1

Introduction

1.1 Background and Motivation

Model predictive control (MPC), also known as receding horizon control, is an optimization-based control strategy. The fundamental concept of MPC dates back to the 1960s [1], and since the 1980s, it has been successfully applied to industrial systems to address constrained multivariable control problems. From that point onward, MPC emerged as an advanced control methodology and gathered increasing interest from both academia and industry. The term "model predictive control" comprises three elements—model, predictive, and control—each conveying essential aspects of the method's underlying principles.

Model: In practical control system design, engineers aim to gain deeper insights into the characteristics of physical systems and to construct accurate dynamic models. A well-defined model plays a critical role in controller synthesis. For systems lacking an explicit model, identification techniques can be employed to derive suitable representations. Despite potential mismatches and uncertainties in the identified models, they still serve as a solid foundation for controller design.

Predictive: The ability to make forward-looking predictions based on current conditions—coupled with strategic planning to achieve predefined objectives—is an innate biological capability. In the MPC framework, this predictive ability constitutes a core mechanism that enables the method to realize its superior performance characteristics.

Control: Control actions in MPC are determined by solving a mathematically tractable constrained optimization problem. This process entails necessary analysis of key properties of the resulting control system.

MPC offers the following features. It facilitates systematic and flexible handling of state and input constraints. It employs an optimization-centric strategy that can deliver near-optimal performance. The algorithm exhibits inherent

robustness, which can be further enhanced through robust MPC formulations to mitigate external disturbances, parameter uncertainties, and noise. It harnesses predictive capabilities to proactively improve system performance. These attributes collectively establish MPC as a promising paradigm for future optimization-based control strategies.

Despite the promising prospects of MPC, the method relies on numerical optimization to compute control inputs, imposing stringent requirements on computational efficiency. Currently, most embedded system processors exhibit limited computational capabilities, thereby restricting the large-scale deployment of MPC. Furthermore, with the advancement of wireless communication technologies, the transmission of information between sensors and controllers is increasingly facilitated via wireless networks. Traditional communication protocols typically adopt time-triggered periodic transmission schemes, possibly resulting in communication and computation resource usage exceeding actual system needs, further constraining MPC applications.

Event-triggered control addresses these limitations by introducing an event mechanism and threshold condition such that updates occur only when specific criteria are met. The core concept behind the event-triggered control is to update the control only when the system violates predefined performance specifications. This enables a more favorable trade-off between control performance and overall communication load compared to conventional periodic schemes.

The integration of event-triggered control strategies with MPC can effectively harness this mechanism to reduce communication capacity and avoid unnecessary resource utilization. Moreover, it lowers the frequency of optimization problem-solving, thereby alleviating computational demands [2]. Event-triggered MPC continues to be a highly active research area, particularly in the context of systems subject to external disturbances, packet loss in communication, and multi-agent cooperation—each scenario presenting compelling challenges and valuable research opportunities.

Control of multi-agent systems has become one of the prominent research topics in modern control theory. Within the framework of MPC, multi-agent coordination has achieved a number of representative advancements in recent years [3, 4]. However, the practical implementation of MPC in multi-agent systems remains limited due to the inherent complexity of inter-agent network communication and the computational demands of solving predictive control problems. Consequently, reducing the frequency of communication among agents as well as the individual computational burden associated with MPC updates presents a significant research challenge. Current investigations in this area are still in early stages. Event-triggered multi-agent MPC methods offer a compelling direction for further exploration. By selectively updating control actions only when specific triggering conditions are met, such approaches have the potential to improve scalability and efficiency while preserving desirable overall system performance.

This book is dedicated to exploring event-triggered MPC methodologies, with focused investigations into four main areas:

- Event-based two-phase MPC for nonlinear continuous-time systems.
- Event-based aperiodic sampling MPC for nonlinear discrete-time systems.
- Event-triggered MPC for disturbed systems.
- Periodic sampling-based event-triggered MPC for distributed systems.

Through a detailed examination of event-triggering mechanisms, robust handling of system uncertainties, and the study of inter-agent communication protocols, the objective is to reduce communication load and minimize the frequency of optimization computations. These efforts aim to enhance the practical value of event-triggered MPC and facilitate its wider application across engineering domains.

1.2 Model Predictive Control

Modern industrial control systems are influenced by a variety of factors. In addition to physical constraints such as actuator saturation, considerations of safety, process requirements, and economic performance must also be addressed. For instance, system parameters like pressure and temperature must be maintained within permissible limits. When dealing with constrained systems, the goal is not only to achieve stabilization but also to ensure enhanced performance. Moreover, industrial processes are often characterized by high-dimensional complexity, making the precise formulation of mathematical models exceedingly difficult. Due to external disturbances and parameter variations, uncertainties arise that render optimal control strategies—derived solely from idealized models—ineffective in practical applications. Such mismatches may even result in severe degradation of control performance. Consequently, the field of control engineering demands a new class of optimization-based methodologies that can simultaneously satisfy various system constraints, optimize performance indicators, and exhibit robustness against uncertainties and disturbances.

To address the aforementioned needs, MPC was proposed by researchers in the 1970s. Due to its capacity to handle complex multivariable control problems and its ability to enforce input, output, and state constraints in real time within an optimization framework, alongside its inherent disturbance-rejection capabilities, MPC has found extensive applications in practical industrial processes. Its application spans critical military and civilian sectors, including petroleum, chemical engineering, transportation, and aerospace [5–7]. In parallel, the theoretical development of MPC has attracted considerable attention from the research community [8, 9].

The underlying mechanism of MPC can be described as follows: At each sampling instant, the controller utilizes current state measurements to formulate and solve an optimization problem over a finite prediction horizon. The computed control sequence is then implemented by applying the first—or in some cases, a subset—of its elements to the plant. At the next sampling instant, the optimization problem is updated with new measurements and solved, and this process continues in a receding horizon fashion. MPC is particularly well-suited for addressing multivariable constrained optimization problems that Proportional-Integral-Derivative (PID) control struggles to manage. As a model-based control strategy, MPC is capable of handling nonlinear, time-varying, and time-delay systems subject to complex constraints. The MPC algorithm typically involves three main steps:

- Prediction: Estimating the future evolution of the system using a predictive model.
- Optimization: Solving a finite-horizon constrained optimization problem via numerical methods.
- Implementation: Implementing a portion of the optimal control sequence on the actual system.

Compared with other control methodologies, the primary characteristics of MPC include (but not limited to):

- Model-based Prediction: MPC relies on a model that describes the dynamic behavior of the controlled system. This model is used to forecast future system responses, hence the term "predictive model"—which is the reason MPC is considered a model-based control strategy. Within the MPC framework, the model serves as a tool to predict the system's future trajectory based on current states and anticipated future inputs. Notably, the formulation of the model itself is not restricted to any specific representation. To adjust future outputs toward desired reference trajectories, control inputs are optimized accordingly. The form in which the predictive model is expressed (e.g., state-space equations, transfer functions, or convolution representations such as step and impulse responses) does not affect this predictive process. In fact, the model may also be derived from data using machine learning or other artificial intelligence techniques, making it suitable for complex systems where first-principles modeling is infeasible.
- Receding Horizon Optimization: Due to limitations in computational capacity, the prediction horizon used in numerically solving the optimization problem cannot be indefinitely long. As a result, MPC algorithms typically employ a finite prediction horizon. At each sampling instant, MPC computes an optimal control sequence by solving a constrained optimization problem, and only the initial portion of the solution is implemented to the system. This

process is carried out iteratively in a receding horizon fashion, allowing the controller to continuously update and correct for external disturbances and model mismatches. By solving the optimization problem at each step using the latest system measurements, MPC adapts to real-time changes and environmental uncertainties. This dynamic updating mechanism enhances control performance and enables more effective realization of optimal control objectives.

- Integration of Feedforward and Feedback in the Control Structure: Feedforward refers to the use of a predictive model to predict system state evolution over a finite time horizon, thereby enabling proactive compensation in the computation of control inputs. For instance, during vehicle turning or obstacle avoidance, the predictive model facilitates early response actions, enabling the control inputs with a degree of foresight. Since each optimization problem is solved based on measurements of the current system states or outputs, the resulting optimal control input can essentially be regarded as a function of the present measurements. When the optimization is repeated at the next sampling instant using updated measurements, the newly computed control inputs naturally incorporate this feedback. Thus, the MPC algorithm inherently exhibits a feedback structure, even while leveraging model-based predictions for feedforward control.
- Explicit Constraint Handling: MPC enables the explicit incorporation of constraints within the optimization framework, including constraints on system states and control inputs. The constraints are due to hard constraints imposed by environmental or operational conditions, as well as constraints resulting from control capacity. Compared to other control strategies, MPC exhibits a distinct advantage in simultaneously solving for optimal control actions while effectively addressing constraint-related challenges. This capacity to manage complex constraint scenarios underscores its strong practical relevance and engineering applicability.

MPC is a numerically-based control algorithm that emerged from industrial process control. Predictive control originated in practical industrial production and has since evolved into various distinct forms. One category includes industrial predictive control methods that were developed through continuous exploration and innovation within industrial applications. These approaches often rely on step or impulse response models, with typical representatives including dynamic matrix control and model predictive heuristic control [10, 11]. Although widely used in practice, these methods lack formal guarantees of closed-loop stability from a theoretical point of view.

Another class comprises adaptive predictive control methods, rooted in adaptive control and minimum variance control. A major contribution in this area

was the Generalized Predictive Control (GPC) algorithm proposed in the 1980s [12]. GPC facilitated the development of quantitative analysis tools, but its application to complex nonlinear systems remains challenging, hindering further theoretical breakthroughs. Both industrial and generalized predictive control methods are commonly referred to as classical predictive control, characterized by their emphasis on practical engineering requirements rather than rigorous theoretical analysis.

Beginning in the 1990s, a new phase of predictive control research emerged, focused on achieving formal stability guarantees. MPC began to incorporate concepts from optimal control, along with invariant set theory and Lyapunov-based approaches. This shift enabled the design of controllers that could meet both stability requirements and performance specifications within a principled framework. Since then, MPC theory has matured significantly and continues to evolve through integration with modern information technologies. It has gradually become a prominent and emerging direction within the control community, offering tremendous potential for future applications [13–15].

Consider a general deterministic discrete-time system described by

$$x_{k+1} = f(x_k, u_k),$$

where x_k and u_k represent the system's state and control input at time step k, respectively. Both the state and input are subject to constraints:

$$x_k \in \mathbb{X}, \quad u_k \in \mathbb{U},$$

where $\mathbb{X}$ and $\mathbb{U}$ are convex, compact sets that contain the origin. The origin is assumed to be an equilibrium point of the system, and the system states are assumed to be available in real time.

The control objective is to stabilize the system to the origin. At the core of MPC is the real-time solution of an optimal control problem over a finite time horizon, based on a given performance index. This yields a control sequence that optimizes the predicted behavior of the future system states, where the system dynamics are used to forecast the trajectory. The finite-horizon optimal control problem is formulated as follows:

$$\begin{aligned}
&\min_{u_k} \; J(x_k, u_k) = \sum_{i=0}^{N-1} L(x_{k+i|k}, u_{k+i|k}) + V_N(x_{k+N|k}) \\
\text{s.t.} \quad &x_{k+i+1|k} = f(x_{k+i|k}, u_{k+i|k}), \\
&x_{k|k} = x_k, \\
&x_{k+i|k} \in \mathbb{X}, \;\; u_{k+i|k} \in \mathbb{U}, \\
&x_{k+N|k} \in X_f,
\end{aligned} \tag{1.1}$$

where N denotes the prediction horizon. The control input sequence is represented by u_k, and x_k is the system state at time step k. The notation $x_{k+i|k}$ indicates the predicted state at time $k+i$, computed at time k. The stage cost function $L(x_{k+i|k}, u_{k+i|k})$ captures the intermediate performance over the horizon and is typically expressed in a quadratic form involving the state and control input, reflecting their respective weightings throughout the prediction. The terminal cost $V_N(x_{k+N|k})$ is defined as a quadratic function of the terminal state. The terminal set X_f imposes constraints on the terminal state. By properly designing the terminal cost and terminal set, it is possible to enforce a monotonic decrease in the performance index, thereby contributing to the stability of the control algorithm.

For a continuous-time system given by

$$\dot{x}(t) = f(x(t), u(t)),$$

where the system state and control input satisfy the constraints $x(t) \in \mathbb{X}$ and $u(t) \in \mathbb{U}$, respectively, an optimal control problem is solved at time t_k. The total cost function to be minimized is designed by

$$J(x(t), u(t)) = \int_{t_k}^{t_k+T} L(x(t), u(t))\, dt + V_f(x(t_k + T)), \tag{1.2}$$

where T denotes the prediction horizon, and $u(t)$ is the predicted control signal over the interval. The stage cost function $L(x(t), u(t))$, typically expressed as a quadratic function of the state and input, captures their relative importance throughout the prediction horizon. The terminal cost $V_f(x(t_k + T))$ is usually a quadratic function of the terminal state, representing its weighted contribution at the end of the horizon. The total cost function combines the accumulated stage costs with the terminal cost, thereby accounting for both the cumulative influence of state and input over the prediction horizon and the desired behavior of the system at its terminal state.

In the stability analysis of MPC closed-loop system, the optimal cost function is commonly employed as a Lyapunov candidate. For discrete-time systems, let u_k^* denote the optimal control input sequence obtained at time step k, with its associated cost function denoted by J_k^*. Similarly, u_{k+1}^* and J_{k+1}^* represent the optimal control input and corresponding cost function at time $k+1$. Since these optimal control sequences are independently computed based on the system states at k and $k+1$, they lack a direct analytical relationship—making it challenging to rigorously establish the connection between successive Lyapunov candidate functions J_k^* and J_{k+1}^*. To address this issue, researchers introduced a feasible solution approach in stability analysis. By constructing a feasible (not necessarily optimal) control input sequence u_{k+1} at time $k+1$, and denoting its cost as J_{k+1}, an intermediate trajectory is established that facilitates a comparison between J_k^* and J_{k+1}^*.

Early literature often adopted the simplifying assumption of a zero terminal set, ensuring that the predicted state at the terminal step N converges to the origin. During theoretical analysis, the latter part of the feasible control sequence was frequently set to zero, which allowed derivation of asymptotic stability conditions relatively easily. However, the imposition of a zero terminal constraint—despite its analytical convenience—can render the optimization problem infeasible in practice, or restrict its feasibility set to an impractically small region.

Subsequent studies relaxed the terminal equality constraint and introduced the concept of terminal inequality constraints together with invariant set theory [16]. The terminal set is selected as a small neighborhood around the origin wherein a local feedback control law guarantees system stability. Crucially, this terminal set is designed as a control invariant set, meaning that under the specified feedback policy, the system state remains within the set indefinitely. This refinement allows the construction of a feasible control input sequence at each time step that connects J_k^* and J_{k+1}^*, thereby enabling rigorous demonstration of system stability [17]. Consequently, the integration of terminal cost, terminal set, and terminal control law concepts provides the foundation for a theoretically sound MPC framework, one that successfully addresses feasibility and stability concerns for constrained discrete-time systems [8].

1.3 MPC in the Presence of Uncertainties

One critical issue in MPC is how to effectively manage system uncertainties, such as additive disturbances and parameter uncertainties. Owing to its receding-horizon optimization structure, standard MPC exhibits a degree of inherent robustness against small uncertainties [18]. However, to ensure robust stability, constraint satisfaction, and favorable closed-loop performance under uncertainty, researchers have developed rigorous theoretical frameworks for robust MPC.

According to the existing literature, three main approaches have been proposed for addressing robustness in MPC:

- Tube-based Methods: These approaches construct a bounded tube around the nominal trajectory to contain all possible state deviations due to disturbances, ensuring constraint satisfaction and stability.
- Constraint-tightening Methods: By systematically contracting the feasible constraint sets, these methods proactively account for worst-case uncertainty effects during optimization.
- Min-max Optimization Strategies: These methods explicitly formulate a min-max optimization problem to derive control policies that perform optimally against the worst-case disturbance scenarios.

For systems subject to uncertainty, the tube-based MPC approach was proposed. By adopting simplified parameterized local control policies within the optimization, this method facilitates tractable implementation while partially sacrificing optimality [19]. Due to uncertainties, each control action induces a set of possible trajectories, collectively referred to as a tube, where each trajectory corresponds to a specific uncertainty realization. The core idea of tube-based MPC is to impose additional constraints on the optimization problem such that all possible trajectories within the tube remain confined to a small neighborhood around a nominal trajectory [20]. As a result, tube-based MPC preserves a computational complexity comparable to that of standard MPC, while achieving less conservative closed-loop performance.

The fundamental idea behind tube-based MPC is to tighten state constraints around a nominal trajectory using a local feedback control law, ensuring that all possible realizations of system uncertainty remain within the constraints. This is accomplished by constructing a positively invariant set that bounds the deviation between the actual and predicted system states under feedback control [21]. In Ref. [19], the state and input constraint sets are contracted to

$$\bar{X} = X \ominus S, \quad \bar{U} = U \ominus KS,$$

where K is a stabilizing feedback gain, S is a disturbance invariant set, and $\ominus$ denotes the Pontryagin difference between sets. This constraint tightening ensures that even in the presence of disturbances, the true system states and inputs stay within the original admissible sets.

Later work [22] introduced an improved formulation by including the nominal system's initial state as a decision variable in the MPC optimization problem. This allowed the feasible set to expand and enabled robust exponential stability. The key innovation is that the nominal initial state is optimized online rather than fixed. This design offers several advantages: the cost function is zero inside a robust invariant set Z, facilitating proofs of stability and attraction; the terminal set associated with the tube is no longer constrained to lie within Z, which relaxes restrictions compared to deterministic MPC formulations; it can be shown that the closed-loop system is robustly exponentially stable with respect to the set Z. This tube-based MPC framework has been extended to handle linear time-varying systems [23] and nonlinear systems [24], further broadening its applicability and robustness in real-world control scenarios.

In the aforementioned tube-based MPC methods, both the feedback gain and the positively invariant set are typically computed offline. Consequently, the cross-sections of the state and input tubes remain fixed throughout the optimization, yielding a framework often referred to as rigid tube MPC, which maintains computational complexity comparable to standard MPC. However, when the system

uncertainty is state-dependent, the inability to adapt tube geometry online can lead to overly conservative performance. This limitation has driven the development of dynamic tube MPC strategies.

Dynamic tube MPC introduces tube parameters as decision variables within the optimization problem, allowing the tube's geometry, specifically its shape and size, to be updated online in response to system variations. Several representative approaches have emerged from this paradigm. In similarity-based tube MPC [25], the shape of the tube is predetermined offline, while its magnitude is adjusted during runtime. Elastic tube MPC [26] enhances this concept by treating the tube geometry as a vector-valued decision variable, thus enabling simultaneous optimization of shape and size. Furthermore, decoupled state-feedback tube MPC [27] parameterizes state predictions and control inputs using partial sequences, constructing tube as combinations of these subcomponents. This formulation facilitates finer control over tube geometry compared to elastic tube MPC, potentially yielding improved closed-loop performance.

Despite their benefits, dynamic tube MPC methods inevitably introduce additional decision variables and structural complexity into the optimization process. As such, their implementation requires a careful balance between computational burden and reduction in conservativeness to ensure practical applicability.

Tube-based MPC can be viewed as a feedback MPC strategy that incorporates constraint-tightening techniques. Its implementation relies on appropriately parameterizing the control input to shape bounded tubes around the nominal system trajectory. For linear systems, such parameterizations are relatively straightforward to construct. However, due to the inherent complexity of nonlinear dynamics, designing suitable tube structures in nonlinear systems is significantly more challenging.

As an alternative to tube-based approaches in nonlinear settings, researchers have proposed constraint-tightening methods within open-loop optimization [28]. For example, Ref. [29] employed Lipschitz constants to quantify the error between the actual and predicted states, using this information to contract the admissible state constraint set accordingly. In contrast, Ref. [30] avoided explicitly constructing error sets and instead imposed robustness constraints directly on the predicted states using Lipschitz bounds. Building on this framework, the authors developed a robust distributed MPC scheme, where predicted state trajectories are both monotonic and bounded. Although these constraint-tightening methods may yield more conservative performance than nonlinear tube-based MPC due to their open-loop nature, their optimization problems are simpler and more tractable. This computational advantage makes constraint-tightening-based nonlinear MPC methods particularly attractive for practical applications.

When both parametric uncertainty and additive disturbances are present, the design requirements for robust MPC become increasingly demanding. Min-max MPC addresses this by minimizing a worst-case cost function over all possible uncertainty realizations, differing from tube-based approaches by explicitly optimizing for the worst disturbance scenario. Initially proposed for Single-Input-Single-Output (SISO) systems [31], it was later extended to general linear [32] and Linear Parameter-varying (LPV) systems [33]. For nonlinear systems, open-loop min-max formulations [28] suffer from limited domains of attraction and computational burden. To mitigate conservativeness, feedback-based min-max MPC schemes incorporate state-dependent control laws [34], ensuring input-to-state stability [35]. Further improvements include frameworks that enforce explicit bounds on the closed-loop trajectories [36], and multistage MPC methods that approximate uncertainty via scenario trees, leveraging future measurements for enhanced robustness [37].

Despite its robustness benefits, the high computational complexity of min-max MPC limits its practical use. To address this, several methods have been developed to reduce computation. For example, Ref. [32] approximates worst-case scenarios using extreme disturbance realizations, effective only for low-dimensional systems. Offline strategies to approximate feedback min-max MPC solutions were proposed in Refs. [38] and [39]. In Ref. [40], convex reformulation via specific feedback design significantly improved efficiency. Hybrid schemes combining multistage MPC and tube-based MPC have also emerged [41], assigning large uncertainties to multistage optimization and smaller ones to tube methods. Furthermore, tailored algorithms have been developed for constrained nonlinear systems with network-induced delays and packet loss [42], actuator uncertainty [43], and to reduce communication costs via self-triggered control in general nonlinear systems [44].

Integrating disturbance observers (DOBs) with MPC enables simultaneous disturbance estimation and preemptive compensation, improving model accuracy and enhancing the robustness of MPC strategies. In Ref. [45], DOB-based compensation was applied to SISO affine nonlinear systems under MPC. Similarly, Ref. [46] proposed a DOB-augmented predictive torque control for induction motor systems, effectively mitigating parameter uncertainty and improving dynamic speed and torque response. Beyond direct compensation, estimated disturbance sequences can be embedded into the predictive model to reduce forecast errors. For example, Ref. [47] addressed slowly varying disturbances by generating predicted sequences from their rate of change, while Ref. [48] used statistical learning to model the relationship between disturbances, control inputs, and system states, enabling data-driven disturbance prediction.

1.4 Event-based MPC

In control systems, sampling, computation, and actuator updates are performed at fixed time intervals. This approach is known as time-triggered control. This methodology has been extensively studied over the past several decades. However, with the continued evolution of cyber-physical systems, communication networks have become central to information exchange among sensors, controllers, and actuators. Within this context, time-triggered control and communication strategies can significantly increase network load, potentially leading to congestion, delays, or packet losses that degrade system performance and, in some cases, induce instability.

To maintain satisfactory control performance while mitigating the risk of network congestion, it is essential to reduce communication overhead. Event-triggered strategies have proven to be an effective means of alleviating communication burden. An event is a specific, predefined condition that, when met, triggers a control-related action such as state sampling, control update computation, or data transmission. Within an event-triggered mechanism, the condition is typically defined as the moment when a calculated error function—which characterizes the deviation between the current system state and the state at the last update—exceeds a predefined threshold function. The core idea underlying event-triggered control and communication schemes is that information transmission and control input updates are executed only upon the occurrence of predefined events, while ensuring the desired control performance is maintained. Compared to time-triggered methods, event-triggered control and communication frameworks not only reduce the computational demand placed on the controller but also alleviate communication load across the network.

To effectively utilize limited computational and communication resources, data sampling, transmission actions, and control or estimation updates should be minimized while maintaining the desired control and estimation performance. This necessity has spurred substantial research into event-triggered mechanisms. An event-triggered mechanism typically comprises two key components.

- Error Function: This function characterizes the deviation between the current value and the value at the last triggering instant. Mathematically,

$$f(e(t)) = f(x(t) - x(t_m))$$

  where t_m denotes the previous triggering time and t is the current time.
- Threshold Function: This function quantitatively defines the magnitude criterion.

Based on the interplay between the error function and the threshold function, the expression for determining the next event-triggering instant can be formulated by

$$t_{m+1} = \inf\{t > t_m \mid f(e(t)) \geq \sigma(t)\}. \tag{1.3}$$

By appropriately designing the triggering conditions, data sampling and control actions are executed only when necessary to ensure system stability and performance, thereby significantly reducing the consumption of available computational and communication resources. From the expression governing the triggering instants, two key insights emerge: (i) Threshold Function: This function directly influences the frequency of event triggering. Specifically, higher thresholds result in fewer triggering events, thus lowering the overall rate of event occurrences. Existing studies have explored a wide range of threshold functions to accommodate different system models and problem formulations. (ii) Performance–Resource Trade-off: A fundamental trade-off exists between the desired control performance and resource efficiency. As resource consumption decreases, overall control performance tends to deteriorate to some extent. This is primarily due to the reduced volume of data available for controller design and implementation. Therefore, it is essential to strike a balance that minimizes resource usage while preserving acceptable system performance.

Due to the distinctive properties of event-based control, the implementation of nonperiodic MPC based on event-triggering mechanisms has attracted increasing attention. Presently, event-based MPC approaches primarily consist of event-triggered MPC and self-triggered MPC schemes. In event-triggered MPC, the triggering mechanism determines the sampling and communication instants, thereby reducing both communication and computational loads. Event-triggered MPC activates control updates by continuously monitoring the deviation between the actual state trajectory and the optimal predicted trajectory, making it a promising research direction within event-triggered control strategies. Notably, given that MPC algorithms typically require considerable computational resources, embedding an event-triggering mechanism within MPC not only substantially reduces communication frequency but also alleviates the computational burden. The key advantage of event-triggered MPC lies in its capacity to lower the frequency of state sampling and optimization execution. The concept of event-triggered MPC was initially explored in [49], and has since garnered increasing research interest. In general, the existing literature on event-triggered MPC can be categorized into the following three groups:

1. Event-triggered MPC Based on Absolute Threshold Mechanisms: An implicit event-triggered MPC algorithm for nonlinear continuous-time systems was first proposed in [49]. The feasibility and convergence of the algorithm are ensured

provided that the inter-trigger intervals are bounded above and below. The trade-off between triggering frequency and control performance was examined in [50]. In [51], a constant-threshold event-triggering condition was introduced within a dual-mode MPC framework, which significantly reduced communication and computational power consumption compared to conventional periodic time-triggered MPC. Subsequently, [52] proposed a novel event-triggering condition for dual-mode MPC, where the threshold is formulated as a function of the minimum allowable inter-event time interval. In [53], input-to-state stability was analyzed for systems employing event-triggered mechanisms under quasi-infinite horizon MPC and dual-mode MPC frameworks.

2. Event-triggered MPC Based on Relative Threshold Mechanisms: By employing integral sliding mode control in conjunction with a relative-threshold-based event-triggering condition, a robust event-triggered MPC scheme was proposed in [54], capable of compensating for model uncertainties. In [55], an event-triggered nonlinear MPC framework was developed for distributed cooperative control of a group of dynamically decoupled nonlinear systems. Each subsystem is equipped with its own event-triggering condition, which is determined by the error derived from comparing local information with that of neighboring subsystems.

3. Event-triggered MPC Based on Lyapunov Functions: An event-triggered nonlinear MPC scheme based on Lyapunov functions was introduced in [56], where the triggering condition is derived by ensuring the monotonic decrease of the Lyapunov function starting from each self-triggered instant. In [57], two event-triggering conditions based on a non-monotonic Lyapunov approach were investigated, considering continuous-time nonlinear systems subjected to additive disturbances. In the context of distributed control, [58] studied a distributed event-triggered MPC framework for linear time-invariant systems, where cost-based event-triggering conditions are designed to determine when updated control data should be transmitted to other controllers, reducing inter-subsystem communication. The study in [59] proposed an event-triggering condition for distributed MPC that relies solely on local subsystem information. Later, [60] introduced neighbor-informed event-triggering conditions into distributed MPC. These conditions are derived from stability analysis, ensuring a continuous decrease in the Lyapunov function over successive triggering instants.

In the context of aperiodic MPC strategies, self-triggered MPC determines the next sampling instant solely based on the measured state obtained at the current triggering instant, without the need for continuous or periodic monitoring of the system state. As a result, state measurements are updated only at triggering instants, which significantly reduces communication burden. A widely adopted approach for designing self-triggered MPC strategies involves exploiting certain properties of the cost function to derive the triggering condition. For example,

in [61], a self-triggered mechanism was developed based on the Lipschitz continuity of the cost function, along with guarantees of recursive feasibility and closed-loop stability. In [62], a sample-and-hold mechanism for control inputs was introduced between two consecutive triggering instants to further reduce communication demands. In [63], the researchers proposed a computationally efficient method that adaptively selects the sampling interval while ensuring a certain degree of performance sub-optimality. For systems with state constraints under disturbances, the original constraints can be appropriately tightened—essentially using the Gronwall–Bellman inequality—to ensure robust constraint satisfaction.

In addition, a class of self-triggered MPC methods introduces a new decision variable into the cost function, offering additional flexibility to balance communication cost and control performance. The basic idea is to relax the state penalty by adjusting this auxiliary variable when the controller is not triggered during the current interval. By solving a more complex optimization problem, the next triggering instant can be explicitly determined at the current triggering time. The reduction in the optimal cost can be interpreted as a reward for employing longer sampling intervals. Following this idea, [64] incorporated a term inversely proportional to the triggering interval into the cost function for an undisturbed linear system to model communication cost. Through optimization, a trade-off between communication load and control performance was effectively achieved. The work in [65] extended these results to linear systems subject to disturbances. To handle disturbances and enable output feedback, robust tube-based MPC and Luenberger observer-based approaches have been widely investigated. In [66], a disturbance-free nonlinear system was considered, and the standard cost function was modified by dividing the cost accumulated over the triggering interval by a constant greater than one. By treating the triggering interval as an additional decision variable, both the optimal control input and triggering interval can be determined simultaneously. For disturbed linear systems, [67] adopted the same strategy in combination with a tube-based MPC framework. In the case of linear systems with probabilistic constraints, [68] proposed a stochastic self-triggered MPC approach. For uncertain nonlinear systems, [44] developed a self-triggered control strategy based on min-max MPC and a similar relaxation mechanism.

Although most existing self-triggered MPC methods fall into the aforementioned two categories, there are a few notable exceptions. For instance, [69] considered continuous-time nonlinear systems and designed an MPC controller where the cost function over different time intervals and multiple prediction models are formulated in a discrete-time manner. Specifically, a zero-order hold mechanism is assumed to apply across all prediction intervals, and the holding time is treated as a function of the predicted system state. By solving an optimization problem, both the control input and the holding time can be simultaneously determined, resulting in a self-triggered controller. In [70], a relaxed dynamic programming

inequality was incorporated into the MPC framework to construct a self-triggered strategy. By combining this strategy with a tube-based MPC approach, the authors investigated disturbed distributed linear systems.

1.5 Book Outline

The objective of this book is to integrate event-triggered mechanisms into MPC frameworks, and to explore event-triggered MPC strategies for both continuous-time and discrete-time systems. Moreover, it aims to incorporate disturbance attenuation into event-triggered MPC for uncertain systems, and to investigate the extension of event-triggered MPC approaches to multi-agent systems. The major challenges and key research difficulties involved include:

1. Design of Triggering Mechanisms: Absolute event-triggered mechanisms require continuous monitoring of the system state and real-time evaluation of the triggering condition by sensors, which imposes stringent demands on sensor hardware. Designing appropriate sampling and triggering mechanisms to reduce monitoring and decision frequency is a critical research issue.
2. Zeno Behavior in Continuous-time Systems: In event-triggered control of continuous systems, Zeno behavior may arise, referring to an infinite number of triggering events occurring in finite time—an undesirable phenomenon in practical systems. In event-triggered MPC, the system is triggered when the deviation between the actual and predicted states exceeds a threshold, meaning that the choice of triggering threshold directly influences the occurrence of Zeno behavior. How to select an appropriate threshold based on system characteristics, and how to compute the minimum inter-event time accordingly, remains a key design challenge.
3. Handling Bounded Disturbances: When the system is subject to bounded disturbances, how to effectively handle them becomes a crucial problem. One must design constraint-tightening schemes to ensure that the actual system behavior respects the imposed constraints. For systems with partial knowledge of the disturbances, leveraging such information in the design of the MPC scheme is also a significant research focus.
4. Event-triggered MPC in multi-agent Systems: The application of MPC in multi-agent systems requires information exchange among neighboring agents. Determining efficient and minimal communication protocols, and integrating them effectively with event-triggered MPC strategies, presents a complex and open challenge.

In response to the aforementioned research objectives and key challenges, the structure of this book and the relationships among its subsequent chapters are outlined as follows.

Chapter 2 presents an in-depth study of event-triggered mechanisms for continuous-time systems. A predictive two-phase event-triggered MPC approach is proposed, aiming to reduce the number of sampling instances and lower the triggering frequency.

Building on this, Chapter 3 addresses the discontinuous nature of states and integer-valued sampling intervals in discrete-time systems. By fully exploiting state information at sampling instants, an event-triggered aperiodic sampling MPC method is developed.

Furthermore, considering the slowly varying nature of disturbances in certain scenarios, Chapter 4 introduces a composite event-triggered MPC method with disturbance compensation.

Finally, in Chapter 5, the focus is extended from a single system to multiple disturbed systems. Taking into account the communication and interaction requirements among multiple systems, a distributed MPC method based on periodic sampling and event-triggered communication is proposed.