Scientific Journal of Astana IT University
ISSN (P): 2707-9031 ISSN (E): 2707-904X
VOLUME 11, SEPTEMBER 2022

DOI: 10.37943/AEPO1273

A. Khaimuldin
Master of Technical Sciences, Senior-Lecturer of Computer Engineering Department
askar.khaimuldin@astanait.edu.kz, orcid.org/0000-0002-2070-0886
Astana IT University, Kazakhstan

T. Mukatayev
Master of Technical Sciences, Senior-Lecturer of Computer Engineering Department
tleuzhan.mukatayev@astanait.edu.kz, orcid.org/0000-0003-0818-2825
Astana IT University, Kazakhstan

N. Assanova
Master of Technical Sciences, Senior-Lecturer of Computer Engineering Department
nurgul.assanova@astanait.edu.kz, orcid.org/0000-0002-2179-6103
Astana IT University, Kazakhstan

N. Khaimuldin
Master of Technical Sciences, Senior-Lecturer of Computer Engineering Department
n.khaimuldin@astanait.edu.kz, orcid.org/0000-0002-3767-4418
Astana IT University, Kazakhstan

S. Alshynov
Master of Technical Sciences, Senior-Lecturer of Computer Engineering Department
shyngys.alshynov@astanait.edu.kz, orcid.org/0000-0003-2646-6696
Astana IT University, Kazakhstan

TRACKING OF NON-STANDARD TRAJECTORIES USING MPC METHODS WITH CONSTRAINTS HANDLING ALGORITHM

Abstract: In recent decades, a Model-Based Predictive Control (MPC) has revealed its dominance over other control methods such as having an ability of constraints handling and input optimization in terms of the value function. However, the complexity of the realization of the MPC algorithm on real mechatronic systems remains one of the major challenges. Traditional predictive control approaches are based on zero regulation or a step change. Nevertheless, more complicated systems still exist that need to track setpoint trajectories.

Currently, there is an active development of robotics and the creation of transport networks of movement without human participation. Therefore, the issue of programming the given trajectories of vehicles is relevant. In this article, authors reveal the alternative solution for tracking non-standard trajectories in spheres such as robotics, IT in mechatronics, etc., that could be used in self-driving cars, drones, rockets, robot arms and any other automized systems in factories.

The ability of Model-Based Predictive Control (MPC) such as the constraints handling and optimization of input in terms of the value function makes it extremely attractive in the industry. Nevertheless, the complexity of implementation of MPC algorithm on real mechatronic systems remains one of the main challenges.

Secondly, common predictive control algorithms are based on the regulation approach or a simple step shift. However, there exist systems that are more complicated where a setpoint to be tracked is given in the form of trajectories. In this project, there were made several modifications in order to improve an MPC algorithm to make better use of information about the trajectories.

Keywords: Model-Based Predictive Control, MPC, SSTO, trajectory tracking, finite horizon, PID, feedback loop, state-space models, Hessian matrix, cost function, LQR problem.

Introduction

The general concept of control systems is based on the feedback loop. Usually, controllers perform conventional steps, such as gathering actual outputs of the system, comparing them with the requested result, and designing control to stabilize the system.

The technique that allows to control system is called a control law. Thus, designing a suitable control law is the most important in system engineering since many objectives are needed to be considered. PID is one of the most popular and widely used controllers in industries. However, there are different types of controllers such as Model-based Predictive Control (MPC), which is the advanced method in control process and that has been investigated until now. MPC is not an algorithm with a set of equations; it is an art of control with various approaches, such as DMC, GPC, PFC, RHC, etc. The key idea of predictive control is:

- Accurate use of a model to predict the process output of the system;
- Computation of an approach sequence that will minimize an objective function;
- Receding horizon idea, at each iteration the horizon is moved to forward, which enables the use of the first control signal of the sequence computed at each iteration.

There are illustrated advantages of MPC over other methods such as:

- It is possible to design control with limited knowledge of control, since the steps are very intuitive, and the tuning is relatively easy;
- Control could be applied to simply dynamics as well as to complex ones;
- Can be applied to the multivariable case;
- It represents feed forward control in a natural way to compensate for measurable disturbances;
- Constraints could be applied to the approach during the design process;
- It is very beneficial when future references are established.

Part 1. Problem Definition

Along with the benefits of the MPC approach appear to have drawbacks. Even though it is assumed that this type of control law is easy to implement, the system's derivation becomes more complicated when constraints are applied to the system. Also, in extended cases, MPC could require a lot of computational effort. Nevertheless, the main difficulty is the need of precise model of the system. Because the model of the system is prior knowledge based on which decisions will be made.

In addition, the usual rule of predictive control is that a traceable set point must be zero or a step change. Many systems have more complicated set point trajectories, so the traditional MPC approach cannot be applied in these cases and need to be modified. This project intends to deal with the problems that occur when advance knowledge of future set point changes is available and the algorithm used to track the reference signal is a Model-based Predictive Controller, specifically Dual Mode MPC is implemented to carry out the experiments needed to reason the proposed solutions.

Scientific Journal of Astana IT University
ISSN (P): 2707-9031 ISSN (E): 2707-904X
VOLUME 11, SEPTEMBER 2022

Part 2. Model-Based Predictive Control

MPC is a control approach when the operating control input is computed at every iteration by minimizing the value function. This turns out to be a finite horizon LQR problem, when the state at each step is taken as an initial point, and using this information, it is needed to generate the most optimum input in order to implement one step ahead algorithm. Consequently, the optimization leads to the optimal control vector where the first element is used in the system. Hence, the basic distinction of MPC from other control laws is that it does not use pre-calculated values for the off-line control law [1].

MPC relates to optimal control. The main MPC approach is a dynamic model usage to predict the plant`s future values and minimize cost function to make the most optimum solution. Then the controller does a step ahead and start all over again. MPC tries to read all past data and predict the future behavior of the system applying the best optimum input to the system. The same processes are done in each step dynamically. This differentiates the model-based predictive control approach from other control laws.

For instance, the PID control law is not calculated at each step and there are no future predictions made in each step. The basic approach of PID is to obtain some gains from the start and to keep applying the same gains throughout the whole process. Conversely, MPC estimates the future over a specific horizon at each step.

In the other word Model Predictive Control uses the same principles as a human being where the optimum action is applied in order to obtain the best output. For example, when a human drives a car, the objective is to get home as soon as possible while not breaking the speed limit. As human eyes cannot see the home from their workplace, they cannot predict the whole way, but the best thing to do is to predict as much as they can see the so-called prediction horizon and apply the best decisions each time they go ahead.

Receding Horizon

Optimization-based control links to the usage of online and optimum future path calculation as a part of the feedback stabilization of a (which is usually nonlinear) system. The main idea is to utilize a receding horizon control method: the optimum predicted path is calculated from the current state to the desired state over a finite horizon N, used for only this specific iteration, and recalculated further based on the new data. The basic principle of receding horizon is illustrated in Figure 1.

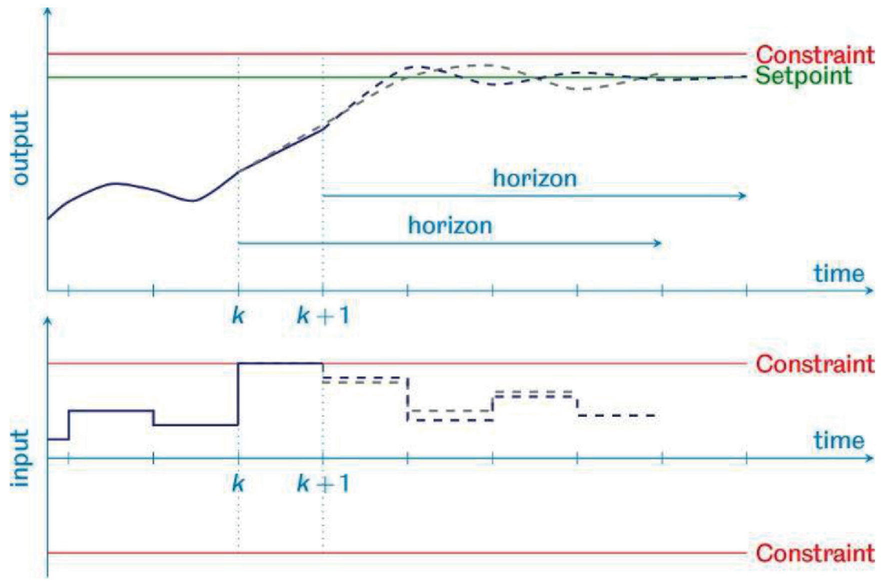

Figure 1. Receding horizon concept [2]

The horizon must contain all important dynamics as settling time; otherwise, performance can be poor and some significant points can be lost or unobserved.

Constraints are very common in real life but it is very difficult to handle. Traditional methods are aiming to handle the constraints off-line [2].

Controllability and observability

There is a frequent usage of terms like stabilizability and detectability. The basic concept of system controllability, observability, stabilizability, and detectability could be described as follows:

Controllability: A control system is said to be reachable if, for any given finite time interval $[t_0, t_f]$, it is possible to find some input signal $u(t)$ for all $t \in [t_0, t_f]$ that will steer any given initial state $x(t_0)$ to any final state $x(t_f)$.

Observability: A control system is said to be observable if, for any $t > t_0$, it is possible to determine the state of the system $x(t)$ through measurements $y(t)$ and $u(t)$ for $t \in [t_0, t_f]$.

Stabilizability: A linear system is said to be stabilizable if all its unstable modes, if any, are controllable.

Detectability: A linear system is said to be detectable if all its unstable modes, if any, are observable [3].

Theorem: A control system is said to be controllable if and only if the rank of $n \times nm$ controllability matrix C shown in equation 2.1 is equal to n.

$$\mathrm{C} = [B\ AB\ \ldots A^{n-1}B] \tag{2.1}$$

The significance of stabilizability is that even if some modes is not controllable, in the case of having them stable or asymptotically stable, these specific modes will stay bounded or decay to zero [3].

Theorem: A control system is said to be observable if and only if the rank of $np \times n$ observability matrix $\mathcal{O}$ shown in equation 2.2 is equal to n.

$$\mathcal{O} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix} \tag{2.2}$$

The significance of detectability is that even if some modes is unstable, at least this behavior can be observed.

Existing model overview

Numerous studies on the computational efficiency of trajectory tracking are currently available. Jiao and Wang [8] looked into an event-triggered trajectory tracking control method based on the guidance-control framework for fully actuated surface vessels. The suggested method results in fewer controller executions and less calculation and signal transfer. An event-triggered reset controller with a nonlinear disturbance observer was created by Wang and Zhang [9]. It has been demonstrated that the suggested plan is significantly simpler to adopt and can save resources more effectively. In order to track unicycle robots, Sun et al. [10] suggested two event-based adaptive prediction horizon MPC algorithms that took into account the computational complexity at each triggering instant. The outcomes demonstrated that the methods are successful in reducing the computational load [11].

Additionally, a number of research addressed employing meta-heuristic algorithms to solve MPC optimization problems. Three different meta heuristic optimization strategies were covered by Merabti et al. [12] in order to finish the optimization of the nonlinear MPC for

Scientific Journal of Astana IT University
ISSN (P): 2707-9031 ISSN (E): 2707-904X
VOLUME 11, SEPTEMBER 2022

control of tracking the course of the mobile robot. MPC and path planning were developed by Falcone et al. [13] using a bicycle vehicle model. Alternative MPC was created by Yakub and Mori [14] based on the Borelli principle paired with a feed forward controller. However, these investigations relied on trial-and-error techniques to calculate the MPC controller gain value. Additionally, the linearization of the vehicle's dynamic model, which the MPC controller uses, is done at a chosen speed [15].

Part 3. Designing MPC

The basic aim of model-based predictive control is to provide controls u so that the system meets all control goals such as having output y and tracking a set point r, and zero regulation of x state. Furthermore, it is assumed that the controller should be tuned so that the cost is minimized, and the performance is maximized. Finally, the system must meet all constraints. (Trodden, 2015)

Therefore, the process of controlling via MPC is as follows:

- Calculate the state or output of the system at the current iteration;

- Solve an optimization problem in order to obtain the best input which is further used in real device. The optimization:

- uses a dynamic model of the system to predict/forecast behavior for a limited time ahead from the current state;

- chooses the forecast that optimizes the performance measure while satisfying all constraints.

- Apply the optimum control;

- Repeat at the next iteration [4].

The graphical design of the controlling process of the system through MPC is illustrated in Figure 2 below.

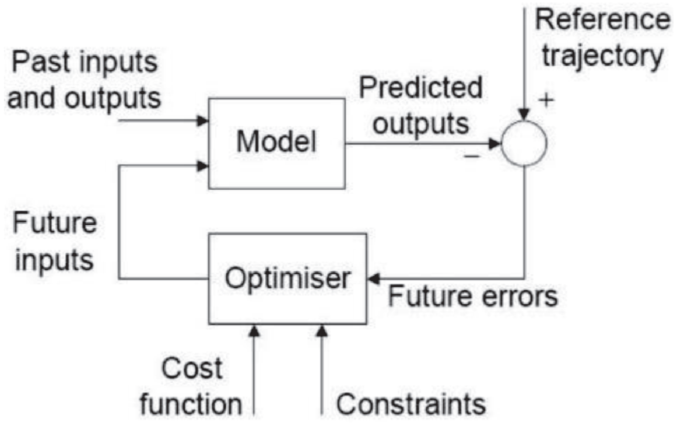

Figure 2. The structure of MPC [1]

Discrete-time linear state-space models are often convenient if the system of interest is sampled at discrete times. If the sampling rate is selected in a right way, the behavior between the samples can be safely ignored and the model describes exclusively the behavior at the sample times. The infinite dimensional, linear, time-invariant, discrete time model is shown in equation 3.1:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) + Du(k) \\ x(0) &= x_0 \end{aligned} \tag{3.1}$$

where state $x(k)$ is measurable at every step k and $u(k)$ is a vector of manipulated variables to be determined by the controller to regulate x to 0 while minimizing a stage cost function $l(x(k), u(k))$. From an initial state x_0 the cost function shown in equation 3.2:

$$\textstyle\sum_{k=0}^{\infty} x^T(k)Qx(k) + u^T(k)Ru(k) \tag{3.2}$$

is subject to (2.1) and has a unique optimal solution $u(k) = K_\infty x(k)$ if R is positive definite (p.d.), Q is positive semidefinite (p.s.d.), the pair (A, B) is stabilizable and the pair $(Q^{1/2}, A)$ is detectable. (Rawlings and Mayne, 2012)

In this case it is assumed that $l(x,u) = \frac{1}{2}[x^TQx + u^TRu]$ which is an infinite-horizon LQR problem where Q, R are weighting matrices which are commonly p.d.

Positive definite (p.d. or $Q > 0$) means that all its eigenvalues are positive whereas positive semidefinite (p.s.d. or $Q \geq 0$) means that its eigenvalues are non-negative and at least one of them is zero.

Infinite horizon control law assumes an infinite sequence and decision variables in the optimization problem, which is not the case in reality because of its intractability [5].

Thereby, the prediction horizon length is reduced to N so that its cost function transforms as it is demonstrated in equation 3.3:

$$\min_{\{u(o),u(1),\dots,u(N-1)\}} \sum_{k=0}^{N-1} (x^T(k)Qx(k) + u^T(k)Ru(k) + x^T(N)Px(N)) \\ \text{subject to } x(k+1) = Ax(k) + Bu(k) \text{ for } k = 0, 1, 2, \dots, N-1 \tag{3.3}$$

where P is a terminal cost matrix. In this case only N control inputs are optimized from x_0. Hence, the obtained control sequence is $\{u(0),\ u(1),\ \dots\ ,\ u(N-1)\}$ and recursive application of this method leads to 3.4:

$$\begin{gathered} x(1) = Ax(0) + Bu(0) \\ x(2) = Ax(1) + Bu(1) = A^2x(0) + ABu(0) + Bu(1) \\ \vdots \\ x(N) = A^Nx(0) + A^{N-1}Bu(0) + A^{N-2}Bu(1) + \dots + Bu(N-1) \end{gathered} \tag{3.4}$$

and combining everything in the form of $\overrightarrow{x_0} = \boldsymbol{Fx(0)} + \boldsymbol{G}\overrightarrow{u_0} =$ for every $k = 1,2,\ \dots\ , N$ where state vector $\overrightarrow{x_0}$, input vector $\overrightarrow{u_0}$ and prediction matrices F and G are characterized as in the following equation 3.5:

$$\vec{x}_0 = \begin{bmatrix} x(1) \\ x(2) \\ \vdots \\ x(N) \end{bmatrix}, \vec{u}_0 = \begin{bmatrix} u(0) \\ u(1) \\ \vdots \\ u(N-1) \end{bmatrix}, F = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}, G = \begin{bmatrix} B & 0 & \cdots & 0 \\ AB & B & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ A^{N-1}B & A^{N-2}B & \cdots & B \end{bmatrix} \tag{3.5}$$

The cost function of finite-horizon LQR is rewritten as in the equation 3.6:

Scientific Journal of Astana IT University
ISSN (P): 2707-9031 ISSN (E): 2707-904X
VOLUME 11, SEPTEMBER 2022

$$\sum_{k=0}^{N-1}\left(x^T(k)Qx(k)+u^T(k)Ru(k)\right)+x^T(N)Px(N)$$
$$x^T(0)Qx(0)+u^T(0)Ru(0)+x^T(1)Qx(1)+u^T(1)Ru(1)+\cdots+x^T(N)Px(N)$$
$$x^T(0)Qx(0)+\begin{bmatrix}x(1)\\x(2)\\\vdots\\x(N)\end{bmatrix}^T\begin{bmatrix}Q&0&\cdots&0\\0&Q&\cdots&0\\\vdots&\vdots&\ddots&0\\0&0&\cdots&P\end{bmatrix}\begin{bmatrix}x(1)\\x(2)\\\vdots\\x(N)\end{bmatrix}+\begin{bmatrix}u(0)\\u(1)\\\vdots\\u(N-1)\end{bmatrix}^T\begin{bmatrix}R&0&\cdots&0\\0&R&\cdots&0\\\vdots&\vdots&\ddots&0\\0&0&\cdots&R\end{bmatrix}\begin{bmatrix}u(0)\\u(1)\\\vdots\\u(N-1)\end{bmatrix} \tag{3.6}$$

Therefore, the problem is reformulated into the form of min $x^T(0)Qx(0)+\vec{x}_0^T\tilde{Q}\overrightarrow{x_0}+\vec{u}_0^T\tilde{R}\overrightarrow{u_0}$ which is subject to $\overrightarrow{x_0}=\boldsymbol{Fx}(\mathbf{0})+\boldsymbol{G}\overrightarrow{u_0}$ where:

$$\tilde{Q}=\begin{bmatrix}Q&0&\dots&0\\0&Q&\dots&0\\\vdots&\vdots&\ddots&\vdots\\0&0&\dots&P\end{bmatrix}\text{ and }\tilde{R}=\begin{bmatrix}R&0&\dots&0\\0&R&\dots&0\\\vdots&\vdots&\ddots&\vdots\\0&0&\dots&R\end{bmatrix}$$

Finally, at initial x_0 the finite-horizon LQR task can be written as following 3.7:

$$\min\frac{1}{2}\vec{u}_0^T H\overrightarrow{u_0}+c^T\overrightarrow{u_0}+\alpha \tag{3.7}$$

where $H=2\left(G^T\tilde{Q}G+\tilde{R}\right)$ is Hessian matrix, $c=Lx(0)$ and $\alpha=x^T(0)Mx(0)$, where $L=2G^T\tilde{Q}F, M=F^T\tilde{Q}F+Q, Q\geq 0, P\geq 0$ and $R>0$ which leads to $H>0$.

In order to solve this optimization task, it is necessary to take a gradient of the cost function and set it to 0. This helps to determine the basic $\overrightarrow{u_0^*}$ that could be applied as an input, which is described in (3.8):

$$\begin{gathered}J(x(0),\overrightarrow{u_0})=\frac{1}{2}\vec{u}_0^T H\overrightarrow{u_0}+c^T\overrightarrow{u_0}+\alpha\\\nabla_u J(x(0),\overrightarrow{u_0})=H\overrightarrow{u_0}+c=0\\\overrightarrow{u_0^*}=-H^{-1}c\\\overrightarrow{u_0^*}=-H^{-1}Lx(0)\end{gathered} \tag{3.8}$$

In the case of having $H>0$ the input $\overrightarrow{u_0^*}$ is unique global minimum. The Hessian matrix H is not directly inverted, because it can even be non-invertible. Instead, the pseudo-inverse technique is used in order to handle this problem as in equation 3.9:

$$\overrightarrow{u_0^*}=-(H^TH)^{-1}H^TLx(0) \tag{3.9}$$

The same principle made at initial x_0 is continuously repeated at every step. Therefore, the overall procedure can be described as:

1. At every step k measure $x(k)$;
2. Solve finite-horizon LQR task;
3. Apply the first control input from $\overrightarrow{u_k^*}$;
4. Go to first step.

Note that at each step k:

$$\vec{x}_k = \begin{bmatrix} x(k+1|k) \\ x(k+2|k) \\ \vdots \\ x(k+N|k) \end{bmatrix}, \vec{u}_k = \begin{bmatrix} u(k|k) \\ u(k+1|k) \\ \vdots \\ u(k+N-1|k) \end{bmatrix}$$

$$V(k) = \min_{u(k)} \underbrace{\sum_{j=0}^{N-1} \left(x^T(k+j|k)Qx(k+j|k) + u^T(k+j|k)Ru(k+j|k)\right)}_{cost\ over\ finite\ horizon} + \underbrace{x^T(k+N|k)Px(k+N|k)}_{terminal\ cost}$$

$$\text{subject to: } x(k+j+1) = Ax(k+j|k) + Bu(k+j|k),\ \ j = 0,1,\dots,N-1$$
$$x(k|k) = x(k)$$

Figure 3. Finite-horizon

So, the receding horizon control law $u(k) = K_N x(k)$ where $K_N = [I, 0 \dots 0] - H^{-1}L$. By this method only the first control input from $\overrightarrow{u_k^*}$ is applied to system. The horizon N is chosen to be as small as it can afford itself to be. Because the more horizon the more computational cost and the more Q and P matrices are out of condition [6].

However, very small horizon can cause significant difference between K_N and K_∞ , hence, bad prediction and performance as it is illustrated in Figure 4. In this figure it is revealed that when $N \to \infty$, then $K_N \to K_\infty$.

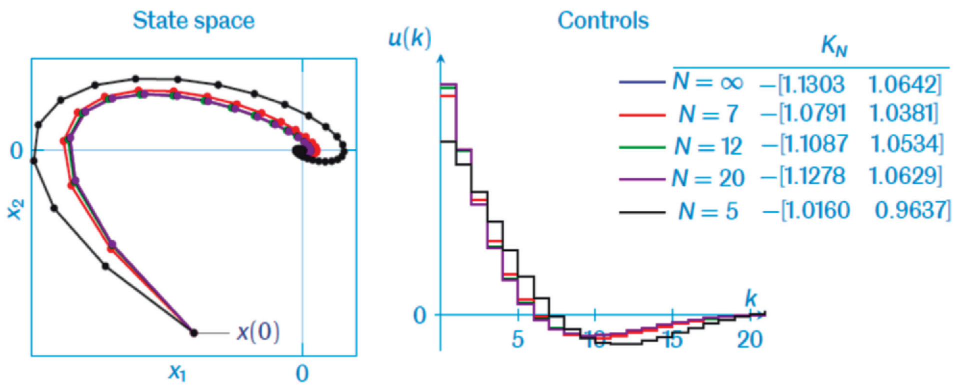

Figure 4. Example of Linear Quadratic MPC vs IH-LQR [2]

Sometimes K_N could not even be stabilizing the closed loop of the system $(A + BK_N)$. And one of the method that could handle that problem is to set terminal state constraint $x(k+N|k) = 0$. This means that the last state is forced to be zero and this continues to be zero for every $k \geq N$. However, this approach is very rude and can lead to bad robustness in regard to uncertainties of modelling error and disturbances. Moreover, it can also cause to have some problems while constraints handling.

Instead of doing this another approach so called Dual-Mode MPC can be used in order to improve the performance having two modes $k < N$ and $k \geq N$.

Part 4. Constraints handling

One of the essential advantages of MPC is that it can easily handle constraints, as it is known that in real life, constraints are just everywhere. For instance, the physical limits of actuators, capacity limits, or even safety limits like having boundaries in temperature, in pressure, etc.

Scientific Journal of Astana IT University
ISSN (P): 2707-9031 ISSN (E): 2707-904X
VOLUME 11, SEPTEMBER 2022

However, control in the presence of constraints is difficult, even for linear systems. The trouble is that optimal operation usually means operating at limits, as seen in Figure 5.

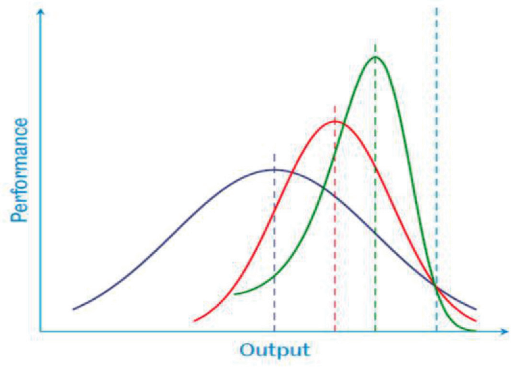

Figure 5. Operating near limits. [4]

The linear inequality method is used to handle the constraints:

$$P_x x(k+j|k) \leq q_x$$

$$P_u \mathrm{u}\,(k+j|k) \leq q_u$$

$$P_{xN}\ x(k+N|k) \leq q_{xN}$$

Consider the input and state constraints, which can be rewritten as follows:

$$\begin{bmatrix} P_u & 0 & \dots & 0 \\ 0 & P_u & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & P_u \end{bmatrix} \vec{u}_k \leq \begin{bmatrix} q_u \\ q_u \\ \vdots \\ q_u \end{bmatrix} \quad => \quad \tilde{P}_u \vec{u}_k \leq \tilde{q}_u$$

$$\begin{bmatrix} P_x \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} x(k|k) + \begin{bmatrix} 0 & 0 & \dots & 0 \\ P_x & 0 & \dots & 0 \\ 0 & P_x & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & P_{xN} \end{bmatrix} \vec{x}_k \leq \begin{bmatrix} q_x \\ q_x \\ q_x \\ \vdots \\ q_{xN} \end{bmatrix} \quad => \quad \tilde{P}_{x_0} x(k) + \tilde{P}_x (Fx(k) + G\vec{u}_k) \leq \tilde{q}_x$$

$$\begin{bmatrix} \tilde{P}_u \\ \tilde{P}_x G \end{bmatrix} \vec{u}_k \leq \begin{bmatrix} \tilde{q}_u \\ \tilde{q}_x \end{bmatrix} + \begin{bmatrix} 0 \\ -\tilde{P}_{x_0} - \tilde{P}_x F \end{bmatrix} x(k) \quad => \quad P_c \leq q_c + S_c x(k)$$

The method above could give the constraints in the form of $P_c \leq q_c + S_c x(k)$. The $\overrightarrow{u_k^*}$ is linear system when there are no constraints, but it is non-linear in the presence of limits. In this case, the feasible region may not cover the unconstrained minimum and the best choice is to take the point from feasible region, which is the nearest to the unconstrained minimum. Therefore, it is required to recalculate the $\overrightarrow{u_k^*}$ at every step through solving constrained Quadratic Program. Constrained system can be locally stable only for some x_0, and in order to ensure the local stability it is necessary to have value function as a Lyapunov function and asymptotical to 0. Lyapunov function means that the value function is positive definite and is reducing monotonically. In the case of having the ability to meet all those criteria, we would also guarantee the extremely attractive property of recursive feasibility [7].

Recursive feasibility can be reached through the invariant set. The set Ω is said to be invariant if $x(0) \in \Omega => x(k) \in \Omega$ for all $k > 0$. So the basic objective is to have $x(k+N|k) \in \Omega$, which leads to have $x(k+N+j|k) \in \Omega,\ \forall j \geq 0$ as it is represented in Figure 6.

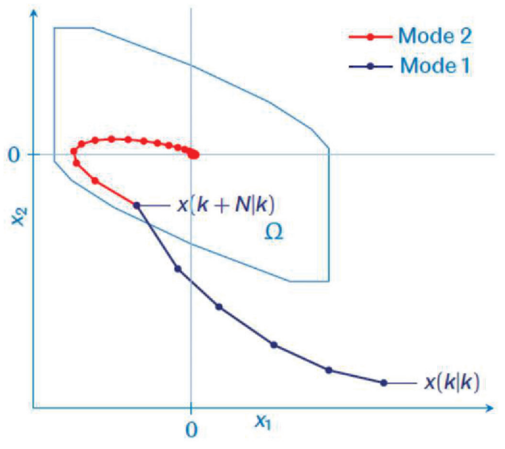

Figure 6. Example of invariant terminal set [2]

Part 5. Trajectory tracking

A conventional Model-based predictive control algorithm, which is used to track a set point, expects either regulation or a standard step shift. In this approach, it is assumed that the set point ☒ equals to a static variable. Nevertheless, instead of this traditional method, the setpoint ☒ can be chosen to be not a variable but a vector. This vector can consist of the basic points of the trajectory, which is supposed to be tracked as in 5.1:

$$r = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{bmatrix} \tag{5.1}$$

where n is the number of basic points of the trajectory. The figure 7 illustrates the key idea of this approach.

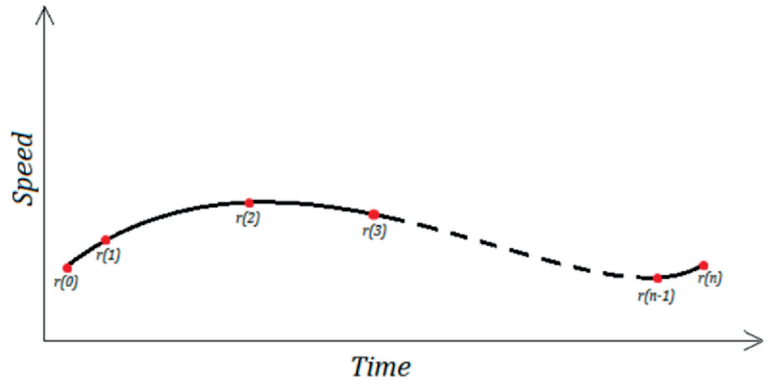

Figure 7. Example of tracking speed trajectory

In this graph it is seen that some basic points are taken from the required trajectory for the speed. These points can form a vector of setpoints which can be further used in the MPC algorithm.

Thereby, now a setpoint r to be tracked becomes a vector of points. This means that the output $y(k) \rightarrow r(i)$ having $i = 1, 2, \ldots, n$. Thus, the steady-state target optimization (SSTO) turned to calculate $(x_{ss}(i), u_{ss}\ (i))$ for the provided set point vector as it is shown in Figure 8.

Scientific Journal of Astana IT University
ISSN (P): 2707-9031 ISSN (E): 2707-904X
VOLUME 11, SEPTEMBER 2022

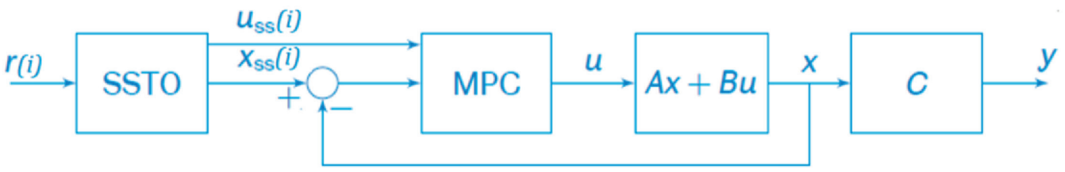

Figure 8. Application of SSTO while tracking the trajectory

As in this case the $x(k) \to x_{ss}(i)$ and $u(k) \to u_{ss}\ (i)$ it is necessary to regulate the state error $\varepsilon(k)$ and output error $\vartheta(k)$ to zero as it is described in 5.2:

$$\varepsilon(k) = x(k) - x_{ss}(i) \Rightarrow \varepsilon(k) \to 0$$
$$\vartheta(k) = u(k) - u_{ss}\ (i) \quad \Rightarrow \vartheta(k) \to 0 \tag{5.2}$$

The setpoint equilibriums $(x_{ss}(i),\ u_{ss}\ (i))$ is at the equilibrium when $x_{ss}(i) = Ax_{ss}(i) + Bu_{ss}(i)$ and the output steers to the setpoint $r(i)$ without any errors only when $Cx_{ss}(i) = r(i)$.

This leads to the following:

$$\begin{bmatrix} I-A & -B \\ C & 0 \end{bmatrix} \begin{bmatrix} x_{ss}(i) \\ u_{ss}\ (\mathrm{i}) \end{bmatrix} = \begin{bmatrix} 0 \\ r(i) \end{bmatrix},\ T = \begin{bmatrix} I-A & -B \\ C & 0 \end{bmatrix} \tag{5.3}$$

As it could be seen the matrix T is not changing which makes it very easy to implement and make only a few modifications to the algorithm.

Conclusion

This scientific paper performs a realization of Model-based Predictive Control with constraints handling algorithm. Furthermore, regular MPC logic is refined so that the controller can allow the system to track a set point trajectory. The realization of this approach was executed without violating the constraints, and all goals were achieved successfully. In addition, a traditional MPC algorithm is improved in order to deal with trajectory tracking problems, and the realization is performed successfully without any constraint violation. All the mathematical computations, explanations, and reasoning were provided throughout the whole process. The approach and methodology provided in this article could be applied to different areas where control systems are needed, i.e., Mechatronics, Robotics, IoT, etc.

References

1. Rossiter, J.A. (2003). *Model-Based Predictive Control: A Practical Approach (ControlSeries)*. 1st Edition. CRC Press.
2. Trodden, P. (2015). Model Predictive Control Lecture Notes. The University of Sheffield.
3. Guo, L. (2015). State Space Design Methods Lecture Notes. The University of Sheffield.
4. Maciejowski, J. (2000). *Predictive Control with Constraints*. 1st Edition. Pearson Education Limited. Prentice Hall. https://doi.org/10.1002/acs.736
5. Rawlings, J.B. & Mayne, D.Q. (2012). *Model Predictive Control: Theory and Design*. Nob Hill Publishing.
6. Murray, R. (2010). *Optimization-Based Control. Control and Dynamical Systems*, California Institute of Technology.
7. Kasdirin, H. (2006). *Model Predictive Control (MPC) For Use in Autopilot Design*. The University of Sheffield.
8. Jiao, J., & Wang, G. (2016). Event triggered trajectory tracking control approach for fully actuated surface vessel. *Neurocomputing*, 182.

9. Wang, H., Zhang, S. (2021). Event-triggered reset trajectory tracking control for unmanned surface vessel system, *Proc. Inst. Mech. Eng. Part I J. Syst. Control Eng.*, 235.
10. Sun, Z., Xia, Y., Dai, L., & Campoy, P. (2020). Tracking of unicycle robots using event-based MPC with adaptive prediction horizon. *IEEE/ASME Trans. Mechatronics*, 25.
11. Yuan, S., Liu, Z., Zheng, L., Sun, Y., & Wang, Z. (2022). Event-based adaptive horizon nonlinear model predictive control for trajectory tracking of marine surface vessel. *Ocean Engineering*, *258*,111082
12. Merabti, H., Belarbi, K., & Bouchemal, B. (2016). Nonlinear Predictive Control of a Mobile Robot: A Solution using Metaheuristcs. *Journal of the Chinese Institute of Engineers*, *39*,282-290.
13. Falcone, P., Borrelli, F., Asgari, J., Tseng, H. E., & Hrovat, D. (2007). Predictive Active Steering Control for Autonomous Vehicle Systems. Control Systems Technology, *IEEE Transactions* on. 15, 566-580.
14. Yakub, F. & Mori, Y. (2015). Comparative Study of Autonomous Path-following Vehicle Control Via Model Predictive Control and Linear Quadratic Control. *Journal of Automobile Engineering*, *229*(12), 1695-1713.
15. Leman, Z.A., Ariff, M.H.M., Zamzuri, H., Rahman, M.A.A., Mazlan, S.A., Bahiuddin, I., &Yakub, F. (2022). Adaptive model predictive controller for trajectory tracking and obstacle avoidance on autonomous vehicle. *Jurnal Teknologi*, *84*(4), 139-148.