Notes on Mechatronic Systems

December 8, 2024

1 Lateral Control of a Car

1.1 The Bicycle Model

The car is moving forward at $V_c = 20\,\mathrm{m\,s^{-1}}$ in its longitudinal direction. The input to the system is the rotation of the front wheel δ in rad. The output of the system is the translation position X, Y in m, and the yaw angle ψ in rad in the world inertial frame.

Since the car is moving at a constant longitudinal velocity and changes in Y are small compared to X, we don't need to consider the X coordinate as a controlled output of the system.

The approach to model the vehicle is to consider it as a bicycle. The two rear wheels are merged together and the two front wheels are merged together. This eliminates the need to consider the two front wheels as separate inputs into the system. We just have the angle of the single merged front wheel δ.

The lateral direction y is to the left of the forward moving vehicle in its body frame. The force and moments in the lateral direction are,

$$F_{yr} + F_{yf} = ma_y \tag{1}$$

$$-F_{yr}l_r + F_{yf}l_f = J\ddot{\psi} \tag{2}$$

where F_{yr} is the laternal force applied by the rear wheel, F_{yf} is the lateral force applied by the front wheel, m is the mass of the vehicle, a_y is acceleration in the lateral direction, l_r is the distance between the center of mass and the rear wheel, l_f is the distance between the center of mass and the front wheel, J is the moment of inertia around the center of mass, and ψ is yaw.

The angular acceleration a_y is not only the change in the lateral velocity $\ddot{y}$, but it also consists of a component called the centripetal acceleration that causes a change in the longitudinal velocity. This is the product of longitudal velocity and the yaw rate,

$$a_y = \ddot{y} + \dot{x}\dot{\psi} \tag{3}$$

The centripetal motion appears in these equations since the equations are set in the non-inertial body frame and it accounts for the rotation of the body frame in the inertial frame. Therefore the equations of motion become,

$$F_{yr} + F_{yf} = m(\ddot{y} + \dot{x}\dot{\psi}) \tag{4}$$

$$-F_{yr}l_r + F_{yf}l_f = J\ddot{\psi} \tag{5}$$

The next concept for modeling the car are the slip angles of the tires. At high speeds the tire's direction and angle of velocity won't be the same. The angle between the tire's velocity vector and the axis of the body are θ_{vf} and θ_{vr} for the front and rear wheels respectively. The slip angles of the tires α_f and α_r are the angles between the direction of the tire and their velocity vector. The relationship between the angles of the tires are,

$$\alpha_f = \delta - \theta_{vr} \tag{6}$$

$$\alpha_r = -\theta_{vr} \tag{7}$$

It has been experimentally determined for small slip angles, the lateral forces are proportional to the slip angles, that is,

$$F_{yf} = 2C_{\alpha f}\alpha_f \tag{8}$$
$$F_{yr} = 2C_{\alpha r}\alpha_r \tag{9}$$

where the constants of proportionality are called the cornering stiffnesses with units $\mathrm{N\,rad^{-1}}$. The factor of 2 exists because we have combined two wheels into one for the bicycle model. The lateral forces can then be expressed in terms of the δ, θ_{vf}, and θ_{vr} as,

$$F_{yf} = 2C_{\alpha f}(\delta - \theta_{vf}) \tag{10}$$
$$F_{yr} = 2C_{\alpha r}(-\theta_{vr}) \tag{11}$$

We need to express the velocity angles θ_{vf} and θ_{vr} in terms of model constants and states. The expressions for these angles are,

$$\tan(\theta_{vf}) = \frac{\dot{y} + l_f\dot{\psi}}{\dot{x}} \tag{12}$$
$$\tan(\theta_{vf}) = \frac{\dot{y} + l_r\dot{\psi}}{\dot{x}} \tag{13}$$

We use a small angle approximation to simplify these equations, that is $\tan(x) \approx x$, when employing these angles in the state space model of the system.

By combining the equations for the state space model from input δ to the states $(\dot{y}, \dot{\psi})$, the elements of the state and input matrices of the second order system are:

$$a_{11} = -\frac{2C_{\alpha f} + 2C_{\alpha r}}{m\dot{x}} \qquad a_{12} = -\dot{x} - \frac{2C_{\alpha f}l_f - 2C_{\alpha r}l_r}{m\dot{x}} \tag{14}$$
$$a_{21} = \frac{-2C_{\alpha f}l_f + 2C_{\alpha r}l_r}{J\dot{x}} \qquad a_{22} = \frac{-2C_{\alpha f}l_f^2 - 2C_{\alpha r}l_r^2}{J\dot{x}} \tag{15}$$
$$b_1 = \frac{2C_{\alpha f}}{m} \qquad b_2 = \frac{2C_{\alpha f}l_f}{J} \tag{16}$$

Then furthermore we have two additional states (Y, ψ) which extends the state space equations to a fourth order system. Note that Y is the y-coordinate in the inerital frame of reference. Using the inertial frame of reference introduces a non-linearity. Given the states $x = (\dot{y}, \psi, \dot{\psi}, Y)$, and the input $u = \delta$, the non-linear state space equations are:

$$\ddot{y} = a_{11}\dot{y} + a_{12}\dot{\psi} + b_1\delta \tag{17}$$
$$\dot{\psi} = \dot{\psi} \tag{18}$$
$$\ddot{\psi} = a_{21}\dot{y} + a_{22}\dot{\psi} + b_2\delta \tag{19}$$
$$\dot{Y} = \dot{y}\cos(\psi) + \dot{x}\sin(\psi) \tag{20}$$

If have small yaw angles, we can linearize the last equation as $\dot{Y} = \dot{y} + \dot{x}\psi$.

1.2 Discretizing & Augmenting the Bicycle Model

We are going to design an MPC controller for the linearized bicycle model given by,

$$\dot{x} = A_c x + B_c u \tag{21}$$

where the state $x = (\dot{y}, \psi, \dot{\psi}, Y)$ is $u = \delta$, and the continuous time state space matrices are,

$$A_c = \begin{bmatrix} a_{11} & 0 & a_{12} & 0 \\ 0 & 0 & 1 & 0 \\ a_{21} & 0 & a_{22} & 0 \\ 1 & \dot{x} & 0 & 0 \end{bmatrix} \qquad B_c = \begin{bmatrix} b_1 \\ 0 \\ b_2 \\ 0 \end{bmatrix} \tag{22}$$

MPC is designed to be applied to a discrete time model. We use a method called zero-order hold to discretize the model with sampling rate T_s,

$$A_d = e^{A_c T_s} \qquad B_d = \int_0^{T_s} e^{A_c \tau} B_c \, d\tau \tag{23}$$

where A_d and B_d are the discrete time state and input matrices fo the model,

$$x_{k+1} = A_d x_k + B_d u_k \tag{24}$$

The next step into preparing our model for the application of MPC is that we want to change the input from the wheel angle, to the change in wheel angle. Therefore we augment the system to make the wheel angle a state and the input becomes the change of wheel angle. The discrete time state of the augmented system is $x = (\dot{y}, \psi, \dot{\psi}, Y, \delta)$, the input is $u = \Delta\delta$, and the discrete time state and input matrices are:

$$A_a = \begin{bmatrix} A_d & B_d \\ 0 & I \end{bmatrix} \qquad B_a = \begin{bmatrix} B_d \\ I \end{bmatrix} \tag{25}$$

With the augmented model, we are able to design a MPC control law following the analysis in the next section.

1.3 Solving MPC for Unconstrained LTI Systems

Given the discrete time system,

$$x_{k+1} = Ax_k + Bu_k \tag{26}$$
$$y_k = Cx_k + Du_k \tag{27}$$

the kth state is computed as,

$$x_k = A^k x_0 + \begin{bmatrix} A^{k-1}B & A^{k-2}B & \dots & AB & B \end{bmatrix} \begin{bmatrix} u_0 & u_1 & \dots & u_{k-2} & u_{k-1} \end{bmatrix}^T \tag{28}$$

Let's define a global state vector and global input vector along a finite horizon of N steps as:

$$x_G = \begin{bmatrix} x_1^T & x_2^T & \dots & x_N^T \end{bmatrix}^T \tag{29}$$
$$u_G = \begin{bmatrix} u_0^T & u_1^T & \dots & u_{N-1}^T \end{bmatrix}^T \tag{30}$$

Then we can compute this global state vector from the initial state and the global input vector as,

$$x_G = \overline{C} u_G + \overline{A} x_0 \tag{31}$$

where the two system matrices are,

$$\overline{C} = \begin{bmatrix} B & 0 & \dots & 0 & 0 \\ AB & B & \dots & 0 & 0 \\ \vdots & \vdots & & & \\ A^{N-1}B & A^{N-2}B & \dots & AB & B \end{bmatrix} \qquad \overline{A} = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix} \tag{32}$$

The cost function of MPC to minimze is,

$$J = \tfrac{1}{2} e_N^T S e_N + \tfrac{1}{2} \sum_{i=0}^{N-1} \left[e_i^T Q e_i + u_i^T R u_i \right] \tag{33}$$

This basically states that we should control the system to minimize the weighted sum of the errors and control actions, with a dedicated weighting given to the error at the end of the finite horizon. We aim to find an input vector $[u_0, \dots, u_{N-1}]$ that minimizes this cost function. The weighting matrices S, Q, and R, are used to tune the performance of the control system. To derive a solution to this equation, the first thing we do is substitute in the error term,

$$e_k = r_k - \tilde{C} x_k \tag{34}$$

into the cost function. $\tilde{C}$ transforms the state into a signal that we are interested to control. Expanding the cost function gives,

$$J = \tfrac{1}{2}\left(r_N - \tilde{C}x_N\right)^T S\left(r_N - \tilde{C}x_N\right)^T + \tfrac{1}{2}\sum_{i=0}^{N-1}\left[\left(r_i - \tilde{C}x_i\right)^T Q\left(r_i - \tilde{C}x_i\right)^T + u_i^T R u_i\right] \tag{35}$$

$$= \tfrac{1}{2}\left[r_N^T S r_N - r_N^T S\tilde{C}x_N - x_N^T\tilde{C}^T S r_N + x_N^T\tilde{C}^T S\tilde{C}x_N\right] + \tfrac{1}{2}\sum_{i=0}^{N-1}\left[r_i^T Q r_i - r_i^T Q\tilde{C}x_i - x_i^T\tilde{C}^T Q r_i + x_i^T\tilde{C}^T Q\tilde{C}x_i + u_i^T R u_i\right] \tag{36}$$

We can combine the quadratic terms that are equal (the terms that contain both state and reference vectors can be paired and summed due to their equality, see Equations (38) to (40)). The cost function simplifies to,

$$J = \tfrac{1}{2}\left[r_N^T S r_N - 2r_N^T S\tilde{C}x_N + x_N^T\tilde{C}^T S\tilde{C}x_N\right] + \tfrac{1}{2}\sum_{i=0}^{N-1}\left[r_i^T Q r_i - 2r_i^T Q\tilde{C}x_i + x_i^T\tilde{C}^T Q\tilde{C}x_i + u_i^T R u_i\right] \tag{37}$$

Cost Function Terms with r_i and x_i are Equal.

$$\frac{1}{2}r_i^T Q\tilde{C}x_i = \frac{1}{2}\left(x_i^T\tilde{C}^T Q^T r_i\right)^T \qquad \text{From properties of tranpose matrices.} \tag{38}$$

$$= \frac{1}{2}\left(x_i^T\tilde{C}^T Q r_i\right)^T \qquad \text{Q is a symmetric matrix.} \tag{39}$$

$$= \frac{1}{2}x_i^T\tilde{C}^T Q r_i \qquad \text{LHS is scalar thus symmetric.} \tag{40}$$

Since this is a cost function that we need to minimize, all constant terms have no effect on the solution. So we are going to remove constant terms to derive a simpler cost function with the same solution. Any terms that depend only on the reference r_k or the initial state x_0 are constant. The new cost function is,

$$J' = \tfrac{1}{2}\left[-2r_N^T S\tilde{C}x_N + x_N^T\tilde{C}^T S\tilde{C}x_N\right] + \tfrac{1}{2}\sum_{i=1}^{N-1}\left[-2r_i^T Q\tilde{C}x_i + x_i^T\tilde{C}^T Q\tilde{C}x_i + u_i^T R u_i\right] + \tfrac{1}{2}u_0^T R u_0 \tag{41}$$

Then we are going to move the cost function expression above into a matrix form. In the same way we have previously defined a global state vector and global input vector in Equations (29) to (30) we will also define a global reference vector as,

$$r_G = \begin{bmatrix} r_1^T & r_2^T & \dots & r_N^T \end{bmatrix}^T \tag{42}$$

Then the cost function can be rewritten in a matrix form as,

$$J' = \tfrac{1}{2}x_G^T\overline{Q}x_G - r_G^T\overline{T}x_G + \tfrac{1}{2}u_G^T\overline{R}u_G \tag{43}$$

where $\overline{Q}$, $\overline{T}$, and $\overline{R}$ are block diagonal matrices given as,

$$\overline{Q} = \begin{bmatrix} \tilde{C}^T Q\tilde{C} & & & & \\ & \tilde{C}^T Q\tilde{C} & & & \\ & & \ddots & & \\ & & & \tilde{C}^T Q\tilde{C} & \\ & & & & \tilde{C}^T S\tilde{C} \end{bmatrix} \quad \overline{T} = \begin{bmatrix} Q\tilde{C} & & & \\ & Q\tilde{C} & & \\ & & \ddots & \\ & & & S\tilde{C} \end{bmatrix} \quad \overline{R} = \begin{bmatrix} R & & & \\ & R & & \\ & & \ddots & \\ & & & R \end{bmatrix} \tag{44}$$

Then we substitute the expression for the global state vector in Equation (31) into the cost function,

$$J' = \tfrac{1}{2}\left(u_G^T\overline{C}^T + x_0^T\overline{A}^T\right)\overline{Q}\left(\overline{C}u_G + \overline{A}x_0\right) - r_G\overline{T}\left(\overline{C}u_G + \overline{A}x_0\right) + \frac{1}{2}u_G^T\overline{R}u_G \tag{45}$$

$$= \tfrac{1}{2}u_G^T\overline{C}^T\overline{QC}u_G + \tfrac{1}{2}x_0^T\overline{A}^T\overline{QC}u_G + \tfrac{1}{2}u_g^T\overline{C}^T\overline{QA}x_0 + \tfrac{1}{2}x_0^T\overline{A}^T\overline{QA}x_0 - r_G^T\overline{TC}u_G - r_g^T\overline{TA}x_0 + \frac{1}{2}u_G^T\overline{R}u_G \tag{46}$$

$$= \tfrac{1}{2}u_G^T\overline{C}^T\overline{QC}u_G + x_0^T\overline{A}^T\overline{QC}u_G + \tfrac{1}{2}x_0^T\overline{A}^T\overline{QA}x_0 - r_G^T\overline{TC}u_G - r_g^T\overline{TA}x_0 + \frac{1}{2}u_G^T\overline{R}u_G \tag{47}$$

and again, since there are constant terms in this cost function (those terms that are only a function of the reference and initial state), we can create a simplifed cost function with the same solution by removing constant terms,

$$J'' = \tfrac{1}{2}u_G^T\overline{C}^T\overline{QC}u_G + x_0^T\overline{A}^T\overline{QC}u_G - r_G^T\overline{TC}u_G + \tfrac{1}{2}u_G^T\overline{R}u_G \tag{48}$$

$$= \tfrac{1}{2}u_G^T\left(\overline{C}^T\overline{QC} + \overline{R}\right)u_G + \begin{bmatrix} x_0^T & r_G^T \end{bmatrix}\begin{bmatrix} \overline{A}^T\overline{QC} \\ -\overline{TC} \end{bmatrix}u_G \tag{49}$$

$$= \tfrac{1}{2}u_G^T\overline{H}u_G + \begin{bmatrix} x_0^T & r_G^T \end{bmatrix}\overline{F}^T u_G \tag{50}$$

Note that this is now a quadratic function with respect to the control input u_G which is what we want to find. Taking the the gradient of this function (see Equations (52) to (53)) and equating it to zero gives the control law,

$$u_G = -\overline{H}^{-1}\overline{F}\begin{bmatrix} x_0 \\ r_G \end{bmatrix} \tag{51}$$

Typically we only apply the first control action from the global input vector before recomputing it to account for uncertainties in the system model and disturbances.

Gradient of Quadratic Functions

Given the following quadratic function where A is symmetric (ie. $A = A^T$),

$$y = \frac{1}{2}x^T Ax + B^T x \tag{52}$$

the gradient of y w.r.t. x is,

$$\nabla\, y = Au + B \tag{53}$$

2 Automonomous Car

2.1 The Equations of Motion

First we will derive all the forces and moments on the vehicle in the body frame. The forces are due to the force from the turning front wheel, the engine, the rolling drag, and the slip of the rear wheel. They are expressed as,

$$F_x^B = F_a - F_r - F_{yf}sin(\delta) \tag{54}$$

$$= ma - \mu mg - F_{yf}sin(\delta) \tag{55}$$

$$F_y^B = F_{yr} + F_{yf}cos(\delta) \tag{56}$$

$$M_z^B = F_{yf}cos(\delta)l_f - F_{yr}l_r \tag{57}$$

where (F_x^B, F_x^B, M_z^B) are the net forces and moments in body frame, F_a is the applied force, F_r is the rolling resistance force, m is the mass of the car, a is applied logitudinal acceleration, μ is the rolling resistance coefficient, g is acceleration due to gravity, F_{yr} is the lateral force applied by the front wheel, δ is the angle of the front wheel, F_{yr} is the lateral force applied by the rear wheel, l_f is the distance between the front wheel and center of mass, and l_r is the distance between the rear wheel and the center of mass.

The lateral forces on front and rear wheels are,

$$F_{yf} = C_{\alpha f}\alpha_f = C_{\alpha f}(\delta - \theta_{vf}) \tag{58}$$

$$F_{yr} = C_{\alpha r}\alpha_r = C_{\alpha r}(-\theta_{vr}) \tag{59}$$

where $C_{\alpha f}$ and $C_{\alpha r}$ are the cornering stiffnesses of the front and rear wheels. α_f and α_r are the slip angles and is the difference between velocity of the tire and the angle of the tire. Tire angles are,

$$\tan(\theta_{vf}) = \frac{\dot{y}}{\dot{x}} + \frac{\psi l_f}{\dot{x}} \tag{60}$$

$$\tan(\theta_{vr}) = \frac{\dot{y}}{\dot{x}} - \frac{\psi l_r}{\dot{x}} \tag{61}$$

And we apply a small angle assumption ($tan(x) = x$).

Newton's laws in a non-inertial frame for reference.

Let's say we have a non-inerial frame of reference in 2D space. The unit axes of the frame of reference are (i, j). This frame of reference is the body frame of a mass of m kg denoted by B. We have a force applied to the body F^B which can be decomposed along the axes of the body frame. Then we apply Newton's second law.

$$F^B = F_x^B i + F_y^B j \tag{62}$$
$$= m(a_x i + a_y j) \tag{63}$$
$$= m(\frac{d(\dot{x}i)}{dt} + \frac{d(\dot{y}j)}{dt}) \tag{64}$$
$$= m(\ddot{x}i + \dot{x}\dot{i} + \ddot{y}j + \dot{y}\dot{j}) \tag{65}$$

Since it is a non-inertial frame of referece, the change in the unit vectors $(\dot{i}, \dot{j})$ are non-zero. In 2D space, the change in the body frame occurs due to rotation around the z-axis. Let's consider the rotation vector $\omega = [0, 0, \dot{\psi}]$ around the z-axis coming out of the 2D plane. Then the expression from above becomes,

$$= m(\ddot{x}i + \ddot{y}j + \dot{x}(\omega \times i) + \dot{y}(\omega \times j)) \tag{66}$$
$$= m(\ddot{x}i + \ddot{y}j + \omega \times (\dot{x}i + \dot{y}j)) \tag{67}$$

Then we group based on axes,

$$\begin{bmatrix} F_x^B \\ F_y^B \end{bmatrix} = m\left(\begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \times \begin{bmatrix} \dot{x} \\ \dot{y} \\ 0 \end{bmatrix}\right) \tag{68}$$
$$= m\begin{bmatrix} \ddot{x} - \dot{\psi}\dot{y} \\ \ddot{y} + \dot{\psi}\dot{x} \end{bmatrix} \tag{69}$$

This gives the accelerations in the body frame,

$$a_x = \ddot{x} - \dot{\psi}\dot{y} \tag{70}$$
$$a_y = \ddot{y} + \dot{\psi}\dot{x} \tag{71}$$
$$\tag{72}$$

We get a similar expression for moments. For a 2D frame of reference, we only rotate in the the z-axis denoted with unit vector k. For an body with moment interia I, the net angular acceleration is found with,

$$M^B = M_z^B k = I\frac{d(\dot{\psi}k)}{dt} = I\left(\ddot{\psi}k + \dot{\psi}\dot{k}\right) \tag{73}$$

Since the unit vector k doesn't change for a 2D frame of reference, $\dot{k} = 0$. Thefore the angular acceleration due to the net momement is $\ddot{\psi}$.

Then the laws of motion for the vehicle are,

$$F_x^B = m(\ddot{x} - \dot{\psi}) = ma_x \tag{74}$$
$$F_y^B = m(\ddot{y} + \dot{\psi}) = ma_y \tag{75}$$
$$M_z^B = I\ddot{\psi} \tag{76}$$

Finally we need to convert these equations in a state space form. We add two additional states since we are interested in the position in the inertial frame of reference. The input to the system is both the acceleration a and the steering angle

δ. The equations are,

$$\ddot{x} = a - \frac{F_{yf} sin(\delta)}{m} - \mu g + \dot{\psi}\dot{y} \tag{77}$$

$$\ddot{y} = \frac{F_{yr}}{m} + \frac{F_{yf} cos(\delta)}{m} - \dot{\psi}\dot{x} \tag{78}$$

$$\dot{\psi} = \dot{\psi} \tag{79}$$

$$\ddot{\psi} = \frac{F_{yf} cos(\delta) l_f}{I} - \frac{F_{yr} l_r}{I} \tag{80}$$

$$\dot{X} = \dot{x} cos(\psi) - \dot{y} sin(\psi) \tag{81}$$

$$\dot{Y} = \dot{x} sin(\psi) + \dot{y} cos(\psi) \tag{82}$$

2.2 Unconstrained Linear Parameter Varying MPC

The linear parameter varying (LPV) form of the non-linear model is,

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \dot{\psi} \\ \ddot{\psi} \\ \dot{X} \\ \dot{Y} \end{bmatrix} = \begin{bmatrix} \frac{-\mu g}{\dot{x}} & \frac{C_{\alpha f} \sin(\delta)}{m\dot{x}} & 0 & \left(\frac{C_{\alpha f} \sin(\delta) l_f}{m\dot{x}} + \dot{y}\right) & 0 & 0 \\ 0 & \frac{-C_{\alpha r} + C_{\alpha f} \cos(\delta)}{m\dot{x}} & 0 & \left(-\frac{(C_{\alpha f} \cos(\delta) l_f - C_{\alpha r} l_r)}{m\dot{x}} - \dot{x}\right) & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & \left(-\frac{(C_{\alpha f} \cos(\delta) l_f - C_{\alpha r} l_r)}{I_z \dot{x}}\right) & 0 & \left(-\frac{(C_{\alpha f} \cos(\delta) l_f^2 - C_{\alpha r} l_r^2)}{I_z \dot{x}}\right) & 0 & 0 \\ \cos(\psi) & -\sin(\psi) & 0 & 0 & 0 & 0 \\ \sin(\psi) & \cos(\psi) & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \psi \\ \dot{\psi} \\ X \\ Y \end{bmatrix} + \begin{bmatrix} \frac{-C_{\alpha f} sin(\delta)}{m} & 1 \\ \frac{C_{\alpha f} cos(\delta)}{m} & 0 \\ 0 & 0 \\ \frac{C_{\alpha f} cos(\delta) l_f}{m} & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta \\ a \end{bmatrix} \tag{83}$$

The planner which produces the trajectory for the vehicle to follow not only generates the desired X, Y, ψ, we also control the lateral velocity $\dot{x}$. Given a trajectory $t -> (X, Y)$, we can find the tangent of the trajectory to compute ψ. And for the lateral velocity we differentiate (X, Y) and rotate the velocities into the body frame using the reference ψ. Therefore the output matrix C is,

$$\begin{bmatrix} \dot{x} \\ \psi \\ X \\ Y \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \psi \\ \dot{\psi} \\ X \\ Y \end{bmatrix} \tag{84}$$

We want to change the inputs to the system to be the rate of change of δ and a. We follow the same process as described in Section 1.2. The augmented system make the inputs $\Delta\delta$ and δa. Given the discrete time state space matrices A_d, B_d, C_d (assume $D_d = 0$), the augmented system used in the MPC algorithm has the following system matrices,

$$A_a = \begin{bmatrix} A_d & B_d \\ 0 & I \end{bmatrix} \qquad B_a = \begin{bmatrix} B_d \\ I \end{bmatrix} \qquad C_a = \begin{bmatrix} C_d & 0 \end{bmatrix} \tag{85}$$

To execute MPC for a LPV model we substitute the current state into the equations above to derive a LTI model. Then we derive the MPC cost function matrices in the same manner as the unconstrained LTI described in Section 1.3. To summarize the cost function from Equation (50) is,

$$J'' = \frac{1}{2} u_G^T \overline{H} u_G + \begin{bmatrix} x_0^T & r_G^T \end{bmatrix} \overline{F}^T u_G \tag{86}$$

where u_G is the global input vector that we want to find, x_0 is the initial state, r_G is the global reference vector, $\overline{H}$ and $\overline{F}$ are matrices derived in Section 1.3. As in the unconstrained case this cost function can be solved analytically for u_G and the control action applied to the system. For each control iteration, we need to convert the LPV to LTI each time as the state changes and recompute the MPC cost function to solve it for u_G.

Note that the cost function assumes the state changes is small over the time horizon and thus the state space matrices are constant for the MPC derivations. This allows us to use the LTI formulation to dervie the cost function matrices. If the state was expected to change significantly during the MPC prediction, we should iteraively calculate new state space matrices for each time step over the MPC horizon.

2.3 Constrained Linear Parameter Varying MPC

When we place constraints on the inputs or states, we no longer can derive an analytical solution. We must use a quadratic program (QP) solver instead. A solver requires us to provide details of the cost function and constraints and it will compute a solutions.

Using QPSolver in Python

The QP solver https://qpsolvers.github.io/qpsolvers/quadratic-programming.html requires the QP in the form,

$$\min \frac{1}{2} x^T P x + q^T x \tag{87}$$

$$\text{s.t.} \quad Gx \leq h \tag{88}$$

Therefore we need to supply the matrices P, q, G, h to solve the MPC problem.

The cost function for the unconstrained and constrained problem is the same so when using QPSolver $P = \overline{H}$ and $q = \overline{F}\left[x_o^T \; r_G^T\right]^T$. We need to derive the constraint matricies. We will start with the constraints on the states. We specify upper and lower bounds on a transformed version of the states. That is,

$$x_l \leq C_c x \leq x_u \tag{89}$$

The constraint selection matrix C_c is used to select the states we want to apply constraints to and consists of 0's and 1's. The first thing to note is that the MPC problem uses the global state vector x_G rather than just a single state vector so we expand the constraints as,

$$\overline{x}_l \leq \overline{C}_c x_G \leq \overline{x}_u \tag{90}$$

where,

$$\overline{x}_l = \begin{bmatrix} x_l \\ \vdots \\ x_l \end{bmatrix} \qquad \overline{C}_c = \begin{bmatrix} C_c & & \\ & \ddots & \\ & & C_c \end{bmatrix} \qquad \overline{x}_u = \begin{bmatrix} x_u \\ \vdots \\ x_u \end{bmatrix} \tag{91}$$

Furthermore we need to transform the global state vector to the global input vector as that is what we are solving for. We substitute in Equation (31) and rearrange,

$$\overline{x}_l \leq \overline{C}_c \left(\overline{C} u_G + \overline{A} x_0\right) \leq \overline{x}_u \tag{92}$$

$$\overline{x}_l - \overline{C}_C \overline{A} x_0 \leq \overline{C}_c \overline{C} u_G \leq \overline{x}_u - \overline{C}_C \overline{A} x_0 \tag{93}$$

Next we add in explicit constraints on the gloabl state vector $\overline{u}_l$ and $\overline{u}_u$ and then rearrange to put the constraints in the form required for the QP solver,

$$\begin{bmatrix} \overline{C}_C \overline{C} \\ -\overline{C}_C \overline{C} \\ I \\ -I \end{bmatrix} u_G \leq \begin{bmatrix} \overline{x}_u - \overline{C}_C \overline{A} x_0 \\ -\overline{x}_l + \overline{C}_C \overline{A} x_0 \\ \overline{u}_u \\ -\overline{u}_l \end{bmatrix} \tag{94}$$

The LHS matrix is the G required by QPSolver, and the RHS matrix is the h required by QPSolver. Thus all required information is available to solve for an iteration of the MPC algorithm for u_G.

As an externsion you can have time varying constraint to, for example, make sure one state is always smaller than another.

3 References

https://engineeringmedia.com/controlblog/the-kalman-filter

https://www.youtube.com/playlist?list=PLMLojHoA_QPmRiPotD_TnfdUkglTexuqm