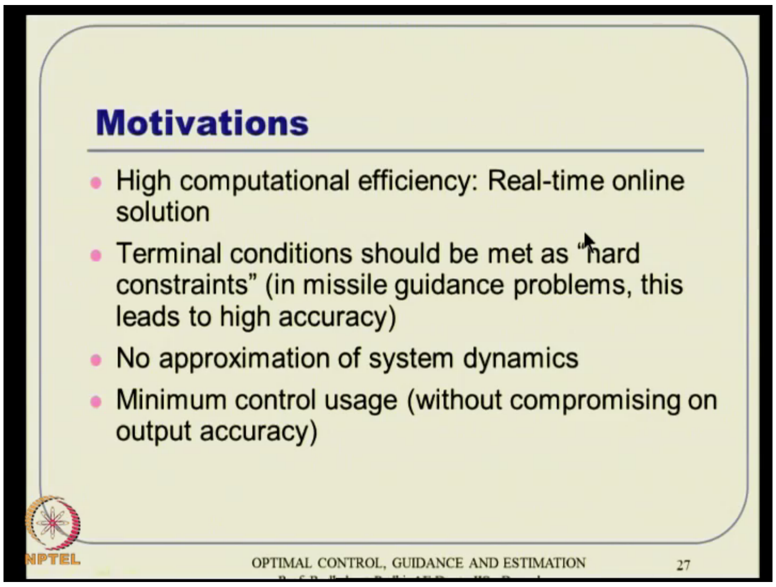
P. N. Dwivedi, A. Bhattacharya and Radhakant Padhi	Suboptimal Mid-course Guidance of Interceptors for High Speed Targets with Alignment Angle Constraint	*AIAA Journal of Guidance, Control and Dynamics, Vol. 34, No. 3, 2011, pp. 860-877.*

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 25

Further reference, if you are really interested to know more about this particular problem, I encourage you to read this paper. It is well documented there. It is a good journal of a good journal paper with something like 17 pages of write of you can see many things out there basically ok. All right.

Now, let us move quickly to what we talk about the second distinction against very recent development as of now. So, this is where you talk about generalized MPSP. The whole idea here is we want to develop it in continuous time domain actually. So, we do not want to discretise the system dynamic to begin with really. So, again motivations are very similar. We want high computational efficiency.

(Refer Slide Time: 33:04)



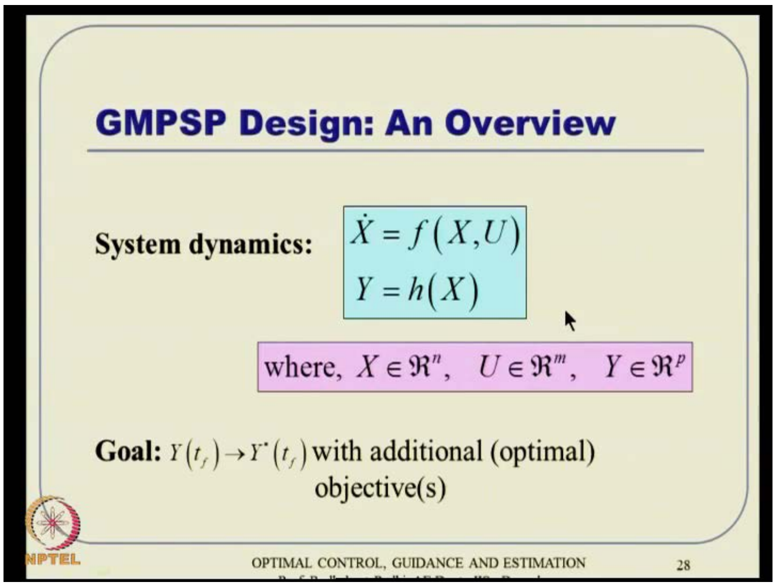
Motivations

- High computational efficiency: Real-time online solution
- Terminal conditions should be met as "hard constraints" (in missile guidance problems, this leads to high accuracy)
- No approximation of system dynamics
- Minimum control usage (without compromising on output accuracy)

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 27

So, real time online solution, that what you are interested in. Terminal conditions should be met as hard constraints, no approximation of system dynamics, minimum control usage and the last point is we want it without discretization. Otherwise, the question is can the discretized problem formulation be avoided basically? (()) uncertain out to be here and let us see how it is actually.

(Refer Slide Time: 33:41)



GMPSP Design: An Overview

System dynamics: $\dot{X} = f(X,U)$
$Y = h(X)$

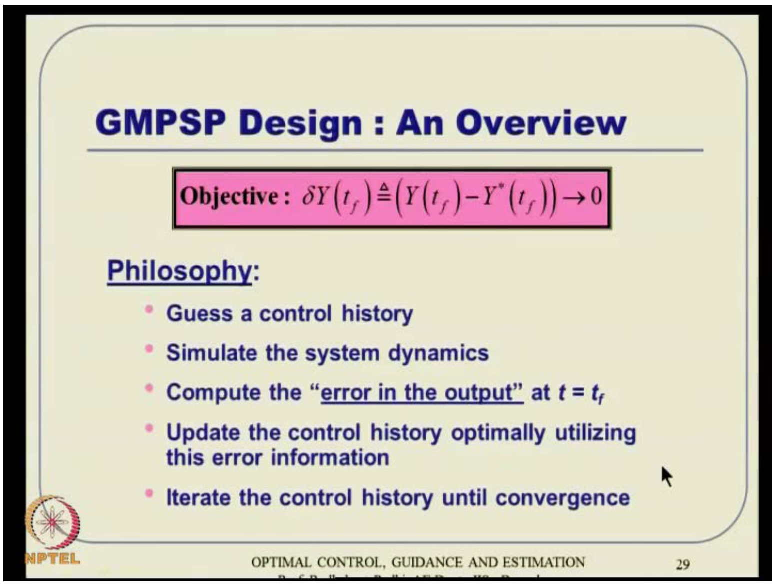
where, $X \in \Re^n$, $U \in \Re^m$, $Y \in \Re^p$

Goal: $Y(t_f) \to Y^*(t_f)$ with additional (optimal) objective(s)

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 28

All right. So, you have a system dynamic X dot id f of X U and Y is h of X, where X is n dimensional problem, I mean n dimensional vector, Y is sorry, U is m dimensional vector X is p dimensional vector standard. We also think that Y of t f should goes to Y star of t f that with some additional objectives and that is typically the control minimize objective basically. Thus, the objective remains exactly same, but we are not interested in discretizing the system dynamics to begin. That is the whole idea here.

(Refer Slide Time: 34:16)



GMPSP Design : An Overview

Objective: $\delta Y(t_f) \triangleq \left(Y(t_f) - Y^*(t_f)\right) \to 0$

Philosophy:

- Guess a control history
- Simulate the system dynamics
- Compute the "error in the output" at $t = t_f$
- Update the control history optimally utilizing this error information
- Iterate the control history until convergence

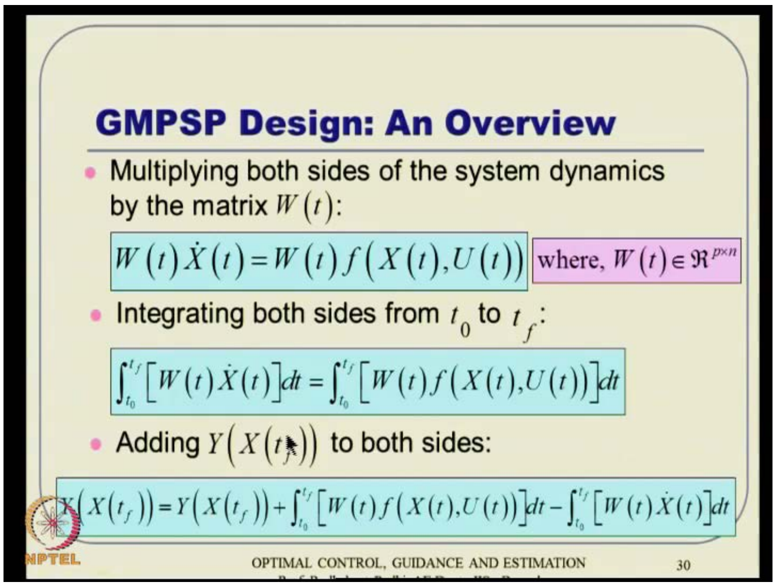
OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 29

So, what is the final thing? The final thing is the error. In other words, Y of t f minus Y star of t f should go to 0. Philosophy is also same. First we have to guess it and guess a control history, write and simulate the system dynamics to find out the error at t equal to t f and then, update the control history optimally utilizing this error.

So, the philosophy remains very very parallel actually. All right. Now, let us see how we do that. So, what you do is to begin with we know this system dynamics and this system dynamics we multiply it with some sort of a waiting matrix actually ok and time this time varying waiting matrix, we multiply it to make the system dimension same as the output put vector. In other words, after multiplying this both sides, the dimension I mean the system dynamics that we are looking at is something like in the p dimension actually and p is nothing, but the output variable dimension actually.

Ultimately, you want to see the error in Y. So, that is the whole idea why you want to do actually. So, you multiply both sides Y W of t. W is something like p by n matrix, so that the dynamics what you see is in terms of p dimensions, not in n dimensions actually.

(Refer Slide Time: 34:40)



GMPSP Design: An Overview

- Multiplying both sides of the system dynamics by the matrix $W(t)$:

$$W(t)\dot{X}(t) = W(t)f(X(t),U(t))$$ where, $W(t) \in \Re^{p \times n}$

- Integrating both sides from t_0 to t_f:

$$\int_{t_0}^{t_f}\left[W(t)\dot{X}(t)\right]dt = \int_{t_0}^{t_f}\left[W(t)f(X(t),U(t))\right]dt$$

- Adding $Y(X(t_f))$ to both sides:

$$Y(X(t_f)) = Y(X(t_f)) + \int_{t_0}^{t_f}\left[W(t)f(X(t),U(t))\right]dt - \int_{t_0}^{t_f}\left[W(t)\dot{X}(t)\right]dt$$

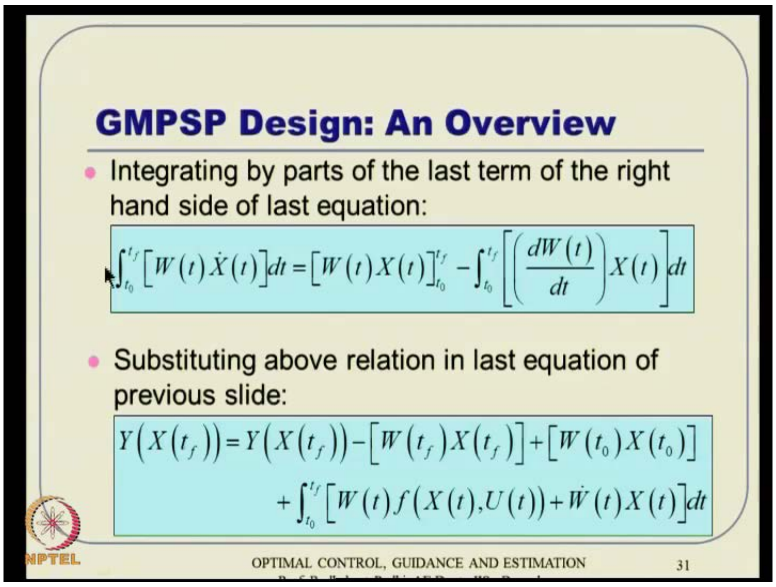
OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 30

How do you come up with this W of t? That is the whole idea here how do you, we want to kind of propose an idea using which we can actually compute the W f of t. All right. Now, what you do is integrate both sides, take this equation and try to integrate both sides actually. After you integrate both sides, you can kind of add this on both sides Y of X of t f or Y of t f basically.

Y is a function of h of X actually. So, that is why Y is a function of X and Y of X f is what we want to multiply actually. So, I mean add it actually. So, you have this constraint equation after integration and it adds this this expression Y of X of t f to both sides. So, we have this. Now, what you remember take, this one this side. By the way, this expression is taken that side. So, it is nothing, but 0, 0 equal to that. Now, Y of X f is nothing, but Y of X t f plus that, whatever else act as thing.

So, other words this whole expression minus this expression coming this side. Now, add both sides. What you get is Y of X of t f is nothing, but this below, same thing plus 0 and 0 is this one minus that one sort of thing here. Now, if you look at this expression, it contains integral of a derivative and when you have something like this and if you remember your calculus of variation a little bit, when ever expressions like this are there where typically use our integration by parts actually. So, we will do that here. So, this expression what you get here we use this integration by parts to come up with this this expression actually and if you substitute this expression back in here, what you get here is something like this.

(Refer Slide Time: 37:14)



GMPSP Design: An Overview

- Integrating by parts of the last term of the right hand side of last equation:

$$\int_{t_0}^{t_f}\left[W(t)\dot{X}(t)\right]dt=\left[W(t)X(t)\right]_{t_0}^{t_f}-\int_{t_0}^{t_f}\left[\left(\frac{dW(t)}{dt}\right)X(t)\right]dt$$

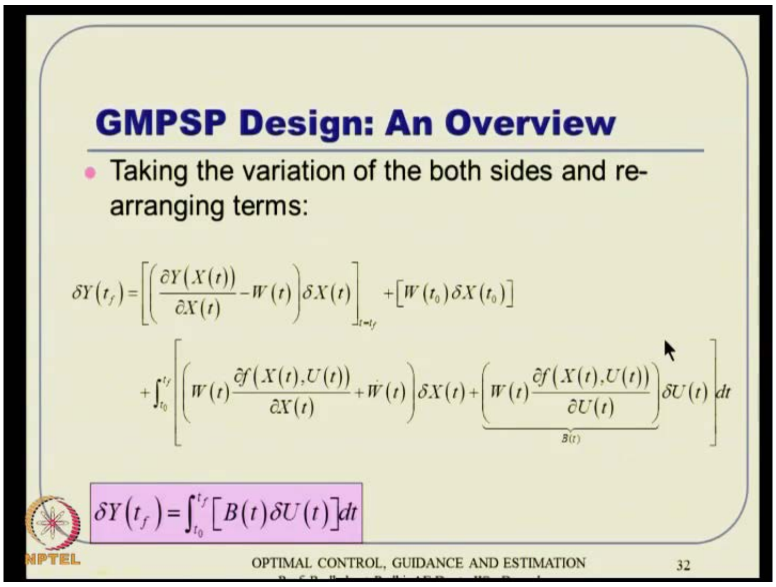
- Substituting above relation in last equation of previous slide:

$$Y\left(X\left(t_f\right)\right)=Y\left(X\left(t_f\right)\right)-\left[W\left(t_f\right)X\left(t_f\right)\right]+\left[W\left(t_0\right)X\left(t_0\right)\right]$$
$$+\int_{t_0}^{t_f}\left[W(t)f\left(X(t),U(t)\right)+\dot{W}(t)X(t)\right]dt$$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 31

I suggest that you do it with pen paper yourself, so that you will be much more convinced actually. Now, what happens? Now, you take, now this expression what you have, you take variations in both sides and try to combine terms actually.

(Refer Slide Time: 37:40)



GMPSP Design: An Overview

- Taking the variation of the both sides and re-arranging terms:

$$\delta Y\left(t_f\right)=\left[\left(\frac{\partial Y\left(X(t)\right)}{\partial X(t)}-W(t)\right)\delta X(t)\right]_{t=t_f}+\left[W\left(t_0\right)\delta X\left(t_0\right)\right]$$
$$+\int_{t_0}^{t_f}\left[\left(W(t)\frac{\partial f\left(X(t),U(t)\right)}{\partial X(t)}+\dot{W}(t)\right)\delta X(t)+\underbrace{\left(W(t)\frac{\partial f\left(X(t),U(t)\right)}{\partial U(t)}\right)}_{B(t)}\delta U(t)\right]dt$$

$$\boxed{\delta Y\left(t_f\right)=\int_{t_0}^{t_f}\left[B(t)\delta U(t)\right]dt}$$

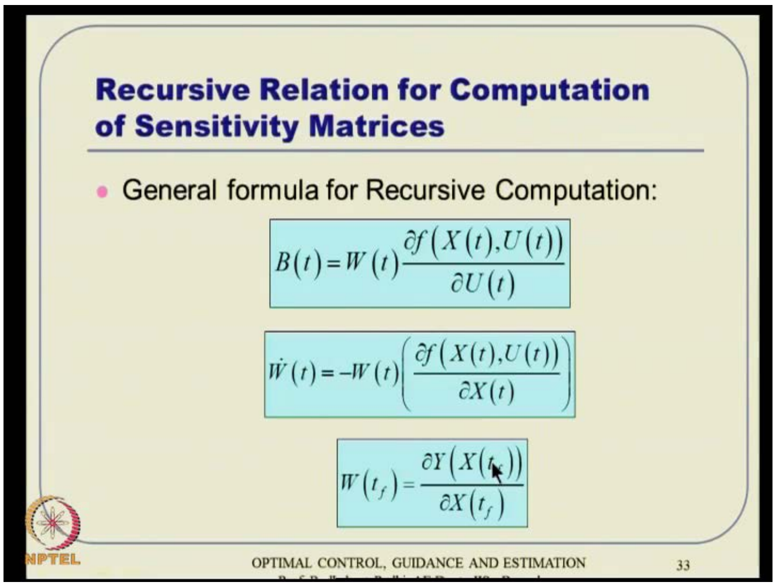
OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 32

So, what you have to see here is take variations on that, I mean y of t f variations of first variation of that and apply it in the right hand side and try combine terms and all that. Ultimately, we will land up with something like this. So, this is something like a little bit idea of calculus of variations coming into picture basically.

Then, you tell this is ok, we want it to be 0. This our idea basically because delta delta x is something that we do not know. So, obviously, 1, 2 make sure that this expression is independent of that, so we make it 0. Then, you tell initial condition is not perfectly, so you know variation of that. So, that has to go to 0 and by design, you also want to impose that this has to be 0.

Then, what happens, this first variation of Y t f turns out to be just this one actually, and if you suppose, we define this W of t, then it turns out to be like this. Again in general, it is not really recursive computed some sort of thing. Well, you can think of backward integration basically. That also we can call that is recursive, but it is not a distinct formulation. So, it is strictly speaking is nothing called recursive here actually.

(Refer Slide Time: 38:50)



Recursive Relation for Computation of Sensitivity Matrices

- General formula for Recursive Computation:

$$\boxed{B(t) = W(t)\frac{\partial f(X(t),U(t))}{\partial U(t)}}$$

$$\boxed{\dot{W}(t) = -W(t)\left(\frac{\partial f(X(t),U(t))}{\partial X(t)}\right)}$$

$$\boxed{W(t_f) = \frac{\partial Y(X(t_f))}{\partial X(t_f)}}$$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 33

So, anyway, coming back to that. What you are getting here is by definition B of t is like this. So, B of t is by definition like that. W of t, sorry W dot of t is this entire expression has to be 0. Remember that ok. So, W dot of t is negative that basically really. So, W dot of t is a negative of that expression what you are getting here. So, that will give us a differential equation to compute W of t using any numerical integration scheme including four third of numeric method actually, but for integrating this, we also need a boundary condition. Then, this boundary condition comes from this.

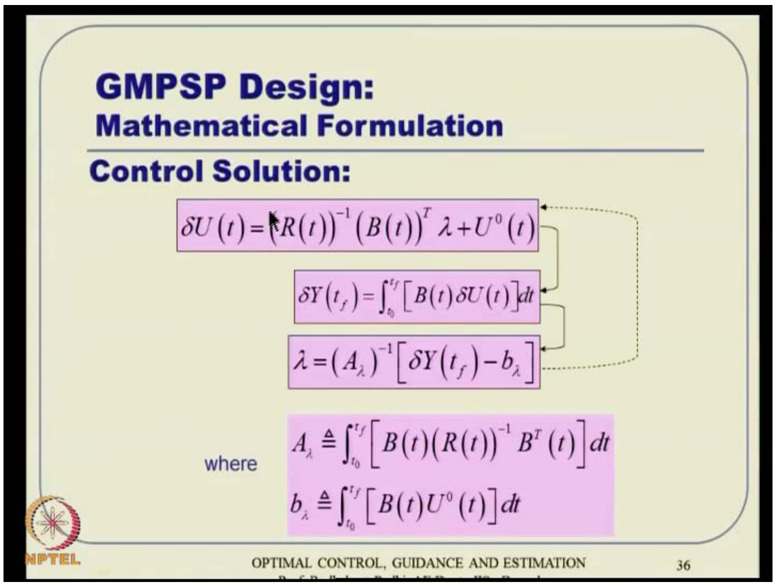
Remember, this is t equal to t f. So, this boundary condition comes out from this actually. That means, using this boundary condition and this equation, I mean this differential equation, we can quickly propagate it backward from t f to t naught and then, you will W of t everywhere. Once you get W of t everywhere, then B of t can be computed directly that way all right.

So, I hope it is clear now. Now, what happens here is we got this expression. This is a constraint equation again and this constraint equation we want to, I mean we have an account for while minimizing this cost function actually. So, that means, again remember this is your updated control. So, updated control means transpose times, r times, this updated control. That means we want the control minimization to happen actually ok. So, this is the cost function that we want to optimize, minimize rather subject to this constraint equation that we are getting after imposing this actually. Once we impose this, this turns out to be like this. So, this turns out to be the same constraint equation actually.

Now, the procedure is very similar. We have the augmented cost function similar to calculus of variation. So, augmented cost function and you tell, first differentiation is always to be 0 basically ok. So, you take about del j c bar by del of del U and then, that turns out to be 0 and this del lambda, I mean del j c bar by del lambda also it needs to be equal to 0. That gives us the same boundary condition that we started with just compatible to standard results actually.

So, if you do that and solve for this, we get something like this expression at these 3. Now, if you substitute for delta u here, then you get some expression in terms of lambda, then you solve for lambda. Once you get solve, once you solve for lambda, you put it back and then, tell this is my delta U t.

(Refer Slide Time: 41:31)



GMPSP Design: Mathematical Formulation

Control Solution:

$$\delta U(t) = (R(t))^{-1}(B(t))^T \lambda + U^0(t)$$

$$\delta Y(t_f) = \int_{t_0}^{t_f} [B(t)\delta U(t)]dt$$

$$\lambda = (A_\lambda)^{-1}[\delta Y(t_f) - b_\lambda]$$

where

$$A_\lambda \triangleq \int_{t_0}^{t_f} \left[B(t)(R(t))^{-1} B^T(t)\right]dt$$

$$b_\lambda \triangleq \int_{t_0}^{t_f} [B(t)U^0(t)]dt$$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 36

Once you get delta U, you are done actually because the control is, I mean the updated control is nothing, but U naught minus delta U basically that way. All right. So, U naught minus delta U is a control update and that is delta U. You just computed here this expression and that is what it turns out to be ok, where lambda is like this. Lambda turns out to be like this. A lambda and b lambda are also defined that way very close to what we know in MPSP, but these are all continuous time expression actually. Remember, A lambda and b lambda were summations before. Now, it has become integrations actually which is very compatible to what you know ok.

Algorithm sense, it is similar, very similar again, but the only difference is we we compute the weighting matrix by backward integration. It is you know recursive computation of sensitivity matrixes. We call that is weighting matrix here and then, we compute the weighting matrix by backward integration and then, update the control if necessary very quickly basically.

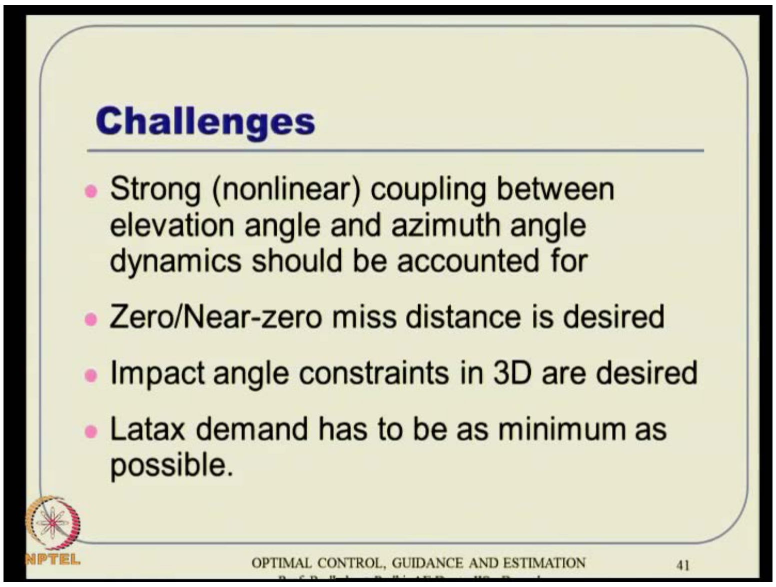
Now, using usage of this particular in the same problem let us revisit and see what are all results you are getting and as expected results will be fairly similar to what you got before actually. So, this is in the same problem that we talked in the previous lecture, tactical missile guidance with 3-D impact angle constraint and the motivations of why, what do things like that has been discussed in the last lectures. So, I am not going to talk about that again.

So, motivation again is something like, is it possible to I mean, is it possible to achieve impact angle constraint in 3-D. simultaneously, in some of 3-D image Azimuth and Elevation, both actually? Can you do that in some optimal manner basically that way? Can this terminal constraint in both the angle, that means, Azimuth angle which which talks about direction of heading and Elevation angle, which talks about pitch angle or something like top angle from the top and all that. Can it be dictated? Can it to be told to the guidance logic actually? Then, can the above objective be achieved for stationary targets or moving targets or maneuvering targets? All are ground targets, but the target can be either stationary or just moving or just maneuvering and when it maneuvers, it can have constraint latax expression, it can have time varying expression, all sort of the things actually ok.

So, these things I mean is it possible to do this. That is what we are talking actually. Also, we are telling you can this be achieved with minimum latax distribution demand. We can recapitulate or revisit the challenges. First thing is the system dynamics is non-linear and something like strong non-linear coupling between Elevation angle and Azimuth angle has to be accounted for. You cannot think about decoupling them and just design one at a time sort of thing.

Well, 0 or near 0 miss distance is desired because without 0 miss distance, angle constraint have no meaning actually unless the vehicle falls on the target towards the full valley of angles otherwise. So, that is why, it is extremely important here. 3D import angle constraints are desired. Essentially two angle constraints actually simultaneously and latax demand has to be minimum, as minimum as possible throughout sort of thing. Also remember, this latax resolution history has some implications of what is called as induced drive actually. If the vehicle turns more, then the induced track component turns out to be more also. So, hence we by minimizing the latax dissipation, we minimize the the induced track component of it and hence, the various some range extension implication and all that actually.

(Refer Slide Time: 44:34)



Challenges

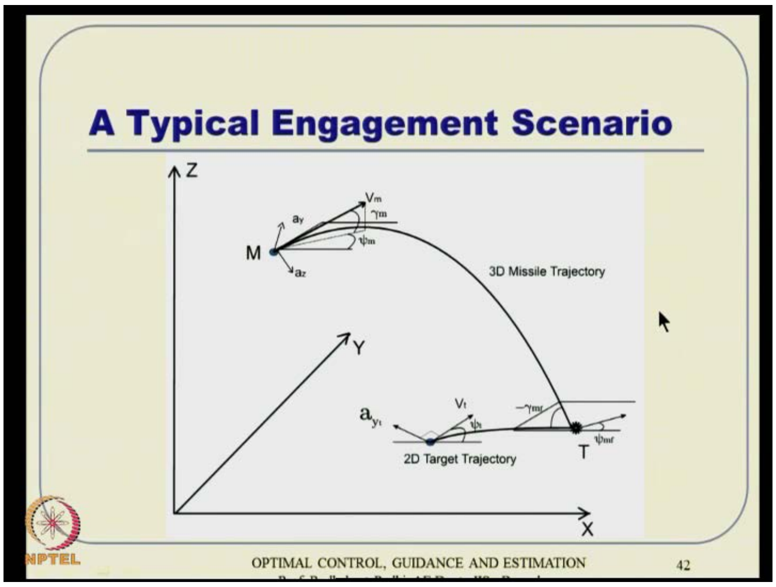
- Strong (nonlinear) coupling between elevation angle and azimuth angle dynamics should be accounted for
- Zero/Near-zero miss distance is desired
- Impact angle constraints in 3D are desired
- Latax demand has to be as minimum as possible.

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 41

So, the engagement scenario is very similar to what we have discussed in the last class. It starts with some sort of initial condition and finally, falls with some target and that point of time there are two angle constraints. One is this angle, the other is that angle.

(Refer Slide Time: 45:46)

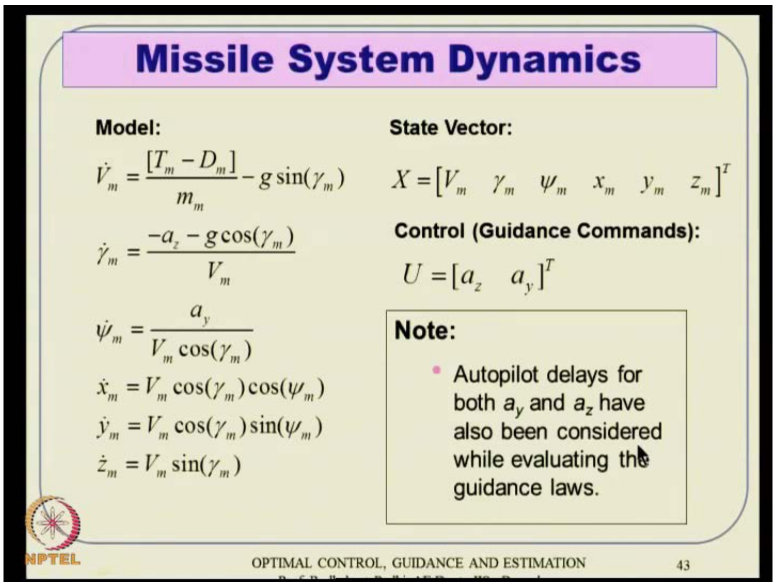
A Typical Engagement Scenario



OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 42

Essentially, this angle turns out to be something like if you extend this trajectory little bit and this angle is the negative gamma f, but this angle is same as the other side of negative gamma f actually. All right. So, that is what is actually.

(Refer Slide Time: 46:19)



Missile System Dynamics

Model:

$$\dot{V}_m = \frac{[T_m - D_m]}{m_m} - g\sin(\gamma_m)$$

$$\dot{\gamma}_m = \frac{-a_z - g\cos(\gamma_m)}{V_m}$$

$$\dot{\psi}_m = \frac{a_y}{V_m\cos(\gamma_m)}$$

$$\dot{x}_m = V_m\cos(\gamma_m)\cos(\psi_m)$$

$$\dot{y}_m = V_m\cos(\gamma_m)\sin(\psi_m)$$

$$\dot{z}_m = V_m\sin(\gamma_m)$$

State Vector:

$$X = [V_m \quad \gamma_m \quad \psi_m \quad x_m \quad y_m \quad z_m]^T$$

Control (Guidance Commands):

$$U = [a_z \quad a_y]^T$$

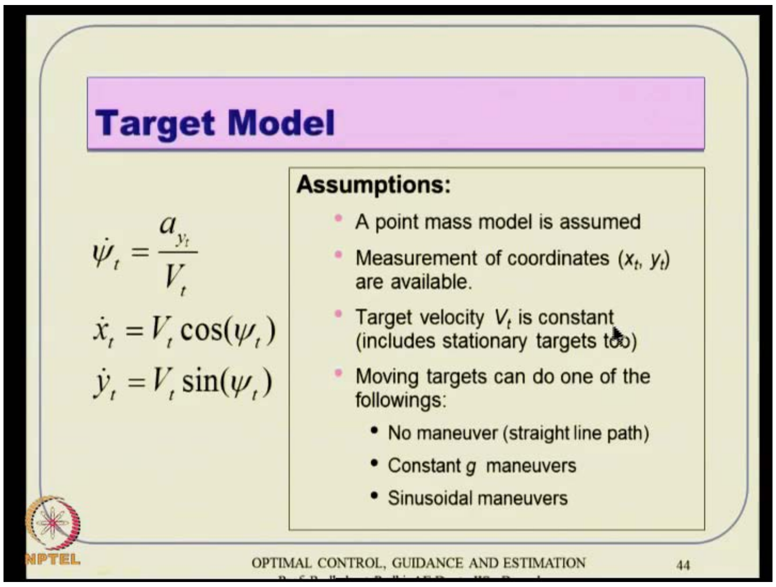
Note:

- Autopilot delays for both a_y and a_z have also been considered while evaluating the guidance laws.

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 43

So, system dynamics very same what we have discussed before. State vector, V gamma psi and x y z, control variables are z a y and delays and all that is also been accounted for while evaluating the guidance law sort of thing. Target model is a ground target. So, we have only x y because x x dot and y dot is dictated by psi. I mean psi is dictated by latax recreation that the target applies to itself actually.

(Refer Slide time: 46:36)



Target Model

$$\dot{\psi}_t = \frac{a_{y_t}}{V_t}$$

$$\dot{x}_t = V_t\cos(\psi_t)$$

$$\dot{y}_t = V_t\sin(\psi_t)$$

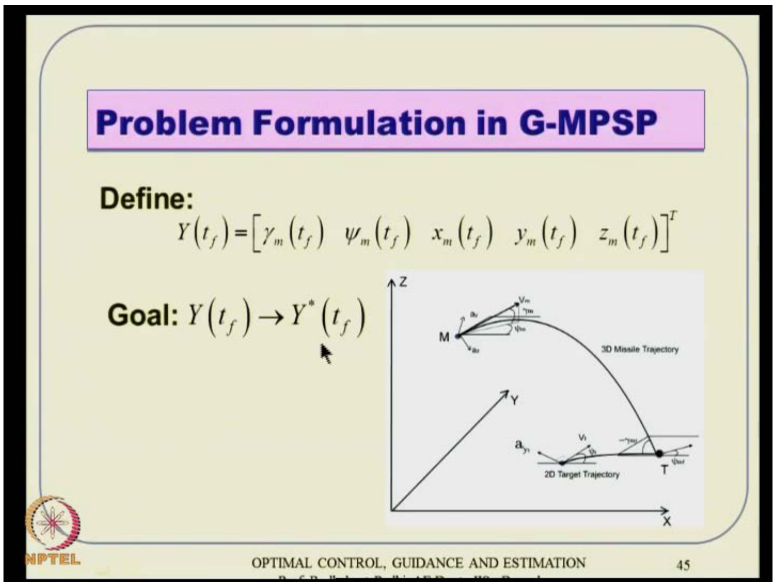
Assumptions:

- A point mass model is assumed
- Measurement of coordinates (x_t, y_t) are available.
- Target velocity V_t is constant (includes stationary targets too)
- Moving targets can do one of the followings:
  - No maneuver (straight line path)
  - Constant g maneuvers
  - Sinusoidal maneuvers

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 44

So, assumptions point mass model is assumed, measurement of coordinates are available, target velocity is constant and things like that actually. So, these are somewhat I mean, realistic assumptions, but this can always be relaxed with some sort of a estimation in the loop, which we are not talking here actually. If the target is moving, it can do the following. That means it can simply continue to move in a straight line. No constraint maneuver, I mean no maneuver. What it does is simply moves in a straight line. Otherwise, it can also have constant g maneuver. It keeps on turning in constant manners as well as it can have sinusoidal maneuvers as well actually and a combination of that kind is also a possibility. Here what you are telling is whatever the target does is kind of know to the guidance out of the missile actually.

(Refer Slide Time: 47:37)



Problem Formulation in G-MPSP

Define:

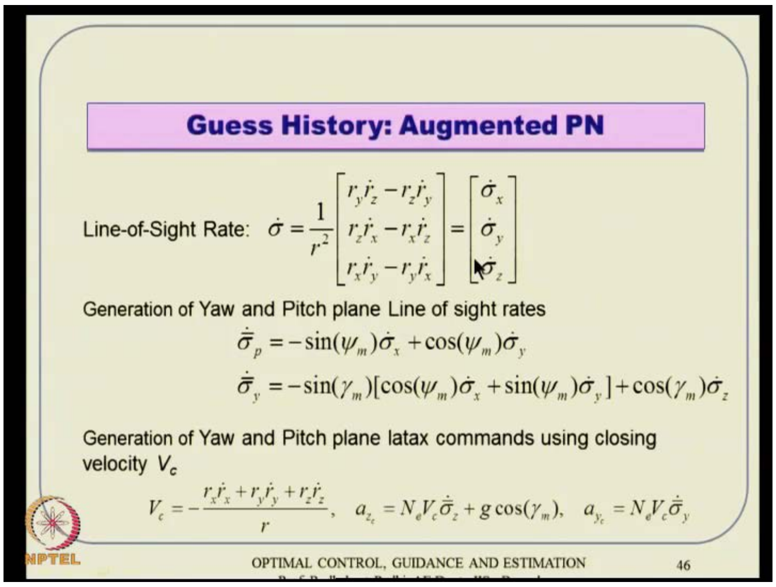
$$Y(t_f) = \left[\gamma_m(t_f) \quad \psi_m(t_f) \quad x_m(t_f) \quad y_m(t_f) \quad z_m(t_f)\right]^T$$

Goal: $Y(t_f) \rightarrow Y^*(t_f)$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 45

Typically, that turns out to be hard problem also and that is why estimations and logics are important and an estimation is supposed to give target estimation rather. Actually, suppose to give the required information to the guidance law actually, but anyway coming back to that, this is the problem formulation. We have this final Y, which is given in terms of gamma and psi and in terms of x y z. They have to go to some desired values by star basically ok.

(Refer Slide Time: 48:13)



Guess History: Augmented PN

Line-of-Sight Rate: $\dot{\sigma} = \frac{1}{r^2}\begin{bmatrix} r_y\dot{r}_z - r_z\dot{r}_y \\ r_z\dot{r}_x - r_x\dot{r}_z \\ r_x\dot{r}_y - r_y\dot{r}_x \end{bmatrix} = \begin{bmatrix} \dot{\sigma}_x \\ \dot{\sigma}_y \\ \dot{\sigma}_z \end{bmatrix}$

Generation of Yaw and Pitch plane Line of sight rates

$$\dot{\bar{\sigma}}_p = -\sin(\psi_m)\dot{\sigma}_x + \cos(\psi_m)\dot{\sigma}_y$$

$$\dot{\bar{\sigma}}_y = -\sin(\gamma_m)[\cos(\psi_m)\dot{\sigma}_x + \sin(\psi_m)\dot{\sigma}_y] + \cos(\gamma_m)\dot{\sigma}_z$$

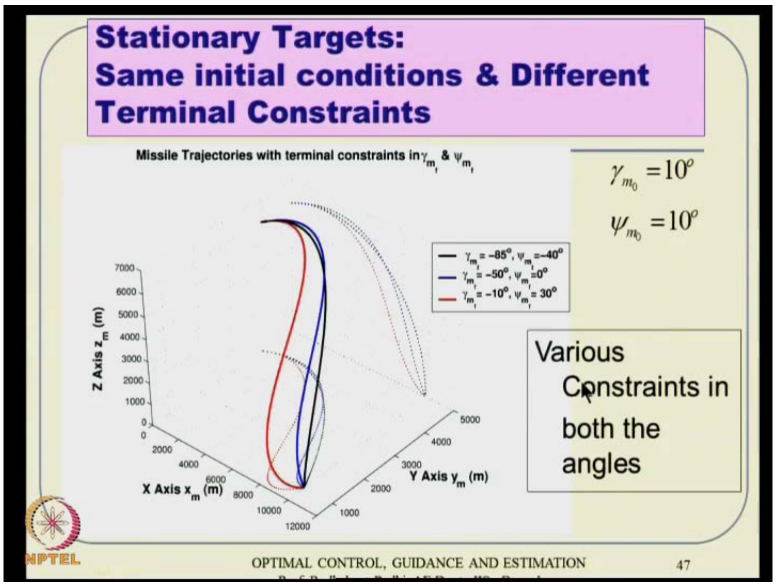
Generation of Yaw and Pitch plane latax commands using closing velocity V_c

$$V_c = -\frac{r_x\dot{r}_x + r_y\dot{r}_y + r_z\dot{r}_z}{r}, \quad a_{z_c} = N_e V_c \dot{\bar{\sigma}}_z + g\cos(\gamma_m), \quad a_{y_c} = N_e V_c \dot{\bar{\sigma}}_y$$

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 46

The guess history is augmented pm. The same thing is what you were shown in the last class. Sigma dot you compute first, then these 3 components sigma dot x, sigma dot y, sigma dot z. In turn using this sigma dot x, sigma dot y, you can compute sigma dot p and sigma dot y. So, this kind of thing and then, it turns out to be ok. You can compute this V c as well using this expression and then, my n z and n y, sorry a y can be computed something like this ok.

(Refer Slide Time: 48:48)



Stationary Targets: Same initial conditions & Different Terminal Constraints

$\gamma_{m_0} = 10^o$

$\psi_{m_0} = 10^o$

Various Constraints in both the angles

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 47

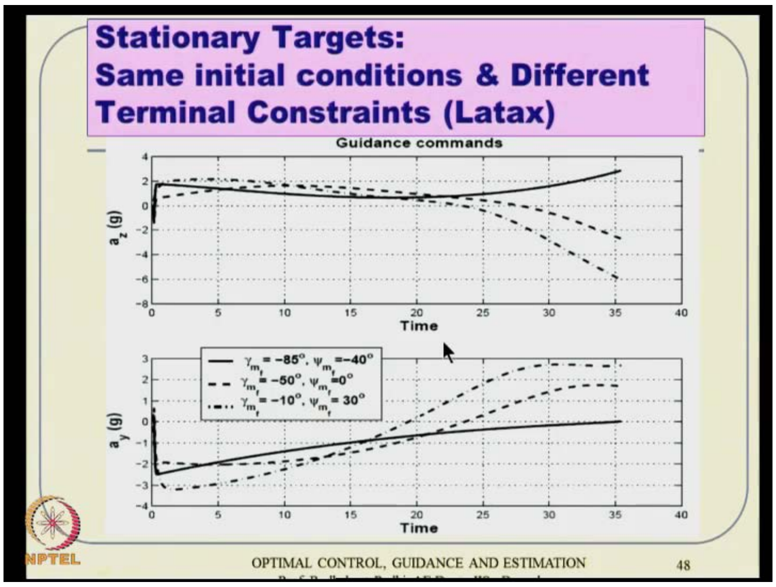
So, more is available in external literature basically. Then, we apply our newly developed continuous time MPSP or rather generalization MPSP actually. Then, we get very similar results what we have done even before; I mean so before in the lecture ok.

So, here we talk about initial condition being sent to angles, but 3 different cases are there where you derived values of gamma and psi are different combinations actually ok and this is just 3 cases that you are telling, but that is been validated with number of cases like this. It would be in some time in much more case, much more number of cases arbitrarily selected values like that actually, ok. Now, if you see that no matter whatever angle consistency put, you are able to go there, meet there actually.

Now, 3D sense it may not be so, easy to see, but that is why he has plotted these image plot and all that. One image plot is in terms of z and x that is image plane and one in terms of x and y that is image plane actually. That is also somewhat clear how the angles are developing, but if you really want particular clear value of representation and all that, you can always plot into d and c. Anyway, guidance commands tends out to be like this and you can see that the latax escalation demand is not really very much. So, about 6g, I mean within 6g here and something like within 3g here and here also if you see minus 3 point, probably 3.2g maximum value.

They are very much achievable in a good missile actually, so that the latax escalation demand turns out to be very much all right and also you can see that there are not too much of fluctuations on the way. It is just kind of (()) very smoothly. Initially, there is some sort of a small rapid development and later onwards it remains somewhat fairly smooth ok. So, this this kind of laterization is very good because inner loops can track this history very well actually ok. All right.

(Refer Slide Time: 49:52)

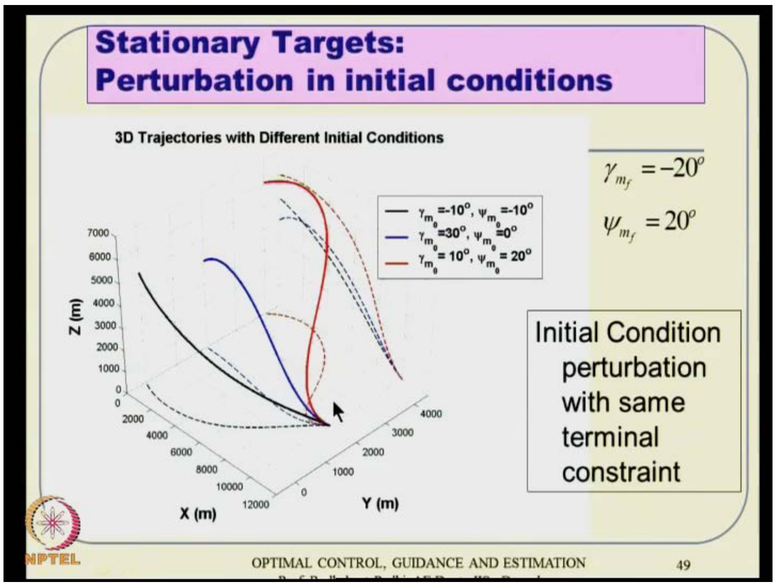


Stationary Targets: Same initial conditions & Different Terminal Constraints (Latax)

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 48

So, now, we can see from trajectories for a differential initial condition, but same final condition. Final condition is required same for the initial condition can be different. So, that is what happens. Again, it is able to meet its perturbation, whatever is our initial condition actually ok.

(Refer Slide Time: 51:05)



Stationary Targets: Perturbation in initial conditions

$\gamma_{m_f} = -20^\circ$

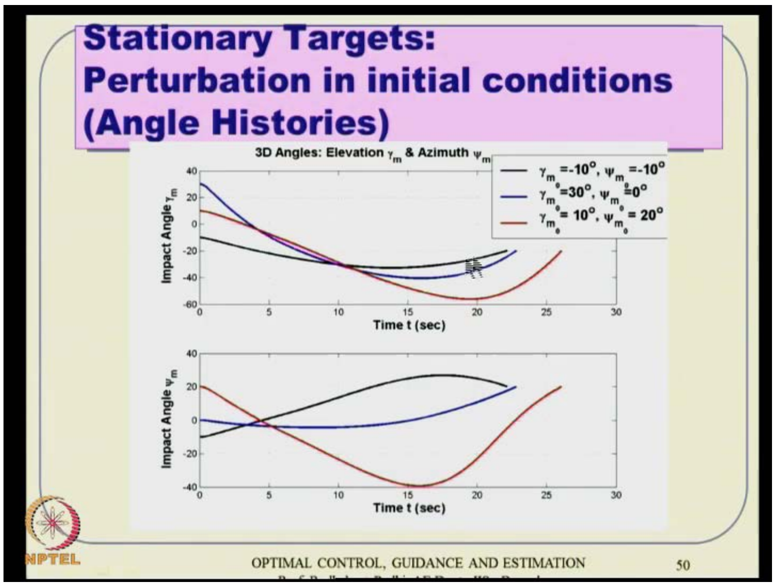
$\psi_{m_f} = 20^\circ$

Initial Condition perturbation with same terminal constraint

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 49

Now, if you see whether it is happening or not happening, you can actually plot these angle values as how they are developing.

(Refer Slide Time: 51:11)

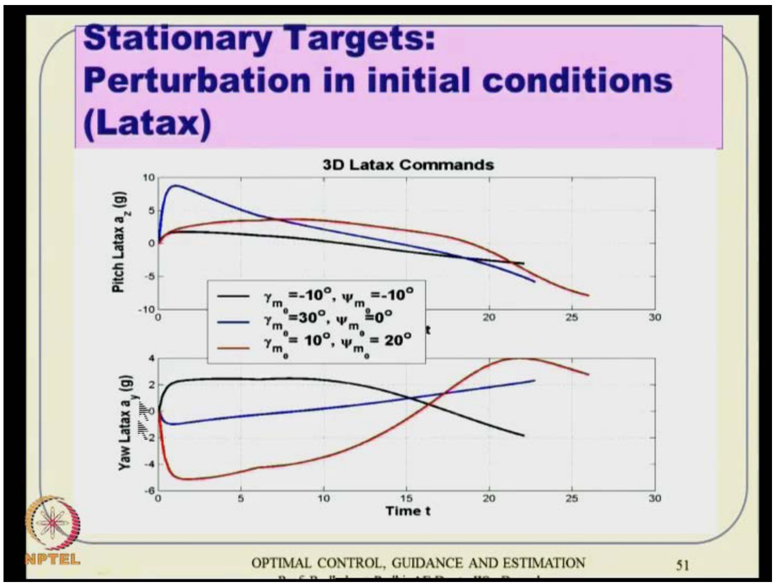


Stationary Targets: Perturbation in initial conditions (Angle Histories)

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 50

Well, at the end, all these are meeting at the same values minus 20 degree and all these values are meeting at the same, plus 20 degree and that is what we wanted, minus 20 and plus 20.

(Refer Slide Time: 51:37)



Stationary Targets: Perturbation in initial conditions (Latax)

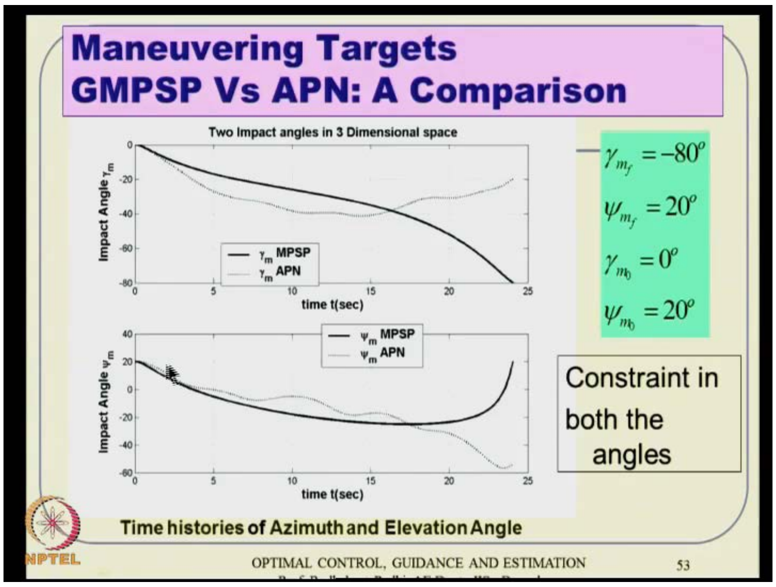
OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 51

So, that gives us validation of what is happening in a clear way basically. Now, if you take stationary targets again for perturbation of initial conditions, what happens is this is what we are talking about and what happens in terms of latax escalation. You can see as z n y again it is within some minus 9.5g sort of thing and here also, it is 4g maximum. May be 5g is here if you look at it actually. Remember these missiles are typically the latax escalation constraints are typically about of 10g, 15 g like that actually ok. So, these numbers what you see here, what you are looking at here is very much within the capability.

Now, how does it compare with respect to APN, the augmented PN. Remember, the augmented PN is the one who is searches the guess history irrespective of whatever conditions we want to put. That is kind of a deform guess actually. Then, you can ok if I take gamma naught in psi naught is 0, 20, but I want this particular values minus 80 and 20. Then, what is the, well how can it I mean how does it compare? We can see one is does not talk about angle constant. So, it goes it own way, but it finally (()) its position, but the other one not only meets it, but satisfies the angle constraints actually.

So, the trajectory circle happens like, I mean if I will show that the final constraints are met actually. We can again see these angles gamma m and psi m and one case, they are very far of what you decide, the solid plane and is what we what we ultimately want ok. This dotted line is what happens in APN. If there are some varying, it means if it is varying in its behaviour, ultimately the point is it is too far away from what we required. What we required is almost vertical impact. That means, minus a b d b with 20 degree smooth angle constraint ok.

(Refer Slide Time: 53:08)



Maneuvering Targets
GMPSP Vs APN: A Comparison

$\gamma_{m_f} = -80^o$

$\psi_{m_f} = 20^o$

$\gamma_{m_0} = 0^o$

$\psi_{m_0} = 20^o$

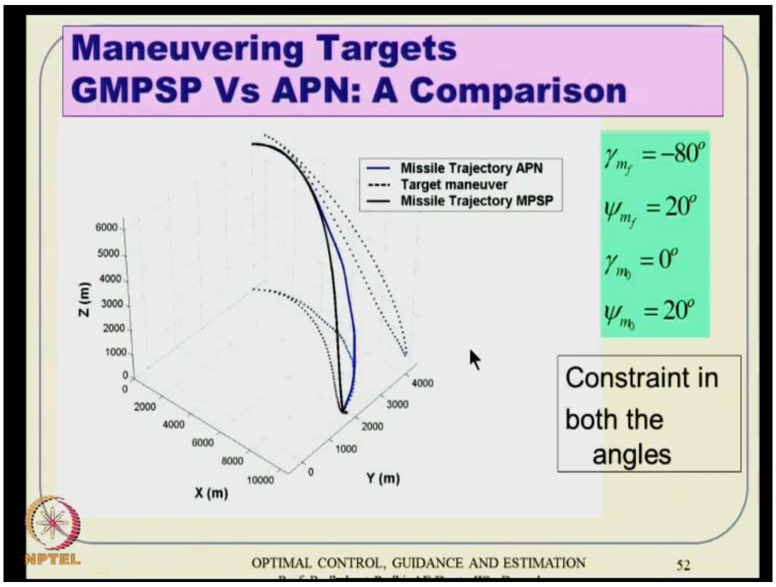
Constraint in both the angles

Time histories of Azimuth and Elevation Angle

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 53

So, this is what 20 degree you can see, the block diagram, the solid line ok. So, that is the difference. If it do not get it there, but what it get it there in a very smooth phase of thing.

(Refer Slide Time: 52:21)



Maneuvering Targets
GMPSP Vs APN: A Comparison

$\gamma_{m_f} = -80^o$

$\psi_{m_f} = 20^o$

$\gamma_{m_0} = 0^o$

$\psi_{m_0} = 20^o$

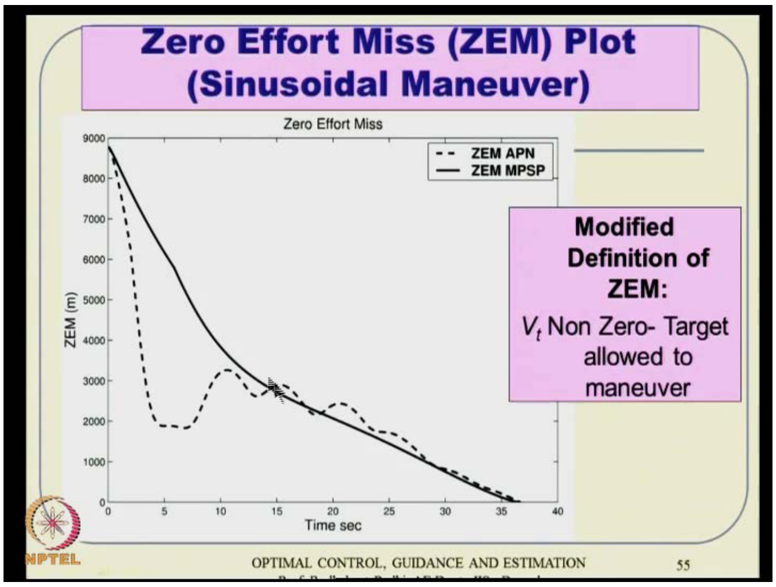
Constraint in both the angles

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 52

This is another plot which tells us that if the target is the straight line or a constant turning or sinusoidal behaviour, then can I do that and it said yes each of the phases we are able to enlarge with target, but also remember, here we are assuming that each of the behavior of the target is actually known from our good estimation logic which we are not discussing here ok.

So, any miss distance in this particular cases, we will turn out to be there is a large contribution from the estimation group, really is not with respect to the guidance loop actually ok. That validation has been solved here in this picture, all right. This is also you can see number wise. The final numbers are here. One case minus 20, other case minus 40, other case minus 60 and here, it will minus 20, some minus 30 and minus 40. Everything is met.

(Refer Slide Time: 54:51)



Zero Effort Miss (ZEM) Plot (Sinusoidal Maneuver)

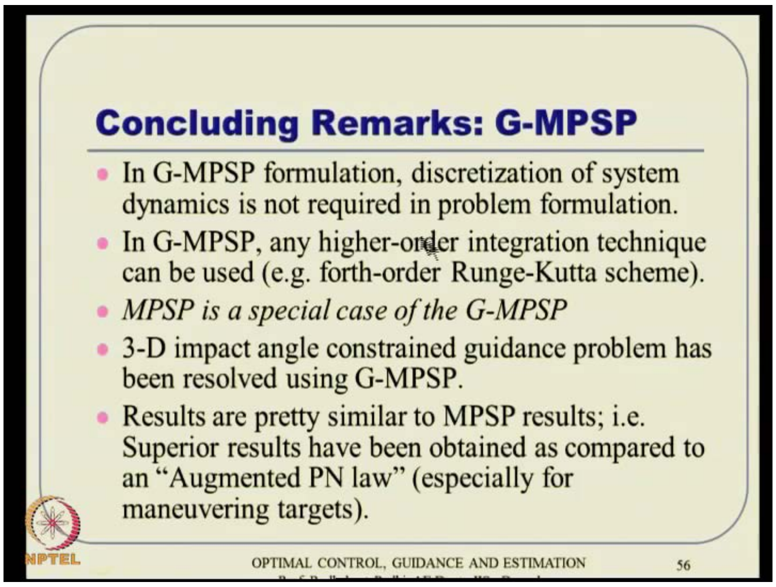
Modified Definition of ZEM:

V_t Non Zero- Target allowed to maneuver

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 55

Interestingly, there is another comparison which talks about comparison with something called 0 effort miss plot and we can see 0 effort miss for APN is very very ultimately goes to 0, but MPSC z m turns out to be much more smoother throughout actually. That is the good sign of a good guidance law.

(Refer Slide Time: 55:12)



Concluding Remarks: G-MPSP

- In G-MPSP formulation, discretization of system dynamics is not required in problem formulation.
- In G-MPSP, any higher-order integration technique can be used (e.g. forth-order Runge-Kutta scheme).
- *MPSP is a special case of the G-MPSP*
- 3-D impact angle constrained guidance problem has been resolved using G-MPSP.
- Results are pretty similar to MPSP results; i.e. Superior results have been obtained as compared to an "Augmented PN law" (especially for maneuvering targets).

OPTIMAL CONTROL, GUIDANCE AND ESTIMATION 56

So, concluding remarks of G-MPSP, this generalization G-MPSP formulation discretization of system dynamics is not required and any higher order technique can be used advance under thing can be about forth-order Runge-Kutta scheme. This is standard in integrating numerical, numerically the differential equation actually.

So, if you think about putting that, you can simply put it. If you want different integration scheme other than Runge-Kutta method, that also you are welcome to do that actually, and that turns out that MPSP is a special case of generalized MPSP and those things I am not going to discuss here. Then, we can apply these 2 or 3D impact angle guidance constrained guidance problem. It has been kind of solved using this technique actually. Results are pretty similar to MPSP results and sometimes some superior results also have been seen. Result is similar or superior because suddenly, the discretization are (()) from the beginning. So, accuracy level is very good from the beginning itself actually.

Then, sometime I mean law, why sometimes? Sometimes it is relative superior to MPSP, but many times, it is superior to PN guidance law and that is both for regular MPSP or G-MPSP actually because what you are telling here is, you are making use of these benefits of non-linear optimal control theory actually. That is why the results are much superior as compared to the augmented PN law, all right. So, that is what I wanted to discuss in this particular lecture. Thank you.